Banking Frauds: An Agent-Based Training Framework to Follow-up the Swindlers Learning Process

Mauricio Paletta [1] Pilar Herrero [2] and Gonzalo Méndez Pozo [3]

[1] Departamento de Ciencia y Tecnología. Universidad de Guayana. Av. Atlántico. Ciudad Guayana. Venezuela
Email: mpaletta@uneg.edu.ve
[2] Facultad de Informática. Universidad Politécnica de Madrid. Campus de Montegancedo S/N. 28.660 Boadilla del Monte. Madrid. Spain
Email: pherrero@fi.upm.es
[3] Departamento de Ingeniería del Software e Inteligencia Artificial. Facultad de Informática. Universidad Complutense de Madrid. Ciudad Universitaria. 28.040 Madrid. Spain
Email: gmendez@fdi.ucm.es

Abstract: Electronic bank transactions are very common today. Services given by an Automatic Teller Machine (ATM), for example, are very popular and widely used by bank clients. Unfortunately, in the same way as the use of these devices is increasing, the proliferation of different frauds to try to violate these systems to steal user's money is also increasing. In this paper, a framework designed to follow up the swindlers' agents learning process is presented. This framework is based on an open and flexible architecture that emphasizes on the swindler agents learning process to fulfil not only more human-like agent behaviour but also a more realistic interaction with the environment.

Keywords: IVA, learning, architecture, ATM, electronic bank fraud.

1. Introduction

Electronic banking frauds have attracted significant international attention, since individuals and organizations have lost billions of dollars worldwide. Electronic banking speeds up transactions and creates new "promising" services, altering banking operations, and dramatically expanding the reach of financial institutions. Services given by an Automatic Teller Machine (ATM), for example, are very popular and widely used by bank clients. In fact, this kind of transaction is leading the current payment system.

The proliferation of different frauds to try to violate these systems to steal user's money is also increasing. Bank institutions are continuously receiving claims from victims of this type of crimes and the only thing they can do is to inform their employees and clients about one particular modus operandi once it is discovered.

The detection of this modus operandi is not easy and once it is discovered, criminals find another more skilful way to proceed. Sometimes, the modus operandi involves different people and techniques making them more difficult to detect. In this sense, having an environment capable of following up the swindler agents learning processes would be very useful to discover how they improve their modus operandi day by day and how they "create" new techniques and schemes to dodge the bank systems devoted to detect this kind of frauds.

This paper presents an environment inhabited by Intelligent Virtual Agents (IVAs) each of which has been endowed with an open and flexible architecture based on human learning process works [7]. Therefore, the swindler agents endowed with this architecture will learn from their own mistakes and experiences in the same way that human swindlers would learn in real life. This agent-based environment will allow banks to follow up the swindlers' learning process, giving them the chance to anticipate new schemes, situations and actions.

The results presented in this paper are part of the research program that is being carried out at the Universidad de Guayana (Venezuela) in close collaboration with the Universidad Politécnica de Madrid (Spain). This research program is being supported by XPECTRA, a Venezuelan enterprise that has been working for a long time on new techniques for tackling banking frauds in collaboration with well-known trademarks manufacturing ATM.

2. Related Work

A large amount of architectures for intelligent agents have been defined, and BDI (Belief-Desire-Intention) [10] is one of the best known and used currently, but it is important to emphasize that only a few of these architectures undertake the specific area of IVA. An example is represented by the work of Jean-Sébastien Monzani [5] and, more recently, the proposals of De Antonio et al [2] and Herrero et al [3].

There have been several works that have exploited the idea of open and flexible architectures. Martin et al [4] explain the structure and elements for the construction of agent-based systems using OAA (Open Agent Architecture), in which various agents help each other under the premise of requiring and/or providing services based on their capabilities. In this approach, each agent that participates in the system defines and publishes a group of capabilities through ICL (Inter-agent Communication Language), also defined in OAA.

