AI & Soc (2009) 24:25–34
DOI 10.1007/s00146-009-0193-6

ORIGINAL ARTICLE

Escape and intervention in multi-agent systems

G. B. Roest · N. B. Szirbik

Received: 15 January 2008 / Accepted: 19 December 2008 / Published online: 19 February 2009

Abstract This paper describes the escape/intervention concept as it is used in the agent growing environment framework. The Escape and Intervention is used in many multi-disciplinary areas, including agent research, artificial intelligence, groupware and workflow, process support, software engineering, and social sciences. Based on an ontological perspective, this paper explains how an interaction-oriented agent architecture and language (used for modelling, simulation, and development) makes use of an interaction pattern that is inspired from social contexts seen as multi-agent systems.

1 Introduction

Human involvement in the execution of a computer-based system gained importance since the systems became interactive and systems themselves started to be designed to interact with each other. Systemic thinking, especially the one dedicated to loosely coupled, distributed systems has investigated ways to allow humans to remain in control, given the increasing levels of automation and non-human decision making (Chapanis 1996).

This paper discusses a novel way to reason about, model, simulate, and implement agent-based systems, a way based on the *escape/intervention* concept. The concept has been introduced previously (Roest and Szirbik 2006), but from the perspective of agent-oriented software engineering only. A series of published papers present various aspects of this approach, like the focus on interaction (Stuit and Szirbik 2006), the concept of local behaviour (or interaction belief, (Stuit et al. 2007), and behaviour alignment (Meyer and Szirbik 2007a). Since the escape/intervention is inspired from social sciences and organisational theory, it is necessary to explain the concept from a perspective that is anchored in a social context (Sierhuis et al. 2003). Some researchers, like Ekdahl (2000), emphasize that this social context is crucial for any development related to agent methodologies. Agents are a metaphor inspired from social reality, and their defining characteristics like autonomy, empowerment, high-level language and communication skills, negotiating ability, argumentation of beliefs, trust evaluation, and the ability to reason about organizational knowledge, are intrinsically social.

The presentation is based on an ontology that answers question like the following: "What is an agent with respect to escape/intervention?", "What are the main concepts that are used in conjunction with the agent concept?", "How the agents manage to work together?" The ontological commitments are important to any agent research, and it is very useful to define a semi-formal ontology on which the research framework is built (one may say: the ontology is the framework). The basic concepts of this framework (which is implemented as the AGE toolset—"Agent Growing Environment", AGE), like agent, interaction, role, behaviour, alignment, growth—iterative development, are introduced via ontological definitions. The central concept of escape/intervention is logically related to these basic concepts.

G. B. Roest (✉) · N. B. Szirbik
The Agent Lab, Department of Business and ICT,
Faculty of Economics and Business, University of Groningen,
Landleven 5, P.O. Box 800, 9700 AV Groningen,
The Netherlands
e-mail: G.B.Roest@rug.nl

N. B. Szirbik
e-mail: N.B.Szirbik@rug.nl

The research presented here involves modelling, simulation and the development of agents that support *interactional processes*, that is, the process itself can be decomposed in a set of interactions between agents. These interactions can be regulated exchanges of information but also can be more informal interactions like dialogues and meetings. The community of Social Intelligence Design emphasises the importance of the interaction concept. In the work of Fruchter it is argued (Fruchter 2001) that any new collaboration technology will require the rethinking of "interactions among people in terms of the individual's behaviour, interaction dynamics…, protocols, collaboration processes…, interactivity with the content of interest". Other works investigate various aspects related to the interaction concept like virtual representations of physical reality (gestures, body language) in (Nijholt et al. 2006), and also categorize the interactional processes in terms of the nature of the interaction space (Rosenberg et al. 2005).

This paper is organized as follows: Sect. 2 presents the multidimensional spectrum where this research can be positioned and introduces the specific approach in Sect. 3 by taking into account similarities and differences with other approaches. Section 4 discusses the ontological commitments, defining an agent, an interaction, a belief, and behaviour. Section 5 explains in detail the escape/intervention concept and presents how it can be used to incrementally enrich the behaviour of the participating agents. The discussion in Sects. 6 and 7 concludes the paper and outlines some immediate issues for future research.

2 Support paradigms for organisational processes that are based on social interaction

The support (via software components) for the class of processes that exhibit high levels of social interaction can be categorized on a multidimensional spectrum defined by the degree of global process explicitness, visibility of the process description, flexibility, and whether the approach has a centralistic view. The following subsections compare three main R&D approaches. The class of processes under investigation is called interactional processes in the following text.

