Composing Cognitive Agents from Behavioural Models in PRESTO

Paolo Busetta
Delta Informatica Spa
Trento, Italy
Email: paolo.busetta@deltainformatica.eu

Mauro Dragoni
FBK
Trento, Italy
Email: mauro.dragoni@fbk.eu

***Abstract*—The PRESTO project applies agent technologies to serious gaming. PRESTO has developed an AI infrastructure and an agent framework called DICE for the creation of game-independent, modular Non-Player Characters (NPC) behaviours based on a BDI (Belief-Desire-Intention) approach enriched with cognitive extensions for human simulation. Behavioural models can be combined via end-user development tools to form the behavioural profiles of NPCs in a game. DICE provides the coordination between body-controlling behavioural models (for navigation as well as posture, facial expressions, actioning) and decision-making models representing e.g. the standard operating procedures of professional roles, the cognitive appraisal of events and perceptions, the modality of reaction to unplanned events occurring during a game. Behavioural models are largely if not completely independent of the specific scenario or even game engine thanks to abstractions of both the environment and the internal state of the NPC provided by means of ontologies. PRESTO is producing a set of behavioural models targeted at its pilot project's needs or expected to be of common use, including navigation in the virtual environment sensitive to the cognitive state of the NPC. This paper gives a brief overview of PRESTO, DICE and of its ontologies.**

I. Introduction

PRESTO (Plausible Representation of Emergency Scenarios for Training Operations) [2], [3] aims at adding semantics to a virtual environment and modularising the artificial intelligence controlling the behaviours of NPCs (Non Player Characters, i.e. artificial players in a game). Its main goal is to support a productive end-user development environment directed to trainers building scenarios for serious games (in particular to simulate emergency situations such as road and industrial accidents, fires and so on) and in general to game masters wanting to customize and enrich the human player's experience. The framework for behavioural modeling in PRESTO, called DICE, was inspired by a BDI (Belief-Desire-Intention) [1], [9] multi-agent system with cognitive extensions, CoJACK [10], [6]. PRESTO offers powerful end-user development tools for defining the parts played by virtual actors (as end user-written behaviours) and the overall session script of a game. PRESTO supports a specific virtual reality, XVR from E-Semble, a well known tool in use for Emergency Management and Training (EMT) in a number of schools and organisations around the world, as well as Unity 3D and, at least in principle, is agnostic with respect to the game engine in use.

The next section explains the motivations behind PRESTO with an example and gives an overview of the overall system, while Sec. III briefly presents DICE. The motivations and design of the semantic and end-user facilities are sketched out in Sec. IV. Sec. V provides details about the navigation subsystem and how it is affected by the cognitive state of agents. Sec. VI briefly presents the work currently in progress in the area of coordination, while Sec. VII summarizes the state of development and experimentation at the time of writing.

II. Directing NPCs as virtual actors in a virtual stage.

Serious games have the potential to dramatically improve the quality of training in a number of fields where the trainee has to face complex and potentially life-threatening situations. In particular, open-world 3D simulations (also called ''sand-box" or ''free-roaming" games) have been used for quite a long time by the military, with a few products reaching a significant market success, and are becoming common in civilian emergency training because they allow the rapid construction of scenarios for the rehearsal of safety procedures. The main limitation of current technology concerns NPCs, whose behaviour may be quite sophisticated when performing predefined tasks but is often unaffected by context; further, a professional programmer is required for the implementation of any procedure that cannot be described with the simple selection of a few waypoints and the choice of a few actions, let alone introducing variants due to psychological factors. These issues lead to repetitive and hardly credible scenarios and to the slow and costly development of new ones when many NPCs are involved.

