

A Framework for the Development and Maintenance of Adaptive, Dynamic, Context-Aware Information Services

Manel Palau¹, Ignasi Gómez-Sebastià², Luigi Ceccaroni³, Javier Vázquez-Salceda²,
Juan Carlos Nieves²,

¹ TMT Factory, C/ Moià 1, planta 1, E08006, Barcelona, Spain
manel.palau@tmtfactory.com

² Knowledge Engineering & Machine Learning Group, UPC, C/Jordi Girona 1-3, E08034,
Barcelona, Spain

{igomez, jvazquez, jcnieves}@lsi.upc.edu

³ Barcelona Digital Centre Tecnològic, C/Roc Boronat 117, 5^a planta, E08018 Barcelona, Spain
lceccaroni@bdigital.com

Abstract. This paper presents an agent-based methodological approach to design distributed service-oriented systems which can adapt their behaviour according to changes in the environment and in the user needs, even taking the initiative to make suggestions and proactive choices. The highly dynamic, regulated, complex nature of the distributed, interconnected services is tackled through a methodological framework composed of three interconnected levels. The framework relies on coordination and organisational techniques, as well as on semantically annotated Web services to design, deploy and maintain a distributed system, using both a top-down and bottom-up approach. We present results based on a real use case: interactive community displays with tourist information and services, dynamically personalised according to user context and preferences.

Keywords: Multi-agent systems, coordination and organisational theory, context awareness, personalised recommendation, semantic Web services.

1 Introduction

Urban information services are often provided in ways which have not changed much in a century. This scenario brings up the opportunity to improve services provided to people living in or visiting a city, with the novel possibility of ubiquitously accessing personalised, multimedia content [12]. On one hand, there are numerous, dynamic services that have to be composed and coordinated in order to provide higher-value services. For instance, an advanced entertainment service can be provided by combining information coming from cinema, restaurant and museum services, along with transport and mapping services. These services are not static, as existing services can leave the system and new ones can enter it; and the service used in a given moment for a given task might not be available later or it might happen that a more

suitable service becomes available. On the other hand, the incoming content has to be filtered and adapted, to make it compatible with user's context (e.g. location, time and date), preferences and requirements, and with existing regulations. For example, the system should not suggest a pub to an underage user if local laws do not allow it.

An additional challenge for systems in highly dynamic environments, where unexpected events can arise at any time (e.g. transport not in time due to a traffic jam), is to be able to react and adapt to these events.

We consider that this complex scenario can benefit from the combination of multi-agent techniques and semantic Web services [6] to enable dynamic, context-aware service composition [21], thus providing users with relevant high-level services depending on their current context. Moreover, technologies concerning organisational and coordination theories applied to (intelligent) Web services [14] are also important in order to effectively maintain a system operating in such a constrained (due to user's preferences and local laws) and dynamic environment.

Additionally, the scenario presents the need of integrating new functionalities, new services or new actors (humans or artificial intelligence systems) into an existing running system. This integration is especially difficult taking into account that the scenario presents a system formed by active, distributed and interdependent processes.

In the ALIVE European project [1] a new software-engineering methodology is being explored [22]. The approach aims to bring together leading methods from coordination technology, organisation theory and *model driven design* (MDD) to create a framework for software engineering to address a reality composed of live, open systems of active services. Apart from the ones described in this paper, other examples of these services can be found in Quillinan et al. [19]. The ALIVE's framework is a multi-level architecture composed of three levels:

- the organisational level, which provides context for the other levels, supporting an explicit representation of the organisational structure of the system (composed by roles, objectives and the dependencies among them), and effectively allowing a structural adaptation of distributed systems over time;
- the coordination level, which provides the means to specify, at a high level, the patterns of interaction among services, transforming the organisational representation coming from Organisational Level into coordination plans (including information flows, constraints, tasks and agents);
- the service level, which allows the selection of the most appropriate services for a given task based on the semantic description of services and, effectively supporting high-level, dynamic service composition.

