Deployment of adaptive workflows in intelligent environments

Christos Goumopoulos, Ioannis Calemis
Research Academic Computer Technology Institute
N. Kazantzaki, 26500 Rio Patras, Greece
{goumop, calemis}@cti.gr

Achilles Kameas
Hellenic Open University & RACTI
16, Sahtouri Str, Patras, Greece
kameas@eap.gr

***Abstract*—Workflows have been used to model repeatable tasks or operations in a number of different industries including manufacturing and software. In this paper we examine the use of workflows to model the interaction of services that can be found in intelligent environments to support user tasks and goals. The deployment of such workflows needs to take care special design considerations, including context awareness, adaptation management, device heterogeneity, and user empowerment. In this paper, we present a framework for the deployment of adaptive workflows. The deployment infrastructure supports BPEL-like, design-time compositions that are complemented by mechanisms for the selection and binding of services at runtime. Workflow behaviour can also adjust dynamically in response to detected changes and unforeseen events by a suit of agents whose initial relationships are specified in the workflows.**

Keywords- SOA; workflows; BPEL; dynamic compositions; ubiquitous computing; adaptation

I. INTRODUCTION

Intelligent environments (IE), like smart homes, offices and public spaces, are featured with a large number of devices and services that help users in performing efficiently various kinds of tasks. Combining existing services in pervasive computing environments to create new composite services is in line with the Service-Oriented Architecture (SOA) paradigm [1] and involves special design considerations, including context awareness, adaptation management, device heterogeneity, and user empowerment [2].

In many respects, a composite service can be modeled as a workflow [3]. The definition of a composite service includes a set of atomic services together with the control and data flow among the services. Similarly, a workflow is the automation of a business process, in whole or part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules [4]. Workflows have been used to model repeatable tasks or operations in a number of different industries including manufacturing and software. In recent years, workflows have increasingly used distributed resources and Web services through resource models such as grid and cloud computing. In this paper, we argue that workflows can be used to model how various services should interact with one another as well as with the user in IEs depending on available resources, environment characteristics, user tasks and profile.

Web services are a key implementation technology of the SOA paradigm which is characterized by dynamism and flexibility. However, the main standards proposed to implement the SOA paradigm (i.e., WSDL, UDDI, SOAP) emphasize interoperability rather than the capability to accommodate seamless changes at runtime. Frameworks based on ontologies, such as METEOR-S [5], also lack flexible mechanisms for the distribution of information about services as they require the adoption of shared ontologies that impose the distribution policy. Regarding composition, BPEL (Business Process Execution Language [6]) is the de-facto standard. It takes a workflow-oriented approach to the coordination of cooperating services and provides a good solution for the design–time composition of heterogeneous components wrapped as WSDL services. However, runtime identification of partner services is not addressed and thus the degree of dynamism and flexibility is limited.

In the context of the EU funded R&D project ATRACO we are developing a conceptual framework and a system architecture that supports the realization of adaptive and trusted ambient intelligent systems [7]. Our approach is based on a number of well established engineering principles, such as the distribution of control and the separation of service interfaces from the service implementation, adopting a SOA model combined with intelligent agents and ontologies. Agents support adaptive task realization and enhanced human machine interaction based on a dynamically composed ontology of the properties, services and state of the IE resources.

In this paper, we focus on the service composition mechanism used in ATRACO and how the plan of activities that serve a user goal is specified as a workflow of resource and system services using a streamlined version of BPEL. Our approach is that since a workflow describes the relationship between services and if an agent is represented by such a service, then the relationship between the agents would be possible to specify. Following such an agent-based SOA approach, means that a workflow could be used to establish the initial relationships of the ATRACO multiagent system. Once the basic system has been deployed, the agents could be working proactively so they can adapt to unforeseen circumstances and automatically handle the extension of the workflow description. This leads finally to a decentralized workflow execution model that on the one hand does not suffer from the inflexibility of static workflows, and on the other hand can avoid the computationally expensive cost of frequent replanning of composite services, because of the agent-based proactive behaviour.

