Multi-agent Topologies over WSANs in the Context of Fault Tolerant Supervision

Gonçalo Nunes[1,3], Alberto Cardoso[1], Amâncio Santos[1,2], and Paulo Gil[1,3]

[1] CISUC, Department of Informatics Engineering, University of Coimbra, Portugal
[2] Instituto Superior de Engenharia de Coimbra, Portugal
[3] Departamento de Engenharia Electrotécnica, Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa, Portugal
{gnunes,alberto}@dei.uc.pt, amancio@isec.pt, psg@fct.unl.pt

Abstract. Wireless Sensor and Actuator Networks can be used to detect and classify ephemeral distributed events, where different process components with different behaviours are involved. Agents implementing Distributed Artificial Intelligence techniques are a key value in improving the overall system's performance. This paper proposes a general WSAN Multi-Agent based architecture for robust supervision and fault tolerant control.

Keywords: Agents, Mobile Agents, Multi-Agent Systems, Fault Detection and Supervision, Wireless Sensor and Actuator Networks.

1 Introduction

Wireless Sensor and Actuator Networks (WSANs) has attracted considerable attention in the last few years. They are distributed networks of sensors and actuators nodes, which act together in order to monitor and/or control a diversity of physical environments [1]. Each node is a small electronic device with wireless communication capabilities, including data storing and processing power, which can be programmed to interact with the physical environment by means of incorporated sensors and actuators. Additionally, they present reduced dimensions and can be used in a number of applications, including military, medical, process industry, environmental tracking, home automation, surveillance systems, just to name a few [2], [3]. In the industrial context, WSANs may be used in rare event detection or periodic data collection. In uncommon event detection, nodes are used to detect and classify rare, random, and ephemeral events, such as alarms or faults detection notifications. On the other hand, periodic data acquisition can be required for operations such as monitoring and control, reducing installation and operation costs [4], [5]. Furthermore, unlike traditional wired networks, nodes of a WSAN can be deployed in hostile or inaccessible environments, which is impractical with normal wired approaches. Because of their intrinsic features, WSAN can be a useful and powerful solution for a number of practical applications. However, since a WSAN is a preprogrammed system, it does not allow coping with unpredictable contingencies. What happens when the sensor network is faced with specific constrains, like energy consumption, data optimization, quality of service, for which it was not designed? How to achieve flexibility in the

Luis M. Camarinha-Matos (Ed.): DoCEIS 2011, IFIP AICT 349, pp. 383–390, 2011.

WSAN? The answer relies to some extent on the incorporation in a single framework of distributed artificial intelligence (DAI) methodologies along with Multi-Agent Systems (MAS) [6].

Agents are intelligent and autonomous software programs capable of interacting with other software components within a given application, and sharing a common goal. The integration of agents in a given environment can be remarkable advantageous in the case of several distinct process components (exosystems) with different behaviors and dynamics, for which it is, required to communicate with each other to perform a given global task. Agents can be organized in a particular multi-agent framework, where they cooperate in order to solve particular problems, inexorably constrained to the system's teleonomy, and taking advantage of their specific skills and individual knowledge. In industrial environments they can be used, for instance, in fault diagnosis or in the implementation of reconfiguration heuristics applied to local digital controllers, or creating drivers that exhibit a certain degree of tolerance to faults.

The present work proposes a robust data processing multi-agent based framework, ensuring the performance (reliability, timeliness, and precision) and data quality monitoring (QoS - Quality of Service), as well as fault diagnosis capabilities to deal with common WSANs constraints. Section 2 introduces the concept of agent and Multi-agent based systems, describing common multi-agent topologies and their features from the perspectives of physical and software abstractions. In section 3, the multi-agent based WSAN architecture is described focusing on structural and functional issues. Finally in section 4 some conclusions are drawn.

2 Technological Innovations for Sustainability in Multi-agent Based Systems over WSANs

Agents can be defined as computing entities, comprising a certain degree of autonomy and having the ability to feel and/or actuate in order to achieve predefined goals [1], [7], [3]. Other features include *i*) *Autonomy*: Agents are independent entities, able to accomplish a given task, without any programming or direct intervention; *ii*) *Reactivity*: agents are capable of perceiving their environment and respond quickly and effectively to changes; *iii*) *Pro-Activity*: agents are able to take initiative goals and behave in order to meet them; *iv*) *Cooperation*: agents have the ability to interact and communicate with each other to meet their goals and *v*) *Intelligence*: in order to evaluate and take over a task in autonomous way, the agent should incorporate intelligent techniques [7], [5].

