An AmI approach to anticipate home inhabitant's needs

Vittorio Miori[*] and Dario Russo
Institute of Information Science and Technologies (ISTI) – National Research Council of Italy (CNR), Pisa, Italy

Abstract. According to the vision of Ambient Intelligence (AmI), the most advanced technologies are those that disappear: at maturity, computer technology should become invisible. All the objects surrounding us must possess sufficient computing capacity to interact with users, the surroundings and each other. The entire physical environment in which users are immersed should thus be a hidden computer system equipped with the appropriate software in order to exhibit intelligent behavior. The objective of our research is to take steps in this direction by proposing a feasible software application able to learn the behavior and habits of home inhabitants in order to anticipate their needs. The result is an adaptive, context-aware application that works as an integral part of a dedicated middleware designed to make today's heterogeneous, mostly incompatible domotic systems fully interoperable. By applying machine learning techniques, it offers a complete, ready-to-use practical application that learns through interaction with the user in order to improve quality of life in a technological living environment, such as a house, a smart city and so on. Although the proposed solution is currently suitable for application to comfort issues, it also represents an opportunity to provide greater autonomy and safety to disabled and elderly occupants, especially the critically ill, as its results can be utilized in applications that can recognize unusual, potentially hazardous situations. In fact, this modeling process serves as the basis for implementation of an e-health system that can actively contribute to anticipating, and thereby preventing, emergency situations. The prototype has been developed and is currently running at the Pisa CNR laboratory, where a home environment has been faithfully recreated.

Keywords: Ambient Intelligent, AmI, Association Rules, Data Mining, DomoNet, Domotics, Home Environment, Machine Learning, Web Services, XML.

1. Introduction

Ubiquitous Computing and Ambient Intelligence (AmI) refer to a vision of the future information society where humans will be surrounded by intelligent interfaces supported by computing and networking technologies that will be everywhere and, largely thanks to the miniaturization of computer components, embedded in objects such as furniture, clothes, vehicles, roads and smart materials.

New advanced services will be created by exploiting the ability to make these objects interact with other people's objects and with the environment. Such technologies will be designed to be 'invisible' to people, who will use them without even realizing it. Thus, the actual computing capacity should remain in the background, in the periphery of our attention, and should only move to the center if and when necessary.

Unfortunately, the state of the art of the Weiser Ambient Intelligence (AmI) vision [1] is still far from becoming a reality, though research activities are continuously proposing new services, algorithms and perspectives [2] able to provide ever more powerful and sophisticated solutions.

If we focus on the home, Ambient Intelligence may be seen as the layer on top of the domotics, per se. Its aim is to progress from the mere programming of isolated devices, to integrating them in order to achieve global, unified goals. Home networks, that is, the specific Local Area Networks for home environments, include both networks to the home (access networks) and networks throughout the home.

The key idea underlying many AmI projects and applications is *context awareness*, which is based

---

[*] Corresponding author. E-mail: vittorio.miori@isti.cnr.it.

mainly on their capacity to identify users and their locations. The AmI vision foresees that our homes will contain a distributed network of intelligent devices that can adapt themselves so as to anticipate users' needs. Thus, Ambient Intelligence refers to the presence of a digital environment that is sensitive, adaptive and responsive to people's presence [3].

In the near future the home will be a technologically rich environment able to offer a wide range of network-based services through support middleware that will make them discoverable by and accessible to residential environments. Such services might range from simple video entertainment streams to complex home energy-management packages. The domestic network will connect all household appliances and displays regardless of manufacturer, and will moreover interact with the personal area network and the body network of each person in the home. Within homes, local networks with potentially different underlying implementations can be combined, or at least managed, as one logical network. The need to support interoperability services, such as bridges, gateways or adapters [4] between different networks in the backbone, as well as in access and local networks, is an essential prerequisite for all AmI applications.

The current immaturity of the field of domotics and, more specifically, the lack of definition of application requirements, have led to the development of a large number of ad hoc proposals, which unfortunately are often limited and difficult to integrate. In order to make the advent of genuine AmI applications possible, there is a crucial need to define and develop a standard way forward. Although domotic and smart home technologies are currently ready and operational, they have as yet not been able to garner a broad consumer market, some of the main causes being the lack of standards and interoperability, as well as the absence of any "must-have" new functionality provided at an appealing cost.

One interesting and open issue regarding home automation and Ambient Intelligence is related to recognizing unusual or dangerous situations in order to anticipate health problems or special individual home user needs. Indeed, every user has different behaviors and habits that may depend on his preferences, inabilities or state of health. Such problems may be addressed by monitoring users' habitual activities, which enables creating rules-based profiles in order to capture and formalize their normal behavior. Such monitoring activities can be carried out using a system based on machine learning, which exploits artificial intelligence algorithms to learn user habits and build a knowledge base that enables taking into account experiences accumulated during day-to-day activities. However, many people have privacy concerns about having their behaviors monitored [5], the rules for monitoring implementation and what information is transmitted and to whom.

AmI based systems can offer significant advantages to the elderly and/or disabled, as well. For these segments of the population, even the simplest of everyday actions may represent an insurmountable obstacle. So a system that learns their habits and performs actions in their place can offer significant benefits, and thereby also contribute to reducing the currently acute problem of the digital divide.

Today's e-health solutions provide important contributions to the health management of the elderly and chronically ill within their own homes. Indeed, they provide for constant monitoring of many vital parameters via portable sensors (inserted into shirts, bracelets, watches, etc.), in order to be able to identify and opportunely signal any hazardous situations requiring intervention. In most cases, however, by the time the call for help is issued, the emergency is already in progress. A system able to anticipate danger before life-threatening situations arise would certainly lead to faster, more effective intervention. The ability to anticipate and recognize certain behaviors or events heralding serious health problems in time can often save lives.

