Virtual Customers in a Multiagent Training Application

Philippe Mathieu, David Panzoli, and Sébastien Picault

Laboratoire d'Informatique Fondamentale de Lille (LIFL)
Lille I University,
59655 Villeneuve d'Ascq,
France
firstname.surname@lifl.fr

Abstract. Format-Store is a serious game application designed for training salesmen and managers in the context of a retail store or a larger supermarket. In this paper, we argue that a relevant way to train a salesperson to their daily activities (e.g. customer relationship management, store management and stock control) consists in immersing them in a 3d environment populated with realistic virtual customers. The first part of this paper presents the multiagent approach we apply to the design of the intelligent customers. Specifically, we analyse the contribution of the interaction-oriented methodology Ioda in facilitating the conception of a game for non computer-scientists by means of a user-friendly design tool and the automated implementation of the conceptual model. The second part describes the organisation of the game around scenarios modelled with respect to the pedagogical requirements. We discuss how the multiagent simulation is wrapped by several modules for the purpose of controlling the learning experience of the player.

1 Introduction

Format-Store is a serious game designed to train business school undergraduate students to retail trade and customer relationship management (CRM) in the context of an organic convenience store. Format-Store aims to complement a traditional learning content management system (LCMS) which contains approximately 25 thematic lectures dealing with customer welcome, information or argumentation and illustrated with practical examples of dialogues and case studies. In this context, the added value of the serious game is expressed by the following problematics: i) contextualising the knowledge from the LCMS by offering a complementary tool where the learner can apply their newly learnt skills *in situ* in a virtual store and experiment with different ways of dealing with a customer; ii) training the learner to new skills like task prioritisation or time management by means of the realistic simulation of a store, and; iii) offering a flexible evaluation of the learner, in contrast with the traditional "pen and paper" evaluation.

Assuming the role of a salesperson in a virtual replica of an actual store populated with autonomous customers, the learner is confronted to the daily tasks of managing a store and dealing with customers. Different skills are targeted by the learning game:

– Store management skills help maintaining the store functional and safe to the customers. Unpleasant to hazardous incidents can happen like a crate obstructing an aisle or a slippery stain. A salesperson must be able to notice such an event and make a decision quickly.
– Stock control is related to making sure the store is supplied with products at any time. A salesperson is expected to check the expiration date of the goods every morning and if necessary take an appropriate answer – clear the item, restock or order supplies from the wholesaler.
– Customer relationship management (CRM) is the most important aspect of a trade. It consists of looking after the customers' satisfaction, solving their problems, giving them information, etc. CRM involves a good knowledge of the products but more importantly a good practice in dealing and arguing with customers of different profiles.

The next section presents some related projects dealing with the immersion of a user in a populated virtual environment and the training of a learner to business-related activities. Section 3 introduces multiagent systems and details the interaction-oriented methodology on which Format-Store is grounded. Section 4 describes how a multiagent simulation of virtual customers is wrapped into a serious game. Some outstanding features like scenario integration and adaptive difficulty are detailed.

2 Related Work

Populating a virtual environment with artificial characters for the sake of experiential learning has been investigated in many games or projects, for example for visiting the no more existing Pennsylvania station [23] or discovering the life of Romans in ancient Pompeii [16]. In the Metropolis project [17] or Roma Nova [18], an emphasis is put on the crowd's ability to acknowledge the presence of the player, by means of gaze behaviour or basic dialogic interactions, in order for them to feel part of the population. Yet, the integration of the player is way beyond the requirements of a business training application, where expert-designed scenarios and dialogues constitute the core of the learning activity. The three following projects investigate how training may occur between a learner and a virtual customer.

Knowledge Drive is a serious game developed by Caspian Learning [1] for Volvo Car UK and aimed at replicating the experience of an actual showroom. The main objective of the game is to train salesmen to the products sold by Volvo but also raise their awareness regarding the legislation. In a 3d environment (see figure 1.a), the learner meets virtual customers and builds profiles on the basis of clues they give during dialogues. As the learner makes assumptions regarding

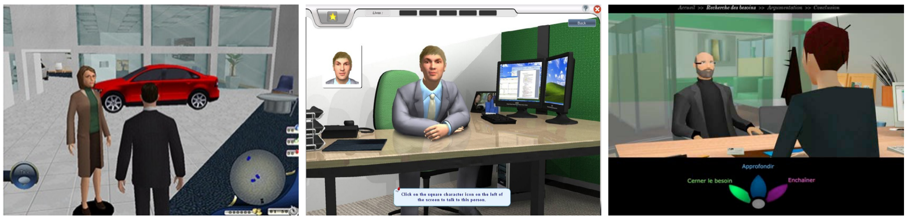


Fig. 1. (a) “Knowledge Drive” from Caspian Learning replicates the experience of an actual car showroom where the learner builds a line of argument to eventually close the deal. (b) The “Sales Game” from PIXELearning broadens the activity of the learner by enabling them to manage a professional network or a customer database. (c) The “BCV bank” project trains advisors to argue with scripted customers about financial products and bank services.

the profile and the expectations of the customers, he/she is expected to identify the right car for them and build an appropriate presentation by ruling out the irrelevant arguments from an initial argumentation. Law breaking scenarios are introduced in the argumentation; they must be identified and discarded by the learner. The Sales Game by PIXELearning [2] spans a broader range of missions related to business. In addition to sales training (depicted in figure 1.b), the learner is expected to attend virtual meetings, meet virtual colleagues, manage a professional network and build a customer database. Gaining in experience, the learner increases their knowledge and their skills and competes for the salesperson of the year election. Another game of interest has been developed by Daesign [3] for the Cantonale Vaudoise bank (BCV) to train customer advisors selling financial products and services. The game reenacts an interview with a virtual customer. Although the development of the dialogue is mostly scripted (greeting the customer, analysing the needs, arguing with the customer and closing the deal), the player must select at each significant step of the interview one option among several attitudes: analyse, elaborate, carry on (figure 1.c).

