A Flexible Agent-Oriented Solution to Model Organisational and Normative Requirements in Assistive Technologies

Ignasi GÓMEZ-SEBASTIÀ, Dario GARCIA-GASULLA, Cristian BARRUÈ, Javier VÁZQUEZ-SALCEDA and Ulisés CORTÉS [a,1]

[a] *Knowledge Engineering and Machine Learning Group*

Abstract. Assistive technologies represent a recent application area of a wide variety of Artificial Intelligence methods and tools to support people in their activities of daily living. But most approaches do only center in the direct interaction between the user and the assistive tool, without taking into consideration the important role that other actors (caregivers, relatives) may have in the user activities, nor they explicitly reflect the norms and regulations that apply in such scenarios. In this paper we present an approach to the development of assistive technologies which uses organisational and normative elements to ease the design of both the social network arround the user and their expected behavioural patterns.

Keywords. Assistive technologies, personalised services, normative systems, normative agents, agent-oriented software design.

Introduction

Assistive technologies are an application area where Artificial Intelligence methods and tools have been applied to support people in their daily life. Most approaches tend to center in the user-assistive tool interaction, though. But next generations of assistive technologies should provide ways to connect users with caregivers and relatives, and provide ways to monitor unwanted behaviour.

In this paper we present an approach to the development of distributed assistive technologies using organisational and normative elements to include all relevant actors related to a given patient and their role in the patient's activities. We will use as example an scenario where the technology should facilitate some necessary and periodic tasks which would require an impeded patient to leave his/her house. The scenario focuses on those tasks that allow a patient to be periodically controlled by a doctor and to obtain his/her medication, according to the treatment assigned. The scenario is inspired in the assistive technology developed in the SHARE-it [9] project (§1).

This scenario presents several tasks that must be coordinated by a range of actors in a in a highly regulated environment where the roles for some of the actors and the

[1]Departament de Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya (UPC) C/Jordi Girona 1-3, E-08034 , Barcelona, Spain. E-Mail:**{igomez, dariog,cbarrue,jvazquez,ia}@lsi.upc.edu**

way they may interact with patients (e.g. who can change the patient medication) are clearly stated. This complex scenario can benefit from the combination of norm-aware, organisational-aware multiagent techniques and Semantic Web Services to enable dynamic and regulated service composition. What's more, technologies involving organizational and coordination theories applied to Web Services are also key in order to effectively maintain a system operating in such a constrained and dynamic environment. We will use the ALIVE [5] framework and its tools to model and develop this scenario, as it has been already successfully applied to other dynamic and complex scenarios [8,7].

The rest of this paper is structured as follows: the paper summarizes an use case inspired on the *SHARE-it* project in §2. Later, the ALIVE framework is briefly explained in §3. The paper goes on by describing the application of the ALIVE framework to the use case. Finally authors state their conclusions and lines of future work.

1. SHARE-it

The main goal of *SHARE-it* (Supported Human Autonomy for Recovery and Enhancement of cognitive and motor abilities using information technologies) project was to contribute to the development of the next generation of intelligent and semi-autonomous assistive devices for older persons and people with disabilities (both cognitive and/or motor). The focus of *SHARE-it* was to develop a scalable, adaptive system of add-ons composed of sensor and assistive technologies so that they can be modularly integrated into an intelligent home environment to enhance the individual's autonomy.

The basic assumption in this research effort is that the target population by using these supporting devices could be self-dependent enough as to autonomously live in their community, staying at home as long as possible with a maximum safety and comfort; this possibility would increase their Quality of Life, and, at the same time, delay their institutionalization. At least in part, *how well* these elders live and their abilities to maintain independent life styles will depend on their health and the degree to which they have remained totally able or *frail* or *disabled*. This in turn, will depend to at least some extent on *how well* the artificial and built environments in which they live conform to their needs and their age-related losses in abilities and somatic integrity.

Mobility is a key aspect of the *SHARE-it* project. As a result, four different assistive platforms (i-Walker [3], Spherik, CARMEN [2] and Rolland [6]) were developed and deployed in real environments, whose objective is to be able to provide mobility assistance for wide range of users, paying particular attention to the needs of elderly and impaired people. These platforms were tested in a special Ambient Intelligent environment, the *Casa Agevole*, a house specially designed for people with disabilities and equipped with a wide array of domotic capabilities and sensors.