The paper is organized as follows. Section 2 gives a basic background on ATRACO concepts, architecture and the role of the main components. Section 3 contains the main contribution of this paper, a service composition framework for deploying adaptive workflows in IEs. We first give an example scenario, and then we discuss the mechanisms developed, the BPEL extensions made in order to express ATRACO specific features in the workflows, and deployment issues. Related work is presented in Section 4 and our conclusion and future work in Section 5.

II. BACKGROUND

In ATRACO, we propose a combination of the SOA model with agents and ontologies (Fig. 1). We adopt SOA both at the resource level to integrate resources, such as devices, sensors and context in applications and at the system level to combine ATRACO services that provide adaptation and trust features into applications. We have defined the concept of an *Activity Sphere* (*AS*), to be both the model and the realization of the set of information, knowledge, services and other resources required to achieve an individual goal within an IE. ATRACO approach adopts a unique standpoint in modeling and realizing ASs. We assume that various IEs are already available; each of them hosting a dynamically changing set of heterogeneous and closed smart objects and components. They, nevertheless, contain heterogeneous descriptions of their capabilities and services that can only be accessed from but not modified by other components. Thus, these objects can collaborate in the realization of ubiquitous computing applications within the hosting IE, but the structure of these applications may dynamically change and their efficient operation depends on the orchestration of heterogeneous services. In order to achieve task-based collaboration amongst them, one has to deal with this heterogeneity, while at the same time achieving independence between a task description and its respective realization within a specific IE.

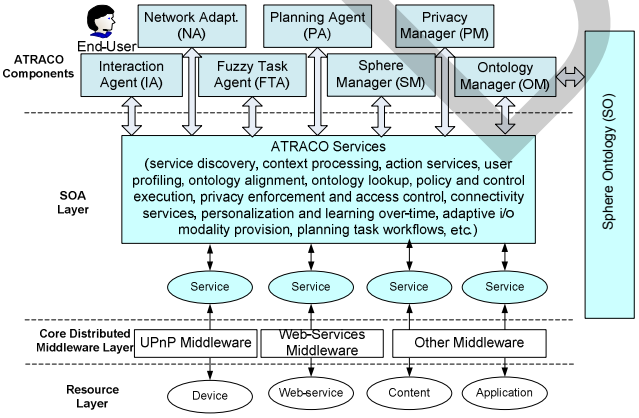

Figure 1. ATRACO architecture

The ATRACO architecture consists of ontologies, active entities, passive entities, and the user who as the occupant of the IE is at the centre of each AS. Active entities are agents and managers. The role of the ATRACO agents is to provide task planning (Planning Agent or PA), adaptive task realization (Fuzzy systems based Task Agent or FTA) and adaptive human-machine interaction (Interaction Agent or IA). The PA encapsulates a search engine that exploits hierarchical planning and partial-order causal-link planning to select atomic services that form a composite service (workflow) [8]. One or more FTAs oversee the realization of given tasks within a given IE. These agents are able to learn the user behavior and model it by monitoring the user actions. The agents then create fuzzy based linguistic models which could be evolved and adapted online in a life learning mode [9]. The IA provides a multimodal front end to the user. Depending on a local ontology it optimizes task-related dialogue for the specific situation and user [10]. The IA may be triggered both by the FTA and the PA to retrieve further context information needed to realize and plan tasks by interacting with the user. On the other hand, ontologies complement agents regarding adaptation by tackling the semantic heterogeneity that arises in IEs by using ontology alignment mechanisms to generate the so-called, Sphere Ontology (SO). There are two main kinds of ontologies: local ontologies, which are provided by both active and passive entities and encode their state, properties, capabilities, and services and the SO, which serves as the core of an AS by representing the combined knowledge of all entities [11].

