Autonomic Behaviors in an Ambient Intelligence System

Article

Accepted version

A. De Paola, P. Ferraro, S. Gaglio, G. Lo Re

In Proceedings of the IEEE 2014 Symposium on Computational Intelligence for Human-like Intelligence (SSCI 2014)

Autonomic Behaviors in an Ambient Intelligence System

Alessandra De Paola*, Pierluca Ferraro*, Salvatore Gaglio*† and Giuseppe Lo Re*
*University of Palermo, Viale delle Scienze, ed. 6 90128 Palermo, Italy
†ICAR-CNR, National Research Council of Italy, Viale delle Scienze, ed. 11, 90128 Palermo, Italy
{alessandra.depaola, pierluca.ferraro, salvatore.gaglio, giuseppe.lore}@unipa.it

***Abstract*—Ambient Intelligence (AmI) systems are constantly evolving and becoming ever more complex, so it is increasingly difficult to design and develop them successfully. Moreover, because of the complexity of an AmI system as a whole, it is not always easy for developers to predict its behavior in the event of unforeseen circumstances. A possible solution to this problem might lie in delegating certain decisions to the machines themselves, making them more autonomous and able to self-configure and self-manage, in line with the paradigm of Autonomic Computing. In this regard, many researchers have emphasized the importance of adaptability in building agents that are suitable to operate in real-world environments, which are characterized by a high degree of uncertainty. In the light of these considerations, we propose a multi-tier architecture for an autonomic AmI system capable of analyzing itself and its monitoring processes, and consequently of managing and reconfiguring its own sub-modules to better satisfy users' needs. To achieve such a degree of autonomy and self-awareness, our AmI system exploits the knowledge contained in an ontology that formally describes the environment it operates in, as well as the structure of the system itself.**

I. Introduction

Ambient Intelligence (AmI) is an application paradigm of Artificial Intelligence, which focuses on users and their needs, with the aim of designing intelligent pervasive systems capable of providing the best possible environmental conditions for end-users. In order to achieve such objectives, small intelligent devices are pervasively deployed in the environment, and, by exploiting Artificial Intelligence techniques, they are programmed to act autonomously or in a collaborative approach to learn and satisfy user preferences [1]. The goal is to create an intelligent environment which feels natural and with which people can interact easily and effortlessly [2], [3]. However, traditional programming techniques may not be able to cope with the AmI paradigm [4], which require more autonomous systems that can manage and reconfigure themselves, thereby freeing designers from these demanding tasks. Thus, self-configuring may well be the key to building truly autonomous and self-aware agents, capable of acting in an uncertain world, as suggested by a body of research carried out in the field of Artificial Consciousness in recent years [5].

Within such a scenario, we propose a multi-layer cognitive architecture for an autonomic AmI system, which expands on previous work [6] and is characterized by a sense of self-awareness, incorporating some aspects of introspection. The work presented in [6] described a multi-layer architecture inspired by the structure of the human nervous system [7], but lacked introspection and autonomic behaviors. In this paper, our goal is to develop an adaptive system inspired by features such as self-awareness, embodiment, situatedness, integration, attention and self-reasoning. It is thought advisable that even machines and software agents should exhibit these properties, so as to be more autonomous, thereby reducing operating costs as well as improving the performance of the system [4].

With these considerations in mind, the introspective behavior of our system is achieved by continuous self-modeling and self-monitoring activities, performed according to the Autonomic Computing paradigm. That is, a truly autonomous agent should rely on an explicit model of itself, of the environment in which it operates, and of its interactions with users. Exploiting such extensive knowledge, the system is thus able to understand the state of the environment and user preferences, as well as using this knowledge to decide what actions should be performed next. Furthermore, the internal model of the agent and the model of the external world should be described using a common conceptualization, so as to simplify introspection and self-reasoning [8].

To this end, we decided to exploit ontologies to model the domain knowledge in a unified and machine-computable format, in order to drive the process of knowledge abstraction from raw sensory data up to higher-level concepts. In this way, the system is better able to understand the interactions between its cognitive subsystem, its physical subsystem (i.e., sensors and actuators) and the environment, thus acting in an appropriate manner to achieve its goals. By exploiting a rule-based inference engine to analyze the ontology and reason about its internal structure, the system is thus able to self-configure its sub-modules and modify its own behavior at runtime, in order to minimize energy consumption.

This paper presents the implementation of our proposal in the context of a Building Management System (BMS) capable of handling environmental conditions, such as temperature, humidity and lighting in an office environment, with the twofold goal of maximizing user comfort and minimizing the energy consumption of sensors and actuators.

The remainder of this paper is organized as follows. Section II analyzes some relevant approaches in the fields of adaptive behavior and Autonomic Computing. Section III introduces the paradigm of Ambient Intelligence and presents Sensor9k, a testbed designed to facilitate reasoning about user comfort and energy saving in an office environment. Section IV outlines the general architecture of the system proposed here, highlighting the aspects that ensure its adaptivity and context-awareness. Section V describes the proposed ontology, which defines the concepts used by the system, that relate to the environment, the user and its interactions with it. Section VI gives an outline of the rule-based inference engine, which allows our system to configure and manage itself. Finally, Section VII presents our conclusions.

