Preliminary Design of an Agent-based System for Human Collaboration in Chemical Incidents Response [*]

Mihnea Scafeş and Costin Bădică

University of Craiova, Software Engineering Department
Bvd.Decebal 107, Craiova, RO-200440, Romania

Abstract. In this paper we introduce initial design of an agent-based system to support human collaboration processes occurring in situations of chemical incidents response. The design process includes: (i) description of initial agent reasoning mechanisms using goals and plans; (ii) abstraction of external agent functionalities using tasks and services; (iii) organization of agents into communities. The design is illustrated by considering a realistic utilization scenario.

1 Introduction

The work reported in this paper was carried out in the context of a project interested in creation of a distributed information system to support population and environment protection against hazards produced by chemical incidents in urban and industrial areas: DIADEM – *Distributed information acquisition and decision-making for environmental management*[1]. This aim is achieved by employing heterogeneous techniques derived from various domains including software engineering, decision making, human-computer interaction, chemical engineering, and environmental and crisis management.

Management of efficient response to hazards produced by chemical incidents requires inter-organizational collaboration of multiple and possibly geographically distributed stakeholders, usually assisted by appropriate tools. Therefore, at the core of our proposed solution there is an agent-based service-oriented infrastructure that will support flexible and distributed collaboration processes between humans and agents engaged in arbitrary problem solving and reasoning tasks, as they are typically encountered in real situations dealing with chemical incidents response.

Our solution will be demonstrated by application on a number of representative real-world situations that are currently being captured as utilization scenarios. They will help us to identify, among other things, the various stakeholders, the tasks they are engaged with, together with their typical collaboration patterns. A key aspect of our proposed system is that it is agent-based. Basically this means that software agents are the "glue" that (i) will integrate all the active entities including active stakeholders, information sources, and support tools and (ii) will orchestrate them into coherent cooperative processes for efficient response to chemical incidents.

[*]Work of Mihnea Scafeş and Costin Bădică was supported by FP7 ICT-2007.6.3 DIADEM project, reference 224318.

[1]DIADEM Web site: `http://www.ist-diadem.eu/`

This paper is focused on introducing our initial proposal for capturing and agent-based modeling of collaborative processes, as they are required to support realistic utilization scenarios. Our approach is based on: (i) describing agent reasoning mechanisms using goals and plans; (ii) abstracting external agent functionalities using tasks and services; (iii) organizing agents into communities. The proposal is illustrated by considering the first utilization scenario inspired from the requirements document [2].

The paper is structured as follows. We start with presentation of the general modeling approach, followed by the discussion of the utilization scenario and associated stakeholders. The next section introduces the system goal model and defines agent behaviors using plans, tasks and services. We follow with agent communities and finally we conclude and point to future works.

2 General Approach and Utilization Scenario

General Approach. Special features of software agents, including autonomy and proactiveness, demand for special support during the development of agent-based software. This support goes beyond what is provided by traditional structured and/or object-object oriented development methodologies. Therefore, researches in the area of agent-oriented software engineering resulted in a number of proposals of methodologies to guide developers during the development of agent-based software systems [3].

In particular, problem instances (i.e. incidents) of our application domain (i.e. disaster management) are characterized as very complex situations that require fast and efficient response through flexible and dynamic collaboration between various actors including emergency units, local and/or regional authorities, human experts, a.o. Usually underlying business processes involve interactions that go beyond organizational and geographical boundaries and that are highly dynamic and unpredictable in nature, thus demanding flexible composition of problem solving and reasoning patterns that cannot be captured using traditional workflow technologies. In this context, sound analysis and design of support systems require application of principled agent-based methodologies.

Basically, an agent-oriented methodology defines the elements of the development process together with the associated products. In our work we did not strictly adhere to a particular methodology. Rather, we have adopted a more pragmatic approach by employing concepts inspired from Prometheus methodology [5] with the goal of producing an initial software prototype useful for checking and refining our solution.

In our approach we start with utilization scenarios that guide us in the identification of the agent types and system goals and sub-goals. We then follow with assignment of goals to agents and with definition of goals as plans composed of actions, sub-goals and tasks assigned to external agents.

