Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. V (2010), No. 1, pp. 52-70

Coordinating Aerial Robots and Unattended Ground Sensors for Intelligent Surveillance Systems

E. Pignaton de Freitas, T. Heimfarth, R. Schmidt Allgayer, F. Rech Wagner, T. Larsson, C. E. Pereira, A. Morado Ferreira

Edison Pignaton de Freitas and Tony Larsson
Halmstad University
School of Information Science, Computer and Electrical Engineering
Halmstad, Sweden
E-mail: {edison.pignaton, tony.larsson}@hh.se

Tales Heimfarth and Flávio Rech Wagner
Federal University of Rio Grande do Sul
Institute of Informatics
Porto Alegre, Brazil
E-mail: {theimfarth, flavio}@inf.ufrgs.br

Rodrigo Schmidt Allgayer and Carlos Eduardo Pereira
Federal University of Rio Grande do Sul
Electrical Engineering Department
Porto Alegre, Brazil
E-mail: {allgayer, cpereira}@ece.ufrgs.br

Armando Morado Ferreira
Military Institute of Engineering
Defense Engineering Graduate Program
Rio de Janeiro, Brazil
E-mail: armando@ime.eb.br

Abstract:
Sensor networks are being used to implement different types of sophisticated emerging applications, such as those aimed at supporting ambient intelligence and surveillance systems. This usage is enhanced by employing sensors with different characteristics in terms of sensing, computing and mobility capabilities, working cooperatively in the network. However, the design and deployment of these heterogeneous systems present several issues that have to be handled in order to meet the user expectations. The main problems are related to the nodes‘ interoperability and the overall resource allocation, both inter and intra nodes. The first problem requires a common platform that abstracts the nodes’ heterogeneity and provides a smooth communication, while the second is handled by cooperation mechanisms supported by the platform. Moreover, as the nodes are supposed to be heterogeneous, a customizable platform is required to support both resource rich and poorer nodes. This paper analyses surveillance systems based on a heterogeneous sensor network, which is composed by low-end ground sensor nodes and autonomous aerial robots, i.e. Unmanned Aerial Vehicles (UAVs), carrying different kinds of sensors. The approach proposed in this work tackles the two above mentioned problems by using a customizable hardware platform and a middleware to support interoperability. Experimental results are also provided.
Keywords: Sensor Networks, Unmanned Vehicles Systems, Wireless Communication, Heterogeneous Platforms

1 Introduction

New applications based on the interaction between autonomous robots and static sensor nodes are emerging. Part of the growing interest in using this interaction comes from the potential benefits it can provide, such as distributed processing and data gathering enrichment. Moreover, the deployment of such systems is becoming a tangible reality due to advances in small and efficient processors, sensor devices, and wireless networking. The integration of static and mobile sensor nodes enhances capabilities of the overall sensor network. It enables new applications in which fixed sensor nodes provide an active response capability, while unmanned mobile nodes acquire a spatial vision of the region, allowing monitoring of larger areas. Military and civilian applications, such as borderline patrol, search and rescue, area surveillance, communications relaying, and mapping of hostile territory, can take advantage of this kind of enlarged sensor networks. Since these tasks may be repetitive, tedious, and dangerous, they are ideal for autonomous unmanned vehicles [1].

Indeed, applications of sensor network can greatly profit of using different kinds of mobile and sophisticated sensors in addition to static ones, which typically are simpler and more resource constrained. The cooperation between these different types of sensors provides advanced functionalities that were not feasible before [2] and thus also opens a vast range of new application scenarios.

Wireless sensor networks are usually developed to perform specific applications. Sensor nodes have usually a small footprint including only the resources that are necessary to meet specific application requirements. However, reconfigurable and customizable architectures are important to make both sensor nodes and sensor networks more flexible. Depending on the application, sensor nodes vary from resource-constrained to high performance computing nodes, resulting in a heterogeneous network composed by a variety of sensor node types.

The main issues in developing heterogeneous sensor networks are: (i) support for cooperation among heterogeneous nodes; and (ii) customization of sensor nodes. The former is related to concerns such as message exchange synchronization, QoS requirements management, task (re-) allocation, and network adaptation. The later is related to the diversity of node platforms, which may be built upon very distinct hardware components controlled by very different pieces of software.

Considering (i), the use of a middleware is a suitable approach to address the mentioned concerns, since it can integrate the technologies used in different nodes by means of a common communication interface and cooperation support. Regarding (ii), customizable architectures can be very useful to build platforms for different sensor network nodes, from the very simple to the more sophisticated ones. This kind of architecture can provide a common base capability for all nodes. However, for nodes that need more advanced capabilities, the required resources can be incorporated. Hence, even though all nodes should have the same base capability, some of them could be equipped with additional resources, thus making the sensor network more powerful due to this allowed heterogeneity.

This paper presents a flexible and adaptable platform infrastructure intended to support heterogeneous sensor network applications composed by static sensor nodes on the ground and mobile sensors carried by autonomous aerial robotic platforms, such as autonomous UAVs. It is based on the proposal of (i) a flexible middleware [3] and, (ii) on a customizable hardware architecture aimed for sensor nodes, called FemtoNode [4]. The key idea is to use this customizable platform to deploy different kinds of sensor nodes, from very tiny and resource constrained up to more sophisticated ones. Both types of nodes run a common middleware in order to provide the desired interoperability that will allow the cooperation among different sensor nodes. Nodes may be built upon the FemtoNode architecture and alternatively upon nodes with other hardware platforms, such as SunSPOT [5] or similar ones. Moreover, the middleware provides services to allow autonomous decision making by the nodes, in order to perform reflection about the runtime conditions and adapt the system according to the current needs. These features allow the mobile nodes to take decisions about their movements and all nodes, i.e. also the fixed nodes, to take decisions about network parameters, such as QoS and resource usage. In addition, this paper presents a bio-inspired coordination mechanism that uses artificial pheromones used to provide information about the position of the mobile sensor nodes, facilitating the cooperation among these nodes and the others that make part of the network.

The remaining of this paper is organized as follows: Section 2 presents related work in the area. In Section 3, the application scenario is highlighted, characterizing the network heterogeneity. Section 4 presents an overview of the proposed middleware, while Section 5 presents details about the key issues addressed by the middleware. The pheromone-based coordination among static and mobile sensors is the focus of Section 6. In Section 7, the FemtoNode customizable hardware architecture is described. Section 8 presents a case study while Section 9 provides results for some of the presented features of both middleware and customizable hardware. Finally, Section 10 draws concluding remarks and gives directions for future work.

2 Related Works

AWARE [6] is a middleware whose goal is to provide integration of the information gathered by different types of sensors, including low-end sensor nodes in a wireless sensor network and mobile robots equipped with more sophisticated sensors. Our proposal not only addresses heterogeneous sensors and their coordination, but also concerns like QoS and runtime reflection in order to cope with changes in the environment and in the network, which is missing in [6]. Moreover, in our approach, the autonomous decisions taken by the mobile nodes characterize a higher intelligence degree of our mobile robots if compared with the proposal presented in [6].

Batalin et al. [7] propose an approach that uses a sensor network to guide a mobile robot with limited sensor capabilities. This work presents aspects of cooperation among different sensors, but it does not address the same problem as proposed in our work. In that approach, the mobile node has limited sensor capabilities and only gathers data from the sensor network to guide the robot's movement. In our work, mobile nodes are equipped with sophisticated sensors that can provide data, which may be merged with data that come from the other nodes in order to achieve more refined decisions to guide the movement of the mobile nodes.

