Journal of Ambient Intelligence and Smart Environments 2 (2010) 21–30
DOI 10.3233/AIS-2010-0050
IOS Press



# Improving user verification by implementing an agent-based security system

Erik Dovgan*, Boštjan Kaluža, Tea Tušar and Matjaž Gams
*Department of Intelligent Systems, Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia*

**Abstract.** This paper presents an agent-based, high-level security system for user verification. The system verifies a user's identity by processing data from several low-lever sensors mounted at an entry point combined with knowledge of the user's past behavior. The data from a new entry are processed by several agents, which store the knowledge in an ontology. A single agent's classifications are integrated into the overall decision. The system successfully detects intruders, even when they optimally fake low-level sensors, e.g. fingerprints, and regular personnel under the influence of drugs.

Keywords: User verification, security systems, access control, agent-based systems

## 1. Introduction

Trust is essential in security tasks. A subject can only be trusted when he or she is identified. After a positive identification, he or she can obtain the relevant permissions to operate inside a system. However, a critical situation can occur when a non-authorized person bypasses the identification procedure and obtains high-level permissions. To prevent such intrusions, advanced modules need to be added to the system, since the traditional identification stage, on its own, does not ensure a high level of security control. For example, a traditional identification is often implemented with identification cards, which can be easily stolen or faked.

Verification is an essential part of any high-security system, since it approves the user's true identity. Only when the user successfully performs the identification and verification, can he or she gain access to security areas with assigned permissions.

We present an intelligent security system that improves the traditional verification stage with intelligent agents. These agents verify the users on the basis of previous behavior [13]. Since the system controls a high-security object, the users are instructed to behave consistently in terms of their behavior models, i.e., to enter as normally as possible and not to be too casual. Behavior models do not only determine normal behavior, they also improve the agents' knowledge and awareness of the environment. These models are stored in the form of an ontology [1]. By comparing the current behavior of a person wishing to enter with the usual behavior of the person it identifies, they improve security. Unusual behavior triggers an alarm with a user-friendly explanation.

Agents also provide some false alarms, but the cost of raising a false alarm from time to time is irrelevant compared to the cost of not raising an alarm during a critical situation, e.g., an intruder entering the building. Our initial system, without an ontology and agents, was presented in [3].

The paper is organized as follows. Section 2 describes the related work. Section 3 presents the general architecture of the proposed security system with the accompanying ontology. The experimental environment and the agents are described in Section 4. Section 5 presents the experiments, while Section 6 summarizes the work done and concludes with a discussion.

* Corresponding author. E-mail: erik.dovgan@ijs.si.

## 2. Related work

Security issues have been analyzed by several authors. The presented paper describes the security issues related to the surveillance of persons as a part of agents' ambient-intelligence modeling. The following papers also describe ambient-intelligence security systems. Wilson [18] presented an intelligent system for video surveillance, which adds data-mining methods to the existing system of cameras in a building. The intelligent methods are used to extract people from video images, recognize possible intruders, follow persons and recognize unpredicted paths or recognize threats. Sun et al. [16] described an intelligent method for user control based on face recognition, which combines 2D and 3D facial features. The information about the 3D face is derived using a Hopfield neural network. Its features are retrieved with a local autocorrelation coefficient. The 2D facial features are retrieved using a principle component analysis. Next, the 2D and 3D features are combined, which improves the user's face recognition. Lambort et al. [5] introduced an intelligent system that consists of several heterogeneous sensors with a weighted voting algorithm for computing the final result. To avoid a large number of false alarms, the results from several sensors were integrated into the classification, thus creating an advance situational awareness. Several data-mining methods were used, e.g., k-nearest neighbors, neural networks and support vector machines.

Previously described publications take into account only the ambient-intelligence security aspect. A distributed autonomous system including agents also has to take into account other aspects, e.g., the autonomy of several components. The following papers describe such autonomous systems. Pavon et al. [8] presented an intelligent, multi-sensor surveillance system with agents. The sensors are installed on fixed and mobile devices and provide a lot of information that has to be correlated and integrated in order to recognize special situations. They are applied in highly dynamic environments with strict security requirements. However, each component can act with a certain degree of autonomy. In addition, the components cooperate for complete tracking. The surveillance is achieved by controlling a set of rules and by user identification with biometric data. Pikoulas et al. [10] developed an agent-based security system for recognizing security threats. They added a Bayesian forecasting technique to predict the user's actions as an additional level to the username and password authentication. The goal is to predict whether a user acts normally or not and if there has been a security violation caused by an external user. This is done by comparing the user's current behavior with his/her typical behavior and with a predefined behavior pattern.