The Sphere Manager (SM) and Ontology Manager (OM) components are responsible for the formation, adaptation and evolution of the user applications (modeled in ATRACO as ASs) and will be further examined in this paper. In the current version of the system there is also a Privacy Manager (PM) that provides a set of privacy enhancing techniques in order to support privacy in an adaptive and individualized way. Finaly, devices in the IE that may come from heterogeneous networks (e.g., LonWorks, ZigBee, Z-Wave, etc.) and services (e.g., Network Time, VoIP, Real Time Streaming, etc.) are accessed transparently through a service representation layer exporting them to the ATRACO clients as UPnP services. This layer is implemented in the Network Adaptation (NA) component [12].

III. SERVICE COMPOSITION FRAMEWORK

A. Example Scenario

In order to test our service composition framework and to illustrate how workflows can be used to fit user interaction with an IE, we give a simple scenario. This example corresponds to an AS that supports the realization of goal named "Feel comfortable upon arrival at home".

Martha arrives at the door of her smart apartment. The system recognizes her, through an RFID card, and opens the door. On entering the space the system greets Martha by saying "Welcome home" and then when she has entered the living space the lights and A/C are switched on and brightness and temperature are automatically adjusted according to her profile, season, and time of day, to make her feel comfortable. Martha then sits at the sofa to relax and after a while, the system asks "Would you also like some music?" Martha responds positively and the music plays (according to predetermined preferences). Following this, the system asks "Would you like to view yesterday's party photos?" Martha responds positively and a rolling slide show appears in a picture frame in front of her. After a while, Martha gets up, walks towards the window and opens it. Fresh air pours into

the room. Temperature level drops. Brightness level increases. Some of the lights are automatically switched off, in an attempt to maintain the previous level of brightness in the room. After a while, the A/C is switched off because of the open window. Suddenly, the picture frame goes off! The system finds a proper replacement and as a result, photos are displayed in the TV set, while Martha is informed on the event.

Since workflows are essentially graphs of activities, it is useful to express those using UML activity diagrams. Fig. 5 describes the sequence of activities for the example scenario. Note that the tasks "AdjustLights", "AdjustAC" "ShowPhotos" and "PlayMusic" can run in parallel and therefore they have been enclosed in a fork-join block. Note also that the exception events are not part of the workflow description but they are handled by the corresponding ATRACO active entities.

B. *Service Composition Mechanism*

We have developed a service composition mechanism which includes 3 phases: *task workflow planning*, *dynamic service binding* and *execution management and control* as illustrated in Fig. 2.

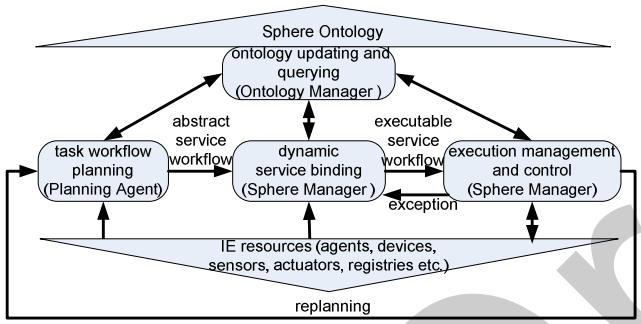

Figure 2. Service composition process in ATRACO

The planning problem can be stated as "discover an execution path of services (tasks) given some state of the world to achieve a goal". In ATRACO, we use a library of abstract plans which model specific user goals. An abstract plan contains a sequence of abstract services which are actually ontological descriptions of service operations that cannot be directly invoked, but will be resolved by the SM during runtime. Having an abstract service workflow description, which is given in a BPEL-like language, the Dynamic Service Binding module of the SM applies a semantic-based discovery mechanism and uses information about available services and context, acquired by the SO through the OM, to discover suitable services or devices in registries able to perform each abstract service. The output of this process is an executable service workflow. In the execution management and control phase the SM executes and continuously monitors the deployed services and the termination condition of the workflow. Fig. 3 gives a conceptual view of the dynamic service binding process. A workflow is mapped into a number of tasks and a workflow task is mapped into one or more abstract services. In addition, each service would also require certain physical resources for its implementation. Mapping of the task to the services can be specified at design time by the PA as per users' functional requirements. However, mapping of the service to the actual human and physical resources is done at runtime, in keeping with service orientation. This dynamic binding is therefore dependent on the context in which the binding occurs.