In this section we introduce a sample utilization scenario and show how this scenario can guide the identification of agent types and their initial responsibilities. Nevertheless, our solution is general enough to be applicable to other, possibly more complex, scenarios. Also note that, due to paper restrictions, the utilization scenario has been simplified, while retaining most of the actions that we considered relevant for treatment of hazards produced by chemical incidents. This simplification resulted also in the order of some actions being changed, but this does not affect the relevance of this work for the purpose of the modelling illustrated in this paper. Finally note that we omitted the location names, because they are not relevant in the context of the paper.

Scenario Description. The utilization scenario is based on a real chemical incident that occurred in an urban area. The description captures the viewpoint of *DEMA Rescue Centre*[2]. Following requirements document [2], we provide in table 1 a simplified (but complete for the purpose of this paper) narrative description of the scenario.

Table 1. The utilization scenario.

”*A freight train crashes into a stationary passenger train. The engines of the two trains clash and goods wagons are scattered all over the tracks. A railway worker on the platform notices the incident and gives the alarm, calling someone to resolve the incident. The local fire department and the local police department engage in resolving the incident.*
The police commander arrives.
The fire service on-scene commander arrives and investigates the incident, discovering a leaking tank in the set of goods wagons.
The tank wagon is tipped over onto the platform and has a leak originating at the bottom. The out flowing liquid turns into an aerosol-gas mixture, leading to more investigation. The investigation reveals that the tank was supplied with orange plates on both sides. He requests information about the substance. He then finds out the substance involved is Chlorine and that a plastic based cover has to be used to prevent the chlorine from spreading. DEMA auto alarms itself.
The on-scene commander requests information about a safety distance, which is computed by DEMA with help from a weather agent that uses weather forecasting tool to predict the atmospheric conditions.
As the local fire brigade does not have access to a proper cover, the on-scene commander delegates the covering task to DEMA. DEMA dispatches its duty hazmat officer to the incident location.
The duty hazmat officer makes contact with the provider of the chlorine. The provider agrees to coordinate the work of getting hold of a tank to transfer the remaining chlorine into.
The first cars from the DEMA Rescue Center arrive. They start applying the plastic based cover over the chlorine.
The DEMA team has applied the plastic cover and stabilized the situation. The safety distance is reduced to less that 200 m.
The transfer tank arrives and the transfer of the remaining chlorine is started.”

Agents in the System. Analyzing the utilization scenario we were able to identify an initial list of stakeholders that are involved in the collaborative incident resolving process. Each stakeholder is mapped onto a software agent with a series of responsibilities. The list of these agents together with their brief description is illustrated in table 2.

Table 2. The list of agents for the utilization scenario.

Agent	**Description**
Worker	The railway worker that gives the alarm.
Fire Department	The fire brigade center.
Fire Fighter Commander	On-scene local fire brigade commander. Responsible for all coordinating on the scene of the incident. Not necessarily experienced with Hazmat incidents.
Police Department	The police center.
Police Commander	On-scene local police commander. Not necessarily experienced with Hazmat incidents.
DEMA Rescue Centre	Regional DEMA center that can provide manpower (conscripts trained in fire fighting and rescue) and specialized equipment for Hazmat incidents.
DEMA Duty Hazmat Officer	DEMA official responsible for guidance about Hazmat incidents.
DEMA Staff	Employees of DEMA.
Hazmat Owner	The company or governmental department owning the hazardous material involved in the incident.
Weather Agent	Agent equipped with weather prediction tools.

[2]Danish Emergency Management Agency – DEMA Rescue Centre, see `http://www.brs.dk/uk`.