As an example, consider a fire breaking in a hospital ward during daytime with patients with different impairments, visitors of various ages and professionals with different roles, experiences and training. In this scenario, which is taken from the pilot project of PRESTO, most characters are NPCs while the human players, i.e. the trainees, are either health professionals that could be in charge for a ward at the time of an accident or emergency staff called to help. A training session would require two apparently conflicting abilities from NPCs. From the one hand, they should act autonomously according to a variety of parameters concerning e.g. their physical and psychological state, their current position, their capabilities; e.g. visitors may act rationally and follow well-marked escape routes or flee panicking to the closest exits, nurses at the start of their shift are fully responsive and careful while at the end of the shift fatigue may lead to errors, and so on. On the other hand, in order to make training effective

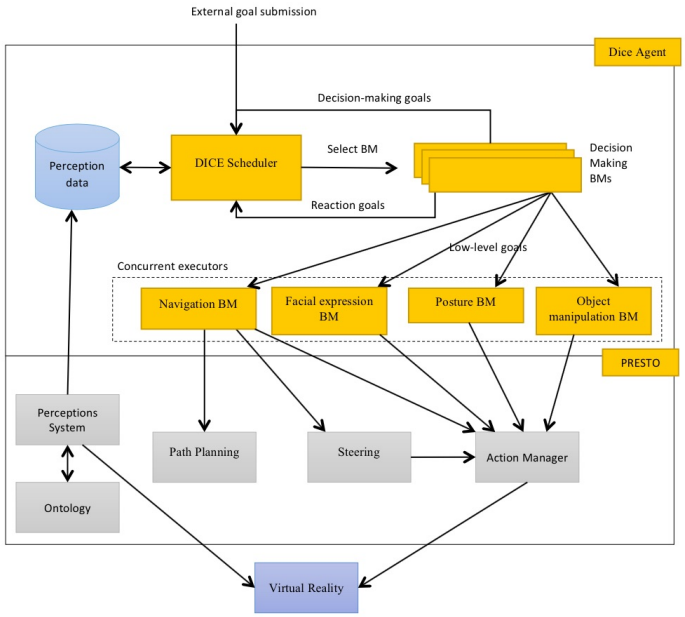

Fig. 1. Simplified DICE architecture with navigation highlighted (BM: Behavioural Model)

and engaging, the trainer supervising a simulation session should be able to temporarily suspend it (e.g. to give feedback to the trainees), change the course of events or affect the way certain characters behave (e.g. to introduce more drama or rehearse different procedures), as well as introducing or removing characters in following runs of the same scenario. Hard-coding all possibilities, assuming that this is supported by the game in use, is a laborious task to say the least.

The objective of PRESTO is to allow NPCs to act as "virtual actors" because they are able to "interpret" a part written at a higher level of abstraction than with common scripting languages, with additional modalities (that may correspond to, e.g., levels of skills or psychological profiles) that can be selected at the beginning but changed during a game as a result of the application of rules or by explicit user choice. The game's master (i.e. the trainer) is empowered to become a "director" able to "brief" virtual actors, that is, to define the parts the artificial characters have to play by means of a language aimed to non-programmers that composes more fundamental even if potentially very complex behaviours into game-specific sequences. Key enablers are end-user development tools [7] and the ability to mix and match behavioural components taken off-the-shelf from a market place (similar in principle to asset stores in popular gaming platforms such as Unity).

III. NPC Cognitive Architecture

PRESTO creates a DICE agent for each NPC in a game according to scenario-specific configurations.

DICE (Fig. 1) is a BDI framework that supports multi-goal modeling of NPC behaviours, where navigation, body postures and facial expressions, manipulation of objects and decision-making concerning tactical and long-term objectives are controlled by concurrent threads (implemented, in BDI speak, as intention trees achieving independent hierarchies of goals and subgoals). Furthermore, decision-making in DICE happens at two levels, controlled by independent "planned" and "reaction" intention trees. A decision-making behaviour started in reaction to an event pre-empts and blocks the execution of a planned behaviour until it is fully completed, at which point the planned behaviour is resumed. This allows, for instance, to have short-term reactions to perceptions (such as hearing a noise) that partially change the NPC state (e.g. by pointing the head towards the source of the noise) while not affecting navigation or longer-term procedures if not required. All behaviours in the body-controlling intention trees and in decision-making can be overriden by new behaviours at any time, e.g. as new perceptions are processed, as part of a decision-making routine, as a user choice from a GUI, as a command from a PRESTO session-controlling script; at any time, no more than one behaviour for each intention tree is active.