2.1 Business process execution and management support

On one extreme border of the multidimensional spectrum, both the interactional process *structure* and *dynamics* are specified in a central point, and all participants are playing their roles according to these descriptions. This is the typical way *workflow enactment services* are designed. For these services, there is a strong emphasis on designing the formal roles involved, the role-related protocols and some of the exception procedures. The process structure is developed during the early phases of the development, the behaviour of the participants is strongly regulated, and preferably all the known exceptions have to be captured *before* the system is released for use. The main advantage here is the clarity and visibility of the overall organizational behaviour. In certain types of organizations (e.g. insurance companies, financial services), it brings discipline and good quality of service. The disadvantages stem from the rigidity of the enacted system and from the inherent difficulty to model activities that are based on social interaction like peer-to-peer dialogues, negotiations, and multiple participant meetings. Moreover, any local change in behaviour should immediately be made visible to the central point of view, and any change of the global process description will have its immediate impact on every participant. This makes dynamic changes very difficult. If changes are inevitable and occur often, one solution is to maintain multiple versions of the process model for different instances of the process, but this leads very often to errors (van der Aalst 2004) and in some cases is impossible when new regulations require new process definitions. Current research investigates the *decentralisation* of the business process definition, bringing the field closer to the agent paradigm (Norta 2004).

2.2 Groupware support

Groupware tools (or Computer Supported Cooperative Work (CSCW)—technology) help people to collaborate by allowing them to send information to each other in a structured manner (Ellis 2000). Since these tools focus more on information sharing they are not 'process aware' meaning they cannot control and execute a process in an semi-automated way unless certain extra (workflow-like) components are added. Overall, existing solutions do not support unstructured processes in which different (physically or logically) distributed participants collaborate via social interactions (van der Aalst 2007). The process is executed exclusively by the actions of the participants, who react and know what to do when faced with certain information. Albeit CSCW tools enable meetings, negotiations, and informal dialogues, the structure of each process instance (assuming that there is always a certain process unravelling) is determined mostly by the decisions and actions of the participants. The reason why it is difficult to transform a CSCW tool into a Business Process Enactment tool is that every process instance is very different from the previous ones (Ellis et al. 1991). With experience, some generic activities can be identified and routing constraints (like doing certain actions in parallel or doing them in a

given sequence) can be formalised within the scope of the CSCW toolset.

The CSCW approach can be placed in the spectrum on a position where the process model is not visible (or explicitly built), and there is no central component that orchestrates the actions of the participants, but flexibility is very high.

2.3 Completely decentralised agent oriented support

Situated in another side of the spectrum is the support for complex interactional processes via an agent approach. Each agent performs its own activities within the process, but it is also able to interact with other agents. The agent paradigm states that there are no central points of control or global representations of the process structure available to the agents (Jennings et al. 1998). The agent's beliefs should be locally tuned in order to obtain a global behaviour that is actually the desired process behaviour. Although simulations point out that this is sometimes possible (Wooldridge 2002), it is extremely difficult when the process is complex and heavily interaction-oriented and when the agents do not have extensive beliefs about the intricacies of the organizational structure. This is happening in reality when the human agents in an organisation do not have enough experience in the organisation (they are "new"). The main advantage of the agent approach is that it is easier to model dialogue and unstructured information exchange, compared to the business process approach.

However, advocates of the agent paradigm insist that the overall behaviour of the agent community should emerge in the desired process execution, and this is happening only if the agents themselves know exactly what to do in any circumstance, they are learning, and they are able to predict what the other agents will do when interacting.

3 The AGE approach

Considering that the business process enactment and agent approaches are situated at two extreme sides of the multi-dimensional spectrum, this approach can be positioned in the middle. This brings together conceptual frameworks from both agent and workflow methods. In a sense, one can say that groupware software tends to do the same, but a lack of explicit agent representation devoids this approach from the clarity it could have.

The AGE enables the modelling and simulation of complex processes consisting of social interaction-based activities. AGE is also a development tool for multi-agent systems (MAS) that are deployed to support the modelled and simulated processes. Based on a concept taken from the workflow-oriented approach, AGE allows a set of interacting agents where each has a partial view of the whole process. The main difference with the workflow approach is that these descriptions are merely considered as agents' beliefs that are allowed to mismatch and contradict.

The agent-oriented approach gives meaning to the concept of locality (as opposed to centralisation), and expands concepts that are ontologically defined, as *belief*, *autonomy* (Ekdahl 2000), (legal) *ownership*, and *responsibility*. Other features that come with the use of agents are dependability and robustness of the overall system. A feature (which is more common for Artificial Neural Network-based systems) that ensures that an increasing level of 'noise' will slowly degrade the performance of the system is called *graceful degradation*. It means that if more exceptions and unforeseen contexts occur, the number of necessary external interventions by users will not increase steeply at one or more points. Graceful degradation means the system will not collapse and halt due to a single-point-failure, but will just gradually decrease performance.

3.1 Roles and interaction beliefs

Taking an internal viewpoint, the main difference between the AGE-based conceptual architecture and the existing ones is the structured, procedural kind of belief of the agents that is called *behaviour*. In structural terms, these are actually workflows, capturing the way an agent believes he has to interact with other agents, all from a local perspective. The agent "carrying" such an interaction belief knows what he is supposed to do in an interaction and has expectations of what the other agents are doing. In other words, the agent has an acquaintance model about the behaviour of the agents it is interacting with. The interpretation that is built with these interaction beliefs is the foundation for plans, which are generally and necessarily vague to accommodate inconsistencies (Suchman 1987) that will surely occur in dynamic environments with distributed knowledge.

From an external viewpoint, roles are used (widely used in other methodologies, and put in this agent perspective by (Stuit and Szirbik 2007)), to augment the interaction descriptions and to act as placeholders for the agents that may participate. Both role and interaction are used as building blocks for the model that a particular agent has about its environment.

