A Case-Based Planning Mechanism for a Hardware-Embedded Reactive Agents Platform

Juan F. de Paz, Ricardo S. Alonso, and Dante I. Tapia

Computers and Automation Department, University of Salamanca. Plaza de la Merced, s/n, 37008, Spain
{fcofds,ralorin,dantetapia}@usal.es

Abstract. Wireless Sensor Networks is a key technology for gathering relevant information from different sources. In this sense, Multi-Agent Systems can facilitate the integration of heterogeneous sensor networks and expand the sensors' capabilities changing their behavior dynamically and personalizing their reactions. Both Wireless Sensor Networks and Multi-Agent Systems can be successfully applied to different management scenarios, such as logistics, supply chain or production. The Hardware-Embedded Reactive Agents (HERA) platform allows developing applications where agents are directly embedded in heterogeneous wireless sensor nodes with reduced computational resources. This paper presents the reasoning mechanism included in HERA to provide HERA Agents with Case-Based Planning features that allow solving problems considering past experiences.

Keywords. Wireless Sensor Networks, Multi-Agent Systems, Case-Based Planning, Service-Oriented Architectures.

1 Introduction

There are many situations where is useful to collect context information about users and their environment. Some of these situations include home automation, healthcare and also management scenarios, such as logistics, inventory, assembly lines and even offices. In this sense, Wireless Sensor Networks (WSNs) are widely used to obtain context information about users and their environment [1]. However, the integration of wireless devices coming from different technologies is not an easy task. Although there are plenty of technologies for implementing WSNs (e.g., ZigBee, Wi-Fi or Bluetooth), the lack of a common architecture may lead to additional costs due to the necessity of deploying non-transparent interconnection elements among different networks [2].

In this sense, the implementation of distributed architectures is presented as a solution to these problems [3]. One of the most prevalent alternatives in distributed architectures is Multi-Agent Systems (MAS), which can help to distribute resources and reduce the central unit tasks [4]. A distributed agent-based architecture provides more flexible ways to move functions to where actions are needed, thus obtaining better responses at execution time, autonomy, services continuity, and superior levels of flexibility and scalability than centralized architectures [5].

These are some of the main motivations for developing the HERA (*Hardware-Embedded Reactive Agents*) platform [6]. In HERA, unlike other approaches, agents are directly embedded on the WSN nodes, and their services (i.e., functionalities) can be invoked from other nodes in the same WSN or other WSN connected to the former one. HERA focuses specially on devices with reduced resources to save CPU time, memory size and power consumption.

This paper presents the mechanism included in HERA to provide HERA Agents with reasoning features. This reasoning mechanism is based on Case-Based Reasoning (CBR) [7] and Case-Based Planning (CBP) models, where problems are solved by using solutions to similar past problems [8]. Solutions are stored into a case memory, which the mechanism can consult to find better solutions for new problems. HERA Agents use this mechanism to learn from past experiences and to adapt their behavior according to the context information.

The rest of the paper is organized as follows. The following section presents the problem description that essentially motivated the development of HERA and introduces the CBR and CBP models. Then, it is depicted the main characteristics and components of HERA. After that, the CBP mechanism used in HERA is described. Finally, it is presented the conclusions and the future lines of work.