A DICE agent is built by composing so-called ''behavioural models", which are BDI capabilities [5] able to achieve a predefined set of goals (''role"). Changes in behaviour due to emotions, fatigue or other non-rational factors can be dealt within DICE in various ways, of which the most novel (and dramatic) is by defining behavioral rules that select alternative behavioural models according to the current cognitive state of the NPC. These rules can be defined directly by the end user, who is enabled to change the behavioural profiles of her characters according to the evolution of the game or even in real-time by explicit choice and from the session-level script. As in CoJACK [10], cognitive states are represented in DICE by moderators (i.e. numeric values modeling specific factors such as fear and fatigue levels) and a set of cognitive parameters computed from those moderators (modeling e.g. reactivity and accuracy), even if greatly simplified with respect to the original. Any behavioural model can use moderators and cognitive parameters to tune its own internal parameters, e.g. to decide the speed of execution of action or memory fading. Changes to moderators are normally performed by behavioural models for cognition according to appraisal rules (concerning e.g. the perception of threatening things) and time; however, it is possible to force the value of moderators at any time from any behavioural model (e.g. because of the realization of a dangerous situation) or from the session-controlling script, thus allowing the trainer to fully control the overall behaviour of an NPC during a game.

Figures 2 and 3 illustrate a (simplified) example NPC profile built as a DICE agent. The example is of a shopper able to achieve a "shop visited" goal. Fig. 2 provides a static view, organized in descriptions of capabilities (''roles"), behaviours implementing those capabilities and switching rules based on moderators that select which behavioural models are currently active. Observe that a single role (e.g. "Shopper" in the figure) may correspond to multiple behavioural models; the switching rules determine the current profile of the NPC, i.e. which models are active for each role at run-time. Roles and behavioural models are described by metadata, and they can be composed and configured by means of graphical editors.

Fig. 3 represents a snapshot of the dynamic state of the agent, with the concurrently executing intention trees and the two-level decision making. One of the implications of

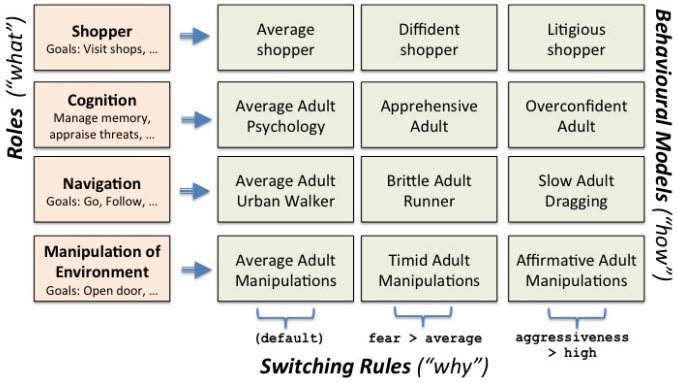

Fig. 2. Simplified DICE profile of a shopper NPC

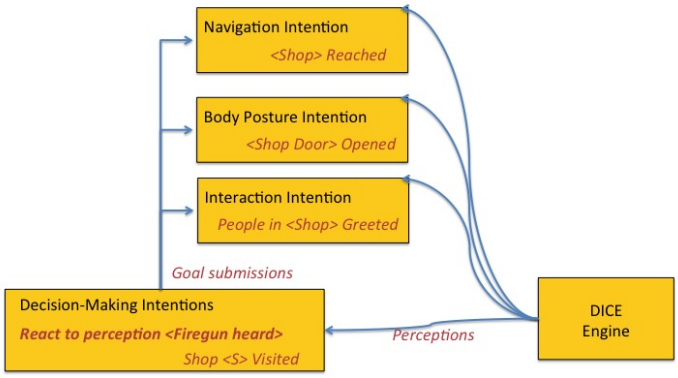

Fig. 3. Simplified DICE view of a shopper NPC during a game

the DICE approach on navigation is that, at any time, the travel direction (decided by a behaviour) can be changed and may be resumed later (e.g. when a reaction is completed); similarly, any body-controlling behaviour can be overridden and then resumed later. The APIs make programming this concurrent machinery a straightforward business, while the end-user development tool for behaviour modeling (called the DICE Parts Editor) provides an extremely powerful yet intuitive way to write scripts that affect one or more intention trees at each step [8].

