COORDINATION AND ORGANISATIONAL MECHANISMS APPLIED TO THE DEVELOPMENT OF A DYNAMIC, CONTEXT-AWARE INFORMATION SERVICE

Manel Palau, Luigi Ceccaroni
TMT Factory, C/ Moià 1, planta 1, E08006, Barcelona, Spain

Ignasi Gómez-Sebastià, Javier Vázquez-Salceda, Juan Carlos Nieves
Knowledge Engineering & Machine Learning Group, UPC, C/Jordi Girona 1-3, E08034, Barcelona, Spain

Keywords: Multi-agent systems, Coordination and organisational theory, Context awareness, Personalised recommendation, Semantic Web services.

Abstract: A multi-agent system design-methodology is used to address the highly dynamic, regulated, complex, distributed environment of interconnected services. A framework of three interconnected levels is applied to tackle this issue relying on coordination and organisational techniques, as well as on Web-services and new methodologies to design, deploy and maintain a distributed system. This paper presents results based on a real use case: interactive community displays with touristic information and services, dynamically personalised according to user context and preferences.

1 INTRODUCTION

Urban information services are often provided in ways which have not changed much in a century. This scenario should inevitably evolve, bringing 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 (Greenfield, 2006).

On the one hand, this scenario presents 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; 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 suddenly available.

On the other hand, the scenario has to be able to filter and adapt the incoming content, making it compatible with user's preferences and location, and with existing regulations. For example, a recommendation 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 context can benefit from the combination of multi-agent techniques, semantic Web services and machine learning (Comas et al., 1999) to enable dynamic, context-aware service composition (Vallée et al., 2005), 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 (Luck et al., 2006) 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 AI systems) into an existing *running* system. This integration is especially difficult taking into account that the scenario

Palau M., Ceccaroni L., Gómez-Sebastià I., Vázquez-Salceda J. and Carlos Nieves J. (2010).
COORDINATION AND ORGANISATIONAL MECHANISMS APPLIED TO THE DEVELOPMENT OF A DYNAMIC, CONTEXT-AWARE INFORMATION SERVICE.
In *Proceedings of the 2nd International Conference on Agents and Artificial Intelligence - Agents*, pages 88-95
DOI: 10.5220/0002731600880095

presents a system formed by active, distributed and interdependent processes.

In the ALIVE European project a new software-engineering methodology is being explored (Vázquez-Salceda et al., 2009). This approach has as objective to bring together leading methods from coordination technology, organisation theory and *model driven design* (MDD) to create a framework for software engineering addressing a new reality composed of live, open systems of active services. Examples of these services are the ones described in this paper or the ones involved in the crisis management scenario described in Quillinan et al. (2009). 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 (composed by patterns and rules) of the system, 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 (including information flows, constraints, tasks and agents) coming from Organisational Level into coordination plans; and
- the *service level*, which allows the semantic description of services and the selection of the most appropriate services for a given task (based on the semantic information contained in the service description), effectively supporting high-level, dynamic service composition.

In this paper, results of the application of the ALIVE's approach (see section 3) to the design of dynamic, adaptive systems are presented. In particular, the design for each level is described (see section 4). First, the complete design of the organisational level of an *interactive community display* (ICD) scenario (see 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 SCENARIO

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 (Ceccaroni et al., 2009; Gómez-Sebastià et al., 2009). 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 starting point of the scenario is a user interacting with the system's interface (the ICD) in search for entertainment or cultural activities around the city. The user identifies herself. Then the system accesses the user profile (if available) from a remote repository, and adapts the interface format and the interaction mode according to user preferences. If the user requests a service, the system composes an initial recommendation considering user preferences, requirements and, above all, time and location. The environmental context includes components such as weather and traffic reports. Ratings and reviews about venues (restaurants, cinemas, shops and museums) are also taken into account, as well as legislation. Finally, activities related to the service requested are presented, located on a map together with basic information, such as a brief description, address and pictures.

When the user requests information about a venue, for instance a cinema, the system shows its detailed description (e.g., movies, sessions and prices). 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 (cinemas, restaurants, maps and buses).