2 Background and Problem Description

HERA have stemmed from the necessity to cover more efficiently several of the challenges found on systems that use WSNs in their infrastructure. The fusion of the multi-agent technology and WSNs is not easy due to the difficulty in developing, debugging and testing distributed applications for devices with limited resources. The interfaces developed for these distributed applications are either too simple or, in some cases, do not even exist, which complicates even more their maintenance. Therefore, there are researches that develop methodologies for the systematic development of Muti-Agent Systems (MAS) for WSNs [9]. ActorNet [10] is a study that describes a mobile agent platform for WSNs. Implementing agent programs over WSNs is complicated due to the limitations of the sensor nodes, their limited memory, small bandwidth and low energy autonomy. ActorNet allows a wide range of dynamic applications, including customized queries and aggregation functions, in the sensor network platform. However, each mobile agent is only centered on a sensor node. Baker *et al.* [11] present the integration of an agent-based WSN within an existing MAS focused on condition monitoring. In this research, it is used SubSense, a multi-agent middleware platform developed to allow condition monitoring agents to be deployed onto a WSN. The architecture of the SubSense platform is based on the FIPA-defined model, but customized so that agents are embedded into sensor nodes. However, SubSense platform is not focused on working with heterogeneous WSNs and is implemented over 512KB RAM SunSPOT sensor nodes using the Java Mobile Edition (J2ME), while HERA platform can run on lightweight sensor nodes with just 8KB RAM. Furthermore, most of the works that relate Multi-Agent Systems and WSNs talk about Mobile Agents based on WSN (MAWSN). Zboril *et al.* [12] proposes WSageNt, a platform that is implemented through mobile agents running on wireless sensor nodes. WSageNt is supposed to be fault tolerant and not to be only focused on WSNs. However, it has not context-awareness features and does not contemplate the interconnection of heterogeneous WSNs.

Multi-Agent Systems can integrate Case-Based Reasoning (CBR) and Case-Based Planning (CBP) mechanisms [7] [8], which allow agents to make use of past experiences to create better plans and achieve their goals. These mechanisms provide the agents greater learning and adaptation capabilities. Case-based Reasoning (CBR) is a type of reasoning based on past experiences [7]. CBR mechanisms solve new problems by adapting solutions that have been used to solve similar problems in the past, and learn from each new experience. The primary concept when working with CBR mechanisms is the concept of case, which is described as a past experience composed of three elements: an initial state or problem description that is represented as a belief; a solution, which provides the sequence of actions carried out to solve the problem; and a final state, which is represented as a set of goals. CBR manages cases (past experiences) to solve new problems. The way cases are managed is known as the CBR cycle, and consists of four sequential phases: retrieve, reuse, revise and retain. CBP comes from CBR, but is specially designed to generate plans (sequence of actions) [8]. In CBP, the proposed solution for solving a given problem is a plan. This solution is generated by taking into account the plans applied for solving similar problems in the past. The problems and their corresponding plans are stored in a plans memory. The reasoning mechanism generates plans using past experiences and planning strategies, which is how the concept of Case-Based Planning is obtained [8]. CBP consists of four sequential stages similar to CBR stages. Problem description (initial state) and solution (situation when final state is achieved) are represented as beliefs, the final state as a goal (or set of goals), and the sequences of actions as plans. The CBP cycle is implemented through goals and plans. When the goal corresponding to one of the stages is triggered, different plans (algorithms) can be executed concurrently to achieve the goal or objective. Each plan can trigger new sub-goals and, consequently, cause the execution of new plans. CBR and CBP mechanisms have been successfully applied into Multi-Agent Systems in other research works to predict crises in business intelligence [7] or assign roles in virtual organizations of agents [13].

In this sense, HERA includes a Case-Based Planning (CBP) mechanism which allows the agents to make use of past experiences to create better plans and achieve their goals. HERA allows devices from different radio and networks technologies to coexist in the same distributed network. A totally distributed approach and the use of heterogeneous WSNs provide an architecture that expands the agents' capabilities to obtain information about the context and to automatically react over the environment.

3 The HERA Platform

HERA platform [6] facilitates the inclusion of agents into dynamic and self-adaptable heterogeneous WSNs. HERA is an evolution of the SYLPH (*Services laYers over Light PHysical devices*) platform [3] [14] and is focused specifically on devices with reduced resources to save CPU time, memory size and energy consumption. SYLPH allows the interconnection of several networks from different wireless technologies, such as ZigBee or Bluetooth. In this case, both WSNs are interconnected by a set of intermediate gateways (known as SYLPH Gateways [14]) simultaneously connected to several wireless interfaces. The underlying layers of HERA, provided by SYLPH, follow a Service-Oriented Architecture (SOA) model [2]. Unlike those approaches that integrate WSNs and MASs [9] [10] [11] [12], agents in HERA are directly embedded on the WSN nodes and their services can be invoked from other nodes in the same network or another network connected to the original one. Likewise, HERA can be executed over multiple wireless devices independently of their microcontroller or the programming language they use. This facilitates the development of context-aware capabilities into systems because developers can dynamically integrate and remove nodes on demand.