Schmidt et al. [8] present an approach that uses artificial intelligence techniques to configure an underlying middleware. Their approach uses the concepts of missions and goals to plan the allocation of tasks in a network of homogeneous nodes. The handling of heterogeneous nodes is one of the differences between our work and their approach. Additionally, in that work, the intelligence is outside the middleware and influences it by just sending commands or adjusting its parameters. In our approach, the decision making support is an integral part of the middleware, spreading intelligence over the network.

Jin et al. [9] provide a very consistent proposal to handle the problem of balance between target search and response by a team of UAVs. The work evaluates the tradeoff between search and response within the framework, presenting a predictive algorithm that provides a good balance between these tasks. The first difference between our approach and this work is that we handle only the alarm response, abstracting the concern about the UAVs movement planning to perform the search for new targets. This difference is due to the peculiarity of the distinct missions addressed in the current paper and in [9]. We focus on area surveillance, while they focus on target acquisition. In our case, the whole area must be covered, which may not be true in the target acquisition they address. Another difference is that we use the UAVs in coordination with ground sensor nodes. Besides, the assumption of a centralized information base considered in their work is not used in our proposal. Their initial centralized off-line task assignment is another premise that is not valid in our work.

In [10], an approach using digital pheromones to control a swarm of UAVs is presented. The method proposed by the authors uses digital pheromones to bias the movements of individual units within a swarm toward particular areas of interest that are attractive, from the point of view of the mission that the swarm is performing, and away from areas that are dangerous or just unattractive. In the large sense, the pheromone-based strategy used in our work has a similar goal, driving the UAVs to areas of interest. However, differently from their approach, we use the pheromone traces to localize the UAVs when an alarm is issued by a ground sensor node informing an event of interest and then drive the UAVs to the location where the event happened.

3 Application Scenario Overview and Network Heterogeneity

In the following, heterogeneity means that nodes in the network may have different sensing capabilities, computation power, and communication abilities. Additionally, it means that they run on different hardware and operating systems. Therefore, such sensor networks are made up of low- and high-end nodes. Moreover, sensor nodes may have fixed positions or be able to move, being carried by mobile aerial robots (UAV platforms), which can also vary from very small, as in [11], up to huge aircraft platforms, like GlobalHawk [12].

Low-end sensor nodes are those with constrained capabilities, for instance piezoelectric resistive tilt sensors, with limited processing support and communication resource capabilities. High-end sensor nodes include powerful devices like radar, high definition visible light cameras, or infrared sensors, which are supported by moderate to rich computing and communication resources.

Mobility, as mentioned, is another important characteristic related to the heterogeneity addressed in this work and requires special attention. Sensor nodes can be statically placed on the ground or can move on the ground or fly at some altitude over the target area in which the observed phenomenon is occurring. Figure 1 graphically represents the idea of the three heterogeneity dimensions considered in this work, in which each axis represents one of the considered characteristics.

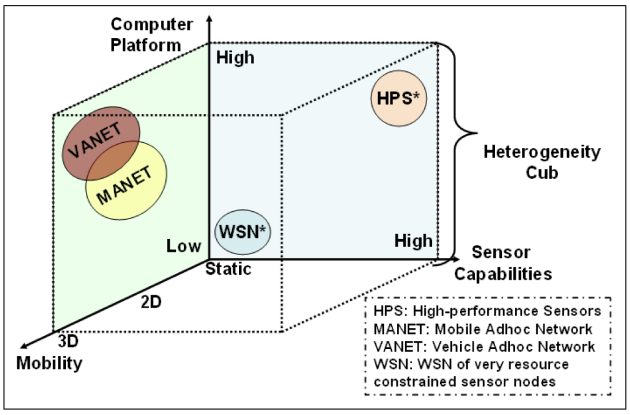
Computer Platform
High
HPS*
Heterogeneity Cub
VANET
MANET
WSN*
Low
High
Static
Sensor Capabilities
2D
3D
Mobility
HPS: High-performance Sensors
MANET: Mobile Adhoc Network
VANET: Vehicle Adhoc Network
WSN: WSN of very resource constrained sensor nodes

Figure 1: Heterogeneity Dimensions.

The reason for heterogeneity in the sensor nodes is to support a large range of applications that deal with very dynamic and changing scenarios, which require different types of sensor capabilities. Moreover, these different scenarios require also adaptations in the network, in terms of choosing suitable sensors for the tasks at hand as well as feasible QoS parameters, among others.

In order to illustrate the above idea, suppose that a network has the mission of providing a certain kind of information during a given period of time. The set of sensors selected in the beginning of a mission may not be the most suitable one during the execution of the whole mission. The network must be thus able to choose a better alternative, among the set of all available options, in order to accomplish the mission. For example, an area surveillance system may receive the mission to observe if certain types of vehicles that are not allowed to pass through the surveyed area make any such violation and report if that is the case. To perform this in an efficient way ground sensors are set to alarm in the presence of unauthorized vehicles. Suddenly, an alarm is triggered by one of these sensors. Then, in order to verify the occurrence, UAVs equipped with visible-light cameras are commanded to fly over the area where the ground sensor has issued the alarm. Due to a sudden change in the weather (e.g. the area becomes foggy or cloudy), visible-light cameras become useless. However, the mission must still be accomplished. Therefore, nodes must be coordinated to accomplish the mission by sending UAVs equipped with other kinds of sensors that can provide the required information in bad weather conditions, such as infrared cameras.

4 Middleware Support Overview

One of the key ideas of the proposed work is to drive the use of sophisticated sensors carried by UAVs using data from low-end nodes. UAVs' equipped with intelligent behavior uses such data to decide the trajectory direction during the system runtime. Consequently, the proposed middleware must be able to provide cooperation among the different sensors and still fit in both the low-end and rich sensor nodes.

The middleware must thus be lightweight and provide enough customization in order to address the needs of both types of sensor nodes. These goals are achieved by using aspect- and component-oriented techniques, as in [13] and [14], and also a mobile multi-agent approach, as discussed in [3]. Figure 2 depicts an overview of the middleware layers, whose description is provided in the following.

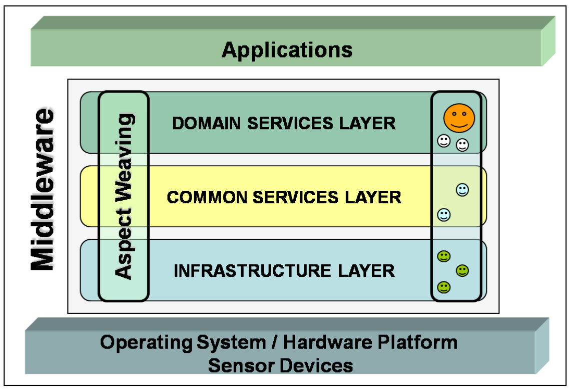

Figure 2: Overview of the Middleware Layers.

The bottom layer is called *Infrastructure Layer*. It is responsible for the interaction with the underlying operating system and also for the management of node resources, such as available communication and sensing capabilities, remaining energy, etc. This layer is also responsible for coordination of necessary resource sharing.

The intermediate layer is called *Common Services Layer*. It provides services that are common to different types of applications, such as QoS negotiation, quality of data assurance, and data compression. Other concerns also handled within this layer are: deadline expiration alarms; timeouts for data transmissions; number of retries and delivery failure announcements; resource reservation negotiation among applications (based on priorities established by missions and operation conditions); bindings; and synchronous/asynchronous concurrent requests.

The top layer is called *Domain-Services Layer*, whose goal is to support domain specific needs, including data fusion support and specific data semantic support, in order to allow the production of application-related information from raw data processing. Fuzzy classifiers, special types of mathematical filters (e.g. Kalman Filter), and functions that can be reused among different applications in the same domain are found in this layer.