II. Related Work

In the field of Ambient Intelligence, as shown by [9], it is very difficult, if not practically impossible, to build accurate mathematical models capable of grasping the complexity and dynamism of the real world. Therefore, it is necessary to adopt advanced techniques of Artificial Intelligence to manage the multitude of devices placed in the environment and build a layer of distributed intelligence able to handle the uncertainty inherent in the data collected. Some of the most common AI techniques used in previous research to study the behavior of users and learn their preferences have been Bayesian networks [10], fuzzy systems [11] and neural networks [12].

In such a scenario, the Autonomic Computing paradigm might be the key to developing truly adaptive and self-aware agents [13]. The paradigm was originally proposed by IBM, in analogy to the autonomic nervous system [14], which allows the human body to maintain the balance between all of its complex subsystems, responding to unpredictable external stimuli and overcoming the dangers caused by external agents.

Similarly, computational architectures inspired by such a model are composed of autonomic elements that are able to respond to changes autonomously, adapting to the environment in order to better achieve their goals. A self-configuring and self-managing AmI system should therefore include and analyze knowledge about itself and the environment in which it acts, so as to constantly update its model of the world [15]. By describing concepts from both domains in a machine-computable way [8], the agent is able to unify the mechanisms that it uses to reason about the world and about itself, exploiting the semantic enrichment of processed data.

One issue on which many researchers agree is that an autonomous agent should not be limited by fixed criteria, because real environments are characterized by uncertainty, and not all contingencies can be foreseen at design time. In this regard, a key aspect of every autonomous and self-aware agent is its capacity for adaptive decision-making [16], which allows it to make correct decisions based on the current situation. That is, it improves its context-awareness so as to achieve its goals in the best possible way. Several researchers use concepts such as agency [17] or cognition [18], [19] to define a set of conditions that an agent must have in order to be considered truly autonomous and capable of "minimally cognitive behavior" [20]. The function of cognition, in particular, is considered a key feature for improving the capabilities of perception, learning, memory and decision-making of every autonomous agent [21], and thus constitutes a prerequisite to developing systems that are truly context-aware and adaptive. Research into the relationship between the agent and the environment has also focused on other important features of autonomous machines, including situatedness, embodiment and attention [22], [23].

To implement these paradigms, various approaches have been proposed in the literature. For example, the authors of [24] propose an architecture based on neural networks that allows a robot to learn complex sequences of actions so as to coordinate vision and arms movements. Others have used genetic algorithms [25], Bayesian networks [26] and reinforcement learning techniques such as Q-learning [27] to build robust agents capable of operating in an uncertain world.

III. Ambient Intelligence and BMS

Ambient Intelligence envisages innovative scenarios in which intelligent systems assist human beings in their daily activities, but without the inconvenience of intrusive technologies which, in this paradigm, remain confined to the background [3].

AmI techniques are currently used in different contexts such as care for the elderly, medical assistance, kindergartens, farm automation, the car industry and Building Management Systems (BMSs). Our work focuses on the design of intelligent BMSs, whose main goal is the control of environmental conditions in buildings (e.g., temperature, humidity and lighting), in order to satisfy user requirements and minimize energy consumption. In order to accomplish their goals, BMSs need a sensory and actuator infrastructure constituting their direct link to the real world. Sensors acquire environmental data (e.g., temperature, light intensity, noise, humidity, etc.) and context information (e.g., user presence and user activities), whereas the actuator infrastructure will consist of all those physical devices in the building that can influence the state of the environment (e.g., artificial lighting systems and heating, ventilation and air conditioning (HVAC) systems).

One fundamental requirement of AmI systems is the low intrusiveness of the underlying technology. This means that the sensory and actuator infrastructure have to be characterized by a low degree of physical invasiveness. Moreover, its management should require little interaction with the user, the requirement which inspired the proposed autonomic behavior.

In this paper, we took as our reference the BMS architecture proposed in Sensor9k [28], a pervasive testbed whose aim is to support the development of energy-aware Ambient Intelligence systems. The name of the testbed recalls the fictional *HAL 9000* artificial intelligent system, whose sensory and actuator terminations permeated the spaceship in "2001: A Space Odyssey". The system proposed in this article represents a further step towards this visionary target, paving the way for the design of systems aware of their own physicality and cognitive processes. The physical infrastructure of Sensor9k exploits the technology of Wireless Sensor and Actuator Networks (WSANs) [29]. These networks are composed of a set of small devices, called *nodes*, which are in most cases energetically autonomous, programmable, and able to perform

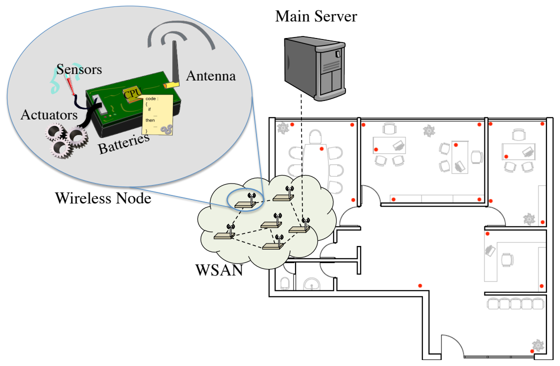

Fig. 1. High-level scheme of a Wireless Sensor and Actuator Network (WSAN) installed in an office environment.