In [2] the authors show an architecture for the development of intelligent agents based on a set of cooperative software agents. The idea of using agents to structure the architecture of the intelligent agent is exploited here.

Some authors [1] [5] [8] identify a conceptual separation between body and mind, incarnate agent and intelligent virtual agent. The first one is responsible to manage the animation of the body and the interaction with the environment; the second is the module that makes decisions and has behaviour. It refers to the perception of the environment on one side and to the intelligence component on the other. The importance of considering relative elements not only to the behaviour but also to the perception of the environment, in the architecture of the intelligent virtual agents, is emphasized here.

However, none of the previous authors has focused on the main objective from the perspective of this paper: to design and develop an open and flexible framework based on the learning process to be applied to the electronic banking fraud.

In the next section, the way in which an IVA has been structured will be presented, by means of an open and flexible architecture based on the IVA Learning process (IVAL) [7], to tackle ATM banking frauds.

3. A Training Framework for Human-Like Agents

3.1 The IVAL architecture

IVAL is a BDI architecture based on a group of agents that cooperate to achieve the specific objectives of the IVA. Each of these agents is specialized in a particular service associated with some abilities: deliberate, react, socialize, interact with the environment, learn, and others. These agents are divided into: one Agent of Control (AC) and several Agents of Service (AS).

The AC is the main piece of the IVAL architecture. Its main function is to conform itself with all the abilities that, as a group, offer not only the same AC but also all the AS with which it interacts. This allows an IVA to be seen, from outside, by the rest of the IVAs that surround it, like an agent equipped with all these abilities. Additionally, the AC is the key element to make this architecture extensible. In order to comply with these functions, the AC will have (see Fig. 1):

- A knowledge base that contains: the IVA objectives; its planning; the current abilities provided by the AS with which it is interacting; and the knowledge learned through the corresponding AS.
- The following functional modules:
  1) Communication module, which takes care of handling not only the socket of connection but also the communication protocol with the AS.
  2) Planning module, which contains the algorithm that allows the IVA to execute its action plans to achieve the objectives.
  3) Learning module, which contains the set of basic learning algorithms of the IVA. If there is any AS in the IVA structure any AS with the learning ability, this module is responsible for unifying the learning process so that it can be perceived from the outside as if it was a single unit. The module is also responsible for resolving possible inconsistencies.
  4) Interaction module, which handles the basic interaction of the IVA with the environment.
  5) Central module, which synchronizes the execution of the remainder modules and allows the interaction of the shared memory (described below) with the other agents.

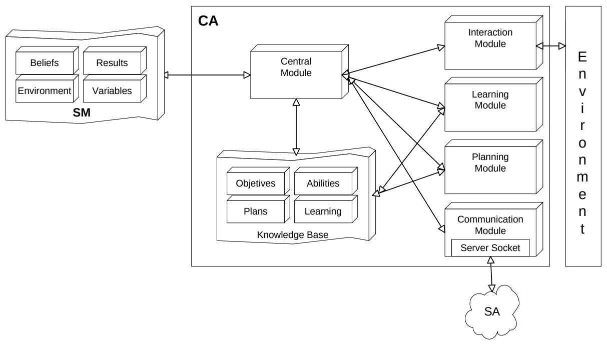

Fig. 1. Internal structure of the AC.

The main function of the AS is to cover the ability that has been conferred to them, in addition to communicating their results to the AC. The AS can be added / removed from the agent system originating an increase / decrease of the IVA´s abilities. These AS do not have direct interaction with the environment unless they are related to specialized services associated with typical functionalities of interface with the environment: sound management, images processing, natural language processing, etc. This particular treatment with the environment is important for this type of agents since they shall be continuously interacting with not only the surrounding environment but also other agents inside the same environment.

