Computing and Informatics, Vol. 34, 2015, 1201–1233

A NOVEL WAY OF USING SIMULATIONS TO SUPPORT URBAN SECURITY OPERATIONS

Marcel Kvassay, Ladislav Hluchý, Štefan Dlugolinský

Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9
845 07 Bratislava, Slovakia
e-mail: {marcel.kvassay, ladislav.hluchy, stefan.dlugolinsky}@savba.sk

Bernhard Schneider

Airbus Defence and Space GmbH, Rechliner Strasse, 85077 Manching, Germany
e-mail: bernhard.schneider@airbus.com

Holger Bracker

Airbus Defence and Space GmbH
Willy-Messerschmitt-Strasse 1, 82024 Taufkirchen, Germany
e-mail: holger.bracker@airbus.com

Aleš Tavčar, Matjaž Gams

Department of Intelligent Systems
Jožef Stefan Institute
Jamova 39
1000 Ljubljana, Slovenia
e-mail: {ales.tavcar, matjaz.gams}@ijs.si

Marc Contat

AIP (TCOIC4) – Advanced Information Processing
Airbus Defence and Space
1, Bd Jean Moulin, CS40001, MetaPole
78996 Elancourt Cedex – France
e-mail: marc.contat@airbus.com

Łukasz Dutka

Academic Computer Centre Cyfronet AGH
AGH University of Science and Technology
ul. Nawojki 11, 30-059 Kraków, Poland
e-mail: `l.dutka@cyfronet.pl`

Dariusz Król, Michał Wrzeszcz, Jacek Kitowski

Academic Computer Centre Cyfronet AGH
AGH University of Science and Technology
ul. Nawojki 11, 30-059 Kraków, Poland
&
Department of Computer Science
AGH University of Science and Technology
Al. Mickiewicza 30, 30-059 Kraków, Poland
e-mail: {`dkrol, wrzeszcz, kito`}`@agh.edu.pl`

Abstract. The growing importance of security operations in urban terrain has triggered many attempts to address the perceived gaps in the readiness of security forces for this type of combat. One way to tackle the problem is to employ simulation techniques. Simulations are widely used to support both mission rehearsal and mission analysis, but these two applications tend to be seen as distinctly separate. We argue that integrating them in a unified framework can bring significant benefits for end-users. We perform a structured walk-through of such a unified system, in which a novel approach to integration through the behaviour cloning enabled the system to capture the operational knowledge of security experts, which is often difficult to express verbally. This capability emerged as essential for the operation of the integrated system. We also illustrate how the interplay between the system components for the mission analysis and mission rehearsal is realized.

Keywords: Agent-based simulation, human behaviour modelling, behaviour cloning, virtual training, mission analysis, security operations, military operations in urban terrain, asymmetric warfare

Mathematics Subject Classification 2010: 68T42, 68U20, 91C99

1 INTRODUCTION

The growing urbanisation and industrialisation of the world implies that future conflicts "will not just spill over into the conurbations, but will in fact be centred on the cities" [4]. There is already a noticeable tendency for militarily weaker enemies "to utilise the complex urban environment to generate a combat power, especially when dealing with an opposing force which possesses more sophisticated technological weapon systems" [5]. Industrially developed countries are targeted too, most recently by various forms of terrorism. In a sense, they are victims of their own success, since few adversaries will be tempted to engage them in conventional warfare [16].

Urban combat is both difficult and dangerous. Dense networks of buildings, walls and other obstacles obstruct the view and isolate combat units, sometimes even individual soldiers [11]. At the same time, dense civilian populations and humanitarian concerns hamper the application of their superior military power. The adversary is often highly camouflaged or mingles with the civilian population and prefers sudden engagements at a very short range [4]. These are some of the reasons why urban combat "has been expensive in deaths of soldiers and non-combatants, destruction of infrastructure, consumption of military logistics resources, and often also in political terms" [5]. Most contemporary theorists agree that security forces face significant challenges in this respect [21, 8]. Regarding the perceived gaps in their preparedness, Worley, Wahlman and Gleeson in their extensive survey [21] listed a number of issues, of which the most relevant for our present purpose are

- the inadequacy of training facilities;
- the nature of operations being determined as much by human occupants as by physical structures;
- the need for rules of engagement to be simple, dynamic, and tailored to specific situations.

In this article, which is related to our conference poster [13], we deal with these issues from the point of view of asymmetric urban security threats. These are confrontational situations or open conflicts in urban areas in which the opponents of security forces are not regular military forces but rather civilian rioters (as in the case of 2001 Gothenburg riots[1]) or insurgents (as in the case of the recent ISAF mission[2] in Afghanistan). Typically, security forces are trained for such scenarios through *live training* (also termed *live simulation*) in which real people operate real equipment in specialized training centres like CENZUB[3] in France or STANTA[4] in

[1] `http://en.wikipedia.org/wiki/Gothenburg_Riots`

[2] `http://en.wikipedia.org/wiki/ISAF`

[3] `https://en.wikipedia.org/wiki/Centre_d'entra\%C3\%AEnement_aux_actions_en_zone_urbaine`

[4] `https://en.wikipedia.org/wiki/Stanford_Training_Area`

UK [17]. The main disadvantages of this training modality are that the repertoire of available urban settings is limited, trainings with large number of participants are costly, and certain scenarios cannot be trained adequately due to the physical danger they entail. These problems can be overcome, at the cost of somewhat decreased realism, by *virtual trainings* (or *virtual simulations*) in which real people representing security forces operate simulated equipment in a simulated world which includes their simulated opponents (rioters and insurgents), as well as other participants (casual bystanders, etc.). Virtual and live training together comprise *mission rehearsal*.

In contrast, by the *mission analysis* we mean a detailed investigation of the properties of the preferred and alternative courses of action stipulated by the mission guidelines. These properties are usually expressed in the form of various measures of effectiveness (MoE), such as the number of injured persons at the end of a scenario. They help mission planners assess the mission guidelines and their robustness with respect to potentially varying behaviour by the opponents of security forces. Such investigations typically require a larger number of scenario replays than can be handled in live or even virtual trainings. In consequence, they are performed through *constructive simulations* or *data farming* in which both security forces and their opponents are simulated.

The main thesis of our paper is that an integrated approach linking the mission rehearsal with the mission analysis through behaviour cloning can bring significant benefits for end-users. We argue our case in the context of project EUSAS ("European Urban Simulation for Asymmetric Scenarios") financed by 20 nations under the *Joint Investment Program on Force Protection* of the European Defence Agency. The system that we designed and prototyped in this project addressed the first issue – the inadequacy of training facilities – by offering a mission rehearsal capability in a highly realistic 3-D cyber environment VBS2. The second issue – the role of human participants – was addressed by modelling human behaviour on the basis of latest findings in psychology. In this respect, the PECS reference model [20, 18] served as a solid modelling basis. Finally, the third issue – the flexibility of rules of engagement – was addressed by behaviour cloning in combination with a breakdown of high-level scenarios into smaller units, so-called vignettes. These represent dynamic and adaptable micro-scenarios and each can be associated with a different "optimal" approach.

The main benefits of the integrated system can be summarized as follows:

1. It captures the security expert knowledge through behaviour cloning and uses it for realistic simulation of security personnel.
2. It guarantees that the mission rehearsal is consistent with the mission analysis since both are based on the same simulation scenario and share the executable code driving the behaviour of simulated human characters.
3. Optimizations found in the mission analysis can be quickly validated in a new training session.

In all of these benefits, behaviour cloning acts as a kingpin tying together the mission analysis and the mission rehearsal components of the system.

The rest of the article is structured as follows. Part 2 deals with the structure and workflows of the EUSAS system. In Section 2.1 we introduce its main operation modes; in Section 2.2 we describe its core component – our agent-based simulator; in Section 2.3 we briefly explain how users can define and calibrate new agent types. Section 2.4 then presents the basic ideas and principles underlying the core activity of the EUSAS system – the behaviour cloning algorithm CMASDA. Sections 2.5 and 2.6 conclude the overview with salient aspects of data farming, data analysis and mission rehearsal. Part 3 is dedicated to experimentation and validation activities. In Section 3.1 we briefly reiterate our early experiments and results, most of which were already published. In Sections 3.2 and 3.3 we present, for the first time, the comprehensive verification and validation results for our behaviour cloning approach and the final project demonstration of the EUSAS system. Finally, in the conclusion, we summarise our findings and directions for the future work.

2 THE EUSAS SYSTEM

2.1 Main Operation Modes

Activities in the integrated EUSAS system can be broadly grouped into two main operation modes – preparation and regular use – as shown in Figure 1. Dashed line arrows indicate the preparation, standard arrows the regular use. Dotted line arrows correspond to calibration feedback leading to agent model adjustments. Preparation and calibration comprise "the training of the system", while the regular use can be viewed as "the training of trainees". Numbering of activities suggests their natural sequence, but there is no hard and fast rule about it, and some activities, e.g. calibration, can be performed more than once during one cycle.

The following enumeration provides an overview of activities in their natural order.

1. **Mission guidelines creation:** drawing on their previous experience, mission planners formulate a first draft of new mission guidelines. These specify rules and procedures for security forces in order to cope with specific security threats and situations.
2. **Adapting agent models:** based on the intended training situation, civilian agents with scenario related characteristics are defined and assembled from predefined model components. At this stage, the agents representing security forces consist of simple scripts where the main purpose is to help in calibrating the civilian models.
3. **Adapting simulation scenarios:** A map of terrain for the desired vignette (micro-scenario) is selected, agent instances are placed in appropriate starting positions and their initial parameter values are set.

