Workflows Recognition through Multi Agents in Surveillance systems

Manolis Sardis[1], Vasilis Anagnostopoulos[1], and Theodora Varvarigou[1]

[1] National Technical University of Athens (NTUA), Athens, Greece
{sardis, vanag, dora}@telecom.ntua.gr

Abstract. Workflow management systems exactly enact business procedures and processes described in a process description language. This strict adherence to the prescribed workflow makes it impossible for the system to adapt to unforeseen circumstances. Surveillance systems have unpredicted information especially in difficult environments like the industrial ones. In this paper, we are presenting a workflow recognition architecture through the use of a multi agent system that controls and evaluates the recognized processes from the surveillance algorithms and adaptively creates environment warnings or alarms. The related methodology is based on Java technologies which are presented and latest innovations from the multi agents and workflow processes composition.

Keywords: multi agents, workflows, surveillance.

1 Introduction

Watching a video from an online camera that is performing surveillance in a room, in a road, or in an industrial environment like the picture in Fig. 1, is very difficult even for the human eyes to recognize objects, humans or even more workflows of objects and humans. In Fig. 1, the workflow is performed by two workers that are getting from the automobile spare parts boxes a spare part and transfer it to a soldering environment, where a robot is performing the related accurate and fast soldering. The sequence of events either in place or time is critical for the start and stop of robot activity. Any delay or fault from the workers could cause a delay or even worst a stop in the whole manufacturing chain. This is an example of a workflow problem, where humans need to know online and automatically the workflow steps and any possible problems, in order to protect and prevent system and business process delays and faults. This paper presents an architecture framework for solving the above problem, using the state of the art of multi agent technologies and web technologies according on the surveillance restrictions of the algorithms that provide the recognized objects or the humans in the scenes.

Fig. 1. An industrial environment, a workflow stages, human recognition (white box)

Paper section two presents the problem statement and its requirements. Section three continues with the scientific tools technologies that we are using for our architecture. Section four presents the architecture details and implementation, and finally in section five the conclusions with future research activities.

2 System and Problem Environment

The workflow sequence in Fig. 2, presents the three steps/states (x1, x2, and x3) that should be performed by one worker in order to fulfill the workflow number 1. The worker is available to go through a12 task to x2 state, and a23 to x3 state. The problem is that the worker can pause the execution from x2 to x3, if he returns back to x1 using the a21 route. The IT system should get all related information from the environment using the four sensors (y1, y2, y3, and y4), which are installed cameras for surveying the place there.

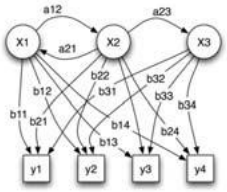

Fig. 2. Workflow composition from different procedures

The signals from the y_i (i=1,..,4) sensors should inform the IT system that the workflow has not been completed. These dynamic changes, in a workflow tasks (coming as signals b_j in the system) execution, which cannot be described in a static workflow sequence, are the base of our problem in that paper.

3 Scientific Tools

There are several commercial implementations of workflow engines, with their own proprietary formats. None of them is yet focused on multi-agent systems, but in technologies like web services. Web services workflow can be automated using many tools such as Web Services Business Process Execution Language (WSBPEL) [1] and Microsoft's X Language (XLANG) [2] specification. Workflow description is even more necessary, when composing a system from multiple web services, with multiple operations. Web Services Flow Language (WSFL) [3] can be used to define separately the flow model and the global composition model and its hierarchies. Web Services Conversation Language (WSCL) [4] focuses on protocols and document exchange and supports workflow concepts. WSBPEL is ideally for facilitating static Web Service composition. In case of dynamic workflows modeling we need an extra layer "*individual task layer*" where dynamic linking and manipulation is performed through WSBPEL. This layer in our case is performed by agents' functionality as will be presented in the rest of the paper for enhancing and integrating the dynamic manipulation of web services [5]. The comparisons in [6] show that semantic web services need higher level information for agent-based automation. Modeling a workflow as an extended transaction means that the sub-transactions correspond to

the tasks of the workflow and the execution structure of the extended transaction corresponds to the control flow of the workflow.