In addition, there is a memory area shared by all the agents (SM – Shared Memory), which it is located in a space, preferably in the same location where the AC resides in. All the agents know the location of this memory to be able to access it. This approach follows the basic principles of the blackboard architecture model [11], in which the agents share the available information about the system state and environment. Each agent gets from the blackboard the information that is useful for it and transcribes on it the obtained results at any moment during the execution.

This SM is formed by XML based documents that represent and store the description of the environment, the beliefs of the IVA, the conclusions and results of the AS called services, and the variables and internal states. This information represents the minimum necessary that all agents require, including both AC and AS, to be free of showing their abilities and to achieve the specific objectives for which they have been designed.

The agents know the environment with which they are interacting and how they are interacting with it, as well as the general beliefs of the IVA. From the conclusions and results of the other agents and environment, a specific agent is able to: perform particular functions as, for example, learning and reinforce the answer originated by some particular service, in the case of sharing abilities with other agents. Finally, the variables and internal states represent the collection of any other information level that shall be shared among the agents, so they could develop their abilities.

The way in which an AS is incorporated into the IVA structure follows the same idea considered in the Web Services with the publication of the services. As in the case of the existence of a WSDL (Web Service Description Language) document, that describes the content of a Web Service, there is an XML based document that describes the abilities of the IVA (see section 3.2 for more details).

The communication between the AS and AC is carried out by flows of information in XML format, in the same way as SOAP (Simple Object Access Protocol) is used in Web Service systems. The AC is constantly listening a server/client TCP/IP communication channel to allow the AS to be connected. The first thing an AS shall do when it is connected is to send the corresponding XML document that describes its abilities. The AC updates its knowledge base with the list of the current abilities of the IVA.

Taking in consideration what was described before, the way in which the AS and AC agents collaborate with each other to achieve the IVA objectives is by means of a continuous interchange of messages and storing and sharing certain information in the SM.

3.2 The XML based IVAL languages

In order to accomplish the representation and the interchange of the necessary information IVAL needs to, for example, describes the AS abilities, the message interchange between the AC and any AS, and the knowledge representation, the IVAL specifications include the definitions of the languages based on XML: IVAL-SADL, IVAL-VACL and IVAL-KBEL, respectively.

IVAL-SADL (Service Agent Description Language for IVAL) is used to have each AS identify and describe its abilities or services. The IVAL-SADL file of an agent contains three levels of information: 1) basic information related to the implementation, such as the version and author; 2) the set of abilities that the agent implements; 3) the set of service or protocol commands to which the agent can respond.

Regarding the communication between the AS and the AC, the IVAL-VACL (Virtual Agent Communication Language for IVAL) language allows the AC to send service requests to the AS and these, at the same time, to send the answers to the AC. In order to achieve this, an exchange of messages occurs, each of which shall contain the following: 1) the identification of the agent that makes the delivery; 2) the identification of the involved service or process, and 3) the parameters or corresponding response. An example is shown in Fig. 2.

```
<IVAL-VACL:Message sender="an agent id">
  <IVAL-VACL:Service>
    <IVAL-VACL:Name> service name </IVAL-VACL:Name>
    <IVAL-VACL:Parameters quantity="2">
      <IVAL-VACL:Parameter-1> par1
      </IVAL-VACL:Parameter-1>
      <IVAL-VACL:Parameter-2> par2
      </IVAL-VACL:Parameter-2>
    </IVAL-VACL:Parameters>
    <IVAL-VACL:Result> the result </IVAL-VACL:Result>
  </IVAL-VACL:Service>
</IVAL-VACL:Message>
```

Fig. 2. Example of a message written in IVAL-VACL.

Finally, in order to accomplish the knowledge representation, IVAL-KBEL (Knowledge Base Experience Language for IVAL) has been designed. It is described in detail in the next section.