"Smile faces" in Figure 2 represent autonomous agents that can provide specific services in a certain node at a given moment during system runtime. The *Domain-Services Layer* hosts a special agent (called *planning-agent*), which performs a special task related to the reasoning about the missions that a node is responsible for performing.

Concerns that affect elements in more than one middleware layer, such as security and real-time requirements control, are represented as cross-layer features. These crosscutting concerns are addressed by the aspect-oriented approach presented in [13].

A. Planning-agent Model Overview

As an important element in the proposed approach, the model of the planning-agent is briefly presented.

The model used for the planning-agent is a cognitive one, based on the model of mental attitudes, known as BDI model (Belief-Desire-Intention), presented in [15]. The motivation is that the BDI model appears to suite well to the surveillance problem addressed, as some decisions that need to be taken by the planning-agent require cognitive skills to evaluate if certain actions are adequate to achieve a desired result. These decisions are based on knowledge about conditions that may interfere on the performance of those actions. In the current problem formulation, it is desired to obtain information by means of sensing activities, which are the goals of a sensing mission. This knowledge is the ”belief” that the node has about the relevant conditions, and the intentions are translated into the actions required to retrieve the desired information.

As an example, when a UAV receives an alarm, it will consult its beliefs in order to decide if it is able to respond to that alarm. If so, it will take the responsibility for the respective alarm and include the incoming information from the alarm into its beliefs, as well as include the accomplishment of the alarm handling into its desires. Based on that, the UAV will then intend to fly to the place where the alarm was issued in order to execute the respective sensing over the target.

B. Middleware Services Usage Example

Taking the application scenario given as example in Section 3 and the middleware layers described above, this subsection provides some examples of the utilization of middleware services and of the interaction between nodes.

Assuming the described example, in the case that alarms are issued by several ground sensor nodes, the *Domain Services Layer* provides data aggregation, for fusion of data from many sensors in an area, which can provide richer information, such as the direction of a crossing vehicle, or handle problems such as alarm duplicity. The delivery of such information has an expiration time, since after a given time threshold, it is probable that the vehicle may have changed its trajectory, making the previous collected data useless. Therefore, the *Common Services Layer* associates QoS with the delivery of messages as well as a guarantee mechanism to assure that an alarm has being correctly delivered and in time to a UAV. The *Infrastructure Layer* uses these QoS parameters to manage resources utilization.

In the UAVs, the middleware makes the complementary task, providing data fusion of images with matching alarm messages received from low-end nodes, i.e. fusion of position information included in the alarm messages with images that are being taken by the UAVs, in the *Domain Services Layer*. This is the case, for example, when more than one alarm has been issued in a given location. The data fusion helps in distinguishing the source of the alarms. QoS verification of incoming messages is performed by the *Common Services Layer*, which checks if either stored data can be used by the application or a request for fresh data must be sent. This is the case, for example, if the ground sensor network experiences problems and takes a too long time to deliver an alarm message to a UAV. It may happen that, when this alarm comes to the UAV, the object that triggered the alarm is not anymore at the location where it was detected. At this point, a request for confirmation can be sent by the UAV to another UAV in the region around the location of that alarm, requiring certain levels of QoS.

Depending on the results from data processing, as described above, the UAVs autonomously decide their own placements over the surveillance area, by means of the reasoning mechanisms of their respective *planning-agents*. Additionally, other factors influence this decision, e.g. specific needs of the current situation and also sensible data that must be sent to the base station or to another UAV. In this case, data segregation and network utilization control play a crucial role in the efficiency of the system. Important data must be prioritized. Thus, network resources are allocated to data transmission according to the established priorities. Possibly, the specific needs faced by the system in a given situation may require adjustments in the resource usage policy, which influences the services provided by the *Common Services Layer*. For instance, a change to a ”Mandatory Priority” may take place in order to assure that data about a detected object or phenomena arrive within a given deadline at the base station via relayed communication through other UAVs.

In order to consider all the mentioned factors, necessary support for reflection about the network conditions and mission needs must be available. This support is provided by the *planning-agent* installed in the *Domain Services Layer*, which will, for instance, analyze the current conditions and requirements and, taking into account the information provided by other sensors, decide the best placement of the UAV in order to meet the system needs, thus selecting the next steps to move.

5 Addressing Key Issues of Sensor Networks with the Proposed Middleware

The proposed middleware uses the publish-subscribe paradigm and is inspired by the *Data Distribution Service for Real-time Systems* (DSS) specification, from OMG [16]. Some nodes publish their capabilities and the offered data, while others subscribe to data in which they are interested.

Although largely being inspired by the OMG DSS standard, the proposed middleware does not follow the whole specification. As it is intended to fit in both low-end nodes (based on simple and constrained platforms) and more sophisticated ones (carried by the UAVs), it must not only be lightweight but also provide capabilities for customizations to cope with the needs of the different sensor nodes. Consequently, the middleware uses a minimalist approach, being kept as simple as possible in each node. Required features are included by adding components or weaving aspects to handle real-time and other non-functional crosscutting requirements in the minimal middleware. Using aspects to tune the middleware is out of the scope of this paper, and for more information readers are referred to [13] and [17].

The following subsections show how the proposed middleware addresses some of the main platform needs in heterogeneous sensor networks, enabling the intelligent behavior of the mobile nodes and providing the required data with real-time guarantees.

A. Flexibility

The middleware provides full communication control, i.e. it does not use underlying mechanisms available in the nodes' network layer. Instead, it provides its own communication control. This means that all parameters related to communication are controlled by the middleware, which uses only basic connectionless communication services offered by the nodes' network layer. The middleware handles parameters such as number of retries, message priority, memory utilization for buffering, and timing. This provides more flexibility, with direct impact in the reduction of message delivery latency.

B. Network Reflection

Reflection over the state of the network is a feature that enables the sensor nodes to take decisions regarding their participation in accomplishing a specific mission or sub-mission. In the mobile sensor nodes, this feature is responsible for the decision regarding the movement of the robot platform that carries the sensor, in order to take it to a place or the area related to a mission. The reflection considering the network conditions is performed inside the middleware by the planning-agent, which schedules the activities that should take place in order to accomplish a given mission.

C. Dynamicity

When a node gets into the network, its services are announced using the publish-subscribe paradigm. Then, all interested nodes can subscribe for those services. This eliminates the need for a dedicated server node that centralizes all available services in the network. Additionally, this approach reduces latency in acquiring data because there is no intermediary node between the data producer and consumer.

D. Minimum Message Exchange

The publish-subscribe paradigm by itself already reduces the number of exchanged messages due to the elimination of intermediate nodes (such as brokers). However, bandwidth for control messages is still required. This bandwidth need can be further reduced with the use of smart techniques such as QoS contracts and attaching freshness timestamps to data, thus avoiding unnecessary request for data re-send. Another important characteristic for minimizing bandwidth requirements is the selection of routing strategies that optimize the use of the communication channel, finding the best path to be followed to arrive at the destination node of a message. An example of such strategy is a routing mechanism based on pheromone traces left by the mobile nodes, which will be explained in the following section and highlighted when presenting the experimental results.

E. Multicast Communication

The middleware uses multicast communication to reach selected destination nodes. This type of communication positively influences the latency and throughput, as data is sent at the same time to several nodes without unnecessary broadcast and delays, which would occur in unicast communication. A negative-acknowledgement (NACK) strategy (controlled by timeout expiration) is adopted in order to reduce acknowledgement messages in the network. However, very sensible data may require a positive acknowledgement to assure their delivery. Hence, positive acknowledgement is also made available as a middleware service and can be used when required.