3 THE ALIVE FRAMEWORK

The ALIVE framework is being developed in collaboration by several universities and enterprises within the frame of 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

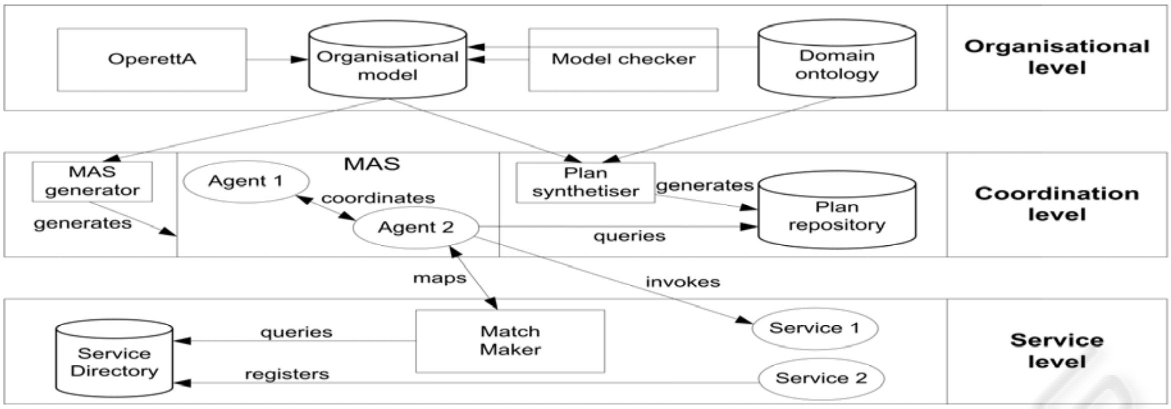


Figure 1: Main components of the ALIVE architecture.

multi-level approach (see Figure 1 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.

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, requirements and restrictions. The model is formalised according to the Opera methodology (Dignum, 2004). The model includes *objectives*, or goals, of the organisation (e.g. receive personalised content, choose a suitable content provider or provide content); the *roles* (e.g. user, service broker or content provider) that are groups of activity types played by actors (i.e. the agents, or human users); and the *landmarks* (e.g. content provider chosen and content provided).

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; 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 service broker looks for the most suitable content provider. The set of all roles and the relations among them is 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 Figure 5). Relations between scenes (i.e. scene transitions) can be modelled by organising them in an *interaction structure* (see Figure 4).

The organisational level supports the definition of *norms*, *rights* and *obligations* of the actors forming a *normative structure*. Norms are suitable for highly regulated scenarios like the one presented (e.g. deadlines on user-system interactions and strict regulations applying to users' personal data).

The *social structure*, the *interaction structure* and the *normative structure* are the three components of the *organisational model*.

The *OperettA* tool (Okouya and Dignum, 2008) supports system designers in specifying and visually analysing an organisational model, whereas, the *model checker* verifies its consistency. 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 (e.g., Noy et al., 2003; Ceccaroni and Kendall, 2003).

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

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 (Lesser et al., 2004). Workflows bring the system from a landmark state to the next one (see Figure 2). Tasks contain both pre- and post-conditions that describe the state of the system before and after the task is performed.

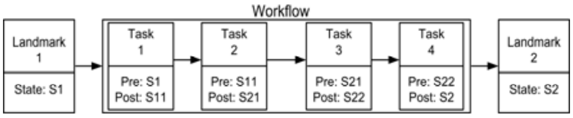


Figure 2: Workflow example, connecting two landmarks.

A set of intelligent agents (MAS) deployed on the AgentScape platform (Overeinder and Brazier, 2004) 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: the *brain module*, implemented in 2APL (Dastani et al., 2007), 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 *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.

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 (Martin et al., 2004), facilitating the process of composing services (Sirin et al., 2004) 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.

4 MODELLING THE SCENARIO

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, norms, landmarks and workflows.

An organisational *social structure* is defined using OperettA (see Figure 3). It includes several (external/internal) roles (e.g. content adaptor and content provider) in the domain, and their dependencies in terms of delegated sub-goals (e.g. provide content). Roles are represented as nodes and sub-objective dependencies as directional arrows. The possible objectives or goals considered for the *user* role (sign up, sign in, receive personalised content, change profile, interests or requirements and change the interface format or the interaction mode) have been 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.

Figure 3 shows how *user's* objectives presented above are delegated to the *interface* that collects the petition (sub-objective collect_petitions) and the *interaction task manager* that manages them (e.g. sub-objective manage_sign_up_petition). Then, in the case of sign up, the petition is managed by subdividing and delegating it to the user modeller (create_new_profile) and the authenticator (assign_open_id).