Roles are also used for internal representations, namely in the behavioural descriptions. In Fig. 1, it is shown how a "sales manager" agent believes that he will interact with a shop floor scheduler and a tactical planner (within the same organisation). In the AGE-related modelling language (named TALL), such a construct is called an "interaction belief" and shows the behaviours of the three roles involved in the interaction. The activities of each role are

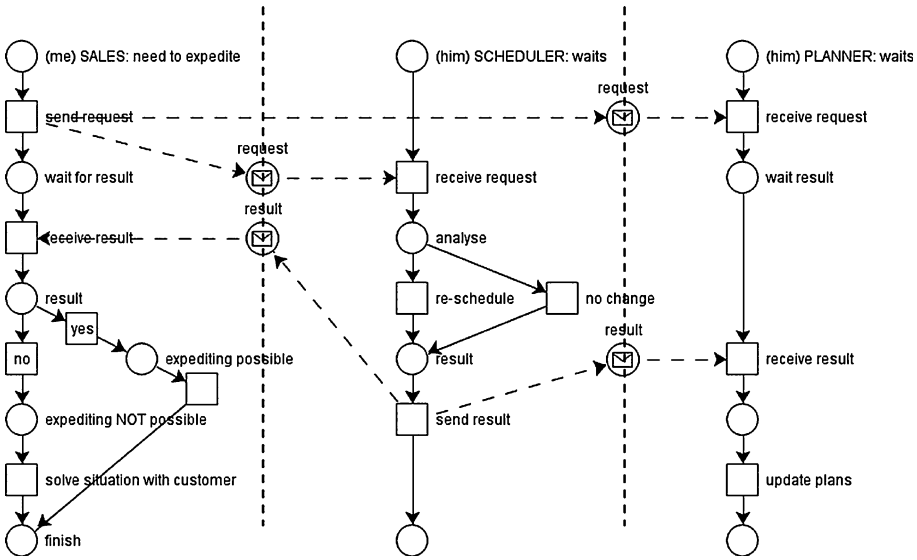

Fig. 1 An example of an interaction belief

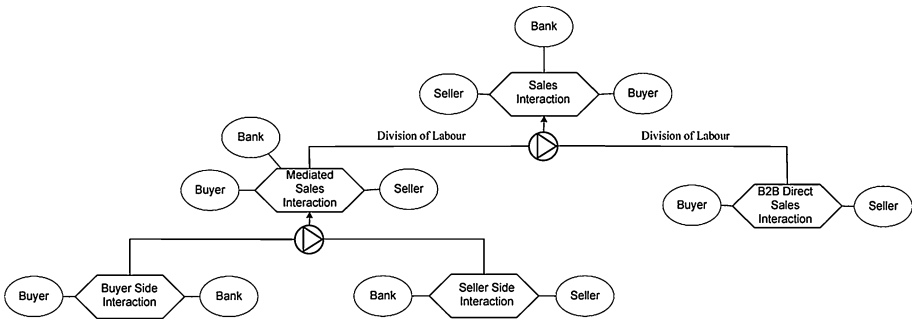

Fig. 2 An interaction composition diagram in TALL

described in a separate swimlane, the first one being the "me" swimlane, indicating that this particular belief is owned by the sales manager agent. He intends to execute the actions on his swimlane as regulated by the Petri Net description (van der Aalst 2004), and expects that the other two participants, playing the other roles, will perform their associated actions. However, it is possible and permitted in AGE that different agents can have interaction beliefs that are not aligned to each other. Alignment can be enforced by the users of the system, can be figured out by the developer, or can in some simple cases even be automatically achieved by software agents (Meyer and Szirbik 2007b). Vagueness is achieved by leaving out details of the behaviours of the other swimlanes—meaning that "me" is not completely aware of how others should do their part in the interaction.

3.2 The modelling language used in AGE

The TALL (Stuit and Szirbik 2007) language is able to capture the structure and the basic building blocks of an organization that runs via social interaction-based processes by having special symbols for agents, roles, interactions and behaviours. TALL is a graphical modelling language (as seen in Figs. 1 and 2) and borrows from workflow specific languages, especially Petri Nets. The language has multiple purposes in the context of this research. First, it is used for organizational modelling, helping the stakeholders to understand their organization and eventually change the models, and based on this the organizational processes and structure.

Second, the language is also used to build the simulation models, having precise denotational and operational semantics. What makes this language different from almost any other agent-oriented description language is its focus on agent-to-agent interaction. Prior to this approach, interactions as explicit modelling symbols (represented as elongated hexagons in Fig. 2) appeared only in the MESSAGE approach (another modelling language framework (Caire et al. 2001) and less explicitly in AORml (Wagner 2003) where chains of interactions can describe a process or a workflow in a diagram. The problem with these previous approaches is that the view is external to the agents and therefore centralistic. The agents have to obey in any case these external descriptions about how they should interact. TALL allows for composition and decomposition

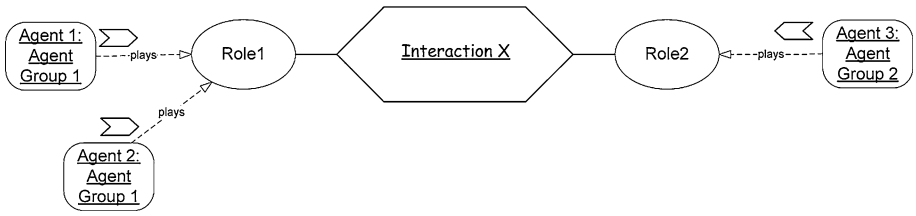

Fig. 3 The model of an instance of an interaction

of interactions. These structures are sets of cascading interactions, which ultimately represent social interaction processes and can have local representations in each (set of) agent(s)—like in Fig. 2.