The main distinction between multi agents and web services is that web services are user-driven; multi agents act on their own [7] after user has given the necessary initial system instructions, providing adaptability on the workflow changes. Autonomous agents need more content information from the web service standards. Integrating web services and multi agent systems to a composite service requires a common method for using the system [8]; there has to be standard client software for the user, to contact agents, web services, or both. The need for this extra middleware software in our case is implemented through the usage of the *Modeler Agent*. The tasks and its position in the core architecture are presented in the following Fig. 3. The aim of this agent is to ensure that global constraints are not violated and that global efficiencies can be achieved.

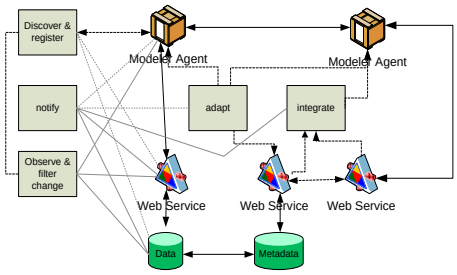

Fig. 3. *Modeler Agent* interconnections

According to Fig. 3, the *Modeler Agent* is responsible to perform all the intelligence of data (from sensors) manipulation. Data manipulation means that the web services data has to be evaluated and filtered in a manner readable for the end user. The system architecture is composed of multi-*Modeler Agents* that control and support a part of web services sensors. Each agent works standalone and is responsible for the online data from the web service. The distributed architecture, per sensor manipulation, benefits the system performance and response time, something very important for online systems and online characterization or identification of workflows in surveillance systems.

Workflows and Agents Development Environment (WADE) [9] enables a group of agents to cooperatively execute complex tasks defined as workflow. An Agent to perform its tasks may need to communicate with other Agents in the Platform. The Java Agent Development Framework (JADE) [10] provides the agents with the ability to communicate. WADE adds to JADE the support to the workflow execution. In our case, the use of WADE is to implement the internal behavior of each single system, and the full web services orchestration will be performed by BPEL usage.

4 Composition of Workflows Tasks and system Implementation

The workflows are synthesized by tasks and related procedures where an object/human has to perform in a sequence, and based on related time restrictions. The main types of workflows can be separated in two categories of related tasks/procedures of web services composition. The *static* and the *dynamic* web services composition, in order the system to recognize and label a workflow.

In *static* composition, the service to which the agents will be connected is determined before the workflow execution takes place (prior to run-time). Assume that we know exactly for a robot in the surveillance environment the root of its actions and the related paths during time. For internal automation of tasks, the recognition of the sub events from the *Camera Agent* (

Fig. **4.**) are composed by the workflow *Modeler Agent*, which links services offered by different services (from different *Camera Agents*) creating a static composition since the existence of such a service is known prior to run-time (for the case of the robot performance in the scene) and the agent can have design-time knowledge of connecting to such a service. Static composition of web services can be implemented through execution of BPEL constraints. Although the case works perfect for constant robots' based routes in a scene, not all services and procedures are known in case of humans actions, especially during the workflow design time. However, some of the services need to be discovered during workflow execution (run-time).

Dynamic composition of web services can be supported through the multi agents' environment where agents' services provide system adaptability [11]. Using a number of cameras and related services per *Camera Agent* the intelligence system is able to provide the correct answers for the recognized event and its position in the workflow sequence. This 'intelligence' is performed through the *Modeler Agents* that using WADE and according on the workflow process constraints, are manipulating and report the achievement or failure of the related surveyed workflow. The end user is informed online through the generated alarms or warnings signals from the gui. Dynamic composition and adaptability support can be ensured by applying more agents in the surveillance scene. The *Sensor Agents* can provide more detailed information of sub tasks. This information that can be from the simple *start* and *stop* of an event, into more detailed functions of the workflow allows the delays, the route changes, and the time and position drifting from the original positions of the surveyed object. The result is that the *Modeler Agents* have available extra information that limits possible dummy workflows processes that could create errors on the workflow recognition.

