Situation-awareness as a Key for Proactive Actions in Ambient Assisted Living

Alencar Machado[1,2], Ana Marilza Pernas[4], Iara Augustin[3], Lucinéia Heloisa Thom[1], Leandro Krug Wives[1] and José Palazzo Moreira de Oliveira[1]

[1]*PPGC, Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, Brazil*
[2] *Politécnico, Universidade Federal de Santa Maria (UFSM), Santa Maria, Brazil*
[3] *Programa de Pós-Graduação em Informática, Universidade Federal de Santa Maria (UFSM), Santa Maria, Brazil*
[4] *Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas (UFPEL), Pelotas, Brazil*

Keywords: Situation-aware, Ambient Assisted Living, Context-aware, Proactive Actions.

Abstract: With the augmentation of population's life expectancy as a whole, there is a need for intelligent applications to assist citizens in their daily activities. Ambient Assisted Living – AAL – are emerging as a way to allow technology and medical assistance to help people who need special supervision, providing support to medical emergencies. AAL are equipped with ubiquitous technologies, and use sensors as their main element for environmental data collection, providing systems with updated information. To offer personal home assistance, these computer-supported environments detect situations of interest such as the patient´s current health state, and proactively act to adapt the home environment accordingly to the patient´s specific needs. This paper presents results from an approach to support adaptive behavior in AAL. In particular, we discuss the design of intelligent systems for monitoring and adaptation of AAL. Additionally, we describe a middleware for the management of pervasive applications, which is capable of detecting the current situation of a citizen and identifying the most suitable action.

1 INTRODUCTION

Recent studies show that population is aging (Christensen et al., 2009). Diseases associated with ageing or decreased ability to perform individual activities make people with reduced cognition more prone to risky situations in their daily activities. Proactive Ambient Intelligence could improve the quality of life in a residence, providing a semi-automated assistance for people who reside in it. Ambient Assisted Living (AAL) are environments equipped with ubiquitous technologies, which provide technology in a non-invasive way, offering contextual data needed to achieve environment´s adaptations that are specific to personal preferences (Sun et al., 2009).

The inclusion of context-aware systems (Dey and Abowd, 2006) in AAL makes these systems more intelligent, proactive and reactive to support the life of citizens with decreased cognitive ability. The identification of the current citizen's situation in different contexts helps the adaptation of environments to individual features. Situation-aware behavior is especially useful in AAL, where the situation abstraction allows the modeler or pervasive application (appPerv) designer to manage each piece of the world being represented in order to deduce the specific situations in which each entity being evaluated is, or to detect a situation change (Ye et al., 2012).

To design situation-aware applications in AAL, it is necessary to model the infrastructure of these environments. Thus, this paper presents a conceptual model to determine the vision of world for appPerv, allowing triggering proactive actions according to the situations detected. To enable these features, we developed a middleware that provides the management of *SItuations as a Service* (SIaaS) for any appPerv being developed. The proposed SIaaS middleware is able to manage the context and the situations of interest for an appPerv. We also developed a solution to build generic appPerv (deployed in SIaaS) to be executed in specific domains.

We have applied our approach in a specific case study that is based on an agitation behavior model. In fact, it is a process model that describes an elderly person presenting mood and behavior changes. In

DOI: 10.5220/0004418004180426
In *Proceedings of the 15th International Conference on Enterprise Information Systems* (ICEIS-2013), pages 418-426
ISBN: 978-989-8565-60-0

particular, an appPerv based on this process model was implemented. To support appPerv, we have implemented a SIaaS prototype. The prototype can detect situations of interest defined by the appPerv and is able to send proactive actions supported by simulated environment. This paper is organized as follows: section 2 discusses our vision of ubiquitous environments; section 3 presents the proposed architecture for deploying and running appPerv based on our approach; section 4 presents the appPerv developed to analyze the case study already discussed; section 5 present our conclusions and future work.

2 UBIQUITOUS ENVIRONMENTS

In a ubiquitous environment, sensors represent objects in the world. In this work, the worldview is obtained by information captured by the sensors but the action in the world is achieved by functionalities implemented as Web services that are embedded in objects as mobile phones, televisions, microwave ovens, among other incorporated in the environment**.**

The ubiquitous environment can be characterized as a flow for aggregation of data that the sensors collect, thus seeking to express what is happening. From a high-level of aggregation, called as "situation", it is possible to act on the environment through capabilities that electronics provide, thus seeking to automate the environment according to the detected situation and preferences of citizens.