F. Network Resources Usage Control

In order to improve the overall system performance, the control of the use of communication media and of transmission buffers is crucial. The middleware performs this task by taking into account two factors: (i) the priority associated to each application; and (ii) the resource sharing policy adopted in the system. There are three available resource sharing policies: (a) **Fair Sharing:** priorities are not considered and thus all applications have the same right to use the resources in a round-robin scheme, which is organized in an incoming FIFO queue; (b) **Soft Priority Sorted:** the priorities are taken in account, but in a relaxed way. If a higher priority application needs a resource already used by a lower priority one, it must wait until the resource is released. Due to its higher priority, it will get access to the resource before other applications, which may be waiting for the resource; (c) **Mandatory Priority:** higher priority applications can preempt lower priority ones in order to access the desired resources. Priority inversion issues are handled by a priority inheritance mechanism.

G. QoS Control

QoS control is performed through a contract between the provider and the requester of data. When a node publishes a data service, it informs about the QoS (level) offered. Nodes interested in the published data service should accept the offered QoS and subscribe to the service. However, if a node is interested in the data but does not agree with the offered QoS, it has two alternatives: (a) if the application that is requiring the data has a priority lower than any other one using the same service, the requester node looks for another data provider; (b) if its priority is higher than all other applications, the requester node negotiates with the data provider node, in order to obtain the desired QoS. This renegotiation occurs in spite of undesired consequences that may affect other lower priority applications, which need to look for another data provider if the QoS could not be accepted anymore.

H. Use of Cached Values

The use of cache in both data providers and requesters may avoid unnecessary data communication. When the measurement device gathers a new value, the data provider publishes the new value, thus updating its subscribers. If the data size is large, requiring many packets to be transmitted, a differential value can be sent instead of the whole data value in order to reduce packets transmission. This option is arranged in advance, at the time when the nodes are negotiating the QoS contract.

I. Data Segregation

There are two kinds of data exchanged among nodes in the network: control data and application data. Control data is small and can usually not experience latency or unexpected delays to achieve their destination. Thus, control data are separated from application data by receiving higher priority to be forwarded. Moreover, data from different types of applications are handled according to their type and semantics, e.g. the communication of a video stream is handled differently from the communication of character strings. Data from a specific application can be handled with higher or lower priority, depending on its semantics.

J. Synchronous and Asynchronous Calls

The middleware is intended to support both synchronous and asynchronous calls. Synchronous calls are time bounded in order to avoid unpredictable waiting periods. The waiting time and number of retries are configurable and negotiated during the QoS negotiation. Asynchronous calls are also provided in order to support the handling of external unpredictable event.

6 Pheromone-based Coordination Strategy

The coordination strategy used in this work to make mobile sensor nodes cooperate with static sensor nodes is based on pheromone traces handed over by the mobile sensors to the static ones. Artificial pheromones are usually applied to distributed coordination by means of stigmergy, the indirect communication using environment cues [17]. A pheromone trail is deposited in the environment when the entities are moving. The pheromone provides information to other entities when they pass over it. Artificial pheromone also looses its strength along the time, modeling the evaporation of the real pheromones. In the UAV research field, pheromones are used to guide the movement of UAV swarms, for instance in surveillance and patrolling applications [18] [19].

Differently from other existing approaches, in our work pheromones are used to guide the selection and assignment of a suitable UAV to handle an alarm issued by a ground sensor node. When an alarm is issued by the detection of a target, the network is responsible for selecting an appropriate UAV to respond to the alarm. This is performed by routing a given alarm to the UAV that has the strongest pheromone trace over the area. Having this information, the UAVs will base their movement decisions in a way to respond to the received alarms. This strategy is called here heuristic-P.

Following the above outlined principles, the UAVs that are not engaged in the handling of any target will leave pheromone traces over the area which they cross. This pheromone trace is represented by a piece of information that is taken by the ground sensor nodes that are deployed in the area through which the UAVs have passed. When a target is detected by a ground sensor node, an alarm is issued, as already mentioned. The decision about which UAV that will handle the potential target indicated by the issued alarm will be taken by the ground sensor nodes, by routing the alarm in the direction that points to the UAV which has the strongest pheromone trace over that area of the network. This process just considers the pheromone trace handed over by the UAVs to ground sensor nodes. This means that the only parameter taken into account is the time interval since a UAV passed by that specific location. Heuristic-P is inspired in [20], which presents a pheromone-based strategy to migrate services in a sensor network, in which the pheromone concentration determines the places where the services are required. In heuristic-P, instead of services, alarms are moved through the network following the pheromone concentration. Figure 3 presents a scenario that illustrates the strategy. A ground sensor node in the left border of the area detects a target. Then it issues an alarm, which is received by its neighbors. However, only those which have pheromone information about a UAV stronger than that of the alarm issuer will forward the alarm. This way, the alarm will follow a path to the closest UAV, which is represented in the figure by the shaded sensors, until the alarm delivery.

Figure 4 illustrates the choice of the strongest pheromone trace to be followed by an issued alarm. It is possible to observe that the alarm follows the strongest trace, which corresponds to UAV-A, until its delivery to this UAV. The arrows illustrated besides each sensor node represent how strong the pheromone

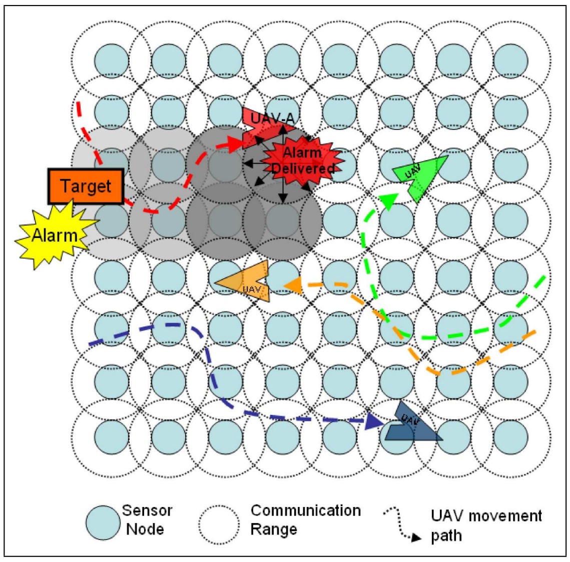

Figure 3: Illustrative scenario for the pheromone strategy.

of each UAV is. As it is possible to see, the pheromone level of UAV-A is increasing to the left, while the pheromone level of UAV-B is increasing to the right.

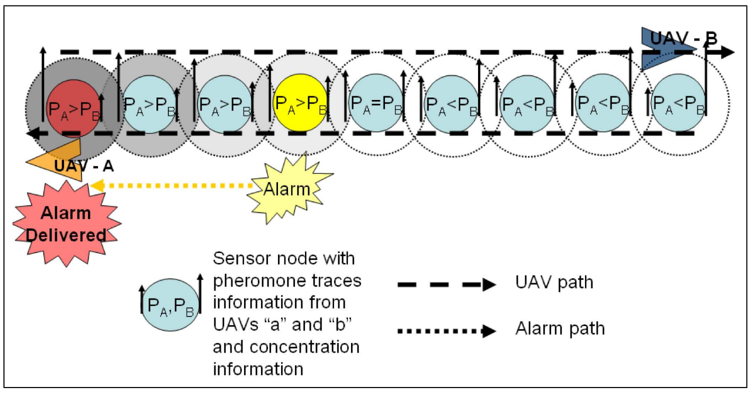

Figure 4: Choice of a UAV based on the pheromone strategy.

When an alarm reaches the UAV indicated by the strongest pheromone trace, if this UAV is not engaged in the handling of another alarm it sends a confirmation message to the node that had delivered the alarm. If the suggested UAV is already engaged in another alarm, the current alarm follows the second strongest pheromone trace to find another UAV to engage.

When an idle UAV detects a new target, it takes the responsibility for handling it. In case that the UAV is already busy with another alarm response mission, it relays the incoming alarm that will be routed to another UAV, according to the pheromone-based heuristic-P strategy explained above.