In order to anticipate and recognize such situations, data on user activity and the surrounding environment need to be contextualized and enhanced semantically. This aim can be achieved by exploiting one of the most important paradigms underlying the *Web 3.0*: *Semantic Intelligence* [6]. The *Semantic Web* [7] is an Internet space that includes documents (or portions thereof) describing the explicit relations between things, and containing semantic information intended for automated processing by machines. Not only Web pages, but databases, programs, sensors and even household appliances will be able to present data, multimedia, and status information in forms that powerful computing agents can use to search, filter and prepare information in new and exciting ways. By way of definition then, in the *Semantic Web* an *ontology* [8] is a partial conceptualization of a given knowledge domain, shared by a community of users, that has been defined in a formal, machine-processable language for the explicit purpose of sharing semantic information across automated systems.

2. Related works

A large body of literature underscores the currently great research interest in Ambient Intelligence and methodologies for anticipating user needs. Chin [9] has proposed three methodologies to address the relevant issues:

- *pre-programmed rules*: usually created by the developers or manufacturers. These suffer from a seemingly intrinsic limitation, which is that once the device has been programmed it is difficult to modify appliance settings in order to personalize and adapt them to changing user needs;
- *user-programmed rules*: created through end-user programming, leaving up to users the task of setting up the system and adapting it to their changing needs. Such an approach cannot always be applied because it requires skilled end-users, which may represent an obstacle to accessibility for many people;
- *agent-programmed rules*: created by intelligent agents, artificial intelligence and machine learning mechanisms. This appears to be the most promising approach, as it leaves entirely up to the system the creation, management, removal and modification of rules in order to obtain a personalized, automatically adaptable environment designed around the user.

CASAS [10] introduces an adaptive smart home system that uses some interesting techniques to discover frequent and periodic patterns in users' daily actions via machine-learning and data-mining approaches. This project is in line with our approach, because it relies on the system to discover all patterns without any initial assumptions.

Aztiria et al. [9] propose a system based on pattern recognition to discover sequences of user events and thereby understand users' behaviors and act accordingly to automate actions and devices. Users are able to accept system proposals using speech recognition, which allows them to interact with the system to modified the learned patterns. From the AmI perspective, one drawback of such an approach is the need for direct user intervention: the system should be able to understand and, if necessary, automatically remove any undesired patterns.

Tapia [10] has applied the *Navie Bayesian classifier* to recognizing everyday life activities, such as bathing, toileting, dressing and preparing lunch, by analyzing data collected from a set of simple small state-change sensors. In recent years, data mining techniques have been used to process data captured from a collection of sensors placed throughout the environment. Luihr [11], having arrived at conclusions similar to our own, uses the data mining method known as *association rule learning* to detect new behaviors occurring within smart homes. Zheng [12] uses a self-adaptive neural network, called Growing Self Organizing Maps, to conduct cluster analysis of day-to-day human activities. Hasan [13] proposes a method that uses Bayes RN-Meta-networks to support online preference discovery mechanisms in context-aware environments. It aims to find the best solution when the presence of a number of different users sharing the same spaces creates conflicts in the environmental settings. Mileo [14] follows an interesting approach that uses logic programming techniques to reason about different information to support independent living and well-being for people with no critical chronic condition living in their homes.

Much of the recently emerging work on Ambient Intelligence exploits the power of semantics and ontologies. Santofimia [15] proposes some clear and formal definitions of the meanings of *context*, *action, event, service, device* and *object.* He proposes an agent-based architecture, whereby each agent interacts with the environment according to the *BDI* paradigm (*Belief*, *Desire* and *Intention*). Okeyo [16] remarks on the importance of user activity recognition and proposes an architecture for activity learning and activity model evolution through the analysis of ontology-based activity traces. Chen [17] proposes a platform that makes use of ontologies to model, represent and infer the lifecycle of activity recognition. He also maps the data coming from sensors into ontological classes and properties, and uses them for the assertional part of the ontology. Stavropoulos [18] proposes the use of semantic annotations in WSDL documents to enrich them with semantics. He moreover stresses the importance of adding semantics to the preconditions for and the effects of actions in order to be able to obtain a thorough description of the context. Finally, Panagiotopoulos [19] proposes a solution for profiling users using ontologies.

Unfortunately, none of these works seems to address the crucial issue of the current interoperability between technologically different sensors and actuators.

3. DomoNet ecosystem

The many domotic systems crowding today's market are rarely interoperable. One consequence of this situation is that consumers cannot choose devices according to their requirements or other fundamental criteria, such as cost, performance, trends and confidence, but are liable to worry about issues of compatibility with their existing systems. Unfortunately, current market practice generally binds consumers to proprietary technologies, forcing them to purchase only devices that conform to a specific standard in order to obtain a suitable level of interoperability. In order to overcome these issues and achieve functional, integrated AmI applications, a fully interoperable domotics environment must be created.

To address this issue, our laboratory has developed a digital 'ecosystem' based on the *DomoNet framework* [20] – an open source software released under the GPL license, written using the Java language and open source libraries and tools. The core of *DomoNet* consists of middleware based on *Web Services* and *Service Oriented Architecture* (*SOA*) paradigms, in which the services coincide with the functionalities offered by the devices available in the environment in question.

A pivotal component of the *DomoNet* framework is *DomoNetWS* – a *WebServices*-based engine whose task is to enable true cooperation between nodes. It constructs a unique view of the system, including all the devices belonging to all the different domotic systems available (figure 4) through a set of modules that work as gateways, called *Tech Managers*, to deal with specific domotic systems (e.g. KNX, Lon, X10, Jini, HAVi, UPnP, ZigBee, etc.).

It should be noted that these technology choices are enabling factors for remote control, as well as interoperability. In fact, *DomoNetWS* is an actual Internet node designed to share environments and services in a distributed fashion with any other *DomoNetWS* or *DomoNet* application to monitor and control all home environments at a distance through the use of Internet-capable devices, such as PCs, PDAs, Tablets, Smartphones and so on, irrespective of the technologies adopted in the specific domotic devices.

DomoNet moreover defines a standard language, called *DomoML*, that enables abstraction of heterogeneous systems in order to describe device functionalities, data types, messages and models of the interactions and communications among framework entities. It represents a "lingua franca" that links several single domotic systems, providing them with common information and control exchange mechanisms. As such, it can be used as a sort of universal domotic language to be applied in any domotic context. Such a solution, although functional, has the limitation that it leaves it up to humans to understand what each device represents and what effects its actions have within the environment. Instead, as described in the following, we also need the software to be able to act, based on information about which it can 'reason' with regard to a particular environmental context.