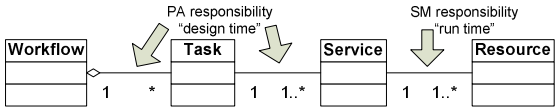

Figure 3. Conceptual model for dynamic service binding

After service binding the SM starts any interaction task in conjunction with the IA and also any FTA task and executes the workflow preserving the precedence constraints or the conditions that are specified in the workflow. At runtime a *Workflow* object aggregates a number of *Task* objects where each object represents a task in the workflow. The services that this task requires for running are divided into input and output services and are connected with the appropriate resources. The resources that are bound to the Task object can be either devices that the Task directly controls (i.e., input sensors and actuation devices) or agents, such as the IA or the FTA. In either case the Task object is informed on the status of the resource and operates according to the pattern specified by its type. The sequence diagram in Fig. 4 shows the basic interaction of the software components during the instantiation of the "Feel Comfortable" AS, which employs the dynamic service binding process mentioned earlier. In the diagram, this process is implemented by the methods used inside the two loops.

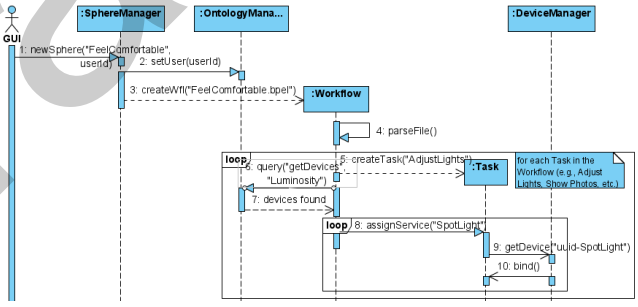

Figure 4. Example AS instantiation and binding of devices to services

In addition, the SM handles exception events that affect the configuration of the AS. For example, exceptions during the execution of the workflow, such as disconnection or failure of devices trigger an adaptation of the workflow by rebinding services to alternative devices. Context changes during the execution of the workflow may invalidate preconditions that were valid during the workflow instantiation. For example, if the user changes location and a follow-me property has been defined for a display service, then the execution state needs to be updated and a new display service instance to be scheduled. In order to achieve workflow adaptation, replanning capabilities may be required by the PA. Replanning comes into play when the dynamic binding fails during workflow execution or update. When replanning is requested a new planning problem is defined with the services that are actually available, and the PA solves the problem and delivers a new workflow.

During AS instantiation in IEs there could be multiple devices or services providing similar functionality from which the system will have to choose. Thus, the ATRACO system must provide mechanisms for selection between similar devices or services and decide which of them is the most suitable to participate in the AS. Device selection is based on criteria such as: task suitability, efficiency (as device's proximity to the user, quality of the service or device and stability), user distraction (the inconvenience a user experiences when the system selects different groups of devices than those that the user prefers or is used to use for a specific task) and confliction to other tasks. For calculating the rank for each device we use a scoring mechanism that is similar to that proposed in [13] and is based on multi-attribute utility theory (MAUT). The overall rank of a device given a specific task is defined as a weighted sum of its evaluation with respect to its relevant orthogonal value dimensions (attributes). For ATRACO the relevant value dimensions are scores for task suitability, efficiency, negative of user distraction and negative of confliction to other tasks.