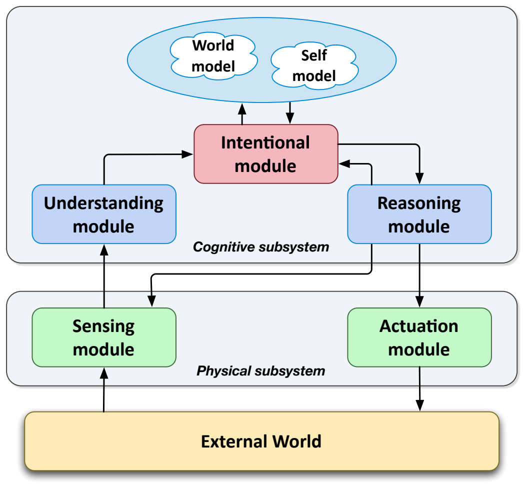

Fig. 2. Architecture of the autonomic Ambient Intelligence system.

small computations on boards and to wirelessly communicate with each other. Each node can be equipped with different sensors and can control certain connected actuators.

In our settings, sensors installed on nodes make it possible to gather information about the environmental state and about the context, as well as information about the state of the node itself, such as, for instance, residual battery energy. Furthermore, by exploiting a set of ad-hoc sensors, it is possible to monitor the energy consumption of all the electrical appliances installed in the monitored environment.

Appliances are controlled by the actuators installed on sensor nodes, by means of the transmission of opportune control signals. Moreover, Sensor9k also lets the user manually control the available appliances, and apposite sensors capture each of these interactions. The simpler actuators provided by Sensor9k are represented by remotely controllable power relays enriched with the capability of providing information about their current state (e.g., the artificial lighting relay controller). More complex forms of actuators are those controlling domestic appliances (typically by means of IR remotes) typically found in homes or offices, such as HVAC systems.

The sensory information acquired by the WSANs, through the hierarchical communication infrastructure provided by Sensor9k, is sent to a centralized server, which hosts the reasoning components. The choice of centralizing reasoning activity makes it possible to preserve its consistency and uniqueness [30], even in the presence of a distributed and pervasive physical layer. A high-level scheme of the Sensor9k's WSAN for an office environment is shown in Fig. 1.

Furthermore, the middleware provided by Sensor9k enables dynamic tuning of the sensors' behavior by means of control messages sent at runtime. By means of these control messages it is possible, for instance, to tune the sampling rate of sensor nodes, or to switch nodes to a low-power mode in order to minimize their energy consumption. In order to reduce the sensory infrastructure handling effort for the user, programmability is also exploited by the adoption of replicas of critical WSAN devices, thereby guaranteeing fault tolerance and robustness. The whole sensory infrastructure comprises a set of fully active sensor nodes and a complementary set of duplicate nodes. Replica nodes are started in low-power mode with data gathering and transmission functionalities disabled. Listening to control messages is the only active functionality in low-power mode. A specific control message can switch a node from this condition to normal functioning mode in order to activate ambient monitoring. The adaptive system exploits such features in order to resume its full operability when a given node switches into fault condition.

IV. Architecture Design

The architecture proposed in this paper is inspired by the paradigms of Ambient Intelligence and Autonomic Computing. Our goal is to design an AmI system that is able to self-configure and self-manage, by taking advantage of certain intrinsic features of human beings, such as introspection and self-awareness, to improve the adaptivity of the resulting agent. Using the information represented by the ontologies and instantiated in the knowledge base of the system, the agent can then monitor the status of its own sub-modules, dynamically adapting its behavior to the context, so as to best suit the users' requirements, according to the high-level policies defined by administrators.

To achieve this result, the system must be able to aggregate the raw data fed to it by its own sensory infrastructure, integrating heterogeneous and distributed information to get a unified view of the current situation and efficiently plan the actions that must be performed to satisfy the needs of the users. Although the Artificial Intelligence literature includes several techniques for representing the knowledge necessary for the functioning of such a system [31], [32], the amount of data collected by sensors in a realistic setting is too great, and would lead to serious problems in terms of computational efficiency.

For this reason, our system filters information from sensors, by using modules that operate at different levels of abstraction and data mining techniques for the efficient fusion, modeling and interpretation of these data. As shown in Fig. 2, the architecture we propose here consists of five main modules, each of which has a highly flexible and configurable structure:

- a *Sensing module*, based on Wireless Sensor Networks (WSNs), which is responsible for collecting the raw sensory data about the relevant features of the surrounding environment and sending that information to higher levels;
- an *Understanding module* which processes sensory data, aggregating and representing them in gradually increasing levels of abstraction, so as to describe the environment through high-level concepts, which summarize in a succinct way the huge amount of data extracted by sensors;
- an *Intentional module* that directly accesses the ontological models of the environment and of itself, and updates them according to the data collected, deciding the short-term goals of the system in compliance with a high-level policy;
- a *Reasoning module*, which plans the most appropriate sequence of actions that the system must perform to achieve the goals set by the Intentional module, so as to better satisfy user needs;
- an *Actuation module* that makes it possible to change the status of the environment, by operating heating, ventilation, air conditioning (HVAC) and lighting systems, as well as other elements that affect the environmental parameters.

