Wireless Sensor Network, 2013, 5, 14-25
http://dx.doi.org/10.4236/wsn.2013.52003 Published Online February 2013 (http://www.scirp.org/journal/wsn)

Multi-Agent Architecture for the Design of WSN Applications

Abdelhakim Hamzi[1], Mouloud Koudil[1], Jean-Paul Jamont[2], Michel Occello[2]
[1]Ecole Nationale Supérieure d'Informatique, LMCS, Alger, Algeria
[2]University of Grenoble, LCIS/INPG-UPMF, Valence, France
Email: a_hamzi@esi.dz, m_koudil@esi.dz, Jean-paul.jamont@iut-valence.fr, michel.occello@iut-valence.fr

Received November 2, 2011; revised December 7, 2011; accepted December 15, 2011

ABSTRACT

Complex and distributed systems are more and more associated with the application of WSN (Wireless Sensor Network) technology. The design of such applications presents important challenges and requires the assistance of several methodologies and tools. Multi-Agent systems (MAS) have been identified as one of the most suitable technologies to contribute to this domain due to their appropriateness for modeling distributed and autonomous complex systems. This work aims to contribute in the help of the design of WSN applications. The proposed architecture exploits the advantages of MAS for modeling WSN services, network topologies and sensor device architectures.

Keywords: Wireless Sensor Networks; Distributed System Design; Multi-Agents System; Agent Metamodel

1. Introduction

Wireless sensor networks (WSNs) are emerging as powerful platforms for distributed embedded computing supporting a variety of high-impact applications such as disaster/crime prevention and military applications, environmental applications, health applications, and smart spaces [1]. These networks pose important challenges for engineers working in the development of such systems: potential high number of nodes, limited resources, sensor heterogeneity, unreliable networks of changing topology, several levels of data processing [2].

In the last decade, wireless sensor networks have been interested by lot of researchers, the result is a huge number of works covering sensor devices design, routing and MAC protocols implementation, energy management, operating system and middleware, simulation, security, localization, deployment, etc. Despite these efforts, this domain still presenting important challenges and needs more potential focus, especially in methodologies and architectures helping the design of complex distributed systems and applications for WSNs [3].

Multi-Agent systems (MAS) have been identified as one of the most suitable technologies to contribute to this domain due to their appropriateness for modeling distributed and autonomous complex systems. In this paper, we propose a multi-agent architecture for the design of WSN applications where we tried to cover almost services of such system taking in consideration the heterogeneous configuration of the network. First, we will present the characteristics, requirements and design process of WSN applications with some related works using MAS paradigm. Before detailing our architecture and the agents composing its different modules, a WSN nodes types and agents classification is necessary to understand the proposed models. Finally, we will finish by a conclusion and some perspectives.

2. Background

2.1. WSNs Characteristics

WSNs are usually wireless dynamic networks composed of a large number of diverse nodes generally equipped with autonomous devices with continuous sensing, data processing and communication capabilities. These nodes are densely distributed in some area without human participation. Each sensor node is capable of detecting environmental conditions such as sound, temperature, and the presence of certain objects.

Although WSNs share common problems and solutions with embedded systems and ad hoc networks in what concerns hardware and network issues, there are new challenges that rise from the fact that these devices are resource constrained in term of energy, processing, storage and communication capabilities.

Table 1 presents a comparison between Wireless Sensor Network technology and other monitoring technologies [4].

The research community has identified sensor networks as a very challenging domain because of the following distinguishing features [4]:

Table 1. Differences between WSNs and other technologies.

Wireless Sensor Networks	Alternative Technologies
Low-cost low-power simple sensors	Expensive high-power consuming complex sensors
Cover wide-range areas	Cover small-size areas
Monitor remote or hostile environments	Monitor highly-controlled environments
Fault-tolerance and robust to node failures	Non-robust
Non-invasive	Invasive
Irregular sampled datasets	Regularly sampled datasets
Intrinsic distributed structure	Intrinsic centralistic structure
Low-bandwidth connectivity	High-bandwidth connectivity
Battery-powered	Electric-powered

Complexity**:** there is no easy way to manually design a sensor network that acts properly in all possible environmental and network changes.

Scale**:** they are usually composed of thousands of nodes, making infeasible approaches where the computational cost is exponential to the number of sensors.

Physical distribution**:** during their operation sensors have to deal with computation and information sources that are physically distributed.

Dynamics**:** sensor networks are dynamic systems that by effect of its internal changes or by effects of external forces change over time.

Resource availability**:** a key characteristic of sensor networks is that the demand for resources such as computation, power or bandwidth, is always higher than supply. Thus, there is a need for resource-awareness at all operation levels of the network.

Interdependence**:** the network may need to coordinate different sensors to achieve high-level tasks. Therefore, there are dependencies among sensors that make necessary to coordinate them during the sensor network operation.

Situatedness**:** Sensor networks are usually located in rapidly changing environments where the decision-making process of the sensor network has severe time restrictions.

2.2. WSNs Requirements

Any WSN system or application needs to response to some requirements inherent to its functioning such as nodes hardware and software requirement, deployment, communication, and energy management. Additional requirement, like QoS (Quality of Services) and security are sometimes necessary to guarantee an efficient operation of the WSN. Each requirement has it own parameters which can differ from an application to another.

2.2.1. Devices Parameters

The software and hardware parameters of sensor nodes are very important for the development of any WSN system. The hardware parameters include hardware platform specification (memory, chip, interfaces, radio, GPS, actuator, and energy resources) and sensor platform descriptions. The software parameters include the required OS/Virtual machine, needed amount of memory (RAM/ROM), a list of modules that the considered software module requires to operate and requirements to the hardware, like a radio or a specific sensor support.