In order to increase the robustness of the proposal, in case an alarm is issued by a node that has no pheromone trace, a direction is randomly chosen and the alarm is sent in that direction until it finds a pheromone trace. When the trace is found, it follows the trace as explained above. This situation is more likely to occur in the initialization of the system, especially in cases in which the number of UAVs deployed in the system is very low with regard to the area under surveillance.

When a UAV receives an alarm and is not able to perform the task, it may send the alarm back to the network, which will try to find another UAV following the traces, or hand it over directly to another UAV. This situation may occur when the type of the sensor that the UAV carries is not appropriate to handle the event that was detected.

7 FemtoNode - Wireless Sensor Architecture

The architecture of a sensor node aims at efficiently supporting specific application needs. It requires a dedicated processing module, including a wireless communication interface, which meets both energy and performance requirements, as well as respects footprint constraints. The fact that application requirements as well as environment and other operational conditions may change during system run time imposes a major challenge [21]. In this context, the use of reconfigurable hardware [22] appears as an interesting alternative. Therefore, a customizable sensor node called FemtoNode is proposed. It contains a customizable ASIC and a wireless communication interface, which are configured according to application requirements.

The nodes use the RT-FemtoJava processor [23], a stack-based microcontroller that natively executes Java byte-codes. It implements an execution engine for Java in hardware, through a stack machine that is compatible with the specification of Java Virtual Machine. The customized application code is generated by the Sashimi design environment [24]. The code also includes a VHDL description of the processor core and ROM (programs) and RAM (variables) memories. The Sashimi environment has been extended to incorporate an API that supports concurrent tasks, implementing the RTSJ standard [25].

As RT-FemtoJava is customizable, its code can be optimized according to the application requirements, reducing the occupied hardware area and also the energy consumption and dissipation. The customizable hardware architecture of the FemtoNode allows the use of the sensor node as either a low- or high- end node. If the application requires higher performance resources to handle more complex data, such as image processing, additional resources can be included in the FemtoNode implementation. However, if the application is aimed at processing simple data, such as those from presence sensors, a reduced set of resources is used in the processor. This feature is important for the sensor node, because energy consumption is a great concern in wireless sensor networks, due to the nodes' limited energy resource. Besides, reducing the unused resources during its synthesis the sensor node architecture allows its implementation in reconfigurable circuits with fewer available logical units, which is a feature that provides a larger application portability between different reconfigurable architectures with fewer available resources.

In the current implementation, the FemtoNode includes a wireless transceiver of Texas Instruments CC2420, which utilizes the IEEE802.15.4 standard communication protocol targeted to wireless sensor network applications with a low data rate. A module adapter described in VHDL implements the interface with the wireless transceiver. The module uses data and address buses to communicate with the processor, performing the exchange of data and allowing the transceiver parameters configuration.

As the data transfer rate from the wireless transceiver is low, compared to the processor frequency, the wireless communication module implements a buffer to store data, preventing delays while providing the necessary data to the processor. The module uses an interrupt system to inform the processor when a reception occurred.

To facilitate the use of the wireless communication module by the application developers, a communication API has been developed. The Wireless-API abstracts details of the communication media between the sensor nodes, offering a simplified form for the configuration of the data transfer module.

The Wireless-API is used by the middleware communication services in the *Common Services Layer* and also by resource management services in the *Infrastructure Layer*, in order to implement end-to-end communication with the desired QoS and reliability control.

8 Case Study Description

In order to illustrate the use of the proposed platform infrastructure, including the customizable FemtoNode and the adaptable middleware, an area surveillance application is studied. In this application, low-end sensors nodes are scattered on the ground along a borderline. In case an unauthorized vehicle crosses the borderline limit, the sensors issue an alarm which will trigger the use of Unmanned Aerial Vehicles (UAVs), which are equipped with more sophisticated sensors, such as radars or visible light cameras, in order to perform the recognition of the vehicle. Figure 5 presents this scenario.

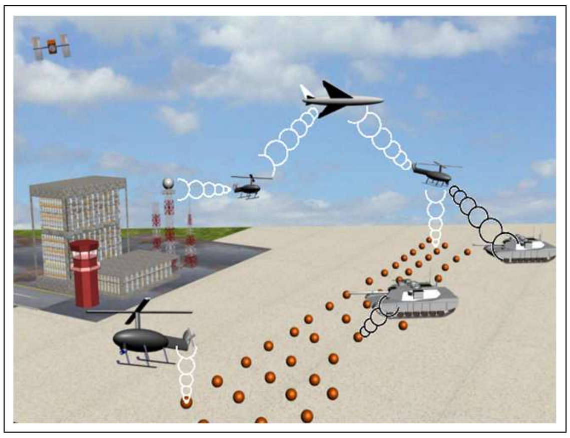

Figure 5: Area Surveillance Application Scenario.

Sensor nodes with two different architectures compose the described surveillance system, one based on the FemtoNode and another one on the SunSpot. Each of them includes all necessary resources to meet the requirements of their utilization. Thus, based on the application specifications, a customization of the FemtoNode architecture was implemented. The UAVs' architecture is a FemtoNode with a large set of resources, capable of processing a large amount of data. On the other hand, both FemtoNodes and SunSpots were used to populate the sensor network on the ground. The FemtoNode architecture used in the ground sensor nodes is simpler and more constrained in terms of available resources, as the SunSpot one, and hence only capable of processing simple data, like those produced by piezoelectric sensors or accelerometers, which inform only if an object over a certain weight threshold passed over its sensing area.

According to the coordination strategy based on pheromones presented in Section 6, the sensor nodes on the ground route the alarms according to the pheromone trace left by the UAVs, choosing the strongest trace to follow. When the alarm achieves a node close to the UAV, the alarm is delivered. This mechanism addresses several problems related to the communication between nodes, such as controlled delay, delivery assurance, and alarm duplicity handling.

9 Results

Several simulations of the scenario described in Section 8 have been performed. They do not cover all aspects mentioned in the above description, but focus on the behavior of the system using the coordination strategy described in Section 6. These simulations were conducted using ShoX [26], a powerful wireless network simulator implemented in Java.

Additional results related to the customizable FemtoNode working in a heterogeneous network together with SunSpot nodes are also presented. These results were gathered from the deployment of a laboratory-size testbed demonstrator, which shows the applicability of the platform.

The simulations provided results in an ”ideal-like” operation condition environment for the proposed technique, while the demonstrator was built in order to assess if these results provided by the simulations are possible to be achieved in a real deployment.

For more results related to the mechanisms that support the other features mentioned in Section 5, interested readers are referred to [27], [28], and [29].

A. Simulation Results

The metrics evaluated in the simulations were: 1) the mean response time to the alarms generated in the system, and 2) the number of alarms lost, due to communication failures.

The simulation setup was the following: The surveillance area has dimensions 10 Km x 10 Km, in which 20,000 ground sensor nodes are randomly deployed with independent uniform probability (homogeneous Poisson point process in two dimensions, which generates a geometrical random graph). This distribution gives more than 70% probability that the nodes in the network will form a connected graph [30], for a communication range of 500 meters. Six UAVs of three different types, equally distributed, patrol the area, having a communication range of 1.5 Km and flying at speeds from 100 Km/h up to 120 Km/h. Three different runs were simulated, with one, three, and five targets respectively. The targets can further be of five different types, randomly chosen, with speeds from 50 Km/h up to 80 Km/h.