The key element of the architecture is the feedback loop, which requires that the system is aware of itself, of the surrounding environment and of their interactions with each other. This allows the system to keep its world model and self model constantly updated, based on the interactions between the physical subsystem, the cognitive subsystem and the external world.

In other words, by using an appropriate sensory infrastructure, continuous monitoring allows the system to identify relevant events concerning the users and the environment, so as to start the reasoning and planning processes, and then initiate the actions that are carried out by means of suitable actuators. In addition, each of the main modules consists of sub-modules performing specific tasks within the feedback loop. The behavior of the system as a whole emerges from the interaction and collaboration of these subsystems.

As shown in Fig. 3, the Reasoning module is organized into three different sub-modules, named *Planner*, *Self-reasoning* and *World-reasoning*. The Understanding module is also divided into three tiers, namely the *Subsymbolic*, *Conceptual* and *Symbolic* tiers.

The sub-modules implementation can vary depending on how the architecture is instantiated. For example, as regards the representation of sensory measurements, the Subsymbolic tier is responsible for pre-processing the data coming from sensors, by uniforming the sampling rate and eliminating outliers. The Conceptual module is an intermediate tier between the Subsymbolic and the Symbolic levels, similarly to what has been proposed in [33], [34]. This tier uses appropriate clustering and classification techniques in order to identify fuzzy concepts which are then represented in the Symbolic level.

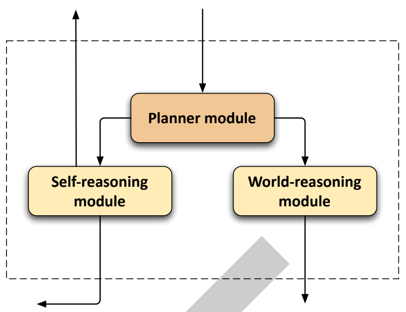

Fig. 3. Details of the Reasoning sub-module.

Fig. 4 shows the data flow among the modules, highlighting the process of knowledge abstraction from raw sensory data up to higher-level symbolic data.

The basic sensors deployed in the monitored premises represent the input devices of the system, and their collected data are sent to the Understanding module. In addition, the nodes of the Wireless Sensor Network send information about their status, thus allowing the AmI system to achieve full self-awareness and to reason about its own sub-modules. Symbolic data constitute the input of the Intentional module, which updates the world and self models according to the data collected, planning the short-term goals of the system and triggering the Reasoning module and its sub-modules, which carry out the reasoning and self-reasoning activities. Finally, in order to close the loop, the World-reasoning module sends action commands to the actuators, while nodes and sensors receive configuration commands from the Self-reasoning module.

It is important to note that the Self-reasoning and the World-reasoning modules can be implemented using different Artificial Intelligence approaches. For example, in the solution described here, the Self-reasoning module is a rule-based system, while the World-reasoning module is based on fuzzy logic.

V. Ontology-based Self Modeling

In order to dynamically adapt its behavior to the context where it is running, an autonomous agent should possess an explicit model of itself, of the surrounding environment, and of the ways it can interact with users [35]. Therefore, the system has to manage a large amount of information that may change quickly and often. In order to represent such knowledge in an efficient and machine-computable way, then, it is advisable to use ontologies, which are documents that formally define the relationships among a set of terms belonging to a specific domain [36].

Our ontology, written in the OWL 2 language [37], provides a uniform terminology to describe the environment and its properties, the user and his interactions with it, as well as the system components and their interconnections. Furthermore, it describes how data flows within the system, highlighting the

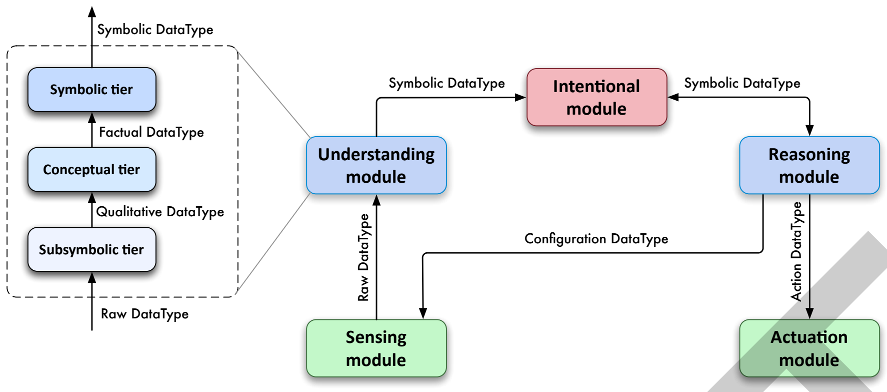

Fig. 4. Data types used as inputs and outputs of the different sub-modules.

relationships between the sensors and the monitored environmental properties, and accomplishes several objectives:

- it represents in a formal and machine-computable way the relationships between concepts belonging to the domain of interest;
- it enables the system to dynamically and autonomously reconfigure itself, according to the goals defined by the Intentional module;
- it allows system administrators to define high-level policies affecting the overall behavior of the system;
- it makes it easy to represent the rules needed to infer appropriate control actions, taking into account the status of the environment and the current goals of the system;
- it provides the system with some form of self-awareness, which in turn allows it to better understand its relationships with the environment.