In summary, the presented approaches use several intelligent methods, but each approach focuses on only one, i.e., the most promising method. Furthermore, knowledge is stored in a method-appropriate way; therefore, it is not commonly accessible and cannot be used by other agents for further data analysis. In addition, the presented publications only focus on either the ambient-intelligence aspect or on the agent aspect. None of them have efficiently handled both aspects, as we do in this paper. We propose a new approach to behavior analysis and intrusion detection. It uses several intrusion-detection classification techniques, realized as agents enabling multimodal interfaces with the advantages described in [12,14]. The knowledge is stored in a common ontology, which enables the interchangeability of knowledge between agents. Consequently, additional agents or applications can be easily added to the system.

## 3. Architecture

This section firstly describes the system architecture and in the second part, the ontology is presented.

### *3.1. Agent-based security-system architecture*

The system consists of four hardware levels. The first level consists of a set of access points, which are located at the entry points of high-security rooms or buildings. The second level is a set of sensors and sensor agents, which are located on the access points, where one access point has one or more sensors and agents. The third part is a network (e.g., a TCP/IP channel), which is used as a communication channel between the sensor agents at the access points and a main server. The last part is the main server, which stores the knowledge in the ontology and is where the intelligent agents are. It receives the event information from the sensor agents, delivers this information to the intelligent agents, stores it in the ontology, implements the communication channels between the intelligent agents and presents the alarms to the administrator. The intelligent agents are implemented as threads, which operate in a

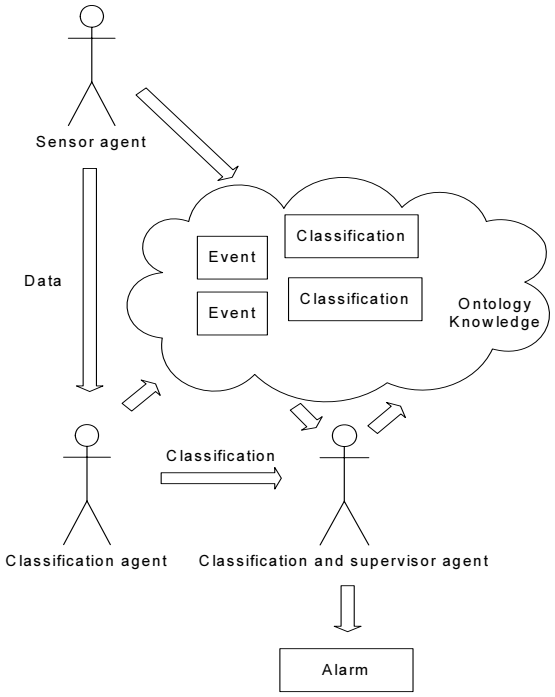


Fig. 1. Agent communication.

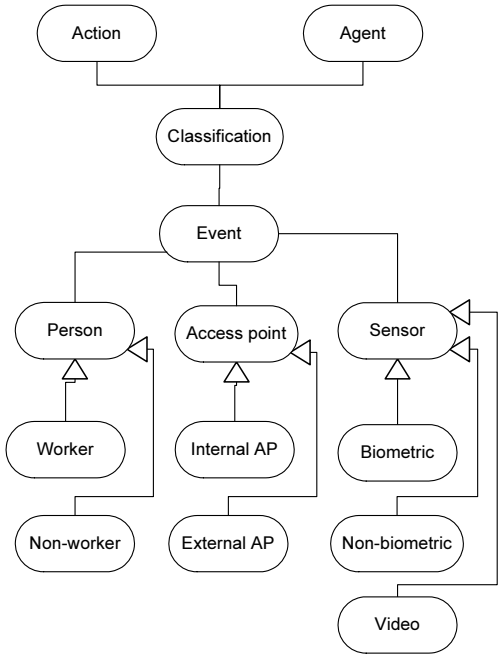


Fig. 2. The system ontology.

common system framework. They communicate by sending messages.

In the proposed security system, an arbitrary number of sensors can be applied. The commonly used sensors provide information about a person's identity card, fingerprint, face, cornea, etc. In the system there can also be an arbitrary number of intelligent agents, integrated into one classifying system. Each agent receives data from the sensor agents and reacts if the relevant patterns appear. Furthermore, each intelligent agent sends the classification result to the supervisor agent. An agent can also trigger an action relating to the classification result, e.g., show the result on the screen, prevent a user's entry, or set off an alarm. The top agent is the supervisor agent, providing a final decision about the entry.

