From Modeling to Enactment of Distributed Workflows:
An Agent-based Approach
Giancarlo Fortino, Alfredo Garro, Wilma Russo
DEIS – Università della Calabria
Via P.Bucci, cubo 41C, 87036 Rende (CS), Italy
+39.0984.494063
{g.fortino, a.garro. w.russo}@unical.it

ABSTRACT
This paper proposes an agent-based approach which seamlessly supports the phases of modeling and enactment of distributed workflows. An agent-based distributed workflow model is basically obtained by instantiating a MAS (Multi-Agent System) meta-model for the management of distributed workflows by means of a workflow schema based on the Workflow Patterns. In order to enact an obtained agent-based model, a JADE-based workflow enactment framework, directly designed upon the MAS meta-model, was developed.

Categories and Subject Descriptors
C.2.4 [Distributed Systems]: Distributed Applications.

General Terms
Design, Management.

Keywords
Multi Agent Systems, Workflow Patterns, Distilled Statecharts.

1. INTRODUCTION
In the last decades Workflow Management Systems (WFMS) have been developed to provide support to the modeling, improvement and automation of business management, industrial engineering and data-intensive scientific processes which are characterized by being composed of many interrelated activities distributed across organizations [1,3]. Since they require an effective modeling paradigm and an efficient management technology, the emerging Agents paradigm and technology [5] can be exploited due to their widely recognized suitability for the modeling and implementation of complex software systems in distributed environments.

In this paper, we propose an agent-based approach which aims at covering the phases of modeling and enactment of distributed workflows so exploiting the enormous potential of Agents in completely supporting the development of complex systems from their modeling to their implementation. The modeling phase is enabled by a MAS (Multi-Agent System) meta-model for the management of workflows which can be instantiated by a workflow schema based on the Workflow Patterns (WPs) identified by van der Aalst [1]. An instance consists of a MAS model which represents the model of an enactment engine able to manage instances of the workflow schema used for the instantiation. The implementation phase is supported by a workflow enactment framework based on JADE [2] and designed on the basis of the MAS meta-model so allowing a quasi-seamless transition from modeling to enactment of distributed workflows.

2. A MAS META-MODEL FOR DISTRIBUTED WORKFLOWS
A MAS is a system composed of a number of autonomous entities, named agents, having a cooperative behavior that allows to obtain the desired function/service. A MAS meta-model is a structural representation of the elements (agent, role, behavior, ontology, etc) that constitutes a MAS along with their composition relationships. A MAS meta-model for the management of distributed workflows should formalize the types, the acquaintance relationships and the behaviors of the agents involved in the activation, execution and monitoring of a distributed workflow schema. The instantiation of the MAS meta-model should produce a MAS model able to enact the workflow schema used to instantiate the MAS meta-model. A produced MAS model is therefore the agent-based enactment engine model of the Workflow Schema (WS) used for the instantiation. Our MAS meta-model (Fig. 1) consists of two layers: Structural and Behavioral.

The Structural layer defines the types of agents involved in the management of a distributed workflow and their acquaintance relationships. In particular, the types of agents are:
- EnactorAgent, which is responsible for the activation and monitoring of workflows.
- ManagerAgent, which is responsible for the execution and control of workflows. A single ManagerAgent allows for flat workflow management whereas a hierarchical structure of ManagerAgents, formed according to the parent/child model, allows for a hierarchical workflow management.
- TaskAgent, which is responsible for the execution of internal tasks and/or for the wrapping of external tasks.

The associations among the agent types are:
- EnactorAgent/ManagerAgent, which is enabled by the Enactor/Manager Interaction Protocol (EMIP).
- ManagerAgent/ManagerAgent, which is enabled by the Manager(master)/Manager(slave) Interaction Protocol (MMIP).
- ManagerAgent/TaskAgent, which is enabled by the Manager/Task Interaction Protocol (MTIP).