The sign in petition is delegated to roles *authenticator* (validate_user), *user modeller* (assign_session_preferences). Roles *interface* *(*adapt_interface_format*) and multimodal interaction*

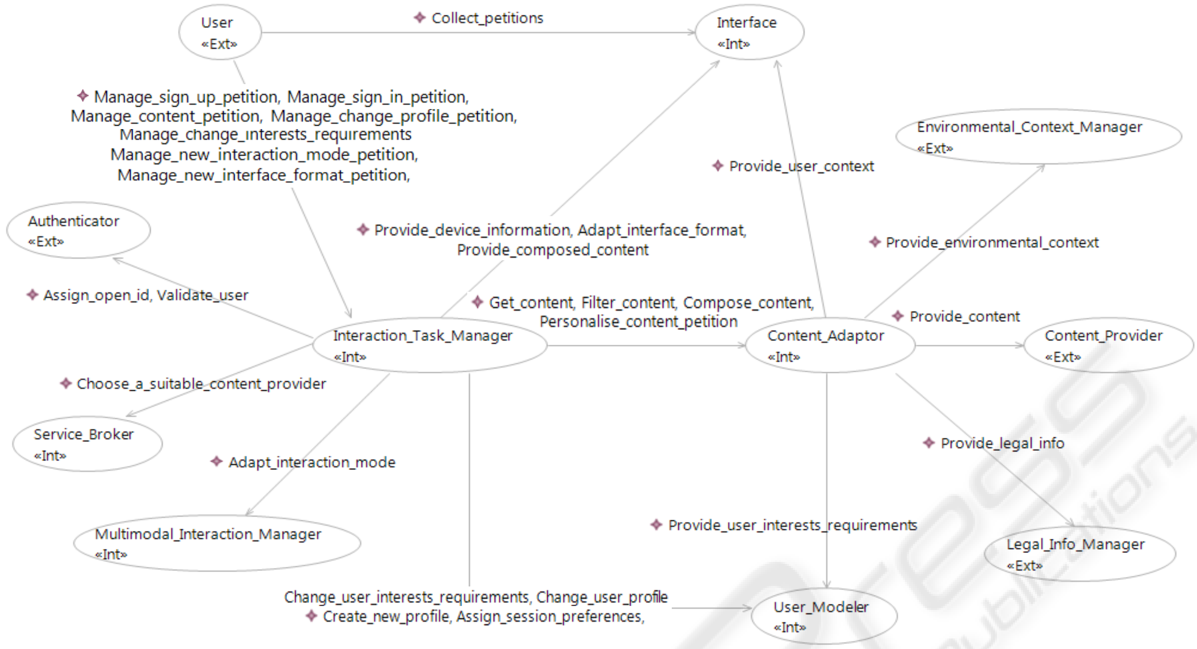

User
«Ext»
Collect_petitions
Interface
«Int»
Manage_sign_up_petition, Manage_sign_in_petition,
Manage_content_petition, Manage_change_profile_petition,
Manage_change_interests_requirements
Manage_new_interaction_mode_petition,
Manage_new_interface_format_petition,
Provide_user_context
Environmental_Context_Manager
«Ext»
Authenticator
«Ext»
Provide_device_information, Adapt_interface_format,
Provide_composed_content
Provide_environmental_context
Assign_open_id, Validate_user
Get_content, Filter_content, Compose_content,
Personalise_content_petition
Provide_content
Interaction_Task_Manager
«Int»
Content_Adaptor
«Int»
Content_Provider
«Ext»
Choose_a_suitable_content_provider
Service_Broker
«Int»
Adapt_interaction_mode
Provide_legal_info
Multimodal_Interaction_Manager
«Int»
Provide_user_interests_requirements
Legal_Info_Manager
«Ext»
Change_user_interests_requirements, Change_user_profile
Create_new_profile, Assign_session_preferences,
User_Modeler
«Int»

Figure 3: Operetta model of the social structure, showing roles (nodes) and their sub-objective dependencies (arrows).

manager (adapt_interaction mode) can also get delegated objectives if the output is to be adapted to user's preferences and device features.