A user entry is classified as an alarm if the behavior differs significantly from his or her usual behavior. However, this might cause false alarms if the user broke his/her leg, he/she carries some heavy objects, or his/her bag drops on the floor. But as mentioned before, we deal with high-security entry and any abnormality attracts human supervision, easily distinguishing between true dangers and false alarms. Of course, when someone's behavior changes permanently, e.g., a user has a broken leg, the administrator starts to build a new behavior model. In this case the administrator deletes the user's old behavior model and the agents learn a new behavior model from the entries learning online.

### *3.2. Ontology for storing knowledge*

The ontology is used to store the knowledge of the agent. In this way, each agent can use the knowledge of other agents, as shown in Fig. 1. It is also possible to use other agents' classifications, thus enabling several additional classification layers, such as meta-learning, meta-meta-learning, etc. The system presented here uses three classification layers.

The presented knowledge and data are stored with the Web Ontology Language (OWL) [17], a standard language for describing the classes or entities, the properties, the relations between classes, the rich types of properties, the cardinality of properties, and the characteristics of properties.

The system was implemented in Protégé OWL [4]. Figure 2 shows a model of the central system ontology, with the following entities:

**Access point entities** denote the physical areas, e.g., a building consisting of public and/or restricted areas. There are two types of access-point entities:

1. Internal access points define the various degrees of security for different areas.
2. External access points control the entries at several "border" or "perimeter" locations.

**Person entities** incorporate the data of people. There are two types of person entities:

1. Worker entities, which store the data either of regular workers, having access to less-restricted areas, or of principal workers, having access to more-restricted areas.
2. Non-worker entities, which have access to public and/or unrestricted areas in the building.

**Sensor entities** consist of the sensor data acquired during the events. There are three types of sensor entities:

1. Biometric sensor entities, e.g., fingerprint reader and face reader.
2. Non-biometric sensor entities, e.g., card reader and door sensor.
3. Video sensor entities, implemented in such a way as to prevent storage overload.

**Agent entities** store data and information from the agents. They can also entail procedures and parameters used for the verification of events. A decision-tree learner and an interpreter are typical examples of an agent.

**Action entities** describe actions that are performed when a condition is satisfied. The actions are typically presented as a set of program commands.

**Classification entities** are related to event classifications by the agents. For example, if the classification of a module is alarm, then the following actions can be performed: display a message on the screen, call a supervisor on a cell phone, call the police, deny access, raise an alarm, etc.

**Event entities** describe events produced by one person at one access point. This access point can contain many sensors, the values of which are part of the event. Each single and integrated classification is also part of an event.

## 4. The experimental environment

An agent-based system was implemented to verify the presented approach. Its hardware and software components are described in the following subsections.

### *4.1. System configuration*

The system schema is very flexible, and applicable for a large or small number of sensors and access points. We tested the system by using one access point with a door equipped with an open/close door sensor, an identification-card reader, a fingerprint reader, and a camera. These sensors, except for the camera, are connected to a DOX Access Controller [15], which can control up to seven sensors and can be hierarchically connected to other DOX Controllers. The controller communicates with the main server via a TCP/IP communication channel. The camera communicates with the main server directly via the TCP/IP communication channel.

When an event occurs, the four sensor agents read the data from the sensors and send the data to the ontology and to each of the classification agents at the main server. The event data are processed and classified by the classification agents. The results are then sent to the supervisor agent and stored in the ontology. Next, the supervisor agent calculates the final classification and if it confirms the entry, the door is unlocked and the user is able to pass through. Each agent classifies an event into one of the following three classes: OK, Warning, or Alarm. It also provides a detailed explanation of its decision, which is comprehensible to the supervisor. In the future, we plan to include more access points with a larger number of sensors.

### *4.2. Implemented agents*

Four groups of classification agents were implemented in the experimental setup, i.e., reflex agents, short-term agent, long-term agents and movement agent, and one supervisor agent as shown in Fig. 3. In the next sections, the agents are described in more detail.