To this aim, semantic support has thus been added to the framework, specifically utilizing the upper *layers* of the *application stack*. In this way, *DomoNet* is able to contextualize, that is, give meaning to both the environmental spaces and the sensors, actuators, furniture, etc.

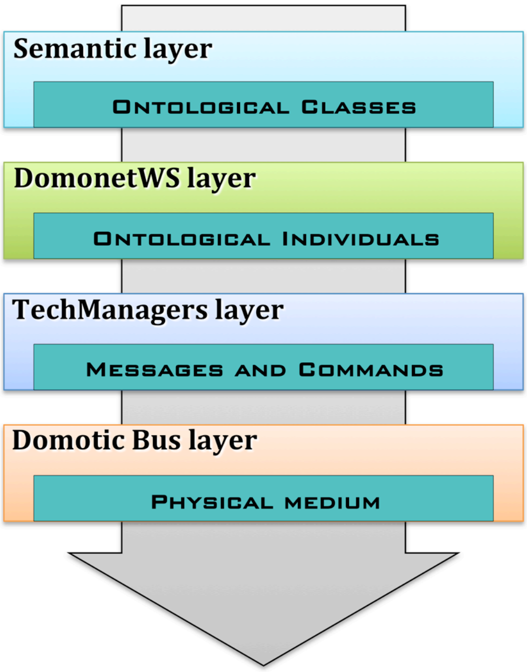

Fig. 1. Global platform architecture

Furthermore, the introduction of semantics has provided a much more powerful way of managing the issue of interoperability between devices conforming to different technological standards. The features and related effects on the environment of the activation of each device are described semantically. Thus, by setting a target (for example, more light in a room), the system is able to decide for itself whether to turn on a lamp or open a shutter, adapting its choices according to the preferences learned from the user's behavior.

The system is now able to consider whether or not to perform certain operations in order to avoid situations hazardous to the user (e.g., an Alzheimer's patient's can be prevented from leaving the house alone). Moreover, by semantically formalizing the furnishings and how objects are distributed within the spaces, it is also possible to identify an object by its position in the environment (for example, the lamp on the table, the bathroom light, etc.).

The new *application stack* (Figure 1) consists of four *layers*:

- *Semantic Layer:* the semantic level provides a semantic description of the operations of the home automation devices, the furniture, the environments and the lexicon that are supported by the platform.

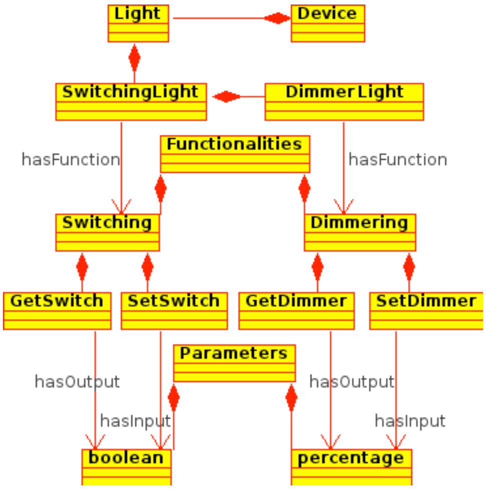

Fig. 2: Some DomOnt classes with their related relations

At this level all of the descriptions are formalized through a set of dedicated *Ontological Classes*, whose concepts are applicable and always true independent of the context in which they are used. For example, figure 2 shows the semantic model of the device " light switch" and its functionalities (“SetSwitch to set the light on or off using a Boolean value, and “GetSwitch” to get the light’s current status). The "lamp" device also includes a "dimmer light", which has the extra feature (over a "simple light") of being adjustable in intensity. Such information is described in the *DomOnt* ontology (*Domotic Ontology*), which defines an integrated ‘taxonomy’ of domotic devices and their functionalities. This ontology defines separate classes of devices, their functionalities, input/output parameters and the effects they have on the environment.

Regarding the furniture, *FurOnt* (*Furniture Ontology*) provides a simple taxonomy of the home furnishings. In particular, it defines and classifies objects such as tables, chairs, mirrors and so on, while the places where the furniture and equipment can be placed are described using the ontology *EnvOnt* (*Environment Ontology*).

This ontology also describes the overall home environment, that is, it defines the meaning of different rooms, such as “bathroom”, “kitchen” or “bedroom” and includes relations such as “contains bed”, “the place for eating”, etc. In this way, it is possible to locate a specific room without calling it explicitly, by simply referring to it by its characteristics (e.g. “the room with the bed”).

Finally, the lexicon is defined by the ontology *LexOnt* (*Lexical Ontology*). It defines the vocabulary suitable for specifying the positions of objects and devices inside the environment and their inter-relations. The defining words here are, for example, “near”, “on”, “under” and so on.

- *DomoNetWS Layer:* creates the *Ontological Individuals* relating to the *Ontological Class* defined at the top level (i.e. the semantic layer). These *Individuals* enable contextualizing the specific design of the environment where the user is located. Each entity (device, object, space) has its own corresponding *Individual* within the ontology. Note that at this level the features and functionalities of each *Individual* are already known, since they have been defined at the upper level as general characteristics of its corresponding class. To achieve interoperability between two natively incompatible devices, the functionalities of the related ontological *individuals* are thus set in relation. For example, if we want to interoperate with a push button and a lamp actuator belonging to two incompatible home automation systems, all that is needed is to specify the relation and describe ontologically how to convert data from one standard to the other through associations of equivalence (e.g. on ≡ true, off ≡ false). In this way, when the “*setStatus*” push button functionality is called, the “*setSwitch*” function of the lamp is also called, using the same input parameter value (e.g. *on* or *off*) for the two functions.

- *TechManagers Layer*: is composed of a set of software artifacts, one for each supported domotic system. Each artifact is responsible for managing the mapping between real devices belonging to a particular domotic system to the respective ontological one (*Individual*). That is, interoperability reasons, each of these artifacts is responsible for the creation, removal, and sending of domotic commands as well as for the translation of message in the correct formats.