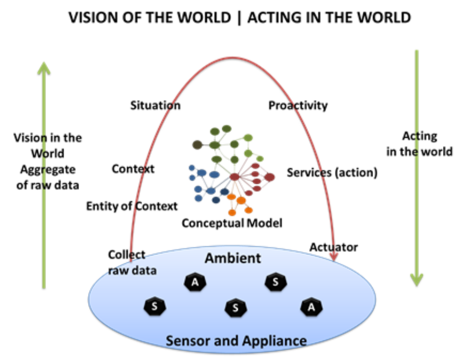

Figure 1: Viewing and Acting in the World.

These concepts are represented in Figure 1, where the arrow means the flow for: (i) aggregate raw data collected from sensors, to (ii) process this data, and to (iii) infer the world and the current situation. The part most accentuated of the curved arrow (Situation/Proactivity) contains algorithms that employ the services to act proactively in the environment according to a detected situation. These algorithms are inside the appPerv and implement solutions for managing specific situations.

2.1 Defining Situation

Proactive systems in smart environments act on behalf of the user. The key problems of proactivity in computer-supported systems are (i) to meet adequately and satisfactorily the needs of the user, without explicitly or intrusively showing this control and (ii) to allow users to maintain the perception of control about the environment. To meet these requirements, we consider situation-awareness as the key element for designing the system.

To ensure an accurate understanding of the proposed model and how the user's situation is represented, some definitions are illustrated in Figure 2 and explained as: Sensor (S): According to Compton et al. (2009), sensors are observers of the physical quality (temperature, depth, among others) of a resource; they also publish these observations. These are represented in Figure 2 by rectangles; Context Data (D): Raw data captured from the environment without semantic characterization. These data connected to an entity become abstract or physical quality. In Figure 2, context data is represented by solid circles; Entity (E): Concrete or abstract concepts used to reason about a domain of interest. For example: person, place, time, sensor, electronic device and appliance. They are the source under which data is captured and contextualized Entities are represented by circles with the letter E inside, in Figure 2.

Dey and Abowd (2006) characterize context as the situation of an entity. In the present work, the environment is represented by a set of entities and their semantic relations which characterize the context of the environment. All appPerv that make subscriptions in the proposed SIaaS middleware are interested in a subset of the context, which is known as the context of interest of an appPerv, as shown in Figure 2. The context of interest of an application is all contextualized data (entities and their data) and their semantic relationships. Following this definition, each appPerv has its special Context of Interest (**C**), which represents a group of semantic relations that are valid in a specific moment.

Thus, we represent the context of interest from a specific appPerv by the following statement (1):

$$C: (app, \{R\}) \tag{1}$$

Where: *(C)* Context of Interest; *(app)* a specific appPerv; *{R}* consists of a set of semantic relations, each one represented by <Es, P, Eo> where *Es*

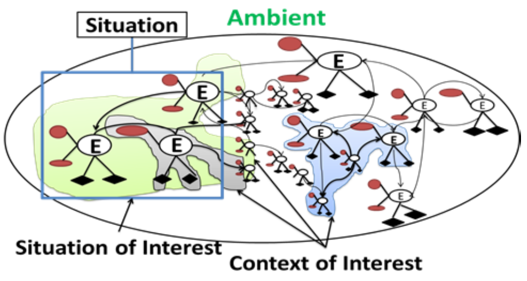

Figure 2: Representation of Ambient, Situation, Context, Context Data, Sensor and semantic relations.

are entities that are contextualized and are the subject of the relationship; *P* are predicates (relations) and *Eo* are entities that are and are the object of the relationship. Therefore, the context of interest is a subset of contextualized data (entities, their data and semantic relations) and the application itself, i.e. the information of interest to an application for making decision on which proactive action triggers.

A situation of interest is viewed as the state of the entities that characterize an event in the environment. It is composed by *{R}* and logical rules *{L}* that, when evaluated to true, characterize a fact in the environment. A situation is formally represented by the following statement (2):

```
S: (app, {R}, {L})
```
(2)

Where *(S)* Situation of Interest; *(app)* represents a specific appPerv; *{R}* set of semantic relations presented in (1) and *{L}* logical rules. The logical rules *{L}* are defined according to each situation, and are related with specific values that are relevant in the activity of AAL. For example: heartbeat > 100 per minute. A formalized situation rule *{L}* related with the ontology build for this work (in OWL language) is presented inside the Environmental Architecture, in subsection 3.2.