Figure 6 presents the simulation results in terms of the mean time required to respond to the alarms. Both raw data from each run (total of 20 runs for each number of targets) and the average value (lines with squared dots) are plotted in the figure. It is possible to observe that, in the worst case, the mean time to find a UAV that is idle to engage in the handling of an alarm is around 4 seconds, in the scenario with the maximum number of targets. On the other hand, in the best case, when there is just one target, the time needed to find a UAV is in average less than 1 second. An explanation for this behavior is that it is more probable to find an idle UAV when the number of targets is smaller. This may happen because, when there are more targets, an alarm message may follow a pheromone trace of a UAV that has just engaged in handling a target announced by another alarm, so the alarm must be retransmitted to the network and follow another trace. However, the solution does scale, as the increase in the mean time to find an idle UAV is linear with the increase in the number of targets, as can be concluded by taking the average values for all runs for each number of targets.

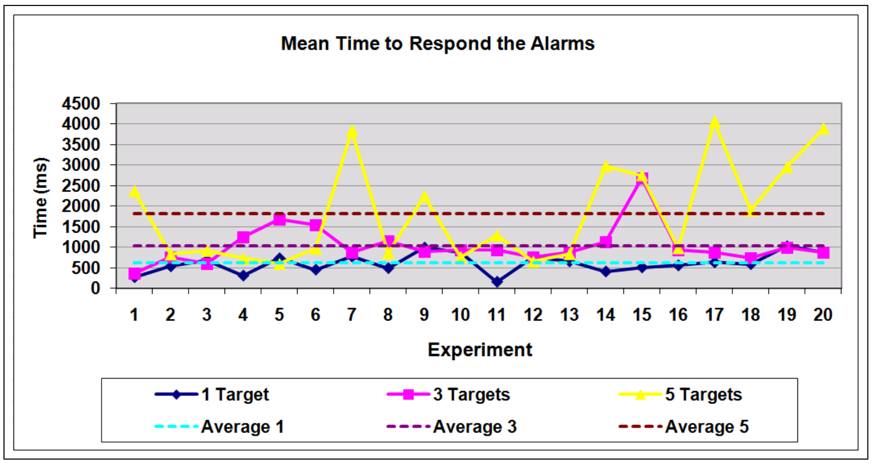

Figure 6: Alarm Response Time Achieve by the Simulation.

The second metric evaluates the system efficiency in terms of detecting a target and correctly routing the alarm message to an idle UAV. For all simulation runs, no alarm was lost, which means that the system had 100% efficiency for the simulated scenario and correctly found an idle UAV at all occasions when an alarm was issued.

B. Demonstrator Results

The simulations performed in ShoX showed that the approach proposed in this paper works well in the described scenario. However, wireless communications are very sensible to interferences and unpredictable variations. This means that simulation data, such as communication reachability and delays, are not always confirmed in real deployments. This fact motivated the deployment of a demonstrator to assess the properties of a network used to evaluate part of the system presented in this paper.

The deployed demonstrator is composed as a network consisting of sixteen static ground sensor nodes (nine SunSpots and the others FemtoNodes) and one mobile node (FemtoNode). The ground sensor nodes are equally distributed in a grid in an area of 225 square meters. The mobile FemtoNode, moved manually, represents a UAV that ”flies” over this area leaving pheromones over the ground sensor nodes via a periodic beacon message sent to the network. Upon the occurrence of an alarm, the nodes route it in the direction of the nodes with stronger pheromone traces, until it arrives at a node which has communication with the UAV. Figure 7 presents the demonstrator setup. The radio in the nodes was adjusted to provide a communication range of 5 meters, such that the nodes are capable of communicating only with their immediate vertical and horizontal neighbors, which are 5 meters apart, but not with their diagonal neighbors or any other node in the grid. The mobile node, representing the UAV, has the same communication range configuration as the static nodes.

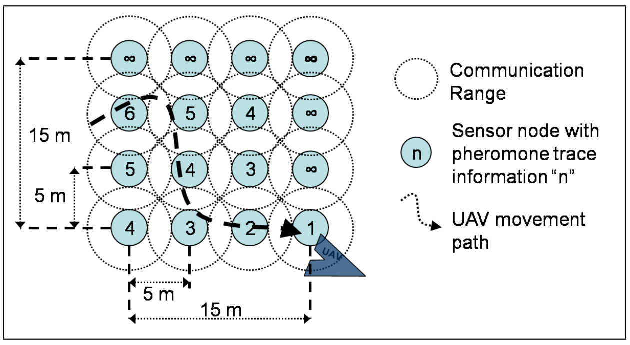

Figure 7: Demonstrator Setup.

The pheromone traces in the nodes are represented by the numbers in the center of the circles representing the ground sensor nodes in the figure. The smaller the number is, the stronger the pheromone. This translates the idea of the time past since a ground sensor node received the last pheromone beacon from a UAV. When a ground sensor node receives this pheromone beacon, it sends this information to its neighbors with a pheromone one point weaker (a number one unit greater than the one representing the node’s pheromone information). This is an indirect beacon that helps the other nodes find the traces to route the alarms. The nodes that receive the indirect beacons do not forward it. The symbol ∞ means that the node has no pheromone trace, i.e. the last beacon (directly from a UAV or indirectly from another ground node) was received a long time before, above a tunable threshold. The number representing the pheromone is periodically incremented, indicating that the pheromone trace becomes weaker when time elapses, until disappearing (become ∞). Figure 8 presents an example of how an alarm issued by a sensor node (Figure 8-A) is routed through the network, following the pheromone traces (Figure 8 from A to D), until it is delivered to a UAV (Figure 8-E).

Twenty runs were performed. In each of them, an alarm was generated by one of the static nodes, randomly chosen, which had to be routed to the UAV according to the pheromone mechanism described in Section 6 and implemented as described above. Some of the middleware features presented in Section 5 were installed in the static nodes, such as message priorization and QoS control, in terms of delay to forward an alarm message [31]. In order to stress the network and test these mechanisms, random messages were generated by the static nodes, which competed with the beacon and alarm messages for

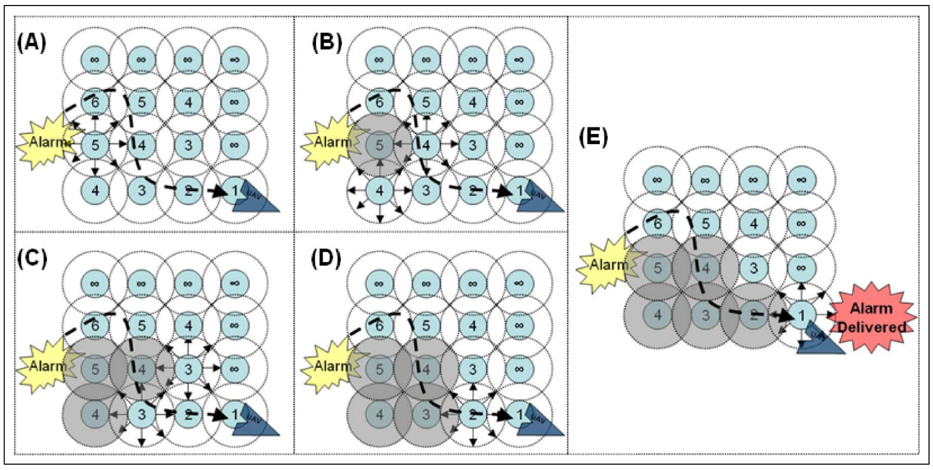

Figure 8: Alarm Routing and Delivery.

the utilization of the communication resources.

The evaluated parameter with the described testbed was the time to respond to the alarms generated in the system. By obtaining this metric, the delay of one hop communication was calculated and compared with the one achieved in the simulation results described before. Figure 9 presents the time taken by the system to deliver the alarm to the UAV.

The average number of hops to deliver the alarm was 5 hops for the 20 runs of the testbed. Taking the average of the time to deliver an alarm, 538.85 ms, and the average number of hops, we get an average delay of 107.77 ms in each hop.