3.3 The IVAL learning process

Thanks to the corresponding module in the AC structure (Fig. 1), IVAL is provided with elementary learning abilities based on the principles of the production systems [9] and using a suitable reasoning mechanism. One of the fundamental purposes of this module is to maintain the knowledge learned by the IVA. Since learning is one of the abilities or specializations that can be associated with an AS, the other function of this module is related to the coordination of possible AS with learning abilities existing in the architecture. The AS defined to cover this ability are informed, through the AC, about everything that happens in the SM of the IVA as well as everything that happens in the rest of the environment when the remainder agents complete any task or satisfy any of their objectives.

Since the agent associated with the IVAL architecture is acting according to the received stimuli, the relation between what it is received and the given reactions needs to be represented and stored. A stimulus consists on a pair <data, type> representing the information coming from the environment. The IVA's reaction consists of a set of executed services with their corresponding parameters; for example, in the services associated to the talking ability, the parameter will be the string that the IVA wants to say. Additionally, each service is accompanied by its corresponding execution result or reaction, and it is given by a pair $<service\_name(p_1, p_2, \ldots, p_n), result>$ where p_i, $1 \leq i \leq n$, is the i^{th} parameter. It is important to mention that the result of a service not only indicates whether the service was executed properly or not, but it also it indicates if the execution of this service was convenient, was not adequate or is not known yet. This is useful in order not to commit the same mistakes when the experience is inappropriate and to repeat the same action when the experience is successful.

The reasoning process occurs when a new stimulus arrives and the AC looks for an adequate answer; it consults the learning module to know whether there is any experience with the present state or stimulus. If a knowledge rule associated with the received stimulus is found, all the services identified in the rule with the appropriate result value, are executed in the same order they are executed when the same stimulus is received during the learning process.

If a different service is executed after the evaluation of the knowledge rule, this new information is considered to reinforce or give feedback to the knowledge according to what it was learned. This learning strategy continues until a new stimulus is received. This means that all the services executed after a stimulus is received and before a new stimulus arrives are taken in consideration to upgrade the knowledge rule associated with the first stimulus.

The way in which the knowledge is represented, using the IVAL-KBEL language, follows the rule's representation [6]: a conditional or left part and a consequence or right part. In this sense, IVAL-KBEL defines the request labels to structure a knowledge rule. Any rule has a different identification number that is used to indicate the corresponding sequence into the knowledge base. This is useful to know what has occurred before the arrival of the stimulus of each rule. The IVAL-KBEL document structure can be appreciated in Fig. 3.

```
<?xml version="1.0"?>
<IVAL-KBEL:Knowledge
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
 xsi:noNamespaceSchemaLocation="IVAL-KBEL.xsd"
 xmlns:IVAL-KBEL="http://www.Ival.org">
 <IVAL-KBEL:Experience id="…">
  <IVAL-KBEL:Stimulus source="…">
   <IVAL-KBEL:Data>…</IVAL-KBEL:Data>
  </IVAL-KBEL:Stimulus>
  <IVAL-KBEL:Service name="…">
   <IVAL-KBEL:Parameter name="…" type="…">…
  </IVAL-KBEL:Parameter>
   … more parameters from this service
   <IVAL-KBEL:Result type="…">…</IVAL-KBEL:Result>
  </IVAL-KBEL:Service>
  <IVAL-KBEL:Service name="…">
   <IVAL-KBEL:Parameter name="…" type="…">…
  </IVAL-KBEL:Parameter>
   … more parameters from this service
   <IVAL-KBEL:Result type="…">…</IVAL-KBEL:Result>
  </IVAL-KBEL:Service>
  … more services from this experience
 </IVAL-KBEL:Experience>
 … more experience from this knowledge base
</IVAL-KBEL:Knowledge>
```

Fig. 3. The IVAL-KBEL document structure.

3.4 IVA-IVAL agents and the ATM banking frauds

This section presents a specific kind of problem of banking frauds: Automatic Teller Machine (ATM). Due to the specific capabilities of interaction with the environment that this problem requires, the swindler IVA needs to be endowed with a set of AS supporting all these capabilities.