In order to combine and manage the information from the ambient intelligence, the robotic platforms actuators and functionalities and offer a set of cognitive services to the users, a multi-agent system was developed as control middleware. The *SHARE-it* agent layer architecture focuses on delivering three main kind of services: monitoring, navigation support and cognitive support.

2. Use Case

The use case we present in this paper, called *Alberto takes his drugs*, extends the one originally presented in [1]. In it we model a scenario with a physically disabled person (*Alberto*), whose disability makes it very difficult for him to leave his house regularly. The problem we try to solve is supplying *Alberto* with his required medications without him leaving his house, as well as facilitating the interaction with his doctor. The main stakeholders of the use case are: 1) *Alberto*, as the patient. 2) *Doctor*, as the responsible person for his medical treatment. 3) *Health Insurance Company*, as the entity organizing and controlling the interactions. 4) *Pharmaceutic*, as the medication retailer. 5) *Logistics*, as the person/company responsible for home delivery. 6) *Domotic House*, the house Alberto lives in, which can automatically open the main door, granting entrance to authorized people. 7) *Medical Dispenser*, a device which automatically provides medications doses (*i.e.* pills). 8) *Medical Monitor*, a device controlling the Medical Dispenser embedding it with intelligent behavior.

The scenario starts with a visit of *Alberto* to his doctor, who assigns a periodical treatment. The Doctor sends the authorized treatment to the Medical Monitor which automatically communicates with the Logistics (the monitor should be able to choose the most suitable Logistics person taking into account delivery routes, proximity and workload), requesting the supply of the medication to *Alberto*. the Medical Monitor communicates with the Pharmaceutic as well (again, choosing the most suitable one, taking into account pharmacy locations and medication stocks), requesting the handover of the medication to the Logistics. The Medical Monitor also grants permissions to both of them to do their task (*i.e.* electronic medical recipes). Once the Logistics has fetched the medication from the Pharmaceutic, goes to Alberto's house and is identified and granted access by the Domotic House. Once inside *Alberto*'s house, Logistics fills the Medical Dispenser with the requested medication. The Medical Monitor detects the update of the medication stock quantity, and keeps monitoring it to avoid stock breaks.

Once the Medical Monitor detects an incoming lack of medication (given the current stock in the Medical Dispenser and the daily medication dose specified by the Doctor) it communicates with the doctor, requiring it to approve an extension of *Alberto*'s treatment. The Doctor may approve the renewal or change to a new one, in which case orders and permissions are automatically sent to the Logistics and the Pharmaceutic by the Medical Monitor, or arrange a meeting with *Alberto* if required.

The Health Insurance Company imposes some rules and restrictions on the process. For example, it requires that the Doctor, the Logistics and the Pharmaceutic fulfill their obligations within a established time gap. In case some of those actors take too long to perform their tasks and violate their deadline, a sanction is issued and measures are taken to make sure *Alberto takes his drugs*. On the other hand, it requires some basic safety rules are accomplished, such as checking medication's *best before date* before refilling the Medical Dispenser.

3. The ALIVE framework

The ALIVE framework combines model-driven design techniques and agent-based system engineering with organisational and coordination mechanisms effectively provid-