2.2.2. Deployment or Network Parameters

Network parameters allow to describe the sensor network where the application is deployed. A good sensor network deployment should address a variety of problems such as sensing coverage (Topology, devices heterogeneity, density, average network diameter) network connectivity (bandwidth, mobility, fault rate), and deployment method (deterministic or ad hoc, terrain specifics, Moving object characteristics). Another deployment objective is the trade-off between the network lifetime and the number of sensor nodes (coverage, connectivity and redundancy).

2.2.3. Communication

Three major requirements are the target of every network protocol design: bounded delay, energy awareness and low overhead imposed to the network. These three requirements however seem to be contradictory to each other, thus demanding a trade-off in order to enhance one aspect of network performance at the expense of the rest.

Network Layer: Routing protocols are basically grouped by the proactive or reactive way they create routes. No standard metric exists, since each application imposes different requirements on the routing protocol. Some approaches relate this metric to energy consumption and some others to the real-time performance (hop-count, bandwidth).

MAC Layer: MAC protocols cover the handling of collision, minimizing the overhead and synchronization between nodes (idle, listening).

2.2.4. Energy Management

Due to the energetic resources restriction, both local and global power consumption management is necessary in order to increase the lifetime of the network. Each node can locally control its devices operations (sensing, processing, communication, etc.) according to the present energy. A good tasks scheduling must take in consideration the energetic capability of each node.

2.2.5. Query Management

Since the WSN is used to response to different kind of systems and applications requests, an efficient scheduling of queries is necessary. Some parameters should be taken in consideration when constructing queries such as: sources of data, periodic or instantaneous gathering, deadline, aggregation function, etc.

2.2.6. Quality of Services (QoS)

QoS refers to the capability of a network to provide better service to selected network traffic over various technologies. Delay and fault-tolerance are two main QoS parameters of wireless sensor networks. The first specifies the maximum allowable propagation delay between the gateway (base station or sink) and the farthest node from it, and the second is the tolerance of the WSN to node failure. A compromise between QoS requirements and WSN cost is necessary according to the type of the application.

2.2.7. Security

All security mechanisms are based upon cryptographic algorithms, discriminated in two major categories: private (symmetric) and public (asymmetric). Private cryptography is less demanding on computational power, but this imposes considerable control overhead. Public cryptography solves the key number issue, but it affects negatively the node performance and results into larger cipher data, burdening memory usage. Efficient management of the keys is crucial for WSN security architecture because of the ad hoc topology of the WSN and the restriction of nodes' resources.

2.3. Designing WSNs

Designing and programming WSNs is a complex task as system level code and application level code are often intertwined (see **Figure 1**). The users (and developers) of WSNs are often not software engineers but domain experts in other fields, who use WSNs to obtain data relevant to their work. Although this disparity in expertise, they are required to operate in a concurrent, realtime, resource constrained computing environment that is potentially complex and hard to program correctly. Several programming approach have been used in order to facilitate this task such as Object-Oriented [5], Component-Oriented [6], Service-Oriented [7] and Agent-Oriented programming [8].

The concept of agent provides a convenient and powerful way to describe a complex software entity that is capable of acting with a certain degree of autonomy in order to accomplish some tasks. The agent can be defined as autonomous, problem solving computational entities capable of effective operation in dynamic and open environments like WSNs. Typical characteristics of the agents are autonomy, intelligence, mobility, and social ability. The agents can be classified into multi-agent and mobile agent. The multi-agent is for handing complex operations requiring collaboration between the agents. The mobile agent moves through the network to process the tasks, and it is widely used for mobile computing with wireless network [4].

Various agent-oriented methodologies, metamodels and architectures exist to design and develop distributed and complex systems in an abstract manner. Regarding to WSN, almost contributions focus on the use of mobile agents to solve a particular problem such as routing, clustering, deployment, localization and security [9]. To make programming mobile agent for WSN easier, some architecture was proposed. The most popular are Sensorware [10] and Agilla [11].

Some interesting architectures tried to cover the gap in autonomous systems and WSN design. For example [12] is a tailored approach using MAS for the design of embedded collective systems and have been validated with the development of WSN application. [13] discuss a design methodology based on the service-oriented architecture and agile development principles for wireless embedded and sensor networks. [14] presented an agent-based framework acting as an integral part of a middleware to support autonomous setup and adaptation of sensor networks. [15] tried to exploit the advantages of model-driven engineering (MDE) approach to propose an architecture for the design of WSN based on the INGENIAS modeling language. [16] proposed a specification language centered on the concept of lightweight agent based on events and states. [17] proposed an interoperable model provided generalized frameworks to address some critical issues existed in the WSN such as interoperability, time synchronization, power management and distributed computation.

Frequently, almost proposed architectures and frameworks specialize on particular parts of WSN design and applications and only few works have been invested to derive a general architecture. This field still needs more investigation. Our work fills in this objective. It defers from related works in term that it tries to cover almost WSN's application requirements and different kind of network topology and nodes architecture.

3. Agents Definition Criteria

There are two main parameters that define the role of the agent in the system: the type of the node with the role it plays in the WSN and the type of the agent itself.