The simulations in AGE are not closed experiments. These can be regarded as interactive games. Players can interact with the simulated agents and change their behaviour if necessary—that is, they can redraw interaction belief diagrams. The interaction composition models help the system to trigger top-down or bottom up other interactions, allowing more agents to take part in the simulation run.

In Fig. 2, a generic sales interaction is modelled as a tree of sub-interactions. This is also a partial belief of an agent about how a process can be reduced to a topological set of related interactions. In Fig. 3, an instance of an interaction is figured. This model can show a post-mortem of what happened, but also can be a snapshot of the run-time situation at a given moment. The difference with the diagram in Fig. 2 is that the agents that carry out the interaction are already allocated to their respective roles.

4 Ontological commitments and definitions

The AGE ontology is centred around two primary concepts: something that *is performed*, (the social interaction) something that *is,* (the agents) and two secondary terms that help associate them in a process (role and behaviour). They are related to each other in the following sentence: *Agents play roles and perform behaviours to participate in interactions*. In the following subsections, the TALL agent and role concepts are thoroughly detailed.

4.1 Agent and role

The link between business process (or workflow) enactment and agent systems comes down to the introduction of dynamic role assignment. The occupancy of a role in a running interaction by an agent, gives a sort of dynamic identity to that agent identifiable by the agent in the name defined by itself in the behaviour it is exhibiting during the interaction. Figure 1 shows how the name of the self-role appears as a label of the swimlane marked "me".

There are three agent meta-types in AGE (0) that provide information about the agent's characteristics:

4.1.1 Human agent

At the conceptual level, these are representing the real humans. In the simulation, these appear as simulated humans (and at the simulation level can be regarded as mere software components), but they are also interfaces between external human players and the rest of the simulation. They act as representatives of the external humans and relay messages from and to other agents. We consider human agents to be atomic entities with a physical presence.

4.1.2 Institutional/organizational/synthetic agent

At the conceptual and modelling levels, these are abstractions that provide an interface within groups of agents. Synthetic agents typically do not have a responsible human attached, but a blackboard mechanism, which is monitored by an (human) agent or external human. Synthetic agents are composed from other agents and are highly artificial and abstract in this sense they represent an organization or a social system. However, during simulation, these agents are represented as software components and can play roles and perform behaviours. The sum of all behaviours of the agents in the represented organization is not necessarily equal to the total set of behaviours available to the real institutional/organizational agent. It is possible to represent and enact in simulation a synthetic agent that exhibits the behaviour of an organisation but which has no "internal agents" yet. These agents (atomic or synthetic) can be added later.

4.1.3 Software agent

These are autonomous programs that have been delegated with some of the decision-making powers of a human agent. This can be done by formally identifying some of the behaviour of the human and these representations have been "coded" in the software agent. It is important to note that the responsibility for any action of a software agent can always be traced back to a human or organization that owns the software agent (both in simulations and after deployment). Software agents are atomic, and are always owned by a human or an organization.

4.2 Interaction

An interaction during AGE simulation sessions is initiated by one of the agents when it performs a behaviour (which is running due to another, previously started interaction). The system starts the interaction and assigns the agent to the appropriate role in the interaction. The agent may or may not know the role name as known by the overall system (even the agent has an internal label for it—and this one can be different). If the role name does not exist yet in the system (the interaction is incompletely defined), the system is entering that state we call Escape-Mode and a new role is added to this interaction by the experimenter—after an eventual consultation with the players. Alternatively, a new role can be created automatically, and the players will attach those swimlanes of their behaviours that are appropriate to this role, aligning also the names they use for the role to a unique label, selected by the most "powerful" agent present in the interaction who can be the experimenter. Normally, in other approaches, the role-names are considered global knowledge because they are linked to an organizational structure (or in other words: agents share the same ontology and problem space).

When the agent is starting to play a role in the interaction, the rest of the roles have to be assigned to other agents in the system. This can be done automatically, or manually. These agents will use their existing behaviours. It is desirable that all the behaviours match. If it is not the case, the system will resort to Escape-Mode again, and the alignment can be achieved manually by the players with the help of the experimenter, or automatically, if alignment mechanism have been implemented. For some cases of simple behaviours and light cases of mismatch, automatic procedures for alignment have been already implemented (Meyer and Szirbik 2007a, b).

During gaming/simulation sessions, some interactions and portions of the process can be carried outside the system. The role of the experimenter in this case is to log this external activity and try to formalize it in more behaviours and interaction descriptions. These newly defined parts of the process can be added to the next simulation sessions.