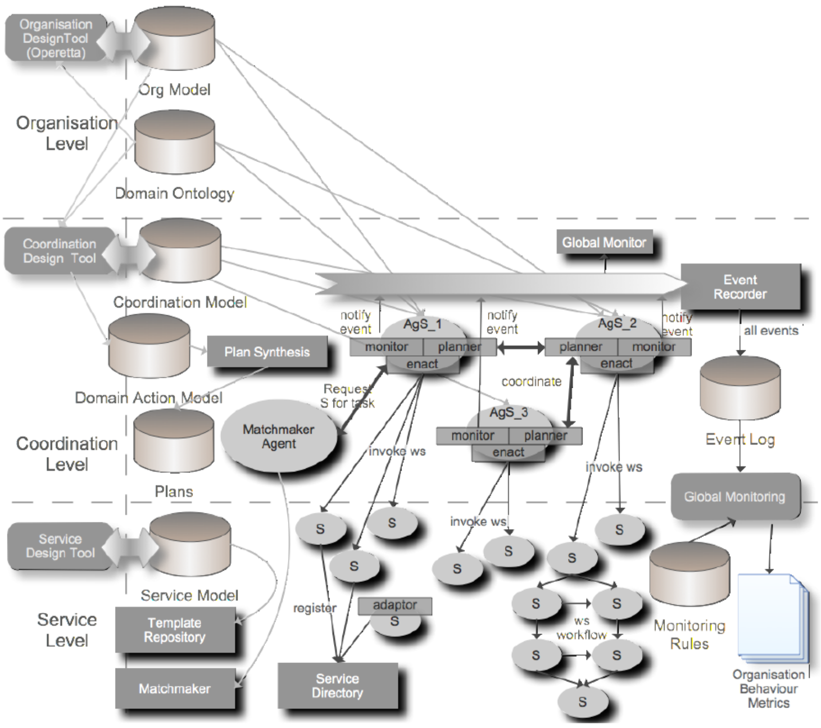

Figure 1. ALIVE architecture [4] (S stands for Service)

ing support for *live* and open system of flexible, service-oriented systems. On the one hand ALIVE's multi-level approach helps on designing, deploying and maintaining distributed systems, by combining re-organising and adapting services. On the other hand, ALIVE's organisational and normative structures make it suitable for highly regulated scenarios like the one presented in this paper. The ALIVE framework applies substantive norms that define commitments agreed upon actors and are expected to be enforced by authoritative agents, imposing repair actions and sanctions if invalid states are reached. Substantive norms allow the system to be flexible, by giving actors (human or computer-controlled) the choice to cause a violation if this decision is beneficial from an individual or collective perspective.

3.1. Organisation Level

The organizational level represents the organizational structure of the system via the organizational model (Org. model). The organizational model is formalized following the Opera methodology [10], including the following concepts: 1) Objectives: States of the world pursued by actors. Derived from organisation's goals. Examples of objectives on the use case presented include *Fill dispenser* and *Have medication*. 2) Roles: Groups of activity types played by actors (either agents or human users). Examples of roles on the use case presented include *Pharmacy* and *Medical Dispenser*. The set of roles and the relations between them constitutes the *Social Structure*. 3) Landmarks: Represent important states of the world regarding the achievement of goals. They are identified by the set of propositions that are true on the state of the world represented by the landmark. Examples of landmarks on the use case presented include *Request Medication* and *Visit Doctor*.

Apart from the concepts inherited from the Opera methodology, the organisation level supports the definition of norms, effectively adding a normative structure to the social and interaction structures. The normative structure is useful for imposing patterns of behavior and model highly regulated scenarios. The elements on the normative structure contain the following main components, expressed using Partial State Descriptions of the world: a) Activation Condition: when the world reaches the state specified in this condition, the norm starts to be checked. For instance, the obligation to *check best before date* is not active until the goal *provide medication* has been activated. b) Expiration Condition: when the world reaches the state specified in this condition, the norm stops to be checked, and has not been violated. For instance, the obligation to *check best before date* is not active anymore when the goal *check best before date* has been fulfilled, and the norm has not been violated. c) Maintenance Condition: when the world reaches the negation of the state specified in this condition, the norm stops to be checked, and has been violated. For instance, the obligation to *check best before date* is not active anymore when the goal *Provide medication* has been fulfilled, and the norm has been violated, as the goal *check best before date* has not been fulfilled before. *Figure* 4 shows an example of an ALIVE norm.

3.2. Coordination Level

The coordination level provides the patterns of interaction among actors, transforming the organisational model into coordination plans, or work-flows. Work-flows bring the system from the state represented by a landmark to the next one (as defined on the interaction structure) and are formed by chains of tasks. Tasks, as defined on the *task model*, contain both pre and post-conditions, that define the state of the world before and after the task is performed. Tasks contain as well semantic information that binds them to abstract services on the service level.

The plan synthesis uses information from the organisation model, domain ontology and tasks models in order to generate the work-flows the agents will enact. These work-flows are stored in a repository, from where they will be retrieved when required.

A set of intelligent agents, deployed on the AgentScape platform[2] enacts the work-flows in a distributed and coordinated fashion. Agents analyse and monitor work-flow execution and react to unexpected events, either by enacting alternative work-flows or by communicating the incident to other levels.

