Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 619 – 626

ISBN 978-83-60810-22-4
ISSN 1896-7094

An Agent Framework to Support Sensor Networks' Setup and Adaptation

Edison Pignaton de Freitas
IDE - Halmstad University
Halmstad, Sweden
Institute of Informatics - UFRGS
Porto Alegre, Brazil
Email: edison.pignaton@hh.se

Tales Heimfarth
Computer Science Department - UFLA, Lavras, Brazil
Institute of Informatics - UFRGS
Porto Alegre, Brazil
Email: tales@dcc.ufla.br

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

Flávio Rech Wagner
Institute of Informatics - UFRGS
Porto Alegre, Brazil
Email: flavio@inf.ufrgs.br

Carlos Eduardo Pereira
Institute of Informatics - UFRGS
Porto Alegre, Brazil
Email: cpereira@ece.ufrgs.br

Tony Larsson
IDE - Halmstad University
Halmstad, Sweden
Email: tony.larsson@hh.se

***Abstract*—Sensor networks are being used in several emerging applications not even imagined some years ago due to advances in sensing, computing, and communication techniques. However, these advances also pose various challenges that must be faced. One important challenge is related to the autonomous capability needed to setup and adapt the networks, which decentralizes the control of the network, saving communication and energy resources. Middleware technology helps in addressing this kind of problem, but there is still a need for additional solutions, particularly considering dynamic changes in users` requirements and operation conditions. This paper presents an agent-based framework acting as an integral part of a middleware to support autonomous setup and adaptation of sensor networks. It adds interoperability among heterogeneous nodes in the network, by means of autonomous behavior and reasoning. These features also address the needs for system setup and adaptations in the network, reducing the communication overhead and decentralizing the decision making mechanism. Additionally, preliminary results are also presented.**

I. Introduction

ADVANCES in sensor network technology are providing means for the development new applications that were not viable until a few years ago. Surveillance is an application field that is experiencing great progress thanks to these advances. Sensor networks are becoming able to provide meaningful information that makes surveillance systems more reliable and efficient, being a strategic tool in different applications, such as military and environmental surveillance and rescue assistance support.

However, these emerging applications usually use different types of nodes working in the network, which poses different challenges to the integration of the system and especially in its coordination. Moreover, the scenarios in which they are deployed are generally characterized by a great dynamicity. This requires flexibility and autonomous decision making by the nodes to reconfigure and adapt the network to new conditions. Waiting for an operator intervention to drive such adaptations may not respond to the needs in terms of timing and accuracy.

The challenges posed by those new applications of sensor networks bring a need for distributed intelligence over the network and, at the same time, require a capability of awareness about changes in the environment and in the operation conditions that may enforce adaptations [1]. Traditional approaches to control and retrieve information from sensor networks, such as TinyDB [2], do not provide necessary features to overcome these challenges. State-of-the-art middleware are not capable of handling autonomous adaptations in the sensor nodes` behavior. They just provide means to make the information retrieval easier, but, if something changes, new configurations have to be implemented. Moreover, they consider networks of homogeneous nodes. Other proposals provide some advances in the direction of adding features that allow a certain degree of autonomous adaptation, such as in Agilla [3]. However, there is a lack of proposals that distribute the decision making capability among the network sensor nodes. This is desired firstly to avoid the dependence from a central entity that would have to deal with all changes that may take place in the network, and secondly because of the dynamic nature of the application scenarios. In order to tackle these problems, autonomy is a real need, and, in order to provide it, distribution of the decision making is required.

The aim of this work is to provide embedded software solutions for area surveillance systems, including setup and configuration mechanisms and capabilities that can adapt the system according to the application needs. This paper presents an agent framework that provides autonomous capabilities to nodes such that they may self-configure during a setup phase. The framework also promotes adaptations in the network and its nodes by means of autonomous reason-

This work is partially founded by the Brazilian CNPq (National Research Council) under the PNPD project, and by the KK- Foundation in Sweden.

ing among intelligent agents. Agents compose a middleware that promotes an intelligent interoperability among nodes in the network. This agent-based middleware also uses other technologies such as aspect-oriented handling of non-functional crosscutting concerns and component encapsulation, which play an important role in the customization of the middleware. However, the use of these techniques is out of the scope of this paper, which aims at presenting the agent framework, detailing the various types of agents and how they work in order to provide the desired features mentioned above.

In Section II, a presentation of the application scenario and a discussion about the related requirements that emerge from this scenario are provided. Section III gives a short overview of the general approach in which the agent framework is to be used. Section IV provides the core of the paper, presenting the framework and details about the agents that compose it. Results are presented in Section V, while Section VI discusses related work. Finally, Section VII draws the conclusions and gives directions for future work.