The objectives of the *content adaptor* role are collect, personalise and filter (if necessary) the content according to laws, user preferences/ requirements and context (information such as the weather conditions). To this end, the role relies on information provided by other roles: the *interface* (providing information about the user context, such as localisation), the *user modeller* (providing the user's interests 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 *service broker*.

The *interaction structure* for the chosen scenario (see Figure 4) is composed by four scenes representing the petitions requested by the users. The transitions among these scenes, starting on the left circle and finalising on the right one, show several sequences of interactions. The *user model creation* scene contemplates the sign up petition, while, within the *user model adaptation* scene, the updating of user's profile, interests and requirements is managed.

The *interaction interface adaption* scene contains the following petitions: sign in (just for registered users), new interface format and new interaction mode. From then on, the information about the user can be either uploaded or not, depending on her authentication status (either authenticated or not authenticated). Finally, the *content adaption* scene contemplates the content petition, which happens in a state where the interaction and interface preferences have been loaded, so interface format and interaction mode are already adapted.

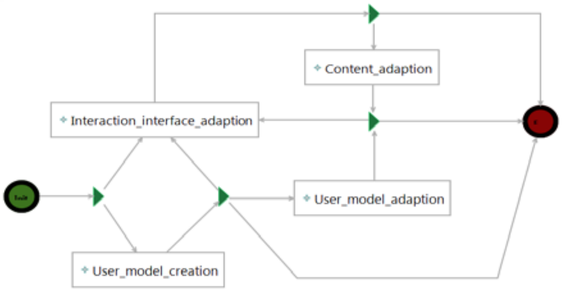


Figure 4: Interaction structure.

Each scene can be defined by 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, within the *content adaption* scene, the system passes from one landmark (state) to the next one as depicted in Figure 5.

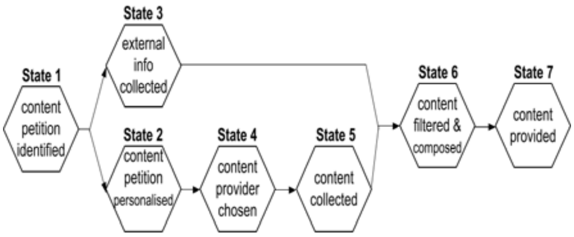
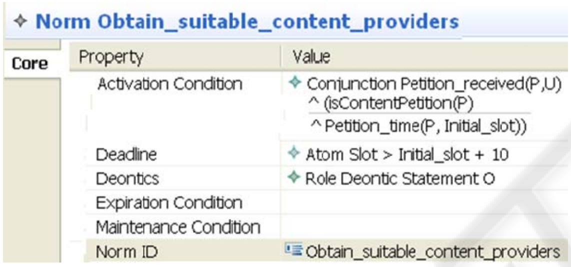


Figure 5: Landmark pattern of the content adaption scene.

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 now in the *content adaption* scene (see Figure 8), there is a distributed plan where several agents coordinate performing tasks in parallel, in order to maximise the performance. For instance, the *interface* provides user's context and the *user modeller* provides user's interests and requirements (if available).



✧ Norm Obtain_suitable_content_providers

Core	Property	Value
	Activation Condition	✧ Conjunction Petition_received(P,U) ^ (isContentPetition(P) ^ Petition_time(P, Initial_slot))
	Deadline	✧ Atom Slot > Initial_slot + 10
	Deontics	✧ Role Deontic Statement ()
	Expiration Condition	
	Maintenance Condition	
	Norm ID	Obtain_suitable_content_providers

Figure 6: Norm example.

There are several norms applied to the interaction among roles. For instance, as seen in Figure 6, the content provider has the obligation of performing his task before a deadline (10 slots time). This norm has the objective of preventing the user from having to wait too long before its petition is processed.

Then, the *content adaptor* personalises the petition and collects content taking into account the gathered user interests and requirements. Meanwhile *legal body* and *environmental context manager* will provide external information such as legal info or weather and traffic reports. Once all information has been gathered, including content information coming from content providers as well as users' and external information, the *content adaptor* role filters and adapts the content according to laws, user's model and current state (weather forecast, traffic, actual time…). The content is composed on a list and on a map to finally provide it to the user through the *interface* role.