The manufacturing industry has entered an era in which computer technology has refocused attention from hardware platforms to operating systems and software components [7]. The need for continuous real-time information (available at any time to many people) is currently pushing information technology providers to develop control system models and management systems to support this need. Multi-Agent Systems offer a new approach to designing and building complex distributed systems that significantly extends previous approaches and methodologies, like object-oriented or distributed computing [8]. It promises to be a valuable software engineering solution concerning the development of complex industrial systems, where complexity and spatial distribution of processes call for new methodologies.

Committed with the industrial panorama, the WSAN appears to be a powerful solution. Although, sensor/actuator nodes have limited resources, namely, reduced physical size, small memory, limited computation, small energy budget, and narrow bandwidth [3], [13]. A large group of this small and low-cost sensor nodes can be deployed to a given domain of interest and form a distributed networking system, in which collaboration among several sensor/actuator nodes is crucial to overcome the limited sensing and processing capabilities of each node and to improve the reliability of the decision making process [6]. The concept of mobile agent is a valuable solution to deal with these challenges.

In a mobile-agent based approach [9], [11], the transfer unit is the software agent itself and is based on three intrinsic premises: *autonomy*, *communication* and *mobility*. This approach may be advantageous for collaborative information processing in sensor networks, extending the inherent functionalities of networks, allowing saving the network bandwidth and computation time, since unnecessary nodes visits and agent migrations are avoided [9]. Since their scale is very small and bandwidth requirements are reduced, the necessary energy for transmission process is very low, thereby optimizing the energy autonomy of each node [12].

3 Three-Level Architecture

This section focus on a multi-agent based data processing architecture for WSAN, guaranteeing a certain level of performance, data quality monitoring and fault diagnosis functionalities, as well as the possibility of integrating Fault Detection and Identification (FDI) techniques with Fault Tolerant Control (FTC) modules. The overall goal is to provide a suitable framework for robust supervision purposes over WSANs, to meet application specific performance targets to integrate with industry resource systems. The framework facilitates the integration of the physical environment with software agents by means of the two main agent's characteristics, namely, communication and mobility, and enables a comprehensive handling of the fault management problem by the association of each monitoring activity to a particular agent service. The multi-agent approach conceptualized in this architecture will bring to the WSAN system, the following advantages: *i*) *Modularity and Scalability*, instead of appending new features to a system, agents can be added/launch or deleted without breaking or interrupting the overall process or even the task currently running in a given node; *ii*) *Mobility*, when, in a local node, an agent thumps, it can readily be regenerated by uploading it again from the server to the local node; *iii*) *Reliability*, in case of an active link break-down from the local node to the supervision system, local agents could take over the closed loop system stabilization role, considering the last reference vector, until communications are re-established; *iv*) *Concurrency*, agents are able to perform tasks in parallel, giving more flexibility to the networked system itself and also speeding up the computation and *v*) *Collaborative dynamics*, agents share their resources to perform their goals.

3.1 Architecture Overview

The conceptual multi-agent based WSAN hierarchical (3-level) architecture (Figure 1) comprises the following constituting parts:

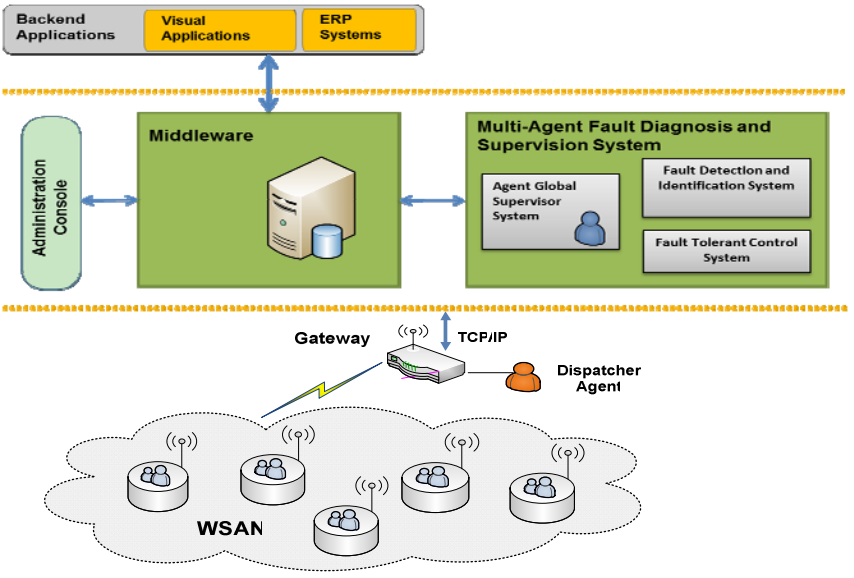

Fig. 1. Multi-agent based approach for distributed fault diagnosis and control

i) *Middleware platform*: Middleware platforms are used to connect applications, services and components, which interact within a given system architecture. It is positioned in the middle of a typical three tier architecture. In this work an event-based middleware provides the application user with an extensive set of functionalities for developing distributed systems. It allows users to collect process and reason real-time data, as it is processed through the system. This middleware, as an event-based system, provides a strong concept of decoupling which applies to both internal middleware components (query processing and distribution or adapter framework) and external components (the WSAN, at the process level, and, at higher level, the business applications, like visual applications and ERP systems), which communicate using the middleware;

ii) *Multi-agent fault diagnosis and supervision system:* This module is responsible for accommodating three components, namely, the Global Agent Supervisor system (GAS), which houses the agent platform responsible for the global monitoring and management of local manager agents (see section 3.3); Fault Detection and Identification system (FDI), devoted to the implementation of FDI techniques; Fault Tolerant Control module (FTC), assigned to the implementation of fault tolerant control techniques, which can be used together with the FDI system;

iii) *Wireless Sensor and Actuator Network:* The WSAN comprises several sensor/actuator nodes, deployed in the environment for monitoring and control. Each node acquires data from exogenous systems under monitoring, to which it is assigned, this information being subsequently perceived and processed by dedicated software agents' services (see section 3.2). These local agents provide nodes with the autonomy to respond accordingly to faulty events, for instance, in case of

time-outs or latencies exceeding a predefined threshold in the wireless communications. The exogenous system outputs are converted from analogue to digital, and vice-versa, by ADCs (Analogue-to-Digital Converters) and DACs (Digital-to-Analogue Converters) that are embedded in the sensor/actuator node microcontroller (Figure 2).

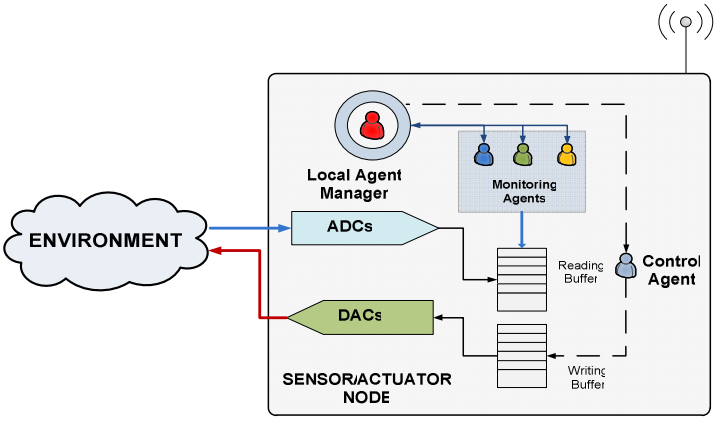

Fig. 2. Local multi-agent architecture

3.2 Local Multi-agent System

The architecture is based on the multi-agent paradigm [10], where each agent is responsible for a specific task. In terms of *modus operandi*, this framework relies on a collaborative and sharing profile/approach, which is a necessary condition to any distributed system. As can be seen in figure 2, each local sensor/actuator node includes a set of agents for monitoring and control purposes, accordingly to environment sensing and actuator constrains, as well as to trigger predefined faulty alarms to a local macro-agent (Local Agent Manager) responsible to reacts to these faulty events. The following features and services are included in each node:

i) Local Agent Manager

The main purpose of the Local Agent Manager (LAM) is to carry out extensive management routines related to other dependent (lower lever) local agents, and monitoring the communication status between the sensor node and the gateway. Whenever active, it automatically launches specialized monitoring agents, with the purposes of real-time monitoring of exogenous output.

