Software Agents for Workflows Support

Zakaria Maamar, Nader Troudi, and Pamela Rostal

1 Introduction

The purpose of this paper is to describe an on-going research project that deals with the use of *Software Agents* (SAs) in the design, development, and support of *Workflow* (WF) systems.

With the increasing development of information and communication technologies, users are becoming more and more demanding on companies and hence, looking for new kinds of specialized services and advanced tools. To cope with these growing and complex requirements, system designers and developers need new approaches to support their work. As an emerging technology, SAs seem to be a good candidate and a promising approach [1]. Within an organization, SAs would be able to decide with whom to collaborate, what services to offer, what services to require, and what "visible parts" (from private to public and *vice-versa*), in term of behaviors, to exhibit.

To coordinate and streamline their business processes, organizations tend to use WF systems and hence, succeed to identify who is in charge of doing what [2]. Generally, WFs are designed and saved for later use. Therefore, these WFs are *pre-defined* and *static*. In this context, one potential *drawback* of the WFs results from their *no-adaptability*. For example, if a situation that raises an exception has not been taken into account during the WF design, then this WF will be stopped and often, restarted. Moreover, with the recent progress of communication middlewares, for instance Object Request Brokers (ORBs), and programming languages for distributed applications, for instance JAVA, WFs need to be *flexible* and *scalable*. In this paper, as a possible solution, we suggest to consider a WF as a dynamic collection of *collaborative processes* that are carried out by SAs. Whenever needed and taking into account the characteristics of the environment in which they evolve, SAs *cluster*, i.e. "glue", processes and *order* them in order to constitute the WFs. Such an approach offers a great flexibility since process combination could be directly based on the agent's states, rather than being pre-defined in advance. Moreover, in this approach processes could be re-used as components of other WFs. Hence, the same process could be involved in different WFs.

Within the Defence Research Establishment Valcartier (DREV), we are conducting a research project that deals with the integration of SAs and WFs in the design of a Commander-Advisor System (CAS). This system is dedicated to the Fighter Group Operations Center (FGOC) that prepares combat plans and monitors their fulfillment. The FGOC consists of an *executive element* and a *support staff*[1] (Figure 1). The support staff has two groups: G3 Operations and G3 Plans. The executive element consists of the Commander (COMD); G3 heads the support staff; G4 heads the logistics staff; and

[1] For our research needs, we assume that the executive element and the support staff are not located in the same building. We aim at studying WFs taking place in a distributed environment.

finally, G5 heads the strategic planning staff. The support staff reports to the executive element through G3. These designations, i.e. COMD, G3, G4, and G5, will be used throughout this paper. Currently, the different groups carry out their operations manually. For example, monitoring an area of operations requires from a person to detect events, analyze the gathered information, and produce documents. We believe that it would be more appropriate if such operations could be "*packaged*" into multiple WFs and completed in collaboration with SAs. In this paper, we describe the SAMWF (for *Software Agents Meet WorkFlows*) approach used in the design of the FGOC's Commander Advisor System.

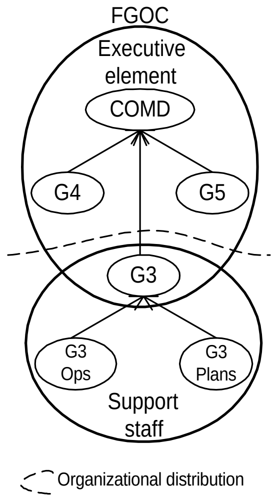

Figure 1 FGOC organization

The remainder of this paper is structured as follows. Section 2 describes the functioning of the FGOC and defines the concepts of SAs and WFs. Section 3 introduces the four models of the SAMWF approach, respectively called role, agent, service, and contract. Section 4 surveys the related work. Finally, Section 5 summarizes the paper and gives insights on topics for further research.

2 Background

This section is divided into two parts. The first part describes the FGOC's functionalities and the second part briefly defines the concepts of SAs and WFs.

2.1 FGOC Functional Overview

In what follows, the FGOC description has been simplified for the purpose of explanation. The FGOC is responsible for allocating the regional resources and reacting to the readiness preparation commands from Air Command. In most times, the FGOC support staff only mans the FGOC. The executive element becomes involved in certain situations, such as the creation of a new Air Tasking Order (ATO) to the management of a battle.