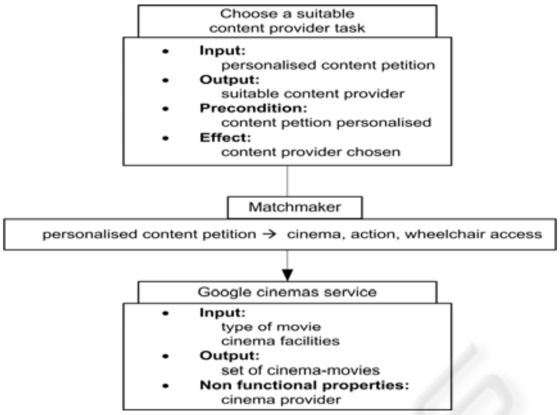

Choose a suitable content provider task
Input:
personalised content petition
Output:
suitable content provider
Precondition:
content pettion personalised
Effect:
content provider chosen
Matchmaker
personalised content petition → cinema, action, wheelchair access
Google cinemas service
Input:
type of movie
cinema facilities
Output:
set of cinema-movies
Non functional properties:
cinema provider

Figure 7: Mapping task to service.

Some tasks are implemented as web services which are semantically annotated; describing them in terms of OWL-S service profiles (i.e. inputs, outputs, preconditions and effects). In order to perform the task "choose a suitable content provider" the matchmaking component maps "abstract content providers" to concrete ones (e.g. "Google cinemas"). Doing it this way allows system to readapt dynamically to failures of a "concrete content provider" (e.g. remapping to "Yahoo cinemas" if "Google cinemas" is not available). Figure 7 shows an example of this mapping.

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

5 RELATED WORK

Several tourism-related projects, such as E-travel (Gordon and Paprzycki, 2005) and Deep Map (Malaka and Zipf, 2000), take advantage of agent technology integrated with the semantic Web. The ALIVE project is also close to the Interactive Collaborative Information Systems (ICIS) project (Ghijsen et al., 2007), 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.

6 FUTURE WORK

Future work will be on integrating further services, such as booking, payment or planning routes.

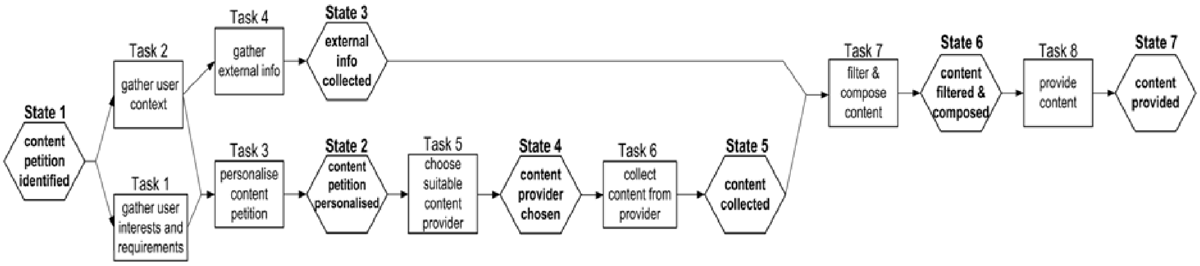


Figure 8: Workflows to achieve the state "content provided".

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 kill 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.

7 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.

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 matchmaker 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 (Castelfranchi et al., 1999). 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.

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 (http://www.ist-alive.eu). The views expressed in this paper are not necessarily those of the ALIVE consortium.

REFERENCES

Castelfranchi, C., Frank, D., Jonker, C. and Treur, J. (1999). *Deliberative Normative Agents: Principles and Architecture.* Proceedings of ATAL'99, 6th International Workshop on Agent Theories, Architectures and Languages. LNCS, vol. 1757, pp. 364-378. Springer.

Ceccaroni, L., Codina, V., Palau, M. and Pous, M. (2009). *PaTac: Urban, ubiquitous, personalized services for citizens and tourists.* Proceedings of ICDS'09 conference, 3rd International Conference on Digital Society, pp. 7-12. IEEE Computer Society.

Ceccaroni, L. and Kendall, E. (2003). *A graphical environment for ontology development*. Proceedings of the second international joint conference on autonomous agents and multiagent systems (AAMAS 2003), pp. 958-959. ACM New York, NY, USA.

Comas, Q., R-Roda, I., Ceccaroni, L. and Sànchez-Marrè, M. (1999). Semiautomatic learning with quantitative and qualitative features. Proceedings of the VIII Conference of the Spanish Association for Artificial Intelligence (CAEPIA-99), pp. 17-25. Asociación Española para la Inteligencia Artificial, Spain.