In this paper, results of the application of the ALIVE's approach (introduced in section 3) for designing dynamic, adaptive systems are presented. In particular, the design for each level is described (see section 4). First, a complete organisational level design of an *interactive community display* (ICD) scenario (introduced in section 2) is presented. This design shows that the correct identification of roles in the organisational level allows a dynamic adaptation in the coordination level. Then, the complete design of the coordination level of the ICD scenario is outlined. This design shows how, given a set of landmarks (which are states of special interest), a system

can be dynamically adaptable. Finally, the specifications of the Web services involved are presented. In Sections 5 and 6, a discussion of related and future work is outlined.

2 The Use Case

We will use a personalised recommendation tool for entertainment and cultural activities as the basis for exemplifying the scenario. The personalisation is offered via ICDs, which are multimedia information points offering interactive services in public areas [4] [10]. The aim is to bring city services closer to people living in or visiting a city by interconnecting people, service providers and locations. In the scenario, it is taken into account that services and information provided, and how user information is stored, processed and distributed, are all subject of various municipal, national and European regulations.

The scenario considered for this purpose starts when a user interacts with the system's interface (the ICD). The system accesses the user profile (if available) or a group profile from a remote repository. Then, the system adapts the interface format and the interaction mode presenting the initial interface composed by several services, such as cinema or monument. If the user requests one of these services, the system manages the user request considering (if available) user personal data, preferences, requirements and, above all, time, date and location (i.e. user context). Furthermore, environmental context and legislations including components such as weather, traffic reports and legal adult age can be considered.

Finally, the system presents an ordered list of activities located on a map together with basic information, such as a brief description, address and pictures. Moreover, it informs on transportation (e.g., bus and metro) to reach the venue and, if time is appropriate, it suggests a restaurant along the way, thus composing information from different services such as cinemas, restaurants, maps and transport (see Figure 1).

3 The ALIVE Framework

The ALIVE framework is being developed in collaboration by several universities and enterprises within the frame of the European project ALIVE. It combines MDD and agent-based system engineering with coordination and organisational mechanisms, providing support for "live" (that is, highly dynamic) and open systems of services. ALIVE's multi-level approach (see Figure 2 and following sections) helps to design, deploy and maintain distributed systems by combining, reorganising and adapting services. As shown in Section 4, this framework is suitable for scenarios with new services entering the system and existing services leaving it at run-time.