In this way, the following AS agents should be added to the IVAL structure (see Fig. 1):

1) ASSound, to interact with their surroundings by means of hearing (swindlers' listening and speaking);
2) ASSee, to perceive the environment by means of the sight;
3) ASTouch, to interact with their surroundings by means of the touch (swindlers' touching);
4) ASFace, to represent and interpret the swindlers' facial expressions such as happiness, fright, fear, etc.

Any possible situation related to ATM banking fraud would be covered by these four AS. Moreover, the learning algorithm described previously (section 3.3) allows the IVA-IVAL agent to learn of their own experiences.

4. Evaluation

Several scenarios, related to the ATM banking fraud, have been raised with the aim of validating the IVA's structure previously described. However, only one of them is going to be presented in this section. In fact, this scenario corresponds to a real situation which has been happening in Venezuela during the last years. A lot of clients have claimed their money back for this reason, with the corresponding banking damage.

With the aim of explaining better the scenario and the way in which the IVAL elements are interrelated in the architecture, each of the steps that are going to be presented in this scenario is going to be divided in two different parts: 1) a brief description of the simulated situation and 2) the explanation of how the IVAL architecture tackles this scene or step.

The environment where the scenario is carried out is a precinct of ATM with a couple of machines: ATM-1 and ATM-2. There are three agents: the delinquent D1, a second delinquent D2 and the client C1. The fraud is carried out because while the client C1 is initiating the transaction in ATM-2, the delinquent D1, who is using ATM-1, has programmed this ATM to intercept the transaction of ATM-2 acting itself as if it were ATM-2. Then, D1 prevents C1 from using ATM-2 pretending it does not work, while D2 is already finishing in ATM-1 the transaction that C1 initiated in ATM-2.

4.1 Steps of the first round of the simulation (the learning process)

Step 1: 1) C1 arrives to the ATM room while another client (D1) is using the ATM-1 machine and the ATM-2 is free. **2)** The environment perception starts once the AC agent sends an IVAL-VACL message to the ASSee agent. ASSee returns, in response, a set of IVAL-VACL messages with the corresponding visual stimuli associated with the specific environment situation. In this case, the stimulus received would be <"ATM-1 busy", seen> & <"ATM-2 free", seen>. The information about the environment must be stored in the SM by using the IVAL-VWDL document.

Step 2: 1) C1 is located in front of ATM-2. It introduces its card and its password while another client (D2) enters in the ATM room. **2)** AC has already received the stimuli from the ASSee agent and updated the environment information in the SM. According to section 3.3, as well as the algorithm described in section 3.3, since C1's knowledge about the fraud is initially null, C1's reasoning about the fraud is null and therefore it didn't react to the delinquent intentions based on its experience; C1 keeps its main purpose "to use the ATM machine as soon as possible" (based on its plans) and therefore it tries to go to the ATM-2 and to use this machine; using the IVAL-VACL appropriate message, AC orders to the ASTouch agent to execute the services "Put("card")" & "Put("password")". In this moment, C1 starts learning about what is happening, and as a consequence a knowledge rule is created based on the last stimulus received as well as on the actions executed by ASTouch; AC updates the knowledge with this new rule, modifying therefore the IVAL-KBEL document.

Step 3: 1) D2 is in the ATM-1 queue, behind D1; D1 moves towards C1 with a serious facial gesture and tell him that the ATM he is intended to use (ATM-2) counts the money but it will not be able to get the cash from the ATM (Fig. 4-a). **2)** As D2 is in the ATM-1 queue, ASSee realises that the environment has changed and then it decides to send to AC the IVAL-VACL message associated to the stimulus: <"new client in ATM-1", seen>; C1 doesn't react to this environment; moreover, the oral message that D1 sends to C1 is captured by ASSound; this agent sends to AC the IVAL-VACL message with the stimulus: <"ATM does not give money", listened>.