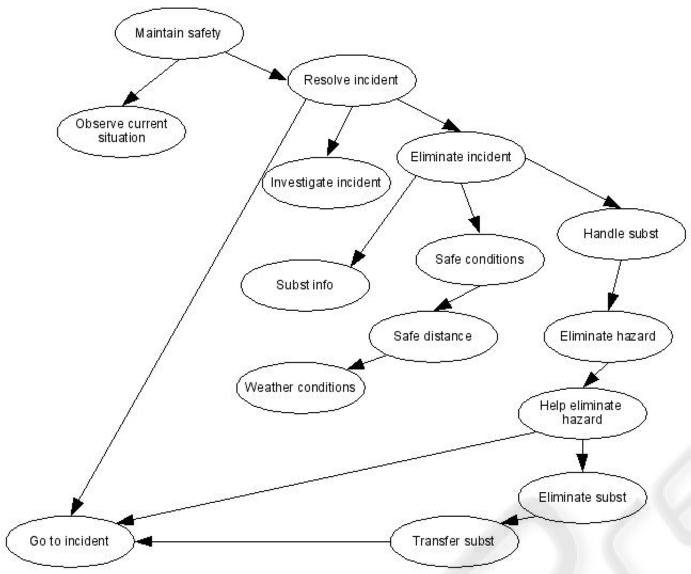

Fig. 1. Goal diagram for the utilization scenario.

3 Agent Model

In this section we develop an agent model using concepts of goals, plans, and beliefs.

Goal Model. The goal concept is commonly agreed in organizational and multi-agent modeling for describing system objectives. A goal characterizes a property of a desired state-of-affair that the system as a whole would like to achieve. An objective is usually conceptualized by a set of goals. Most agent methodologies include goals as primitive modeling elements. For example, Prometheus requires capturing of goals in the early stages of system specification by means of goal diagrams [5].

A goal diagram represents the system goals and how they are decomposed in sub-goals. The decomposition dependency between goals and sub-goals is represented by a directed acyclic graph. Each graph node represents a goal (or sub-goal) while the set of nodes that are adjacent out from a give node represent the associated set of sub-goals.

By examining the utilization scenario, we could identify an initial set of goals of the system. This set was further refined by decomposing each goal into a set of sub-goals. The decomposition process was iterated, finally producing a goal diagram that contains all the goals in the system together with their inter-dependencies (sub-goals). The resulted goal diagram for our utilization scenario is illustrated in figure 1.

Goals are assigned to agents in the system (see table 3). Note that this assignment process may introduce new goals or remove/merge existing goals. For example, *Fire Department* is an organization that usually contains many *Fire Fighter Commander* agents to which the task of resolving an incident may be delegated. We introduce a new goal in the system – *Help Resolve Incident* of *Fire Fighter Commander* agent to model this example. Generalizing, this situation may occur in goal delegation chains from manager to subordinate agents. The *Maintain safety* goal has been erased and each of the two descendant goals have been assigned to two agents. The *Worker* agent monitors the situation and whenever an incident emerges it triggers an alarm that will be propagated as a percept to all the agents. The *Fire Department* and the *Police Department* will handle this percept. Also the *Handle substance* goal has been refined.

Plans. Achievement of agent goals is realized using plans. A plan is a sequence of plan elements. In our approach a plan element can be a goal (actually a sub-goal of the goal to which the plan is assigned), an action or a task. A plan can optionally specify a contextual condition that must evaluate to true before the plan can be executed. Note that most practical reasoning approaches use plan libraries rather than planning from first principles from action descriptions [6].

Actions represent agent capabilities of changing the state of the environment. For example *Fire Fighter Commander* agent is able to mark the unsafe zone for people protection against chemical contamination. Tasks model situations when agents cannot successfully achieve goals by themselves and consequently they need to use capabilities of other agents and must ask these others to adopt their goals.

Note that the view of this goal adoption on the requester agent side is called *task*, while the view of goal adoption on the provider agent side is called *service*. An example is when *Fire Fighter Commander* agent needs more information about the chemical substance and asks *DEMA Rescue Centre* agent to provide this information via the *Provide Substance Information* service.

The distinction between tasks and services is important for at least two reasons: (i) we might have several agents in the system that provide the same service, i.e. that are able to realize the same task; (ii) we cannot always assume that a service provider agent will automatically adopt a task requested by a requester agent; for example the service agent might be overloaded or it might even consider that pursuing the required task is inappropriate given the current context.