IV. End-User Adaptation to Specific Scenarios

Currently, the programming of NPCs mostly relies on ad hoc specifications / implementations of their behaviors done by game developers. Thus, a specific behavior (e.g., a function emulating a panicking reaction) is hardwired to a specific item (e.g., the element "Caucasian_boy_17" in XVR) directly in the code. This generates a number of problems typical of ad hoc, low level solutions: the solution is scarcely reusable, it often depends on the specific knowledge of the code of a specific developer, and is cumbersome to modify, since every change required by the trainer has to be communicated to the developers and directly implemented in the code in a case by case manner. While this is not perceived as a major issue in entertainment games (but economics and a push for better game experiences are changing this, too), in serious gaming the cost and complexity of ad-hoc development is not covered by normal budgets. Typical solutions to this problem include, in multi-player games, the recruitment of experts to impersonate characters (such as team mates, enemies, victims, injured people, and so on) or, as in XVR, letting the trainer changing the scenario in real-time by hand.

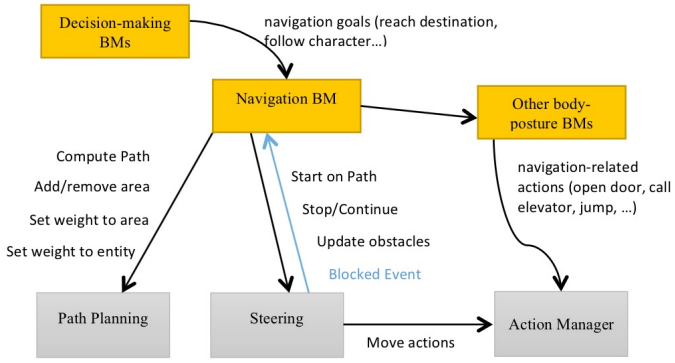

Fig. 4. Navigation subsystem architecture

PRESTO provides three main mechanisms that enable the reuse and adaptation of behavioural models to different scenarios, games or even game engines: semantic facilities, an interpreter of scripts in DICE, and facilities for game session control.

The semantization of the game environment and of part of the cognitive states of an NPC supports decision-making based on game- and scenario-independent properties. To this end, ontologies are used for the classification of objects and locations [4], for annotating them with properties and states (called "qualities") that allow abstract reasoning and for the (agent-specific) appraisals of perceptions, in particular to deal with potentially dangerous situations.

The current version of the PRESTO ontologies, targeted at its pilot project in a hospital domain, have been based on the upper level ontology DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) [11] and the classification of elements provided by XVR. DOLCE was chosen as this ontology not only provides one of the most known upper level ontologies in literature but it is also built with a strong cognitive bias, as it takes into account the ontological categories that underlie natural language and human common sense. This cognitive perspective was considered appropriate for the description of an artificial world that needs to be plausible from a human perspective. The decision to use the classification of elements provided by XVR was due to the extensive range of item available in their libraries (approximatively one thousand elements describing mainly human characters, vehicles, road related elements, and artifacts like parts of buildings) and the popularity of XVR as virtual reality platform for emergency

management and training.

The construction of the PRESTO ontologies was performed by following a middle-out approach, which combined the reuse and adaptation of the conceptual characterization of top-level entities provided by DOLCE and the description of extremely concrete entities provided by the XVR environment. More in detail,

- we performed an analysis and review of the conceptual entities contained in DOLCE-lite [11] together with virtual reality experts (both trainers and developers) and selected the ones referring to concepts than needed to be described in a scenario; this analysis has originated the top part of the PRESTO ontology;
- we performed a similar analysis and review of the XVR items, together with their classifications, in order to select general concepts (e.g, vehicle, building, and so on) that refer to general game scenarios; this analysis has originated the middle part of the PRESTO ontology;
- as a third step we have injected (mapped) the specific XVR items into the ontology, thus linking the domain independent, virtual reality platform independent ontology to the specific libraries of a specific platform.