#### *4.2.1. Reflex agents*

The reflex agents implement rules and do not learn from past events. They are realized as generic reflex agents and become operable when their parameter values are defined. The final classification is an alarm as soon as one reflex agent classifies the entry as an alarm. In the experimental system, the following generic reflex agents were implemented and parameterized: a long-term time agent, a short-term time agent, a long-term event-sequence agent and a short-term event-sequence agent. From these 4

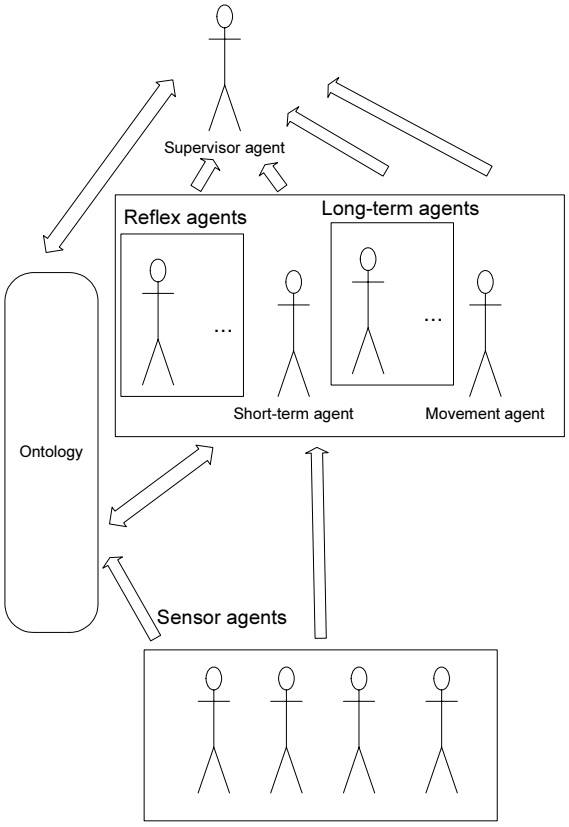


Fig. 3. Agents in the experimental setup.

generic agents, 10 executable reflex agents were generated.

The rules were defined in collaboration with security experts from the Slovenian Ministry of Defense. The parameterization and the tests of the rules were also executed under the supervision of the same security experts.

The rules are implemented in SWRL (Semantic Web rule language), an extension of OWL, using a subset of RuleML [6]. It enables the use of complex predicates for a precise definition of the concepts and deductive reasoning. The SWRL rules have two parts: the IF part or antecedents, and the THEN part or consequents [7]. There are three types of atoms:

1. Atoms defining a class variable, for example, a person(?X).
2. Atoms checking the property of class instances, for example, a name(?X, ?Y).
3. Built-in atoms for basic arithmetic and comparison operations. An example of a basic arithmetic atom is the subtraction built-in atom swrlb:subtract(?C, ?B, ?A), which calculates C = B – A.

The execution of the rules is implemented with the Jess rule engine.

In the presented system, four types of rules are implemented. The first type of rules represents short-term time rules, checking the short-term times of a person at the access point: when the person enters through an access point, a sequence of events is recorded by a sensor and the time differences between each pair of entry-event times are measured. The experimental system has four sensors; therefore, the short-term time rules check three time differences:

1. The time difference between the time of acceptance of the identity card and the time of acceptance of the fingerprint.
2. The time difference between the time of acceptance of the fingerprint and the time of the door opening.
3. The time difference between the time of the door opening and the time of the door closing.

The rule is used for identifying the following unusual events:

1. The door remains open for a too long period of time.
2. The entry speed from the identification-card reader to the door closing differs by n% compared to the usual entry, e.g., a user goes through the check point too slowly, which means the user is possibly ill or drugged.
3. A person goes through the check point too fast, because, for example, he/she is running away from danger, etc.

The second type of rules is related to long-term times – the time in the day and the day of the week for the person at the access point. The rule is used for the identification of unusual events:

1. at night,
2. on Sunday,
3. on Saturday,
4. during out-of-hours' times, etc.

The third type of rules deals with sequences of short-term events. The rule is used for the identification of the following unusual events:

1. The door opens without any previous identification or verification, which happens, for ex-

ample, when a bomb attack occurs and the door is smashed.
2. The door does not close, etc.

The fourth type of rules deals with long-term sequence rules, verifying whether a person initiates too many events in a limited time, for example:

1. A person is stealing several objects, and is thus making too many entries.
2. An identity has been stolen or faked and the unauthorized person creates "impossible" events, e.g., being at two locations at the same time.

The following examples show the implementation of the described types of rules. A syntax explanation can be found in [7]. The first rule shows an example of the first type of rule, which checks the time difference between the time of the door opening and the time of the door closing. It produces an alarm if the difference is greater than 7 seconds.

```
event (?event) ^
sensor (?sensor1) ^
eventSensor3 (?event, ?sensor1) ^
name (?sensor1, ?name1) ^
swrlb:equal (?name1, "doorOpens") ^
sensor (?sensor2) ^
eventSensor4 (?event, ?sensor2) ^
name (?sensor2, ?name2) ^
swrlb:equal (?name2, "doorCloses") ^
time (?sensor1, ?time1) ^
time (?sensor2, ?time2) ^
swrlb:subtract
(?timeDiff, ?time2, ?time1) ^
swrlb:greaterThan (?timeDiff, "7")
->
classification (?event, "Alarm") ^
explanation (?event, "The time differ-
ence between doorOpens and doorCloses
is greater than 7 seconds.")
```