2.2 Defining of Proactive Actions and Services

An environment can be impregnated of sensors for data collection. Sensors are aggregated to provide relevant information about the current situation. In addition, the ambient can provide services (actions) for the citizens who are or live in it. Objects can provide specific actions to adapt the environment based on the citizen's needs. In this work, services represent features (functions) of Entities inserted in the environment. These features are provided by device themselves or obtained by suppliers of AAL services**.** Services are defined as statements containing inputs (*I*) and one output (*O*). A service is formally represented by the following statement (3):

```
Se: ({I}, O);
```
(3)

For example, a service to warn the user using sounds could receive, as input, the specific device to be used in a service and the warning message to be reproduced, as presented below:

```
Se:  warningSound;
{I}: Radio_Device, "take drug";
O:   Device.
```

The set of services {*Se*} for Proactive Actions is organized in a logical order (i.e., linked list) of execution, where the logical order matches corresponding actions in the environment, and can be grouped under some logical aspect (e.g., location, run time).

A proactive action is formally represented by rule (4), as follows:

```
PA: ({Sp}, {Se}, {R});
```
(4)

Where: *{Sp}* consists on a situation that must be true for the correct execution of proactive actions; *{Se} is a* set of Services and *{R}* set of semantic relations presented in (1). For instance, a proactive action can have agitation as a pre-condition (*Sp*). This action sends a SMS for a caregiver (Service 1) and calls emergency (Service 2). The relations that determine interactions with the device to activate the service are represented by semantic relations {*R*}, as exemplified below:

```
(hasService(Device_Radio,Song));
(hasService(Device_TV,Image));
```

3 ARCHITECTURE FOR DEPLOYING PERVASIVE APPLICATIONS

Knowing the existing complexity in ubiquitous environments with heterogeneous technologies of sensors and electronic objects, the design of an architecture that reduces the complexity to generate intelligent applications (pervasive) becomes necessary. This article proposes a software architecture to enable the execution of appPerv in the described AAL. As represented in Figure 3, the entities of the environment are invariant in their normal state (a standard state with semantic links pre-defined as stable). Occasionally, unwanted situations may become true. These are situations that are prerequisites for firing automatic actions in the environment. But the decision of which action should be triggered in the environment due to some detected situation is a task of appPerv.

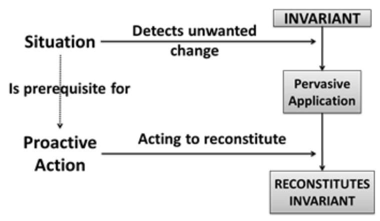

Figure 3: Flow to Trigger Proactivity Actions.

They contain the knowledge to trigger the most suitable action to a particular situation. The proactive action is triggered by the appPerv itself, which should attempt to reconstruct the invariant to its normal state. To support this flow, a service-oriented architecture is employed (see Figure 4), which provides a stable environment to install third-party solutions (appPerv).

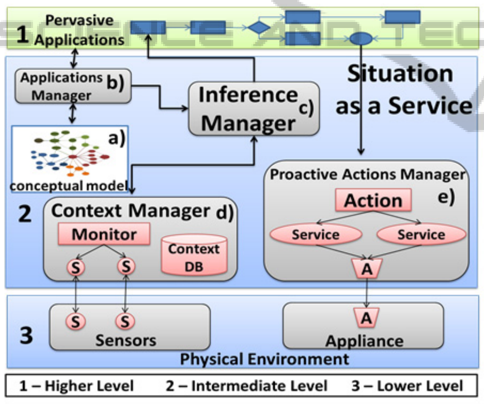

Figure 4: Architecture for Pervasive Applications.

3.1 Physical Environment

The **Lower Level (3)** layer in Figure 4 corresponds to the physical environment containing a sensor network infrastructure and providing means for monitoring and capturing raw data from the ambient. This level includes the physical infrastructure of the residence, with standardization of protocols for communication and home automation (Domotic / Home Automation) (Gill et al., 2009).

3.2 Situation as a Service (SIaaS)

The **Intermediate Level (2)** contains the software layer (SIaaS middleware) to provide a stable and secure environment for pervasive applications (appPerv) requesting actions for the SIaaS. It notifies the appPerv whenever situations of interest for the application become true. This layer monitors the environment and provides contextualized information (context of interest) according to the interest of the appPerv. This layer builds a link between the physical environment and the appPerv. SIaaS provides interfaces to context information that is generated and to the services available. So, it can be used by third-party applications to design solutions that could be incorporated in the residential architecture.