The flow of perceptions, the properties of entities, the appraisal values of DICE behavioural models are classified by means of the PRESTO ontologies, thus enabling the development of generic BDI logic (goals, plans and beliefs) independent of the scenario of use. Additionally, DICE provides an interpreter for high-level scripts, called "DICE Parts", written by means of a graphical editor by the end-user (typically a trainer during the preparation of a specific scenario). A DICE Part can invoke multiple goals concurrently on the various DICE-managed intention trees, terminate them when specific events happen (including timeouts and perceptions), define reactions to perceptions or to modifications of the internal state of the agent (including appraisals and moderators), change the state of the agents itself, and so on. While the DICE Part language is limited in its expressivity, the cost of producing a part is minuscule with respect to directly programming the underlying BDI logic; further, DICE Parts can be distributed as decision-making behavioural models on their own. Thus, an effort is required on developers of behavioural models in BDI logic to provide goal-directed behaviours that are suitable for composition within user-written parts and adaptable to different scenarios thanks to semantic-based reasoning; the PRESTO pilot project and other demonstrators are helping in accumulating experience that will feed future guidelines.

Finally, PRESTO has an end-user facility to edit and control session-level scripts inspired by interactive books. A session script is composed by a set of scenes connected as a graph. At each scene, goals can be given to NPCs (which may trigger user-written parts), their internal state changed (including emotions) and objects manipulated. The trainer starts a script at the beginning of a training session and advances it by manually navigating the graph of scenes or letting PRESTO choose the next one e.g. when certain events happen or when a timer expires. This allows a large, potentially unlimited number of different sessions to unfold from a single script with no need to reprogram the NPCs once equipped with all required behavioural models and DICE Parts. In the hospital ward example presented earlier, the initial scene would command visitors, patients and nurses to accomplish their routine goals; the script may continue with alternative scenes such as "fire breaking in a patient room" or "fire breaking in a surgical facility", each with different people involved, and then with sequences that may lead e.g. to smoke filling the area and visitors fleeing or an orderly managed situation with the intervention of fire fighters, chosen according to the decisions of the trainer and the events occurring during a session.

V. Cognitive navigation in PRESTO

As extensively discussed in [3], at the time of writing the most developed models concern the navigation in virtual environments. As it can be seen in picture in Fig. 4, navigation is split in two levels: the lower level facilities look after path planning and steering within the decided path and are implemented within the PRESTO infrastructure, close to the game engine; the higher level is concerned with control and is implemented as behavioural models for DICE. In turn, navigation control models are of two types. One type, identified as "navigation BM" in Fig. 1 and 4, satisfies the navigation goals submitted by decision-making behaviours (e.g., of reaching a destination); slightly different navigation models are provided that depend on the main physical features of the NPC, e.g. of being a human rather than a vehicle, and consequently on the NPC's ability to move and affect the environment. As mentioned above, the navigation BM runs in its own intention tree (thread of execution) concurrently with decision-making and other body-controlling behaviours. The navigation BM calls path planning and controls steering, acting according to the latter's indication in particular when it blocks because there are obstacles or there is a closed gate. A number of different decisions can be taken according to the model and to the semantics of gates or obstructing objects, which may in turn cause goals to be submitted to other body-parts behaviours (e.g. opening a door, calling a lift, and so on).

A second type of behavioural model, referred to as "navigation capabilities" and included as a decision-making module in DICE, looks after some of the cognitive aspects of navigation. In particular, the navigation capability of an NPC decides which navigation mesh (i.e. navigable surface data) to use on creation, then changes the default speed, default animations and so on according to the current sub-rational state of the agent (i.e. its moderators and cognitive parameters). Thus, PRESTO can provide capabilities specialized e.g. for quiet or excited people, for permanent or temporary physical impairments, for different types of vehicles, and so on. Navigation capabilities may access the cognitive state to tune their parameters (e.g. speed or animations); furthermore, behavioural rules may be defined to switch navigation capabilities entirely during a game depending on the NPC's moderators. For instance, a high level of fear may select a model whose default speed is running and movement animations are jerky, while a high level of fatigue may select a model doing exactly the opposite. Furthermore, the navigation capabilities satisfy goals concerning path selection, such as "stay out of sight of entity E" or "don't go through location L" (which may have been classified as dangerous by a decision-making model according to the appraisal rules of the agent), by taking note of what to

avoid and manipulating the navigation data accordingly, based on current knowledge and the flow of perceptions.