- *Domotic Bus Layer* includes the physical infrastructures of the domotic systems. They are substantially represented by the communication channels, which can be wired or wireless, depending on the domotic system in question.

The following describes the details of how communication takes place between two devices, regardless of the specific domotics technology they utilize. Returning to the example of the switch and the light bulb, when you press a button on the panel, the event generated by this action passes through the *domotic bus* (*Domotic Bus layer*) and is subsequently captured by the corresponding *TechManager* (*TechManager Layer*). The *TechManager* identifies the corresponding ontological *Individual* of the device and the specific feature of the *Individual* generating the event. The *DomoNetWS* (*DomoNetWS layer*) verifies the relation between the button *Individual* with one of the lamps. The *DomoNetWS* identifies the *TechManager* responsible for the lamp and will find the corresponding function to invoke. Then, after having carried out the mapping, the *TechManager* sends a write request over the *domotic bus* for execution of the associated service.

The entire communications infrastructure between ontologies belonging to the same or different levels have been implemented using the *JADE* open-source library [21]. *JADE* provides for an agent-based network architecture in which each agent is responsible for managing a specific ontology. The various agents used for manipulating the ontologies have instead been built using the *OWL API* library [22].

4. DomoPredict Framework

4.1. Anticipating needs

With the aim of enabling the system to anticipate users' needs, we define a scenario as a set of user actions that are in some way related to each other. By monitoring users' activities in a highly enriched domotic home environment, it is possible to learn to recognize such scenarios and anticipate the needs of its inhabitants. To this end, we have built a software system able to analyze the data collected through such monitoring and, based on its analyses, take decisions and act in place of the users.

In pervasive environments, we define the *context* as the representation of the knowledge that a system has about its own state. It is not simply a snapshot of the environment at a particular time, but instead usually represents information over a given period of time during the life of the environment itself. It provides the knowledge bases for systems that learn from past contexts and experiences, thereby allowing for advanced adaptive capacities and facilitating proactive decision support with varying degrees of autonomy.

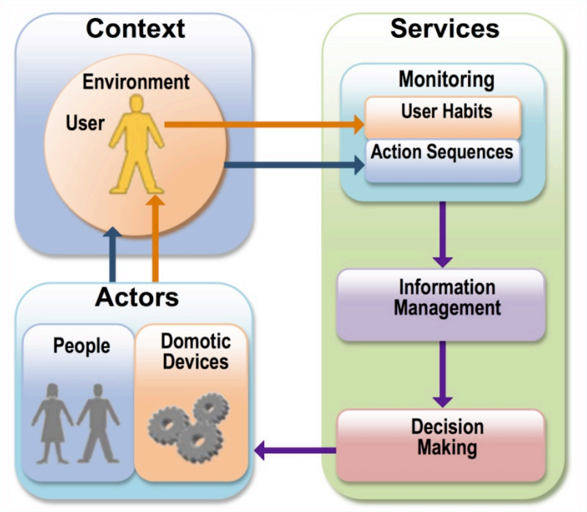

Fig. 3. Scheme of AmI based systems.

Exploiting an Artificial Intelligence system based on an *AmI* paradigm, the system configuration stage is limited to the physical installation of the devices and software without regard for adjustments and settings, which often can be difficult to understand and put into practice. Users can simply go about their usual business within their living quarters and just ignore the technology surrounding them.

As shown in figure 3, the lifecycle of an *AmI*-based system includes the acquisition of information about users and their environment by means of a *monitoring service* software module. The collected data is then analyzed and processed by the *information manager* module. A *decision-making* software application then uses this processed information to identify the actions to be performed using machine learning techniques. Lastly, decisions are translated into commands and sent to recipient domotic devices, which together with any reactions on the part of occupants, modify the initial settings. The operation of the system can therefore be viewed as an infinite loop whose steps are performed in sequence each time that an update notification is received of the status of a device in the domotic network.

4.2. The architecture

To exploit the above-described approach, isolated devices must be integrated in order to achieve global, unified goals. Interoperability and standards are crucial in this respect.

The *DomoNet Ecosystem* has been applied to tackle this problem. By virtue of its capacity to abstract the peculiarities of underlying heterogeneous, well-established domotic technologies, it enables them to co-exist and interwork, without eliminating their differences.

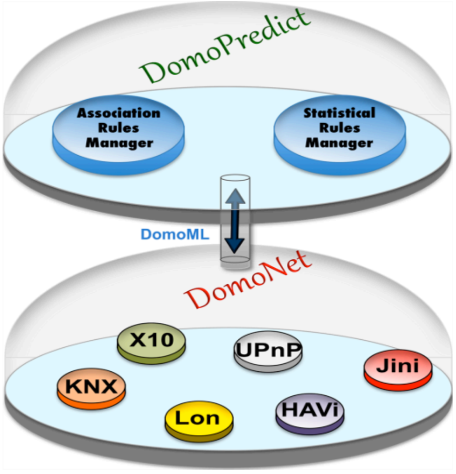

Fig. 4. DomoPredict is a DomoNet client

The *AmI* prototype, *DomoPredict*, is a software component of the *DomoNet* project. As shown in figure 4, *DomoPredict* is able to communicate with *DomoNet* and through it, send commands and receive notifications of any state change in any domotic device. This enables the AmI system to keep track of all the actions performed by users in order to generate rules.

In particular, the lifecycle of *DomoPredict* can be divided into 4 steps:

- the *information collection* step is activated when a device changes state and consequently a multicast update message is sent from *DomoNet* to all *SOA* clients that have subscribed to the update notification service, according to the *publish/subscribe* design pattern paradigm. Since *DomoPredict* is a subscriber to this service, when it receives an update message, it analyzes its content and builds and enhances the rules data-set .