4 Manel Palau¹, Ignasi Gómez-Sebastià², Luigi Ceccaroni³, Javier Vázquez-Salceda², Juan Carlos Nieves²,

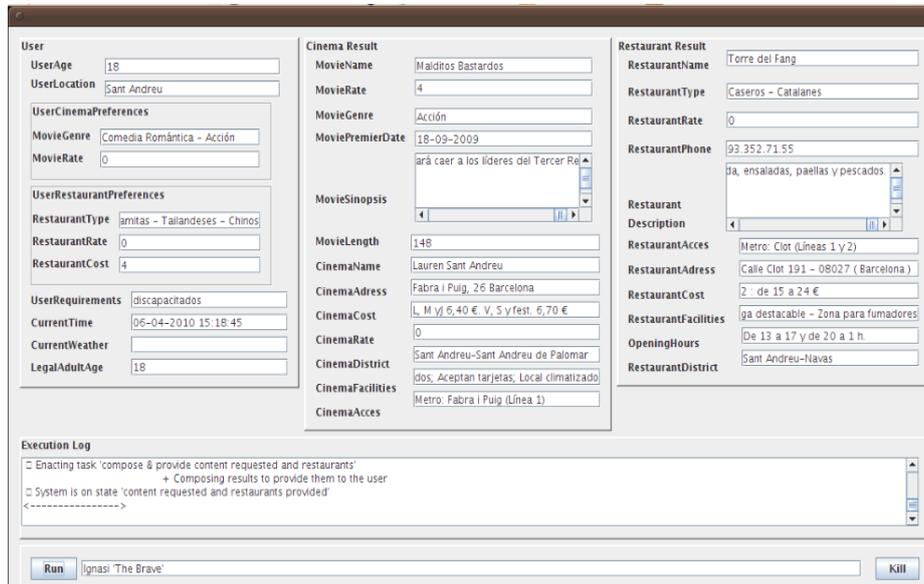


Fig. 1. Application interface showing food and movies suggestions

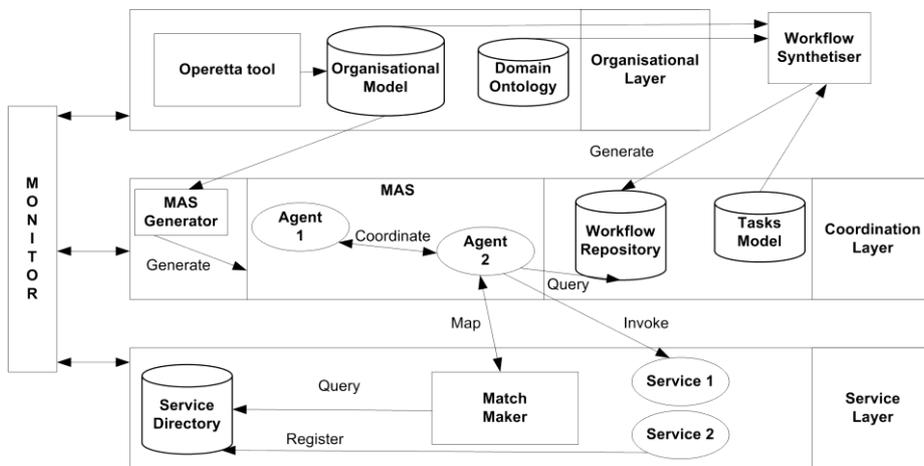


Fig. 2. Main components of the ALIVE architecture

3.1 Organisational Level

The organisational level provides an explicit representation of the organisational structure of the system. The organisational model is the main component of the organisational level, representing the organisation as a social system created by autonomous actors (i.e. they have their own interests) to achieve common goals.

Stakeholders and their relations are represented, together with formal goals and restrictions. The model is formalised according to the Opera methodology [8] including goals (e.g. gather petition type, obtain personalised content or gather and compose content); roles (e.g. user, interaction task manager or content provider) that are groups of activity types played by actors (i.e. the agents or human users); and landmarks (e.g. user data collected, external data collected and content to be provided personalised).

Objectives are assigned to roles, among which three kinds of relations exist: the hierarchical relation, where a parent role can delegate an objective to a child role; the market relation where a child role can request the assignment of an objective to the parent role; and the network relation, where both parent and child roles can request an objective to the other one. Each relation is assigned according to what interaction type the designer expects to happen. For instance, in the presented scenario, there is a market relation when the content adaptor role asks the content provider to obtain food suggestion. The set of all roles and the relations among them constitutes the social structure.

Landmarks are important states in the achievement of a goal, and landmark patterns impose an ordering over landmarks to be reached. A set of landmarks and their relations is known as *scene* (see section 4). Scene transitions can be modelled by organising them in an *interaction structure* (see section 4). The organisational level supports the definition of norms, rights and obligations (suitable for highly regulated scenarios) of the actors, effectively forming a normative structure. The social structure, the interaction structure and the normative structure are the three components of the organisational model.

The OperettA tool [17] supports system designers in specifying and visually analysing an organisational model. The domain ontology represents the shared understanding of the domain, providing a common vocabulary about all concepts and their properties, definitions, relations and constraints, and can be defined using existing ontology-editors [5].

The organisational model is used by the multi-agent system (MAS) generator in the coordination level to create the agents that populate the system. For each role defined in the organisational model one or more agents are generated.

3.2 Coordination Level

The coordination level provides the patterns of interaction among actors, transforming the organisational model into coordination plans, or workflows. Workflows are defined using *generalised partial global planning* (GPGP), a framework for coordinating multiple AI systems that are cooperating in a distributed network [13]. Workflows bring the system from a landmark state to the next one (see Figure 3) and are formed by chains of tasks.

Tasks, defined on the coordination level's task model, contain both pre- and post-conditions that describe the state of the system before and after the task is performed. Tasks are the bridge between the Coordination and Service levels, containing information that binds them to abstract services on the service level (e.g. inputs and outputs) and to elements of the organisational model (e.g. roles assigned to the task).

6 Manel Palau¹, Ignasi Gómez-Sebastià², Luigi Ceccaroni³, Javier Vázquez-Salceda², Juan Carlos Nieves²,

The workflow synthesiser uses information from the organisational model, domain ontology and tasks model in order to generate the workflows the agents will enact. These workflows are stored in the workflow repository where they are retrieved when required.