Although all the games cited in the previous paragraph focus on the CRM only, several reasons make them particularly interesting for the scope of this paper. Firstly, they point out the many advantages of teaching the relationship with the customer using an interactive simulation over relying on traditional teaching methods. Also, they demonstrate the usefulness of a game in complement of a knowledge base – the aforementioned games are bundled with a traditional learning platform – for the knowledge to be contextualised and translated into skills by the learner. Indeed, although a LCMS enables the content to be personalised to the learner, the knowledge is neither personally constructed nor applied. A game offers this opportunity as the learner uses it as a playground where new skills can be tested and old ones can be rehearsed.

The ambition of Format-Store is to combine the capacities of an immersive environment populated with intelligent customers (unpredictable expectation and needs of the customers, necessity for prioritising the tasks, ability to identify a customer needing information or help) to a tutoring system where scenarios can be designed in order for the domain experts to control the learning experience. Those requirements advocate for using a robust and modular artificial intelligence system, not only able to model the coherent behaviour of a customer but a whole crowd of them. Multiagent systems represent a suitable answer to this problematic, particularly the interaction-oriented approach detailed in the next section.

3 IODA: An Interaction-Oriented Multiagent Design Methodology

A multiagent system (MAS) [27] is an organised set of entities called agents interacting in an environment. The term was coined early in the 1990s and encompasses every simulation of a complex phenomenon where interacting particles can be identified.

Multiagent systems have been gathering an increasing interest lately as an alternative approach to mathematical modelling which more traditionally aims at modelling a phenomenon with equations. The main interest in MASs is the ability to consider a complex phenomenon as the – often emergent – result of simple agents interacting with each other. In this context, MASs offer the ability to apply a bottom-up methodology by locally defining the role and the behaviour of each agent participating in the global phenomenon. Besides, MASs help understanding the contribution of each agent whereas a mathematical model can only describe the global mechanisms of the phenomenon. Application areas of MASs include simulating natural phenomena like molecular biology [9] or ecosystems [7], animating artificial worlds [6], video games or computer-generated imagery [4], social behaviour [10], economy [5] or disaster management [19].

Individual-Centred Simulations. Whereas classical simulations aim at modelling a phenomenon with one or several mathematical equations, MASs focus on the individuals participating in the phenomenon. The question MASs address is: knowing the entities participating in a given phenomenon, what must be the behaviour of each individual for the whole phenomenon to demonstrate a desired property?

To solve Artificial Life (AL) problems, MASs have proved well suited for modelling emerging collective phenomenons composed of simple interacting individuals. Reynolds [22] has shown that a visually realistic flock of virtual birds can be obtained by applying a local behaviour composed of three simple rules to each flockmate. Another famous illustration of emergent complexity is provided by Resnick [21] with his simulation of ants and termites foraging behaviour. The common point of these simulations is the proof that self-organisation can emerge without the need for a supervising body provided the agents are locally endowed with an appropriate behaviour and means to communicate with each other.

As a consequence, simulating a phenomenon using a MAS does not require one to understand the phenomenon but merely observe the agents participating in the phenomenon. An individual-centred methodology thus offers a great advantage over a more complex approach where the global description of the phenomenon is necessary. Besides, the principle of emergence remains valid regardless of the reactive or cognitive nature of the agents.

Reactive and Cognitive Agents. All the agents in a MAS support common characteristics although some properties are inherent to different kinds of agents.

An agent is always situated in an environment according to one or several metrics attached to the environment. Possible metrics can be a distance in an Euclidean space or relationships in a social network for instance. Relying on the metric, an agent has a neighbourhood and is itself located in the neighbourhood of other agents. Each agent has a local perception of the environment, usually restrained to its neighborhood. Similarly, an agent is allowed to act or interact locally with the other agents in the environment. Interactions can be direct between two agents or indirect when a media (usually the environment itself) is necessary for a message to be conveyed from one agent to another. Finally, the behaviour of an agent is defined by an internal perception-decision-action loop. The agent is said autonomous as the decision is locally and internally taken by the agent itself.

Traditionally, a dichotomy is made between reactive and cognitive agents. Reactive agents use a trivial decision process that mostly consists in triggering an action according to a perception. Cognitive agents are proactive and plan actions in the long term to achieve internal goals or objectives. The Belief-Desire-Intention model [20] is a good illustration of how such an agent works.

The MAS design methodology involves modelling the behaviour of the agents and their interactions. Depending on where the priority is set, two methodologies exist: agent-oriented programming (AOP, [24]) and interaction-oriented programming (IOP, [25,15]).

3.1 Ioda: Interaction-Oriented Design of Agents

IODA [14] is a multiagent simulation methodology whose originality is to focus primarily on how the agents interact instead of how they behave. This methodology is grounded on a simple observation coming from experimental experience. The design process a simulation always involves making assumptions at some point, irrespective of the phenomenon to simulate. In the context of a multiagent-based simulation – e.g. when the actual phenomenon is the obvious outcome of multiple interacting entities – the description of these interactions is the only objective assumption one can formulate. The set of processes occurring "inside" each entity and leading to the interaction can only be guessed. In order to avoid introducing a bias too early in the process, a safe approach would consider setting the foundations of every model on observable characteristics, then building on this model with the constant concern of delaying the introduction of hypothetical assumptions as long as possible. IODA follows this principle by discarding the first hypothesis concerning the selection of which entity is an agent and which is a mere passive object, by providing a user-friendly tool (JEDI) for domain-experts to participate in the design of the interactions and by providing a tool for translating the model (Jedi-Builder) into a set of classes where the interactions can be implemented by a programmer.