Considering the simulation results, taking the worst case scenario, the one with 5 targets, in average, the number of hops for an alarm to be delivered was 13.78. Taking the average of worst case scenario, 1,821.65 ms to deliver an alarm, we get a 132.14 ms delay for an alarm to be forwarded among the static nodes in each hop.

Comparing the delays obtained from the simulation runs and from the demonstrator, it is possible to observe that they are very close to each other. The delays obtained with the demonstrator are even better than the ones achieved by simulation, which shows the applicability of the approach described in this paper.

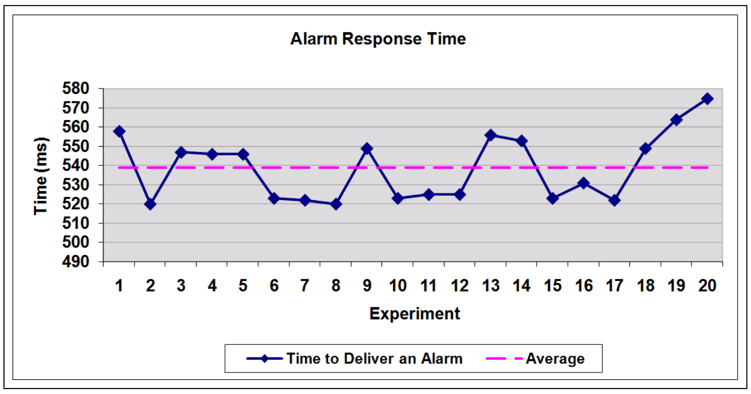

Figure 9: Alarm Response Time Achieved by the Demonstrator.

10 Conclusion and Future Work

This paper presented a system solution to provide interoperability and coordination support for heterogeneous sensor networks composed by ground static sensor nodes and mobile sensors carried by autonomous aerial robots. This solution is based on customizable sensor nodes and an adaptive middleware to manage different resources available in each customized sensor node. The FemtoNode platform provides a support to the development of different types of nodes, customized according to specific requirements. The middleware gives support to the network heterogeneity, enables adaptations to fulfill requirements that may change during the system run-time, and also promotes the necessary coordination among nodes.

The coordination and cooperation among distinct nodes in the network allow intelligent behavior of the unmanned vehicles, resulting in autonomous decisions regarding their movement and also how they can complement the work performed by other nodes. This intelligent behavior is based on data exchanged between nodes that then are aggregated and analyzed in order to support the autonomous decisions.

Simulation and testbed results were provided. These results assessed the suitability of the pheromone coordination mechanism, presenting delay time results for the delivering of alarm messages that drive the movement of the mobile sensor nodes in the system.

Additional simulations are planned in order to assess the effectiveness of other features of the middleware that were not yet explored, such as the use of cached values and multicast communication. In order to validate the overall proposal, a larger scale demonstrator is also planned to be deployed, with enlarged communication range and real UAVs controlled by FemtoNodes.

Acknowledgments

E. P. Freitas thanks the Swedish Knowledge Foundation and the Brazilian Army for the grant to follow the PhD program in Embedded Real-time Systems at Halmstad University in cooperation with UFRGS in Brazil.

Bibliography

[1] D. A. Schoenwald. AUVs: In Space, Air, Water, and on the Ground. *IEEE Control Systems Magazine*, Vol. 20, No. 6, pp. 15-18, 2000.

[2] D. Culler, D. Estrin, and M. Srivastava. Overview of sensor networks. *IEEE Computer*, Vol. 37, No. 8, pp. 41-49, 2004.

[3] E. P. Freitas, M. A. Wehrmeister, C. E. Pereira, and T. Larsson. Reflective middleware for heterogeneous sensor networks. *Proceedings of 7th Workshop on Adaptive and Reflective Middleware*, ACM, pp. 49-50, 2008.

[4] R. S. Allgayer, M. Götz, and C. E. Pereira. FemtoNode: Reconfigurable and Customizable Architecture for Wireless Sensor Networks. *Proceedings of 10th International Embedded Systems Symposium (IESS'09)*, Langenargen, Germany, pp. 302-309, 2009.

[5] Sun Microsystems. SunSPOT, www.sunspotworld.com

[6] A. T. Erman, L. Hoesel, and P. Havinga. Enabling Mobility in Heterogeneous Wireless Sensor Networks Cooperating with UAVs for Mission-Critical Management. *IEEE Wireless Communications*, Vol. 15, Issue 6, pp. 38-46, 2008.

[7] M. A. Batalin, M. Hatting, and G. S. Sukhatme. Mobile Robot Navigation using a Sensor Network. *IEEE Intl. Conf. on Robotics and Automation*, 2004.

[8] J. S. Kinnebrew, A. Gupta, N. Shankaran, G. Biswas, and D. C. Schmidt. A Decision-Theoretic Planner with Dynamic Component Reconfiguration for Distributed Real-Time Applications. *Proceedings of 8th International Symposium on Autonomous Decentralized Systems*, pp. 461-472.

[9] Y. Jin, Y. Liao, A. A. Minai, and M. M. Polycarpou. Balancing Search and Target Response in Cooperative Unmanned Aerial Vehicle (UAV) Teams', *IEEE Transactions on System, Man, Cybernetics-Part B: Cybernetics*, vol. 36, No. 3, pp. 571-587, 2006.

[10] B. Walter, A. Sannier, D. Reinerss, and J. Oliver. UAV Swarm Control: Calculating Digital Pheromone Fields with the GPU. *In The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, vol. 2005 (Conference Theme: One Team. One Fight. One Training Future), 2005.

[11] MSB Co. web site. Submeter-scale aircraft, http://spyplanes.com

[12] R. Leonard, and J. Drezner. Global Hawk and Darkstar. Santa Monica, California : RAND Corporation, vol. 4, 2002.

[13] E. P. Freitas, M. A. Wehrmeister, C. E. Pereira, F. R. Wagner, E. T. Silva Jr., and F. C. Carvalho. DERAF: A High-Level Aspects Framework for Distributed Embedded Real-Time Systems Design. Springer, pp. 55-74, 2007.

[14] E. P. Freitas, M. A. Wehrmeister, C. E. Pereira, and T. Larsson. Using Aspects and Component Concepts to Improve Reuse of Software for Embedded Systems Product Lines. *Proceedings of 13th Early Aspects Workshop at SPLC-08*, pp.105-112, 2008.

[15] M. E. Bratman. Intention, Plans, and Practical Reason. Cambridge, MA, 1987.

[16] Object Management Group (OMG). Distribution Service for Real-time Systems (DSS) Specification, Vol. 1.2, 2007.

[17] E. Bonabeau, M. Dorigo, and G. Theraulaz. Swarm Intelligence: From Natural to Artificial Systems. *Oxford University Press*, Santa Fe Institute Studies in the Sciences of Complexity, NY, 1999.

[18] P. Gaudino, B. Schargel, E. Bonabeu, and B.T Clough. Swarm Intelligence: a New C2 paradigm with an Application to Control of Swarms of UAVs. *Proceedings of 8th International Command and Control Research and Technology Symposium*, 2003.

[19] J. A. Sauter, R. Matthews, H. V. D. Parunak, and S. A. Brueckner. Performance of Digital Pheromones for Swarming Vehicle Control, *Proceedings of 4th International Joint Conference on Autonomous Agents and Multi-Agent Systems*, ACM Press, pp. 903-910, 2005.

[20] T. Heimfarth, and P. Janacik. Experiments with Biologically-Inspired Methods for Service Assignment in Wireless Sensor Networks, *IFIP Intl Federation for Information Processing*, Vol. 268, Eds. Boston: Springer, pp. 71-84, 2008.