The second example shows the second type of rule, which produces an alarm when an event occurs outside the working hours, after 16:00.

```
event (?event) ^
sensor (?sensor) ^
eventSensor3 (?event, ?sensor) ^
name (?sensor, ?name) ^
swrlb:equal (?name, "doorOpens") ^
time (?sensor, ?time) ^
swrlb:greaterThan (?time, "16:00")
->
classification (?event, "Alarm") ^
explanation (?event, "The event did not
occur during the working time.")
```

The third example shows the third type of rule, which produces an alarm when the identification stage was not completed between the last door closing and the next door opening.

```
event (?event) ^
sensor (?sensor1) ^
name (?sensor1, ?name1) ^
swrlb:equal (?name1, "card") ^
sensor (?sensor2) ^
name (?sensor2, ?name2) ^
swrlb:equal (?name2, "doorCloses") ^
sensor (?sensor3) ^
eventSensor3 (?event, ?sensor3) ^
name (?sensor3, ?name3) ^
swrlb:equal (?name3, "doorOpens") ^
time (?sensor1, ?time1) ^
time (?sensor2, ?time2) ^
time (?sensor3, ?time3) ^
swrlb:greaterThan (?time2, ?time1) ^
swrlb:greaterThan (?time3, ?time2)
->
classification (?event, "Alarm") ^
explanation (?event, "No identification
stage has occurred between last door
closing and current door opening.")
```

The last example shows the fourth type of rule, which produces an alarm if a user does more than three events in 60 seconds.

```
event (?event1) ^
event (?event2) ^
event (?event3) ^
event (?event4) ^
person (?event1, ?person) ^
person (?event2, ?person) ^
person (?event3, ?person) ^
person (?event4, ?person) ^
time (?event1, ?time1) ^
time (?event2, ?time2) ^
time (?event3, ?time3) ^
time (?event4, ?time4) ^
swrlb:lessThan (?time1, ?time2) ^
swrlb:lessThan (?time2, ?time3) ^
swrlb:lessThan (?time3, ?time4) ^
swrlb:subtract
(?timeDiff, ?time4, ?time1) ^
swrlb:greaterThan (?timeDiff, "60") ^
->
classification (?event, "Alarm") ^
explanation (?event, "The user produced
more than three events in 60 seconds.")
```

*4.2.2. Short-term and long-term agents*

The short-term agent is designed on the basis of empirical evidence that users enter in their own personalized manner and rarely change their habits over

time, e.g., a female user usually carries her card in a handbag.

The users' habits are modeled as follows. All the event data consist of four short-term times: ID-card time, fingerprint time, door-opening time and door-closing time. The three time differences are calculated between the successive short-term times. These differences determine a 3-dimensional short-term features space. Therefore, every user's entry represents a point in that space. If a new entry is close to the regular-entries cluster of the entering identified user, then the user is considered to be entering normally.

We experimented with several algorithms for outlier detection and, finally, the LOF (Local Outlier Factor) was chosen [2]. Therefore, although several agents were tested, only one was finally implemented. It classifies an event as an alarm if the LOF exceeds a predefined threshold, which was obtained with an algorithm tuning stage.

The long-term agents are focused on the daily routine of the users passing through one access point, and the movements between different access points. For example, some users usually come to work together; some are smokers and pass through a particular access point more often. To detect such routines and the dependencies between users, the following long-term features were used: time, date, day of the week, date in relation to the month, etc.

The top long-term agent is the supervisor second-layer agent, which combines the results of three single long-term agents. The first and the second long-term agents apply the C4.5 algorithm [11] for constructing the decision trees. The first agent uses only the long-term features of the entry, while the second agent uses both the long-term and the short-term features. Therefore, it joins the long-term knowledge with the short-term knowledge, using the ontology. The third agent uses the LOF algorithm on the long-term and the short-term features. It also integrates the long-term knowledge and the short-term knowledge stored in the ontology. The final classification by the top long-term supervisor agent was first obtained by voting and later by more complex algorithms.

*4.2.3. Movement agent*

The movement agent uses the body movement of each person for his/her identity verification with the help of cameras. Like the short-term agent, the movement agent also takes into consideration that users move in their own personalized manner, which rarely changes.