In our plan models from table 3 a plan is denoted by: (i) a plan header containing the goal to which the plan is assigned, optionally qualified with a contextual condition written between square brackets, and (ii) a plan body consisting of a sequence of plan elements. Formally, the abstract syntax of a plan can be described as follows:

$$\begin{array}{ll}
\langle Plan\rangle & := \langle PlanHeader\rangle\langle PlanBody\rangle \\
\langle PlanHeader\rangle & := \langle Goal\rangle\langle Condition\rangle \\
\langle PlanBody\rangle & := {}' \rightarrow' \langle PlanElement\rangle\{' \rightarrow' \langle PlanElement\rangle\}* \\
\langle PlanElement\rangle & := \langle Goal\rangle \mid \langle Task\rangle \mid \langle Action\rangle
\end{array}$$

There are situations when agents must react to respond to significant changes in the current situation. These changes are captured by means of events that can trigger updates of the current plan and/or goal. In our utilization scenario we have initially identified a few situations when events trigger agents to adopt goals. For example, the event that the chemical substance has been covered triggers the adoption of the goal *Safe conditions* by *Fire Fighter Commander* which basically means recalculation of the safe distance (see table 3).

Note that resulted model already has all the necessary ingredients for devising the

Table 3. Agents goals and plans for the utilization scenario.

Agent	Plans	Events
Worker	Observe current situation [Bad situation] → raise alarm (action)	
Fire Department	Handle alarm [Alarm for incident] → Resolve incident (goal) Resolve incident[Alarm for incident] → Help resolve incident (task: Fire Fighter Commander)	alarm → Handle alarm
Fire Fighter Commander	Help resolve incident → Go to incident (goal) → Investigate incident (goal) → Eliminate incident (goal) Investigate incident → gather information (action) Eliminate incident [Chemical substance involved] → Substance info (task: DEMA Rescue Centre) → Safe conditions (goal) → Handle substance (goal) Eliminate incident [NO Chemical substance involved] → Safe conditions (goal) Safe conditions [Chemical substance involved] → Safe distance (task: DEMA Rescue Centre) → mark the unsafe zone (action) Safe conditions [NO Chemical substance involved] → compute safe distance (action) Handle substance [has access to proper cover] → do cover (action) Handle substance [does not have access to cover] → Handle hazard (task: DEMA Rescue Centre) Go to incident → move to incident location (action)	subst covered → Safe conditions subst transferred → Safe conditions
Police Department	Handle alarm [Alarm for incident] → Resolve incident (goal) Resolve incident → Help resolve incident (task: Police Commander)	alarm → Handle alarm
Police Commander	Help resolve incident → Go to incident (goal) Go to incident → move to incident location (action)	
DEMA Rescue Centre	Substance info [Substance identification features available] → identify substance (action) Safe distance [Chemical substance involved] → Weather conditions (task: Weather Agent) → compute safe distance (action) Handle hazard → Do cover the substance (task: DEMA Staff) → Eliminate hazard (goal) Eliminate hazard → Help eliminate hazard (task: DEMA Duty Hazmat Officer)	incident with dangerous substance → Eliminate hazard
DEMA Duty Hazmat Officer	Help eliminate hazard → Go to incident (goal) → Eliminate substance (goal) Go to incident → move to incident location (action) Eliminate substance → Transfer substance (task: Hazmat Owner)	
DEMA Staff	Do cover the substance → Go to incident (goal) → cover subst (action) Go to incident → move to incident location (action)	
Hazmat Owner	Transfer substance → Go to incident (goal) → transfer subst (action) Go to incident → move to incident location (action)	
Weather Agent	Weather conditions → Get weather conditions at location (action)	

agents' reasoning mechanism. In particular, the model has all the elements of belief-desire-intention framework [6], one of the most successful intelligent software agent architectures for practical applications. However, the reasoning mechanism is not yet finalized, being the subject of further investigation. Nevertheless, for the current prototype we chose a simple reasoning mechanism that follows the execution of plans associated to goals and was augmented with special event handling features.