  At the same time, a *log manager* archives the information about the updates in an XML file for diagnostic purposes and appropriate analysis. It stores all the actions performed by users, the device that has changed status, the new status and the identifier of the user who performed the action, as well as the time of activation;
- the *collected information analysis* step enables the *Information Manager* to identify the sets of actions that may lead to the recognition and creation of new scenarios;
- if the analysis step has recognized a new scenario to be created, the *creation / removal of rules* step will create new rules that represent application of the new scenario by the *Decision Maker.* Previously learned rules are modified by the system when, over time, a change in learned habits or external factors occurs. In such circumstances, the system is able to remove and eventually substitute previously learned rules that are deemed no longer valid according to newly acquired experience;
- finally, during the *rules execution* step, the prerequisites for a learned rule are verified, the rule is applied by the system by invoking the corresponding commands to be sent by *DomoPredict* to the *DomoNet* middleware. *DomoNet* will then be able to route them to the appropriate devices.

Given the wide variety of different scenarios that may arise, the software is made up of two complementary, interoperating modules: the *association* and the *statistical rules manager* (figure 5). Working together, they perform real-time analyses aimed at discerning *sequences of events*, that is, a set of interactions with the domotic environment that occur within a fixed short time span (a few seconds, minutes or hours), though not necessarily in a specific sequence, and may therefore represent user habits.

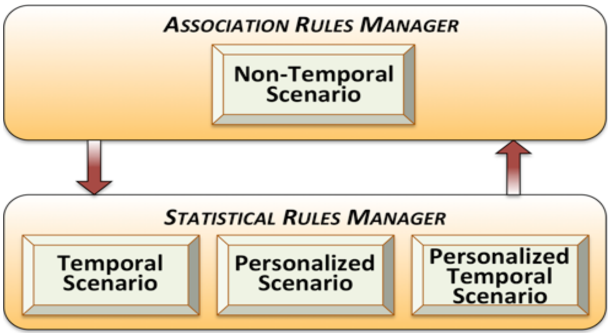

Fig. 5. DomoPredict modules.

4.2.1. Association rules manager

The *association rules manager* is responsible for learning scenarios made up of a set of actions habitually carried out by the user. These scenarios are made up of actions related to each other in the sense that they occur within a short interval from each other, but are unrelated to any specific time of execution. These are called *non-temporal scenarios*.

Because it is very likely that some actions are preset in most scenarios, the necessary condition for determining a non-temporal scenario is that a user perform a set of actions sufficiently regularly to discriminate the scenario to be actuated. For example, a user who, leaving the house, habitually turns off all the lights and closes all the shutters, may occasionally also shut off the gas valve and turn off the TV or stereo, or even forget a light on or leave the shutters deliberately open so the sunshine heats the house in winter.

To recognize such patterns, the manager applies a *Data Mining* paradigm, in particular, the *association rules* method. This allows the generation of opportune rules using binary partitions of the itemset that determine a scenario being learned. It is thus possible to anticipate upcoming actions when a user performs antecedents of such actions (i.e. actions leading up to them). In order to generate the frequent itemsets (which are the potential scenarios), the constraints of the *Apriori* algorithm are used, which generates candidate action sequences via the standard method defined as $F_{k-1} X F_{k-1}$ [23].

The rule for discovering whether a specific scenario is to be activated automatically is represented via the form $\{X => Y\}$, where X and Y represent the two sets of events into which each scenario is divided. X contains the precondition events for scenario activation, while Y contains all other remaining events. The result is: if X occurs, then Y is also performed.

The strength of an *associative rule* can be measured as a function of its *support* and its *confidence*. In our case, support represents the fraction of scenarios that contain both X and Y, while confidence instead represents how often the events in Y are also present in a scenario containing X. The problem of discovering *associative rules*, given a set of scenarios, consists in finding all the rules whose *support* and *confidence* values are equal to or greater than preset thresholds.

The size of the dataset is very important for the proper functioning of *data mining* algorithms. Indeed, data sets can be very large, often requiring days of computer time to create a single model. From this prospective, usual data mining is unsuitable for our purposes, because the dataset is empty at startup and is created in real time via the update broadcast messages sent by *DomoNet* when a device state changes. The solution found to overcome this obstacle is to act on the support parameter of the *Apriori* algorithm, bearing in mind that the few data available initially could lead to the acquisition of erroneous habits. In order to limit such erroneous habits acquisition, the dataset is enriched with a new itemset only when the *minimum support* parameter is greater than a prefixed threshold. As this parameter is used to evaluate whether or not a group of actions is frequent, simply increasing its value when dealing with small datasets will make it more difficult for a given scenario to be learned, thereby preventing infrequent itemsets from being considered. Thus, at first, a rather high value is set, which is then decreased in the long term proportionally with the increasing population size of the dataset, so that eventually most itemsets will be deemed frequent, thereby allowing even rather rare habits to be learned.

The choice of implementing *non-temporal scenarios* using *associative rules* was based on the consideration that sufficient training data is unavailable at start-up time and the dimensions of the learning system hypothesis space should not be fixed. Thus, an *unsupervised machine learning system* had to be adopted, in order to allow the creation of relationships and groupings between similar data [24]. Moreover, the possible scenario to enact must be chosen from among a finite set of possibilities.

To create a new *non-temporal scenario* rule, a *method* is needed that can efficiently perform the following two tasks: (i) classify a group of actions as a scenario and (ii) find a way to enact this scenario after learning it. Indeed, only in ideal cases would users perform the exact same sequence constituting a scenario. Scenarios are much more likely to be identified through a subset of the entire action sequence making up the scenario.

The first task that must be performed in this process is to arrive at some groupings of potentially correlated actions, which will be represented by those performed in succession over a brief time interval. Such time interval, called the *correlation window*, is a system parameter whose duration can be adjusted as needed. It does not represent an absolute time interval within which actions must be performed in order to be deemed correlated, but it is the maximum time gap that indicates a logical dependency between the last action performed by the user and the previously obtained set of correlated actions. Such procedure necessarily requires a stage of data preprocessing, so as to determine the minimum set of actions that enables recognizing the correct scenario. The result is a table containing all the action sequences performed over a brief interval and therefore potentially correlated. This is called a *sequence table* and will be used as the basis for the subsequent stage of learning the scenarios.