HERA implements an organization based on a stack of layers, shown in Figure 1 [6]. Each layer in a node communicates with its peer in another node through an established protocol. In addition, each layer offers specific functionalities to the immediately upper layer in the stack. The HERA layers are added over the SYLPH layers [14] and also the existent application layer of each WSN stack, allowing the platform to be reutilized over different technologies. From lowest to highest, SYLPH and HERA layers are described as follows:

- **SYLPH Message Layer (SML).** SML offers the upper layers the possibility of sending asynchronous messages between two nodes through the *SYLPH Services Protocol* (SSP). These messages specify the service invocation in a *SYLPH Services Definition Language* (SSDL) format.
- **SYLPH Services Directory Sub-layer (SSDS).** SSDS creates dynamical services tables to locate and register services in the network. A node that stores and maintains services tables is called *SYLPH Directory Node* (SDN). A node in the network can make a request to the SDN to know of a certain service.
- **SYLPH Application Layer (SAL).** SAL allows different nodes to directly communicate with each other using SSDL requests and responses that will be delivered in encapsulated SML messages following the SSP.
- **HERA Agents Layer (or just HERA)**. HERA agents are specifically intended to run on devices with reduced resources. To communicate with each other, HERA agents use HERACLES, the agent communication language designed for being used under the HERA platform. There must be at least one facilitator agent in every agent platform [4]. This agent is the first created in the platform and acts as a directory for searching agents. In HERA, the equivalent of these agents is the HERA-SDN (*HERA Spanned Directory Node*).

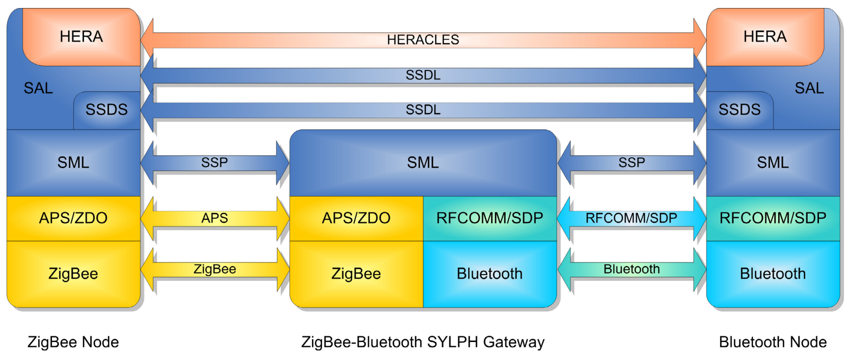

Fig. 1 Layers and protocols of the HERA platform over a heterogeneous Wireless Sensor Network

Similarly, the corresponding protocols used by HERA and SYLPH are:

- **SYLPH Services Protocol (SSP).** SSP is the internetworking protocol of the SYLPH platform. SSP allows sending packets of data from one node to another regardless of the WSN to which each one belongs.
- **SYLPH Services Definition Language (SSDL).** SSDL is the IDL (Interface Definition Language) used by SYLPH. Unlike other IDLs such as WSDL (Web Services Definition Language) [15], SSDL does not use as many intermediate separating tags, and the order of its elements is fixed [14].
- **HERA Communication Language Emphasized to Simplicity (HERACLES).** HERACLES language is directly based on the SSDL language. When developing a program, programmers use the human-readable representation to define agents' functionalities. However, HERA agents transmit the more compact representation of HERACLES as frames [6].

4 HERA Case-Based Planning Mechanism

As previously mentioned, some agents in HERA integrate a Case-Based Planning (CBP) mechanism. The CBP mechanism provides the agents with greater adaptation capabilities. As it is a complex and resources demanding task, the CBP mechanism has been modeled as a service provided by a special HERA Agent, known as HERA Planning Agent, which runs on a central node (i.e., a computer or a wireless device with moderate computational resources). The main characteristics of this mechanism are described in the remainder of this section.