Beliefs. Note that a lot of data is generated or gathered from various stakeholders, during the process of resolving an incident. One important aspect of the system is the management of this data. The system workflow is goal-oriented by executing plans and it assumes, among other things, reading external input data, generating external output data, and performing data transfers between agents. In particular, the data can also be mapped to beliefs ([6]) and stored at the agent level as agent beliefs (i.e. agent local data) and at the system level as system beliefs (system or global data).

Agent beliefs are queried whenever an action is executed, a plan condition is evaluated or information is passed to other agents in the process of contracting tasks. Agent beliefs can be updated when actions are executed or tasks are achieved by service contracting and information is returned to the task initiator agent. In particular, when the system is not capable of deriving a belief, the process of filling in the belief value can be delegated to human experts via specialized GUIs. Finally, agent beliefs can also be updated when percepts (i.e. changes in the environment) arrive. For example, when the *Fire Commander* receives an *alarm* percept he may add an *incident* belief to its belief repository. Percepts are usually generated when a change in the system beliefs occurs. An example of agent belief is an agent's current location.

System beliefs represent data that is useful to a group of agents. If this data is stored at the agent level and passed between agents rather than being kept as system beliefs then the process would be time and resource consuming. An example of system belief is the location of an incident.

4 Agent Communities

Further analysis of the utilization scenario reveals that it is useful to conceptualize the agent society as logically decomposed into communities on the basis of organizational or functional criteria. In this context, the logical partitioning of agents into communities is particularly useful for focusing agent interactions, for defining special relationships (functional, social, or managerial) between agents, and also for partitioning incident information into local (i.e. agent level) or global (i.e. community level) beliefs and context information. For example, the type of organizational relationship, i.e. peer-to-peer or subordination, can influence the degree of openness of allocation of tasks to services [4]. Note also that the decomposition of a society into communities matches well with the requirements of inter-organizational collaboration for incident response.

Following the description of the utilization scenario we identified the following communities: *Fire*, *Police*, *DEMA*, *Weather*, and *Incident*. The first three communities are defined by organizational criteria, and thus they exhibit a specific organizational structure. In contrast, the *Incident* community is defined by functional criteria and contains agents that collaborate in resolving a specific incident. That is why this community can be characterized as dynamic or ad-hoc. Finally the *Weather* community incorporates agents equipped with appropriate tools for weather prediction.
We are using Prometheus system overview diagram [5] for describing system architecture. This diagram is structured as a graph showing agents, events, actions, and messages as nodes represented in Prometheus notation. Note that we augmented the system overview with representation of agent communities (see figure 2). Large ellipses repre-