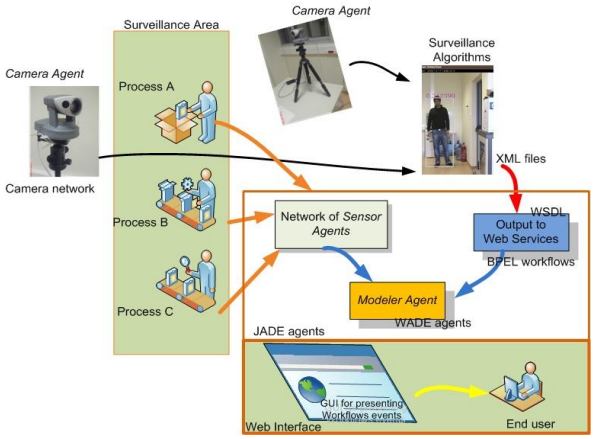

Fig. 4. Architecture framework

The system has been modeled using Java technologies on web services and on multi agents systems. The JADE platform has been used for the construction of the

agents and their manipulation. The surveillance system consists during the lab tests of 3 PTZ Axis 213 cameras saving the surveyed scenes in .jpg files per frame. The network is a Gigabit Ethernet for the control of the cameras network and the agents virtual machines have been applied in 2 computers. Each *Camera Agent* is collecting the surveillance information from the object detection and tracking algorithms and then through an xml format is getting this information from a related web service. Composition of the web services information is done through the *Modeler Agent*. This JADE based agent controls the inputs and cooperates with the extra *Sensors Agents* for extracting correct workflows recognitions. Finalization of the whole process is performed through the end user GUI that integrates the whole architecture.

5 Conclusions

Through this paper the authors presented an architecture for workflows recognition using multi agent technologies on surveillance systems, under complex environments for surveillance of workflows, like the industrial environments. The paper presents the main architecture components and the innovation in manipulating dynamic composition of the processes that a workflow contains and characterizes it. Already the main building blocks of the proposed architecture have been implemented by the authors and there is an on going work especially on the Modeler agent rule engine integration and simulation.

Acknowledgments

The work presented by this paper is supported by the FP7 European Research Project SCOVIS (g. a. no. 216465) (http://www.scovis.eu).

References

1. WSBPEL standard, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html
2. XLANG, http://msdn.microsoft.com/en-us/library/aa577463%28BTS.20%29.aspx
3. WSFL, http://xml.coverpages.org/wsfl.html
4. WSCL, http://www.w3.org/TR/wscl10/
5. Bohn, H., Golatowski, F., Timmermann, D.: Dynamic device and service discovery extensions for WS-BPEL. International Conf. on Service Systems and Service Management, pp.1-6, (2008)
6. Sollazzo, T., Handschuh, S., Staab, S., Frank, M., Stojanovic, N.: Semantic Web Service Architecture–Evolving Web Service Standards toward the Semantic Web, Proc. of the 15th International FLAIRS Conference, (2002).
7. Xie, N., Cao, C., Ma, B., Zhang, C., Si, J.: Ontology-Based Web Agents Using Concept Description Flow, Knowledge-Based Intelligent Information and Engineering Systems 8th International Conference, KES 2004, Wellington, New Zealand, Proc. Part II, (2004)
8. Montebello, M., Abela, C.: DAML enabled Web Services and Agents in the Semantic Web, NetObject Days, Lecture Notes in Computer Science, Web, Web-Services, and Database Systems, Volume 2593/2009, pp. 46-58, (2002)
9. Giovanni, C.: WADE Tutorial, AAMAS 2008, (2008)
10. JADE environment, http://jade.tilab.com/
11. Briot, J.P., Muñoz-Meléndez, A., Cardon, A.: Adaptability and Embodiment Using Multi-Agent Systems, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, Volume 2322/2002, pp. 121-150, (2002).