Conceptual Model: The conceptual model ('a' in Figure 4) is an ontology that describes everything in the environment (context model). The ontology is developed in OWL file (Ontology Web Language) (Bechhofer et al., 2004), sublanguage OWL-DL. It requires concept specialization at the moment of an inclusion of the terms that describe the specific residence managed by the SIaaS middleware. Their instances reflect the real entities in a particular home domain.

Through a copy of this file, the applications can identify which types of entities and semantic relations can be handled. Thus, when the application is built, it can be preprogrammed so that the behaviors become sensitive in relation to this conceptual model. The Situations are implemented as rules in Semantic Web Rule Language (SWRL), which consists on Horn-like rules added to the OWL language as axioms in OWL-DL (Horrocks et al., 2005). The rules are defined in the ontology and reason over the current values of existent instances. The valid values defined in the relationships will make the difference between a situation *S#1* or *S#2*. For this situation, we have the following statement in FOL (First Order Logic), according to the OWL notation defined in (Horrocks et al., 2005), where "EC" and "ER" means concept and relationship respectively.

```
(∀p, s, v, y, w ) ∈ EC(OWL : Thing)
       ( (p) ∈ EC (Patient)
    ∧ (s) ∈ EC (SensorHeartBeat)
     ∧ (p,s) ∈ ER (hasSensor)                    (5)
   ∧ (s,v) ∈ ER (valueCollected)
    ∧ (v,y) ∈ ER (greatherThen)
    → (p,w) ∈ ER(isSituationOf))
```

The above rule (5) translated into SWRL is presented in (*6*).

```
       Patient(p)^SensorHartbeat(s)
    ^hasSensor(p,s)^valueCollect(s,v)
        ^swrlb:greaterThan(v, 40)                (6)
 →isSituationOf(p,Emergency_Situation);
```

Related with the situation rule in (2), section 2.1, where the logical rules $\{L\}$ are represented, in the example, by the SWRL operator in rule (*6*). Rules are defined according to the critical values defined to each patient, which is formulated when the appPerv development and instantiated by the Subsystem to manage appPerv.

To *services* related with the rule in (3) and (4), section 2.2, a proactive action is represented by a started situation, a set of services (which implement the action) and a group of semantic relations (that must be true for this proactive action). The set of services is also described in the ontology (using Semantic Markup for Web Services - OWL-S). In (7) and (8) we show rules, also in FOL, that represent the services and their respective group of semantic relations and in (9) the composition these services through a proactive action.

(7): service to play music in a radio device

```
  (∀a, r, m, z) ϵ EC(OWL : Thing)
      ( (a) ϵ EC (PlayMusic)
        ∧ (r) ϵ EC (Radio)
        ∧ (m) ϵ EC (Music)                    (7)
    ∧ (r, a) ϵ ER (hasService)
      ∧ (a,m) ϵ ER (hasInput)
    ∧ (a,r)   ϵ ER (hasOutput)
  → (a,z) ϵ ER(executeService))
```

(8): service to call an ambulance of health provider

```
(∀ c, h, p, e, k) ϵ EC(OWL : Thing)
     ( (c) ϵ EC (CallAmbulance)
    ∧ (h) ϵ EC (HealthProvider)
        ∧ (p) ϵ EC (Patient)
  ∧ (e) ϵ EC (Status_Emergency)
    ∧ (h,c) ϵ ER (hasService)                 (8)
     ∧ (c,h) ϵ ER (hasInput)
     ∧ (c,p) ϵ ER (hasInput)
   ∧ (c,h)   ϵ ER (hasOutput)
  → (c,k) ϵ ER(executeService))
```

Proactive Action (in (9)): if emergency situation (*5*) is detected in the environment, it will trigger the service to play music (7) and call an ambulance on the health provider (*8*).

```
  (∀ f, a, c, w, v, z, w, o) ϵ
        EC(OWL : Thing)
  ( (f) ϵ EC (ProactiveAction)
      ∧ (a) ϵ EC (Service)
      ∧ (c) ϵ EC (Service)
    ∧ (w) ϵ EC (Situation)
    ∧ (f,z) ϵ ER (hasInput)                   (9)
    ∧ (f,k) ϵ ER (hasInput)
∧ (f,w) ϵ ER (hasPrecontition)
∧ (w,v) ϵ ER (situationDetect-
                ed)
            → (f,o) ϵ
  ER(executeProactiveAction))
```