II. Application Scenario Description

In order to motivate the goals of the work described in this paper, this section presents the application scenario in which surveillance systems are usually used, as well as the requirements for these systems.

Surveillance systems are part of a class called ISR (Intelligence, Surveillance and Reconnaissance) systems, in which sensors are used to gather information about objects and events in a certain area that may represent or expose potential threat or targets, for example in military operations [4]. The operation scenarios of such systems are characterized by being non-deterministic and presenting a high degree of dynamicity, where both operation conditions and user requirements may suffer drastic changes in short time intervals. The more common changes in the operation conditions are sudden weather changes and losses of communication capabilities among nodes for several reasons (e.g., a node may fail, be out of range, or be destroyed). Additionally, user requirements may also change during operation, for instance requiring more details about a given event, such as higher resolution or more accurate data, or even completely changing the focus of a given mission.

ISR systems generally use mobile nodes to acquire the information needed by the users. However, with recent technological advances in the field of sensor networks, the use of both mobile and static nodes cooperating in a unified network is becoming a reality. Static nodes may be deployed by means of mobile nodes that drop sensors on the ground, which will provide valuable information in order to drive the use of mobile nodes. Assuming an area surveillance system composed by different kinds of sensors that may be fixed on the ground or mobile, carried by vehicles such as Unmanned Aerial Vehicles (UAVs), the rationale for the use of both kinds of sensors is that together they can provide large area surveillance with minimal costs.

However, the interoperability among nodes in such systems poses several problems that have to be solved in order to make the system respond according to the users expectations.

The first problem is how to disseminate the mission over the nodes in the network. A trivial solution would be to send mission directives to all nodes, but this is not very smart, considering that not all information concerning a given mission is interesting to all nodes.

The second problem is how to divide the work among the nodes after receiving a new mission. A trivial solution for this problem would be to take decisions in a centralized way and then send the specific part of a given mission (a submission) to the specific node that will take care of it. However, a central decision maker “oracle” must have information about the whole network operation and all environment conditions in order to make a good job. This would require an unnecessary traffic of control information from the entire network to the central “oracle”, wasting communication and energy resources and also taking more time. In turn, a distributed decision autonomously taken among nodes and made in a dynamic context can in reality give better performance within given time, communication, and energy constraints.

A third problem related to this is how to adapt or re-divide the work after a change in users` requirements or in operation conditions. If the system waits for the operator intervention, maybe it can be too late, so autonomous decision making capability is a strong requirement in this context. In order to face the dynamicity of the operation scenarios, the system has to be capable of providing different kinds of services in different places at different times. So, the third problem is how to provide this flexibility to autonomously move services to the places where they are required.

III. Approach Overview

The overall idea of the surveillance system proposed in this work is to facilitate mission definition by users and setup the network as to accomplish it. In order to do this, a high-level Mission Description Language (MDL) is used, which helps the users in specifying missions at a high-level of abstraction by defining a set of statements, without taking care about specific system parameters, such as the choice of sensors that will handle a given mission, or how the raw data should be aggregated in order to provide the information that the user requires. The directions established in these statements are then translated into a Global-Mission (a formal specification of the mission), which is broken down into a set of node-missions (submissions that can be assigned to individual nodes), and then this set is sent to the network. By receiving this set of node-missions, the nodes autonomously decide which node-mission each one will perform to comply with the mission needs. Figure 1 presents this overall scenario.

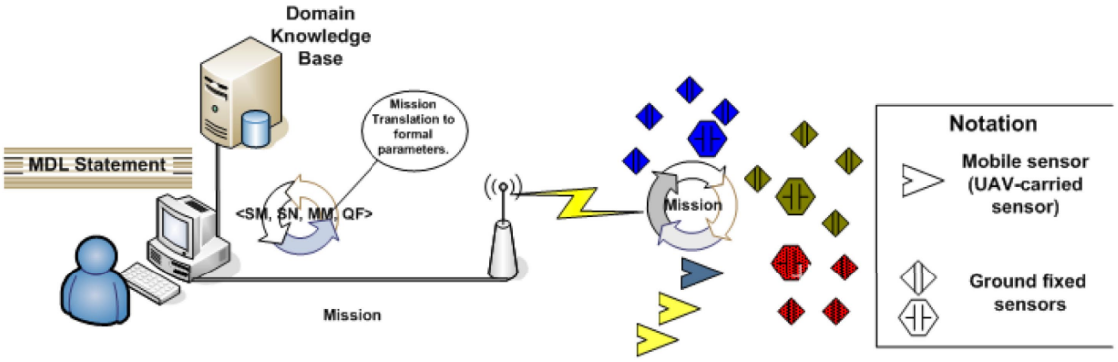

Fig. 1 Overall picture of the system scenario

For details about the MDL statements, its translation into a formal specification of a mission, and the formal specification itself, readers are addressed to [5] .