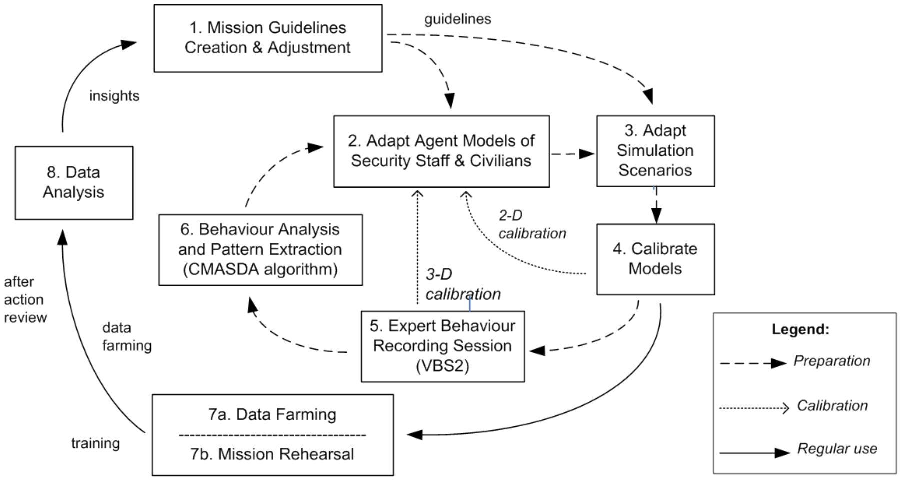

1. Mission Guidelines Creation & Adjustment
guidelines
insights
2. Adapt Agent Models of Security Staff & Civilians
3. Adapt Simulation Scenarios
8. Data Analysis
6. Behaviour Analysis and Pattern Extraction (CMASDA algorithm)
2-D calibration
4. Calibrate Models
3-D calibration
after action review
data farming
5. Expert Behaviour Recording Session (VBS2)
7a. Data Farming
7b. Mission Rehearsal
training
Legend:
Preparation
Calibration
Regular use

Figure 1. Main phases and activities in the EUSAS system

4. **Calibration** (*constructive simulation* with simplified security personnel agents): Its primary purpose is to calibrate *civilian* models e.g. by face validation through a simplified 2-dimensional visualization of their behaviour. The resulting model adjustments are indicated in Figure 1 by the feedback arrow `2-D calibration`.
5. **Expert behaviour recording session:** Calibrated civilian models are coupled with a 3-D virtual rendering engine VBS2, so that experienced security personnel can "play against" them and find effective strategies for handling the situation. At the same time, 3-D visualization may reveal deficiencies in civilian models that went unnoticed during 2-D calibration. Figure 1 indicates such subsequent model adjustments by the second feedback arrow `3-D calibration`. When the security experts feel the civilian models are adequate and find effective strategies for handling them, the logs of such "successful" games are passed on to CMASDA algorithm for behaviour pattern extraction.
6. **Behaviour analysis and pattern extraction:** CMASDA algorithm processes the logs and creates the so-called abstract action graph. This graph is mined for behaviour patterns capturing significant aspects of expert strategies; these patterns are then transferred to the simulated security personnel agents. 2-D calibration is repeated again, this time with the focus on the adequacy of the *cloned security personnel* behaviours. This step concludes the preparation; the system is ready for regular use.
7. **Data farming and mission rehearsal:** It is advisable to commence the regular use of the system with data farming, where a number of constructive simulations of the same scenario are run in parallel, each with slightly different input parameters. Analysis of their results helps determine the robustness of the cloned strategies, i.e. their resilience to the change of model parameters. Once

their robustness for the intended training situation is confirmed, the system can be used for the mission rehearsal (virtual training).

8. **Data analysis:** Both the data farming and mission rehearsal rely on data analysis tools for evaluation. The analysis typically starts with statistical techniques, such as regression trees and histograms. They represent each simulation run by one data point associating one input vector of parameter values with its corresponding output value. The output value is usually one of predefined measures of effectiveness, such as the number of injured persons at the end of a scenario. In order to understand the reasons behind specific results (e.g. why a particular parameter setting led to an extreme number of injuries), more specialized tools may be needed. Logs of mission rehearsals can also be analyzed in this way or through the standard after-action review tool included in the VBS2 package.

An insight gained through data analysis may trigger adjustments to the mission guidelines, and the whole workflow may be repeated as many times as necessary. In the following sections we elaborate on key components and activities of this iterative process, broadly following the natural sequence of steps listed above.

2.2 Agent-Based Models and the Simulator

Adjustments to agent models and scenarios, as well as their calibration (activities 2–4 in Figure 1) are typically performed in the environment of ABS, our agent-based simulator, and related tools. ABS was built by extending the MASON simulation library[5], which the project team selected through an evaluation by implementation exercise reported in [15]. MASON's modularity and flexibility helped us efficiently implement the human behaviour models designed and developed in project EUSAS. These models incorporate the latest findings in psychology and are expressed in the so-called PECS reference model [20]. The acronym PECS stands for Physical conditions, Emotional state, Cognitive capabilities and Social status. According to [18], "PECS is a multi-purpose reference model for the simulation of human behaviour in a social environment," with the emphasis on the "emergent behaviour… typical for groups and societies formation." The outer context in PECS is provided by a so-called "agent world," which comprises the environment, connector and agents.

Our current version of the agent world implemented in ABS is shown in Figure 2. The figure also shows the steps comprising a typical simulation cycle: In step 1, each agent reads relevant messages and events from the connector and updates its internal state accordingly. In step 2, each agent notifies the connector about its own actions and significant internal changes so that these can be noticed and acted upon by other agents, as well as logged. Step 3 includes direct queries to the environment during the behaviour planning process, e.g. an angry protester agent may ask the

[5] `http://www.cs.gmu.edu/~eclab/projects/mason/`

environment for the location of the closest security person in order to attack him or her. Step 4 represents the events generated by the objects in the environment (e.g. explosions) and system broadcasts, which in this way become "visible" to agents. In step 5, the connector logs all the received events and messages for subsequent data analysis.

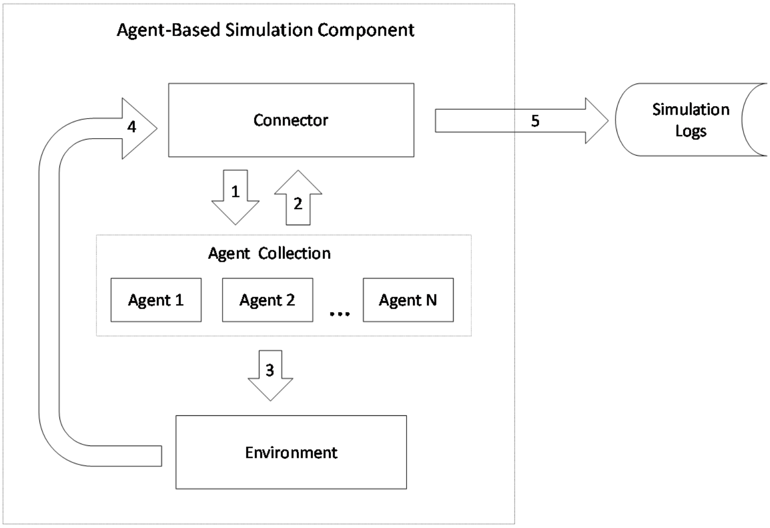


Figure 2. A simplified view of the internal structure of our agent-based simulator (ABS)

The internal structure of PECS agents follows general systems theory and comprises input, internal state and output. Each subsystem is further subdivided. The internal state, for example, consists of Social Status, Cognition, Emotion and Physical conditions components. Components are interconnected by an intricate network of causal dependencies and information flows, and each is represented by its own internal state variables and their state transition functions. The global state transition function of a PECS agent consists of a collection of state transition functions of all its state variables. Their mathematical form is not constrained, giving the modeller an almost total freedom in shaping the agent behaviour. A more detailed description of the PECS reference model is provided in [20, 18]. Based on these ideas, we designed and developed a generic agent structure shown in Figure 3. Although at first sight it might look different from the "canonical" PECS agent structure provided e.g. in [18], it is nevertheless compatible.

Human behaviour modelling is a large topic and the scope of this article does not permit its full elaboration. Instead, we shall briefly introduce the main agent components depicted in Figure 3. Sensory perception of our agents is implemented in the form of reading filtered messages from the connector, with filters representing their sensory limitations. The middle part of the figure shows the components representing their internal state. The collection of motives and states on the right contains

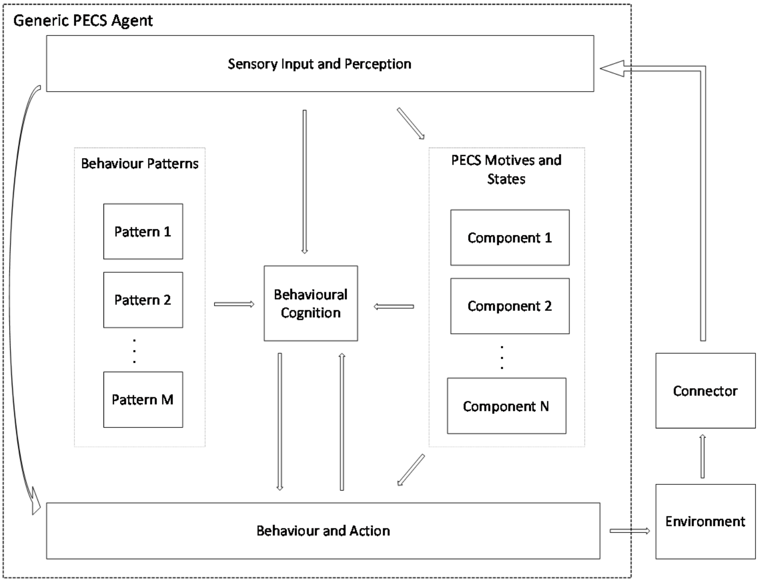


Figure 3. A simplified view of the internal structure of our generic agents used in ABS

factors that either drive the agent to action (e.g. emotions, such as fear or anger), or modulate it (e.g. physical energy the lack of which may slow down or immobilize the agent). Their dynamics may combine a continuous part (differential equations) with a discrete part (typically capturing sudden impacts of external events, such as gun shots). The motives have to be numerically comparable; therefore they are normalized to the percentage scale, with 0 % representing their minimum and 100 % their maximum intensity. The behaviour patterns on the left represent "goal-achieving" mechanisms and incorporate elements of protocol memory and planning of the canonical PECS model. Behaviours are conceptualized as sequences of atomic and uninterruptible *elementary actions,* such as one step in a certain direction, one gun shot or one stone-throw. The "Behavioural Cognition" component in the middle selects the "right" behaviour pattern to be activated at suitable points in time on the basis of the agent's strongest motive. The "Behaviour and Action" component then starts executing that pattern in a stepwise fashion, verifying the feasibility of each elementary action before its execution.