3.3. Service Level

Appropriate services are selected for each abstract task composing the work-flows by matching the semantic information contained on the service description and on the task description. These descriptions are defined in terms of OWL-S[3] service profiles. OWL-S profiles facilitate the process of composing services (effectively matching a chain of ser-

[2]http://www.iids.org/research/aos
[3]http://www.w3.org/Submission/OWL-S/

vices to a given task) and finding alternative ones (effectively matching a set of alternative services to a given task, and reassigning on the fly services to tasks if a given service is not available).

3.4. The Monitor Tool

The monitor tool is the back-bone of the ALIVE framework, connecting all the three levels allowing the exchange of events among them, from a service invocation that fails to an update on the Organisational design (*e.g.* a new role or objective is introduced) that affects the agents in the coordination level. Agents enact their roles by interacting either via direct communication (coordinating among themselves) or via service invocation. The monitor tool observes these interactions and matches them with the normative and organisational states (*e.g.* Obligations, Permissions, Roles) effectively allowing agents to reason about the effects (in a normative sense) of their actions.

4. Use Case Model

This section models the use case introduced in §2 using the ALIVE framework. The section introduces the model from the Organisation, Coordination and Service level perspectives. Elements from the domain ontology are referenced as well when required.

4.1. Organisation level model

We will start by modeling the organisation level, that is identifying the existing set of stakeholders, the goals of each of them, landmarks and scenes related to those goals and the normative structure of the system. We will start with the stakeholders, which are extracted from the use case explanation: *Client, Doctor, HealthInsuranceCompany, Pharmacy, Logistic, Domotic_House, Medical_Dispenser, Medical_Monitor*.

For each *role*, its *goals* within the use case have been identified. Also a hierarchical relation has been defined among the whole set of goals, such that, for example to fullfil the objective *Prepare_deliver* first permissions must be given to both the *Logistic* and the *Pharmacy* and a minimum *best before* date must be established for the requested medication. Each of these objectives is assigned a state description, representing the state of the world when the objective has been fulfilled. For instance, the objective *Reach_pharmacy* has been fulfilled when the logistics person A has been given the order to go to a given pharmacy P and A and P are in the same geographical coordinates. *Figure* 2 shows the roles (nodes) with their objectives and the hierarchical relations between them (edges) in a graph-like representation.

To define how must each of those goals be accomplished, landmarks are defined. A set of ordered landmarks which must happen in order to achieve a certain goal define a scene. For each scene, the execution of it entails a certain state of one or more objectives. For instance, when the objective *Aprove_treatment* is fulfilled, the landmark *Check_treatment* is reached. Role *Doctor* is involved on the landmark as *Player* because it has the objective *Aprove_treatment* assigned. *Figure* 3 shows the scenes (nodes) with their landmarks and the transitions between them (edges) in a graph-like representation.

The last element to be defined on the organisation level are the norms. Norms are defined by the *activation*, *maintenance* and *expiration* condition (as introduced in §3.1). *Figure* 4 shows some of the norms modeled for the use case.