When an ATO is required, G3 orders G3 Plans staff to prepare possible solutions, i.e. options, according to the resource requirements of the occurring events and the state of available resources. Then, G3 presents the options to the executive element. This latter assesses the options based on certain higher level considerations and chooses the appropriate option. Next, G3 Plans staff generates a new ATO according to the option chosen by the executive element. Finally, G3 Plans staff disseminates the ATO into subordinate units.

When a reactive response is required in a battle management process, G3 Ops staff prepares the options. G3 and/or the COMD, in case he is also involved, assess the options and choose the appropriate one. Then, G3 Ops staff generates an immediate ATO and disseminates it into subordinate units.

2.2 Basic Concepts

Software Agents: in Distributed Artificial Intelligence, researchers have studied several issues related to the distribution and coordination of knowledge and actions in environments involving multiple entities, called agents. Agents can take different forms depending on the nature of the environment in which they evolve. A particular type of agents, SAs, has recently attracted much attention. SAs are autonomous entities having the abilities to assist users when performing their operations, to collaborate with each other to jointly solve different problems, and to answer users' needs.
Workflows: a WF 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" (The Workflow Management Coalition). To handle these operations, WorkFlow Management Systems allow designers to design WFs, to manage their instantiation and execution, and to integrate distributed and heterogeneous applications within these WFs.

3 Presentation of the SAMWF Approach

The SAMWF approach is addressed to designers who aim at developing systems based on SAs and WFs. To achieve this operation, the SAMWF approach suggests four models, respectively called *role, agent, service*, and *contract* (Figure 2). Before we detail these four models and their concepts, we define the main principles of the SAMWF approach.

3.1 SAMWF Overview

The SAMWF approach uses two basic elements: SAs and WFs. SAs are used to specify the architecture of the future system, in which the WFs are used to describe this system's operations.

A SA is set up to satisfy users' needs, and hence is assigned to play one or several *roles*. An assignment depends on the role's requirements and the agent's capabilities (Figure 2-A). Furthermore, each a user need is characterized by a *service* that is offered by an agent (Figure 2-B). This agent takes the *responsibility* of fulfilling the service according to the WF associated with this service.

A service consists of several *processes* (Figure 2-C) that belong to the same agent. When an agent cannot carry out a process on its own, i.e. it lacks specific resources, it requires other agents' processes as defined in the *contracts* that are signed between these agents (Figure 2-D). In fact, contracts capture *commitments* between agents.

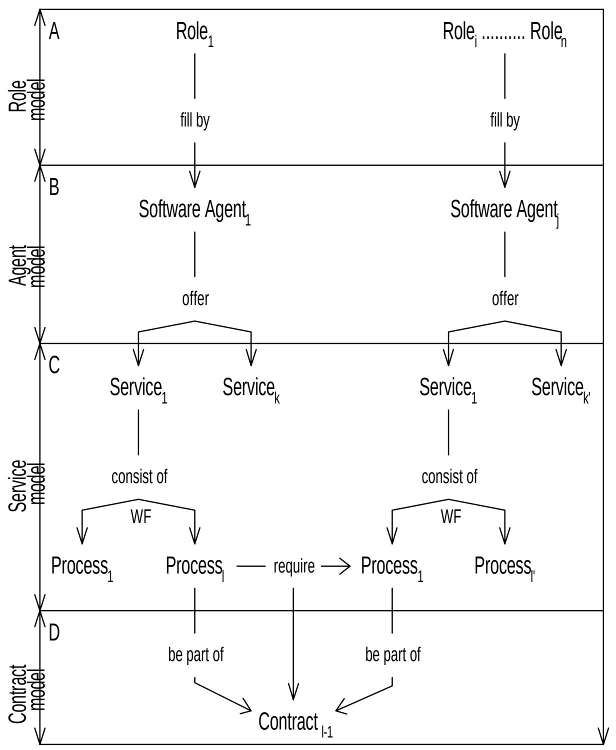

Figure 2 Models and principles of the SAMWF approach

In Figure 2, the different models of the SAMWF approach are presented top-down, but some of them could be done concurrently. Moreover, going down-top could be useful in practice.

3.2 Role Model

In the SAMWF approach, the purpose of the *Role Model* is to illustrate the organizational structure in which the future system will be used. A role illustrates a *position* in the organization, for example manager. In the SAMWF approach, agents are introduced in order to fulfill the functionalities of the roles (Section 3.3).