3.2 Everything Is Agent

The starting point of designing a multiagent simulation consists in identifying the agents participating in the simulation. What defines an agent in a MAS is a minimum degree of behavioural autonomy and the subsequent ability to trigger autonomously an action or an interaction. Historically, "living" characters in a virtual simulation are considered as agents, "inanimate" objects like trees, furniture or items are not. Unlike other approaches, the first simplifying hypothesis of the Ioda methodology is to consider every entity involved in the simulation an agent. This choice is fully argued in [13]. Figure 2 illustrates the virtual supermarket environment used in Format-Store where every character – employee, customer – and almost every object – item, shelf, information sign, etc. – is an agent in the simulation. In the Ioda methodology, families of agents are listed in such a way that every agent in the environment strictly belongs to one family.

Considering every entity as an agent simplifies the first step of the MAS design, but also provides a convenient way to describe the interactions, as detailed in the next sections.

3.3 Interactions Made Concrete

In a similar way every entity in the simulation is represented by an agent family, any behaviour is described by an interaction in Ioda.

Unlike other MAS approaches, where an interaction is virtually expressed in the behaviour of two agents interacting together, each interaction in Ioda has a software tangibility and a central position in the design. An interaction is a rule involving two agents: it is performed by a source agent and undergone by a target agent. It is composed of two parts, a Boolean condition testing if the interaction can be triggered and an action part containing the actual interaction logic. Both these functions rely on generic primitives, left for implementation inside the agents or reused from a template library. This is formalised as follows:

```
InteractionName (Src, Tgt) := CONDITION:  condition primitives
                              ACTION:     action primitives
```

The condition and action parts of the rule rely on generic perception and action primitives so that interactions are independent from the concrete implementation of the agents.

As a consequence, Ioda exhibits two unique features. Firstly, the interactions can be represented independently from the agents, as libraries of interactions for instance. Interactions are reusable from one agent family to another and from one simulation to another. They can be allocated to agents in a plug and play fashion. The other advantage of interactions reification is the ability for all the agents to be processed by a generic engine through a single iteration loop, irrespective of the nature of each agent.

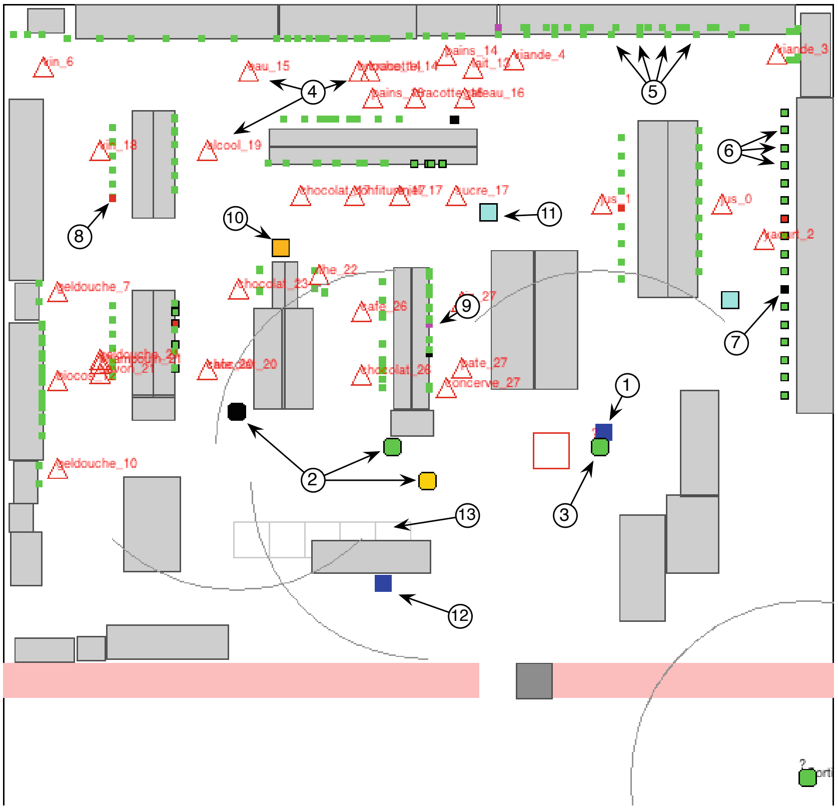


Fig. 2. Almost every object in the environment is an agent on the same account as the characters. In addition to simplifying the design process, considering every object as an agent also allows spreading the "intelligence" of the characters across the objects they interact with. The figure represents a Java AWT rendering of the multiagent simulation used during the development. The agents are: (1) employee; (2) customers with various satisfaction levels; (3) conversing customer; (4) information signs; (5) items; (6) popular items; (7) out-of-stock item; (8) damaged item; (9) expired item; (10) crate; (11) stain on the floor; (12) checkout cashier; (13) waiting queue.

3.4 Interaction Matrix

Having defined the agent families participating in the simulation, the next step of the Ioda methodology consists in describing their interactions. This is achieved by allocating the interactions in a matrix named the interaction matrix. Figure 3 presents the version of the interaction matrix used in Format-Store. All the agents participating in the simulation are listed in the matrix along with their mutual interactions. Note the avatar of the player is included in the matrix like any other agent. In the matrix, each interaction receives two additional parameters.

– the distance defines the minimum distance between the source and the target for the interaction to be allowed. For instance, seizing an item on a shelf requires the character to be standing in front of the item; a character can read an information sign from a certain distance; etc.
– the priority is used for sorting between several interactions whose preconditions are verified.