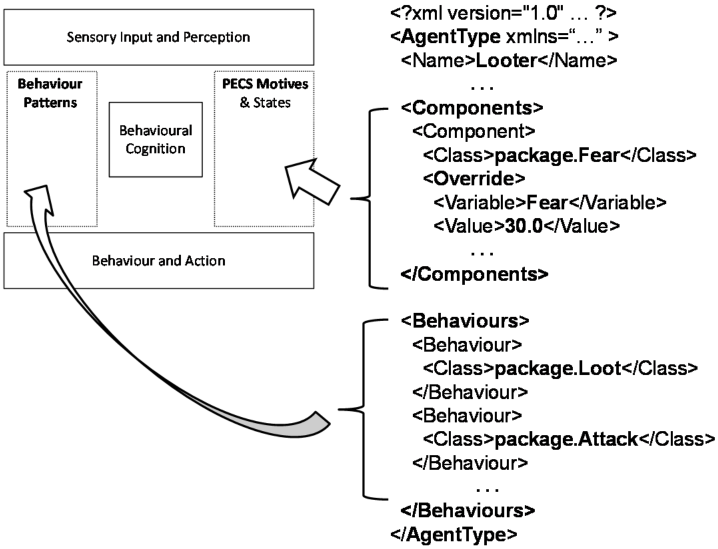
ABS is in many respects the core component of the EUSAS system. In Data Farming, for example, it is the many instances of ABS that run in parallel (with visualization components switched off) and simulate the behaviour of both the security staff and the civilian participants. In mission rehearsals, ABS simulates "the brains of the civilians" and is used as a plug-in for the 3-D visualization environment VBS2, while the real security personnel (trainees) interact with VBS2 through the mouse and keyboard. ABS is also crucial in model calibration, where its 2-D graphical user interface comes to the fore – a topic that we elaborate on in the next subsection.

2.3 User-Defined Agent Types and Their Calibration

Calibration of agents is typically performed in the context of a specific scenario stipulated by the mission guidelines. For illustration, we shall use the "vignettes" (micro-scenarios) of project EUSAS, which are broadly inspired by the recent ISAF Mission in Afghanistan. These vignettes fall into two groups: entrance control and patrol. One representative of the first group is Vignette 1.1, which models turmoil in front of a pedestrian entrance to a military base, and which we used in [9] to describe the first version of the EUSAS system (V1). In the second group there are two vignettes – 2.1 and 2.2 – and both deal with a crowd looting a shop. While in the first case the looting is genuine and the patrol stands a good chance of dispersing the crowd, in the second case the looting has been staged as a trap and the real intention is to attack the approaching patrol from flank and rear with cold weapons. We concentrated on vignettes 2.1 and 2.2 while building the second version of the EUSAS system (V2) and we shall use Vignette 2.2 here to illustrate the process of calibration.

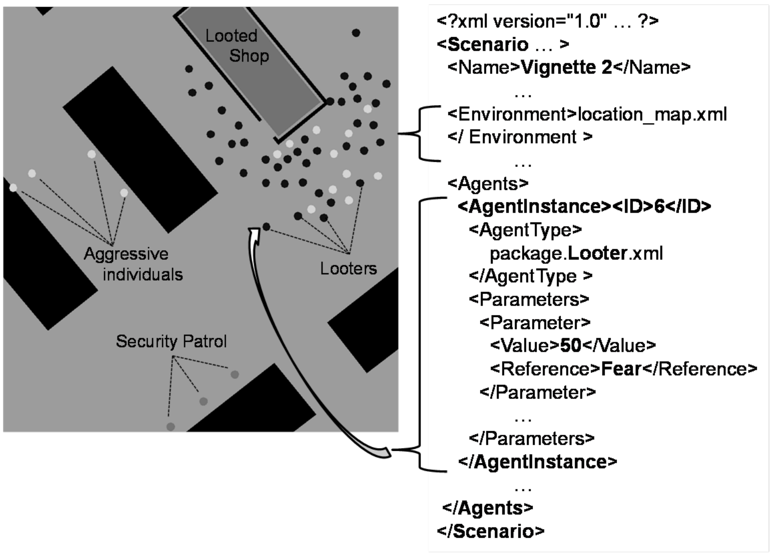
In the EUSAS system, new agent types can be defined by choosing their desired behaviour patterns, the motives that trigger them and their default parameter values. We have developed a prototype of a web-based tool which would guide users through this process and also check the consistency of the new types. The same tool would then be used to define simulation scenarios, i.e., to choose the map of the environment, place the agent instances in their starting positions and, if needed, further adjust their *individual* parameter values. These agent and scenario definitions are stored in dedicated XML files which our simulator loads at the beginning of the simulation. Based on them, the simulator creates the desired environment and agent instances, and then executes the simulation. We illustrate this process in a simplified fashion in Figures 4 and 5.

The simplified XML fragment in the right part of Figure 4 defines a new agent type called *Looter*. This is one of two civilian types participating in Vignette 2.2 (the other being the "violence-prone" civilians). The `<Components>` part of the definition is meant primarily for motives and state variables. Here we see that *Looters* are endowed with simulated emotion *Fear* whose default initial value is set to 30 % of maximum intensity. This emotion is implemented as an executable Java class `package.Fear` specified in the `<Class>` element. When instantiating an agent of this type, our simulator loads the Java classes specified in the `<Class>` section of each `<Component>` element, places them in the agent's internal collection of motives and states, and initializes their internal variables to values specified in their respective `<Override>` sections. Behaviour patterns are specified analogously in the `<Behaviours>` part of the definition and Java classes implementing them are loaded into the agent's second internal collection meant for behaviours. Once the user has defined all needed agent types, he or she can include them in simulation scenarios.



```
<?xml version="1.0" … ?>
<AgentType xmlns="…" >
  <Name>Looter</Name>
        . . .
  <Components>
    <Component>
      <Class>package.Fear</Class>
      <Override>
        <Variable>Fear</Variable>
        <Value>30.0</Value>
          . . .
  </Components>
  <Behaviours>
    <Behaviour>
      <Class>package.Loot</Class>
    </Behaviour>
    <Behaviour>
      <Class>package.Attack</Class>
    </Behaviour>
          . . .
  </Behaviours>
</AgentType>
```

Figure 4. Creating an executable instance of a user-defined agent type from its XML definition



```
<?xml version="1.0" … ?>
<Scenario … >
  <Name>Vignette 2</Name>
        …
  <Environment>location_map.xml
  </ Environment >
        …
  <Agents>
    <AgentInstance><ID>6</ID>
      <AgentType>
        package.Looter.xml
      </AgentType >
      <Parameters>
        <Parameter>
          <Value>50</Value>
          <Reference>Fear</Reference>
        </Parameter>
          …
      </Parameters>
    </AgentInstance>
        …
  </Agents>
</Scenario>
```

Figure 5. Positioning and configuring agent instances through a scenario definition XML file

A simplified XML fragment of Vignette 2.2 definition is shown in the right part of Figure 5. The `<Environment>` element refers to the physical environment (geographical map) in which the agents will move. This map is displayed in the left part of the figure and captures a bird's eye view of the initial setting of Vignette 2.2. Black areas represent buildings and barriers unreachable to agents, while the rectangle with gray interior near the top is the shop being looted. It is surrounded by dots, each representing one agent. The dark dots are looters; the light-coloured ones are violence-prone individuals, whose intention is to attack soldiers. The soldiers comprising the security patrol are represented by the three medium gray dots in the bottom part of the figure.

The `<Agents>` part of the scenario definition enumerates all the participating agent instances. Each instance declares its type and any parameters for which its agent type defaults are to be overridden. Thus we see, for example, that the agent instance with ID = 6 is of the type *Looter*, but instead of simply inheriting the default initial value of *Fear* from its agent type (30 %) it overrides it to 50 %.

The "map" in the left part of Figure 5 is in fact a greyscale reproduction of MASON's full colour 2-D interface (apart from the agent type and the shop labels which we added for clarity). In this interface we use colour to convey important information for calibration and validation teams. For example, agents dynamically change colours depending on their current strongest motive (fear, anger, etc.), thus revealing their intentions. Likewise, we use dashed coloured lines with textual labels (not shown in Figure 5) to indicate interaction between two agents, with both the colour and the label helping to distinguish peaceful negotiation from provocative gesticulation or attack. These enhancements of MASON's GUI were triggered by the feedback from our modellers who had difficulty calibrating and face validating agent models without visual cues of their motives and interactions.

All civilian agents in Vignette 2.2 are endowed with one "default" motive and a matching behaviour by which they try to satisfy it. For looters this leads to "looting" and for the violence-prone individuals to stone-pelting the soldiers. Additionally, the agents monitor what happens around them, which may excite fear or anger, in which case they start behaving fearfully (i.e. run away) or aggressively. As the patrol nears, this may induce fear in some looters who then start leaving the scene. The violence-prone individuals, however, do not get afraid but attack the patrol. The resulting violence may impact the remaining looters in two possible ways: they may either get afraid and leave, or get angry and join the attack. How many get afraid and how many get angry depends on their parameter settings, and to determine the right settings is precisely the purpose of calibration.

It was an important calibration task in Vignette 2.2 to set the parameters of the looters in such a way that the proportion of those who joined the attack to those who ran away felt "right" from the point of view of scenario validators. These parameters could be set either globally in the *Looter* agent type definition or individually in the Vignette 2.2 scenario definition. However, due to an element of uncontrolled randomness in MASON, the desired proportion of angry and fearful looters could not be achieved fully deterministically: it differed even for simulation runs with the same initial parameter settings. Here we need to clarify that our civilian agents are not scripted, so the users by adjusting their parameters do not rigidly enforce on them certain behaviours, but rather increase the probability that the desired kind of behaviour (default, fearful, aggressive) will be autonomously chosen by the agents. This is in line with the PECS modelling methodology and with our goal to create highly realistic and autonomous, yet flexible and adjustable civilian agents.

The situation was different regarding the security personnel agents. At this stage, they only possessed some very basic scripted de-escalate and self-defence behaviours. These enabled them to perform simple actions helping us assess the adequacy of our civilian models. In Vignette 2.2, for instance, we programmed the soldiers to pass by the shop being looted, but it was left to the civilians to initiate the interaction. Soldiers only reacted to them on the basis of their scripted behaviours. Alternatively, it would also have been quite easy to program the soldiers to come close to the shop and perform a series of warning shots or other actions in order to gauge the reaction of the civilians and adjust their parameters accordingly. More sophisticated behaviours for security personnel agents were to be obtained through behaviour cloning (see next section), for which calibrated civilian models are a precondition.