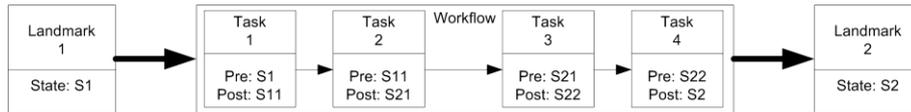


Fig. 3. Workflow example connecting two landmarks

A set of intelligent agents (MAS) deployed on the AgentScape platform [18] enacts the workflows in a coordinated and distributed fashion. Agents analyse and monitor workflow execution, reacting to unexpected events, either by enacting other workflows or by communicating the incident to other levels.

Each agent includes the following components (see Figure 4): the brain module, which provides reasoning and decision-making capabilities; the normative plan analyser, which scans the workflows in order to determine if enacting them will violate any of the norms defined in the organisational model; the Agent Communication Language (ACL) module, which provides agents with the capability of communicating with other agents in the system by sending messages; the GPGP scheduler, which provides an interface for the agents to coordinate and distribute tasks; and the enactment component, which facilitates the invocation of services.

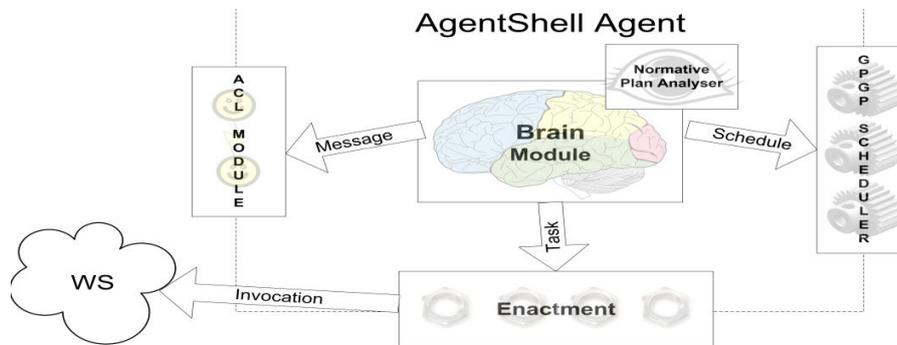


Fig. 4. Agent architecture diagram

3.3 Service Level

Appropriate services are selected for each abstract task in the workflows, using the information included in the service description and in the task description. These descriptions are defined in terms of OWL-S service profiles [16], facilitating the process of composing services [20] and finding alternative services. The reassignment of services to tasks, when a given service is not available, is carried out on the fly.

The match maker component receives an abstract task description from an agent and looks for services that can fulfil this task. It queries the service directory and selects the most appropriate one (if several ones are available), based on the task's semantic description and on quality of service parameters (such as average response time). The service chosen is returned to the agent, and the task is executed and monitored.

At the service level, service composition within the scope of a task is carried out, too. For instance, if a given task requires providing information of a venue on a map, and there are two available services, one to obtain venue information and another to show information on a map, then the task can be bound to the composition of these services.

3.4 The Monitor Tool

The monitor tool is the back-bone of the ALIVE framework [2], 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.

As seen on sections 3.2 and 3.3, 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.