We adopted a semantic representation that is carried out by two different ontologies. The first describes the structure of a generic building management system, and can be easily reused for several applications, since it is not tied to a specific scenario. The second ontology extends the more general one by defining subclasses and individuals, which describe a specific instance of our application scenario, namely the BMS for controlling ambient conditions in an office environment.

The ontological description provides important information about sensors, such as energy consumption, sampling rate, continuity of monitoring and the node on which the sensor is installed. Each device perceives a specific environmental attribute, or acts upon it. These properties can be related to physical phenomena such as light and temperature, observable events like user activity, or the status of a particular environmental element, such as a window or an air cooler.

To represent the system architecture within the domain ontology, we first extended the taxonomic organization defined in the top-level one with new subclasses, and then instantiated the actual modules as individuals of these subclasses, so as to reflect the topological and semantic organization of the monitored environment. In contrast to the top-level ontology, in fact, the domain ontology has to be modified according to the environment, with the involvement of a domain expert. If the structure of the system changes, the domain ontology must be updated manually.

As an example of such a process, the domain ontology particularizes a class of sub-symbolic modules capable of processing the temperature information, and this class is instantiated in a set of specific individuals. Each of these individuals is devoted to the monitoring of a given managed room. The same process is followed for all other monitored environmental properties, such as humidity and lighting.

VI. Rule-based Reasoning

Our implementation of the Autonomic Computing paradigm is based on the classic *monitor-analyze-plan-execute* cycle, which underlies an intelligent control loop [14], as explained above in the description of the data flow within the sub-modules constituting the AmI system. The self-reasoning module uses rule-based reasoning to reconfigure the sensors on the basis of certain parameters, such as the user's presence, the degree of accuracy of the monitored information, the state of the sensors, their energy consumption, and the residual lifetime of the battery-powered sensor nodes. A rule-based approach has been successfully adopted for management tasks in several distributed systems [38], [39], but this choice is not necessarily imposed to the other components of the reasoning module, as explained in Section IV.

Using the knowledge contained in the ontology, the AmI system is able to understand which sub-modules are needed to infer certain concepts, and identifies the relationships between the environmental properties of interest and the specific sensors that perceive them. The Self-reasoning module, then, exploits a set of rules in order to infer new evidence and to plan the sequence of actions that the system must perform. Finally, at the end of the planning process, the resulting configuration commands are sent to the sensors.

In order to perform automated reasoning on the domain and to infer new evidence from the available information, it is necessary to employ an inference engine capable of processing ontologies and their logical rules. In particular, to implement the operations of reasoning and self-reasoning, we adopted

Jess (Java Expert System Shell) [40], a rule-based inference engine developed in Java.

Jess makes it possible to express logical rules with a LISP similar syntax, and uses a pattern-matching algorithm to query the knowledge base and extract the information required by the system. Each fact contained within the knowledge base is a true proposition about the world or the system itself, and belongs to a template, just as every object is a member of a class in the object-oriented programming style.

Jess templates define the name of a fact, the properties that a fact possesses and, optionally, the corresponding range of values that the properties may assume. Four types of templates are used in our system:

- *static knowledge* templates, representing the initial facts known by the reasoner during the setup of the system, such as the locations of nodes and sensors;
- *dynamic knowledge* templates, relating to information that is continuously updated at runtime, such as the status of sensors and actuators or the presence of users;
- templates representing *action commands* for the actuators and *configuration commands* for the sensors;
- templates relating to the *alerts* that are sent to administrators when the self-monitoring modules report any inadequacy of the system.

Besides generating configuration commands for sensors, the Self-reasoning module also yields alerts for system administrators, such as insufficient sensing capability or short sensing time. This latter event occurs if all the sensors capable of measuring a particular environmental phenomenon are battery-powered, and currently register a low level of residual energy.

Jess allows an easy implementation of rules that produce new knowledge, starting from the information already acquired. Each rule takes the form of an "*if <conditions> then*" construct, and it is activated when all its conditions are satisfied, that is, when the knowledge base contains facts that match all the antecedents of the rule. Rules are executed only once for any given set of facts, and are reconsidered only after the insertion of new facts within the knowledge base.

In order to minimize energy consumption, the system dynamically decides whether and how to change the sampling rate of sensors, depending on the current value of a property called *system-condition*, that describes the state of a room. Considering such a property as input, the system adapts to the current situation by changing its configuration, its behavior and its goals so as to focus on the most important aspects of the external environment. In particular, each condition is associated with a different set of rules, so the system responds differently to the same external stimuli depending on its current condition. In this way, limited resources such as residual energy in battery-powered nodes can be managed efficiently, minimizing energy consumption and increasing the survivability of the whole system. The finite state machine in Fig. 5 illustrates the possible values of the *system-condition* property and the transitions between states.

A *stress* condition occurs when the amount of energy currently used is no longer sustainable, i.e., when all the sensors devoted to monitoring a given environmental variable are installed on sensor nodes that are battery-powered, and such devices can guarantee a low level of residual energy.