Source/Target	$\emptyset$	Employee	Customer	Door	Sign	Checkout	Item	Queue	Stain	Crate

Employee	Converse(0)		StartConversation(1,0)
			EndConversation(1,0)
							Remove(1,0)
							Supply(1,0)
							Order(1,0)
									Clean(1,0)	PutAway(1,0)

Customer	Wander(0)
	GoTowards(1)
	Converse(13)
		Wait(2,3)		Exit(1,12)		Pay(2,10)	Get(2,5)	StepIn(5,7)
								MoveOn(1,8)
								WalkOut(1,11)		

Door	SpawnCustomer(1)		Acknowledge(10,0)							
Sign			Acknowledge(10,0)							

Checkout			Acknowledge(10,0)
			CheckOut(2,0)							

Item	Expire(1)
	MakeStain(1)
	SpawnCrate(1)
			Acknowledge(10,0)
			Upset(1,0)
			Ack_OutOfStock(1,0)							

Queue										
Stain			Upset(1,0)							
Crate			Upset(1,0)							

Fig. 3. The interaction matrix presents the interactions allowed for any agent family as a source towards any other agent family as a target (including self) or the environment (degenerate interaction, column $\emptyset$). How to read this matrix? For the Customer agent family for instance, the column labelled Customer lists all the interactions of which a Customer is a target. In this example, the Customer is basically informed by almost any other agent (of the location of Items by a Sign, of the price and quantity of an Item by this Item, etc). The row labelled Customer lists all the interactions of which a Customer can be the source. Priority values (n) are used when several interactions can be applied at the same time. The higher the value, the higher the priority.

Although the interaction matrix offers a simplistic representation of the simulation model, the the behaviour of each agent is exhaustively described. We argue that this representation is functionally equivalent to a more complex algorithm, put aside the difficulty for a non computer scientist to read the latter.

3.5 Computer-Aided Design for Non Experts

We have mentioned earlier in this paper the importance of involving domain-experts as far as possible in the design process. IODA follows this principle by proposing a simplified multiagent methodology made accessible to non computer scientists. The implementation stage has received the same attention with two additional elements of the IODA methodology: JEDI and JEDI-Builder. JEDI is an application programming interface (API) providing a set of Java classes for a user to ensure the rigorous implementation of their IODA conceptual model. JEDI-Builder is a Java application assuming two roles. Firstly, JEDI-Builder provides a computer-aided design tool for the user to model the interaction matrix using a user-friendly graphical interface where agents and interactions can be added in a drag-and-drop fashion. Secondly, JEDI-Builder is also able to translate such a IODA-compliant matrix into a JEDI application that can thereafter be used as a simulation or integrated in a larger project. Using these tools enables a user to implement the major part of a MAS without any particular knowledge of computer programming. At the end of the process though, a computer-scientist is required to implement the core of each interaction, if those interactions are not part of an already existing library – which is often the case as JEDI natively includes a set of predefined generic interactions.

The implementation of an interaction can take many shapes, from the most trivial to more complex tasks. For instance, in FORMAT-STORE, the player as an employee can supply an item on a shelf or converse with a customer. The interaction "Supply" simply consists in increasing the quantity value of a target item. In contrast, the interaction "Converse" involves more complex operations. The camera position is changed for a closer look on the customer. A dialogue window is opened and a script is started. This script displays a narrative text and several possible answers among which the player has to choose one. Depending on the answer, the satisfaction level of the customer decreases or not and the player is rewarded (positively or negatively).

Although complex interactions like the dialogue situation seem at first glance the most useful in terms of behaviour, the very interest of multiagent simulations lies in using multiple simple interactions, as explained in the next section.

3.6 Adaptive Behaviours

This interaction-oriented vision offers an original implementation of the concept of affordances. In his influential book [11], Gibson states that the interaction capabilities – the functions – of any object in a real environment are mainly suggested by the object itself – its shape, position, etc. One interpretation of the concept of affordances has been used many times since in character animation, where computer graphics scientists have found more intuitive to attach the animations and the algorithms necessary for an interaction to the target of this interaction. For instance, a virtual character wanting to open a door would be acknowledged by the door itself of the position it should stand at and the animation it should play. Affordances are implemented in most simulations or games featuring at least one virtual character interacting with objects in the environment. That way, every passive object describes to the character how it should be handled, under which circumstances and what is the outcome of the interaction. The character is therefore freed from that knowledge, putting a focus on managing internal goals and searching the environment for objects likely to solve these goals. Another benefit of using affordances is to avoid any glitch in the animation by carefully adjusting the positions of the interacting entities and synchronising their respective animations. In such simulations though, the character is always at the origin of every interaction. Whichever interaction is triggered is the result of a complex and often time-consuming decision process selected by a cognitive controller.

The idea that the cognitive abilities of an agent may be spread as well among the interactions offered to the agent is part of the multiagent approach, and particularly of IODA which considers that an intelligent behaviour can be expressed by a reactive agent. This argument is close to earlier research in artificial intelligence postulating that intelligence is not the result of planning and reasoning – like a cognitive agent using scripts and complex algorithms – but on the contrary that the mere appearance of such cognitive processes is the result of an agent's reactive behaviour in a dynamic environment [8].

To illustrate this principle, let us consider the behaviour of an intelligent customer shopping in FORMAT-STORE: A customer is endowed with a profile – 10 different profiles were provided initially by the client, which basically corresponds to 5 age ranges times 2 genders – and a shopping list containing items to be purchased. The number of items on the list depends on the customer's profile.

A virtual customer entering the store will consider every interaction in decreasing priority until one is realisable (e.g. the preconditions are verified). Getting out of the store is only realisable when all the items have been paid. Paying requires first to queue at the checkout (actually checking out is managed by the checkout agent). Queuing can be started when all the items in the shopping list have been retrieved. An item can be taken when the customer is actually standing at reach of hand of the item. Moving towards an item relies on standard pathfinding and navigation algorithms but requires knowing the location of the item, which is acknowledged after an interaction with one of the information signs located throughout the store. While shopping in the store, the customer seems to follow a plan. Yet, every action is independent from the following and their sequence has merely been established by selecting priorities.