In AGE, interactions are facilitated by a service of the system acting as a medium on the behalf of the experimenter. It is important to note that everything starts with an interaction. All processes are interaction-driven. Interaction diagrams like in Fig. 3 are also the graphical way in AGE to describe and visualize the evolution and state of the simulated process.

4.3 Behaviour

The third core element in the proposed ontology is the interaction behaviour. It can be seen as a localized belief of an agent, or (part of) an interaction belief—like a swimlane in Fig. 1. The behaviour implies action and it is represented as a structured set of activities and states represented as a Petri Net. An agent can have an exact view only about its own activities, and these are structured along a swimlane that is tagged "me". When performing this behaviour the agent is uncertain about the way the other agents are performing the roles involved in the interaction. In the proposed framework, it is preferable when agents' behaviours contain acquaintance models of behaviours for other roles (they are interacting with).

Two extremes for interaction execution descriptions can be identified during modelling and simulation. On one end, there are the *protocols* or organizational behaviours, which are role-bound and part of a domain's global beliefs. Nevertheless, they remain beliefs if the domain is considered part of a larger encompassing domain. On the other end are the behaviours that are still unspecified, and have to be discovered. This discovery is usually done with the help of an expert in the performance of the tasks related to this behaviour. During AGE gaming/simulation sessions, the external players, with the help of the experimenter, guide their associated simulated agent through the interaction that is unfolding. In MAS use, this happens when the user has to improvise an ad-hoc behaviour in order to finish the interaction where he plays a role, which happens when the agent does not know what to do and resorts to its superior, i.e. escaping. In this way, the superior is instructing the agent how to perform the tasks this agent has not been able to execute.

Initially, in the simulation model, there are no software agents, only simulated humans (which in fact, *are* software components). After using improvised and/or well-known organizational behaviours, these behaviours can be reused by the agents in future interactions even if they have been only partially described. If repeated improvised behaviours (which have to be logged) will emerge as patterns, these are candidates for behaviours that are represented in the simulated human and synthetic agents. These ones will be continuously improved in order to be fully usable by (simulated) software agents that have been "split" from their simulated human agents. When stakeholders agree, these behaviours can be imposed as protocols and linked to roles/interaction (and not to particular agents). With that, the agents are incrementally "grown" by adding new behaviours to them, and some interactions are also "protocolised". It is important to note that in the case of protocol ruled interactions, the agents can still overrule the execution protocol in exceptional contexts and execute the interaction in their own way.

5 The escape-mode and intervention detailed

This concept has three components. The first describes the state the agent enters when searching for help. This is called the Escape-Mode. The second describes the entity that helps and intervenes, which we call the *Deus ex Machina*, and the third describes how the action taken by the Deus ex Machina, in the form of Intervention, is implemented. An agent reaching for something outside of its system is escaping (see Sect. 5.1). Something that changes what is inside a system without being part of it is intervening via the intervention mechanism (see Sect. 5.2). The definition of the Deus ex Machina is given in Sect. 5.2 with the emphasis on its role during simulation and the most important features of the AGE framework are presented in Sects. 5.3 and 5.4.

5.1 The escape-mode

As stated before, an agent in AGE can go into Escape-Mode if it recognizes a situation that demands intervention: when it is not possible to infer automatically which role the agent has to play in an interaction, or when alignment of behaviours has to be done manually. Escape-Mode is defined as the state an agent enters when it fails to grasp the current situation it finds itself in, and needs intervention from something outside its domain. The Escape-Mode is also used when an agent in a game, as a representative of a participant in a process, is confronted with a situation that it is not empowered to take a decision and act in such context, or when the agent is programmed a-priori to go in Escape-Mode on exceptional situations. The most obvious situation is where the agent has a default behaviour to respond to an unknown situation. A concept very close to Escape-Mode is the workflow-management (WfM) and case management (CM) concept of *escalation* (Anonymous 1999), also encountered in some groupware applications. Escalation is triggered when a deadline or time limit is exceeded for a work item, and immediate action is required. In CM, when escalation occurs, it functions as a notification mechanism contacting a number of people assigned to the respective business process case. In WfM, escalation corresponds to the event where the workflow instance/case (or part of) does not end in time. An external process is then started and takes the necessary steps to solve the situation. Often it can be resumed by a mere allocation of additional resources.

Escape-Mode is the part of the mechanism that identifies the patterns of the problems in a situation (but does not solve them). Entering Escape-Mode means that the agent lets someone higher in his internal organization take over. The agent's behaviour becomes guided or fully conducted by a superior (expert) game player, experimenter, or the user after MAS deployment. To describe the nature of the superior with respect to Escape-Mode we use a term from the antique Greek theatre.