The Behavioral layer defines the structure of the behavior of the ManagerAgent, or ManagerBehavior, which is composed of:
- one InitialPseudoActivity, which represents the starting point of the workflow execution in the WS.
- one or more FinalPseudoActivity, which represent points in the workflow schema at which the workflow or a part of it ends. A
FinalPseudoActivity uses a master ManagerAgent for notification purposes. - one or more WFPattern, which represent the control-flow activities. A WFPattern, which can be any of the available WPs [2] (sequence, and-split, and-join, xor-split, xor-join, or-split, multi-merge, discriminator, loop, multiple instances, deferred choice, milestone, etc), uses one or more TaskAgent and one or more child ManagerAgent for activation purposes and a parent ManagerAgent for notification purposes. In order to model a WS, InitialPseudoActivity, FinalPseudoActivity, and WFPattern are linked through source/target control-flow associations.

3. THE JADE-BASED WORKFLOW ENACTMENT FRAMEWORK

On the basis of the MAS meta-model we have designed and implemented in JADE an agent-based framework for the enactment of distributed workflows which aims at supporting the implementation of general workflow enactment engines or specific workflow systems. The structure of the framework is directly designed upon the MAS meta-model classes (Fig. 1). The dynamics of the framework are defined by specifying the interaction protocols among the agent types, the logic of the ManagerBehavior, and the activity of each WFPattern. The specifications were carried out by means of the Distilled StateCharts [4] formalism, an agent-oriented dialect of the Statecharts. According to the WS to enact, the ManagerBehavior in the ControlFlow superstate executes the next control-flow action by invoking the executeNextControlAction method which fetches the next WFPatterns and executes them. Upon completion of a WFPattern execution, two events are generated: (i) executeNextControlAction which allows the ManagerBehavior to invoke the executeNextControlAction method; (ii) stateChangeNOTIFICATION which allows notifying the upper-level ManagerAgent or the EnacterAgent about the control-flow state change of the workflow. If there are no more WFPatterns to execute, the terminateControl event is generated which drives the termination of the ManagerBehavior and the transmission of the related ENDNOTIFICATION to the upper-level ManagerAgent or to the EnacterAgent. The basic actions that a WFPattern can perform are: (i) detection of the completion of a task through the reception of a FIPA ACL message which can also carry the data produced by the completed task; (ii) creation and/or activation of TaskAgents and/or ManagerAgents.

The class diagram of the JADE-based framework is an extension of the MAS meta-model class diagram (Fig. 1). EnactorAgent, ManagerAgent, and TaskAgent extend the Agent class of JADE [2]. EnactBehavior extends OneShotBehaviour, which is a behavior whose action method is executed only once; in fact, as soon as the enactment of the workflow terminates, the behavior of the EnacterAgent terminates too. TaskBehavior and WFPattern extend Behaviour which represents a generic behavior terminating when the end-of-activity condition is met. ManagerBehavior extends FSM Behaviour which models a complex task whose sub-tasks correspond to the activities performed in the states of a finite state machine. In particular, the states of ManagerBehavior correspond to the control-flow states of the workflow (or sub-workflow) that the ManagerAgent is controlling; each state is associated to a WFPattern which is activated when the state becomes active. EMIP, MMIP, and MTIP are appositely defined through sequences of ACL messages, instances of the ACLMessage class, whose performative is INFORM.

4. CONCLUSIONS

This paper has proposed an agent-based approach for the modeling and enactment of distributed workflows. The efficacy of the modeling phase results from the synergic exploitation of the Agents paradigm and the workflow patterns identified by van der Aalst, whereas the flexibility and robustness of the enactment phase mainly depend on the JADE-based workflow enactment framework. The approach has been successfully applied to the implementation of a distributed workflow system for the monitoring of distributed agro-industrial productive processes.

5. ACKNOWLEDGMENTS

This work was partially supported by the Italian Ministry of Instruction, University and Research in the framework of the M.ENTE (Management of integrated ENTERprise) research project PON N°12970-Mis.1.3. The authors also wish to thank Serena Martino for her contribution to the implementation of the JADE-based framework.

6. REFERENCES