Dastani, M., Hobo, D. and Meyer, J.C. (2007). *Practical extensions in agent programming languages.* Proceedings of AAMAS'07, 6th International joint conference on Autonomous Agents and Multiagent Systems. Article No. 138. ACM.

Dignum, V. (2004). *A Model for Organizational Interaction: Based on Agents, Founded in Logic*. SIKS Dissertation Series 2004-1. Utrecht University. PhD Thesis.

Ghijsen, M., Jansweijer, W. and Wielinga, B.B. (2007). *Towards a Framework for Agent Coordination and Reorganization, AgentCoRe.* Proceedings the COIN'07, International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems. LNCS, vol. 4870, pp. 1–14. Springer.

Gómez-Sebastià, I., Palau, M., Nieves, J.C., Vázquez-Salceda J. and Ceccaroni, L. (2009). *Dynamic orchestration of distributed services on interactive community displays: The ALIVE approach.* Proceedings of PAAMS'09, 7th International Conference on Practical Applications of Agents and Multi-Agent Systems. ASC, vol. 55, pp. 450-459. Springer.

Gordon M. and Paprzycki M. (2005). *Designing Agent Based Travel Support System.* Proceedings of *ISPDC'05,* 4th International Symposium on Parallel and Distributed Computing, pp. 207-216. IEEE Computer Society.

Greenfield, A. (2006). *Everyware: The Dawning Age of Ubiquitous Computing (1st ed.).* New riders publishing in association with AIGA.

Lesser, V., Decker, K., Wagner, T., Carver, N. and Garvey, A. (2004). *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.

Luck, M., McBurney, P., Shehory, O. and Willmott, S. (2005). *Agent Technology: Computing as Interaction. A Roadmap for Agent Based Computing.* AgentLink.

Malaka, R. and Zipf, A. (2000). *Deep Map challenging IT research in the framework of a tourist information system.* Proceedings of ENTER 2000, 7th International Congress on Tourism and Communication, pp.15-27. Springer.

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. and Sycara, K. (2004). *OWL-S: Semantic Markup for Web Services.* Technical Report, Member Submission, W3C.

Noy, N.F., Crubezy, M., Fergerson R.W., Knublauch, H., Tu, S.W., Vendetti, J. and Musen, M.A. (2003). *Protege-2000: an open-source ontology-development and knowledge-acquisition environment.* AMIA Annu Symp Proc. 2003; 2003: 953.

Okouya, D.M. and Dignum, V. (2008). *OperettA: A prototype tool for the design, analysis and development of multi-agent organizations.* Proceedings of AAMAS'08, 7th International Conference on Autonomous Agents and Multiagent Systems, pp. 1677-1678. ACM.

Overeinder, B. J. and Brazier, F. (2004). *Scalable middleware environment for agent-based internet applications.* Proceedings of PARA'04, Workshop on State-of-the-Art in Scientific Computing. LNCS, vol. 3732, pp. 675–679, Springer.

Quillinan, T. B., Brazier, F., Aldewereld, H., Dignum, F., Dignum, V., Penserini, L. and Wijngaards, N. (2009). *Developing Agent-based Organizational Models for Crisis Management*. Proceedings of AAMAS-IT'09, 8th International conference on Autonomous Agents and Multiaget Systems.

Sirin, E., Parsia, B. and Hendler, J. (2004). *Filtering and selecting semantic web services with interactive composition techniques.* IEEE Intelligent Systems, vol. 19, num. 4, pp.42-49. IEEE Computer Society.

Vallée, M., Ramparany, F. and Vercouter, L. (2005). *A Multi-Agent System for Dynamic Service Composition in Ambient Intelligence Environments.* Proceedings of PERVASIVE'05, 3rd International Conference on Pervasive Computing. Doctoral Colloquium.

Vázquez-Salceda, J., Ceccaroni, L., Dignum, F., Vasconcelos, W., Padget, J., Clarke, S., Sergeant, P. and Nieuwenhuis, K. (2009). *Combining Organisational and Coordination Theory with Model Driven Approaches to develop Dynamic, Flexible, Distributed Business Systems*. Proceedings of Digibiz'09, 1st International Conference on Digital Businesses. Springer.