C. *ATRACO-BPEL Workflow Specification*

BPEL defines a model and a grammar for describing the behavior of a business process based on interactions between the process and its partners. It allows for creating complex processes by creating and wiring together different activities that can, for example, perform Web services invocations (<invoke>), waiting to be invoked by someone externally (<receive>), generate a response (<reply>), manipulate data (<assign>, throw faults (<throw>), or terminate a process (<exit>). In our case, the business process represents the process model of an AS and the partners can take the form, either of a service of a simple device, or the service of an ATRACO agent. While BPEL is a suitable language for describing workflows, an ATRACO workflow description presents requirements that cannot be completely covered by BPEL. This is due to the following :

i. BPEL partners (partnerLinks) are bound statically to specific Web services. In the context of ATRACO, however, services are not bound at design time but dynamically during the execution of the workflow. Thus, there is a need to describe services in the workflow by their semantics which mainly define ontological related searching terms (for example, “Luminosity” for a light service).

ii. The limitation of the one-to-one mapping of services between communicating partners, supported by BPEL. On the other side, ATRACO tasks may need to handle two or more services that provide input or output to the task.

iii. BPEL supports a single coordinator that executes the orchestration logic. ATRACO workflows normally are centrally handled by the Sphere Manager which implements the workflow execution engine; however a more distributed scheme can also be followed by sharing parts of the workflow with collaborating agents (e.g., IA and FTA). This collaboration sets some special requirements in the description of the workflow.

Given the above requirements a variant of BPEL, called ATRACO-BPEL, was defined in order to provide those ATRACO specific features needed in order specify workflows. In the following we explain how using the ATRACO-BPEL formalism an example task is bound with the appropriate service(s). The task *AdjustLights* is associated with the parnterLink *AdjustLightsPL* as part of the orchestration logic section:

```
<bpel:invoke
   name="AdjustLights" partnerLink="AdjustLightsPL">
</bpel:invoke>
```

The partnerLink *AdjustLightsPL* has an input role called *ATRACO:lightStatus* and an output role (partnerRole) called *ATRACO:triggerLight*. The *Continues* type denotes that the execution of the activity is to be treated as a task that is running continuously, i.e., the workflow does not wait its termination.

```
<bpel:partnerLink
   name="AdjustLightsPL"
   partnerLinkType="ATRACO:Continuous"
   myRole="ATRACO:lightStatus"
   partnerRole="ATRACO:triggerLight">
</bpel:partnerLink>
```

The input role *ATRACO:lightStatus* denotes the appropriate abstract service that must be bound to fulfill the role (Luminosity) along with any other application specific details that are needed for its operation e.g., the task will be monitored by an ATRACO agent for learning user behavior with respect to light adjustments and all found light devices are to be used.

```
<ATRACO:role
   name="lightStatus" type="input" Agent="yes" IAmode
="none">
   <ATRACO:service semantics= "Luminosity" trigger =
"Low"  reset = "none" quantity = "all" rules="">
   </ATRACO:service>
</ATRACO:role>
```

The corresponding definition for the output role will be:

```
<ATRACO:role
   name="triggerLight" type="output" Agent="yes"
IAmode ="withAgent">
   <ATRACO:service semantics= "Actuate Light" trigger
= "On"  reset = "Off" quantity = "all" rules="">
   </ATRACO:service>
</ATRACO:role>
```

In ATRACO-BPEL each partnerLink role is specialized as an ATRACO:role which is a new definition in ATRACO-BPEL. In each ATRACO:role the attributes listed in Table 1 are defined.

TABLE I. ATRACO:ROLE SEMANTICS IN ATRACO-BPEL

Attribute	Semantics
name	The name of the role.
type	Denotes the type of the role. Accepted values are **input**/**output**.
Agent	This attribute defines whether the task is monitored by an ATRACO agent or not. Accepted values are **yes/no**
IAmode	Specifies the interaction mode with the ATRACO Interaction Agent. Accepted values are: **none** – no interaction is needed; **pure** – this value is used to indicate that a single interaction with the user through a dialog interface (spoken, tangible or software) needs to be provided either to provide a message or to receive an input for the system from the user in a form of

	question; **direct** – this value is used when the IA needs to create an interface for an output device; **withAgent** – this value is used to indicate that there is a need to find proper user inputs for the Agent monitored tasks.