The second task is performed under the constraint that any action belonging to a specific scenario cannot be allocated to any other scenario. It is worthwhile underlining that the efficiency of the algorithm does not depend on the order of the completed actions, but only on the temporal correlations existing between the actions performed, since the actions making up a scenario are not always performed in the same order.

Moreover, acquired scenarios are continually subject to modification. An essential requirement for software efficiency is the ability to quickly adapt to users' habits. This goal is achieved by implementing a *reinforcement* function in the machine learning procedure. This function is activated by users when they unconsciously correct undesired or incorrect system actions. For example, if the system enters a scenario that calls for switching on a light, and the user switches off that very light, this fact permits the system to 'understand' that this particular scenario is incorrect. Another example is when users modify their habits with changing seasons. The system adapts itself automatically to new user habits (e.g. turning off the lights later or opening the windows more often in springtime), without changing any hardware device settings. At the beginning, the presence of this reinforcement procedure is essential in order to enable the system to modify any erroneously learned rules due to a dearth of collected experience. The erroneous rules are relegated to a blacklist and cannot be re-learned for a period of time proportional to the number of times the user has provided negative feedback.

For the activation of *non-temporal scenarios* it is necessary to identify the minimum set of actions that enables identifying the scenario to be applied. Automatic identification of this minimum set is the most important and critical feature of the system. For example, let us suppose that the system has learned a scenario that includes the two rules: "switch on the light in the living room" and "switch on the TV". Once the user has switched on the light in the living room, it must be determined whether (s)he wants to turn on the TV as well. To do this, it is necessary to calculate the probability that the performance of a group of actions belonging to a scenario implies execution of the others in that same scenario. The conditional relation need not be one of certainty – a high probability is sufficient grounds for anticipating the need. The associative rules method permits creating a rule in such a way that it associates groups of antecedent actions for executing groups of consequent actions.

The removal of a scenario comes about when the confidence level does not, within a certain time period, reach the preset threshold value for triggering the scenario.

4.2.2. Statistical rules manager

The *statistical rules manage*r is designed to learn scenarios that are not captured by the *association rules manager*. These scenarios include events occurring with systematic periodicity (i.e. each day at the same time or during different periods of the year), as well as the user's living quarters preferences. To this end, the module creates a user profile obtained by statistically analyzing the frequency and percentage of use of appliances. Activation of such preferences can be tied to the particular moment or a sequence of events that has occurred. The percentages of use are calculated both on a daily basis and for shorter periods of time (e.g. in the morning from 8 to 9 am). This enables identifying potential habitual system states that may present at certain times of the day, without the user performing a sequence of actions.

Collected data are recorded in structures called *UsageTables* in the pair format <*device state, percentage*>, which indicate either the percentage time a device is in a particular state, or the percentage time that certain events occur (for instance, listening to favorite music or maintaining a room temperature). A number of different timeframes (daily, weekly, seasonal and perpetual) are considered and a different *UsageTable* created for each. Such data are used to satisfy user preferences through a conditional rule of the sort *condition => set preferences*, where the *condition* is dictated by specific events, such as for example, waking up, returning home, a new season's start, and so forth.

The scenarios captured by the statistical rules manager are:

- *temporal scenarios*: these are made up of one or more events usually occurring at the same time of day or for a long period of time. Once the constituent events of the scenario are learned, they are executed automatically at the pre-established time. To build *temporal scenarios* it is sufficient to observe the relations between actions and time. A reasonable choice seems to be to observe the activation time of the scenario within a preset number of days prior. If an action is performed every day at a certain time within this preset time 'window', we can assume that a relation exists between the action taken and the time of the day it is taken, and we can thus have it performed automatically. The time for its execution can be calculated by taking the average time at which the action was initiated over the preceding days. The removal of a *temporal scenario* is accomplished exclusively through a negative reinforcement mechanism, that is, when a user performs an action contrary to that learned by the system;
- *personalized scenarios*: these define a set of actions/parameters that the system uses to configure the environment according the inhabitant's personal preferences. The learning of a personalized scenario aims to increase the comfort and safety of the user within the environment. To this end, the user preferences learned over time through his/her daily device usage are analyzed. The system is preconfigured to learn the temperature, lighting levels, favorite musical genre and the values of the inhabitant's main vital functions, so as to allow for constant monitoring of the state of health. The rules are created statistically based on the normal distribution over time of the parameter values automatically learned by the system. Activation of a scenario depends on the occurrence of certain situations, such as for instance when a user enters the living quarters. In this case, the actions taken are aimed at controlling the environment according to the user preferences learned.
- *personalized temporal scenarios*: these represent the living parameters learned by analyzing any customary preferences repeatedly set by the user at specific times (e.g., which take place each day or each week at the same time). Although learning this type of scenario depends on user execution at specific times, its removal is accomplished in the same way as for a *personalized scenario*.

5. Use case and Verification

The prototype's functionality has been verified and validated in a use case study aimed at checking the system's ability to learn user habits, anticipate them and recognize potential hazards to the users' health. The software tests were performed at the ISTI-CNR laboratory, where a domotic environment simulating a real residence has been set up. A volunteer member of the research team was assigned the task of interacting with the system during the course of his normal work in the laboratory. During the two-week test period the volunteer carried out customary daily habits under usual circumstances, that is, by simply performing a series of repetitive activities within the test setting. Moreover, a number of actions, suggested by a cardiologist as typical of a potential worsening of a heart patient's state of health, were also simulated. To this end, four parameters representative of cardiac risk factors were inserted into the algorithm: coughing, using the toilet, rest hours and body weight.

The test environment was equipped with the following domotic devices:

- presence sensors: to detect when someone enters or leaves a room;
- dimmer light: another way to detect the presence of someone in the room;
- thermostat: to measure the current ambient temperature;
- pressure sensor under the sofa and bed: to detect when someone is sitting or lying down;
- panic button plus microphone: to send an alarm and communicate from any room in the house;
- a fall detection system: to detect when a user falls;
- flood, fire and gas sensors: detecting anomalies in the environment;

- motion sensors: to verify if particular rooms in the house are not being used. When rooms such as the bathroom and kitchen remain unused, it may signal the user's inability to fulfill the most basic needs;
- door opening sensors: to detect that a person is leaving;
- a pocket accelerometer, so that the user's movements throughout the entire day can be followed (time to climb the stairs, use of an exercycle, number and duration of movements around the house), as well as the duration of subsequent rest periods;
- a microphone was installed in the user's bedroom to measure the frequency of coughing, and its recordings were processed with expressly developed *DSP* (*Digital Signal Processing*) software;
- night-time bathroom use was checked by simply counting the number of times the bathroom light was turned on;
- a body scale to check changes in body weight transmits data to the data control system in real time.