Subsystem to Manage Pervasive Applications: (SMPA) ('b' in Figure 4) registers the appPerv acquired to the user´s residency. For that, it verifies if the OWL file, which contains the situation and context of interest provided by the appPerv, has the same structure of the ontology (OWL file of the SIaaS) that describes the environment. It also tests the consistency in the OWL file provided by appPerv. Finally, it registers the application with their situations and context of interest in the Subsystem to Manage Inferences.

Subsystem to Manage Inferences: (SMI) ('c' in figure 4) is part of the worldview (section 2). This subsystem processes the conceptual model each time that it is notified by the SMC.

Subsystem to Manage Context: (SMC) ('d' in Figure 4) is part of the worldview (section 2). This subsystem receives requests of the SMI containing appPerv and its context of interest. It monitors this context and detects possible changes.

Subsystem to Manage Proactive Actions: (SMPAct) ('e' in Figure 4) is part of the actions of the world (section 2). This contains a repository of semantically described actions, a set of services that can be requested by appPerv. This subsystem receives notifications for the execution of one or more proactive. These actions result in the consumption of services, which are features from the electronic devices which are in the physical environment. This semantic description links the services to its respective device or companies (e.g., Health Providers), and they are described in the conceptual model.

3.3 Pervasive Applications

The **Upper Level (1)** offers a stable computing environment for appPerv . In this research, appPerv are software applications that request trigger actions proactively due to the situations detected by SIaaS in the environment and alter their behavior due to the existing contextual information. Designers of appPerv must implement SWRL rules which correspond (when inferred) to specific situations of the environment and that are of interest of the appPerv. To inform its context of interest, the designers should generate instances of a particular type (e.g., Patient, Sensor, and make the linking semantics between them, as *hasSensor*). Therefore, during registration of the appPerv in SMPA, these instances are compared to the actual instances of the environment (existing in the conceptual model of the middleware).

4 CASE STUDY

To evaluate the feasibility of the proposed approach, the architecture was tested by developing an appPerv that uses the concepts presented so far. This application has the main goal to manage agitation moments of patients with lightweight dementias.

4.1 Case Study Scenario

For the case study, the following fictitious scenario was considered. Imagine 'John', a 78 years old citizen without need for hospitalization, but has lightweight dementias and needs continued treatment. In addition, John has some memory problems. His doctor identified he is presenting agitation behaviors. This situation causes problems for his health. The residence of John is an AAL with enough technology to assist his health management and improve his life quality as well as to adapt to his needs. To assist with John´s implications, his daughter Michelle purchased a software solution that interacts with John's residence in the case agitation situations are detected.

Thus, there is an embedded infrastructure in John's residence that provides automated electronic resource consumption. These resources are provided in terms of services that each residential object with some kind of communication skills can provide. To manage this environment, there is an automated system (SIaaS) who has knowledge of John´s needs (i.e., a conceptual model), as well as knowledge of what the environment can provide to assist those needs. This system provides residential resources for other applications can interact with the environment.

This is the scenario we have designed for a first partial validation of our approach. In this context, an appPerv, called appPervAgitation, was developed and it manages agitation situations in patients with Alzheimer disease based on the process illustrated in Figure 5. This workflow has as objective to show which action may be performed when a patient with Alzheimer disease is agitated. For designing the proposed solution, we defined relevant situations of interest within their respective context of interest and the proactivity actions, all described hereafter.

4.2 Situation of Interest

To identify the level of agitation of a given patient, *appPervAgitation* registers three situations for agitated patient, namely high, medium and low, corresponding to the formalization proposed in (2), section 2.1. The situation of interest is:

```
S: (appPervAgitation,{Ra,Rb},{S1,S2,S3});
```

where:

app = Pervasive Applications for Manage Situations of Agitation (*appPervAgitation*)

```
Ra=(hasSensor(John,Sensor_Heartbeat1);
Rb=(hasValue(Sensor_Heartbeat1,CollectValue)
```

L = High Agitated (S1):

```
Patient (John)  ∧  SensorHeartbeat (Sen
sor_Heartbeat1)  ∧  hasValue (Sen-
sor_Heartbeat1, CollectValue)  ∧
swrlb:greaterThan(CollectValue,200)  →
isSituationOf (John, Emergency_Situation)
```