5.2 Deus ex machina and intervention

The notion of superiority is phrased by the question: "Who is in charge?" Antique theatre gives us a concept, something that can solve all of the present problems and fixes the story, and in the case of AGE the flow of the process. According to its definition in the Encyclopaedia Britannica the Latin phrase *deus ex machina* has been extended to refer to any resolution to a story which does not pay due regard to the story's internal logic. The resolution is so unlikely it challenges suspension of disbelief, and presumably allows the experimenter to end it in the way he or she wanted. In AGE, the Deus ex Machina is played by the experimenter or players who take care of the situations that have not been modelled. They can dynamically change (adapt) the behaviour of the agents. It is possible that even the agents are not able to cope because they do not have the experience or domain knowledge. Hence, they relay the exceptions to their own respective Deus ex Machina, who is presumably an external expert player outside the current session or an organizational agent. It is not absolutely necessarily that the Deus ex Machina is a human agent. It can be a (exceptionally intelligent) software agent who can provide the necessary support, this agent being always owned by an individual human or an organization.

Before setting up AGE simulation sessions, some basic initial beliefs of the agents are set as an incomplete result of preliminary analysis. Still not present in the system are the patterns of behaviour that will emerge, the complete hierarchy of roles and the nature of interactions that will be identified later. During this first session, the Dei ex Machinae (at different levels) will decide which behaviours have to be carried out by themselves and which will be delegated to other agents.

Experimenters control the simulation and have the constant awareness of what is going on in the system. The experimenter's scope is broader and deeper than that of the agents. An agent, giving a notification of an exception of some sort always knows there is someone watching over it that can help in this situation. In its own ontology, it can be defined as Superior, Employer, Owner, Parent, etc. Experimenters are able to use the logged behaviour of the multiple Dei ex Machinae to enrich the behaviour of the agents, change interactions, in short, re-model a small part of the simulation during the AGE sessions.

Intervention may occur without an escape trigger. On their decision, the Dei ex Machinae can intervene when they realise that the process unfolds in a way that it is not desired by them.

5.3 Growth

The concept of Agency lets us define agents at various abstraction levels. From concept to deployment, the agent-concept is untouched, but the physical form changes drastically. One of the problems we address here is that the almost natural form of the agent at the conceptual level is completely different from the software program at the implementation level. At the conceptual level, we assume the agent to be able to grow (in knowledge e.g. beliefs) but at the same time we demand the agent to be part of a formal (rigid) system (e.g. software). One of the AGE framework's characteristics is to solve this.

Software programs running in a system are mere collections of algorithms and abide to the rules defined for Turing-machines. Due to these rules, the notion of growth poses a problem, because software programs cannot change their internal structure (add or change internal states) without violating the bounds of the system (if a virus-scanner is an agent, then the operating system would be its system). One expects that clever design and implementation of the agent will not circumvent these limitations, because agents remain Turing-machines. In that respect, our definition of the trigger to enter Escape-Mode is the same as the event where a Turing-machine enters a state, which has no transition rules. Yet, since our primary objective is to define an agent architecture which allows agents to grow (i.e. have the potential to enter an 'infinite' set of states), we have to find a way out of the system. An agent with the Escape-Mode mechanism has a default transition available to it from every possible state, always and anytime. This transition allows the halting Turing-machine to change to a state which is defined outside its own system, but does exist in the encapsulating system (Ekdahl 2000). It is still possible to describe the finite set of states from the perspective of the agent, by assuming that any state not in the agent's own system exists in the complementing systems. With that, we defined the Escape-Mode-state and -transition.

Any agent going into the Escape transition after detection of a would-be Turing-machine halt, is either entering a state which is defined in the encapsulating system or causes the agent's Deus Ex Machina (in that system) to also trigger its Escape-Mode. This pattern will repeat itself until there are no encapsulating systems left and ultimately the 'top' Deus Ex Machina is reached: the human responsible for the entire agent system (in most cases the experimenter). The human—too complex to be reduced to a Turing-machine (for the time being)—presumably has an (almost) infinite set of states and will intervene through its Deus Ex Machina-interface and put the agent(s) in a new state which can be created on the fly.

5.4 Alignment

Agents are primarily concerned with the interaction with other agents playing roles, by performing their behaviours. In the event of a halting state, the cause is most of the time a conflict between the agents' behaviours. A behaviour that does not exactly match the opposite behaviour(s) (or lack of any behaviour to begin with) brings the agent to a state where the next action is not defined and Intervention is needed. In most cases, this will encompass the alignment of the agent's behaviour with the other behaviours in the interaction by changing the agent's behaviour model.

It is assumed that humans have (at least access to) knowledge about the entire interaction, since they exist in a system encapsulating (at least) the participating agent(s′) system(s). The actions of a human aligning the behaviour of agents in an interaction can be captured and logged as an alignment policy which can be used whenever a similar problem emerges (with the same pattern of interaction, roles and behaviours). The logged alignment policies can be considered as patches applied by the agent themselves on occasion by exception. However, the agent is also capable of changing its own behaviours by applying the policy, and enriching itself with a new behaviour. This is where the AGE framework displays its most important feature: growth.

The graphical language TALL, in which we can represent behaviour, allows for easy dynamic behaviour editing and ensures soundness with respect to the rules defining Petri-nets (which in turn ensures no transitions into Turing's halting states). The editing tools in AGE support the user in charge of the agent to define alignment policies which can be used by the agent or exchanged between agents if necessary (inter-agent-growth / alignment). The formalisms behind the alignment are presented by Meyer in his work (Meyer and Szirbik 2007a, b).