Overall, the calibration functionality of the EUSAS system was achieved by enhancing the MASON's Java classes and GUIs in combination with our web-based agent and scenario definition tool as well as our hierarchy of agent and scenario definition XML files. For correct loading and processing of these multiple parameter overriding options we resorted to semantic aspects of Java language, namely reflection and annotations. Parameter overriding is our key calibration tool and, behind the scenes, it also powers our data farming experiment definition and execution.

2.4 Behaviour Cloning Basics

If ABS is the core component of the EUSAS system, behaviour cloning is its core activity: it makes mission analysis possible by using mission rehearsal components to capture the operational knowledge of security experts. Their knowledge is captured in the form of behaviour patterns, which are then transferred to simulated security personnel agents as a pre-requisite to mission analysis.

The algorithm we use for behaviour cloning is called "Cognitive Multi-Agent Strategy Discovering Algorithm" (CMASDA), originally developed in the robotic soccer domain, as described in [6, 3, 2]. It was designed for advanced data analysis and extraction of behaviour patterns from low-level observations of two opposing groups of agents. In principle, CMASDA is domain independent – not in the sense that it could work without domain knowledge, but that it accepts it as input. In [19], for example, we demonstrated how CMASDA could be used in a simplified security domain.

In this article we use "behaviour cloning" as an umbrella term comprising two phases:

- behaviour analysis, in which CMASDA algorithm processes the low-level logs of human behaviour and extracts representative patterns into a cloned behaviour pattern library;
- behaviour replication, which deduces the next action for agents reproducing human behaviour by matching the patterns in the library to the current state of the simulation.

Our design of behaviour cloning was inspired by the human cognitive process [10]: reasoning in humans is based on memories and a set of assertions (rules). The whole process, including learning, is illustrated in Figure 6. First, the logs of the experts successfully playing the desired scenario in VBS2 are given to CMASDA for analysis and pattern extraction. These are denoted as "Expert Behaviour Logs" in the bottom right of Figure 6. CMASDA then processes them and extracts strategic behaviour patterns into a "Cloned Behaviour Pattern Library". These patterns can be inspected and edited by experts in a tool provided along with CMASDA. Finally, during subsequent simulations, the "CMASDA Behaviour Replicator" inside each cloned agent analyses the log of the ongoing simulation, compares it against the "Cloned Behaviour Pattern Library" and deduces the optimal next action.

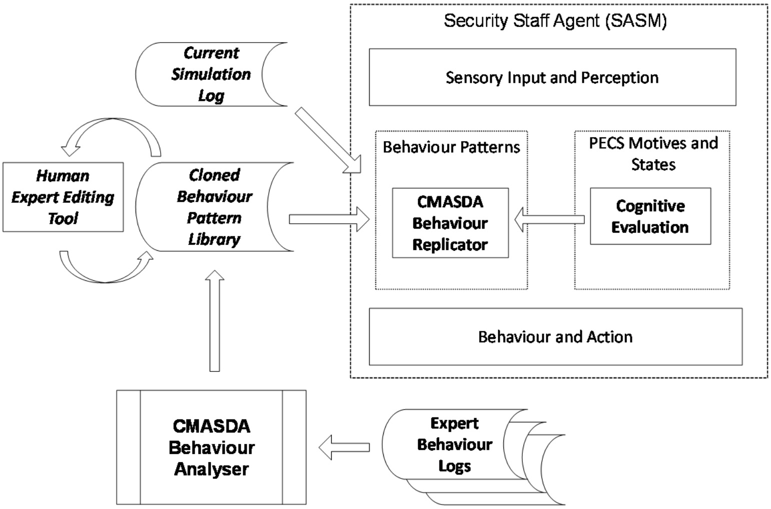
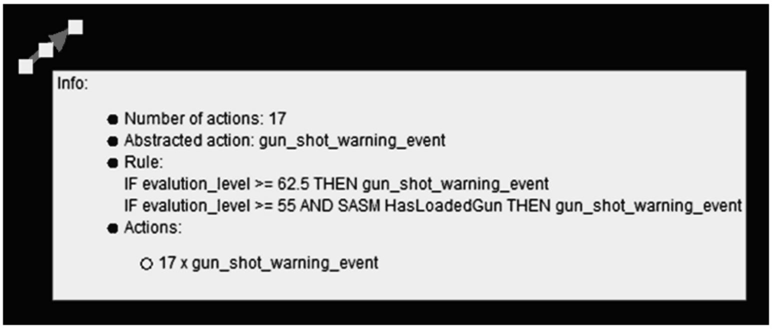

Security Staff Agent (SASM)
Sensory Input and Perception
Current Simulation Log
Behaviour Patterns
PECS Motives and States
Human Expert Editing Tool
Cloned Behaviour Pattern Library
CMASDA Behaviour Replicator
Cognitive Evaluation
Behaviour and Action
CMASDA Behaviour Analyser
Expert Behaviour Logs

Figure 6. Behaviour cloning in project EUSAS

In the remainder of this section we provide some illustrative examples of patterns extracted by CMASDA in the context of Vignette 1.1 and Vignette 2.2.

A behaviour pattern is a sequence of actions (graphical part), in which transitions from one action to the next are "guarded" by logical conditions (symbolic part). Thus, a behaviour pattern can be expressed as a sequence of `IF <condition> THEN <action>` rules, as illustrated in Figure 7. When replicating the cloned behaviour, the selection of appropriate pattern is performed in two steps. First, all the patterns in the library are matched against the action graph constructed from the current simulation data using sub-graph matching algorithms. Next, the logical conditions of each matched pattern are interpreted and evaluated. If the logical condition guarding the currently triggered action concept is satisfied, the entire pattern is stored as a candidate for execution. Among all the candidate patterns, the longest and most informative one is scheduled for execution first, followed until the next one-step sub-pattern is mismatched or a longer candidate pattern appears. This process is repeated forming a sequence of actions, effectively executing appropriate strategies until the end of simulation.



Info:

- Number of actions: 17
- Abstracted action: gun_shot_warning_event
- Rule:
  IF evalution_level >= 62.5 THEN gun_shot_warning_event
  IF evalution_level >= 55 AND SASM HasLoadedGun THEN gun_shot_warning_event
- Actions:
  - 17 x gun_shot_warning_event

Figure 7. A soldier behaviour pattern extracted by CMASDA in the context of Vignette 1.1

The example pattern in Figure 7 was abstracted in the context of Vignette 1.1 and demonstrates how machine learning approximates the rules of human behaviour. The extracted rules are perhaps not exactly as humans would have put them, but they do seem to capture correctly the two "self-defence" situations in which, according to our rules of engagement, soldiers are permitted to use a warning shot in the air: there was an aggressive action by a civilian, for which it is a standard response (the first rule in the figure), or there was a less aggressive action which was repeated (the second rule, in which the attribute `HasLoadedGun` indicates the soldier's previous reaction to some previous aggressive civilian act). The `evaluation_level` parameter in both conditional clauses refers to the perceived level of civilian aggressiveness.

The next pattern shown in Figure 8 relates to the behaviour of security personnel in Vignette 2.2. It describes the situation when the soldier moves towards the civilian who is looting the shop. The `approaching_civilian` action is not an atomic agent action, but composed of several moving events towards a civilian agent. The created rules explain when the soldier close to the shop usually approaches the civilian:

1. if the soldier is close to the looted shop and the average civilian anger is rather high;
2. the soldier is at the right side of the shop where the entry is and the civilian is looting;
3. if the soldier has been attacked close to the shop.

- `Number of actions: 213`
- `Abstracted action: moving`
- `Rule:`
  ```
  IF SASM soldier_is_close_to_shop AND average_civilian_anger > 51 THEN approaching_civilian
  IF SASM RightCorner AND CAO looting THEN approaching_civilian
  IF SASM HasBeenAttacked THEN approaching_civilian
  ```
- Actions
  ```
  87 x approaching_looting_civilian
   2 x protecting_object
   7 x moving_event
  37 x moving_event_SA
  80 x approaching_civilian
  ```

Figure 8. Symbolic part of a soldier behaviour pattern extracted by CMASDA in the context of Vignette 2.2

The patterns that can be discovered by CMASDA are not limited to those of security personnel. The example shown in Figure 9 describes the situation when a civilian agent of the type *Looter* starts looting (atomic `take_event` action). Looting usually starts close to the shop when his internal looting motive is higher than anger or fear and his energy level is high. This properly describes the behaviour provided to the simulated civilian agents, thus confirming the effectiveness of the CMASDA algorithm. Since the behaviour of civilian agents is known beforehand and stored in simulation models, these patterns were excluded from the pattern library.

- `Number of actions: 76`
- `Abstracted action: take_event`
- `Rule:`
  ```
  IF anger_level > 24 AND CAO LootingMotiveHigherThanAnger AND fear_level > 25 THEN take_event
  IF CAO LootingMotiveHigherThanFear AND energy_level = 99 THEN take_event
  IF CAO LootingMotiveHigherThanFear AND fear_level > 26 AND CAO looting THEN take_event
  ```
- Actions
  ```
  76 x take_event
  ```

Figure 9. Symbolic part of a civilian behaviour pattern extracted by CMASDA in the context of Vignette 2.2

The success of behaviour cloning in capturing the security expert strategies (where by a strategy we mean a collection of behaviour patterns along with logical conditions guarding their execution) depends primarily on three factors. The first refers to the `<action>` part of the behaviour rules: the 3-D visualization environment (VBS2) must provide a sufficiently rich repertoire of actions required for the execution of a successful strategy. The second refers to the `<condition>` part of the rules: the logs of successful games must faithfully reproduce (or reasonably approximate) all the important factors that experts take into account when deciding what to do in a given scenario. Thirdly, machine learning algorithms need a sufficient number of repetitions of each behaviour pattern in order to infer the rules correctly. Without these preconditions it is extremely unlikely that machine learning algorithms would extract useful rules.

2.5 Data Farming and Data Analysis

Once the simulated security personnel agents are equipped with expert strategies through behaviour cloning, it is possible to verify the robustness of those strategies through data farming and data analysis.