The adaptive behaviour of the customers is illustrated by the way they select the items. An agent does not embed a map of the product items. The location of every item is provided by information signs located throughout the store, while the customer is wandering or shopping for already known items. That way, selecting the next item – in practice, the closest item whose location is known) depends on the customer's current knowledge, which depends in turn on the signs the customer has already interacted with. In addition, customers dynamically respond to obstacles in the store like a crate obstructing an aisle, a stain on the floor, or another customer standing in their path. They will therefore always take different paths, even when their shopping lists are similar. Figure 4 illustrate this behavioural differentiation within the virtual customers.

In addition to being adaptive, the behaviours expressed are also robust. Disturbances can be created in the game by adding/modifying/removing signs without affecting the ability for each customer to behave coherently. Ultimately, the customers can be placed in a different store and still manage to shop for goods.

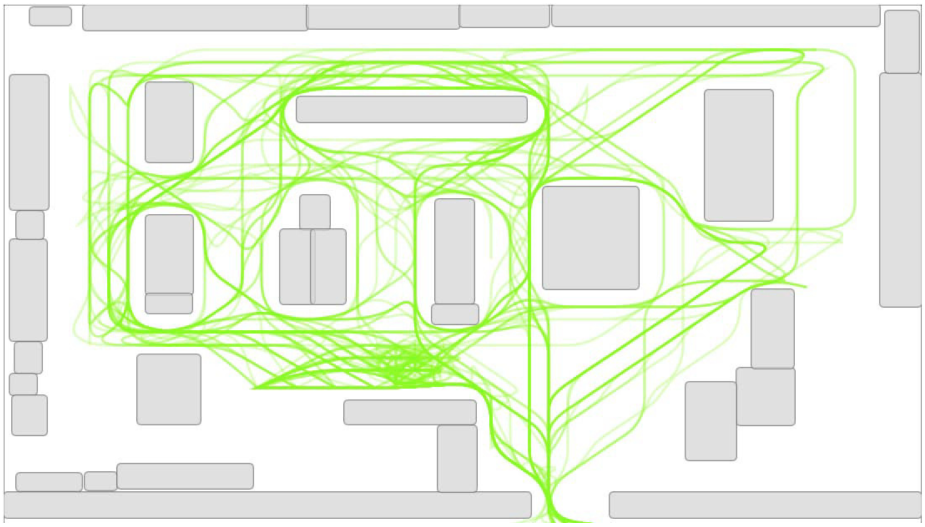


Fig. 4. Recording and drawing the position of each customer during the simulation illustrates how varied routes are obtained owing to the mere attribution of different shopping lists, however relying on IODA's adaptive planning of each customer's behaviour

4 A Serious Game for Immersive Training

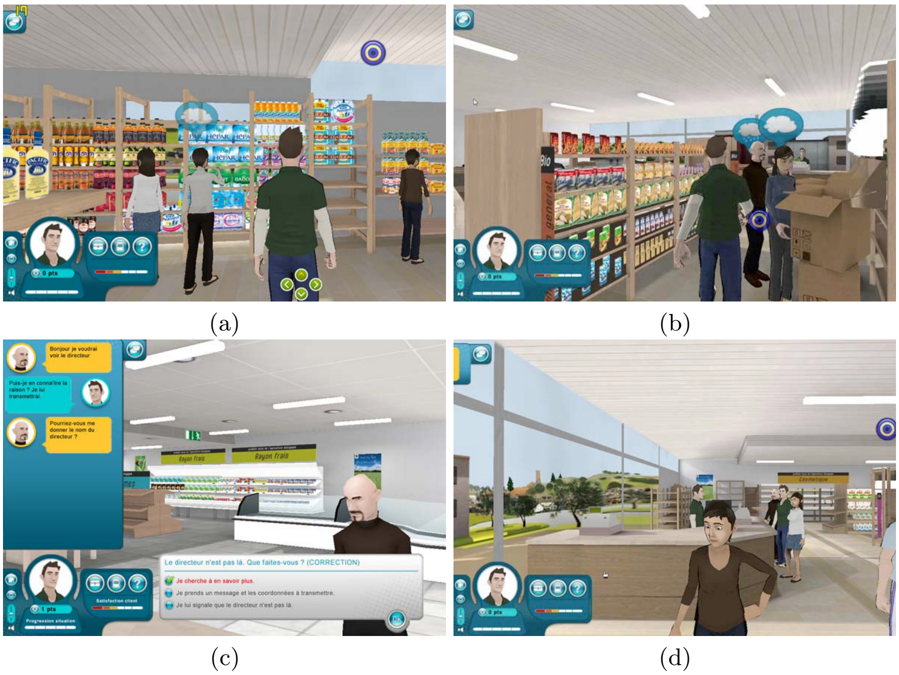
The FORMAT-STORE serious game is grounded on a multiagent simulation of the customers and the virtual store. The process of transforming this simulation into a game consists in a few steps: i) integrating the player into the simulation taking the control of one agent, namely an employee, ii) wrapping the simulation using a game manager for controlling the user scenarios and iii) plugging a performance analyser responsible for the scoring and the adaptive difficulty. All these steps are detailed in the following sections.

4.1 Presentation

Towards its inclusion in the LCMS (see section 1), the FORMAT-STORE serious game is available online. The high-end graphics required by the 3d environment are provided by X3d (formerly VRML) technologies. The game requires a – freely available – plugin at the user's end but displays in return compelling 3d graphics, yet computationally efficient enough to allow for the game to run on an Internet browser.