3.1. WSN Nodes Classification

The type of the node is strongly related to its hardware components which specify its functions in the network. Some functions are done by all nodes, the communication for example. Which define more the type of a node

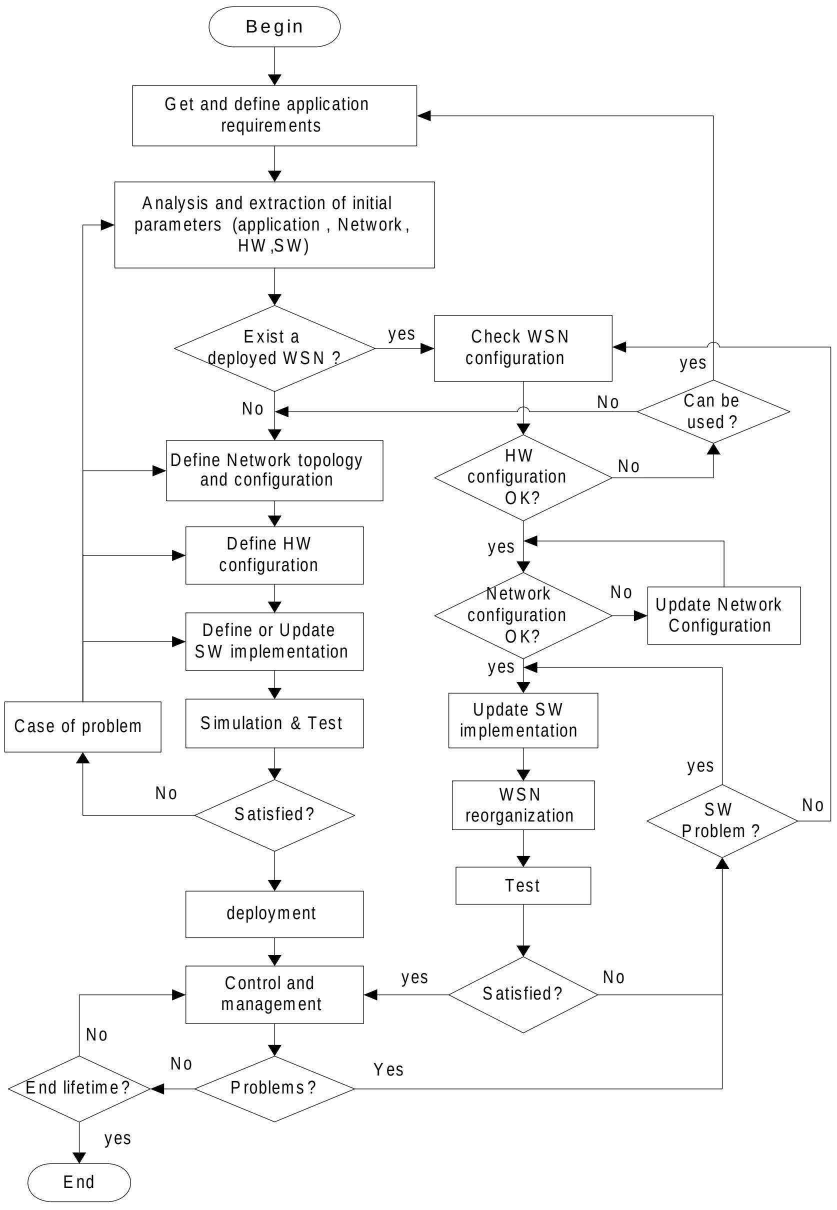

Figure 1. WSN applications design process.

are the additional functions that this node accomplishes.

- ***Base Station*** **(*or Sink*) (*BS*):** a well equipped node with more processing and storage capabilities. It is responsible of queries dissemination and data gathering. It can perform some processing on the data before being transferred. We can have more than one Base Station and some of them can be mobile.
- ***Communication Node*** **(*CN*):** generally, all the nodes can participate in the communication; some nodes can have other roles such as: sensing, processing and storage. The nodes whose role is only the communication are designed as CN.
- ***Sensor Node*** **(*SN*):** this type of node is responsible of the capture of measures and transforming them to a digital form to be sent to other nodes. Commonly used sensors include temperature, pressure, humidity, vibration, etc. Some sophisticated devices can have video camera.
- ***Processing Node*** **(*PN*):** this type of node has more processing capability than other nodes in order to perform some process on the gathered data such as aggregation, compression, encryption, etc. Generally this kind of nodes is designed as cluster-head.
- ***Storage*** **(*or Memory*)** ***Node*** **(*MN*):** generally, these nodes are used as cache memory for the WSN in order to store the data captured from the sensor nodes and not have been yet requested by the Base Station, or the code that may be dispatched to the nodes if they

change their roles in the network.

- ***Actuator Node* (*AN*):** this type of nodes has the capability to react with its environment. For example it can trigger an alarm, send a signal to another system or change its position.
- ***Localization Node* (*LN*):** some applications or protocols need to know the position of some nodes in the observed area. There are lot of localization algorithms, but almost of them are based on the knowledge of the position of predefine nodes called anchors nodes. Generally, the positions of these nodes are calculated using a GPS system.

3.2. Agents Classifications

Before giving a description of the agents composing the proposed architecture and their tasks in the system, it is important to present a short classification of agents. This can considerably help to understand the different types of agents and their roles. This classification is based on the metamodel proposed in [11] with some adaptation. In general, almost agent's types can be classified according to two views: mobile/fixed and reactive/cognitive.

3.2.1. Mobility

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

3.2.1.1. Fixed Agent

This kind of agent is an autonomous program installed in the node for high-level directions of missions. It reasons according to the conditions of the system, the requirements of the missions attributed to it, and its own state and that of its hosting node and neighborhood, in order to take a local decision related to its engagement in a given mission. We define two types of this agent:

Task-agent: this agent has limited deliberation capabilities. It performs specific and predefined tasks without make any intelligent decision. Generally, these tasks are controlled by a planning-agent.

Planning-agent: this agent has intelligent behavior. It is used to complete more complicated mission. Its reasoning and decisions are based on the changes of the state of the component where it is installed as well as on the predefined global-mission of this component.