In order to tackle the problems highlighted in the previous section, the approach proposed in this work is to use an agent framework, which, as an essential complement to the middleware installed in the sensor nodes, provides the necessary support to accomplish the mission needs.

The proposed approach uses agent technology for four purposes: 1) to make decisions about the function to be performed by each sensor in the network, identifying the responsibilities for each node within the proposed mission; 2) to allow dynamic changes in the functionality provided by the middleware, in order to respond to specific needs that may arise due to changes in the environmental conditions, network configuration (nodes coming and leaving the network, new routes, etc), changes in the requirements described by the users, etc; 3) to carry information that can be used by other nodes by means of services provided by the agent; and, 4) to implement and carry applications by means of the interpreted mission direction, which has been translated into node-missions.

The framework and middleware also use other technologies besides agents to support the runtime environment, such as aspect-oriented design to handle non-functional crosscutting requirements and component-based software deployment, but these details are out of the scope of this paper. Readers interested in these subjects are referred to [6] and [7] for more details.

IV. Agent Framework

As highlighted in the previous section, agents are used for different purposes in the approach presented in this paper. This fact requires specialised agents to perform different roles, in order to cover the requirements presented in Section II. This work follows two classifications of the agents that are used, according to their main characteristics. The first classification is related to the mobility of the agent in relation to the hosting node, while the second classification is related to the computational model used by the agent. The former classifies the agents in mobile or fixed, and the latter classifies the agents in reactive or cognitive. The following subsections present each of these classifications and the mental model that explains the framework as a whole.

A. Mobility

According to this classification, the agents can be fixed in hosting nodes or have the ability to move from one node to another.

a) Fixed Agents

This kind of agent, which stays fixed in a sensor node, is responsible for high-level directions of missions. It reasons about the conditions of the network, the requirements of the missions that it gets, and its own state and the one of its hosting node. By making an analysis of its own conditions and of the conditions of the nodes in its neighborhood, this agent is capable of taking a local decision that is related to its engagement in a given mission, without the need to send control messages to other nodes in order to negotiate a mission distribution. This agent is also capable of negotiating when such action is required, but the protocol used for this negotiation is out of the scope of this paper. For more details, interested readers are referred to [5]. This agent is responsible for the construction of a plan that dictates how to perform the node-missions in the node, as well as to evaluate this plan and perform changes when this is required. This agent is called "planning-agent". It is important to highlight that the nodes that host the planning-agents may move, so what is "fixed" is the agent in the node, but not the node itself.

b) Mobile Agents

Mobile agents are classified according to their usage, as presented below:

Service-agents: these are agents that provide specific services, such as encryption, compression, and mathematical functions, among others. They are used to provide a node with a certain kind of service that is not yet installed, as a component or as an agent. Such agents have limited reasoning capabilities. Their intelligence is limited to the decision to move to another node or to clone itself and send its clone to some other node. This kind of decision can be made in response to an incoming message asking for which kind of service it provides, or in response to an event that allows it to foresee that its service will be needed elsewhere.

Update-agents: these are agents that provide changes in the components/agents installed in the middleware or even in aspects that affect them. They are used to perform administrative tasks, such as software updates or patching. They can perform them in a single or in multiple nodes of the

network. This kind of agent is temporary and is excluded from the node just after finishing its updating task.

Mission-agents: these are agents responsible for carrying, disseminating and allocating node-missions to different nodes in the network. Their role is to take the result of the Global-Mission interpretation, i.e. the tuple with the node-missions information, and to carry it to the nodes in the network, according to the location specified in the mission directions. The mechanism works as follows: after the MDL statements translation into a Global-Mission, an agent (or a group of agents, depending on the size of the Global-Mission and on the variety of node-missions that compose it) takes the description of the node-mission(s) that is (are) to be injected into the network. Then, it is (or they are) routed to the location specified in the mission directions, i.e. the Surveillance Area (SA). By arriving at the first node that can be part of the group that will be engaged in a mission execution, the mission-agent clones itself. The number of clones depend on the number of nodes that are in the area specified in the mission and are reachable in one-hop communication from the node currently hosting the agent and that have some relation to the node-missions specified in the Global-Mission that the agent is carrying. This means that the agent is communicated to the nodes that may take part in the set of nodes that will accomplish that mission, i.e. the agent sends the clones to these neighbor nodes. This mechanism is repeated until all pertinent nodes in the SA have a "geocasted" copy of this agent. During the cloning, the mission-agent adds information about the state of the current node, which will be used by the planning-agent in deciding if its respective node will take part in the mission accomplishment. Being in a node, the mission-agent "talks" with the planning-agent and delivers the requirements of the node-mission that fits to that node (for instance, a node-mission that handles measurements of temperature to a temperature sensor node). After the local decision taken by the planning-agents, the nodes that do not take part in a given mission have their respective mission-agent deallocated (or released). Figure 2 represents the above explained mechanism, highlighting the messages exchanged between a mission-agent and the planning-agent, with a "free style" interaction diagram representing the order of exchanged messages and events.