Data farming may be thought of as a kind of ANOVA (analysis of variance), during which a number of simulations run in parallel, each with slightly different input parameters [12]. The success or failure of the analyzed strategy is determined by referring to various measures of effectiveness, such as the number of injured soldiers and civilians, etc. In this way, data farming helps to determine the subset of all possible input vectors in which the analyzed strategy can be considered adequate. This evaluation is performed by various data analysis tools.

Both data farming and data analysis are typically performed on a high-performance computing infrastructure that can be of several types, as illustrated in Figure 10.

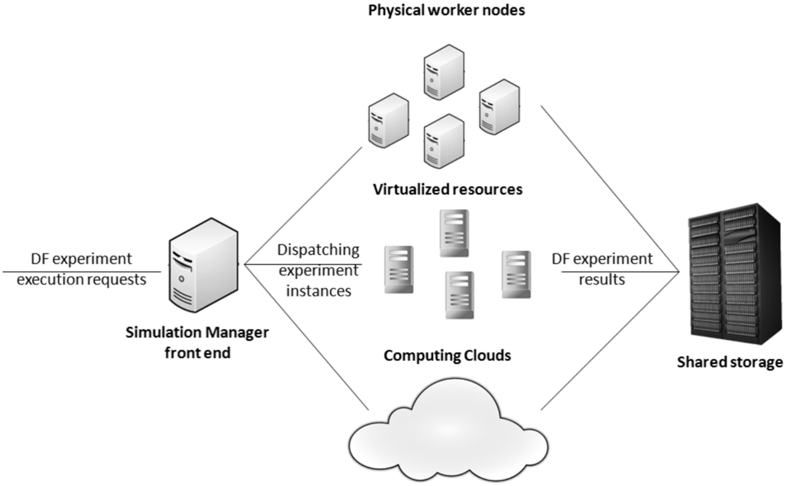


Figure 10. An overview of data farming infrastructure

Data farming experiments are defined through user interfaces that currently support three methods of experiment design: 2^k, full factorial, and Near Orthogonal Latin Hypercube. Our flexible and scalable management of infrastructure permits adding new computing resources to the experiment even while the experiment is running. The results of experiments are typically analyzed first by statistical techniques, such as the regression tree shown in Figure 11.

This particular regression tree has been constructed for MoE (measure of effectiveness) "Global Anger" in Vignette 1.1 and lends further support to the validity of our model for the emergence of collective aggression described in [9]. Each branch of

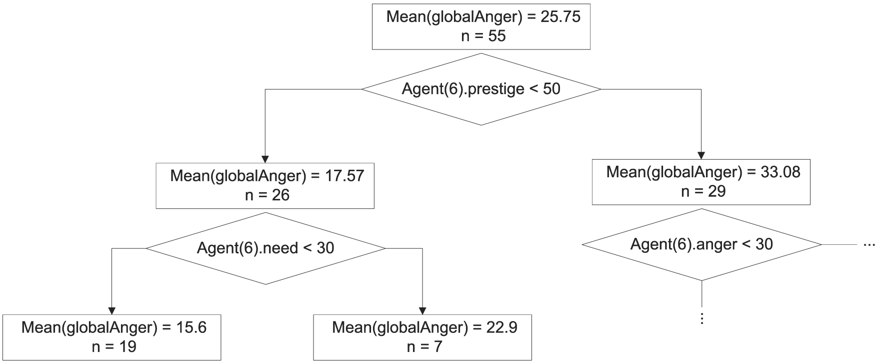


Figure 11. Part of a regression tree for "Global Anger" in a Vignette 1.1 data farming experiment

the tree represents a group of simulation runs: the parameter `n` gives their number and the parameter `mean` the mean value of the chosen measure of effectiveness. At the top of the tree we start with a group of 55 simulations which sample a subset of the overall input vector space. The mean value of "Global Anger" for all the simulation runs was 25.75. Our goal in constructing the regression tree is to explain the variance of "Global Anger" by the variance of input parameters. At the root of the tree, which represents the most significant factor explaining the greatest part of the observed variance, we see the prestige of agent No. 6. This agent happens to be the group leader and his prestige is a measure of his social influence over the members of his group, i.e. their tendency to adjust their emotional mood to his mood. Therefore, a logical condition involving his prestige `Agent(6).prestige < 50` was able to split the simulations into two groups with the largest possible difference in the mean value of Global Anger.

Another regression tree is shown in Figure 12. It was constructed for MoE "Global Escalation" (number of aggressive acts, such as stone throws) in Vignette 2.2. At first sight it might look rather trivial, but in reality it conveys a surprising and counterintuitive result. While the overall aggressiveness of the crowd did indeed increase with the number of aggressive individuals (as expected), it did not do so smoothly but in a highly nonlinear fashion. There appears to be a certain critical number of aggressive individuals (25.5), and the scenarios below this threshold exhibited very low levels of aggression (about 6 aggressive acts on average), while the scenarios above it exhibited very high levels (hundreds of aggressive acts). This seems to indicate that the tactics used by the simulated security personnel could effectively manage looting crowds with up to 25 aggressive individuals.

Besides regression trees, there are other statistical tools available, e.g. histograms[6], but they all treat one simulation run as one data point associating one input vector with its corresponding output value. In order to investigate what has

[6] check `www.scalarm.com` with Massively Self-Scalable Platform for Data Farming

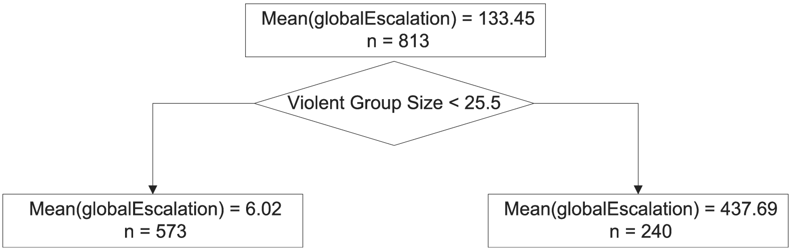

Mean(globalEscalation) = 133.45
n = 813
Violent Group Size < 25.5
Mean(globalEscalation) = 6.02
n = 573
Mean(globalEscalation) = 437.69
n = 240

Figure 12. Regression tree for "Global Escalation" (number of aggressive acts) in a Vignette 2.2 data farming experiment

happened *inside* a particular simulation run other tools are needed. We shall briefly mention three that we explored or prototyped in the EUSAS project: CMASDA Pattern Viewer, Graph Inference Tool and causal partitioning.

CMASDA Pattern Viewer is able to re-play the simulation scenario from logs and visually match the identified behaviour patterns on the screen against a pattern library. The colouring of matched patterns indicates whether the pattern was marked in the library as a success or a failure.

Graph Inference Tool is meant for open-ended exploration of simulation logs on the basis of spreading activation algorithm. It can operate in two modes: there is a text-based search interface, where the user first specifies the starting-point of the search (nodes that receive initial activation), and then gets the result list (related nodes). If the user wishes, it is possible to explore the result list interactively in a graphical form. This makes sense because the result list is typically just a small sub-graph of the complete graph constructed from the full simulation log. It requires certain expertise but it can lead to unexpected insights – it helped us, for example, to discover errors in our initial implementation of the soldiers' self-defence behaviour. A more detailed description of this tool can be found in [19].

Causal partitioning is a technique based on nonlinear structural causal analysis. It is applied to model variables, such as simulated emotions *anger* or *fear*, in order to quantify the proportion in which various influencing factors (their "causes") contributed to their actual numerical values. Interested readers may refer to [14] for a more detailed account.

Overall, data analysis tools verify the robustness of the cloned strategies and help to identify the reasons behind particular results of interest (especially outliers, i.e. simulation runs with extreme output values). Only when a particular strategy proves to be sufficiently robust in the context of the intended training situations, it should be used for mission rehearsal.

2.6 Mission Rehearsal

The EUSAS system provides mission rehearsal capability in virtual training mode, in which real people (security staff) operate simulated equipment and try to prevail over simulated opponents in a simulated world. Our virtual training setup involves a well-known third-party product VBS2[7] coupled with our ABS through a CORBA-based interface, which is also a well-known technology. Therefore we focus here on how a sequence of vignettes (micro-scenarios) might be logically combined for the purposes of mission rehearsal.

The starting point is Vignette 1.1, where a tussle develops in front of the pedestrian entrance to the military base. Depending on how well the trainees manage this tussle, either Vignette 2.1 or 2.2 is assigned to them, but *without* their knowing which one. This would reflect real life: if there was a lot of resentment generated among the civilians in the first vignette, later when the soldiers encounter the same civilians again during patrol, it might lead to an increased level of aggressiveness, or even to an attempt to trap the soldiers, as in Vignette 2.2. So the trainees have to recognize the kind of situation they are facing (is the looting of the shop genuine, or is it just a trap?) on the basis of indirect symptoms, and then react accordingly. This principle can be generalized: mission rehearsal consists of a sequence or a "mosaic" of vignettes, where the results achieved in earlier vignettes influence the type as well as the initial settings of the subsequent ones.

Validation activities and results for mission rehearsal are reported in Section 3.3.

3 EXPERIMENTATION AND VALIDATION

The EUSAS project employed an iterative approach in which design and development were interwoven with verification and validation phases. The project consisted of two iterations producing two versions of the EUSAS system (V1 and V2), but we started with a "zero" iteration where the purpose was to choose a suitable simulation platform and tentatively propose a generic and modular agent architecture that would accommodate the project needs. We have sketched this architecture and its salient features in Sections 2.2 and 2.3. In the next section we briefly reiterate our early experimentation and validation results, most of which were already published. In Section 3.2 we present, for the first time, comprehensive verification and validation results for our behaviour cloning approach. Section 3.3 describes the final project demonstration of the EUSAS system, which took place in the premises of the project partner Cassidian (now Airbus Defence and Space) in Elancourt, France.