The CBP mechanism needs a set of HERA Agents running on a set of nodes (i.e., wireless devices). Each of the nodes is connected physically to different sensors and actuators. This way, each node in the system can transmit commands to the actuators according to the sensors measurements. Each device d_i is defined by the sensors and actuators to which it has access, as expressed in (1).

$$d_i = (S_i, A_i) \tag{1}$$

where S_i is the set of sensors and A_i is the set of actuators.

According to the values read from the sensors, each HERA Agent running in the devices makes use of the actuators in order to achieve the required goal (e.g., stop a heater when a target temperature has been achieved). Thus, the behavior of the CBP mechanism is established by a database generated from the information of the sensors and actuators. This way, when some event produces an interaction with the sensors in the devices (e.g., a user action or a variation in the environment), these devices forward the values from the sensors and actuators to a central node that runs a special HERA Agent that stores this information. This agent is known as HERA Planning Agent. Depending on the size of the whole system (i.e., the number of nodes in the network, as well as the number of sensors and actuators associated to them), this central node can be implemented as a computer with a database stored in a physical disk or as a wireless sensor node with a smaller database stored in an EEPROM or Flash memory. Each device d_j has its own cases memory. Each case of the device d_j follows the structure indicated in (2).

$$c_i^{d_j} = (V_i^{S_j}, V_i^{A_j}) \tag{2}$$

where $V_i^{S_j}$ is the set of values from the sensors associated with the device *j*, and $V_i^{A_j}$ the values associated to the actuators.

The reactive behavior of the HERA Agents is defined as a set of rules that determine the relation among the sensors and the actuators. The rules are generated by a CBP mechanism. There are two kinds of rules: *static rules* and *dynamic rules*. Static rules are pre-defined rules that have priority over dynamic rules. Static rules determine the default behavior of each node and act also as a backup of these behaviors (i.e., they are stored directly in the nodes). Dynamic rules are automatically generated from the defined cases for each of the devices. The dynamic rules are periodically updated on every run of the HERA Agents (i.e., each time they are requested to execute a task). Figure 2 shows the functioning of the CBP mechanism when a device must execute a new task.

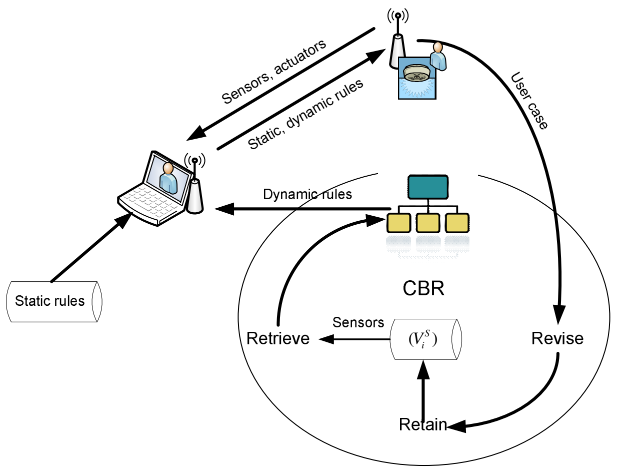

Fig. 2 Functioning of the CBP mechanism

Both static and dynamic rules are defined using the grammar shown as follows. This grammar facilitates the error detection and also the generation of native code that is actually running in the HERA Agents.

```
S::= Rules
Rules::= Rules Rule pyc |Rule pyc
Rule::= Precon then Actuator
Precon::= Precon and Sensor | Sensor
Sensor::= Ref relop Factor
Actuator::= Ref assop Factor
Factor::= nint | nflx
Ref::= id
```

In the shown grammar, the terms that start by uppercase characters correspond with non-terminal symbols (i.e., class of syntactically equivalent groupings), while the terms that start by lowercase characters correspond with terminal symbols. A non-terminal symbol is a symbol that is decomposed through a rule in the grammar, while a terminal symbol represents a symbol that is equivalent to a token. The tokens for each of the terminal nodes are defined as follows:

```
pyc=″;″
then=″then″
and=″and″
relop=″==″|″<=″|″>=″|″>″|″<″|″!=″
assop=″=″
nint= [0-9]+
nflx= ([0-9]+)"."([0-9]*)
Id=[A-Za-z][0-9A-Za-z]*
```

On the one hand, static rules are stored in the database by the HERA Planning Agent following the grammar previously established. An example of rules could be the following:

```
movement==0 then turnlight=0;
movement==1 and light==0.25 then turnlight=1;
```

As can be seen, rules are formed by sensors, literals, comparison and logical operators, as well as the final action to be performed by the actuator.