The LAM is also responsible for continuously monitoring the status of these local subordinate agents. If one of these agents crashes, the manager kills the corresponding thread and re-launches its clone from the local agents' repository. Furthermore, in case of repeated crashes or agent's corruption, a regenerating request is sent back to the GAS, via the gateway Dispatcher Agent (DA). A default function associated with the LAM is to choose which agents should be launched and when, depending on the

environment status. If a monitoring agent is no longer necessary, the local manager removes it from memory, although it can always be re-launched whenever necessary. Another task concerns the message latency estimation, regarded as the elapsed time between the analogue data acquisition and its reception at the gateway's side using an active link. Since control actions are computed in the Fault Diagnosis and Supervision System (FDSS), in the server, and sent to the sensor node, if the communication link breaks-down the local manager should detect this faulty event and proceed accordingly, so as to not compromise the system's stability or performance. The LAM then launches a stabilizing Control Agent (CA) that is responsible to stabilize the system around the last reference received from the gateway. When the connection is re-established the local manager will remove this agent and the control authority is transferred to the FDSS.

ii) Monitoring Agents

The Local Monitoring Agents (LMAs) are launched by the local manager. By accessing the reading buffer, the LMAs continuously follows up the data read from the exogenous system or environment and is programmed to react to faulty events, such as abnormal sensor measurements or structural faults, just to name a few. In those cases, local agents are responsible for triggering alarms that are sent to the LAM. Since the computing power is limited the proposed fault diagnosis methodologies are based exclusively on univariate or multivariate statistics. In the univariate approach, the upper and lowers bounds define nominal operation conditions on the basis of a predefined threshold, and their violation triggers a predefined fault alarm. In multivariate T2 approaches more than one observation variable is used for decision-making. Let us assume we have a data set $X \epsilon R^{nxm}$, comprising m variables and n observations. Then, the corresponding covariance matrix is given by $S=1/(n-1)X^{T}X$. Assuming that the covariance matrix is invertible, the Hotelling's $T2$ statistics [14] can be given by $T2=z^{T}z$, where $z=\Lambda^{-0.5}V^{T}x$, being Λ and V given by the singular value decomposition of S. Appropriate thresholds for $T2$ statistics based on the level of significance α can be determined by assuming that observations are randomly sampled from a multivariate normal distribution. Therefore, faults can be detected for observations outside the elliptical confidence region, that is $T2 \geq T2\alpha$, triggering in this case a predefined fault alarm.

iii) Control Agents

The main task of Control Agents (CAs) is to stabilize the exogenous system around the last reference received from the gateway, in case of communication link breakdown. For this reason, these agents are stacked in the agent's local repository and are only launched by the LAM if this scenario (fault) occurs. Their role is crucial for the autonomy of the WSAN system, allowing the system to respond dynamically to communication time-outs, and thus ensuring the operationally of the system under control, in case of connection failure.

3.3 Dispatcher Agent

The network gateway is a central component to this architecture, since it establishes the interface between the low-layered information and the upper-layered modules, resolving and forwarding packets from the WSAN to the upper modules. Regarding the implementation of supervision policies and fault diagnosis purposes, it is necessary to incorporate some intelligence on this module. The Dispatcher Agent (DA) is then responsible for analyze and identify the sender (agent) of the alarm, before sending the output alarm message to the GAS.. This data processing is very crucial to the overall supervision robustness, providing the necessary information to the Multi-Agent Fault Diagnosis and Supervising System that is implemented in the upper-layer of this architecture. The DA will also receive the action events from the GAS and forwards these events to the corresponding LAMs.