L = Medium Agitated (S2):

```
Patient (John)  ∧  SensorHeartbeat (Sen-
sor_Heartbeat1)  ∧  hasValue (Sen-
sor_Heartbeat1, CollectValue)  ∧
swrlb:greaterThanOrEqual(CollectValue,90)
∧swrlb:lessThanOrEqual (CollectValue,200)→
isSituationOf (John, Emergency_Situation)
```

L = Low Agitated (S3):

```
   Patient (John)  ∧  SensorHeartbeat (Sen-
sor_Heartbeat1)  ∧  hasValue (Sen-
sor_Heartbeat1, CollectValue)   ∧
swrlb:greaterThanOrEqual(CollectValue,50)
∧swrlb:lessThanOrEqual (CollectValue,89)  →
isSituationOf (John, Emergency_Situation)
```

4.3 Context of Interest

To identify which proactive action must be triggered, the context of interest was based on the formalization presented in (1), section 2.1, the context of interest is:

```
         C:(appPervAgitation,
      {R1,R2,R3,R4,R5,R6,R7,R8});
```

where**:**

Entity Person/Patient and Caregiver: the appPerv needs to know the location of patient John and of his caregiver (Michelle) to trigger actions when situation of interest is detected. For example, play music (steps "n" and "h" of Figure 5).

```
R1 = (hasLocation(John,Location_X);
R2 = (hasCaregiver(John,Michelle);
R3 = (hasLocation(Michelle,Location_Y);
```

Entity Music Services: should contain a list of appliances (with their respective locations) that provide such service; so, the appPerv can identify if the patient is in a location close to some of these devices to execute steps "n" and "h" of Figure 5 (play music);

```
R4 = (hasMusicService(Radio_X,Serv_1);
R5 = (hasMusicService(TV_X,Serv_2);
R6 = (hasMusicService(Smartphone_X,Serv_3);
```

Entity Messaging Services: should contain a list of devices that provide this service, so that the application can perform step "i" of Figure 5.

```
R7  =   (hasSMSService(Smartphone_X,Serv_4);
```

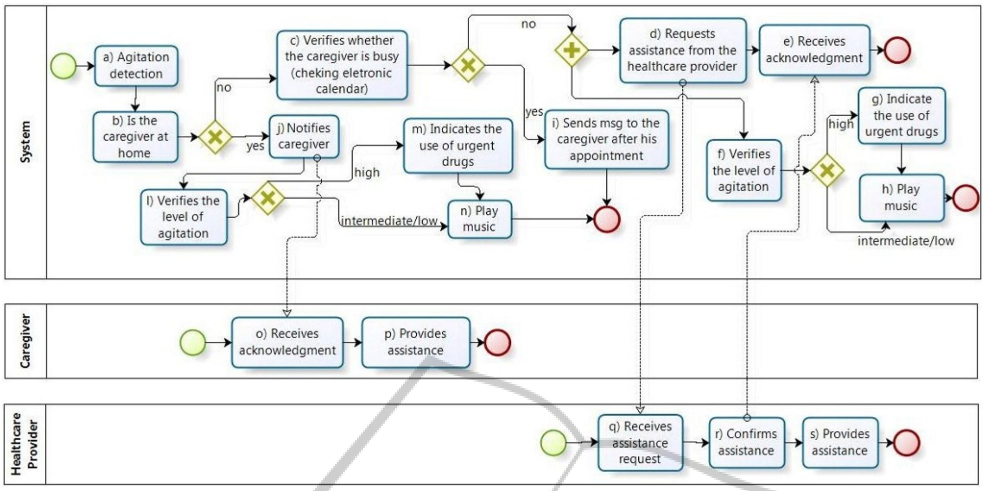
System
Caregiver
Healthcare Provider
a) Agitation detection
b) Is the caregiver at home
no
yes
c) Verifies whether the caregiver is busy (cheking eletronic calendar)
no
yes
d) Requests assistance from the healthcare provider
e) Receives acknowledgment
j) Notifies caregiver
m) Indicates the use of urgent drugs
high
i) Sends msg to the caregiver after his appointment
f) Verifies the level of agitation
high
g) Indicate the use of urgent drugs
h) Play music
intermediate/low
l) Verifies the level of agitation
intermediate/low
n) Play music
o) Receives acknowledgment
p) Provides assistance
q) Receives assistance request
r) Confirms assistance
s) Provides assistance

Figure 5: Agitation behaviour process of a patient with Alzheimer Disease (Siang Fook et al., 2006).