3.2.1.2. Mobile Agent

A mobile agent is an autonomous program that moves from a node to another one, so it can run itself as soon as it needs without being called by other routines. It can determine where to migrate autonomously and carries on running its own program according to the previous state transferred from the last node where it existed. Mobile agents are classified according to their usage, as presented below:

Service-agent: this agent is used to provide a node specific processing services that is not yet installed, such as encryption, compression, and mathematical functions. Such agent have limited reasoning capabilities. Its intelligence is limited to the decision to move to another node or to clone itself and send its clones to some other nodes in response to an incoming message or internal event.

Update-agent: this agent provides changes in the components/agents installed in the system. It is used to perform administrative tasks, such as software updates. This kind of agent is temporary and it is excluded from the node just after finishing its updating task.

Mission-agent: this agent is responsible for carrying, disseminating and allocating node-missions to different nodes in the network. Its role is to take the result of the global-mission interpretation and to carry it to the nodes in the network, according to the location specified in the mission directions.

3.2.2. Computational Model

3.2.2.1. Reactive Agents

Service-agents, update-agents and task-agents have limited deliberation capabilities. Their behavior is basically modeled by a direct reply to a certain messages or events, without any need to perform elaborated reasoning about any statements or parameters interpretation. This fact leads to a reactive-based computational model, where the specific reactions depend on their internal mental state. Generally these agents run by the control of the planning-agent to perform some actions such as: execute-task, move, clone, install, send, and receive (request and reply) messages to and from other agents.

3.2.2.2. Cognitive Agents

Mission-agents and planning-agents have more sophisticated intelligent behaviors. In order to perform their activities, they require some cognitive skills. The cognitive agent model adopted is based on the BDI (Beliefs-Desires-Intentions).

Mission-agent: This agent has to take different decisions during its lifetime, such as: the number of clones it will make from itself in order to disseminate a given mission; the type of node-mission to deliver to each node; and 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 to get information about what to do under given circumstances.

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.

4. The Architecture of the Framework

The proposed architecture is composed of functional components or modules; each one is a composition of agents ensuring the services offered by this component as shown in the **Figure 2**.

4.1. Modules Description

In this section we will describe the main modules without entering in the role of the different agents used by these modules. More details are presented in the next section.

- ***System Module*** **(*SM*):** it presents the kernel of the application. The main role is to group the other modules and guarantee a coherent communication between them firstly, and with the WSN in the other hand.
- ***User Interface Module*** **(*UIM*):** the data collected by the WSNs can be directly used by the end user or by another system that makes some process on the data before being useful, or react according to them. This module presents an interface of communication between the WSN system and the user of the data gathered. This user can be locally connected to the WSN system or using a communication media. Several interface agents may exist because the WSN can be used by many different users (systems or applications) in the same time.
- ***Data Processing Module*** **(*DPM*):** generally the data collected from the area where the WSN is deployed need to be processed before being presented or communicated to the user. In some cases, the system using the WSN receives requests system or applications. Sometimes it is necessary to process and interpret these requests before sending them to the adequate WSN (the system can manage several WSNs).
- ***System Management Module*** **(*SMM*):** this module manages the functions of the system modules and ensures their good operability with the WSNs.
- ***Network Interface Module*** **(*NIM*):** this module is the interface between the system and the different WSNs. The SMM uses the NIM to send requests to the WSNs and receive data from them. It also allows the SMM to manage and control the different WSNs via the Network Management Modules.
- ***Network Management Module*** **(*NMM*):** the system can use and manage several WSNs in the same time for different purposes. Each WSN has its own NMM that ensures its functions and communicates with the system via the NIM under the control of the SMM. The main role is the management of the functional integrity and a coherent communication between the WSN modules.

4.2. Modeling Agents

According to the modules composing the system, the services needed by the WSN, the structure and the roles of the nodes, several agents may exist. These agents respect the classification presented above in order to specify their roles in the system and the network. Using these agents' models in our architecture, we tried to cover almost WSN applications, network topologies and protocols, and nodes architectures.

4.2.1. Interface Agents

The system managing the WSNs can be connected to several users or systems using different media. For each class or type of interface we can have a task-agent that manages this interface. Because these agents can use different means of communication to simply send information produced from the WSNs or perform some reactions, we can have more than one communication-agent of mission-agent and task-agent type.

4.2.2. System Management Agent

This agent is the principal controller of the system. It is a planning-agent that reasons according to the state of the system and the WSNs managed by this system as well as the requests received throw the user interface to make the adequate decisions and order the other agents of the system to perform some tasks.

4.2.3. Data Processing Agents

The system can perform some additional process on the data gathered by the WSNs (analysis, compression, encryption, etc.) before being sent to the user or for helping the System Management Agent to make the right decisions. For each kind of processing we can have a task-agent.

4.2.4. Network Management Agents

The system is interfaced with one or more WSNs using the network interface module. For each WSN there is a planning-agent that controls this network by ensuring a functional integrity and coherent communication between the different agents, and makes its decisions according to the state of the managed WSN and the global mission given by the System Management Agent. The principal roles of these agents are: ensuring an efficient deployment, organization and control of the WSN nodes; preparing queries and receiving responses; managing communications, energy consumption, QoS and security.