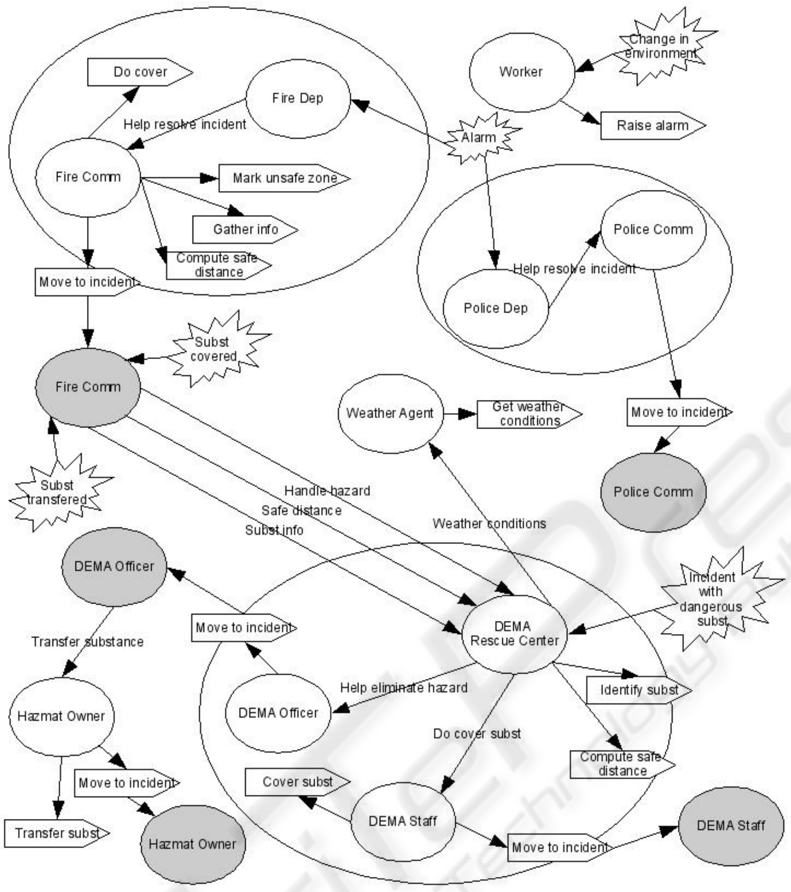

Fig. 2. System overview diagram.

sent communities. *Incident* community contains all the agents that are outside the other communities. Smaller ellipses are agents. Shaded smaller ellipses are agents after they have joined another community. Actions are represented using pentagons and events are represented using "explosions". Tasks that are contracted are depicted using associations between agents.

On this diagram we can observe that some agents are subordinated to other agents. For example, *Police Commander* responds to *Police Department*, *Fire Fighter Commander* responds to *Fire Department*, and both *DEMA Duty Officer* and *DEMA Staff* respond to *DEMA Rescue Centre*. Using agent communities (defined based on organizational relationships in this particular case) manager agents can isolate their subordinates from the rest of society and thus focus communication only on them when necessary. This approach is needed for example when *Fire Department* searches for someone to help resolve the incident. If the subordinates of the *Police Department* would not have been isolated from the rest of society then *Fire Department* would be able to find a *Po-*

lice Commander (as *Police Commander* advertises a *Help resolve incident* service too) and dispatch him to the incident location that is obviously incorrect behavior.

The ad-hoc *Incident* community is of particular importance. This community is dynamically constructed by joining in all agents that are actively involved in resolving a particular chemical incident. By default, *Incident* community initially contains the following agents that represent regional authorities in the incident region: *Police Department*, *Fire Department* and *DEMA Rescue Centre*.

Focusing communication in a community may be accomplished by community oriented service discovery. This requires agents to advertise their services and search for provided services in the context of communities they are members of, using community oriented *Yellow Page* agents. In particular, when an ad-hoc community is created, a new community *Yellow Page* agent will be also created and added to that community. All agents that will later join the community are required to register their services at the community's *Yellow Page* agent. Moreover, when an agent requires an external service, he will have to search for it using *Yellow Page* agents of his communities – note that an agent can be simultaneously member of several communities. For example, on figure 2, when *Fire Commander* agent performs action *Move to incident*, additionally to the action effect of changing the agent location, *Fire Commander* also dynamically joins the *Incident* community, and so, using the *Yellow Pages* mechanism, he will be able to directly talk to *DEMA Rescue Centre* during the process of resolving the incident.

5 Related Work

ADEPT – *Advanced Decision Environment for Process Tasks* is probably one of the first business process management systems that proposed the utilization of intelligent software agents to cope with inter-organizational collaborative aspects of business processes including: multiple and geographically distributed organizations, autonomous management of resources, highly dynamic and unpredictable nature of business processes, decentralized control, mixtures of human activities and automated tasks [4]. ADEPT introduced many concepts that we found useful including loosely coupling of agent tasks and services by means of service matchmaking and negotiation and usage of the notion of agency (with peer-to-peer and hierarchical relationships in organizations) that we found similar with our communities. However ADEPT was an early work and could not benefit on the recent developments including principled approaches of agent-oriented methodologies and technological advancements in software agent platforms. Additionally we did not find in ADEPT the concept of ad-hoc community that we introduced to model teams of agents that act together towards resolving a given incident.