In Figure 2, (1) denotes the arrival of a mission-agent at a node, while messages (2)-(3) are exchanged among agents in order to allow the mission-agent to discover the type of the node on which it is located. In case the node is pertinent to the node-mission carried by the mission-agent, i.e. the node can be a possible candidate to perform the node-mission, the agents exchange more information (4.1a)-(4.1d). Still in this case, the mission-agent decides to clone itself (4.1e) and waits for the decision taken by the planning-agents (4.1f), in order to decide if it shall stay or be deallocated from the node (4.1g). If the node is not pertinent to the node-mission that it carries, the mission-agent follows its moving (4.2a).

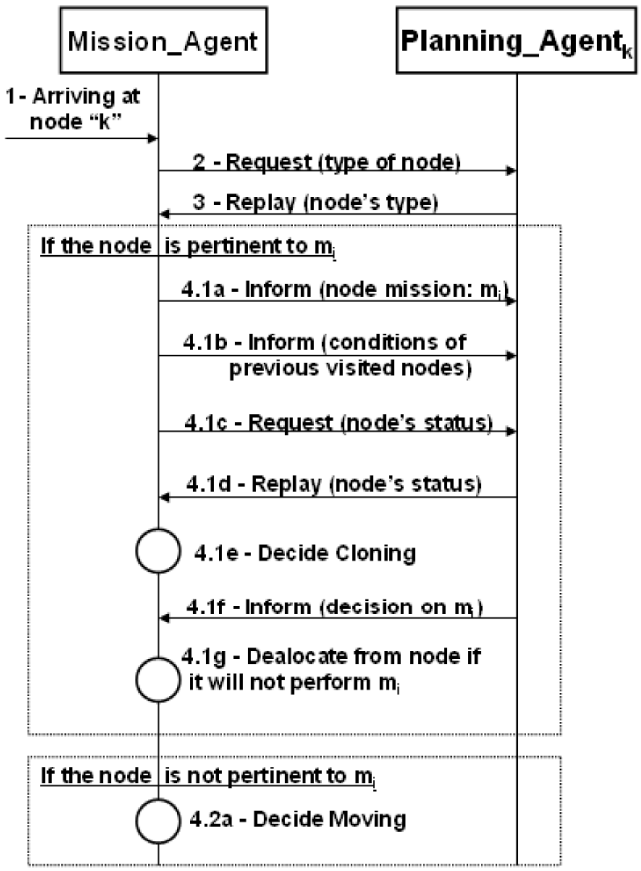

Fig. 2 Messages exchanged between mission-agent and planning-agent in a node.

B. Computational Model

Due to the differences in the usage and deliberation capabilities required by the different kinds of agents, different internal models were used to model them. Those with lower deliberation capabilities are modeled as reactive agents, while the others requiring more sophisticated skills are modeled as cognitive agents.

a) Reactive Agents

Service- and update-agents have limited deliberation capabilities. Their behavior is basically modeled by a direct reply to a certain event, without any need to perform elaborated reasoning about any statements or parameter interpretation. This fact leads to a reactive-based computational model, where the specific reactions depend on their internal mental state.

Service-agent: This agent reacts to calling messages requiring its services and to any other event that represents an indirect call for its services, such as an "order" from a planning-agent of its hosting node to move or clone to another node(s). Another reaction of this type of agent is to answer a request to perform its service.

Actions: move, clone, send, and receive (request and reply) messages to and from other agents.

Update-agent: Injected into the network, this agent moves through the nodes following the path to its destination and, upon arriving there, asks for permission from the

planning-agent that governs that node and then installs the software that it was carrying.

Actions: move, clone, install, patch, send and receive (request and reply) messages to and from other agents.

b) Cognitive Agents

Mission- and Planning-agents have more sophisticated intelligent behaviors. In order to perform their activities, they require some cognitive skills. The cognitive agent model adopted in this proposal is based on the BDI (Beliefs-Desires-Intentions) computational model, presented in [8].