Each ATRACO:role envelopes a set of services that are bound to it. Each role can have more than one abstract service. If the role type is input then the activity waits for all the services to deliver their result before proceeding. If the role type is output then, upon activity completion, all the services enveloped in this role are triggered. For each abstract service specific attributes are defined, providing the necessary support for device discovery and service operation. Table 2 summarizes the service-specific attributes in ATRACO-BPEL.

TABLE II. SERVICE-SPECIFIC ATTRIBUTES IN ATRACO-BPEL

Attribute	Semantics
semantics	The semantics of the service as a set of keywords – these are used to find the specific device that can be bound to this abstract service
trigger	**input role:** denotes a linguistic value that triggers the service **output role:** denotes a linguistic value passed to the service
reset	The reset state (linguistic value) that the service should apply in the case that the activity cannot be performed
quantity	A number that defines how many devices providing this service are needed for the specific activity. If the value is "**all**" then all found devices are used.
rules	Any special constraints need to be met for binding the corresponding device(s).
IAdlg	This attribute is associated with the direct or pure interaction modes with IA in order to give it the proper interaction dialog type. Examples of accepted values are: GreetingMessage, LightInstructions, GrantGuestAccess, MusicQuestion, MusicControl, PhotoFrameQuestion, SlideshowControl.

D. Deployment

The starting point for running an AS is the generation of the corresponding workflow. Workflows are described in ATRACO-BPEL, but they can be represented in a more user friendly way with activity diagrams. The diagram in Fig. 5 illustrates the workflow for the "Feel Comfortable" AS of the example scenario. The diagram is annotated with labels from the source file in an attempt to close the gap between the high-level view of the diagram and the low-level view of the file. For example, the annotation in each box shows the activity type in the main sequence and the task name, the ontological searching term, as well as which ATRACO component, besides SM, has responsibility for running parts of this task.

The technical requirements for the deployment and testing of the ATRACO system include: the runtime versions of the ATRACO components with the specified service interfaces; the devices serving the scenarios, wrapped as UPnP devices; the domain and resource ontologies; the workflows specifying the tasks in each AS; and various third-party run-time libraries. The deployment of the system has been done in two IE testbeds using scenarios similar to the one discussed in this paper.

The implementation technologies and tools used are based on open frameworks and are compatible with the SOA paradigm. Java is the main programming language and UPnP enhanced with semantic descriptions [14] is used as the communication middleware for the integration of devices and services, instead of Web services. OWL has been used for the development of the ontologies as it provides a strong logical reasoning framework for the expression and enforcement of ATRACO policies and rules.

Although there are available (open source) execution engines for BPEL "programs" in ATRACO we need to build a layer upon such engines as a proxy in order to process the parts of the workflow description that are ATRACO-specific. In addition, most engines do not allow for dynamic binding and discovery of services. To address this limitation, the framework uses the SM as a proxy to communicate with service registries to obtain operational descriptions (e.g., UPnP or WSDL files) and instantiate services. This is achieved by encapsulating service search parameters in ATRACO-BPEL (see Table 2) as an input to the dynamic service binding process.