A role is identified by a set of operations, called *processes* with respect to the SAMWF terminology, which are performed by the person filling this role. This person has the *capabilities* to meet this role’s *requirements*, in terms for instance years of experience, fields of expertise, qualification, etc. Moreover, the processes of a role are packaged into different *services*. These services are offered to other persons either in the same organization or in other organizations. Here, the term service denotes a computing procedure, for example battle management for the FGOC. To perform a service, a person uses its own processes, called *internal*, and may require other processes, called *external*, from other persons. In the role model, four types of attributes define a role (Table 1). The first attribute corresponds to the *name* of the role. This name is the label of the position in the organization. For instance, a commander is a role in the FGOC. Next, the second attribute corresponds to the *responsibilities* of the role. Responsibilities are the services associated with the role. For example, battle management is a responsibility of the commander role. The third attribute corresponds to the *rights* given to the role to use the external processes of other roles. Finally, the fourth attribute corresponds to the *authorizations* given to the role to provide its internal processes to other roles.

Table 1 Role definition

Name	Responsibilities *(services to offer)*	Rights (*external processes*)	Authorizations *(internal processes)*
Example			
• Plan developer (G3 Plans)	• Air plans • Air tasking orders	• Dissemination of tasking of G3 Ops	• Situation post-analysis to G5

In the example of Table 1, Plan developer role is described as follows:

- Affiliation: G3 Plans group.
- Responsibilities: Air plans; Air tasking orders.
- Rights: Plan developer has the right to use Dissemination of tasking process of G3 Ops group.
- Authorizations: Plan developer has to provide its Situation post-analysis process to G5 group.

Rights vs. Authorizations

Table 2 presents the relationship that exists between the rights and authorizations of roles' processes. Let *n* be the number of roles in an organization; the notation that is used is as follows:

- Ext. for External; Int. for Internal.
- $\forall$ i, j, k, k', x $\in$ N^* and i$\neq$j
  $Role_i.Right_{jk}$ means that $Role_i$ has the Right to use $Process_k$ of $Role_j$.
  $Role_i.Authorization_{jk}$ means that $Role_i$ has the Authorization to provide $Process_k$ to $Role_j$. In fact, $Process_k$ belongs to $Role_i$.

Table 2 Rights vs. Authorizations

$Role_i$		$Role_j$ (j≠i)	
Ext. Processes	$Right_{jk}$	Ext. Processes	$Right_{xk'}$ ($\exists$x, x=i)
Int. Processes	$Authorization_{jk}$	Int. Processes	$Authorization_{xk'}$ ($\exists$x, x=i)

Remarks: the following relationships are obtained from Table 2

- **When** x=i **and** k=k'
  Then $Role_i.Right_{jk} = Role_j.Authorisation_{xk'}$
  and
  $Role_i.Authorisation_{jk} = Role_j.Right_{xk'}$.
- $\cup_{i=1\ldots n,\ i\neq j} Role_i.\{Right_{jk}\} = Role_j.\{Authorization\}$
  $Role_i.\{Authorization\} = \cup_{j=1\ldots n,\ j\neq i} Role_j.\{Right_{xk'}\}$

3.3 Agent Model

In the SAMWF approach, the next step to complete after the role model is the *Agent Model*. The purpose of the agent model is to describe the basic types of agents that are integrated into the system and the relationships between these agents. In the agent model, relationships are viewed as contracts. In the SAMWF approach, the mapping between the roles of the role model and the types of the required agents of the agent model is not *one-to-one*. For instance, several roles could be associated with just one type of agents.

Therefore, this agent inherits all the characteristics of these roles, for instance their responsibilities, rights, and authorizations.

In the agent model, each agent is represented as a class that is divided into five parts (Figure 3). The first part contains the agent *type*, for example Executive-Agent. This agent will have one or several roles to play in the system. The second part contains the *services* that are offered to users by this agent. Each service is associated with a WF. The third part contains the agent *knowledge* in terms of *goals*, *beliefs*, and *intentions*. A goal indicates a potential state that an agent wants to reach, for example looking for specific information in order to satisfy its user. A belief identifies an information about the world that an agent considers being true or false during a period of time, for example an enemy attack plane has been detected at 3.00pm north of Quebec City. Finally, an intention states what the agent is willing to do. The fourth part contains the internal processes the agent performs. These processes are also, available to other agents through *initiation interfaces*[2]. Interfaces are detailed in the *Contract Model* (Section 3.5). Finally, the fifth part contains a reference to the external processes the agent requires from other agents to complete its services. External processes are identified using the role model (Section 3.2) and the *relationships* that exist between agent classes. In the agent model, each internal/external process is a set of *elementary tasks* and has a *partial goal* to reach. Furthermore, each service accomplishment requires a combination of several internal and/or external processes.