A recent solution for providing agility to goal-oriented business processes using BDI agents based on LS/TS – *Living Systems Technology Suite* is presented in [1] (and papers cited there). While this approach seems to have many similarities to our proposal (excepting maybe service negotiation which is present in our work), we could not find any details of the algorithms and implementation of the suite – only very general descriptions and short application reviews are provided.

Combined Systems – *Chaotic Open world Multi-agent Based Intelligently NEt-*

worked Decision support Systems[3] project was focused on engineering large-scale, open systems in a constantly changing environment [7]. The research was validated on the *Rotterdam Harbor Scenario* that describes a crisis situation when a ship collision produces a spread of toxic gas in the harbor and threatens surrounding areas. Focus in Combined Systems was on development of technologies known as intelligent building blocks for communication, interoperability, iconic information sharing, information fusion, and coordination. However, we noticed that, while Combined Systems was in fact a multi-agent system, it did not employed agent-based development methodologies.

IsyCri – *Interopérabilité des Systèmes en Situation de Crise*[4] project is focused on inter-organizational inter-operability for crisis reduction in order to assure high degrees of responsiveness and flexibility [8]. Specific to IsyCri is the focus on realizing enterprize inter-operability using a meta-modeling approach based on ontologies. This is different from DIADEM, where we are seeking for the development of a practical system to support decision-makers in crisis situations. Nevertheless, we think that IsyCri results, including the crisis meta-model [8], could be useful for devising service ontologies that are subject of further investigation during DIADEM project.

6 Conclusions and Future Work

We have shown how starting from an initial utilization scenario, agent-oriented methodologies (in particular Prometheus) can help us in guiding the systematic development of an agent-based model of collaborative processes for chemical incidents responses. In particular we intend (i) to expand our model with service ontologies and service negotiation; (ii) to investigate potential problems regarding complexity and scalability of the approach; and (iii) to extend the design by considering other utilization scenarios. Currently an initial prototype has been developed based on the initial design described in this paper. We shall report on its evaluation and improvement in subsequent papers.

References

1. Burmeister, B., Arnold, M., Copaciu, F., Rimassa, G.: BDI-agents for agile goal-oriented business processes. In: *Proceedings of the Seventh Conference on Autonomous Agents and Multiagent Systems (AAMAS), industry track.*, 37–44. IFAAMAS (2008).
2. Damen, D., Pavlin, G., Van Der Kooij, C., Bădică, C., Comes, T., Lilienthal, A., Fontaine, B., Schou-Jensen, L., and Jensen, J.S.: DIADEM Environmental Management Requirements Document, Issue 1.12.0 (2009) (work in-progress).
3. Henderson-Sellers, B. and Giorgini, P. *Agent-oriented Methodologies*, Idea Group Publishing (2005).
4. Jennings, N.R., Faratin, P., Norman, T.J., O'Brien, P., and Odgers, B.: Autonomous Agents For Business Process Management. In: *Applied Artificial Intelligence*, 14, 145-189, Taylor & Francis (2000).
5. Padgham, L. and Winikoff, M.: Developing Intelligent Agent Systems, Wiley (2004).
6. Rao, A.S. and Georgeff, M.P.: BDI-agents: from theory to practice. In: *Proceedings of the First Intl. Conference on Multiagent Systems – ICMAS-95*, San Francisco, USA, 312-319 (1995).

[3]See Combined Systems project Web page at `http://combined.decis.nl/`

[4]See IsyCri project Web page at `http://www.irit.fr/IsyCRI`.

7. Storms, P.: Combined Systems A System of Systems Architecture. In: *Proceedings of the First International Workshop on Information Systems for Crisis Response and Management, ISCRAM'2004*, Brussels, 139-144 (2004).
8. Truptil, S., Bnaben, F., Couget, P., Lauras, M., Chapurlat, V., and Pingaud, H.: Interoperability of Information Systems in Crisis Management: Crisis Modeling and Metamodeling. In: Mertins, K., Ruggaber, R., Popplewell, K., and Xu, X. (eds.): *Enterprise Interoperability III. New Challenges and Industrial Approaches*, Springer, 583-594 (2008).