4.2.5. Deployment Agents

These agents are responsible of the deployment, reor-

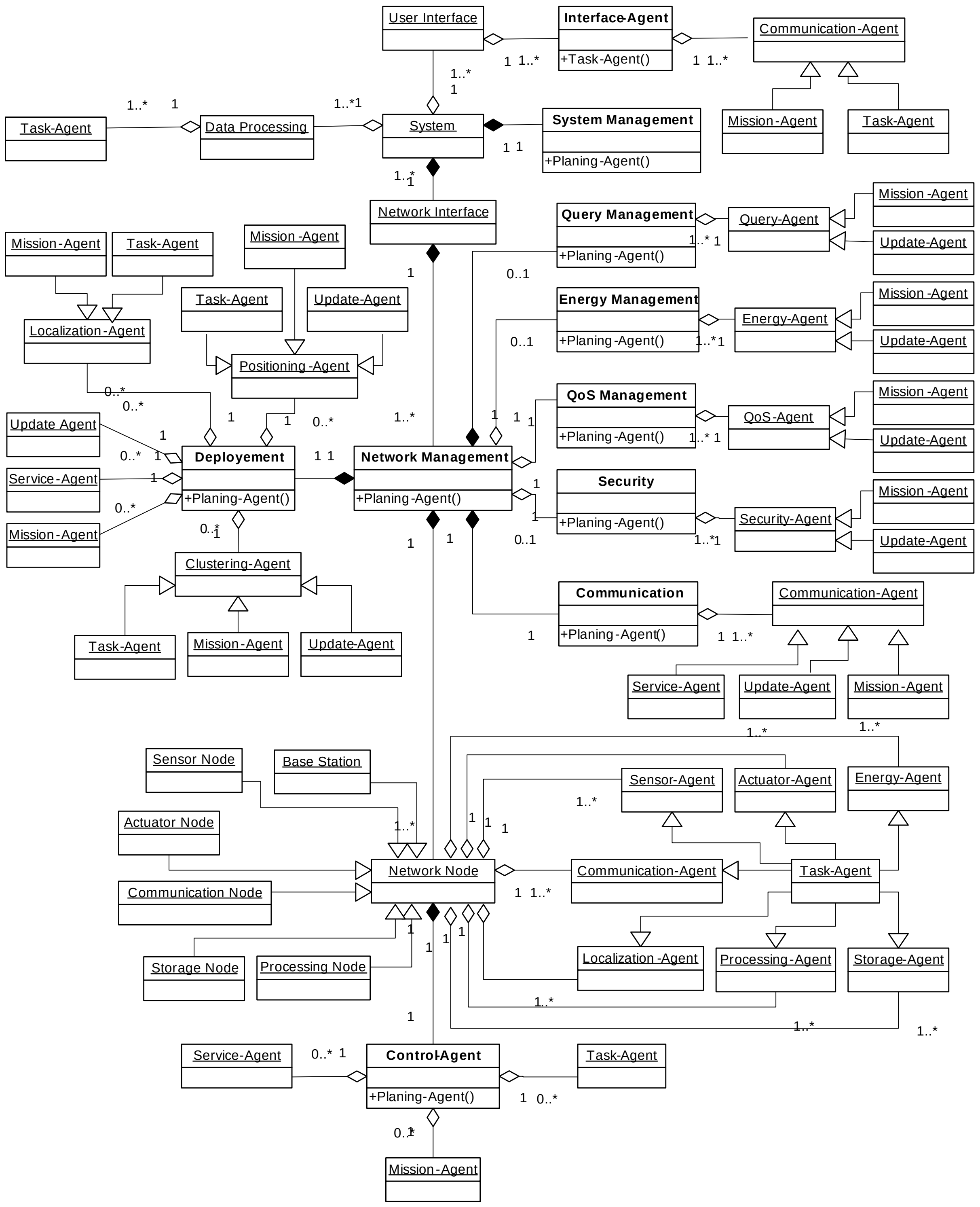

Figure 2. Framework architecture.

ganization and update of the WSN. To do this, a planning-agent that has knowledge about the state of the network and all nodes, control other agents to perform specific tasks such as localization, positioning and clustering. The planning-agent can use other agents for non-predefined tasks, for example the update-agent can be used to install some software update in the nodes, the service-agent is used to perform processing tasks that other agents need to accomplish their tasks and the mission-agent is used to assign the nodes a specific local missions.

- ***Localization Agents*****:** the role of a localization-agent is to calculate the geographic position of the nodes. The method of localization can be centralized; in this case it is carried out by a task-agent, as it can be distributed; in this case a mission-agent is disseminated to the nodes responsible of this task.
- ***Positioning Agents*****:** the role of a positioning-agent is to change the geographic position of the mobile nodes in the network. This can be done by a centralized or decentralized method. In the first case a task-agent choose the position according to the decision of the planning-agent of the deployment module and an update-agent is used to order the mobile node to take the specified position. In the other case, the planning-agent disseminates mission-agents to the mobile nodes to collaborate together in order to negotiate and decide their positions.
- ***Clustering Agents*****:** the clustering in a hierarchical topology network is the reorganization of the WSN on groups of nodes where a cluster-head is elected to manage the group of nodes and perform some processing tasks on the data gathered by its members. The clustering phase is very important for some hierarchical routing protocols, so it is very useful to assign to the WSN such service. Like the positioning, this task can be centralized or decentralized. In the first case, a task-agent specify the cluster-heads and the members of cluster using the parameters given by the planning-agent of the deployment module, then an update-agent is disseminated to update the role of each node in the network. In the other case, the planning-agent disseminates mission-agents to the nodes in order to negotiate and elect the cluster-heads and the members of clusters.

4.2.6. Query Agents

The query agents are responsible of analyzing the received requests, constructing queries and choosing the adequate base stations that can disseminate them to their closed nodes. They can also manage the storage nodes used as cache memory of the captured data. These tasks are managed by a planning-agent which controls other *query-agents*. A query-agent can be a simple update-agent that updates a specific base station or a storage node with a list of queries, or a mission-agent that is disseminated to the base stations which decide according to their states the capability to perform this mission or to collect some data from storage nodes.