4.2 The Human in the Loop

The result of the IODA methodology applied to the FORMAT-STORE requirements is a multiagent simulation of a store filled with goods and populated with



Bonjour je voudrai voir le directeur

Puis-je en connaître la raison ? Je lui transmettrai.

Pourriez-vous me donner le nom du directeur ?

Le directeur n'est pas là. Que faites-vous ? (CORRECTION)

- Je cherche à en savoir plus.
- Je prends un message et les coordonnées à transmettre.
- Je lui signale que le directeur n'est pas là.

Fig. 5. Format-Store features a virtual organic store populated with autonomous customers. (a) The player can control the avatar anywhere in the store using the arrows on the keyboard or graphic controls on the screen. (b) The virtual customers navigate and shop autonomously inside the store. The condition of the store and the items impacts their level of satisfaction. (c) Conversational interactions in Format-Store are rendered by a specific GUI during which the player can select appropriate answers. (d) The realistic behaviour of the customers is reinforced by their ability to avoid colliding with one another or to queue at the checkout.

customers shopping autonomously and seeking for assistance. This multiagent simulation, at the core of the serious game, is best described by the interaction matrix on figure 3 and the adaptive behaviour of the customers explained in section 3.6.

Using the arrows on the keyboard or a graphic user interface (GUI), the player is enabled to move their avatar (namely, the employee) freely in the virtual store (figure 5.a). Using the mouse, the player can also interact with many elements of the store including the product items and the autonomous characters – customers and other employees. In that latter case, a specific GUI is loaded as the player enters a conversational mode (figure 5.c) where they can select propositions.

The player is integrated in the game by means of controlling one of the agents (hence the presence of the employee in the interaction matrix) in the simulation, following a “letterbox” principle: actions from the player are captured, sent and expressed by the agent triggering the corresponding interaction. Conversely, interactions undergone by the agent are notified to the player. As a result, actions from the player are seamlessly conveyed in the simulation, preserving the autonomy of the agents and the independence of the action selection mechanism. On the other hand, the controlled agent introduces perturbations which, combined to the behavioural adaptivity of the customers, fosters a great variety on the situations presented to the player.

Having implemented the simulation part of the game, where the user is free to wander in a living replica of a store, the next part towards achieving a game is to introduce scenarios for the player to actually learn in this realistic context.

4.3 Game Management and Scenarios

Traditionally in games where the player faces virtual characters, unitary scenarios defined in accordance with the educational requirements are represented by the specific behaviour of virtual characters. For instance, in the salesmen training games presented in section 2, each character represents a pedagogical situation to explore (advise the right product in accordance with the client's profile, cope with a customer difficult to argue with, etc). In practice, each character embeds a script describing either a specific dialogue or a predefined sequence of actions/interactions.

In Format-Store, we put an emphasis on the game's scalability, namely to what extent the client will be able to mend or remove existing cases or add new pedagogical elements. We explore an original approach taking advantage of the behaviour's adaptiveness in a multiagent system. The customers wandering in the store at any time are merely going to their business – shopping for goods – trying to fulfil internal goals – purchasing items on a shopping list or querying for information – instead of following a scripted behaviour. In this context a scenario is not attached to a specific character but rather consists in attributing goals to the customers or introducing disturbances in the environment. For instance, an item a customer is looking for can be removed, a customer can be introduced in the store with a question to ask, an information sign can be misplaced, an aisle can be made impassable by an oil stain, etc. As a result of the agents' adaptive behaviour, one or several customers will be affected by the trouble introduced by the scenario and inspire a specific reaction, ranging from being upset to complaining to the employee depending on their profile.

Problem-Situations. The notion of scenario is described by the project's content manager ENACO as a problem-situation, namely a problematic situation including a context and a branching dialogue investigating the different ways for the employee to deal with it. 25 problem-situations were initially provided by ENACO, addressing various issues such as a missing item on a shelf, an aisle obstructed by a stain or a box on the floor, or sale-related questions. Integrating a problem-situation in the game raises two questions. Firstly, how should the context be represented in the customer's behaviour? Secondly, how should the dialogue be integrated as part of the agent's abilities?

Representing the context of each problem-situation is easy using the interaction-oriented methodology IODA. The way a situation arises is implemented within the preconditions of one or more interactions, which must be defined on purpose. For example, customers try to meet the employee and start a conversation when they perceive the employee and have a question to ask. The preconditions are expressed as such. The scenario consists in adding a question to an entering customer. A second example considers that customers are upset and look into complaining when they perceive an expired item. The scenario in that case consists in expiring an item in the store. Problem-situations may require more complex scenarisation, which is achieved using the exact same method. For instance, when a customer attempts to get an item, there is a chance that it is dropped and makes a stain on the floor. When it is detected, the stain as an agent has the ability to upset other customers, which is the context for a specific problem-situation. In a similar way, when an item is missing on the shelves, the employee has the ability to supply it. Doing so, a crate is left on the floor, which can also upset a customer.

Integrating the dialogue itself is equally easy since it can be simply represented as an interaction. That way, the customer is enabled to converse with an employee, the same way it is able to get an item. The dialogues provided for each problem-situations have been digitised in an *ad-hoc* XML format, representing the various branches of a dialogue and the links from one dialogue line to another. The core of the converse interaction therefore consists in using a parser to load the dialogue. Both the employee and the customer are able to converse, and each agent loads the right part of the dialogue based on its current advancement. When several branches are available, the user selects one using a dialogue GUI whereas the customer selects one randomly. When a final branch is reached, the interaction is considered terminated. Depending on the branches chosen by the player, a score is attributed at the end of the dialogue. The score values are represented in the XML structure along with the dialogue lines.