Mission-agent: This agent has to take different decisions during its lifetime, such as: (i) the number of clones it will make from itself in order to disseminate a given mission; (ii) the type of node-mission to deliver to each node; and (iii) whether to deallocate itself from a node. These kinds of actions require mission-agents to communicate with the planning-agent that governs the hosting node, as to get information about what to do under given circumstances.

Beliefs: The location of the node where it is hosted; the type of the hosting node; the number of clones that it may generate; its utility in the hosting node.

Desires: To distribute the mission among the nodes in the correct location; to deallocate itself when it is not needed anymore.

Intentions: To produce a clone of itself in each of the correct neighbor nodes; to be in a node in the correct direction after a move; to have delivered the right node-mission to the node.

Actions: Ask the planning-agent about the current location; calculate the range of interest for a given mission; ask the planning-agent about the number of neighbors in range; ask the planning-agent the type of sensor(s) that the node has, as well as its status to forward to nodes to which it will clone/move; ask the planning-agent the result of the negotiation about node-mission distribution; inform the planning-agent about a node-mission; deallocate itself; clone; move; send and receive (request, reply, inform) messages to and from other agents.

Planning-agent: The planning-agent has the most complex "mental" activity, being responsible for different types of reasoning related to the mission accomplishment. It communicates with all other kinds of agents. This agent is responsible for deciding if the node will take or not a given mission. It also has to maintain updated information about its own state, in order to inform the other planning-agents and be capable of taking right decisions. Environment conditions are also important in some of the deliberations taken by this agent.

Beliefs: Basically consist of four groups of information: 1) background information, such as maps of the region; 2) the planning-agent's own conditions, translated in terms of the actions that it can perform and the node status (energy level, devices status, location, installed services, agents hosted in the node, etc); 3) other nodes status; and 4) environment conditions.

Desires: The planning agent has two types of desires: 1) General Desires, which correspond to "built-in" goals, such as distribution of the node-missions in order to achieve the best overall result efficiently and cooperation with other nodes; and 2) Specific Goals, which are related to the assumed node-missions and that come to its desires' set when it assumes the responsibility for a given node-mission. These goals are ranked according to the node-mission priority. It will be used to drive the construction of the plans that govern the execution of the agent's actions.

Intentions: Following the same idea of the desires, the planning-agent has also two types of intentions: 1) General Intentions, which are directly related to the built-in goals, such as to provide the required resources to a requesting node and the correct information to other agents about data of interest; and 2) Specific Intentions, which specify intentions related to actions to accomplish a node-mission, such as to send a given number of samples with correct accuracy and within timing constraints.

Actions: Calls via operating system or directly to device drivers to perform commands on the underlying software and hardware platform; send and receive (request, inform, reply, notify, subscribe, publish, propose, reject, accept) messages to and from other agents.

It is noteworthy to mention that, although the proposed work has no ambition to be fully compliant to the FIPA specification [9], this specification has been considered. One aspect that is important to highlight is that the types of messages mentioned in the agents' description allow further interoperability with agents running in FIPA-compliant systems such as JADE [10].

C. Decision Making Process

An important part of the system dynamics presented above by agents' behaviours is the process used to take the decision about the mission distribution among the sensor nodes, after the mission dissemination. This process, performed by the planning-agents, is carried out in a way to avoid exchange of additional control messages or broadcasts among nodes, which typically occur when negotiation protocols are used.

After the dissemination of the mission by the mission-agents, the planning-agents will get information about the employability of each node in their neighbourhoods that are eligible to that respective mission. Based on this information, on the employability of the node in which the planning-agent itself is installed, and on the mission directions, which include the requested density of nodes to perform the mission, besides additional requirements such as accuracy or threshold levels, the planning-agent in each node will perform a local decision making process that will result in an indication if the node may join or not the team that will accomplish the mission.

This process uses a weighted probability calculation. By receiving the information about its neighbours, via mission-agents, the planning-agent generalizes to the entire area of interest (SA) the configuration in its neighbourhood. By making this, it assumes that the density of requested nodes in its neighbourhood has to be proportional to the density required in the mission directions for the entire SA.

Each planning-agent sum up the employability values received from its neighbours with the one of its hosting node, calculates the contribution of its node to this result,

and, based on that, evaluates the probability of the node in engaging in the mission, according to equation 1. Each node will then randomly decide to engage or not in the mission, based on that calculated probability.

$$prob_i = \min\left(\frac{g_i}{\sum_{j=i} g_j} \times p \times n,\ 1\right) \quad (1)$$

where: "g_i" is the quality metric calculated by the quality function that evaluates how good the node is to engage in a given mission; "p" is the percentage of nodes capable of accomplishing the mission that should engage on that (node density); and "n" is the number of neighbors equipped with the required sensor.

D. Agent Framework Meta-model

In Figure 3, the class diagram that specifies the meta-model for the above explained Agent Framework is depicted.