The habitual activities had to be performed in relation to precise circumstances, which were: "waking up in the morning and having breakfast", "going out", "returning home from work in the evening and having supper", and "after supper until going to bed". In order to reduce system training time and the subjects' length of stay in the laboratory, data were collected for a further four weeks using realistic random data with information replication in order to have data available on eight weeks of system use. At the end of the test, system validation was carried out according to the *K-Fold Cross Validation* method [29].

The rules produced by *DomoPredict* during execution of the test are:

- at 7:30 am each day => set thermostat to 21° C, switch on bedside lamp, open bedroom blinds, switch on bathroom light;
- successful activation of fingerprint reader => switch on living room light, switch on dining room light, switch on kitchen light, switch on kitchen TV, deactivate intruder alarm, set thermostat to 21° C;
- detection of occupant in living room => switch on TV;
- at 11:00 pm each day => close all blinds and switch off all lights, close water and gas electromagnetic valves, activate intruder alarm, set thermostat to 18° C;
- preferred temperatures 11:00 pm–07:30 am => 18°;
- preferred temperatures 07:30 am–11:00 pm => 21°;
- preferred music upon reentry => Jazz;
- preferred music on awakening => Classical;
- coughing + 30%, toilet use +20%, rest hours +34% => Alarm;
- using the toilet +30%, body weight +0.5% => Alarm;
- rest hours +21%, toilet use +85%; body weight +0.3% => Alarm;

After the test period, the software was able to learn sufficient user baseline routines. Regarding the ability to anticipate user actions, the best results were obtained by setting the *correlation window* size to fifteen minutes, while the best setting for successful prediction of a possible health hazard was seven days. With such a value, the validation performed using the available dataset yielded a *specificity* value of 89%, with 88% *sensitivity* and 89% *accuracy*.

6. Conclusions

The results obtained in the foregoing test serve to illustrate the effectiveness of the approach adopted for implementing the automatic learning system: it was able to learn the four different types of habits (temporal, non-temporal, personalized, personalized temporal), which enable covering a large part of the common behaviors of Ambient Intelligence system users, and can anticipate such users' behaviors quite reliably.

The choice of following a hybrid approach – applying both the *Data Mining* technique of associative rule learning and statistical learning methods – has rendered the system more versatile and reliable. In fact, combining the two forms of learning has led to a summing of the strong points of the two different learning methods, while limiting their respective limitations.

The system developed fits well within the perspective and goals of current *Ambient Intelligence* research. The methods applied for learning the behaviors and habits of *AmI* system users enables the system to anticipate their needs quite well. Moreover, system performance improved over time, as new experience was accumulated. The number of errors committed by the system was relatively low right from the start and then fell further as the system acquired ulterior data.

However, as can be expected, to achieve more realistic results the prototype clearly requires more thorough, longer-duration testing in order to improve learning and enable more careful evaluation of the system parameters (at least 12 weeks). A more extensive dataset will surely enable more accurate validation and evaluation of the system's capabilities. The *DomoPredict* system thus represents a good point of departure for future development, with the main goal of improving the learning capacity achieved so far.

The software validation test performed has also demonstrated that this tool can be an important aid for preventing any possible deterioration of the user's state of health, as it already supports the application of suitable rules and thresholds for recognizing any changes in some user behaviors or sensor parameters that would signal a potential rise in any of the risk factors. In any event, the need for the help and cooperation of medical specialists will undoubtedly be essential for enabling the system to actually be used as a medical prevention tool.

7. Future work

In order to achieve sufficiently reliability deductions, the data acquired through the environmental sensors is alone not sufficient. Such data must be integrated with the information that can be captured via wearable devices able to able to provide data about:

- body temperature;
- blood pressure;
- pulse;
- blood oxygen saturation;
- body weight;
- percentage body fat;
- percentage muscle;
- percentage water;
- blood glucose;
- EKG;
- peak expiratory flow;
- coagulation.

Before performing any test of the system as applied to both real scenarios and patients, the platform will have to be supplemented with wearable device support. Moreover, the awareness of the elderly people and their reactions to being continually monitored and supervised may present obstacles. They will likely feel controlled and managed by something that they do not fully understand and must thus be given a sense of security and protection, without inducing anxiety [5]. For this reason, it is always crucial that any *AmI* solution be as hidden and unobtrusive as possible.

Considering the wearable sensors that may be added, possible candidates for testing will be patients that meet the following criteria:

- *Chronic Disease History*: primary diagnosis of *Chronic Obstructive Pulmonary Disease* (*COPD*) and / or *Chronic Heart trouble*; one or more hospital emergency admission in the last year due to exacerbation of chronic disease and unstable conditions deemed to be due to anxiety about their condition;
- *Patient functioning/ability*: reasonable cognitive ability to report observations and reasonable skill in using the home appliances and peripherals for vital signal measurements (e.g. blood pressure, pulse, glucose, oxygen saturation, and body weight and temperature).

The goal of testing as applied to both actual scenarios and patients is to verify:

- whether the system can be considered a useful, convenient tool for health care delivery;
- whether it will be able to save time and money by reducing hospital admissions, emergency room and medical practitioner visits and associated travel;
- that users feel they are better informed about their health conditions, thus promoting active participation in their health management and empowering them to perform better self-care;
- that the system can improve health management by providing physicians with more accurate, up-to-date information to help them make better decisions.

Finally, further important future work will be dedicated to enabling the system to identify specific users and their locations [25], for instance when they enter or leave a room. To improve such identification, an *RFID*-based strategy [26] [27] can be employed to enable the system to recognize who performs an action, when and where.