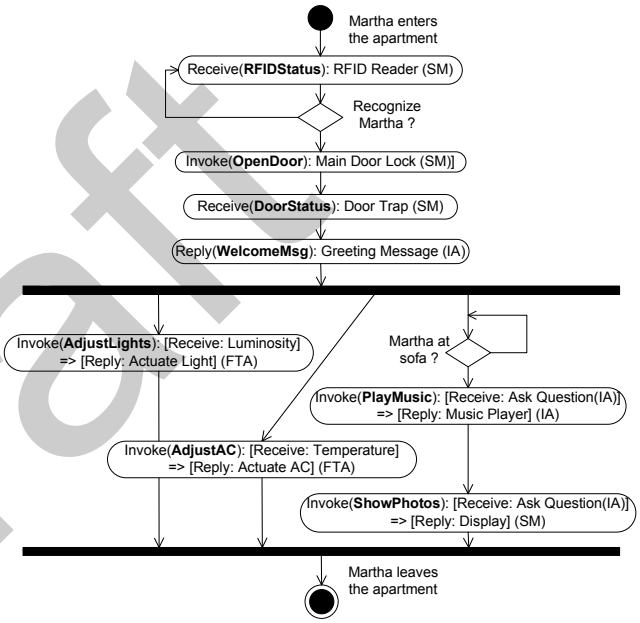

Figure 5. Annotated activity diagram for the "Feel Comfortable" AS

IV. RELATED WORK

There are systems that permit users or agents to aggregate and compose networked devices and services for particular tasks [13]. However, those devices are not context aware but act more as service providers, e.g., Web services usually in the UPnP style. Approaches to modelling and programming such devices for intelligent environments have been investigated, where devices have been modeled as collections of objects [15], as Web services [16], and as agents [17]. However, there has been little work on specifying at a high level of abstraction how such services would work together at the application level taking into account heterogeneous services, which can be combined in dynamic ways with the support of ontologies that provide knowledge representation, management of heterogeneity and semantically rich resource discovery.

Workflows have been extensively used on the composition of Web services. eFlow [18] uses graph-based model in which

the nodes denote invocations to service operations, and the edges denote control-flow dependencies. When a service node is invoked, a search recipe is executed to select a reference to a specific service. Once a service is selected by the search recipe, the eFlow execution engine is responsible for performing the dynamic binding using metadata that it stores in the service repository. A Polymorphic Process Model has been proposed in [19] to model abstract workflows without immediately requiring the implementation details of each activity. The SWORD toolkit uses a rule-based expert engine for determining how to construct a composite Web service from primitive services [20]. However, these approaches are targeted towards automation of business processes in the internet and not towards supporting users to perform activities in intelligent environments. The use of workflows to describe user tasks and interactions in smart environments has been explored in [21]. However, the approach is based on static workflows that do not react to changes in the context of the environment.

V. CONCLUSION

Normally, workflow management systems have not been used for dynamic environments requiring adaptive behaviour. On the contrary, in ATRACO we require adaptive workflows which need to react to varying environmental conditions. Our general idea is that since a workflow describes the relationship between services and if an agent is represented by such a service, then the relationship between the agents would be possible to specify. Following such a combined agent-based and SOA approach means that a workflow could be used to establish the initial relationships of the multiagent system. Multiagent systems can be specified then first with a workflow description using ATRACO-BPEL that defines the most common scenario and fault conditions. Once the basic system has been deployed, the agents could be working proactively so they can adapt to unforeseen circumstances and automatically handle the extension to the workflow description. In addition, run-time adaptations of services and devices are possible since workflows specify abstract services that are bound dynamically. This gives the opportunity to handle events such as a device failure or using a high-quality service that can replace a service selected in the first-place. In that sense, we can see our workflows as adaptive workflows.

As an extension, ATRACO will handle in the near future the simultaneous deployment of multiple workflows in the same space which may cause synchronization problems where different workflows may compete for the same resources and perform conflicting actions. In the future we would like to test our system in public intelligent spaces like shopping malls, museums, airports, etc., to explore how such spaces can be enhanced with services and mobile devices to assist visitors in performing their tasks more efficiently.

ACKNOWLEDGMENT

The research described is partly supported by the ATRACO (ICT-216837) project.

REFERENCES

[1] T. Erl, Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall PTR, 2005.

[2] J. Brønsted, K.M. Hansen, M. Ingstrup, "Service composition issues in pervasive computing", IEEE Pervasive Computing, vol. 9, no. 1, pp. 62-70, 2010.

[3] J. Rao, X. Su, "A survey of automated web service composition methods", LNCS, vol. 3387, Springer-Verlag, pp. 43–54, 2005.

[4] D. Hollingsworth, "The Workflow Reference Model", Workflow Management Coalition, Document Number TC00-1003, 1995.