Behavioural models in DICE have their own configuration parameters, called "background knowledge". As mentioned above, the background knowledge of the navigation capability of an agent determines how much the agent knows *a priori* about the environment – it can be everything or being limited to a few areas; the navigation data is created accordingly. The flow of perceptions arriving from the PRESTO infrastructure includes also the visible polygons of the various navigation meshes; this data is used by the navigation capability to update the navigation data. The cognitive model of DICE, not discussed here, looks after short-term memory management, which includes calling the navigation capability to purge its data; that is, the agent literally forgets about where to navigate according to timing and frequency of perceptions from the environment. Out of scope of the navigation subsystem, and not discussed here, is a "search" behaviour, which is a set of decision-making procedures that can be started when a navigation goal fails with an "unknown path" error.

In the hospital fire scenario presented in the introduction, the navigation capability of a patient on a wheel chair would use a different mesh than the one selected for a visitor with normal walking capabilities, e.g. to avoid steps and stairs. The patient's background knowledge would include the navigation areas of the entire ward (since she has been there for a while) while the visitor's knowledge would be initially empty and populated while she moves in the ward; a decision-making procedure of the visitor that invokes a goal such as "go to patient room nr. 3" would initially fail because, indeed, no path can be computed and a search behaviour would need to be invoked allowing the progressive discovery of the navigation areas of the selected mesh. If, at any time during the game, a fire alarm starts ringing, its perception on both visitor and patient would trigger a (decision-making) reaction that is handled differently according to the currently active behavioural models, which in turn may depend on cognitive states such as fear. The perception of smoke and fire would submit goals such as "don't go through that area" handled by the navigation capability as mentioned above. A rationally-behaving NPC that knows the position of a location ontologically classified as "fire exit" would navigate to the latter, with a speed and a modality that depend on the currently active navigation capability (excited / not excited, walking / pushing the wheel chair); an NPC that doesn't know about fire exits or that it's too fearful to act rationally would run to the closest exit.

VI. Multi-agent coordinated behaviour via semantic tagging

Work is in progress on game-theoretical descriptions of coordinated behaviour, which include queuing and other crowding behaviours, access to shared resources, and so on that allows the definition of policies at a very abstract (meta-) level. This exploits the support in DICE for introspection, semantic tagging of goals and plans, dynamic assignment and aborting of goals and intentions as well as the ability to dynamically manipulate semantic tags (called ''qualities") of any entities including NPCs offered by PRESTO. The specification of policies is expected to substantially reduce the coding required by models and allows the reuse of the same coordination patterns in many different situations, e.g. for queuing to pass through a gate (which will be part of the navigation BMs) as well as for queuing at the entrance of an office or at the cashier in a supermarket (which are decision-making behaviours not related to navigation goals).

A simplistic (but already available and of great practical use) coordinated behaviour exploiting qualities is goal delegation from an agent to another agent. By means of the PRESTO API, any entity in a game can submit a goal to be pursued by any other entity; when the goal is enriched with a few predefined parameters, the destination DICE agent publishes the fact that it has accepted a goal or that has achieved it (or failed to achieve or refused), allowing the submitter (or any other observer, including the session script engine) to monitor and coordinate behaviours without the use of any additional agent protocol.

VII. Current state of developments and experimentation

At the time of writing (May 2015), most of the DICE framework is in place. Work is in progress on the automatic publishing of qualities concerning intentions, required by the meta-level policies described above among others. Its end-user development editors are being evaluated within a laboratory run in collaboration with the University of Trento. To this end, Delta Informatica has developed a serious game with Unity that allows the definition of the behaviour of a multiplicity of characters coordinating to run a business with walk-in visitors, emergencies forced by the human player, and so on. This game is being used as workbench for the students participating to the laboratory and for the testing and validation of the current and future developments for the entire PRESTO project.

The ontologies concerning the hospital fire scenario, used also by the session-control tool described in Sec. IV, are being validated in a pilot project in collaboration with the main Trento hospital. The purpose of this pilot is the introduction of virtual reality training for the management of fires in wards. Given the level of novelty with respect to traditional training support tools (oral presentations, questionnaires and live simulations), at this stage the pilot is still focusing on having the trainers understanding the potentiality of virtual reality by creating courses for teaching basic emergency procedures by means of XVR by E-semble. A few NPCs have been already tested within Delta Informatica's lab; their introduction in more complex training scenarios with the need of managing evacuations, coordinating personnel, etc. are expected to be experimented in the next 12 months.