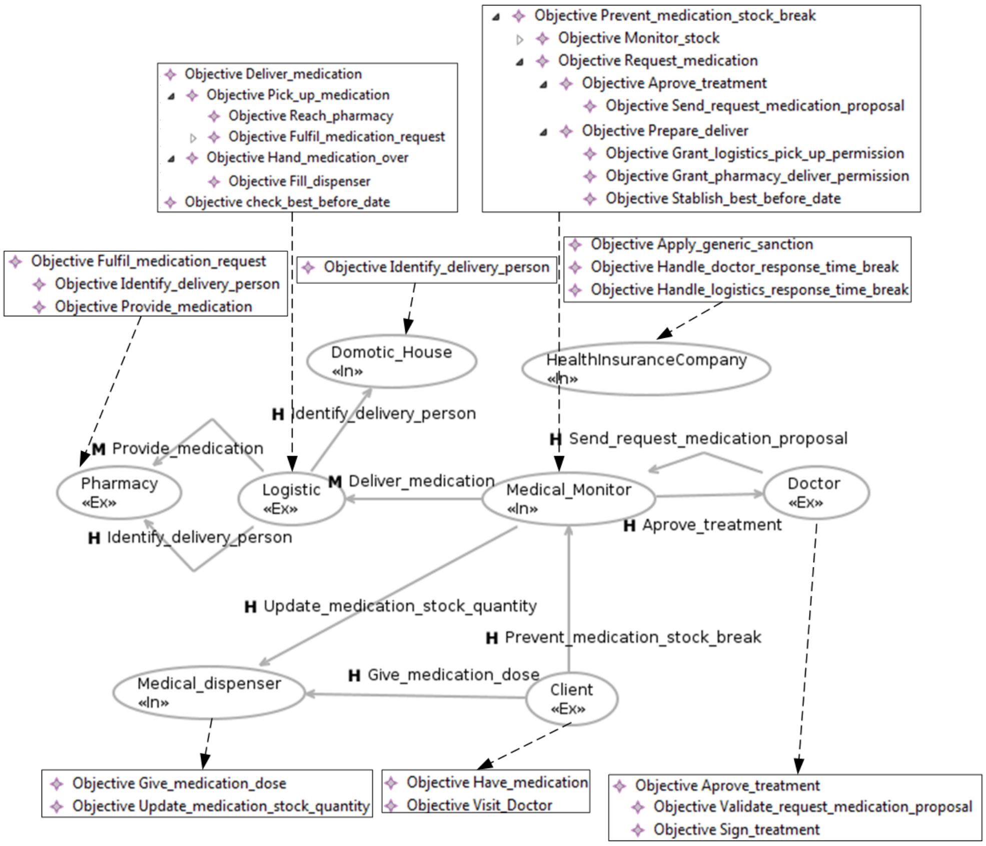

Figure 2. Roles, their objectives and role dependencies

4.2. Coordination level model

The elements on organisation level are derived into the coordination level. This level contain three main elements, actions, plans and agents. Deriving the agents is very straightforward: a single agent is created for each role on the organisation level. Deriving tasks and plans is more complex and explained above. Actions are derived from objectives on the organisation level. Objectives without sub-objectives are derived as *atomic actions*, whereas objectives that have sub-objectives are derived as *composite actions*. Composite actions contain sets of atomic actions inside an structure called *Construct Bag*. Construct bags allow the definition of restrictions among the set of atomic actions that form the composite action (e.g. precedences between the actions or actions that can be enacted in parallel). Atomic actions contain both pre and post-conditions representing the state of the world before and after the task is enacted. Thus, the post-condition of a given task matches the state description of the objective the task is derived from. Atomic actions contain as well Inputs and Outputs, representing the parameters the action is taking and returning respectively. These parameters are mapped to concepts defined on the *Domain Ontology*. For instance, the action *Visit_Doctor* has *timestamp*, *doctor* and *client* as inputs, and *treatment* as output.