When the system falls into a *stress* condition, the Reasoning module activates a set of special rules in order to minimize energy consumption and put the system back into a condition of *normal* functioning, indicating balanced behavior, and improving the chances of meeting both the short-term and long-term goals of the system. Finally, the *attentive* condition is triggered when users are occupying the monitored premises. In this state, the system gives priority to user comfort, activating a set of special rules in order to maximize the accuracy of monitoring. Some rules are described below which govern state transitions. For example, when the sensory infrastructure signals that the user is not present in his office, and the knowledge base contains at least one statement of *short-sensing-life* for a sensor node in the same room, the following rule places the system into the *stress* condition:

```
defrule setStressCondition:
  if (no user is in room $R$) and ($\exists$ "low" battery node in $R$) then
    system-condition $\leftarrow$ "stress"
```

For each environmental property, the following rule checks that there exists at least one sensor node with a sufficient residual charge. If this is not the case, the rule sends a *short-sensing-life* alert to administrators:

```
defrule computeSensingLife:
  if (all sensors that monitor ambient property $AP$ are installed on
      "low" battery nodes) then
    send short-sensing-life alert to administrators
```

More specifically, the system finds all the sensors that monitor a given property within a room. If all of these sensors are installed on battery-powered nodes, and if all these nodes have a *low* level of residual energy, the system sends an alert to administrators, warning them in a proactive way.

The *minimizeRoomSensing* rule makes it possible to infer that, if the system is in a *stress* condition, it is necessary to minimize the monitoring activities for all the measured quantities in the room in question.

```
defrule minimizeRoomSensing:
  if (system-condition is "stress") then
    for each sensor $S$ in room $R$ do
      set minimum sampling rate for $S$
```

The previous rule sets the sampling rate of all the sensors installed in the room at the minimum value, thus minimizing energy consumption. This is an example of a rule that is only triggered when the system is in a particular condition, thus improving adaptability and context awareness.

The *attentive* condition is accomplished when the following rule succeeds, triggered by a user entering an office. As shown by the finite state machine in Fig. 5, this condition has the highest priority. That is, the system switches to the *attentive* condition when the user enters his office, and remains in this state until the user leaves, regardless of other factors, thus prioritizing the accuracy of monitoring.

defrule setAttentiveCondition:

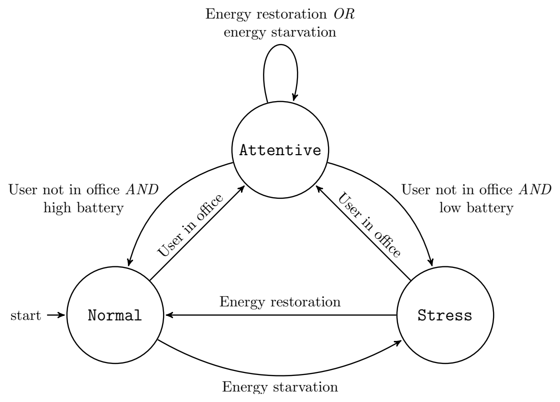
Energy restoration OR energy starvation
Attentive
User not in office AND high battery
User in office
User in office
User not in office AND low battery
start
Normal
Energy restoration
Stress
Energy starvation

Fig. 5. Finite state machine showing the transitions between system conditions.

```
if (user $U$ is in room $R$) then
    system-condition ← "attentive"
```

When in such a condition, the system triggers a set of rules that increase the sampling rate of sensors in order to better monitor the environment and satisfy users' needs. For instance, the *increaseRoomSensing* rule detailed below increases the monitoring rate of any ambient property characterized by a *low* level of accuracy. The rule is triggered only if the system is in the *attentive* condition, and hence, indirectly, if a user is in his office.

```
defrule increaseRoomSensing:
  if (system-condition is "attentive") and
      ($\exists$ "low" accuracy ambient property $AP$ in room $R$) then
    increase sampling rate for $AP$ in $R$
```

The rule increases the sampling rate of the sensors, so as to maximize the accuracy of monitoring. If the sampling rates of all available sensors are already at their maximum value, the rule sends an *insufficient-sensing* alert to administrators, informing them that the sensory infrastructure of the system can not adequately achieve the goals set by the reasoner.

The *setNormalCondition* rule puts the system into the *normal* condition, being triggered when the user is not present in his office and the knowledge base does not contain *short-sensing-life* statements related to the current room:

```
defrule setNormalCondition:
  if (no user is in room $R$) and ($\nexists$ "low" battery node in $R$) then
    system-condition ← "normal"
```

Finally, in cases where any active node consumes all the residual energy, the *activateBackupNode* rule leverages the redundant sensory infrastructure of the WSAN, as described in Section III, to enable the backup sensor nodes to operate. In particular, the rule searches all sensor nodes with a *very low* battery level and tries to activate the corresponding backup nodes. If the operation is successful, the system can delegate the gathering and transmission of sensory measurements to the backup nodes, which then become fully active and replace the original nodes in all respects.

```
defrule activateBackupNode:
  if ($\exists$ "very low" battery node $N$ in room $R$) then
    activate backup node of $N$
```

The purpose of this special rule is to restore the system from a *stress* condition to a *normal* one, thus implementing the self-healing capabilities required by an autonomic system. If the activation of the backup nodes is not successful, the system signals a *discharged-battery* alert to administrators, warning them about the energy shortage and urging the replacement of the uncharged batteries.

Considering that the activation of the backup nodes increases the survivability of the system and does not involve a decrease in monitoring accuracy, the *activateBackupNode* rule is independent of the active *condition*, and can therefore be activated even if the user is present in his office and the system is currently *attentive*.