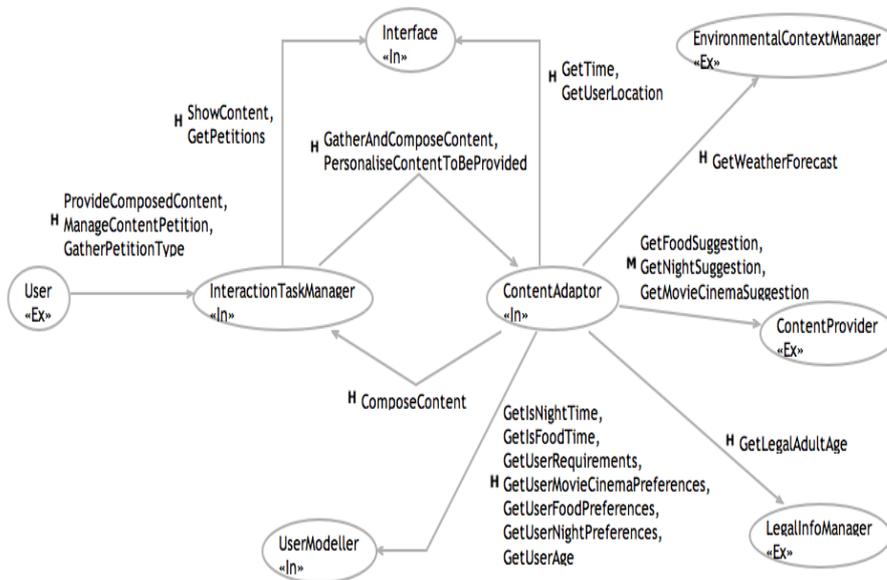


Fig. 5. Operetta model of the social structure

4 Scenario Modelling

This section describes how the ALIVE framework has been applied to the scenario presented in Section 2, and how the system manages user’s petitions by means of the defined roles, objectives, scenes, landmarks, workflows, plans, tasks and services.

An organisational social structure is defined using OperettA (see Figure 5). It includes several (external/internal) roles (e.g. content adaptor, user modeller and content provider) in the domain, and their (hierarchical/market) dependencies in terms of delegated sub-goals (e.g. get user food preferences and get food suggestion).

Roles are represented as nodes and sub-objective dependencies as directional arrows. The objectives or goals considered for the role user (sign up; sign in; obtain personalised content; change profile, preferences or requirements; and change the interface format or the interaction mode) are related with the petitions to be managed by the system. Each user’s goal is subdivided into sub-goals and delegated to other roles. These roles can delegate sub-goals to other roles, too.

From here on, we will focus on the user objective to obtain personalised content. Figure 5 shows how this objective is delegated to the interaction task manager that identifies the type of petition (sub-objective gather petition type) once the interface gets the user petitions (sub-objective get petitions) and then manages the petition. The interaction task manager demands the content adaptor to gather, personalise and compose the content to be provided. To this end, this role relies on information provided by other roles: the interface (providing information about the user context, such as localisation, time and date), the user modeller (providing the user’s personal data, preferences and requirements gathered from the user model) and external information-providers (providing content, and information related to the environmental context and the legislation). The connection with these external information-providers is performed by the match maker component.

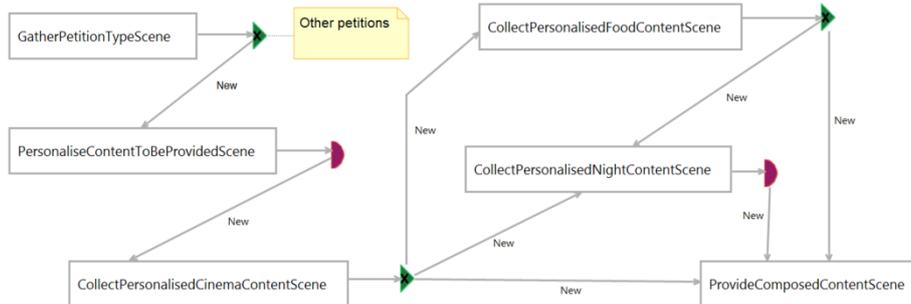


Fig. 6. Interaction structure

The interaction structure for the chosen scenario (see Figure 6) is composed by several scenes representing the petition requested by the user. The transitions among these scenes, starting with the “gather petition type” scene and finalising with the “provide composed content” scene, show several sequences of interactions. Once the petition type is gathered, there are several possible scenes (one per petition type);

however we consider here only a movie cinema petition for simplicity. Hence, next scenes are “personalise content to be provided” and “collect personalised movie cinema content”. There are two possible scenes before providing the composed content, which are “collect personalised food content” and “collect personalised night content”, and are performed depending on user preferences and context.

Each scene is defined by the roles playing within it and a landmark pattern imposing an ordering over the important states (landmarks) that should be reached in the achievement of the goals in the scene. For instance, the personalise content to be provided scene contains four players (interaction task manager, interface, content adaptor and user modeller) and several landmarks (user data collected, external data collected and content to be provided personalised), which follow a partial order among them (see Figure 7).

The plan repository within the coordination level contains the workflows the agents must follow in order to accomplish the landmarks defined in the organisational model. Focusing in the scene personalise content to be provided, there is a distributed plan where several agents coordinate performing tasks in parallel (e.g. the agent enacting the interface role fetches user’s context while the agent enacting the user modeller role looks for user’s personal data, preferences and requirements). The plan personalise content to be provided is adapted depending on several variables, such as night time, food time, user preferences, requirements and context. For instance, the plan can also provide food suggestion taking into account user food time preferences.