3.4 Global Agent Supervisor

The Global Agent Supervisor (GAS) is in charge for supervising and managing all the WSAN agents. It is connected to a global agent database, or a catalogue, consisting of a backup of all existing agents in the WSAN nodes, which can be used to regenerate corrupted local agents, including LAMs. When a LAM is unable to regenerate a given corrupted local monitoring agent a regenerating request is sent to the GAS. This request is processed accordingly and a clone generated from the global multi-agent repository is sent to the gateway DA, which forwards it to the respective node. As a global manager, it has the authority to reconfigure all the local agents, depending on specific premisses. This is a major advantage of the distributed system, since it makes possible, over time, the reconfiguration and adaptability of nodes' functionalities and also enables the scalability of local agents' databases. Other functionality associated with the GAS is to manage different local managers for each node, by uploading, deleting or killing a given LAM, according to specific requirements.

4 Conclusions

A conceptual multi-layered and middleware-driven multi-agent architecture for WSAN applications is presented. The multi-agent approach is a valuable engineering solution to distributed systems, such as WSAN, offering the mobility, autonomy, and intelligence to sensor nodes. This is carried out through independent software modules that use sensors and actuators to interact with the environment. The proposed architecture takes into account the specificities and constraints of sensor networks, in order to reflect the specific needs of WSAN as a distributed system, in the context of faults' monitoring and fault tolerant control. The prototype of the architecture described in this paper is in development, in a real-environment test-bed, having already produced very attractive results, that will be released in future publications.

Acknowledgments. This work has been supported by the European Commission under the contract FP7-ICT-224282 (GINSENG). The author would like to acknowledge this support.

References

1. Mendes, M.J.G.C., Santos, B.M.S., da Costa, S.J.: Multi-agent Platform and Toolbox for Fault Tolerant Networked Control Systems. Journal of Computers 4(4), 303–310 (2009)
2. Ospina, P.A., Canõla, M.A., Carranza, O.D.: Integration Model of Mobile Intelligent Agents within Wireless Sensor Networks. In: IEEE Latin-American Conference on Communications, Medellín Colombia, pp. 1–6 (2009)
3. Tirkawi, F., Fischer, S.: Generality Challenges and Approaches in WSNs. I. J. Communications, Networks and System Sciences 1, 1–89 (2009)
4. Low, K.S., Win, W.N.N., Meng, J.E.: Wireless Sensor Networks for Industrial Environments. In: International Conference on Computational Modeling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies, and Internet Commerce, vol. 2, pp. 271–276 (2005)
5. Cerrada, M., Cardillo, J., Aguilar, J., Faneite, R.: Agents-based design for fault management systems in industrial processes. In: Computers in Industry, vol. 58, pp. 313–328. Elsevier, Amsterdam (2007)
6. Biswas, K.P., Qi, H., Xu, Y.: A Mobile-Agent Based Collaborative Framework for Sensor Network Applications. In: Proceedings of the Third IEEE International Conference on Mobile Ad-hoc and Sensor Systems, Vancouver, pp. 650–655 (2006)
7. Paolucci, M., Sacile, R.: Agent-based manufacturing and control system: new agile manufacturing. CRC Press LLC, Boca Raton (2005)
8. Bussman, S., Nicholas, R.J., Wooldridge, M.: Multi-Agent System for Manufacturing Control: A Design Methodology. Springer Science, Heidelberg (2004)
9. Buse, D.P., Wu, Q.H.: IP Networked-based Multi-Agent System for Industrial Automation – Information Management, Condition Monitoring and Control of Power Systems. Springer Science, Heidelberg (2007)
10. Cheyer, A., Martin, D.: The Open Agent Architecture. Journal of Autonomous Agents and Multi-Agent Systems 4(1), 143–148 (2001)
11. Braun, P., Wilhelm, R.: Mobile Agents-Basic Concepts, Mobility Models and The Tracy Toolkit. Elsevier Inc., Amsterdam (2005)
12. Xiang, H., Bin, L.: A Mobile-agent-based Role Switching Management Mechanism in WSN. In: The 2009 International Conference on Computational Intelligence and Software Engineering, Wuhan, China, pp. 1–4 (2009)
13. Chen, M., Kwon, T., Yuan, Y., Leung, M.C.V.: Mobile Agent Based Wireless Sensor Networks. Journal of Computers 1(1), 14–21 (2006)
14. Chiang, L., Russell, E., Braatz, R.: Fault Detection and Diagnosis in Industrial Systems. Springer, Heidelberg (2001)