It is worth noting that the rules described here are static and there is no mechanism for learning them automatically. The adaptivity of the system lies in the ability to dynamically modify the state of its sensory infrastructure. Nevertheless, the proposed rules are generic and do not depend on a particular instance of the system. Moreover, they implicitly manage the conflicting goals of maximizing sensing accuracy and minimizing energy consumption.

VII. Conclusions and Future Work

This paper describes a multi-tier cognitive architecture that envisages the construction of an autonomous AmI system, capable of self-configuring and self-managing its sensory infrastructure. It self-adapts to continuous changes in the environment for an optimal fulfillment of its own goals. Due to the great complexity of such systems, their overall design and specific deployment is becoming increasingly difficult for human developers. These systems therefore need to be developed according to the paradigm of Autonomic Computing.

For this reason, the architecture we propose allows for the representation of heterogeneous sensory information received as input. Moreover, the entire hardware and software system is represented in increasing levels of abstraction, so as to describe the environment and interactions with the system through high-level symbolic concepts. The system is thus capable of focusing on those aspects of the external environment considered the most important at different times, depending on the context, and thus to manage its resources intelligently and minimize energy consumption. The ontological representation adopted here permits prompt modeling and formalization of the knowledge required to plan the action sequences to be performed by a rule-based inference engine in order to interact with the environment.

The self-reasoning module is seamlessly integrated into the system architecture, thus allowing the autonomic paradigm to handle the complexity of the external real-world environment. The system is also able to manage conflicting objectives such as maximizing the sensing accuracy and minimizing the energy consumption of sensors and actuators. As future development, we are interested in studying how to automatically learn and update the rules used by the Self-reasoning module, so as to make the system even more adaptive. Finally, the use of meta-programming and reflection techniques would allow the system to self-instantiate only on the basis of the ontological description provided by the administrators, greatly simplifying the stages of development and setup of the system.

Acknowledgment

This work has been partially supported by the PO FESR 2007/2013 grant G73F11000130004 funding the SmartBuildings project.

References

[1] D. Cook, J. Augusto, and V. Jakkula, "Ambient Intelligence: Technologies, Applications, and Opportunities," *Pervasive and Mobile Computing*, vol. 5, no. 4, pp. 277–298, 2009.

[2] K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J.-C. Burgelman, *Scenarios for Ambient Intelligence in 2010*. Office for official publications of the European Communities, 2001.

[3] P. Remagnino and G. Foresti, "Ambient Intelligence: A New Multidisciplinary Paradigm," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 35, no. 1, pp. 1–6, 2005.

[4] R. Sanz, I. López, and C. Hernández, "Self-awareness in real-time cognitive control architectures," *AI and Consciousness: Theoretical Foundations and Current Approaches*, 2007.

[5] I. Aleksander, "The potential impact of Machine Consciousness in science and engineering," *International Journal of Machine Consciousness*, vol. 1, no. 01, pp. 1–9, 2009.

[6] A. De Paola, S. Gaglio, G. Lo Re, and M. Ortolani, "An ambient intelligence architecture for extracting knowledge from distributed sensors," in *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, 2009, pp. 104–109.

[7] A. De Paola, A. Farruggia, S. Gaglio, G. Lo Re, and M. Ortolani, "Exploiting the Human Factor in a WSN-Based System for Ambient Intelligence," in *International Conference on Complex, Intelligent and Software Intensive Systems, 2009. CISIS '09*, 2009, pp. 748–753.

[8] J. Bermejo-Alonso, R. Sanz, M. Rodríguez, and C. Hernández, "Ontology-based engineering of autonomous systems," in *Proceedings of the 6th International Conference on Autonomic and Autonomous Systems (ICAS)*. IEEE, 2010, pp. 47–51.

[9] H. Hagras, "Embedding computational intelligence in pervasive spaces," *IEEE Pervasive Computing*, vol. 6, no. 3, pp. 85–89, 2007.

[10] N. Kushwaha, M. Kim, D. Kim, and W. Cho, "An intelligent agent for ubiquitous computing environments: smart home UT-AGENT," in *Proceedings of the 2nd IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems*. Piscataway, NJ, USA: IEEE Press, 2004, pp. 157–159.

[11] F. Doctor, H. Hagras, and V. Callaghan, "A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 35, no. 1, pp. 55–65, 2005.

[12] M. Mozer, "The Neural Network House: An Environment that Adapts to its Inhabitants," in *Proceedings of the Intelligent Environments AAAI Spring Symposium*, Palo Alto, CA, USA, 1998, pp. 110–114.

[13] M. Jacyno, S. Bullock, N. Geard, T. R. Payne, and M. Luck, "Self-organizing agent communities for autonomic resource management," *Adaptive Behavior*, vol. 21, no. 1, pp. 3–28, 2013.

[14] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[15] K. Kawamura, W. Dodd, P. Ratanaswasd, and R. Gutierrez, "Development of a robot with a sense of self," in *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2005)*. IEEE, 2005, pp. 211–217.

[16] G. L. Chadderdon, "Assessing machine volition: An ordinal scale for rating artificial and natural systems," *Adaptive Behavior*, vol. 16, no. 4, pp. 246–263, 2008.

[17] X. E. Barandiaran, E. Di Paolo, and M. Rohde, "Defining agency: Individuality, normativity, asymmetry, and spatio-temporality in action," *Adaptive Behavior*, vol. 17, no. 5, pp. 367–386, 2009.