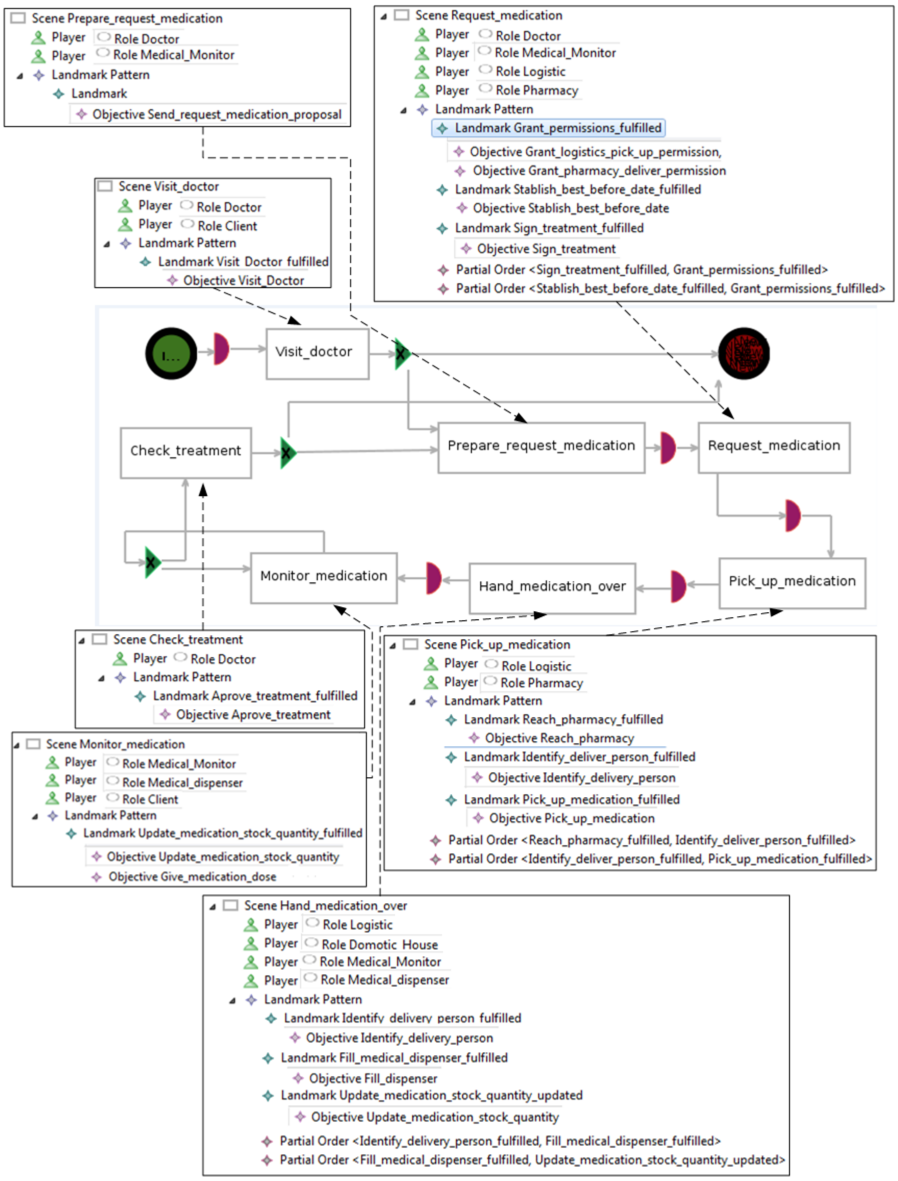

Figure 3. Scene flux, the green circle represents the *Start scene*, the red circle the *End scene*

Plans represent chains of actions. One plan has been modeled for each transition between landmarks or scenes defined on the organisation model. In the case of the transition between the scenes $Prepare\_request\_medication \rightarrow Request\_medication$ and $Pick\_up\_medication \rightarrow Hand\_medication\_over$ one additional plan is provided for each transition. In this case, the agents have two options available in order to bring the world from the state represented by the first scene to the state represented by the second one. The norms defined on the organisation model provide means for pondering both plans and choosing the most appropriate one on each situation.

4.3. Service level model

The service level model consists in a set of OWL-S annotations applied to the web services available in the system. These annotations, modeled in the form of IOPE (i.e. Inputs-Outputs-Preconditions-Effects) allow the matchmaker component to find suitable services for the tasks the agents try to enact. This intermediate component allow a dynamic mapping between the tasks and the services, choosing the most appropriate service on the fly. This means different services (with equivalent IOPE descriptions) can be chosen for the same task depending on service's availability and performance. For instance, the task *identify_delivery_person* can be performed by enacting the services *id_card_reader*, *domotic_door_iris_reader* or *rfid_identifier*.

5. Conclusions and related work

Dynamic service-composition is an issue that has been tackled *via* pre-defined workflow models where nodes are not bound to concrete services, but to abstract tasks at runtime. This work presents a similar approach (through the mapping performed by the match maker component) with the difference that workflows used are not predefined, but dynamically generated from the information provided by an organisational level, and thus, workflows evolve as the organisational information evolves. Due to the connection among levels, a change in the organisational level can trigger changes both in the coordination level (*via* plan and agent generators) and in the service level (new plans will result in the execution of new tasks and, possibly, the invocation of new services). Thanks to this approach, new roles, objectives and norms can be introduced in the organisational level, without designer having to perform any modification to coordination and service levels, as this changes are automatically performed. Intelligent agents at the coordination level present an option for providing both exception handling and organisational-normative awareness capabilities to the system. Exception handling is common in other service-oriented architectures, however, most approaches tend to focus on low-level (*i.e.* service) exception handling. The ALIVE approach enables managing of exceptions at multiple levels either substituting services (service level) looking for alternative workflows to connect two landmarks (coordination level) or even looking to achieve alternative landmarks among the same scene (organisational level). Agents at coordination level