[5] K. Verma, K. Sivashanmugam , A. Sheth, A. Patil , S. Oundhakar, J. Miller, "METEOR-S WSDI: a Scalable P2P infrastructure of registries for semantic publication and discovery of Web services", Information Technology and Management vol. 6, no. 1, pp. 17–39, 2005.

[6] Web Services Business Process Execution Language (WSBPEL), http://www.oasis-pen.org/committees/tc_home.php?wg_abbrev=wsbpel

[7] C. Goumopoulos, et al., "ATRACO: Adaptive and Trusted Ambient Ecologies", Proc. of the 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO), IEEE CS, pp. 96-101, 2008.

[8] J. Bidot, B. Schattenberg, S. Biundo, "Intelligent Planner", ATRACO ICT-1-8.2-216837, Technical Report (D21), 2010.

[9] C. Wagner, H. Hagras, "zSlices - towards bridging the gap between interval and general type-2 fuzzy logic", Proc. of the IEEE International Conference of Fuzzy Systems, pp. 489-497, 2008.

[10] G. Pruvost, A. Kameas, T. Heinroth, L. Seremeti, W. Minker, "Combining Agents and Ontologies to Support Task-Centred Interoperability in Ambient Intelligent Environments", Proc. of the 9th International Conference on Intelligent Systems Design and Applications, IEEE CS, pp 55-60, 2009.

[11] L. Seremeti, C. Goumopoulos and A. Kameas, "Ontology-based modeling of dynamic ubiquitous computing applications as evolving activity spheres", Pervasive and Mobile Computing, vol. 5, no. 5, pp. 574-591, 2009.

[12] N. Papadopoulos, A. Meliones, D. Economou, I. Karras, and I. Liverezas, "A connected home platform and development framework for smart home control applications," Proc. of the 7th IEEE International Conference on Industrial Informatics, pp. 402-409, 2009.

[13] R. Kumar, V. Poladian, I. Greenberg, A. Messer, D. Milojicic, "Selecting Devices for Aggregation, Proc. of the 5th IEEE Workshop on Mobile Computing Systems and Applications, pp. 150-169, 2003.

[14] K. Togias, C. Goumopoulos, A. Kameas, "Ontology-based Representation of UPnP Devices and Services for Dynamic Context-aware Ubiquitous Computing Applications", Proc. of the 2010 International Conference on Models and Ontology-based Design of Protocols, Architectures and Services (MOPAS), 2010.

[15] J. H. Jahnke, M. d'Entremont, and J. Stier, "Facilitating the programming of the smart home", IEEE Wireless Communications, vol. 9, no. 6, pp. 70-76, 2002.

[16] K. Matsuura, T. Hara, A. Watanabe, and T. Nakajima, "A new architecture for home computing", Proc. of the IEEE Workshop on Software Technologies for Future Embedded Systems, IEEE CS, pp. 71-74, 2003.

[17] F. Ramparany, O. Boissier, and H. Brouchoud, "Cooperating autonomous smart devices", Proc. of the Smart Objects Conference (sOc'2003), pp. 182-185, 2003.

[18] F. Casati, M.C. Shan, "Dynamic and adaptive composition of e-services", *Information Systems*, vol. 26 no. 3, pp. 143-162, 2001.

[19] H. Schuster, D. Georgakopoulos, A. Cichoki, and D. Baker, "Modeling and Composing Service-based and Reference Process-based Multi-enterprise Processes", LNCS, vol. 1789, Springer-Verlag, London, pp. 247-263, 2000.

[20] S. Ponnekanti, A.Fox, "SWORD: A developer toolkit for Web service composition", Proc. of the 11th World Wide Web Conference, Honolulu, USA, 2002.

[21] A. Ranganathan and S. McFaddin, "Using workflows to coordinate Web services in pervasive computing environments", Proc. of the IEEE International Conference on Web Services, IEEE CS, pp. 288-295, 2004.