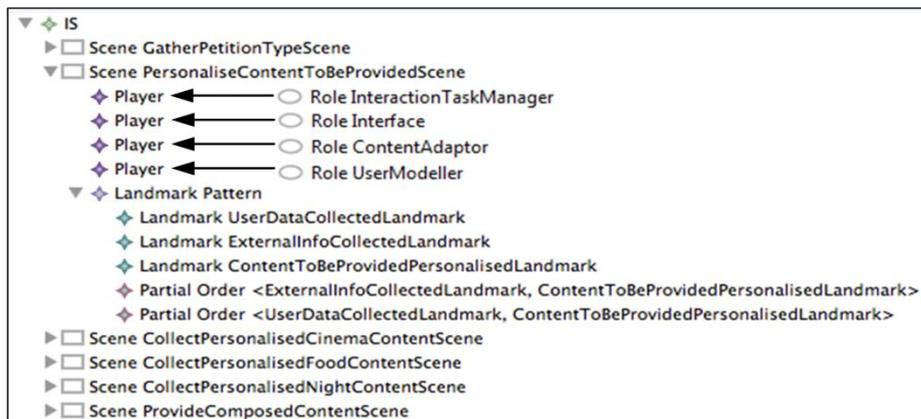


Fig. 7. Landmark pattern for the personalised content to be provided scene

The Coordination level describes a sequence of composite and atomic tasks for each plan. Figure 8 shows the tasks required to provide content to a user considering age and preferences regarding food and night time.

There are several norms applied to the interaction among agents. For instance, the content provider has the obligation of performing his task before a deadline (10 slots time). This norm prevents the user from having to wait too long before its petition is processed.

10 Manel Palau¹, Ignasi Gómez-Sebastià², Luigi Ceccaroni³, Javier Vázquez-Salceda², Juan Carlos Nieves²,

Property	Value
Has Client	
Has Input	Input DateTime, Input AdultAge, Input UserRequirements, Input FoodPreferences
Has Local	
Has Output	Output FoodContent
Has Parameter	
Has Participant	
Id	0
Name	getFoodSuggestion
Performed By	
Performed By Role	ContentProvider
Realizes	

Fig. 8. Composite and atomic tasks to gather content

Tasks are implemented as Web services, which are semantically annotated and described in terms of OWL-S service profiles (i.e. inputs, outputs, preconditions and effects) as shown in Figure 9. In order to perform the task “*get food suggestion*” the match maker component maps *abstract content providers* to concrete ones (e.g. *lanetro* [http://www.lanetro.com] restaurants). Doing it this way allows the system to dynamically readapt to failures of a *concrete content provider* (e.g. remapping to *Atrapalo* [http://www.atrapalo.com] restaurants if *lanetro* restaurants are not available).

```

:profile:Profile rdf:about="#getFoodSuggestionProfile">
  <profile:hasOutput>
    <process:Output rdf:ID="FoodContent">
      <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
        >http://ict-alive.svn.sourceforge.net/viewvc/ict-alive/UseCases/
          THT/trunk/Demo/Model/THTOntology/THT.owl#FoodContent</process:parameterType>
      </process:Output>
    </profile:hasOutput>
    <profile:hasInput>
      <process:Input rdf:ID="DateTime">
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
          >http://ict-alive.svn.sourceforge.net/viewvc/ict-alive/UseCases/
            THT/trunk/Demo/Model/THTOntology/THT.owl#DateTime</process:parameterType>
        </process:Input>
      </profile:hasInput>
      <profile:hasInput>
        <process:Input rdf:ID="AdultAge">
          <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
            >http://ict-alive.svn.sourceforge.net/viewvc/ict-alive/UseCases/
              THT/trunk/Demo/Model/THTOntology/THT.owl#AdultAge</process:parameterType>
          </process:Input>
        </profile:hasInput>
      <profile:hasInput>
        <process:Input rdf:ID="UserRequirements">
          <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
            >http://ict-alive.svn.sourceforge.net/viewvc/ict-alive/UseCases/
              THT/trunk/Demo/Model/THTOntology/THT.owl#UserRequirements</process:parameterType>
          </process:Input>
        </profile:hasInput>
      <profile:hasInput>
        <process:Input rdf:ID="FoodPreferences">
          <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
            >http://ict-alive.svn.sourceforge.net/viewvc/ict-alive/UseCases/
              THT/trunk/Demo/Model/THTOntology/THT.owl#FoodPreferences</process:parameterType>
          </process:Input>
        </profile:hasInput>
      </profile:hasInput>
      <service:presentedBy rdf:resource="#getFoodSuggestion"/>
    </profile:Profile>
  