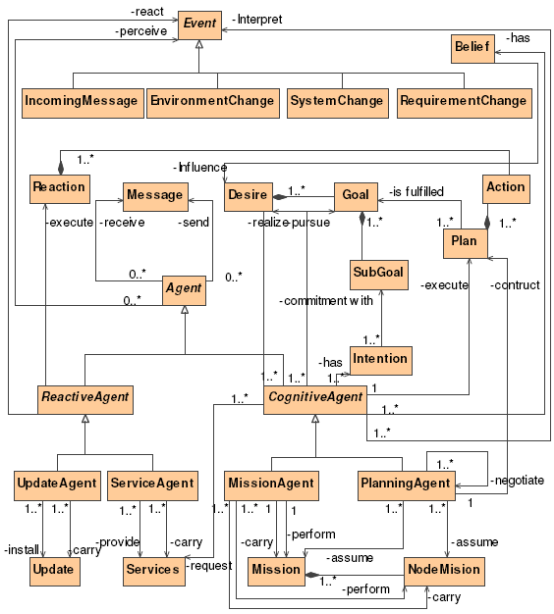

Fig. 3 Proposed agent framework metamodel.

In this proposed meta-model, an agent is classified as reactive or cognitive. The reactive ones are update- and service- agents. The cognitive agents are the mission- and planning- agents. An agent perceives events. An event can be an incoming message, an environmental change, a system change or a user's requirement change. Agents communicate via messages, which they can send and receive.

Reactive agents react to events executing reactions, which are composed by one or more actions.

Cognitive agents have beliefs, which may change by the agents' interpretations of their received events. Beliefs influence desires, which contain the agents' goals. Goals are divided in sub-goals, which represent the commitments of the cognitive agents' intentions. In order to pursue their goals, cognitive agents execute plans, which are composed by a sequence of actions. Particularly the planning-agent constructs its own plans. The execution of a plan may fulfill one or more goals, which will make possible for agents to realize their desires.

Mission-agents carry and perform missions, which are composed by node-missions. Planning-agents assume missions or node-missions, as a result of the decision making process explained above, or negotiations using the protocol described in [5]. Service-agents carry and provide services, which are requested by the cognitive agents in order to perform their activities. Update-agents carry and install updates. This overall picture of the framework summarizes the concepts presented in the previous sub-sections.

V. Simulation and Results

In this section results are shown related to the mission dissemination and first setup allocation mechanisms presented in Section IV, which is done by the use of mission- and planning- agents.

Taking the application scenario context presented in Section II, the use of ground sensor nodes can help define which kind of mobile sensor is more suitable to be used in a given situation. For instance, in order to decide to send a UAV equipped with visible light camera, radar, or infrared camera, the information about weather conditions in SA has key importance. Sensor nodes deployed in the area may provide this information, keeping track of phenomena like fog, driving the use of the most appropriate sensor.

Following the methodology described in , and summarized in Section III, a mission to monitor the presence of fog is stated and sent to the network by a mission-agent. This mission is translated to measurements of temperature and humidity and has a requirement related to the level of accuracy desired by the user as well as a constraint in the energy consumption. The result of this translation will also provide an important parameter that will define the density of the desired nodes to take part in the mission accomplishment.

A. Simulation Setup

The simulation was conducted using ShoX , a powerful wireless network Java-based simulator that provides easy extensions mechanisms. The proposed agent framework has been introduced in this simulator as an extension, in which mission-agents are serializable Java objects that are sent from node to node via communication packets. In the present simulation, the mission is small and fits in a tiny mission-agent that can be sent using just one communication packet of the IEEE 802.11b standard.

The area of interest, SA, has dimensions 5 Km x 5 Km, in which 8000 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 a 71% probability that the nodes in the network form a connected graph . From these nodes, 2000 have the sensing capabilities required (temperature and humidity sensors) to perform the described mission. The communication range used for the sensor nodes is 100 meters.

Each node starts the simulation with different levels of energy and sensor device status. The energy levels are ran-

domly distributed from 50% to 100% of the battery. This parameter affects the decision about the employability of a node to perform a mission, following the idea that nodes with higher levels of batteries are more likely to be employed. The sensor device status, which is also randomly distributed in the same range of the battery level, directly maps to the level of accuracy that the sensor is able to provide. This parameter abstracts aspects such as possible damages in the sensor devices, for example. Depending on the importance given to the accuracy, a relationship between energy and accuracy can be described to be used in the decision about the node employability, by means of the quality metric.