All the events (customers with questions, expired items, probabilities of dropping an item, etc.) at the origin of the various problem-situations arising in the game are controlled by a single module called the game manager and organised within a game session.

Game Session. Designing a game session is inspired by the typical day of a salesperson in a supermarket. The duration of a game session has been fixed arbitrarily to 20 minutes for practical reasons, and therefore the many activities of an average day are condensed within this short time period.

The game manager is an autonomous module operating upstream and downstream the multiagent simulation. It decides when to send a new customer in the store, whether it has a question to ask or not, which items are on its shopping list or when to trigger a new event, depending on the current level of difficulty, and based on adjustable parameters. The level of difficulty corresponds to the maximum number of customers allowed in the store at the same time: the more customers in the store, the more difficult for the player to keep the store tidy and to help every one of them.

When a customer exits the store, the game manager collects feedback information in order to evaluate the player's performance and adjust the level of difficulty (discussed next section).

Scoring and Adaptive Difficulty. In FORMAT-STORE, the player is evaluated by the virtual customers themselves. During their activity in the store, between their entrance (on the game manager decision) and their exit (when their shopping is over), each customer undergoes the influence of the other agents. Every customer entering the store is attributed a level of satisfaction which depends on their profile (grumpy, friendly, etc). Every unfortunate event like trying to buy an expired item, finding an item out of stock, stumbling across a box or a stain, etc. has a negative impact on this level of satisfaction. When the customer is removed from the store, a difference in satisfaction can be computed. Knowing that every loss of satisfaction is due to the player (expired item not replaced, out of stock item not supplied, crate not removed, stain not cleaned, etc.) the difference in satisfaction of a customer translates the performance of the player. Indeed, when the player copes with the simulation, the store is tidy and the customers helped in time; and reciprocally.

Operating as a sub-module of the game manager, the performance analyser collects in real time all these satisfaction levels, along with the dialogue scores (when applicable). Based on this series, a mathematical equation is applied in order to compute the ideal level of difficulty, e.g. the maximum level where the player stays ahead of the tasks. When the difficulty increases, more customers are allowed in the store. When the difficulty decreases, some customers are not replaced when they exit the store.

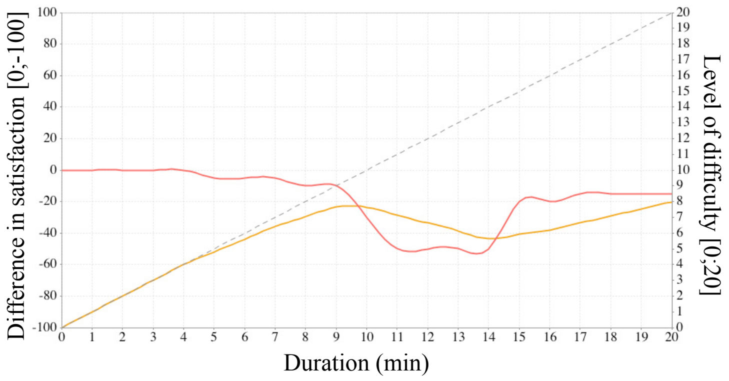


Fig. 6. The level difficulty (orange curve) is tailored to the player's performance (red curve) all along the game session. At minute 8, as the player fails coping with the simulation, the difficulty is decreased. The result is observed a few minutes later: the player gets back to grips with the game and the difficulty resumes its progression

Figure 6 shows how adaptive difficulty helps maintaining the challenge at reach of the player. When the difficulty is too high for the player to cope with, it is decreased until a normal performance is observed again. All along the game session, the player feels no frustration and is not tempted to quit the game prematurely. This is a critical point, we believe, and the next section intends to validate our approach.

Educational Validation. Despite the novelty of the technique, learning in a populated virtual environment can be supported by classical theories. In particular, two models readily apply to Format-Store.

Experiential or exploratory learning models [12] promote the free exploration and the autonomous and personalised construction of cognitive associations and understandings. The idea underlying Kolb's experiential learning is that a realistic virtual environment like Format-Store allows a contextualised learning, as opposed to the declarative and decontextualised learning provided by the LCMS. Format-Store provides means to learn from real life situations: elaborate a routine, learn to prioritise different assignments.

Socio-cultural models of learning [26] point out the fundamental role of social interactions in the development of cognition. Vygotsky's model replaces the social interaction with the teacher, and by extension with the virtual representation of a tutor, at the centre of any learning activity. In practice, this involves the constant delivery to the user of a feedback on his performance. A first requirement for the game designer is therefore to provide means to assess the player's performance and to guide them toward increasing their skills. The feedback can be delivered continuously in a game whereas it is hardly conceivable in real life. Another strong concept, which is a direct consequence of the initial idea, is to scaffold the learning by building new knowledge on top of the existing, whilst consolidating the latter. Vygotsky claims the utter importance of maintaining the learner in what he names the zone of proximal development (ZPD), situated beyond what the learner already knows – in which case the benefit is null – and below what he cannot achieve on his own – in which case no learning can happen but frustration, no matter how much help is provided. In the context of a game, adjusting the difficulty is a relevant way to control the position of the challenge in the learner's ZPD. Besides, a proportionate challenge provides a leverage on the engagement and the motivation of the player. Therefore, the second requirement commands the game designer to ensure that the level of difficulty is adaptive.

5 Conclusion and Future Work

The Format-Store project is a serious game aimed at training salesmen by immersing them in a dynamic virtual store populated with intelligent customers. In addition to its intrinsic originality, Format-Store brings two original features. Instead of scripting the agents, the educational scenarios attribute goals to the adaptive customers or define disturbances in the environment affecting their behaviour. The scalability of the game is therefore increased as changing the educational content can be handled by the content providers themselves. Another aspect concerns the uncompromising implementation of the affordances concept, allowing to simplify the visual animation of the agents, but above all to obtain intelligent behaviours in spite of the reactive nature of the agents.