Entity Health Provider: must contain one or more health care providers, within their contact details, so that the application can notify them in step “d” of Figure 5.

```
R8 = (hasHealhtProvider(John,HosptalX);
```

4.4 Proactive Actions

To demonstrate the formalized proactive actions, we present below steps “n”, “h” and “d” of the Figure 5.

Pervasive Application needs consumed play music of device, based on the formalization presented in (3) and (4) section 2.2.

```
Service: playMusic;
Inputs: Device, Music;
Output: Device.
```

Service to requests assistance for health provider:

```
Service: sendMessage;
Inputs: Device_Smarthphone, Emergency;
Output: HealhtProvider.
```

Proactive Actions to manipulate the agitation situation: PA: Proactive Action for Emergency

```
{Sp}: S1,S2,S3;
{Se}: playMusic, sendMessage;
{R}:  R1,R4,R5,R6,R7.
```

This PA is described on the pool of services, this pool is managed for SMPAct and implements through an OWL-S file.

4.5 Results of Experiments

A pilot prototype for the proposed architecture was developed, considering the concepts discussed in this paper, and to evaluate the impact of SIaaS in the execution of appPervs.

Integration testing was developed to demonstrate the integration of the subsystems and follow the sequence diagram of Figure 6. In (1) the application is registered on the SIaaS. At the moment, the SMPA reads the situations of interest of the appPervAgitation (agitated high, medium, low). It also reads the context of interest (patients, caregivers, healthcare providers, music services and message). After this process, (1.1) the SMPA carries the conceptual model (CM) to (1.2) run the alignment context. In (1.3) the application together with their situations of interest and context are subscribed in the SMI. The SMI initially (1.3.1) subscribe the appPervAgitation with its context of interest in the SMC. The SMC subscribes in sensors (1.3.1.2) that are of interest to the appPervAgitation (as RFID sensor to locate the patient and caregiver) and obtains the remaining data (1.3.1.1) contained in the context database (DBCtx). The SMC (2) processes in parameterizable times the context data.

If identified any changes, (3) it publishes/notify for the SMI, informing which appPervAgitation had its context changed. In (3.1) the SMI performs the rules to detect if these situations become true. Case some situations were detected, (4) it notifies the appPervAgitation sending the context of interest and the situation detected. The appPerv is triggered/initiated by the notification of SMI. The appPervAgitation executes its workflow (4.1) and decides which proactive actions of the environment should be triggered (steps 'n', 'h', 'd', 'i' of Figure 5) to the agitated situation be handled. Thus, (4.2) the appPervAgitation asks SMPAct to execute one or more proactive actions in the environment. Later the SMPAct processes this request (4.2.1) and consumes/triggers on the correct device the desired service by the appPerv (4.2.2).

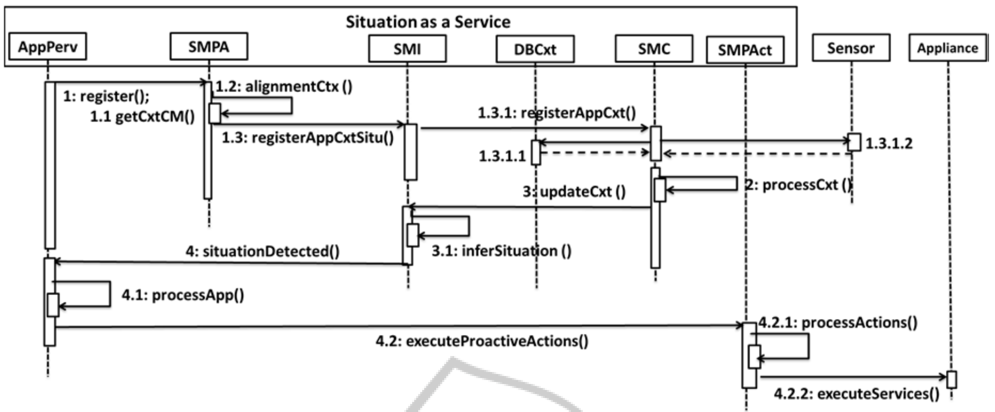

Figure 6: Sequence Diagram for proactive actions in environment.