```

Fig. 9. Ontological concepts related to the task

The modelling presented in this section covers the levels presented in Section 3: organisational level, coordination level, and service level.

5 Conclusions

Users' presence and context can be exploited to provide personalised, dynamic and composed services fulfilling their expectations, needs and functional diversity. As seen on Section 3, this paper presents the design of a multi-agent system that adapts its behaviour according to the environment and the user, and takes the initiative to make suggestions and proactive choices.

orchestration and the re-planning.

Several tourism-related projects, such as E-travel [11] and Deep Map [15], take advantage of agent technology integrated with the semantic Web. The ALIVE project is also close to the Interactive Collaborative Information Systems (ICIS) project [9], in which a MAS takes into consideration unexpected events that happen in the real world in order to obtain a steady and reliable system in dynamic and changing environments, and a higher level view is used to take advantage of the service orchestration and the re-planning.

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).

As outlined in Section 4, 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 SOA 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 enable this medium and high-level exception handling, which are not commonly seen in other SOA approaches. Regarding organisational-normative awareness, to the best of authors' knowledge, no attempts have been made to include normative information in workflows. However, normative agents are common in the literature [3]. 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.

6 Future Work

As future work we plan to extend the system integrating further services, such as booking, payment or planning routes. Considering the unexpected events of the real world, for instance, it might happen that user does not arrive on time to a booked cinema session because of a traffic jam. In this case alternatives must be provided to the user, for instance, booking a ticket in a session that starts a couple of hours later, and suggesting some nearby shops to spend some time while the new session starts. Furthermore, work on the integration of on-time reorganisation mechanisms and Model Driven Design will be performed to be able to promote reliability and stability for services, enabling to keep slowly changing elements separate from dynamic aspects of the environment.

Another interesting aspect that requires additional studies is the improvement of the handling of the user profile and her preferences in the system by means of a logic framework based on Answer Set Programming and possibilistic logic [7]. The framework, called logic programs with possibilistic ordered disjunction (or LPPOD) proposes a flexible formalism which supports: 1) the creation and adaptation of user profile by means of preference rules and necessity values; 2) the reasoning about user preferences to obtain an order among them. In this way it is possible to add dynamically user preferences and to query the user profile against her most preferred interests whenever a service has to be selected. The implementation and integration of the LPPOD framework in the system presented in this paper is currently being investigated. Once the framework is implemented we aim to encapsulate the framework functionality in a dedicated service, which can be queried against user profile and preference at any time.

7 Acknowledgements

This work has been partially supported by the FP7-215890 ALIVE project. Javier 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.

8 References

1. ALIVE European project, <http://www.ist-alive.eu>
2. Aldewereld, H., Álvarez-Napagao, S., Dignum, F., Vázquez-Salceda, J.: Making Norms Concrete. Accepted in: 9th International Conference on Autonomous Agents and Multiagent Systems (2010)
3. Castelfranchi, C., Frank, D., Jonker, C., Treur, J.: Deliberative Normative Agents: Principles and Architecture. In: 6th International Workshop on Agent Theories, Architectures and Languages. LNCS, vol. 1757, pp. 364-378. Springer (1999)