3.1 Early Experimentation and Validation Results

At the start of the project we considered various multi-agent simulation platforms on the basis of information gleaned from published surveys. Two most promising candidates (NetLogo and MASON) were then compared by means of a practical implementation experiment that we reported in [15]. The evaluation criteria covered environment representation (e.g. support for GIS – geographic information system data), means to create, represent and implement agents and their behaviours,

[7] `https://www.bisimulations.com/`

as well as support for logging, model check-pointing, physical movement (flocking, steering, etc.) and agent standards (e.g. HLA[8] or FIPA[9]). Additionally, data farming required flexible parameterization and visualisation that could be switched off. Finally, we also considered development environments, in-built analytical tools and performance. We subsequently implemented the first version of the EUSAS system (V1) in the winning simulation platform (MASON), and described some of its features and early validation results in [9]. The first significant result was the successful (albeit partial) face validation of our computational model for the emergence of collective aggression in the context of Vignette 1.1. This also validated our choice of the simulation platform and our agent architecture: since we were able to implement the requirements of V1 (and later also of V2), it proved to be the right choice and the right design.

As the emotional dynamics of our agents was partly driven by differential equations, the need to solve them numerically led us naturally to the discrete time step (DTS) approach. We retained DTS in spite of the fact that in mission rehearsal our ABS is coupled with VBS2, which is essentially thread-based and event-based, and both have to interoperate in real time. Regarding the potential disadvantages of DTS raised by [1], such as the pressure for small time steps or the need for tie-breaking mechanisms, we have found these relevant, but not blocking. When our ABS interoperates with VBS2 in real time, we ensure that ABS is called at least twice per second, which is feasible given the kind of micro-scenarios that we deal with in the EUSAS project. We also slightly randomize the duration of agent actions so as to limit the number of simultaneous state transitions (such as emitting events to the connector at the end of each action) that occur during one time step. Thanks to the cumulative way in which external events are processed in our models, we have not felt any need for special tie-breaking mechanisms beyond a relatively simple provision for a consistent update of state variables independent of the order in which they are called for update. Moreover, our later experiments reported in [14] indicated that varying the time step did not have a significant impact on the emergent behaviour of the system or the simulation outcome.

On the other hand, we did feel a distinct need to add new visual features to MASON's 2-D interface for the purposes of face validation of our models, notably the dynamic colouring of agents depending on their leading motive and the visualization of their interactions. In this respect we would like to emphasize that MASON's neat structure greatly simplified our task.

3.2 Verification and Validation of Behaviour Cloning

Besides the partial validation of our model for the emergence of collective aggression reported in [9], probably the most impressive early result was the behaviour cloning success in the context of Vignette 1.1. Figure 13 shows how closely CMASDA

[8] `http://en.wikipedia.org/wiki/High_Level_Architecture_(simulation)`

[9] `http://en.wikipedia.org/wiki/FIPA`

was able to match the simulation statistics of the game that it was given to clone. A closer examination reveals that the two sets of time-series are indeed *not* identical, yet they are remarkably similar. The similarity was also evident when the project team watched, side by side, the video recordings of the original simulation and that of its cloned replica. Of course, there are also objective measures, such as those designed on the basis of dynamic time-warping (DTW), and our behaviour cloning group was using them as well.

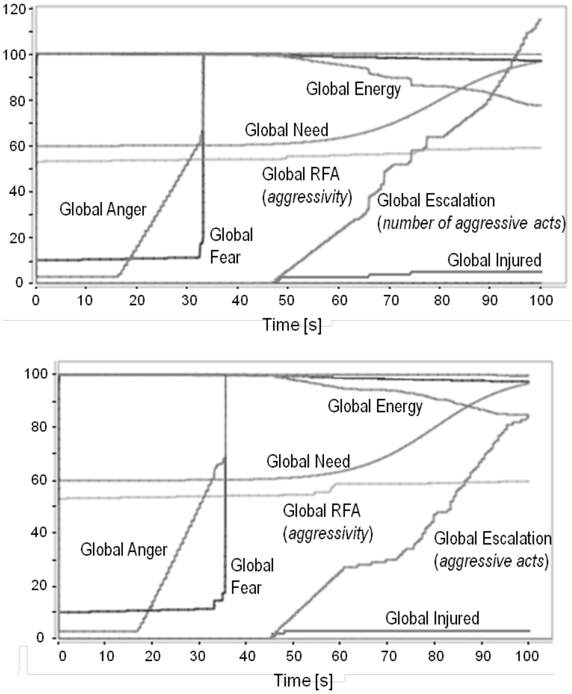


Figure 13. Comparison of simulation variables and statistics for one simulation run of Vignette 1.1 (top) with its cloned reproduction by CMASDA (bottom)

Dynamic Time Warping (DTW) [7] is a well-known distance measure technique that performs a mapping between two time series by minimizing the distance between them. The cumulative differences between the two series give a measure of their similarity. The usability of this method derives from observations that humans can compare time series that may vary in time or speed. DTW compares time series in a flexible way and returns a real value from the interval [0,1], where 1 is returned when comparing two identical and 0 when comparing two completely different sequences. In general, two time series can be deemed similar if their DTW similarity measure is higher than 0.5. In this experiment we used a variation of the algorithms that aligns multiple time series at the same time.

For our experiments, 10 simulation logs were obtained for each of the following simulation modes:

- *constructive mode,* in which the soldiers were simulated by agents using simple scripted de-escalation and self-defence behaviours;
- *virtual mode,* in which the soldiers were played by real human beings;
- *cloned constructive mode,* in which the soldiers were simulated by agents using cloned replicas of the simple de-escalate and self-defence behaviours from the constructive mode;
- *cloned virtual mode,* in which the soldiers were simulated by agents using cloned replicas of the strategies used by real humans in virtual mode.

One DTW comparison was performed by comparing each simulation log of one type with all logs of a different type. At the same time, one simulation log contains a number of time series (graphs) as shown in Figure 13. To compute the similarity of different simulation modes, we have collected a representative subset of simulation variables and statistics including Global Anger, Global Fear, Global Energy, Global Looting Motive, Number Of Effective Shots, Global Injured, Global Killed. These variables characterise the time evolution of simulation and agent parameters throughout the scenario. As an example, the comparison of the simulation modes shown in Figure 13 (*constructive mode* versus *cloned constructive mode*) yielded 0.591 in the third column of the first row of Table 1.

	constructive	virtual	cloned constructive	cloned virtual
constructive	0.714	0.264	0.591	0.297
virtual		0.632	0.391	0.586
cloned constructive			0.788	0.382
cloned virtual				0.851

Table 1. Comparison of similarities between Vignette 1.1 simulation modes

The first row in Table 1 shows the results of comparing the original Vignette 1.1 simulations (*constructive mode*) to all the modes including itself. The result of around 0.6 might appear low, but one should take a look at how similar are the *constructive mode* simulation logs when compared to themselves: the computed value of 0.714 in the first column represents the upper bound of what the cloning can possibly achieve; it would only be reached if the cloning perfectly replicated the original logs. The second number in the first row compares the *constructive* to the *virtual mode.* As can be seen, the two modes are quite different since their similarity measure is low (0.264), meaning that humans played the scenario differently than the computer. The third number in the row, as mentioned, shows the cloning performance (0.591), which is close to the upper bound (0.714). This confirms that cloning can credibly reproduce the simulated soldier behaviour in Vignette 1.1. The low value of the last comparison in the first row is another important indication, showing that the *cloned virtual mode* still differs significantly from the *constructive mode,* as it should, since the *virtual mode* itself differed substantially from it. Therefore, we can conclude that CMASDA is able to discover and clone different behaviours within the same domain.

The second row of Table 1 compares the *virtual* and two cloned modes. Again, the first number (0.632) is the upper bound for similarity that cloning can achieve. The tactics used by humans in the *virtual mode* contain more internal variance than the simulations in *constructive mode,* since the computed similarity of *virtual mode* to itself is lower (0.632). The second number in this row confirms that the *cloned constructive mode* differs from the *virtual* one. The last measure shows how authentically CMASDA was able to extract the behaviour patterns of real human players. Its value (0.586) is very close to the upper bound, which again confirms that CMASDA can successfully discover different behaviours in the Vignette 1.1 domain.

The third row compares the *cloned constructive* and the *cloned virtual* modes. While the *cloned constructive mode* is more similar to itself (0.788) than the *constructive mode* (0.714), the difference of 0.074 is reasonable. As we would expect, the *cloned constructive* and the *cloned virtual* modes are quite dissimilar, as shown in the last column of the third row. Finally, the last value in the fourth row means that the simulations in the *cloned virtual mode* are internally more similar to each other than their originals (i.e. tactics exhibited by real humans in the *virtual mode*), indicating a space for further improvement.

Simulator adaptors and cloning modules were also developed for the Vignette 2.2 domain. As with Vignette 1.1, behaviour patterns discovered by CMASDA credibly reproduced the behaviours exhibited in different simulation modes in Vignette 2.2. Moreover, the patterns were rich enough to capture and then execute different styles of plays. The performance of cloning on Vignette 2.2 is presented in Table 2.

	constructive	virtual	cloned constructive	cloned virtual
constructive	0.647	0.313	0.439	0.371
virtual		0.586	0.298	0.509
cloned constructive			0.523	0.351
cloned virtual				0.656

Table 2. Comparison of similarities between Vignette 2.2 simulation modes

Once it is determined that cloned agents credibly reproduce the behaviour of real humans (security personnel), the next step would be to use these agent clones in the data farming set-up. The idea is to test their behaviour on a representative sample of permissible parameter settings in order to identify the most problematic ones. The logs of such scenarios could then be further analyzed with the data analysis tools presented in Section 2.5, with key findings taken into consideration during subsequent rounds of mission rehearsal.

3.3 Final Project Demonstration of the EUSAS System

The final project demonstration of the EUSAS system (V2) consisted of three parts:

- Virtual training demonstration;
- Behaviour cloning demonstration;
- Data farming demonstration.

Since we have already covered behaviour cloning in Sections 2.4 and 3.2 and data farming in Section 2.5, we shall focus on virtual training here.

Figure 14 shows a virtual training session in progress, in which four members of the French Army (Armée de Terre – Troupes de Marine – RICM) tested the capabilities of the EUSAS system. The team included one squad leader (standing) who commanded three squad members (sitting).

The computing infrastructure consisted of three computers with 64-bit OS "Windows 7 Professional" running on Intel® Xeon® CPU 3.20 GHz (quadcore) with 8 GB RAM, a graphic card "Nvidia Quadro FX 3800" 1 GB, HDD 250 GB, an Ethernet network card, and the installed serious game VBS2 VTK v1.4 (Virtual Battlespace 2 – Virtual Kit Training) of Bohemia Interactive Simulations[10].