In the agent model, when agents belong to different organizational units, a *vertical line* crosses the relationships that link these agent classes. The purpose is to illustrate the organizational aspect in the agent model.

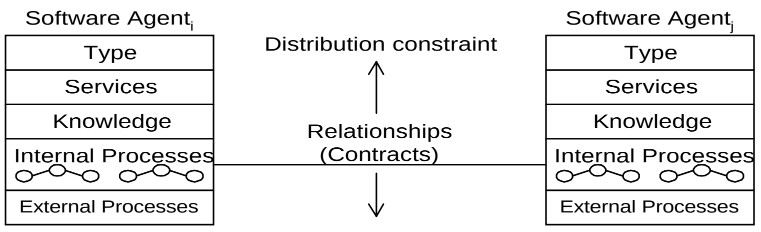

Figure 3 Agent model

Figure 4 illustrates a part of the FGOC agent model. Currently, each FGOC structure, i.e. executive element and support staff, is mapped into an agent class namely Executive-Agent and Support-Agent. Furthermore, each class could be divided into other agent classes (aggregation concept in object-oriented methods). For example, the Support-Agent class could be based on three agent classes: G3-Agent, G3-Plans-Agent, and G3-Ops-Agent. In Figure 4, the Executive-Agent class consists of:

- Battle mgnt as a service to a commander.

[2] The idea of the initiation interfaces corresponds to the interoperability level 5 of the Workflow Management Coalition [3] and the JointFlow workflow facility [4] of the Object Management Group.

- Knowledge instantiation in terms of goals, beliefs, and intentions.
- Personnel mgnt as an internal process.
- Sit. monitoring as an external process of Support-Agent.

In Figure 4, Executive-Agent and Support-Agent are related through the $\text{Contract}_{p1\text{-}p2}$, called personnel-situation.

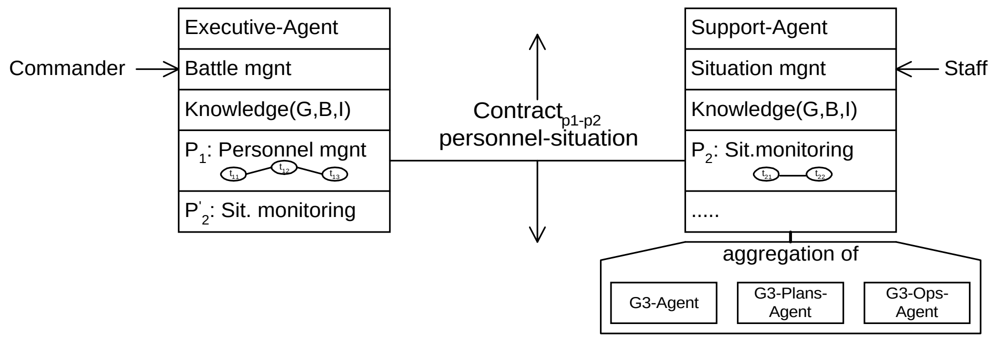

Figure 4 FGOC agent-model

3.4 Service Model

In the SAMWF approach, the next step to complete after the agent model is the *Service Model*. The purpose of the service model is to illustrate how an agent service is performed in terms of internal processes. The external processes are, also, identified in this step.

In the SAMWF approach, each service that is initiated by a user is associated with a WF. This WF has a *global goal* to reach (Figure 5). In the service model, a WF is a set of internal processes that are *clustered* together. The agent that offers this service is the one that performs the operations of goal decomposition [5] and clustering. This clustering consists of creating the dynamic links between the processes and hence, obtaining the needed WFs. Initially, the agent's intention to perform a service is mapped into a global goal. According to the complexity of this goal and the knowledge, i.e. beliefs, it has, this agent decomposes this global goal into a set of *sub-goals* until each sub-goal is associated with a process. Sub-goals are called partial goals with respect to the SAMWF terminology. Finally, each process is detailed in terms of *tasks*, *resources*, and *chronology*. If a task of a process requires a resource of type external process, then this element will be detailed in the Contract Model.