A user is verified by extracting the movement signature from the entry video. The signature is obtained by identifying the elementary movements, estimated with histograms. Then it is compared with the user's past movement signatures. If the similarity is high, the user is positively verified. A detailed explanation of the method is given in [9].

*4.2.4. Supervisor agent*

This third-level agent combines the results from the previously described agents as follows. When the reflex agents report an alarm, all the other agents are irrelevant, as something is very wrong and demands urgent action. Therefore, the reflex agents trigger an alarm and the supervisor agent is just informed.

The supervisor agent observes the classifications of the other agents and provides temporal decisions. After a certain time or when all the classification agents make their assessments, the supervisor agent provides the final decision. An alarm can be triggered at any time during the classification process, when the predefined thresholds are reached.

During the agents' classification their conclusions are sent to the supervisor. The current implementation of the system displays the results on a computer screen. An example of an event explanation is shown in Fig. 4. It shows the classification of three types of agents: rule agents classified the event as ok, while the short-term agent and all the long-term agents classified the event as an alarm. The supervisor agent concluded that the final classification is an alarm. A detailed explanation of each classification is given at the bottom of the figure. The short-term and long-term explanations include the classification value, i.e., the tree classification precision [11] and the LOF value [2]. The classification thresholds for warning and alarm are also given. The long-term explanations also include the attributes used for the classification in the classification tree and their values. The long-term time denotes the event time in minutes.

## 5. Verification

The test of the experimental environment, presented in Section 4, was performed in two stages. The first stage was composed of a collection of real-world entry data. Five workers were asked to produce 37 entries at the described entry point. The achieved data were monitored and if any entry data included irregularities a new entry was produced.

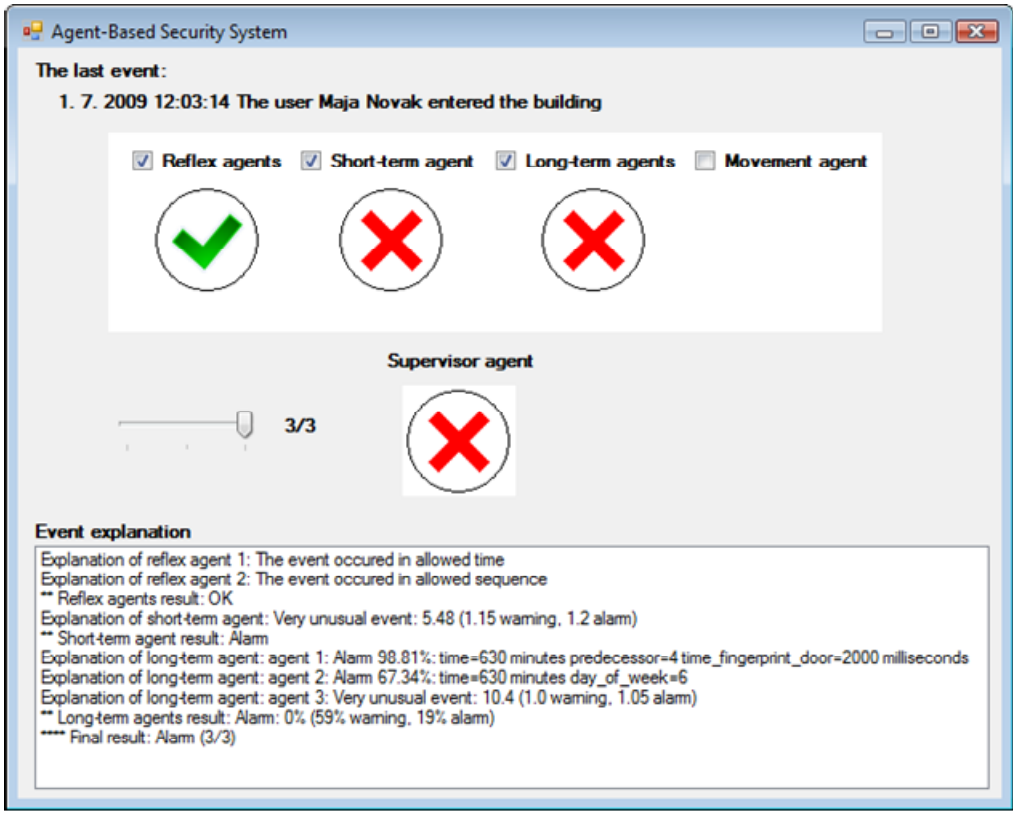


Fig. 4. Explanation of the classifications.

The output from this stage was the data of 37 regular entries for each of 5 workers.

The second stage involved an analysis of the data, which was done with an offline simulation. The simulation did not include video agent classifications, since the video data were not collected at the time when the first stage was running and the simulation of the offline video data is hard to perform.