One computer was used as a server running both ABS and VBS2 in the administrator mode, which allowed the user to choose the mission, to launch it and also to play a character. The server was equipped with two screens: the first one showed the 3-D view generated by VBS2 and the second one the 2-D view generated by ABS. Real-time synchronisation between ABS and VBS2 was realised through a CORBA-based interface. The two remaining computers were connected to the server as VBS2 clients; each was equipped with one screen and permitted the participation of one soldier.

In Figure 14, the squad leader is looking at a bird's eye view of the training situation projected for him on the wall (this was in fact the 2-D view generated by ABS – see Figure 5 in Section 2.3). Meanwhile, the squad members are executing his orders on VBS2 consoles providing simulated 3-D views shown in Figure 15. Labels and icons above the heads of displayed characters and avatars in Figure 15 convey the meaning of their culture-specific gestures, actions and verbal utterances. For simulated civilians, they also indicate their internal motivations. In this way we compensated for the inability of VBS2 to render emotions through facial expressions.

Within 80 minutes allocated for the virtual training, eleven training runs were performed: six on Vignette 1.1 (turmoil in front of the pedestrian entrance) and five on Vignette 2.2 (crowd looting a shop). The repetition was required in order that CMASDA could later analyse the logs and extract useful behaviour patterns as outlined in Section 2.4. Moreover, two different levels of civilian behaviour had been implemented for each vignette: a normal mode (with default parameters) and

[10] `https://www.bisimulations.com/`

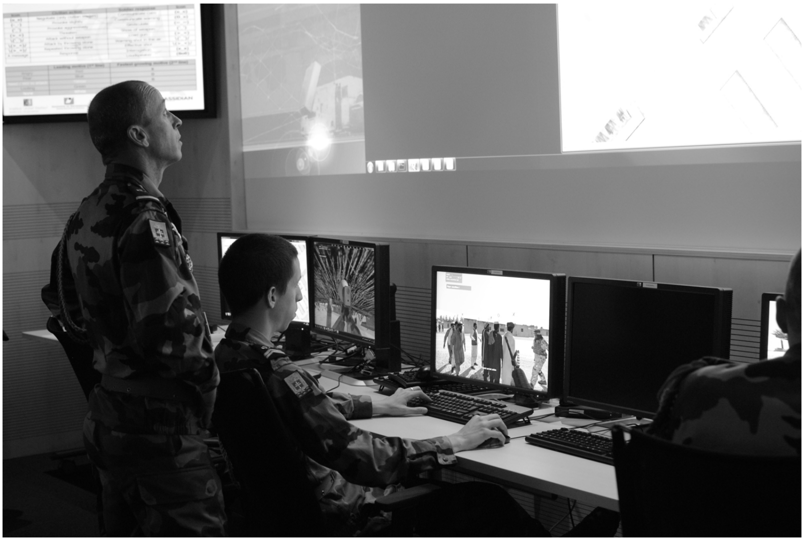


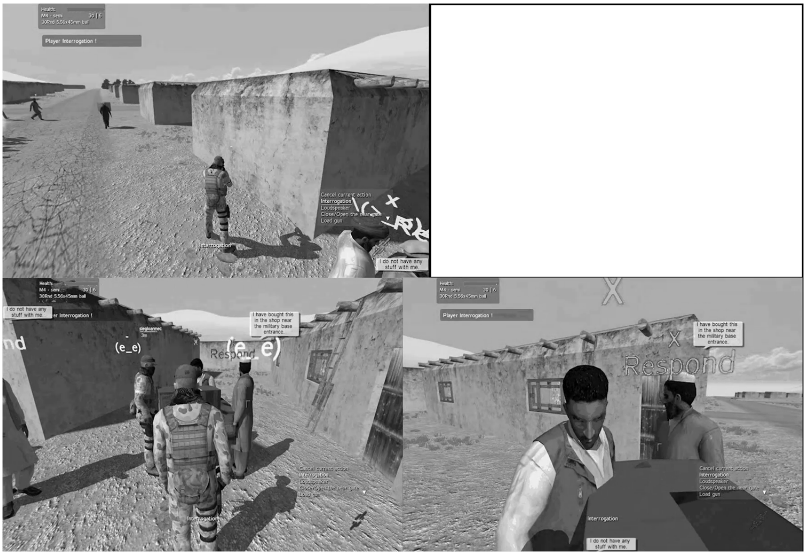
Figure 14. Virtual training session in progress

a mode where the civilians were more aggressive and thus more difficult to deal with.

Apart from a minor technical problem at the beginning (loss of synchronisation between VBS2 instances), the virtual training demonstration proceeded smoothly and according to plan. At the end, the trainees and their leader were asked to fill in a questionnaire with 28 questions assessing the system in the following areas:

- VBS2 Interface and the EUSAS system HMI (human-machine interface);
- The EUSAS System Training and Analysis approach;
- Benefits for Virtual Training;
- Benefits for Mission Preparation.

Each question required an answer rating the system on a scale from 1 ("very little", minimum appreciation) to 7 ("very much", maximum appreciation). Empty space for textual comments was also provided. The overall rating of the system, calculated as the mean of four area ratings, was 5.78. The system performed particularly well in the areas of benefits for mission preparation (rating 6.13) and benefits for training (rating 5.79) while a comparatively lower rating for the VBS2 interface (5.45) indicates a space for further improvement. This operational feedback was highly valuable for potential future applications of the EUSAS system.



Player Interrogation !

I do not have any stuff with me.

I have bought this in the shop near the military base entrance.

Figure 15. Simulated 3-D views generated by VBS2 for Vignette 2.2 scenario

4 CONCLUSION AND FUTURE WORK

The EUSAS system was accepted by the European Defence Agency as meeting the contractual requirements. Its overall technology readiness level[11] was estimated at TRL 4-5 (technology validated/demonstrated in relevant environment). This shows that integrating mission analysis with mission rehearsal through behaviour cloning can indeed provide significant benefits, such as the ability to capture the security expert knowledge in a non-verbal way and to guarantee the consistency of mission rehearsal with mission analysis. Moreover, the use of the PECS reference model for realistic modelling of civilian agents helped to provide a modern virtual training environment that adequately reflects the importance of human factors for security operations in an urban terrain. Finally, the breakdown of large scenarios into smaller units (vignettes) facilitates the formulation of simple and dynamic rules of engagement tailored to specific situations. This is further assisted by behaviour cloning which captures the security expert strategies in the form of simple and straightforward rules that help to focus the attention of trainees on key decision-making factors.

The wide spectrum of technologies used in the EUSAS project opens up multiple directions for future work. First of all, the system might be further improved as a whole. Here the possibilities include, for example, the introduction of playable characters on the civilian side (at present all civilians are simulated), the improvement of civilian models through behaviour cloning (only security personnel agents

[11] https://en.wikipedia.org/wiki/Technology_readiness_level

benefit from it now), or support for squad leader training (at present virtual training is only for squad members). Secondly, individual components of the system might be improved or adapted for new domains. Data farming components, for example, might be enhanced with a semi-automatic expansion of the parameter space and self-scaling, since both have to be done manually now. They could also be generalised so as to work with any kind of simulation that provides a formalized description of its required inputs and outputs, or to become deployable on new types of computing infrastructure (grids with different middleware, different types of computing clouds).

Along similar lines, our behaviour cloning approach could be improved so as to reduce the involvement of human experts in the behaviour analysis phase. This would mean, first, that the behaviour analyser would automatically choose appropriate parameters for pattern extraction, such as the level of abstraction of the abstract action graph from which the patterns are mined. Next, it would have to identify the subset of discovered patterns which best represent the original human behaviour. Last, the chosen subset of patterns might have to be further refined before getting exported to the behaviour library. This automation could reduce the time needed to construct the behaviour library from low-level log data to a matter of minutes.

Finally, our agent-based simulator ABS could also be improved along several lines. One line, purely technical, would involve low-level coding in order to remove known deficiencies, e.g. the element of uncontrolled randomness that we mentioned in Section 2.3. Another line, more abstract, could unify our treatment of simulation parameters: at present there are three types of parameters (component-related, scenario-related, simulator-related) which are not equally accessible for data farming. These two lines of advance would make our ABS more versatile as a general-purpose platform for agent-based simulation. But there is also a third line concerning its use as a platform to develop, study and evaluate behavioural models. This would require specialized data analysis tools, such as those mentioned in Section 2.5. Among these, we are working intensively on causal partitioning, which we consider the most promising: the first part of our work was already published in [14] and a sequel is under preparation.

Acknowledgments

This work was supported by the EDA project A-0938-RT-GC EUSAS, project VEGA No. 2/0054/12 and by the Slovak Research and Development Agency under the contract No. APVV-0809-11.

REFERENCES

[1] Al Rowaei, A. A.—Buss, A. H.—Lieberman, S.: The Effects of Time Advance Mechanism on Simple Agent Behaviors in Combat Simulations. Proceedings of the 2011 Winter Simulation Conference (WSC), December 2011, pp. 2426–2437.

[2] Bezek, A.: Discovering Strategic Multi-Agent Behavior in a Robotic Soccer Domain. Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '05), ACM, July 2005, pp. 1177–1178.

[3] Bezek, A.—Gams, M.—Bratko, I.: Multi-Agent Strategic Modeling in a Robotic Soccer Domain. Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, ACM, May 2006, pp. 457–464.

[4] Chiarini, G. M.: Urban Warfare in Crisis Response Operations. RUSI Defence Systems, Vol. 8, 2005, No. 3, Winter/Spring, pp. 89–92. Available on: `www.rusi.org/downloads/assets/Chiarini.pdf`.

[5] Collyer, R. S.: Human Performance Issues in Urban Military Operations. Defence Science Technology Organisation, Edinburgh, South Australia, 2003. Available on: `http://hdl.handle.net/1947/3845`.

[6] Gams, M.—Bezek, A.: Multi-Agent Strategic Modeling in a Specific Environment. In: Nakashima, H., Aghajan, H., Augusto, J. C. (Eds.): Handbook of Ambient Intelligence and Smart Environments. Springer Science + Business Media, LLC, New York, 2010, pp. 731–750.

[7] Giorgino, T.: Computing and Visualizing Dynamic Time Warping Alignments in R: The dtw Package. Journal of Statistical Software, Vol. 31, 2009, No. 7, pp. 1–24.