Table 3 and Table 4 summarize the main results that are obtained from the service model. In Table 3, an agent associates Service_1 with WF_1. Depending on the conditions in which this agent evolves, in term of beliefs, this service requires three processes, namely Process_1, Process_2, and Process_3. These processes are clustered and executed differently. For example, under $\text{Condition}_1 = \text{Belief}_1$ and Belief_2, the execution chronology starts with Process_1 *then* Process_2. Furthermore, Process_1 is decomposed into two tasks: Task_1 and Task_2. Task_1 requires an internal resource. Therefore, a contract model is not needed. However, this is not the case for Task_2.

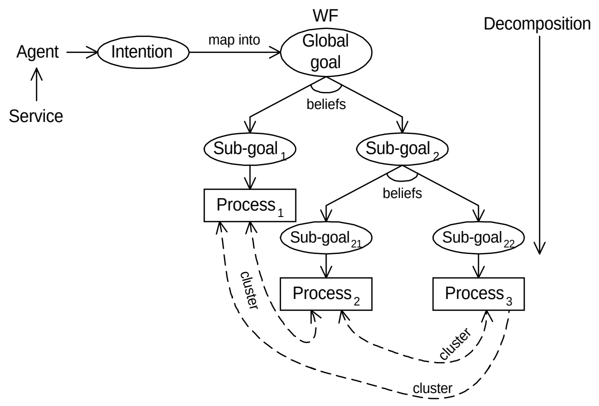

Figure 5 Service model

Table 3 Service description

Identification	Internal Processes	Execution chronology	Under Conditions (beliefs)
Service$_1$: WF$_1$	Process$_1$ Process$_2$	Process$_1$ *then* Process$_2$	Condition$_1$: Belief$_1$ *and* Belief$_2$
	Process$_1$ Process$_2$	Process$_2$ *then* Process$_1$	Condition$_2$: Belief$_1$ *and* Belief$_3$
	Process$_3$	Process$_3$	Condition$_3$: Belief$_2$

Table 4 Process description

Process id	Task id	Type of resources to require (Internal/External)	Contract Model
Process$_1$	Task$_1$	Resource$_1$ (internal)	No
	Taks$_2$	Resource$_2$ (external)	Yes

In what follows, we illustrate Battle mgnt service of Executive-Agent of Figure 4. First, we provide a service model and then, a description of Personnel mgnt process. In Figure 6, manage battle goal is decomposed into two sub-goals: terminate operations and start battle. Next, start battle goal is decomposed into three sub-goals: manage personnel, assess situation, and end battle. Personnel mgnt process of manage personnel sub-goal is divided into three tasks (Table 5): recall personnel; characterize situation; and fill posts. The first task needs to consult FGOC personnel database. The second task requires a report from the Support-Agent. The process Sit. monitoring of this agent provides this type of reports. However, this process is external to the Executive-Agent. Therefore, the link (Task$_{\text{characterize situation}}$,Process$_{\text{Sit. monitoring}}$) will be detailed in the contract model. Finally, the third task requires the Form DND 13-34A.

Example of a decomposition rule:

If Goal(manage_battle) and Belief(detect_enemy_plane)
Then Sub-Goal$_1$(terminate_operations) *and* Sub-Goal$_2$(start_battle)

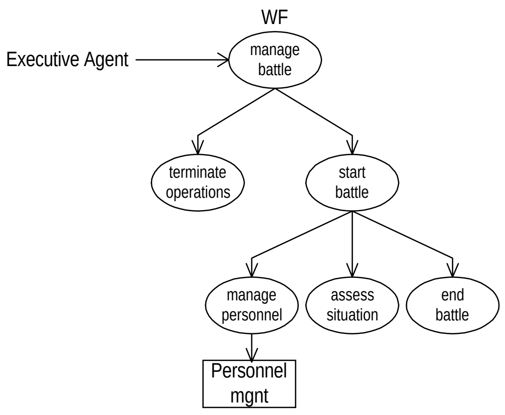

Figure 6 FGOC service-model

Table 5 FGOC Process-description

Process id	Task id	Resource to require and Type (Internal/External)	Contract Model
Personnel mgnt	recall personnel	FGOC's personnel DB (internal)	No
	characterize situation	Sit. monitoring process (external)	Yes
	fill posts	Form DND 13-34A (internal)	No

3.5 Contract Model

In the SAMWF approach, the final step to fulfill is the *Contract Model*. The purpose of the contract model is to describe the interactions that take place between agents, and particularly between their processes. Interactions are handled by the initiation interfaces as defined in the agent model (Section 3.3).