Step 4: 1) C1 reacts to D1's comment by cancelling the transaction which he was carrying. **2)** Since C1 receives the stimulus <"ATM does not give money", listened> described in the previous step, AC tries to make a decision based on its experience. As it has no experience on this respect, AC sends a message to ASFace with the order "ToSet("worry")" and C1's expression change to "worried". In the same way, AC sends a message to ASTouch with the order Put("cancel")" and the transaction is cancelled. AC updates the IVA knowledge with this new knowledge rule, modifying therefore the IVAL-KBEL document.

Step 5: 1) Once C1 has given the order of cancelling the transaction in the ATM-2 and it has waited for the confirmation of the cancellation, it decides to go to the ATM-1. **2)** ASSee realises that the environment has changed and then it sends to AC the IVAL-VACL message corresponding to the stimulus: <"transaction cancelled", seen >; AC tries to makes a decision based on its experience but once more it has no experience on this respect.

Step 6: 1) D1's transaction in ATM-1 is over and it moves towards ATM-2; D2, which was waiting behind D1, starts using ATM-1 and C1 places behind D2 to use ATM-1 as soon as possible. **2)** C1 updates the information about the environment into the SM..

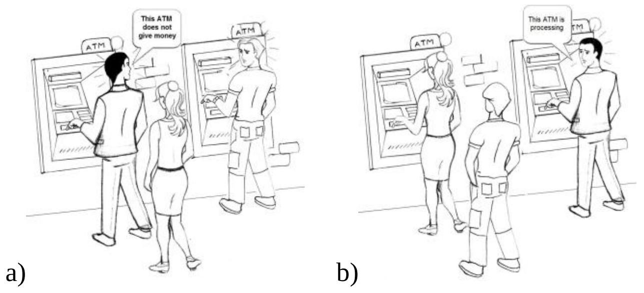

Fig. 4. a) Simulated situation of the step 3; b) simulated situation of the step 7.

Step 7: 1) D1, with a warning facial expression, says to C1 that the ATM-2 is making a transaction (Fig. 4-b). **2)** The oral message that D1 sends to C1 is captured by ASSound and therefore it sends to AC the IVAL-VACL message with the corresponding stimulus: <"ATM-2 processing", listened>; AC tries to react with this new stimulus received.

Step 8: 1) C1, with a worry expression, goes to the ATM-2 to cancel the transaction, while D1 gets out from the ATM room. **2)** AC reacts: 1) sending a message to ASFace with the order of executing the "ToSet("worry")" service (with the aim of changing the face expression), 2) sending a message to ASTouch with the order of executing the "Put("cancel")" service (with the aim of cancelling the transaction), 3) sending a message to ASSee with the order of executing the "ToSee()" service (with the aim of observing the situation); AC introduces a new knowledge rule and updates its learning.

Step 9: 1) C1 cannot observe any kind of transaction movement in ATM-2 and therefore it decides to move again towards ATM-1 where D2 is still finishing its transaction before leaving the ATM room (Fig. 5-a). **2)** AC receives a message with the stimulus <"Insert card", seen> from ASSee; AC reacts sending a message to ASFace with the order of executing the "ToSet("normal")" service; C1 goes back to ATM-1; AC introduces a new knowledge rule and updates its learning.

Step 10: 1) Once C1 is located again in front of ATM-1, it introduces its card, password and the rest of the data to start a new transaction. **2)** ASSee sends to AC the stimulus <"ATM-1 free", seen>; AC reacts sending to ASTouch the necessary messages to execute the following services: "Put("card")", "Put("password")" and "Put("options")"; AC also sends to ASSee a message to execute the "ToSee()" service, in order to get information from the environment; AC updates its learning with the new knowledge rule.