4. Ceccaroni, L., Codina, V., Palau, M., Pous, M.: PaTac - Urban, ubiquitous, personalized services for citizens and tourists. In: 3rd International Conference on Digital Society, pp. 7-12. IEEE Computer Society (2009)
5. Ceccaroni, L., Kendall, E.: A graphical environment for ontology development. In: 2nd International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 958-959. ACM New York, NY, USA (2003)
6. Comas, Q., R-Roda, I., Ceccaroni, L., Sánchez-Marrè, M.: Semiautomatic learning with quantitative and qualitative features. In: VIII Conference of the Spanish Association for Artificial Intelligence, pp. 17-25. Asociación Española para la Inteligencia Artificial, Spain. (1999)
7. Confalonieri, R., Nieves, J.C., Osorio, M., Vázquez-Salceda, J.: Possibilistic Semantics for Logic Programs with Ordered Disjunction. In: Foundations of Information and Knowledge Systems (Prade, H., Link, S., eds.), LNCS vol. 5956, pp. 133-152, Springer, (2010)
8. Dignum, V.: A Model for Organizational Interaction: Based on Agents, Founded in Logic. SIKS Dissertation Series 2004-1. Utrecht University. PhD Thesis (2004)
9. Ghijsen, M., Jansweijer, W., Wielinga, B.B.: Towards a Framework for Agent Coordination and Reorganization, AgentCoRe. In: International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems. LNCS, vol. 4870, pp. 1-14. Springer (2007)
10. Gómez-Sebastià, I., Palau, M., Nieves, J.C., Vázquez-Salceda J., Ceccaroni, L.: Dynamic orchestration of distributed services on interactive community displays: The ALIVE approach. In: 7th International Conference on Practical Applications of Agents and Multi-Agent Systems. ASC, vol. 55, pp. 450-459. Springer (2009)
11. Gordon M. and Paprzycki M.: Designing Agent Based Travel Support System. In: 4th International Symposium on Parallel and Distributed Computing, pp. 207-216. IEEE Computer Society (2005)
12. Greenfield, A.: *Everyware: The Dawning Age of Ubiquitous Computing* (1st ed.). New riders publishing in association with AIGA (2006)
13. Lesser, V., Decker, K., Wagner, T., Carver, N., Garvey, A.: Evolution of the GPGP/TÆMS Domain-Independent Coordination Framework. *Autonomous Agents and Multi-Agent Systems*, vol. 9, num. 1, pp. 87-143. Kluwer Academic Publishers (2004)
14. Luck, M., McBurney, P., Shehory, O., Willmott, S.: *Agent Technology: Computing as Interaction. A Roadmap for Agent Based Computing*. AgentLink (2005)
15. Malaka, R., Zipf, A.: Deep Map challenging IT research in the framework of a tourist information system. In: 7th International Congress on Tourism and Communication, pp.15-27. Springer (2000)
16. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T. R., Sirin, E., Srinivasan, N., Sycara, K.: OWL-S: Semantic Markup for Web Services. Technical Report, Member Submission, W3C (2004)
17. Okouya, D.M., Dignum, V.: OperettA: A prototype tool for the design, analysis and development of multi-agent organizations. In: 7th International Conference on Autonomous Agents and Multiagent Systems, pp. 1677-1678. ACM (2008)
18. Overeinder, B. J., Brazier, F.: Scalable middleware environment for agent-based internet applications. In: Workshop on State-of-the-Art in Scientific Computing. LNCS, vol. 3732, pp. 675-679, Springer (2004)
19. Quillinan, T. B., Brazier, F., Aldewereld, H., Dignum, F., Dignum, V., Penserini, L., Wijngaards, N.: Developing Agent-based Organizational Models for Crisis Management. In: 8th International conference on Autonomous Agents and Multiagent Systems (2009)

14 Manel Palau¹, Ignasi Gómez-Sebastià², Luigi Ceccaroni³, Javier Vázquez-Salceda², Juan Carlos Nieves²,

20. Sirin, E., Parsia, B., Hendler, J.: Filtering and selecting semantic web services with interactive composition techniques. *IEEE Intelligent Systems*, vol. 19, num. 4, pp.42-49. IEEE Computer Society (2004)
21. Vallée, M., Ramparany, F., Vercouter, L.: A Multi-Agent System for Dynamic Service Composition in Ambient Intelligence Environments. In: 3rd International Conference on Pervasive Computing. Doctoral Colloquium (2005)
22. Vázquez-Salceda, J., Ceccaroni, L., Dignum, F., Vasconcelos, W., Padget, J., Clarke, S., Sergeant, P., Nieuwenhuis, K.: Combining Organisational and Coordination Theory with Model Driven Approaches to develop Dynamic, Flexible, Distributed Business Systems. In: 1st International Conference on Digital Businesses. Springer (2009)