The second stage included a test of regular events and the stolen-card and forged-fingerprint experiment. The test of regular events was made with a 10-fold cross-validation for each user independently. Therefore, at one step, each entry of one fold (including 10% of entries, i.e., 4 entries) was tested with the security system, where the agents' behavior models are built with regard to the entries of other folds (90% of entries, i.e., 33 entries) of the same worker.

The last part of the second stage included a simulation of the stolen-card and forged-fingerprint experiment. The experiment tests the system on entries, where the real user is not the user that was identified by the system. Such events occur when an intruder steals an identification card, forges the fingerprint and bypasses the sensors. The recognition of such events adds a real added value since the events are not recognized by the already reliable card identification and fingerprint verification. The simulation of such events was made as follows. We assumed that intruders do not know that the entry behavior is tested and therefore they do not try to imitate the behavior of the real workers, but they behave in their own specific manner. Five tested workers did not try to imitate each other; therefore, their data were also used to simulate intrusions. This was done by declaring a worker as an intruder that had stolen the identity of another worker. The following example shows how such a simulation is performed:

- worker 1 is the intruder that behaves in his/her own specific manner;
- he or she has stolen the identification card and forged the fingerprint of worker 2;
- the identification part of the system (identification-card and fingerprint checks) recognizes him/her as worker 2;

Table 1
Preliminary results for regular entries

| Agents | Reflex | Short-term | Long-term | Supervisor |
|---|---|---|---|---|
| OK | 1 | 0.98 | 0.9 | 0.88 |
| Warning | 0 | 0.02 | 0.1 | 0.12 |
| Alarm | 0 | 0 | 0 | 0 |

Table 2
Preliminary results for stolen-card and forged-fingerprint entries

| Agents | Reflex | Short-term | Long-term | Supervisor |
|---|---|---|---|---|
| OK | 1 | 0.35 | 0.14 | 0.13 |
| Warning | 0 | 0.15 | 0.24 | 0.18 |
| Alarm | 0 | 0.5 | 0.62 | 0.69 |

- consequently, his/her behavior is tested with the behavior models of worker 2;
- since the entry was not produced by worker 2, but by worker 1, who behaves in his/her own manner, the entry behavior is not similar to the usual behavior, defined by the tested behavior models;
- the agents recognize the behavior differences and raise the alarm.

The simulation was made by testing each pair of workers, where, firstly, one of them was the intruder and, secondly, the other was the intruder. In both cases the non-intruder worker was simulating the behavior of a regular worker. Each time all the regular-worker data were used to build the behavior models. Finally, each intruder entry was simulated and tested on the regular-worker behavior models.

Table 1 shows the classifications of 185 regular entries of 5 workers. The system correctly classified 88% of such entries, never triggered an alarm and issued a warning for 12% of the entries – people sometimes move a little irregularly. Table 2 shows the classifications of 185 stolen-card and forged-fingerprint entries of 5 workers, when a user forged another ID, but retained the original movement. The system correctly classified 69% of such events as an alarm. This table also shows that the reflex agents classified all the stolen-card and forged-fingerprint entries as regular entries. In other words, the reflex agents check only the entry regularity and do not classify an entry based on the user's behavior.

It must be emphasized that the tests were made as if the biometric control was trivially by-passed. In real life, the biometric control is the basic security control and is hard to overcome on its own. Therefore, the intelligent agents represent an additional level of security for high-security areas. Furthermore, we believe that adding a movement agent, which was not included in the described tests, only increases the security, which has been proved with other tests that are not presented in the current paper.

For more reliable results, a full-scale evaluation will be performed in the future. Such an evaluation will include the testing of a larger number of workers and a full-system testing. This was not already done because the benchmark tests do not exist. Consequently, a comparison between the presented and other systems' results cannot be made.

## 6. Conclusion and discussion

This paper presents an agent-based security system in which ambient intelligence agents observe entry points and their sensors, which are organized in several classification layers, and their knowledge is stored in a common ontology. The system is very flexible since it allows the easy adding or removing of intelligent agents, entry points and sensors. The intelligent agents model the users' usual entry behavior models and use them for the verification of future entries.

The system was tested with a stolen-card and forged-fingerprint experiment. An example of a stolen-card and forged-fingerprint event is attempting to enter the building with a stolen identification card. Consequently, he or she easily bypasses the low-level identification stage, but the entry is additionally verified by the intelligent agents. In the experimental setup, the system successfully recognized 88% of the regular entries and 69% of the stolen-card and forged-fingerprint entries. The experiments indicate that the additional security level in the form of intelligent agents substantially improves intrusion detection.