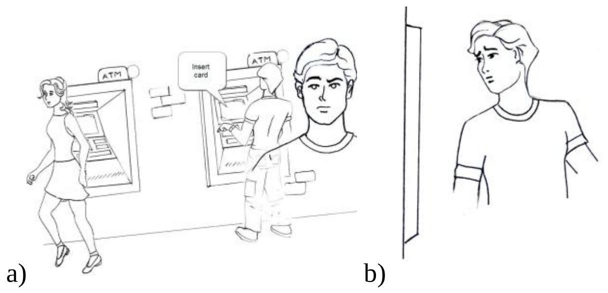

Fig. 5. a) Simulated situation of the step 9; b) simulated situation of the step 11.

Step 11: 1) ATM-1 alerts to C1 that it has already taken out the maximum amount permitted by the bank; C1 reads this message and realises of the fraud Fig. 5-b). **2)** ASSee sends to AC a message with the stimulus <"it has withdrawn the maximum quantity of money permitted for day", seen>; AC reacts sending a message to ASFace with the order of execution of the "ToSet("worry")" service; AC introduces, again, a new knowledge rule and updates its learning. At this moment it is important to mention that, due to the fact that the final result for C1 was negative, all the knowledge rules previous the last one, must be changed in order to set a non acceptable value into the corresponding services results.

4.2 Steps of the second round of the simulation (C1 has learned the modus operandi of this fraud)

Step 1: It is the same step1 from previous section.

Step 2: 1) C1 is located in front of ATM-2. It introduces its card and its password while another client (D2) enters in the ATM room. **2)** AC has already received the stimuli from the ASSee agent and updated the environment information in the SM. According to section 3.3, since C1's knowledge about the fraud is not null, C1's reasoning about the fraud is not null and therefore it reacts based on its experience; in this sense, AC orders to the ASTouch agent to execute the services "Put("card")" & "Put("password")", because this was what C1 learned once it received this stimulus (see step 1 from section 4.1).

Step 3: 1) D2 is in the ATM-1 queue, behind D1; D1 moves towards C1 with a serious facial gesture and tell him that the ATM it is intended to use (ATM-2) counts the money but it will not be able to get the cash from the ATM. **2)** As D2 is in the ATM-1 queue, ASSee realises that the environment has changed and then it decides to send to AC the IVAL-VACL message associated to the stimulus: <"new client in ATM-1", seen>; C1 doesn't react to this environment; moreover, the oral message that D1 sends to C1 is captured by ASSound; this agent sends to AC the IVAL-VACL message with the stimulus: <"ATM does not give money", listened>.

Step 4: 1) As C1 has learned that cancelling the transaction due to a comment like D1 has done is not good, it decides to continue with the transaction normally. **2)** Due to C1 receives the stimulus <"ATM does not give money", listened> described in the previous step, AC tries to make a decision based on its experience. As it has experience on this respect, and the experience is associated with the services "ToSet("worry")" and Put("cancel")" and the corresponding results of this services are not acceptable, C1 decides to continue with the transaction and therefore, AC reacts by sending to ASTouch the message with the order to execute the service "Put("options")". At this point, the knowledge rule must be changed in order to add this new service call into the reaction part. This is the way in which learning is reinforced.

Step 5: 1) The C1's transaction was successful and it receives the money from the ATM-2, so it takes its money and gets out from the ATM room. **2)** AC receives from ASSee a message with the stimulus: <"Take money", seen>; AC reacts: 1) sending a message to ASFace with the order of execute the "ToSet("happy")" service, and 2) sending a message to ASTouch with the order of executing the Take("money") service; AC introduces a new knowledge rule and updates its learning. Due to the fact that the final result for C1 was positive, all the knowledge rules previous the last one and in which the services results are not setting yet, must be changed in order to set an acceptable value into the corresponding services results.

4. Conclusions and Ongoing Research Work

Electronic bank transactions are very common today. Services provided by an Automatic Teller Machine (ATM), for example, are very popular and widely used by bank clients. Electronic banking frauds have also been increasing in parallel and, since individuals and organizations have lost billions of dollars worldwide, currently they have attracted significant international attention.