[18] X. Barandiaran and A. Moreno, "On what makes certain dynamical systems cognitive: A minimally cognitive organization program," *Adaptive Behavior*, vol. 14, no. 2, pp. 171–185, 2006.

[19] M. Van Duijn, F. Keijzer, and D. Franken, "Principles of minimal cognition: Casting cognition as sensorimotor coordination," *Adaptive Behavior*, vol. 14, no. 2, pp. 157–170, 2006.

[20] R. D. Beer, "The dynamics of active categorical perception in an evolved model agent," *Adaptive Behavior*, vol. 11, no. 4, pp. 209–243, 2003.

[21] O. Vakarelov, "The cognitive agent: Overcoming informational limits," *Adaptive Behavior*, vol. 19, no. 2, pp. 83–100, 2011.

[22] J. Bongard, V. Zykov, and H. Lipson, "Resilient machines through continuous self-modeling," *Science*, vol. 314, no. 5802, pp. 1118–1121, 2006.

[23] B. R. Fajen and M. T. Turvey, "Perception, categories, and possibilities for action," *Adaptive Behavior*, vol. 11, no. 4, pp. 276–278, 2003.

[24] P. Andry, P. Gaussier, J. Nadel, and B. Hirsbrunner, "Learning invariant sensorimotor behaviors: A developmental approach to imitation mechanisms," *Adaptive Behavior*, vol. 12, no. 2, pp. 117–140, 2004.

[25] A. Ram, G. Boone, R. Arkin, and M. Pearce, "Using genetic algorithms to learn reactive control parameters for autonomous robotic navigation," *Adaptive Behavior*, vol. 2, no. 3, pp. 277–305, 1994.

[26] J. F. Ferreira, M. Castelo-Branco, and J. Dias, "A hierarchical bayesian framework for multimodal active perception," *Adaptive Behavior*, vol. 20, no. 3, pp. 172–190, 2012.

[27] M. D. Erbas, A. F. Winfield, and L. Bull, "Embodied imitation-enhanced reinforcement learning in multi-agent systems," *Adaptive Behavior*, vol. 22, no. 1, pp. 31–50, 2014.

[28] A. De Paola, S. Gaglio, G. Lo Re, and M. Ortolani, "Sensor9k: A testbed for designing and experimenting with WSN-based Ambient Intelligence applications," *Pervasive and Mobile Computing*, vol. 8, no. 3, pp. 448–466, 2012.

[29] P. De Mil, T. Allemeersch, I. Moerman, P. Demeester, and W. De Kimpe, "A Scalable Low-Power WSAN Solution for Large-Scale Building Automation," in *IEEE International Conference on Communications (ICC '08)*, 2008, pp. 3130–3135.

[30] F. Amigoni, N. Gatti, C. Pinciroli, and M. Roveri, "What planner for Ambient Intelligence applications?" *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 35, no. 1, pp. 7–21, 2004.

[31] T. Gu, X. Wang, H. Pung, and D. Zhang, "An ontology-based context model in intelligent environments," in *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation conference*, 2004, pp. 270–275.

[32] D. Preuveneers, J. Bergh, D. Wagelaar, A. Georges, P. Rigole, T. Clerckx, Y. Berbers, K. Coninx, V. Jonckers, and K. Bosschere, "Towards an Extensible Context Ontology for Ambient Intelligence," in *Ambient Intelligence*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, vol. 3295, pp. 148–159.

[33] A. Augello, S. Gaglio, G. Oliveri, and G. Pilato, "An algebra for the manipulation of conceptual spaces in cognitive agents," *Biologically Inspired Cognitive Architectures*, vol. 6, pp. 23–29, 2013.

[34] ——, "Acting on conceptual spaces in cognitive agents," in *Proceedings of the 1st International Workshop on Artificial Intelligence and Cognition (AIC 2013)*, 2013, pp. 25–32.

[35] C. Hernández, I. López, and R. Sanz, "The operative mind: a functional, computational and modeling approach to Machine Consciousness," *International Journal of Machine Consciousness*, vol. 1, no. 01, pp. 83–98, 2009.

[36] P. Ribino, A. Oliveri, G. Lo Re, and S. Gaglio, "A knowledge management system based on ontologies," in *International Conference on New Trends in Information and Service Science. NISS '09*, 2009, pp. 1025–1033.

[37] World Wide Web Consortium, "OWL 2 Web Ontology Language Document Overview," *W3C recommendation*, 2012. [Online]. Available: http://www.w3.org/TR/owl2-overview/

[38] A. De Paola, S. Fiduccia, S. Gaglio, L. Gatani, G. Lo Re, A. Pizzitola, M. Ortolani, P. Storniolo, and A. Urso, "Rule based reasoning for network management," in *Proceedings of the 7th International Workshop on Computer Architecture for Machine Perception (CAMP 2005)*, 2005, pp. 25–30.

[39] S. Gaglio, L. Gatani, G. Lo Re, and A. Urso, "A logical architecture for active network management," *Journal of Network and Systems Management*, vol. 14, no. 1, pp. 127–146, 2006.

[40] E. Friedman, *Jess in action: rule-based systems in Java*. Manning Publications Co., 2003.