[21] H. Hinkelmann, P. Zipf, and M. Glesner. A Domain-Specific Dynamically Reconfigurable Hardware Platform for Wireless Sensor Networks, *Int. Conf. on Field-Programmable Technology*, pp. 313-316, 2007.

[22] P. Garcia et al. An overview of reconfigurable hardware in embedded systems, *EURASIP J. Embedded Systems*, New York, NY, USA, n.1, pp.13-13, 2006.

[23] S. A. Ito, L. Carro, and R. P. Jacobi. Making Java Work for Microcontroller Applications, *IEEE Design and Test of Computers*, Los Alamitos, Vol. 18, No. 5, pp. 100-110, 2001.

[24] Sashimi Manual, www.inf.ufrgs.br/~lse/sashimi/, 2006.

[25] M. A. Wehrmeister, C. E. Pereira, and L. B. Becker. Optimizing the generation of object-oriented real-time embedded applications based on the real-time specification for Java, *Proceedings of The Design, Automation, and Test in Europe Conference*, Belgium, pp. 806-811, 2006.

[26] J. Lessmann, T. Heimfarth, and P. Janacik. ShoX: An Easy to Use Simulation Platform for Wireless Networks, *Proceedings of 10th International Conference on Computer Modeling and Simulation*, pp. 410-415, 2008.

[27] E. P. Freitas, T. Heimfarth, M. A. Wehrmeister, F. R. Wagner, A. M. Ferreira, C. E. Pereira, and T. Larsson. Using Link Metric to Improve Communication Mechanisms and Real-time Properties in an Adaptive Middleware for Heterogeneous Sensor Networks, *Advances in Information Security and Assurance*, LNCS, 5576, Springer Berlin, Heidelberg, 422-431, 2009.

[28] E. P. Freitas, T. Heimfarth, F. R. Wagner, A. M. Ferreira, C. E. Pereira, and T. Larsson. An Agent Framework to Support Sensor Networks - Setup and Adaptation, *Proceedings of International Multiconference on Computer Science and Information Technology*, pp. 533?540, Mragowo, Poland, 2009.

[29] E. P .Freitas, T. Heimfarth, F. R. Wagner, A. M. Ferreira, C. E. Pereira, and T. Larsson. Evaluation of Coordination Strategies for Heterogeneous Sensor Networks Aiming at Surveillance Applications, *Proceedings of 8th IEEE Sensors*, pp. 591?596, Christchurch, New Zealand, 2009.

[30] C. Bettstetter. On the Minimum Node Degree and Connectivity of a Wireless Multihop Network, *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, ACM, New York, NY, USA, 2002, pp. 80-91.

[31] E. P. Freitas, M. A. Wehrmeister, C. E. Pereira, and T. Larsson. Real-time Support in a Reflective Middleware for Heterogeneous Sensor Network. *In Proceedings of the 29th IEEE Real-Time Systems Symposium: Work-in-Progress Proceedings*, Barcelona, Spain, pp. 57-60, December 2008.

Edison Pignaton de Freitas - has a position as Computer Engineer at the Brazilian Army. He is currently a PhD student at Halmstad University, Sweden, and Federal University of Rio Grande do Sul (UFRGS), Brazil, in the area of sensor networks, performing his research in the embedded systems groups of both universities and has published several papers in this area. He got his Bachelor degree in Computer Engineering from the Military Institute of Engineering, Brazil, in 2003, and his MSc degree in Computer Science from UFRGS, in 2007. In 2001 and 2002 he participated in an interchange program for Engineering students between Brazil and France, studying at the Genie Informatique Industriel program of the Institut National des Sciences Appliques, in Toulouse, and performing an internship at the Systems Department of the AIRBUS, working in the A380 project.

Tales Heimfarth - received his PhD degree in Computer Science from the University of Paderborn, Germany, in 2007. His PhD dissertation was entitled "Biologically Inspired Methods for Organizing Distributed Services on Sensor Networks". He received a MSc degree in Computer Science in 2002, with the thesis "Real-time Communication Platform over an SCI Cluster", and a Bachelor degree in Computer Science in 2000, both from UFRGS. Since 2008 he holds a post-doc position at UFRGS, performing research in the area of sensor networks. He is author of several papers published in international conferences in the area of sensor networks, especially with focus on basic software and auto-organizing protocols. He participates of the ShoX project, which is a open-source, event-oriented simulator for ad-hoc networks, led by the University of Paderborn.

Rodrigo Schmidt Allgayer - is currently a PhD student at UFRGS, Brazil, in the area of wireless sensor networks, embedded systems and reconfigurable architectures, performing his research in the Control, Automation and Robotics Group. He got his Bachelor degree in Electrical Engineering in 2005, and his MSc degree in Electrical Engineering in 2009, both from UFRGS. His research interests include wireless sensor networks, reconfigurable architectures, and embedded systems.

Flávio Rech Wagner - received a BSc degree in Electrical Engineering (1975) and an MSc degree in Computer Science (1977), both from UFRGS, Brazil. He received a PhD degree in Computer Engineering from the University of Kaiserslautern, Germany, in 1983. In 1992 and 2002 he held post-doc positions at INPG (Institut National Polytechnique de Grenoble), France. He was invited professor at the University of Tübingen, Germany, in 1994. He is currently professor at UFRGS, position that he holds since 1977, and since 2006 he is the Director of the Institute of Informatics of UFRGS. He has been President of the Brazilian Computer Society, from 1999 to 2003. He is associate editor of the Design Automation for Embedded Systems journal and has been member of more than 30 program committees of international conferences, having published 3 books, 7 book chapters, and more than 120 papers in conference proceedings and journals.

Tony Larsson - became a Member of IEEE in 1990. He received a MEng degree in 1974, a Tech.Lic. 1986 in Computer Systems and a PhD degree in Computer Science in 1989, all at the Institute of Technology at Linköping University, Sweden. He worked for Ericsson AB from 1974 to 2002, in several different positions, such as engineer, manager, and expert (the latter in system design methods and architecture), in areas such as testing, computer aided design, radio network control, and distributed computer systems platforms for dependable telecommunication applications. He then worked for the Swedish defense material administration in the area of network based defense and is since 2003 Professor in Embedded Systems at Halmstad University.

Carlos Eduardo Pereira - received a BSc degree in Electric Engineering from Escola de Engenharia (1987), Brazil, and an MSc degree in Computer Science from UFRGS in 1990. He received a PhD degree in Electrical Engineering from Technische Universitat Stuttgart, Germany, in 1995. In 2000 he held post-doc positions at United Technologies Research Center (UTRC). He is a professor in the Department of Electrical Engineering and Science Computing at UFRGS and since 2009 he is the Deputy Director of the Engineering of UFRGS. He is a senior member of the IEEE and IFAC Technical Committee on Real-Time Programming. Has experience in Electrical Engineering, focusing on Electronic Automation of Electric and Industrial Processes, acting on the following subjects: Real-Time Systems, Object-oriented Programming, Industrial Automation, Industrial FieldBus and Software Engineering.

Armando Morado Ferreira - received BSc (1990) and MSc (1997) degrees in Mechanic Engineering, both from Military Institute of Engineering. He received his PhD degree in Mechanical Engineering from University of Delaware (2001), and MSc and PhD degrees in Military Sciences from Brazilian Army's Command and High-Staff School (2004 and 2009) Lieutenant-Colonel Military Engineer, professor at Military Institute of Engineering and instructor at Brazilian Army's Command and High-Staff School, he has knowledge in Mechanical Engineering with focus in robotization, concentrating his work in the following themes: trajectory generation, non-linear control and unmanned vehicles control and modelling, besides a great interest in the area of Information Science and Technology Management applied to Defence.