4.2.7. Communication Agents

These agents implement the protocols of communication used for the selection of the efficient multi-hop path that data take from the nodes to the base stations. A planning-agent is responsible for the management of the process of communication. It can decide which role to play by each node involved in the communication according to the topology of the network. It also can ask the deployment-agent to reorganize the network when it observes an inefficient routing. The planning-agent controls three other types of agents: the update-agent is used to update the network layer of the nodes with the implementation of the routing protocol used, the mission-agent is used in the case where the routing protocol is based on a mobile agents, and service-agent is used to add to the routing protocol processing services (aggregation, compression, encryption, etc.).

4.2.8. Energy Management Agents

Because of the energetic limit of the nodes, the management of energy consumption is a crucial task in a WSN. The energy management agent is a planning-agent that has knowledge of the energetic state of nodes, so it can provide other management agents with some parameters that can help them in their decisions. This planning-agent can control tow kinds of *energy-agents*: the update-agent updates the energy-agent of the node with the strategy of the consumption of the energy and the mission-agents that can traverse the WSN in order to make knowledge about the energetic state of the network.

4.2.9. QoS Management Agents

QoS agents try to guarantee some degree of efficiency for real-time and high-level of availability for the WSN applications. A planning-agent collaborates with other agents (deployment, query, energy, communication) in order to assign certain priories to some nodes. A QoS strategy can be defined by the planning-agent then dispatched to the concerned nodes using an update-agent, or mission-agents are disseminated to a specific area, analyze the states of the nodes and their environment then negotiate the adequate strategy of the QoS.

4.2.10. Security Management Agents

The main roles ensured by these agents are: public key management, authentication, data encryption and malicious nodes detection. To do these, two kinds of agents can collaborate under the control of the planning-agent. The update-agent assigns the nodes with security policy and parameters defined by the planning-agent. The mission-agents are used in the case where the security mechanism use mobile agent for example to detect security vulnerability in the network or malicious nodes.

4.2.11. Nodes Agents

Depending on the hardware components of the node and the role it plays in the network, several agents can exit as shown in **Figure 3**.

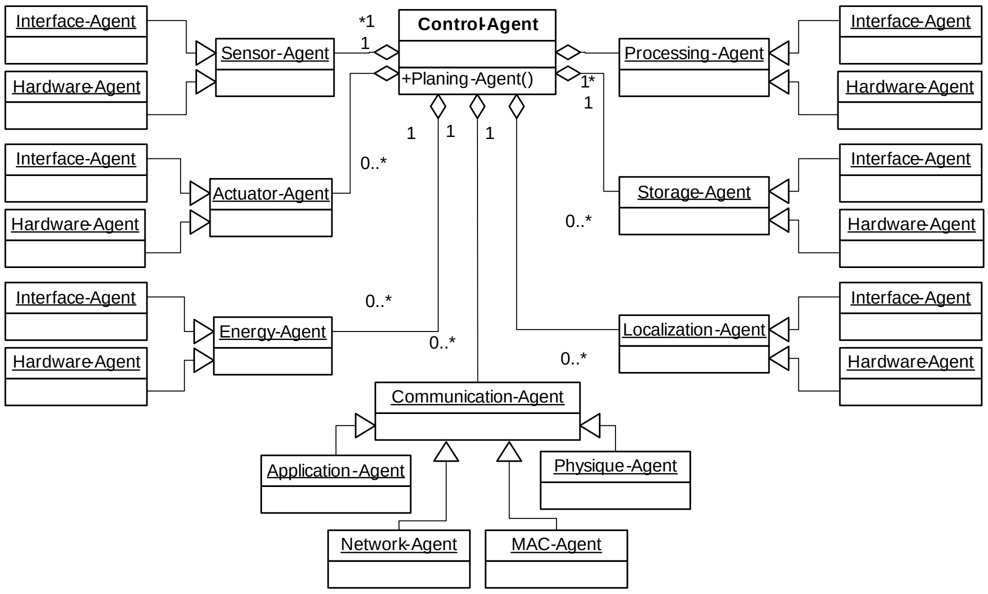

Figure 3. Different types of nodes agents (Hardware- and Interface-agents are Task-agent type).

- ***Control Agent*:** this agent is the coordinator of the node. The main roles are the control of its functions, the collaboration with the control agents of other nodes, the reception of updates and missions, and the coordination between the other agents present in this node. The type of this agent is planning-agent so it has reasoning and decision capabilities based on the global-mission of the WSN, the local-mission assigned to this node by the Network Management Agent, the message received from other nodes and the event trigged by the hardware component controlled by the other local agents.
- ***Communication Agents*:** these agents are responsible of the communication with the other nodes. Because of the communications task is based on layered model, for each layer there is task-agent that ensure the tasks done by this layer under the supervision of the control agent. For example the Physique-agent control the state of the transceiver (idle, transmission, transmission-reception, signal power, etc.), the MAC-agent ensure the timing and synchronization between nodes, the Network-agent implement the routing protocol tasks and the Application-agent perform some processing on the data received from another nodes.
- ***Sensor Agents*:** Sensor Agents are task-agent. They manage all kinds of sensors present in the nodes. There are two kinds of agents: the sensor-hardware-agents which control the function and the state of the sensors (idle, acquisition, sensing parameters, etc) and the sensor-interface-agent which play the role of interface between the Control Agent and the sensors installed in the nodes.
- ***Energy Agents*:** the energy in wireless sensor networks is a crucial constraint. The energy source is classified into three categories: storage (batteries or capacitors), Energy harvesting mechanisms (photovoltaic materials, ambient vibration, thermal gradients), and transfer (inductive coupling, capacitive coupling, and passive backscattering) [18]. The role of energy agents is the control of the energy generation and consumption and informing the control agent with the state of the energetic capacity of the node. There are two kinds of agents: the energy-hardware-agents and the energy-interface-agent.
- ***Processing agents*:** these agents control the processing devices of the node (processor, FPGA, DSP, etc.). Like other hardware components, there are two kinds of agents: processing-hardware-agent and processing-interface-agent.
- ***Storage Agent*:** this agent is responsible for the management of data and code storage. It collaborates with other storage agents in order to ensure an efficient redundancy and distribution of data in the storage nodes present in the network.
- ***Actuator Agents*:** the role of this type of agents is the control of the devices that allow the nodes to react with the environment under the supervision of the control agent (trigger an alarm, send a signal to other devices, and change the position of the node). Like other hardware components, there are two kinds of agents: actuator-hardware-agent and actuator-interface-agent.
- ***Localization Agents*:** the role of this type of agent is the control of the localization system like a GPS. There are two kinds of agents: Localization-hardware-agent and Localization-interface-agent.