VIII. Conclusions and future works

The DICE framework and the underlying PRESTO infrastructure are a practical and powerful way to introduce agent-oriented programming and semantics into games. Benefits against traditional approaches, such as finite state machines, behavioural trees and so on, include the ability to represent non-rational factors such as emotions and the support of powerful session scripting that make a PRESTO-enabled game similar to a sort of theatrical improvisation. We presented examples from the serious gaming world, but nothing prevents using PRESTO in entertainment games or for simulation without human-in-the-loop.

Work in in progress on many aspects, including multi-agent coordination as presented above, libraries of reusable behavioural models, improvements to the various engines and to the end-user tools, including facilities for community sharing of models, parts and scenario scripts.

ACKNOWLEDGMENT

In addition to the members of the implementation team in Delta Informatica (Matteo Pedrotti, Mauro Fruet, Paolo Calanca, Michele Lunelli), we thank all PRESTO research partners and in particular Chiara Ghidini (FBK), Zeno Menestrina ed Antonella De Angeli (University of Trento). PRESTO is funded by the Provincia Autonoma di Trento (PAT).

REFERENCES

[1] Michael E. Bratman, *Intention, Plans, and Practical Reason*, Harvard University Press, November 1987.

[2] Paolo Busetta, Chiara Ghidini, Matteo Pedrotti, Antonella De Angeli, and Zeno Menestrina, 'Briefing virtual actors: a first report on the presto project', in *Proceedings of the AI and Games Symposium at AISB 2014*, ed. Daniela Romano, (April 2014).

[3] Paolo Calanca, and Paolo Busetta, 'Cognitive Navigation in PRESTO', in *Proceedings of the AI and Games Symposium at AISB 2015*, to appear, (April 2015).

[4] Mauro Dragoni, Chiara Ghidini, Paolo Busetta, Mauro Fruet, and Matteo Pedrotti, 'Using ontologies for modeling virtual reality scenarios', in *to appear in Proceedings of ESWC 2015*.

[5] Paolo Busetta, Nicholas Howden, Ralph Rönnquist, and Andrew Hodgson, 'Structuring BDI Agents In Functional Clusters', in *Proceedings of the Workshop on Agent Theories Architectures and Languages (ATAL-99)*, volume 1757 of *Lecture Notes in Artificial Intelligence*, Orlando, Florida, (15-17th July 1999). Springer Verlag.

[6] Rick Evertsz, Matteo Pedrotti, Paolo Busetta, Hasan Acar, and Frank Ritter, 'Populating VBS2 with Realistic Virtual Actors', in *Conference on Behavior Representation in Modeling & Simulation (BRIMS)*, Sundance Resort, Utah, (March 30 – April 2 2009).

[7] Henry Lieberman, Fabio Paternò, Markus Klann, and Volker Wulf, 'End-User Development: An Emerging Paradigm', *End User Development*, **9**, 1–8, (2006).

[8] Zeno Menestrina, Antonella De Angeli, and Paolo Busetta, 'APE: end user development for emergency management training', in *6th International Conference on Games and Virtual Worlds for Serious Applications, VS-GAMES 2014, Valletta, Malta, September 9-12, 2014*, pp. 1–4. IEEE, (2014).

[9] Anand S. Rao and Michael P. Georgeff, 'Bdi agents: From theory to practice', in *IN PROCEEDINGS OF THE FIRST INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS (ICMAS-95*, pp. 312–319, (1995).

[10] Frank E. Ritter, Jennifer L. Bittner, Sue E. Kase, Rick Evertsz, Matteo Pedrotti, and Paolo Busetta, 'CoJACK: A high-level cognitive architecture with demonstrations of moderators, variability, and implications for situation awareness', *Biologically Inspired Cognitive Architectures*, **1**, 2–13, (July 2012).

[11] Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening ontologies with dolce. In: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, Springer-Verlag (2002) 166–181