On the other hand, dynamic rules are defined by the CBP mechanism by means of the information of the cases shown in (2). During the recovery stage the CBP mechanism recovers the information with the sensors and actuators associated to a certain device. During the reutilization stage, automatic rules are generated from this information. For the generation of automatic rules, different algorithms based on decision rules and decision trees can be used. An example of algorithm based on decision rules is M5 [16], while J48 [17] is based on decision trees.

In HERA, the J48 algorithm is used for the generation of rules [17]. The inputs of the classifier are the value of the sensors, and the output belonging to the actuator. Using this configuration the J48 is trained, thus obtaining a decision tree that represents the behaviors of the actuators. In this sense, the different actuators are chosen as leaf nodes in the decision trees, while the sensor values are placed in the intermediate nodes. There is a decision tree according to the sensors situated in the device for each actuator. The devices can select a value of the actuators according to the value of the sensors through the decision tree, and the system only has to follow the conditions in the intermediate nodes until it arrives at a leaf node.

The final schema of the decision trees would be similar to that shown in Figure 3, and rules would be generated as indicated in the schema. The generated rules follow the grammar indicated above. The decision tree generated contains all the necessary information to generate the rules according to the grammar. The system only has to select each leaf node and move up toward the root node, introducing a new condition for each movement throughout the tree.

During the revision and learning stages, the system only stores the values of the actuators and sensors when the actuators are established manually by the users. The automatic decision of the agents are not stored in the system. The system only stores the interactions of the users as it tries to adapt to their behaviors. The system does not store generated cases as it is able to predict the behaviors, and only stores a new case when a user modifies the automatic configuration.

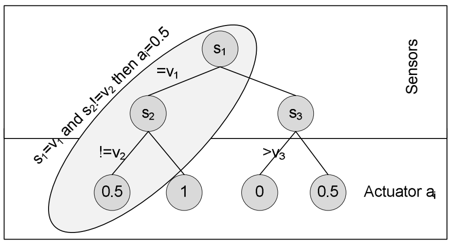

Fig. 3 Example of decision tree for the dynamic rules

For the development of the rules generator system the Weka libraries are used for the implementation of J48 algorithm. JFlex and Cup are used as lexical and syntactic analyzers. By means of Cup it is generated the native code that is then transferred and executed on chips.

5 Conclusions and Future Work

Wireless Sensor Networks provide a distributed and dynamic way to gather useful context information in a wide range of scenarios. These scenarios include business and managerial scenarios, such as inventory or assembly lines. Nevertheless, there are other situations where Wireless Sensor Networks can be successfully applied, such as home and building automation, healthcare or education.

The HERA platform allows wireless devices with reduced computational resources from different technologies to work together in a distributed way. The HERA CBP mechanism can adapt itself according to the values established by the user during its interaction with the platform. Furthermore, it facilitates the inclusion of new sensors dynamically, without need of performing offline training for each of the devices. In addition, the CBP mechanism can determine automatically the influence of the sensors to establish the final state of each of the actuators, so it is not necessary to indicate the relation among sensors and actuators.

Future work includes a experimentation stage to test the CBP mechanism and the publication of the results obtained. Furthermore, it will be designed a mechanism that will allow HERA agents to move throughout different nodes, no matter the WSN technology they use. This way, we will get, for example, HERA agents to move from a ZigBee node to a Bluetooth node through HERA, allowing the use of different programming languages and operating systems.

Acknowledgments. This project has been supported by the Spanish Ministry of Science and Technology project TIN 2009-13839-C03-03.