5. Example: Battlefield Surveillance

5.1. Description of the Application

This application example deals with the design of a system which can detect and classify multiple targets (vehicles and troop movements).

The objectives are: classifying objects, estimating the number of each type of objects, estimating the direction and the speed of movements.

1) The objectives require that Software must be able to:

- Classify targets from the power of vibration signal, the estimated distance from some nodes, some view of image and a database of predefined object characteristics.
- Estimate the number of objects, their direction and the speed of movement from the power of vibration signal, the position and the time when the target has been in this position, and the type of targets.

2) We can make Hardware suppositions:

- We have heterogeneous nodes.
- The majority of nodes are sensor nodes using vibration to detect the target.
- Some mobile nodes which can fly for a small distance (hundred of meters). These flying nodes are enough small that can't be detected. They are equipped additionally to the vibration sensor with low resolution camera and GPS, and can update its power from an energy storage station.
- Some data storage nodes used to store the history data of detected target.
- Some small energy storage stations are used to update the energy of the flying nodes.

5.2. Modeling the Application

5.2.1. Data Structures

- ***Query format*:**

1) Vibration sensor nodes: [Node_Id, position (x, y), signal_value, direction, time].

2) Camera sensor nodes: [Node_Id, position (x, y), image_data, time].

- ***Agent format*:**

(Node_Id, Module_Id, Agent_Id, Role_Id, [List_Task], [List_Object]).

***Node_Id*:** the identifier of the hosting node.

***Module_Id*:** the identifier of the module or the component where the agent is executed. If we have submodules, the identifier of the internal module is the concatenation of all its encapsulating modules identifiers.

***Agent_Id*:** the identifier of the agent. According to the identifier we can know the type of the agent. For example the identifiers of planning-agents is between 0 and 31 of the mission-agent between 32 and 127, and so.

***Role_Id*:** the identifier of the role assigned to the agent. To define the code executed by the agent we use three tables:

1) *Roles table*: each entry has two information: the Role_Id and list of Task_Id.

2) *Tasks table*: each entry has two information: the Task_Id and list of Object_Id.

3) *Objects Table*: each entry has three information: Object_Id, parameters and Object_Code. The parameters is inherent to object execution such as the memory size necessary for the execution of the Object_Code, the energy consumption per time unit by the processor, and energy consumption per time unit by the transceiver (Send_Energy and Receive_Energy).

5.2.2. Definition of Agents

We will define some important agents as illustration.

- ***Query and processing agents***

They request and analyze data in order to classify targets and estimate their number and movements. **Table 2** gives examples of query and processing agents and their roles.

The centralized processing is used in the case where the troop is moving rapidly. It needs a big volume of real-time data so it decreases the lifetime of the network.

For the distributed processing a mission-agent is sent from the base station, traverses storage nodes, analyze data, and identify the requested information. It is used when the troop is in rest in order to minimize the quantity of data transferred to the system.

- ***Deployment agents*:**

The deployment and the reorganization of the WSN is based in several agents as shown in the **Table 3**.

1) For the vibration sensor node, an ad hoc deployment is used. It is important to identify the optimal number of nodes according to the observed area.

2) For energy and data storage node, a deterministic deployment is used in order to ensure an equitable distribution.

Table 2. Query and processing agents.

Agent	Type	Role
Query	Planning-agent	Manage queries.
Update_Query	Update-agent	Update base stations and storage nodes with the format of requested data.
Local_Processing	Task-agent	Centralized processing.
Mobile_Processing	Mission-agent	Sent from the base station. Traverse storage nodes analyzing data, and identify the requested information. Used when the troop is in rest in order to minimize the quantity of data transferred to the system.

Table 3. Deployment agents.

Agent	Type	Role
Deployment	Planning-agent	Manage the deployment, the localization and the reorganization of nodes.
Positioning	Planning-agent	Manage the positioning process.
Position_Update	Update-agent	Order a flying node to take a specific position.
Positon_Define	Mission-agent	The flying node negotiates and decides the position it takes according to the given mission.
Localization	Planning-agent	Manage the localization process.
Update_ Localization	Update_agent	Inform nodes with the position of other nodes.
Negotiate_ Localization	Mission_agent	Nodes calculate their positions with the help of the flying node which knows its position using a GPS.
Clustering	Planning-agent	Manage the clustering process.
Update_Cluster	Update_agent	Update the roles of cluster members.
Negotiate_Cluster	Mission_agent	Decentralized clustering.