In this paper, it is presented a framework designing to follow up the swindlers' agents learning process. This framework is based on an open and flexible architecture that emphasizes on the swindlers' agents learning process to fulfil not only more human-like agent behaviour but also a more realistic interaction with the environment. One of the advantages of this framework is the possibility to build a structure of agents that cooperate with each other to achieve these objectives. Both bankers and clients can be trained to react to this kind of banking frauds in almost real-time.

Introducing the learning process inside the IVA's architecture with IVAL, is based on the significance that the same process has in human activities, primarily in the way human beings can adapt themselves to different changes inside the environment. But modeling the human learning process is not an easy task and therefore in this paper we have focused on a specific part of this process: the way in which knowledge is represented and stored using the XML based language IVAL-KBEL.

A complete scenario of a real fraud situation was presented in this paper, in which was possible to verify that an IVA, structured with IVAL, is capable of learning the modus operandi used by the delinquents to commit the fraud.

Even though IVAL was originally designed for an IVA, it could be applied to other kind of software/intelligent agents such as a mobile agent or an agent advisor, because it basically focuses on the interaction with the environment and the learning process.

One of the drawbacks of this framework is related to the knowledge representation. As it is possible to know if the service execution due to some stimulus received was or not proper, it will be interesting to associate some uncertainty management type to this reaction. For this reason, the next step will be to extend the framework that can deal with this improvement in the knowledge rule.

References

[1] A Caicedo and D Thalmann, Intelligent decision-making for virtual humanoids. In D Floreano et al. (eds) Advances in Artificial Life: Proceedings of 5th European Conference ECAL. Springer, Heidelberg, 1999, pp. 13-17.

[2] A De Antonio, J Ramírez and G Méndez, An Agent-Based Architecture for Virtual Environments for Training. In Developing Future Interactive Systems**,** Chapter VIII, Idea Group Publishing, ISBN 1591404118, 2005, pp. 212-233.

[3] P Herrero and R Imbert, Design of Believable Intelligent Virtual Agents. In Developing Future Interactive Systems**,** Idea Group Publishing, Chapter VII, ISBN 1591404118, 2005, pp. 177-211.

[4] D L Martin, A J Cheyer and B D Moran, The Open Agent Architecture: A Framework for Building Distributed Software Systems. Applied Artificial Intelligence, Vol. 13, Nos. 1-2, 1999, pp. 21-128.

[5] J Monzani, An Architecture for the Behavioral Animation of Virtual Humans. PhD Thesis, École Polytechnique Fédérale de Lausanne, 2002.

[6] A Newell and H A Simon, Human Problem Solving. Englewood Cliffs, Nj: Prentice-Hall, 1972.

[7] M Paletta and P Herrero, Learning from an Active Participation in the Battlefield: A New Web Service Human-based Approach. Proceedings 2nd International Workshop on Agents, Web Services and Ontologies Merging (AWeSOMe'06), Montpellier, France, LNCS 4277, Springer-Verlag, 2006, pp. 68-77.

[8] T Panayiotopoulos, G. Katsirelos, S Vosinakis and S Kousidou, An Intelligent Agent Framework in VRML Worlds. Advances in Intelligent Systems: Concepts, Tools and Applications, S. Tzafestas ed., Chapter 3, Kluwer Academic Publishers, Netherlands, 1999, pp. 33-43.

[9] E Post, Formal reductions of the general Combinatorial Problems. American Journal of Mathematics 65, 1943, pp. 197-268.

[10] A S Rao and M P Georgeff, Modeling Rational Agents within a BDI-Architecture. Proceedings 2nd International Conference on Principles of Knowledge Representation and Reasoning, San Mateo, USA, 1991, pp. 473-484.

[11] S Vranes and M Stanojevic, Integrating Multiple Paradigms within the Blackboard Architecture. IEEE Transactions on Software Engineering, Vol. 21, Number. 3, 1995, pp. 244-262.