For this experiment, Equation 2 gives the function that calculates the metric carried by the mission-agent that will be used in the decision making process:

$$g_i = \alpha \times ac_i + (1-\alpha)e_i \qquad (2)$$

where: "α" is the weight of the parameter under concern for the mission, the accuracy in this case; "ac_i" is the accuracy of the node "i"; and "e_i" is the energy level of the node. In the experiments presented here, the "α" factor used was 0.5.

For all simulations (20 runs in total), the desired node density was established at 0.5, which means that from the total of 2000 sensor nodes capable to engage in the mission, it is desired that 1000 take part in its accomplishment.

B. Results

This section presents metrics to assess the efficiency of the proposed approach. The first metric calculates the number of packets needed to disseminate the mission among the nodes. The second one presents the number of nodes engaged in the disseminated mission, which is related to the desired node density presented above. Finally, the last results present the values of the quality metric used by the planning-agents to carry out the decision making process.

An important rationale for the proposed approach is to provide a way to assign missions to the sensor nodes with low overhead due to transmission of control messages. Aiming at this, the proposed agent-oriented approach provides a distributed way to allocate the mission in which the only overhead is due to the mission dissemination. In Figure 4, this overhead is presented by the number of application packets sent and received. The difference between the number of packets sent and received is due to the nature of the broadcast: a single transmission triggers the receival of a packet in all nodes located in the radio range. The figure confirms that the only overhead of the approach is the single message that each node broadcasts due to the mission dissemination. As there are 8000 nodes, 8000 messages are sent in this flooding process.

Figure 5 presents the number of nodes engaged in the mission disseminated for each run (Actual), compared to the ideal number of nodes defined by the node density factor (Ideal), as well as the average number of nodes along all (Average). It is possible to observe that, in average, the system engage 970 nodes, which is very close to the target number (1000). Moreover, the worst case deviation was around 15%. This shows that it was possible to achieve, with a complete decentralized approach, very good results concerning the number of engaged nodes in the mission.

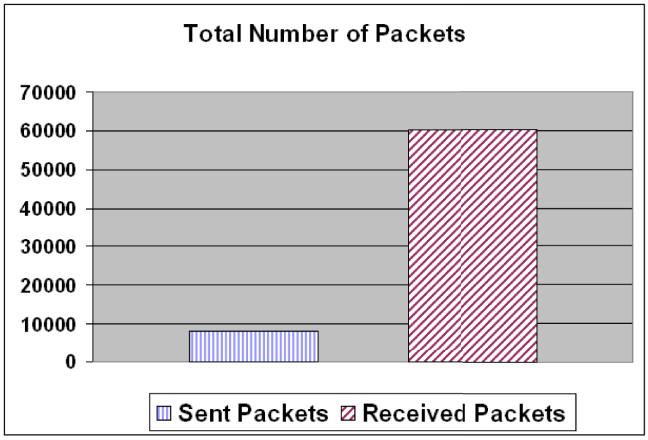

Fig. 4 Number of packets needed to disseminate the mission.

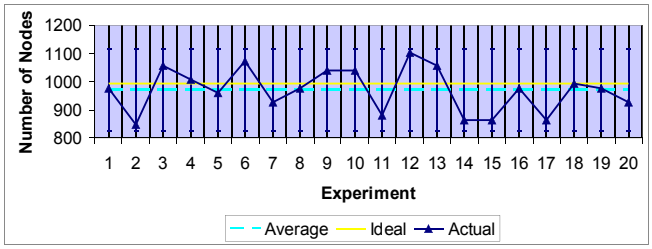

Fig. 5 Number of nodes engaged in the mission.

The information presented in Figure 6 describes the values obtained for the quality metric that the planning-agents use to perform the decision-making. The figure presents the average of the quality metric over all engaged nodes. The objective of the system is to engage the nodes in such a way that this metric is maximized. The figure shows, for each run, the mean value of the quality of the optimal assignment (obtained by a global view of the network). This is called "Optimal Solution". Moreover, the mean quality achieved with the proposed method is presented in the figure as "Distributed Heuristic". Besides the average in each run, it is also presented the average of the entire experiment for both optimal and heuristic solutions, respectively "Average of Optimal" and "Average of Heuristic". In average, the system presents a quality metric which achieved 80% of the optimal one. This means that the proposed heuristic engages very good quality nodes in the mission, not far from the optimal assignment.

VI. Related Work

Agilla [3] is one of the precursors in the use of mobile agents in middleware for WSNs. This approach uses agents that can move from one node to another. It also allows multiple agents to run in the same node. These characteristics provide the desired feature of energy saving, as the agents can run nearest to the data avoiding unnecessary communication. In our approach the use of agents is not restricted to move services around the network but also to help in the network setup, reflection and adaptability.