References

[1] Weiser, *The computer for the twenty-first century*, Scientific American, 265, 3 (1991), 94-104.

[2] E. Aarts and B. de Ruyter, *New research perspectives on Ambient Intelligence*,. J. Ambient Intell. Smart Environ. 1, 1 (2009), 5-14.

[3] B. Fabian, T. Ermakova and C. Muller, *SHARDIS: A Privacy-Enhanced Discovery Service for RFID-Based Product Information*, IEEE Trans. on Industrial Informatics, 8, 3 (2012), 707-718.

[4] S. V. Mudiam, G. C. Gannod and T. E. Lindquist, *Synthesizing and integrating legacy components as services using adapters*, Science of Computer Programming 60, 2 (2006), 134-148.

[5] S. Moran and K. Nakata, *Ubiquitous monitoring and user behaviour: A preliminary model,*. J. Ambient Intell. Smart Environ. 2, 1 (2010), 67-80.

[6] J. Hendler, *Web 3.0: Chicken Farms on the Semantic Web*, Computer, 41, 1 (2008), 106-108. doi: 10.1109/MC.2008.34

[7] T. Berners-Lee, J. Hendler, and O. Lassila, *The Semantic Web*, Scientific Am., (2001), 34–43.

[8] A. Grigoris, G. Antoniou and F. Van Harmelen, *Web Ontology Language: OWL,* In Handbook on Ontologies in Information Systems (2003), 67-92.

[9] J. Chin, V. Callaghan and G. Clarke, *A programming by example approach to customizing digital homes*, in IET-International Conference on Intelligent Environments, Seattle, 21-22 July 2008.

[10] P. Rashidi and D.J. Cook, *Keeping the Resident in the Loop: Adapting the Smart Home to the User*, Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on , 39, 5 (2009), 949-959. doi: 10.1109/TSMCA.2009.2025137.

[11] S. Luihr, G. West and S. Venkatesh, *Recognition of emergent human behaviour in a smart home; A data mining approach*, Pervasive Mob. Comput. 3, 2 (2007), 95-116.

[12] H. Zheng, H. Wang and N. Black, *Human activity detection in smart home environment with self-adaptive neural networks*, in IEEE International Conference Networking, Sensing and Control - ICNSC 2008 (2008) 1505-1510.

[13] L. M. Y.-K. L. Md. Kamrul Hasan, Kim Anh and S. Lee., *Conflict resolution and preference learning in ubiquitous environment*, in K. L. D.-S. Huang and G. Irwin, editors, ICIC 2006, LNAI 4114, 355-366.

[14] A. Mileo, D. Merico, and R. Bisiani, *Support for context-aware monitoring in home healthcare,* J. Ambient Intell. Smart Environ. 2, 1 (2010), 49-66.

[15] M. J. Santofimia, S. E. Fahlman, X. del Toro, F. Moya, J. C. Lopez, *A semantic model for actions and events in ambient intelligence*, Engineering Applications of Artificial Intelligence, 24, 8 (2011), 1432-1445, ISSN 0952-1976, 10.1016/j.engappai.2011.05.008.

[16] G. Okeyo, L. Chen, H. Wang, and R. Sterritt, *Ontology-Enabled Activity Learning and Model Evolution in Smart Homes*, Proc. Seventh Int'l Conf. Ubiquitous Intelligence and Computing, 2010.

[17] L. Chen and C.D. Nugent, *Ontology-Based Activity Recognition in Intelligent Pervasive Environments*, Int'l J. Web Information Systems, 5, 4 (2009), 410-430.

[18] T. G. Stavropoulos, D. Vrakas, D. Vlachava, and N. Bassiliades, *BOnSAI: a smart building ontology for ambient intelligence,* In Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics (WIMS '12). ACM, New York, NY, USA, , Article 30 , 12 pages.

[19] I. Panagiotopoulos, L. Seremeti, and A. Kameas, *An Alignable User Profile Ontology for Ambient Intelligence Environments*, Intelligent Environments (IE), 7th International Conference, 25-28 July 2011, 270-276.

[20] V. Miori, D. Russo and M. Aliberti, *Domotic technologies incompatibility becomes user transparent*, Communication of ACM 53,1 (2010) 153-157.

[21] F. Bellifemine, G. Rimassa, and A. Poggi, *JADE - A FIPA-compliant agent framework*, Proc. of the 4th International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agents, London, 1999.

[22] M. Horridge, and S. Bechhofer, *The OWL API: A Java API for OWL Ontologies,* Semantic Web, 2, 1(2011), 11-21

[23] P. N. Tan, M. Steinbach and V. Kumar, *Introduction to Data Mining, (First Edition)*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[24] J. Dougherty, R. Kohavi, M. Sahami, *Supervised and unsupervised discretization of continuous features*, Proceedings 12th International Conference on Machine Learning, Lake Tahoe, CA, Morgan Kaufmann, Los Altos, CA (1995), pp. 194–202.

[25] A. Prieditis, S. Russell (Eds.), Proceedings 12th International Conference on Machine Learning, Lake Tahoe, CA, Morgan Kaufmann, Los Altos, CA (1995), pp. 194–202

[26] D. Bruckner, C. Picus, R. Velik, W. Herzner and G. Zucker, *Hierarchical Semantic Processing Architecture for Smart Sensors in Surveillance Networks*, IEEE Trans. on Industrial Informatics, 8, 2 (2012), 291-301.

[27] Y. H. Chen, S. J. Horng, R. S. Run, J. L. Lai, R. J. Chen, W. C. Chen and Y. Pan, *A Novel Anti-Collision Algorithm in RFID Systems for Identifying Passive Tags*, IEEE Trans. on Industrial Informatics, 6,1 (2010), 93-104.

[28] B. Wang and M. Ma, *A Server Independent Authentication Scheme for RFID Systems*, IEEE Trans. on Industrial Informatics, 8, 3 (2012), 689-696.

[29] T. Shinozaki, M. Ostendorf, *Cross-validation and aggregated EM training for robust parameter estimation*, Computer Speech & Language, Volume 22, Issue 2, April 2008, Pages 185-195.