It should be pointed out that in real life the system learns the user's behavior, which then does not change. If a user's behavior changes, the administrator has to start building a new behavior model since it is not automatically updated.

The main advantage of the system is the information exchange between the agents and the common knowledge ontology, where the information is accessible to all the agents. Such a system implementation makes it possible to deal with complex knowledge about the environment, e.g., ambient intelligence.

Future work includes, in the first step, a full-scale test of the system. In addition, a larger number of workers will be tested. We will also try to include

the additional knowledge of domain experts. New agents and tests of additional sensors' data will also be included. The following system parts will be added over a long period and the control will be extended to more than one access point. And finally, we will test how the system acts if it learns from the most current behavior, so it updates behavior models constantly, while deleting data older than one week, one month, etc. So the model building will not be undertaken at an administrator's request.

## References


[1] K. Breitman, M. Casanova and W. Truszkowski, Semantic Web: Concepts, Technologies and Applications, Springer, London, 2007.

[2] M.M. Breunig, H.P. Kriegel and J. Sander, "LOF: Identifying density-based local outliers", Proceedings of the International Conference on Management of Data SIGMOD'00, 2000, pp. 93–104.

[3] M. Gams and T. Tušar, "Intelligent High-Security Access Control", Informatica, vol. 31, no. 4, Slovene Society Informatika, 2007, pp. 469–477.

[4] M. Horridge, H. Knublauch, A. Rector, R. Stevens and C. Wroe, A Practical Guide To Building OWL Ontologies Using The Protege-OWL Plugin and CO-ODE Tools Edition 1.0, Stanford University, 2004.

[5] P. Lamborn and P.J. Williams, "Data fusion on a distributed heterogeneous sensor network", Proceedings of SPIE – The International Society for Optical Engineering, vol. 6242, Bellingham, Washington, 2006, pp. 1–8.

[6] C.J. Matheus, K. Baclawski, M.M. Kokar and J.J. Letkowski, Using SWRL and OWL to Capture Domain Knowledge for a Situation Awareness Application Applied to a Supply Logistics Scenario, Springer, Berlin, 2005.

[7] M. O'Connor, S. Tu, C. Nyulas, A. Das and M. Musen, Querying the Semantic Web with SWRL, in: Lecture Notes in Computer Science, Springer, Berlin, 2007, pp. 155–159.

[8] J. Pavon, J. Gomez-Sanz, A. Fernandez-Caballero and J.J. Valencia-Jimenez, Development of intelligent multisensor surveillance systems with agents, Robotics and Autonomous Systems 55 (12), 2007, pp. 892–903.

[9] J. Perš, M. Kristan, M. Perše and S. Kovačič, "Motion based human identification using histograms of optical flow", Proceedings of the 12th Computer Vision Winter Workshop CVWW, Graz University of Technology, 2007, pp. 19–26.

[10] J. Pikoulas, W.J. Buchanan, M. Mannion and K. Triantafyllopoulos, An agent-based Bayesian forecasting model for enhanced network security, Proceedings of the International Symposium and Workshop on Engineering of Computer Based Systems, 2001, pp. 247–254.

[11] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, 1993.

[12] R. Raisamo, V. Surakka, J. Raisamo, J. Rantala, J. Lylykangas and K. Salminen, Haptic interaction becomes reality, Journal of Ambient Intelligence and Smart Environments 1, 2009, pp. 37–41.

[13] C. Ramos, J.C. Augusto and D. Shapiro, "Ambient Intelligence – the Next Step for Artificial Intelligence", J. Intelligent Systems, 2008, pp. 15–18.

[14] N. Sebe, Multimodal interfaces: Challenges and perspectives, Journal of Ambient Intelligence and Smart Environments 1, 2009, pp. 23–30.

[15] Spica International d.o.o. web site, DOX Access Controller, [Online], avaible: http://www.spica.com/products/dl/DOXcontroller.pdf.

[16] T.-H. Sun and F.-C. Tien, "Using backpropagation neural network for face recognition with 2D + 3D hybrid information", Expert Systems with Applications, vol. 35, Elsevier, 2008, pp. 361–372.

[17] M. Thonnat, Semantic Activity Recognition, in: Proceedings on the ECAI 2008, IOS Press, 2008, pp. 3–7.

[18] D.L. Wilson, "Intelligent video systems for perimeter and secured entry access control", Proceedings of the 39th Annual IEEE International Carnahan Conference on Security Technology ICCST, IEEE, 2005, pp. 260–262.