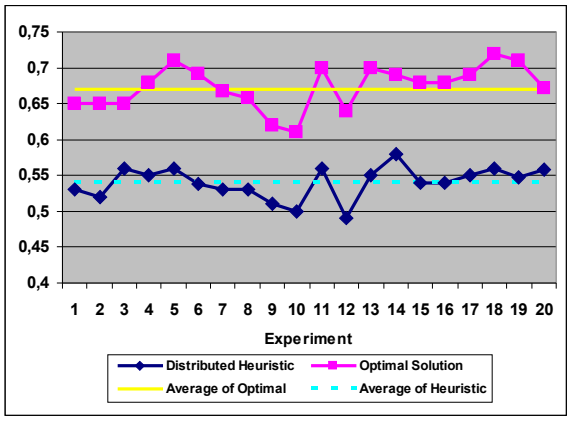

Fig. 6 Node quality assessment values.

In [11] an approach that uses Artificial Intelligence techniques to configure an underlying middleware is presented. This approach uses the concepts of missions and goals to plan the allocation of tasks in nodes of the network. Our approach differs from it because it uses agents with different levels of intelligence to address different types of problems, as discussed in Section II. Additionally, in that work, the intelligence is not part of the middleware and instead configures the middleware by sending external "commands" adjusting its parameters. In our approach, agents make part of the middleware, spreading intelligence over the network.

VII. Conclusions and Future work

This paper presents an agent-based framework to support reflection in a middleware for heterogeneous wireless sensor networks. The support offered by the agents in our approach provides autonomous reasoning capabilities that allow autonomous distributed decisions for different activities, from service allocation to mission distribution.

Moreover, a mission dissemination approach based on a quality metric is also presented. The proposed approach is responsible for recruiting sensors for a given mission, targeting at the optimization of the quality metric. The presented results show that, with very low overhead, the system was able to achieve a mission assignment that has 80% of the quality of the optimal solution. The very low overhead presented is achieved with decentralized coordination, which also brings better scalability than centralized approaches.

As future work, the complete agent framework has to be finished and integrated to the ShoX simulation tool. Additional simulations must be performed to assess the performance of our proposed methods, and comparisons with existing similar approaches have to be done.

Acknowledgment

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

T. Heimfarth thanks the Brazilian CNPq (National Research Council) for the funding provided to his research developed at UFRGS under a PNPD project.

References

[1] K. Henricksen and J. Indulska, "A Software Engineering Framework for Context-aware Pervasive Computing," in *Proceedings of PerCom*, IEEE Computer Society, March 2004, pp. 77–86.

[2] S. Madden, M.J. Franklin, J.M. Hellerstein and W. Hong, "TinyDB: An Acquisitional Query Processing System for Sensor Networks," ACM *Transactions on Database Systems*, 30(1), 2005, pp. 122–173.

[3] C.-L. Fok, G.-C. Roman and C. Lu, "Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications," in *Proceedings of the 24th ICDCS'05*, 2005.

[4] K.S. Tso, G.K. Tharp, W. Zhang and A.T. Tai, "A Multi-Agent Operator Interface for Unmanned Aerial Vehicles," in *Proceedings of 18th Digital Avionics Systems Conference*, 1999, pp. 6.A.4-1--6.A.4-8.

[5] E.P. Freitas, M.A. Wehrmeister, C.E. Pereira, A.M. Ferreira and T. Larsson, "Multi-Agents Supporting Reflection in a Middleware for Mission-Driven Heterogeneous Sensor Networks," in *Proceedings of 3rd International Workshop on Agent Technology for Sensor Networks*, 2009.

[6] 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," in *Proceedings of 10th International Workshop on Early Aspects*, Springer, 2007, pp. 55-74.

[7] A. Tesanovic et al. "Aspects and Components in Real-Time System Development: Towards Reconfigurable and Reusable Software," *Journal of Embedded Computing*, IOS Press, v.1, n.1, 2005, pp. 17-37.

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

[9] Foundation for Intelligent Physical Agents (FIPA). Specifications. 2008. Available from: http://www.fipa.org/specs/fipa00023/index.html

[10] Java Agent Development Framework (JADE). Online documentation. 2008. Available from: http://jade.tilab.com/doc/index.html

[11] D.C. Schmidt et al. "A Decision-Theoretic Planner with Dynamic Component Reconfiguration for Distributed Real-Time Applications," in *Proceedings of 8th ISADS'07* , 2007, pp.461-472.

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

[13] C. Bettstetter, "On the Minimum Node Degree and Connectivity of a Wireless Multihop Network," in *MobiHoc '02: Proceedings of the 3rd ACM International Symposium on Mobile Ad hoc Networking & Computing*, New York, NY, USA, 2002. ACM, pp. 80–91.