6 Discussion

In organisations the pattern of error detection, correction from hierarchically superior participants, and local learning by agents, are the most basic ways to adjust to various forms of change. Change can appear as an infusion of new ideas, appearance/disappearance of new agents in the organisation, pressure from the environment to change, etc. This pattern of adapting to change is typical of interactional processes, and it is natural to use tools and methods inspired from process management. Albeit human agents are not representing mentally their interaction beliefs as graphs (maybe with the exception of those humans who are working intensively with workflow systems that are based on graphical languages), they are very good in

understanding a process description that is depicted by a flowchart, Petri-net, or something alike. If a process is to be supported by software agents in a one-human-to-one-software-agent fashion, the interaction behaviours of these agents are presented to their owners in graphical form, and the owners could intervene when necessary by changing these depictions on the fly. That means that the software agents change their behaviour according to the new understandings of their owners, who respond to escapes triggered by built-in mechanisms of the software agents. They may also intervene when they observe that the behaviour of their agents is not aligned with the behaviour of other agents.

When developing support software in the form of a MAS, most of the current approaches tend to model the behaviour of the agents in its entirety (as perceived from the system requirements), before the system has any role in support. Of course, requirements are usually implemented iteratively, and after each iteration, testing (which can be realized via scenario testing) is performed. However, the system will exhibit support characteristics only when most of the systems requirements have been implemented. The TAL approach is rather different. Starting with a model that has "empty" agents, the process is carried out via escape and interventions of the players and experimenters that participate in the interactive simulation game. These players are "piloting" the simulated agents, guiding them in all the necessary social interactions. All activities and information passing is logged, this providing the basis for building the base of the local interaction beliefs (behavioural descriptions). As the simulation game progresses in complexity, trying more scenarios and looking for exceptions, the agents grow their behavioural base and the process is emerging as an internal capability of the simulated agents, and becomes less "piloted" by the external players.

In an ideal situation, the players can be completely satisfied with the ways the process is carried out, and the simulated agents become the software agents of the MAS used for support in the real organization. If the simulation and growth supposedly covered all the potential scenarios (which is of course impossible), in theory, the MAS can run the process by itself, without human intervention. In reality, such a situation is naturally impossible, and humans are needed to intervene in unforeseen circumstances and help escaping software agents. That leads to the necessity to leave the MAS with the same capabilities for escape and intervention as in the simulated environment provided by AGE. Actually, the infrastructure we intend to use for the MAS, after the system is deployed is just a variant of the one used during simulation and growth.

By enriching incrementally the behaviour of the agents, it could be said that agents "grow". The escape/intervention rate decreases and the game/simulation does not have to be controlled from "outside" by the human players and experimenters (the Dei ex Machinae).

The final and most important argument to include an escape/intervention mechanism in any MAS is ethical. Software agents and business support software are developed with morality principles in mind (agents should not cheat, lie, or steal (Hawkins 2004), and also the laws of Asimov, in a more general moral sense). However, design requirements based on an ethical system developed for a specific business does not automatically lead to a system that enforces 100% the ethical system (it may inhibit negative behaviours and enable positive behaviour, (Gill 2007). An incremental development system where the potential users are permanently in the loop and can see what the systems does and how the agents are interacting, allows non-ethical behaviour to be detected and corrected early. Moreover, if the system is deployed and still has the escape intervention mechanism activated, it is very easy for the users to intervene when software (and also human) agents are violating the ethical systems that is enacted. This means that the users still need access to a visualisation of interactions, in order to be able to "see" the process unfolding.

7 Conclusions

The interaction and escape are patterns that are inspired from real-life social interactions, commonly encountered in organizations and social settings where the actors interact for a meaningful purpose (societal or business-like goal). In these settings, escape and intervention happen more often when the actors have limited experience and knowledge about how the objectives are achieved via a common collaborative effort of all the actors. Human actors who are not sure what to do in a certain situation are asking either other actors with whom they interact for help, but more natural is to ask a hierarchically superior actor, who supposedly "knows better", and can take responsibility. In a symmetric way, hierarchically superior agents intervene when they detect that the activity of subordinate agents is not up to their expectations, or when they detect that the subordinates are just making blatant and visible mistakes. They will try to correct the behaviour of the subordinate agents according to their experience and views. Of course, the intervention-derived change can go in the other way, sometimes the superior agent realizes that the subordinate is exhibiting the correct behaviour and he, the superior agent, has to adapt its own views.

One of the immediate future steps for this research is to deploy the framework in a real environment. A pilot (Pelletier et al. 2005) was prepared with a major gas transporting company, where an AGE-derived MAS will

support the complex short term contracting process with shippers and brokers. Another direction for research, where results have been already published (Roest and Szirbik 2006) is to reduce the escape/intervention rate. Currently, this is slowing down growth and necessitates very long game/simulation sessions and players' time. Some simple automatic alignment algorithms have been implemented and it was investigated how to formalize alignment policies and how to select them. A very interesting avenue for research is to enable software agents themselves to help escaping agents and intervene when necessary. This is in line with the desire to have the graceful degradation characteristic. From the ethical perspective, it would be very interesting to investigate how agents can be triggered to escape by detecting un-ethical behaviour.