Two other aspects are worth mentioning. Owing to a user-friendly design tool, domain-experts are maintained in the conception process farther than any other methodology, thus avoiding the premature introduction of biases and the translation of the formal model into a programming language is partially automated.

More generally, the methods and techniques deployed in Format-Storein terms of game management, scenarios integration and player immersion can be easily reused in other multiagent-based interaction-oriented serious games.

In the long term, we are considering the side development of a decision support tool for supermarket marketing strategy units. We intend to take advantage of our realistically-profiled crowd of intelligent customers in the context of product placement or shelves layout in a large supermarket.

Acknowledgment. The Format-Store project is supported by the French ministry of Economy, Finances and Industry under the “2009 Serious Game” scheme. In-game images are the property of Idées-3Com. The authors are grateful to Jean-Baptiste Leroy for his committed involvement in the project.

References

1. http://www.caspianlearning.co.uk/
2. http://www.pixelearning.com/services-the_sales_game.htm
3. http://www.daesign.com
4. http://www.massivesoftware.com/
5. Arthur, W.J., Holland, B., LeBaron, R., Palmer, T.P.: Asset pricing under endougenous expectations in an artificial stock market. Economic Notes 26, 297–330 (1997)
6. Bonabeau, E.: Agent-based modeling: methods and techniques for simulating human systems. In: Proc. National Academy of Sciences, vol. 99, pp. 7280–7287 (2001)
7. Bousquet, F., Le Page, C.: Multi-agent simulations and ecosystem management: a review. Ecological Modelling 176(3-4), 313–332 (2004)
8. Brooks, R.A.: Intelligence without reason. In: Proceedings of the 1991 International Joint Conference on Artificial Intelligence, pp. 569–595 (1991)
9. Desmeulles, G., Bonneaud, S., Redou, P., Rodin, V., Tisseau, J.: In-virtuo experiments based on the multi-interaction system framework: the RéISCOP meta-model. CMES, Computer Modeling in Engineering & Sciences (2009)
10. Epstein, J.M., Axtell, R.: Growing Artificial Societies: Social Science from the Bottom Up. Brookings Institution Press, Washington (1996)
11. Gibson, J.J.: The ecological approach to visual perception, Hillsdale, New Jersey, London (1979)
12. Kolb, D.A.: Experiential Learning: experience as the source of learning and development. Prentice-Hall, New Jersey (1984)

13. Kubera, Y., Mathieu, P., Picault, S.: Everything can be agent! In: der Hoek, et al. (eds.) Proceedings of the 9th International Joint Conference on Autonomous Agents and Multi-Agent Systems, Toronto, pp. 1547–1548 (2010)
14. Kubera, Y., Mathieu, P., Picault, S.: IODA: an interaction-oriented approach for multi-agent based simulations. Journal of Autonomous Agents and Multi-Agent systems (JAAMAS) 23(3), 303–343 (2011)
15. Kubera, Y., Mathieu, P., Picault, S.: Interaction-oriented agent simulations : From theory to implementation. In: Ghallab, M., Spyropoulos, C., Fakotakis, N., Avouris, N. (eds.) Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008), pp. 383–387. IOS Press (2008)
16. Maïm, J., Haegler, S., Yersin, B., Müller, P., Thalmann, D., Van Gool, L.: Populating Ancient Pompeii with Crowds of Virtual Romans. In: Proceedings of the 8th International Symposium on Virtual Reality, Archeology and Cultural Heritage - VAST (2007)
17. O'Sullivan, C., Cassell, J., Vilhjalmsson, H., Dingliana, J., Dobbyn, S., McNamee, B., Peters, C., Giang, T.: Levels of detail for crowds and groups. Computer Graphics Forum 21(4), 733–742 (2002)
18. Panzoli, D., Peters, C., Dunwell, I., Sanchez, S., Petridis, P.: Levels of Interaction: A User-Guided Experience in Large-Scale Virtual Environments. In: Proceedings of the Second Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES 2010), Braga, Portugal, March 25–26, pp. 87–90 (2010)
19. Querrec, R., Reignier, P., Chevaillier, P.: Humans and autonomous agents interactions in a virtual environment for fire figthing training. In: Virtual Reality International Conference, pp. 57–63 (2001)
20. Rao, A.S., Georgeff, M.P.: Modeling rational agents within a bdi-architecture. In: Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning, pp. 473–484 (1991)
21. Resnick, M.: Turtles, Termites and Traffic Jams. Explorations in Massively Parallel Microworlds. MIT Press, Cambridge (1994)
22. Reynolds, C.W.: Flock, herds ans schools: a distributed behavioural model. In: SIGGRAPH 1987. Computer Graphics, vol. 21(4), pp. 25–34. ACM Press, Anaheim (1987)
23. Shao, W., Terzopoulos, D.: Autonomous pedestrians. Graph. Models 69(5-6), 246–274 (2007)
24. Shoham, Y.: Agent oriented programming. Journal of Artificial Intelligence 60(1), 51–92 (1993)
25. Singh, M.P.: Conceptual Modeling for Multiagent Systems: Applying Interaction-Oriented Programming. In: Goos, G., Hartmanis, J., van Leeuwen, J., Chen, P., Akoka, J., Kangassalu, H., Thalheim, B. (eds.) Conceptual Modeling. LNCS, vol. 1565, pp. 195–210. Springer, Heidelberg (1999)
26. Vygotsky, L.: Mind in Society. Harvard University Press, Cambridge (1978)
27. Wooldridge, M., Jennings, N.R.: Intelligent Agents: Theory and Practice. Knowledge Engineering Review (1994)