3) For flying node, a deterministic deployment according to the position and the capacity of the energy storage node. Using this kind of node ensure a fault tolerance because the flying nodes can occupy the gap caused by nodes failure meanwhile the reorganization of the network.

4) A hierarchical topology is suitable for the detection of the objects their direction and speed.

6. Conclusion

Despite the many and significant contributions to the different research topics in WSNs, more research effort is required especially for design aided architectures. The presented approach is intended to facilitate the high-level design of sensor networks based on MAS. It includes an agent-oriented modeling of WSN services and sensor devices architectures. The high degree of reusability of models and transformations, and working at a higher-level of abstraction than that of code, are expected to reduce the costs of developing the complex autonomous embedded systems. As future works, we will try validate our architecture with the implementation of a real case study application using the proposed models then implementing a middleware based on this work for the automatic (or semi-automatic) generation of low-level code according to the simulators as well as the physical architecture used for the development of WSNs.

REFERENCES

[1] E. Yoneki and J. Bacon, "A Survey of Wireless Sensor Network Technologies: Research Trends and Middleware's Role," *Technical Report*, UCAM-CL-TR-646, University of Cambridge, Cambridge, 2005.

[2] I. F. Akyildiz, W. L. Su, Y. Sankarasubramaniam and E. Cayirci, "A Survey on Sensor Networks," *IEEE Communications Magazine*, Vol. 40, No. 8, 2002, pp. 102-114. doi:10.1109/MCOM.2002.1024422

[3] J. Yick, B. Mukherjee and D. Ghosal, "Wireless Sensor Network Survey," *The International Journal of Computer and Telecommunications Networking*, Vol. 52, No. 12. 2008, pp. 2292-2330.

[4] M. Vinyals, J. A. Rodriguez-Aguilar and J. Cerquides, "A Survey on Sensor Networks from a Multi-Agent Perspective," *Proceedings of the 2nd International Workshop on Agent Technology for Sensor Networks*, Estoril, 13 May 2008, pp. 1-8.

[5] Ž. Živanov, P. Rakić and M. Hajduković, "Wireless Sensor Network Application Programming and Simulation System," *Computer Science and Information Systems*, Vol. 5, No. 1, 2008, pp. 109-126.

[6] A. Dearle, D. Balasubramaniam, J. Lewis and R. Morrison, "A Component-Based Model and Language for Wireless Sensor Network Applications," 32*nd Annual IEEE International on Computer Software and Applications*, Turku, 28 July-1 August 2008, pp. 1303-1308.

[7] E. Meshkova, J. Riihijarvi, F. Oldewurtel, C. Jardak and P. Mahonen, "Service-Oriented Design Methodology for Wireless Sensor Networks: A View through Case Studies," *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, Taichung, 11-13 June 2008, pp. 146-153.

[8] Z. Guessoum and J.-P. Briot, "From Active Objects to Autonomous Agents," *IEEE Concurrency*, Vol. 7, No. 3, 1999, pp. 68-76. doi:10.1109/4434.788781

[9] M. Chen, S. Gonzalez and V. C. M. Leung, "Applications and Design Issues for Mobile Agents in Wireless Sensor Networks," *Wireless Communications*, Vol. 14, No. 6, 2007, pp. 20-26. doi:10.1109/MWC.2007.4407223

[10] A. Boulis , C.-C. Han, R. Shea and M. B. Srivastava, "Sensor Ware: Programming Sensor Networks beyond Code Update and Querying," *Pervasive and Mobile Computing*, Vol. 3, No. 4, 2007, pp. 386-412. doi:10.1016/j.pmcj.2007.04.007

[11] C.-L. Fok, G.-C. Roman and C. Y. Lu, "Agilla: A Mobile Agent Middleware for Self-Adaptive Wireless Sensor Networks," *ACM Transactions on Autonomous and Adaptive Systems*, Vol. 4, No. 3, 2009, Article No. 16.

[12] J. P. Jamont and M. Occello, "Designing Embedded Collective Systems: The DIAMOND Multiagent Method," *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence*, Patras, 29-31 October 2007, Vol. 2, pp. 91-94.

[13] E. Meshkova, J. Riihijarvi, F. Oldewurtel, C. Jardak and P. Mahonen, "Service-Oriented Design Methodology for Wireless Sensor Networks: A View through Case Studies," *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, Taichung, 11-13 June 2008, pp. 146-153.

[14] E. P. de Freitas, T. Heimfarth, A. M. Ferreira, F. R. Wagner, C. E. Pereira and T. Larsson, "An Agent Framework to Support Sensor Networks' Setup and Adaptation," *Computer Science and Information Technology*, Mragowo, 12-14 October 2009, pp. 619-626.

[15] R. Fuentes-Fernández, M. Guijarro and G. Pajares, "A Multi-Agent System Architecture for Sensor Networks," *Sensors*, Vol. 9, No. 12, 2009, pp. 10244-10269. doi:10.3390/s91210244

[16] G. Fortino, A. Garro, S. Mascillaro and W. Russo, "Specifying WSN Applications through Agents Based on Events and States," *International Conference on Sensor Technologies and Applications*, Valencia, 14-20 October 2007, pp. 463-468.

[17] F. Y. Xiong and L. Bai, "Interoperable Wireless Sensor Network Model Using Multi-Agent-Based Middleware," *International Symposium on Intelligent Signal Processing and Communication Systems*, Chengdu, 6-8 December 2010, pp. 1-4.

[18] S. Cheekiralla and D. W. Engels, "A Functional Taxonomy of Wireless Sensor Network Devices," 2*nd International Conference on Broadband Networks*, Boston, 3-7 October 2005, pp. 949-956.