5 CONCLUSIONS

In this paper, we presented current issues for building Ambient Assisted Living – AAL – systems.

These systems should be proactive and finely integrated with the needs of the inhabitants which living in the AAL environment. To the system be able to offer utility, usability and ubiquity, at the same time that provide management of the environment and interacts with the users, it is necessary to express situation awareness.

To support this awareness our architecture provides a Situation as a Service middleware offering a possibility of proactive action. The middleware SIaaS decreases the need to cope with heterogeneity in these environments, using a shared conceptual model and allowing its manipulation through OWL files. Using this conceptual model, developers of pervasive applications can generate software to be deployed in a standard home for specific user needs. The main contributions of this research are (i) the formalization of the environment of AAL (situation awareness and proactive actions); (ii) the design of an architecture that provides resources to run third-party pervasive applications; (iii) the use of the conceptual model to semantic interoperability; and (iv) a method to build generic pervasive applications for specific domains.

Possible directions for future research include (i) the study more advanced of context model in AAL to provide wide usage of semantic to Pervasive Applications in the domotics industry; and (ii) the expansion of the prototype to real physical environment.

REFERENCES

Bechhofer, S.; van Harmelen, F.; Hendler, J.; Horrocks, I.; McGuinness, D.L.; Patel-Schneider, P.F.; and Stein, L.A. (2004) *OWL Web ontology language 1.0 reference. W3C proposed recommendation* (www.w3.org/TR/owl-ref).

Bettini, C., Brdiczkab, O., Henricksen, K., Indulskad, J., Nicklase, D., Ranaganatha, Riboni, D. (2010) *A survey of Context Modelling and Reasoning Techniques*. Pervasive and Mobile Computing, 161-180.

Christensen, K., Doblhammer, G., Rau, R. & Vaupel, J. W. (2009) *Ageing populations: the challenges ahead*. Lancet 374, 1196–1208.

Compton, M.; Henson, C., Lefort, L., Neuhaus, H.: (2009) *A survey of the semantic specification of sensors*. In 2nd International Workshop on Semantic Sensor Networks, CEUR-WS, 17-32,2009.

Dey, A., and Abowd, G., (2006) *The Context Toolkit: Aiding the Development of Context-Aware Applications,* In Proceedings of Human Factors in Computing Systems: Pittsburgh, PA: ACM Press, pp.434-441.

Friedman-Hill, E. (2003): *Jess in Action: Java Rule-based Systems*, Manning Publications Company, June, ISBN 1930110898, http://herzberg.ca.sandia.gov/jess/.

Gill, K. S, Yang, F. Yao, and X. Lu, (2009) *A ZigBee-based home automation system*. Consumer Electronics, IEEE Transactions on, vol. 55, no. 2, pp. 422–430.

Horrocks, I., Patel-Schneider, P. F., Bechhofer, S., Tsarkov, D. (2005) *OWL Rules: A Proposal and Prototype Implementation*. Journal of Web Semantics, v.3, p. 23.

Klusch, M, B. Fries, and K. Sycara. (2009) *OWL-S-MX: A hybrid Semantic Web Service matchmaker for OWL-S services*, Web Semantics: Science, Services and Agents on the World Wide Web, 7(2):121-133.

Moore, P., Hu, B., Zhu, X., Campbell, W., Ratcliffe, M. (2007) *A Survey of Context Modeling for Pervasive Cooperative Learning*. In: Internet Symposium on Information Technologies and Applications in Education - ISITAE 07, Kunming, Yunnan, China, p. 23-25.

Sun, H. De Florio, V. Gui, N. Blondia, C. (2009), *Promis* *es and Challenges of Ambient Assisted Living Systems*, Sixth International Conference on Information Technology: New Generations, ITNG ’09. pp. 1201-1207.

Siang Fook, V. F., Tay, S. C., Jayachandran, M., Biswas, J., and Zhang, D. (2006). *An ontology based context model in monitoring and handling agitation behaviour for persons with dementia*. PERCOMW ’06, Washington, DC, USA. IEEE Computer Society.

Ye, J., Stevenson, G., Dobson, and S. McKeever, (2012) *Situation identification techniques in pervasive computing: a review,* Pervasive Mobile Computing, pp 36-66, Vol. 8, Issue 1, February 2012.

Ye, J., Stevenson, G., Dobson, S. (2011) *A top-level ontology for smart environments*. Pervasive and Mobile Computing. v. 7, p. 359-378.