[8] Hills, A.: Can We Fight in Cities? The RUSI Journal, Vol. 146, 2001, No. 5, pp. 6–10.

[9] Hluchý, L.—Kvassay, M.—Dlugolinský, Š.—Schneider, B.—Bracker, H.—Kryza, B.—Kitowski, J.: Towards More Realistic Human Behaviour Simulation: Modelling Concept, Deriving Ontology and Semantic Framework. Applied Computational Intelligence in Engineering and Information Technology, Topics in Intelligent Engineering and Informatics. Springer Berlin Heidelberg, Vol. 1, 2012, pp. 1–17.

[10] Johnson-Laird, P. N.: Mental Models and Human Reasoning. Proceedings of the National Academy of Sciences, Vol. 107, 2010, No. 43, pp. 18243–18250.

[11] Korte, W.: Challenges and Implications of Urban Operations: A German Army Perspective. RUSI Defence Systems, Vol. 8, 2005, No. 3, Winter/Spring, pp. 99–101. Available on: `www.rusi.org/downloads/assets/Korte.pdf`.

[12] Kryza, B.—Król, D.—Wrzeszcz, M.—Dutka, Ł.—Kitowski, J.: Interactive Cloud Data Farming Environment for Military Mission Planning Support. Computer Science, Vol. 13, 2012, No. 3, pp. 89–100.

[13] Kvassay, M.—Hluchý, L.—Dlugolinský, Š—Laclavík, M.—Schneider, B.—Bracker, H.—Tavčar, A.—Gams, M.—Król, D.—Wrzeszcz, M.—Kitowski, J.: An Integrated Approach to Mission Analysis and Mission Rehearsal. Proceedings of the Winter Simulation Conference, Berlin, December 2012, p. 362.

[14] Kvassay, M.—Hluchý, L.—Krammer, P.—Schneider, B.: Causal Analysis of the Emergent Behavior of a Hybrid Dynamical System. Acta Polytechnica Hungarica, Vol. 11, 2014, No. 4, pp. 21–40.

[15] Laclavík, M.—Dlugolinský, Š.—Šeleng, M.—Kvassay, M.—Schneider, B.—Bracker, H.— Wrzeszcz, M.—Kitowski, J.—Hluchý, L.: Agent-

Based Simulation Platform Evaluation in the Context of Human Behavior Modeling. Advanced Agent Technology, Springer, Berlin Heidelberg, 2012, pp. 396–410.

[16] McDermott, P. L.—Battaglia, D. A.—Phillips, J.—Thordsen, M. L.: Military Operations in Urban Terrain (MOUT): Decision Making in Action. U.S. Army Research Institute for the Behavioral and Social Sciences, Alexandria, VA, 2001. Available on: `http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA391474`.

[17] Page, E. H.—Smith, R.: Introduction to Military Training Simulation: A Guide for Discrete Event Simulationists. Proceedings of the 30th Conference on Winter Simulation, IEEE Computer Society Press, December 1998, pp. 53–60.

[18] Schmidt, B.: Modelling of Human Behaviour: The PECS Reference Model. In: Verbraeck, A., Krug, W. (Eds.): Simulation in Industry – 14th European Simulation Symposium 2002, SCS European Publishing House, Dresden, Germany, 2002.

[19] Tavčar, A.—Gams, M.—Kvassay, M.—Laclavík, M.—Hluchý, L.—Schneider, B.—Bracker, H.: Graph-Based Analysis of Data from Human Behaviour Simulations. IEEE 10th International Symposium on Applied Machine Intelligence and Informatics (SAMI), January 2012, pp. 421–426.

[20] Urban, C.: PECS – A Reference Model for the Simulation of Multiagent Systems. In: Ramzi, S., Klaus, G. T., Gilbert, N. (Eds.): Tools and Techniques for Social Science Simulation. Physica-Verlag, Heidelberg, New York, 2000.

[21] Worley, D. R.—Wahlman, A.—Gleeson Jr., D. J.: Military Operations in Urban Terrain: A Survey of Journal Articles. IDA Document D-2521, Institute for Defense Analyses, Alexandria, Virginia, 2000.

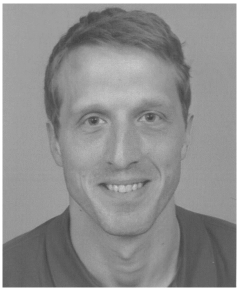
Marcel Kvassay is a researcher and a Ph.D. candidate at the Institute of Informatics of the Slovak Academy of Sciences. He graduated from the Faculty of Electrical Engineering and Information Technology of the Slovak University of Technology in Bratislava in 1991. Prior to joining the Institute in 2009 he worked at various positions as a software engineer, software design coach and software process improvement manager. His research interests include causal analysis, social networks, knowledge-based technologies and multi-agent systems.

Ladislav Hluchý is the Head of the Department of Parallel and Distributed Computing at the Institute of Informatics of the Slovak Academy of Sciences. He received his M.Sc. and Ph.D. degrees, both in computer science. He is R & D Project Manager and Work-Package Leader in a number of 4FP, 5FP, 6FP and 7FP projects, as well as in Slovak national R & D projects.

Štefan Dlugolinský is a researcher in the Parallel and Distributed Computing Department at the Institute of Informatics, Slovak Academy of Sciences. His topics of interest include large-scale parallel and distributed text processing, information extraction and information retrieval. He is the author and co-author of about 40 scientific papers. He was/is also involved in several international (FP7, EDA) and national projects.

Bernhard Schneider graduated in computer science and holds a doctoral degree in human behaviour modelling, received from the University of the German Armed Forces, Munich. Before taking over his current role of an R & T Processes Manager in the Airbus Defence and Space division, he acted as Technology Domain Manager in the Airbus Defence and Space technology management organization, responsible for coordinating security technology related research activities.

Holger Bracker graduated in electrical engineering at the University of Technology in Munich and holds Ph.D. in Signal processing from Universite de Rennes I (France). He works as a research engineer for Airbus Defence and Space in Ottobrunn near Munich. He is managing and contributing to various research projects on national and European level in the fields of civil security, crisis management and systems engineering. His research interests are agent based simulations, data analytics and model based systems engineering.



Aleš Tavčar is an assistant researcher at the Jožef Stefan Institute at the Department of Intelligent Systems and Ph.D. candidate at the Jožef Stefan International Postgraduate School. He is the head of the Agents group at the department. His main field of research covers agent technologies, human behaviour analysis using advanced machine learning techniques and the development of agent-based solutions for smart houses and cities.

Matjaž Gams is Head of Department of Intelligent Systems at the Jožef Stefan Institute and Professor of Computer Science at the University of Ljubljana and MPS, Slovenia. He is or was teaching at 10 faculties in Slovenia and Germany. His professional interest includes intelligent systems, artificial intelligence, cognitive science, intelligent agents, business intelligence and information society. He is member of numerous international program committees of scientific meetings, national and European strategic boards and institutions, editorial boards of 11 journals and he is managing director of the Informatica journal. He has been president of the Organizing Committee of the central IS conference for 18 years `http://is.ijs.si/` delivering the prestigious Michie-Turing award for life achievements. He was co-founder of various societies in Slovenia, e.g. the Engineering Academy where he currently coordinates the research class, AI Society, Cognitive Society, ACM Slovenia (currently president), SLAIS. He cooperated/headed around two hundred national and international projects including top EU projects FP and now H2020. His bibliography list includes over 100 original scientific publications and over 1 100 items in all categories including 7 patent applications.

Marc Contat has joined Airbus Defence and Space in 2003. After graduating in computer science from engineering school, he achieved his Ph.D. degree in 2002 in the French Aerospace Lab ONERA ("Resource Allocation in Multi-Sensor Systems for Classification and Recognition (NCTR)") and published various articles in scientific conferences. Before 2007, he was involved in battlefield surveillance projects for French MoD. Since 2007, he has been managing Airbus Defence and Space contributions or coordinating consortiums in European or French R & T/R & D projects relating to multi-sensor data fusion, WSN, virtual training in urban environment, border and maritime surveillance, critical infrastructures protection, video-surveillance and media mining.

Łukasz Dutka has a significant expertise in grid systems, large-scale systems, development of application for business purposes, team and project management in commercial projects as well as EU IST projects. He obtained his M.Sc. in computer science from the Jagiellonian University, Poland and his Ph.D. in computer science from the University of Science and Technology, Cracow, Poland. He has longstanding experience with managing large development teams in commercial software companies. His scientific interests include large-scale computer systems, system architectures, component approaches as well as exploiting computer technologies of the future for today's solutions. He is the author of a modern software development architecture called the Component-Expert Architecture combining expert systems with component architectures, with successful applications in commercial and scientific environments. He has actively participated in several EU IST projects including CrossGrid and K-WfGrid and since 2008, he is a Technical Director of PL-GRID project.

Dariusz Król is a research assistant at University of Science and Technology in Krakow, Poland, where he earned M.Sc. (in 2009) and Ph.D. (in 2014) degrees in computer science, and a specialist in computer science at the Academic Computer Centre CYFRONET AGH. He is the author or co-author of about 40 scientific papers. He has been involved in many national and international projects, funded by European Commission and EDA, e.g. ViroLab, GREDIA, PL-Grid, EDA EUSAS, PaaSage, VirtRoll. His topics of interest include autonomic computing, large-scale web applications, SOA, high-available systems, and cloud computing.

Michał Wrzeszcz is Ph.D. student of computer science at the AGH University of Science and Technology in Cracow, Poland and researcher at the Academic Computer Centre CYFRONET-AGH. He is the author or co-author of over a dozen scientific papers and conference contributions. His research interests are organizationally and geographically distributed data management as well as artificial intelligence.

Jacek Kitowski (Full Professor of Computer Science) is the Head of Computer Systems Group in the Department of Computer Science of the AGH University of Science and Technology in Cracow, Poland, and senior researcher at the Academic Computer Centre CYFRONET AGH, being responsible for developing high-performance systems and grid environments. He is the author or co-author of about 220 scientific papers. His topics of interest include large-scale computations, multiprocessor architectures, parallel/distributed computing, grid services and cloud computing, knowledge engineering and semantic technologies. He was/is involved in many international (FP5, FP6, FP7, EDA) and national projects, most notably funded by the European Regional Development Fund as part of the Innovative Economy Program, being the director of the PL-Grid Consortium.