Always, any ontologically based development (as is the agent conceptual architecture, description languages, development philosophy) should be inspired and motivated by a social context. As connectionism was inspired by the brain, evolutionary computing by genetics, e-commerce applications on trust models, agent-oriented technologies should be driven by observed patterns from collaborative behaviours and social interactions. Other researchers have been pointing this out for a time, but it is surprising how few patterns of this kind have been added to the agent methods in the last decade. We hope that this natural social pattern of escape/intervention will prove its merits in agent-oriented developments.

References

Anonymous (1999) Technical report WFMC-TC-1011/02/99 workflow management coalition terminology glossary. http://citeseer.ist.psu.edu/255352.html

Caire G et al (2001) Agent oriented analysis using MESSAGE/UML, second international workshop on agent-oriented software engineering. Montreal Canada, May 2001

Chapanis A (1996) Human factors in systems engineering. Wiley, NY

Ekdahl B (2000) Agents as anticipatory systems. In: 4th World Multiconference on Systemic, Cybernetics and Informatics (SCI 2002), Orlando, USA

Ellis CA (2000) An evaluation framework for collaborative system. Boulder (USA): Colorado University Technical Report CU-CS-901-00

Ellis CA, Gibbs SJ, Rein G (1991) Groupware: some issues and experiences. Commun ACM 34(1):39–58

Fruchter R (2001) Bricks & Bits & Interaction. In: Lecture notes in computer science, vol 2253. Springer, Berlin

Gill SP (2007) Socio-ethics of interaction with intelligent interactive technologies, AI & Society, Online first

Hawkins S (2004) Ethical and moral issues facing the virtual organisation. In: Camarinha-Matos LM (ed) Collaborative virtual organisations. Kluwer, Boston

Jennings NR, Sycara K, Wooldridge M (1998) A roadmap of agent research and development. Auton Agent Multi-Agent Syst 1(1):7–38

Meyer GG, Szirbik NB (2007a) Anticipatory alignment mechanisms for behavioural learning in multi agent systems. In: Butz MV et al (eds) Lecture notes in artificial intelligence, vol LNAI 4520. Springer, Berlin, pp 325–344

Meyer GG, Szirbik NB (2007b) Agent behavior alignment: a mechanism to overcome problems in agent interactions during runtime. In: Proceedings of the cooperative information agents (CIA07) workshop, vol LNAI 4576. Delft, September, 2007, pp 19–21

Nijholt A, op den Akker R, Heylen D (2006) Meetings and meeting modelling in smart environments. In: AI & Society, vol 20. Springer, Berlin, pp 202–220

Norta A (2004) Web supported enactment of petri-net based workflows with XLR/flower. In: Proceedings of international conference on application the 25th and theory of petri nets. Bolonga, Italy, 21–25 June 2004

Pelletier C, Maruster L, Snoo de C (2005) Planning in gas transmission: an agent-based approach. In: 7th SIMONE congress, Lednice, Czech Republic, 11–14 October 2005

Roest GB, Szirbik NB (2006) Intervention and escape mode: discovering behaviour in agent game/simulations. In: Proceedings of the agent-oriented software engineering (AOSE) workshop at the fifth international joint conference on autonomous agents & multi agent systems (AAMAS06), Hakodate, Japan, May 2006. pp 109–120

Rosenberg D, Foley S, Lievonen M, Kammas S, Crisp MJ (2005) Interaction spaces in computer-mediated communication, AI & Society, vol 19. Springer, Berlin, pp 22–33

Sierhuis M, Clancey WJ, v Hoof R (2003) Brahms: a multiagent modelling environment for simulation social phenomena. In: First conference of the European Social Simulation Association (SIMSOC VI), Groningen, The Netherlands

Stuit M, Szirbik NB (2006) Embodied interactions: a way to model and execute complex and changing business processes with agents. In: Proceedings of the 7th annual international workshop on engineering societies in the agent world. Dublin, September 2006, pp 169–184

Stuit M, Szirbik NB (2007) Modelling and executing complex and dynamic business processes by reification of agent interactions. In: GO' Hare et al (eds) Lecture notes in artificial intelligence, vol LNAI 4457. Springer, Berlin, pp 106–125

Stuit M, Szirbik NB, de Snoo C (2007) Interaction beliefs: a way to understand emergent organizational behaviour. In: Proceedings of the ICEIS 2007 conference, vol. "Software Agents and Internet Computing", Funchal, Madeira, 12–16 June 2007. pp 241–248

Suchman L (1987) Plans and situated actions: the problem of human-machine communication. Cambridge University Press, Cambridge

van der Aalst WMP (2004) Business process management demystified: a tutorial on models, systems, and standards for workflow management. In: Desel J, Reisig W, Rosenberg G (eds) Lectures on concurrency and petri nets: advances in petri nets. LNCS, vol 3098. Springer, Berlin, pp 1–65

van der Aalst WMP (2007) Exploring the CSCW spectrum using process mining. Adv Eng Informatics 21(2):191–199

Wagner G (2003) The agent-object-relationship metamodel: towards a unified view of state and behaviour. Inf Syst 28(5):475–504

Wooldridge M (2002) An introduction to multi-agent systems. Wiley, Chichester