Property	Value
Activation Condition	isActive(Provide_medication)
Deadline	
Expiration Condition	isFullfilled(check_best_before_date)
Maintenance Condition	~isFullfilled(Provide_medication)
Norm ID	o_check_best_before_date_before_Provide_medication

Property	Value
Activation Condition	isViolated(O_check_best_before_date_before_fill_dispenser)
Deadline	
Expiration Condition	isFullfilled(Apply_generic_sanction)
Maintenance Condition	
Norm ID	o_Apply_Generic_Sanction_fill_dispenser

Figure 4. Example of norms of the use case

enable this medium and high-level exception handling, which are not commonly seen in other service-oriented approaches. Regarding organisational-normative awareness, making normative agents reason about the workflows (and the tasks included in them) before performing them, and discarding the ones that do not comply with organisational norms, adds organisational awareness to the execution of the workflows. Normative agents come in handy on the presented case, as they can perform reasoning about what actions to perform taking into account both the actions available and the norms defined.

Acknowledgements

This work has been partially supported by the FP7-215890 ALIVE project. Authors would like to acknowledge partial support from the Spanish funded project ASISTIR TEC2008-06734-C02-02. J. Vázquez-Salceda's work has been also partially funded by the Ramón y Cajal program of the Spanish Ministry of Education and Science. The authors would like to acknowledge the contributions of their colleagues from the ALIVE consortium. The views expressed in this paper are not necessarily those of the ALIVE consortium.

References

[1] R. Annicchiarico, F. Campana, A. Federici, C. Barrué, U. Cortés, A. Villar, and C. Caltagirone. Using scenarios to draft the support of intelligent tools for frail elders in the share-it approach. In *IWANN (1)*, pages 635–641, 2009.

[2] A. Cortés. Design and execution of a benchmark test for the share-it system. Diploma thesis, Technical University of Catalunya, Computer Software Department, june 2009.

[3] U. Cortés, A.B. Martínez, C. Barrué, T. Benedico, F. Campana, and R. Annicchiarico. An agent-based service to elders mobility using the i-walker. In 7^{th} *International Conference on autonomous agents and Multiagent systems (AAMAS08)*, 2008.

[4] Alvarez-Napagao S. ; Koch F. ; Gómez-Sebastiá I. and Vázquez-Salceda J. Making games alive: an organisational approach. In *Proceedings of the workshop 'Agents for Games and Simulation' AAMAS 2010*, pages 112–124, 2010.

[5] J. S. Chun Lam, W. Vasconcelos, F. Guerin, D. Corsar, A. Chorley, T. J. Norman, J. Vázquez-Salceda, S. Panagiotidi, R. Confalonieri, I. Gomez, S. Hidalgo, S. Álvarez-Napagao, J. C. Nieves, M. Palau Roig, L. Ceccaroni, H. Aldewereld, V. Dignum, F. Dignum, L. Penserini, J. A. Padget, M. De Vos, D. Andreou, O. Cliffe, A. Staikopoulos, R. Popescu, S. Clarke, P. Sergeant, C. Reed, T. Quillinan, and K. Nieuwenhuis. ALIVE: A framework for flexible and adaptive service coordination. In *ESAW*, pages 236–239, 2009.

[6] C. Mandel, K. Huebner, and T. Vierhuff. Towards an autonomous wheelchair: Cognitive aspects in service robotics. In *Proceedings of Towards Autonomous Robotic Systems (TAROS 2005)*, pages 165–172, 2005.

[7] Manel Palau, Luigi Ceccaroni, Ignasi Gómez-Sebastià, Javier Vázquez-Salceda, and Juan Carlos Nieves. Coordination and organisational mechanisms applied to the development of a dynamic, context-aware information service. In *ICAART (2)*, pages 88–95, 2010.

[8] Brazier F. Aldewereld H. Dignum F. Dignum V. Penserini L. Wijngaards N.: Quillinan T. B. Developing agent-based organizational models for crisis management. In *8th International conference on Autonomous Agents and Multiagent Systems*, 2009.

[9] Supported Human Autonomy for Recovery and Enhancement of cognitive and motor abilities using information technologies. http://www.ist-shareit.eu, 2007.

[10] Dignum V. A model for organizational interaction: Based on agents, founded in logic. In Utrecht University, editor, *SIKS Dissertation Series 2204-1*. Utrecht University, 2004.