In the SAMWF approach, an initiation interface defines how an agent invokes an external process without taking into account its implementation details (Figure 7). Thanks to these interfaces, agents' processes can be defined in different languages. For example, if G3 technical staff masters a particular definition language, then it will be easier for this staff to keep using the same language in case it is involved in the design of cross-organizational WFs. Keeping in mind the characteristics of each organizational unit is important. However, to achieve cross-organizational WFs, initiation interfaces have to be well defined and standardized between agents.

In the contract model, four types of attributes characterize an initiation interface (Table 6). The first attribute is the *identifier* of the interface and corresponds to the task that requires an external process. Next, the second attribute is the *identifier* of the contract and corresponds to the name of the processes that are involved in this contract. The third attribute corresponds to the *input* parameters that are transmitted from the task to the external process. Finally, the fourth attribute corresponds to the *output* parameters that need to be instantiated, thanks to the external process. Each input and output parameter has two types: *M* for *Mandatory* and *O* for *Optional*. For example, an input parameter with a mandatory type has to be sent to the external process. This process requires this parameter to perform its operations.

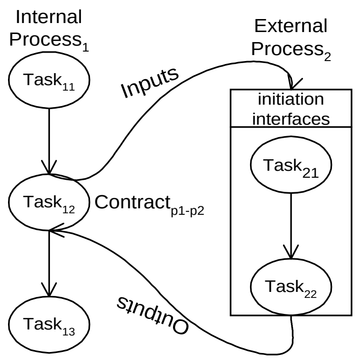

Figure 7 Contract model

Table 6 Initiation-interface definition

Task identifier	Contract identifier	Input parameters		Output parameters	
$Task_{12}$	$Contract_{p1-p2}$	$In.par_j$ = $value_1$	M	$?Out.par_i$	O
		$In.par_{j+1}$ = $value_3$	O	$?Out.par_{i+1}$	M

Figure 8 illustrates a part of the FGOC contract model. In Table 5, Personnel mgnt process of Executive-Agent has been decomposed into the following tasks: recall personnel; characterize situation; and fill posts. Sit. monitoring process of Support-Agent is decomposed into the following tasks: collect data; analyze data; and compile data; As stated in Table 5, characterize situation task requires information from Sit. monitoring process of Support-Agent. The initiation interface between this task and this process is detailed in Table 7.

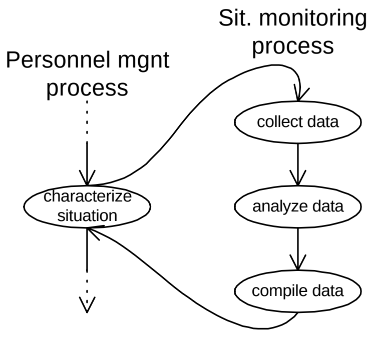

Figure 8 FGOC contract-model

Table 7 FGOC Initiation-interface

Identifier	Contract identifier	Input parameters		Output parameters	
characterize situation	$Contract_{personnel-situation}$	Planes-on-standby=2	M	?situation-type	M
		Planes-in-patrol=2	M	?increase-planes	M

We assume that Sit. monitoring process assigns the value *critic* to ?situation-type parameter and the value *true* to ?increase-planes parameter.

4 Related Work

There exists a number of research projects in the WF management community. All these projects aim at meeting the requirements of dynamic environments. Such environments are characterized by different elements, for instance alliances between financial institutions. One way of achieving these alliances is to link theses institutions' processes, while keeping in mind the characteristics of each institution such as technical standards and terminology. Debenham [6] proposed a 3-layer BDI agent architecture to constructing a WF system. Rostal [7] reported how autonomous agents were used to implement the business process and entity components that compose a WF-intensive call center application. Merz et al. [8] used mobile agents to supporting inter-organizational WF management.

Yu and Schmid [9] explored a framework for agent-oriented and role-based WF modelling. They defined a WF as a set of relating roles that are assigned to agents. Roles and agents correspond to our role and agent models. However, even if WF coordination is achieved by communication between agents in Yu and Schmid's framework, there are no details on the coordination policy as well as the communication language that are used. Service and contract models in the SAWMF approach illustrate the WF coordination at the process level. Ludwig and Hoffner [10] suggested contract-based cross-organizational WFs. However, concepts such agents, roles, flexibility, and services do not exist. Finally, Jennings et al. [11] delegated various components of a business process to a number of agents that negotiate in order to coordinate their actions and to buy the services they require. Negotiation needs a strategy that could be part of our service model.