References

1. Marin-Perianu, M., Meratnia, N., Havinga, P., de Souza, L., Muller, J., Spiess, P., et al.: Decentralized Enterprise Systems: A Multiplatform Wireless Sensor Network Approach. IEEE Wireless Communications 14(6), 57–66 (2007)
2. Cerami, E.: Web Services Essentials: Distributed Applications with XML-RPC, SOAP, UDDI & WSDL, 1st edn. O'Reilly Media, Inc. (2002)
3. Corchado, J.M., Bajo, J., Tapia, D.I., Abraham, A.: Using Heterogeneous Wireless Sensor Networks in a Telemonitoring System for Healthcare. IEEE Transactions on Information Technology in Biomedicine 14, 234–240 (2010)
4. Wooldridge, M.: An Introduction to MultiAgent Systems, 2nd edn. Wiley (2009)
5. Sánchez, D., Isern, D., Rodríguez-Rozas, Á., Moreno, A.: Agent-based platform to support the execution of parallel tasks. Expert Systems with Applications 38, 6644–6656 (2011)
6. Tapia, D.I., Alonso, R.S., García, Ó., Corchado, J.M.: HERA: Hardware-Embedded Reactive Agents Platform. In: Pérez, J.B., Corchado, J.M., Moreno, M.N., Julián, V., Mathieu, P., Canada-Bago, J., Ortega, A., Caballero, A.F., et al. (eds.) Highlights in PAAMS. AISC, vol. 89, pp. 249–256. Springer, Heidelberg (2011)
7. Bajo, J., Borrajo, M.L., De Paz, J.F., Corchado, J.M., Pellicer, M.A.: A multi-agent system for web-based risk management in small and medium business. Expert Systems with Applications 39, 6921–6931 (2012)
8. Corchado, J.M., Bajo, J., de Paz, Y., Tapia, D.I.: Intelligent environment for monitoring Alzheimer patients, agent technology for health care. Decision Support Systems 44, 382–396 (2008)
9. Tynan, R., O'Hare, G., Ruzzelli, A.: Multi-Agent System Methodology for Wireless, Sensor Networks. Multiagent and Grid Systems 2, 491–503 (2006)
10. Kwon, Y., Sundresh, S., Mechitov, K., Agha, G.: ActorNet: an actor platform for wireless sensor networks. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 1297–1300. ACM, Japan (2006)
11. Baker, P., Catterson, V., McArthur, S.: Integrating an Agent-Based Wireless Sensor Network within an Existing Multi-Agent Condition Monitoring System. In: 15th International Conference on Intelligent System Applications to Power Systems, ISAP 2009, pp. 1–6 (2009)

12. Zboril, F., Horacek, J., Spacil, P.: Intelligent Agent Platform and Control Language for Wireless Sensor Networks. In: Third UK Sim European Symposium on Computer Modeling and Simulation, EMS 2009, pp. 482–487 (2009)
13. Zato, C., De Paz, J.F., de Luis, A., Bajo, J., Corchado, J.M.: Model for assigning roles automatically in egovernment virtual organizations. Expert Systems with Applications (2012) (in Press), doi:10.1016/j.eswa.2012.01.185
14. Tapia, D.I., Alonso, R.S., De la Prieta, F., Zato, C., Rodriguez, S., Corchado, E., Bajo, J., Corchado, J.M.: SYLPH: An Ambient Intelligence based platform for integrating heterogeneous Wireless Sensor Networks. In: 2010 IEEE International Conference on Fuzzy Systems (FUZZ), pp. 1–8 (2010)
15. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. IEEE Internet Comput. 6, 86–93 (2002)
16. Holmes, G., Hall, M., Frank, E.: Generating Rule Sets from Model Trees. In: Proc. of the 12th Australian Joint Conf. on Artificial Intelligence, pp. 1–12 (1999)
17. Salzberg, S.L.: C4.5: Programs for Machine Learning by J. Ross Quinlan. In: Machine Learning, vol. 16, pp. 235–240. Morgan Kaufmann Publishers, Inc. (1994)