5 Conclusion

In this paper, we presented how collaborative autonomous software-agents address at least two fundamental limitations of traditional workflows: their static nature and their inability to exist outside workflow engines [12]. Allowing agents to construct workflows dynamically in accordance with their assessment of an emerging situation generates tremendous benefits to an organization that must respond immediately to critical situations.

Furthermore, we focussed our presentation on the SAWMF approach for developing systems based on SAs and WFs. SAs constitute the backbone of these systems while WFs define the behavior of these SAs, when fulfilling these systems' functionalities. The SAWMF approach integrates four models, respectively called role, agent, service, and contract. The role model is the starting point and has an organizational flavor. A possible extension to this model that should be taken into account is services *delegation*. For instance, if a person that fills a role is absent, his services should be delegated to other persons. Delegation is very important, particularly when we evolve in a military context.

The SAWMF approach has be used in the design of a Commander-Advisor System. We have initiated the deployment phase. We are using the Java language to define the behavior of SAs, the ORB Visibroker for Java to carry out distributed operations, and Jess, as an inference engine, to generate WFs.

Références bibliographiques

1 N. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems,* 1(1):7-38, 1998.

2 D. Georgakopoulos, M. Hornick, and A. Sheth. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Database*, 3:119-153, 1995.

3 http://www.aiim.org/wfmc/mainframe.htm.

4 http://www.omg.com.

5 B. Moulin and M. Brassard. A scenario-based design method and an environment for the development of multiagent systems. In *First Australian Workshop on Distributed Artificial Intelligence*, n. 1087, pages 216-231. D. Lukose, Zhang C. (edts.), Lecture Notes in Artificial Intelligence, Springer-Verlag, 1996.

6 J. Debenham. Constructing an intelligent multi-agent workflow system. *Lectures Notes in Computer Science,* 1502, 1998.

7 P. Rostal. Agent-oriented workflow: an experience report. *OOPSLA'99 Workshop on Business Objects*. 1999 (to appear).

8 M. Merz, B. Liberman, and W. Lamersdorf. Using mobile agents to support interorganizational workflow management. *International Journal on Applied Artificial Intelligence*, 11(6), 1997.

9 L. Yu and B. Schmid. A conceptual framework for agent oriented and role based workflow modeling.

10 H. Ludwig and Y. Hoffner. Contract-based cross-organisational workflows, the crossflow project. In *International Workshop on Workflows Coordination and Interoperability*. 1999.

11 N. Jennings, T. Norman, P. Faratin, P. O'Brien, and B. Odgers. Autonomous agents for business process management. *Journal of Applied Artificial Intelligence*. 1999 (to appear)

12 W. Schulze. Relationship of Business Objects and the Workflow Facility. Presentation to the Business Object Domain Task Force (BODTF), OMG Technical Meeting, Tampa, FL. OMG document bom/97-01-16. 1997.

Zakaria Maamar is currently a Scientist at the Defence Research Establishment Valcartier (DREV), Quebec. He has led research on software agent-oriented frameworks for interoperability issues. His current research interests are software agents and workflows. Zakaria received his M.Sc. and Ph.D. in computer sciences from Laval University, Quebec.
Contact
Information System Technology Section
Defence Research Establishment Valcartier
2459 Pie-XI Blvd North Val-Bélair QC G3J 1X5, Canada
zakaria.maamar@drev.dnd.ca

Nader Troudi works for Newtrade Information Technologies Inc., a software company in Montreal. His research interests are software agents for integrating distributed and heterogeneous information sources. Nader received his M.Sc. in computer sciences from Laval University, Quebec.
Contact
Computer Science Department
Laval University
Ste-Foy QC G1K 7P4, Canada
troudi@iad.ift.ulaval.ca

Pamela Rostal is a consultant on object-oriented and distributed technologies for Compuware Professional Services in Minneapolis, MN. She is currently pursuing her doctorate in Computer Information Systems from Nova Southeastern University, emphasizing the integration of workflow technology into the business environment.
Contact
Minneapolis Professional Services Division
Compuware Corporation
3600 West 80th Street Bloomington
Minnesota 55431, USA
prostal@acm.org