Teaching Strategies to Millenial Students

Jean-Pierre Corriveau
School of Computer Science
Carleton University
Ottawa, CANADA
1-613-520-2600 x1192
jeanpier@scs.carleton.ca

Wei Shi
Faculty of Business and IT
UOIT
Oshawa, CANADA
1-905-721-8668 x3803
Wei.Shi@uoit.ca

ABSTRACT

Millenial students are very technology-aware and see technology as a necessity in most aspects of their life including learning. Traditional learning methods, in which the instructor largely controls the learning process, are not well adapted to such a clientele. Conversely, serious gaming environments offer complex and diversified approaches to *active* learning, which millenial students greatly appreciate. In this paper, we report on how one such environment, namely Second Life, was used to create a teaching center for a university course on business strategies.

Categories and Subject Descriptors

K.3.1 [**Computer Uses in Education**]: Computer Assisted Education.

Keywords

Serious games, Second Life, Business Strategies, Scripted Animations

1. INTRODUCTION

The current generation of students is often referred to as *millenial* students [15]. Generally, these students are very technologically literate and see technology as a necessity in most aspects of their life, including learning [12]. For example, consider the ECAR 2009 survey [5], which gathers information about how skilled undergraduate students believe they are with technologies; how they perceive technology is affecting their learning experience; and what their preferences are for Information Technology (IT) in courses. The authors of this survey report, amongst other findings, that *all* categories of students (especially millenial students who are early adopters of technology) have a strong preference to "learn through programs they can control (such as video games, simulations, etc.)"

This point is particularly important: Traditional teacher-centered learning methods and theories typically localize control almost exclusively 'in the hands' of the instructor (who dictates and verifies what is learnt and how and when it is). *Engaging* the students is therefore often reduced to a minimum. Similarly, any use of IT during lectures is also largely controlled by the instructor. In our opinion, it is this near-total appropriation of control by the instructor that clashes with the expectations of millenial students. Let us elaborate.

Millions of people, including countless millenial students, already spend time in virtual environments. For example, the virtual community Second Life [18] has over 9 million residents. Regular users spend an average of 22 hours online each week in these virtual communities [20]. This almost irresistible appeal of having one's self become *immersed* in a virtual world via an avatar has deep psychological origins. Near-total controllability is one of the most important aspects of such appeal. Consequently, a successful introduction of games and simulations in teaching depends first and foremost on the willingness of the instructor to relinquish some control while still carrying out the pedagogical mandate at hand. In particular, engaging millenial students requires that the instructor develop 'interesting' (i.e., immersive) material in the eyes of students. But this is not sufficient: no learning method is complete without proper evaluation of what Dali [3] calls the 'learning results'. And, in the case of millenial students, this presents a problem because another charateristic of these students is that they want to spend less time on tasks and reach success with relatively little effort [15]. Consequently, assignments and exams should downplay the notion of *failure* (and its negative repercussions in real life) and replace it with the notion of a *challenge* (which, is expected to be eventually met).

Thankfully, it is widely acknowledged that games greatly encourage (the avatars of) individuals to stand up to challenges that they might not have considered under similar circumstances in real life.

This suggests that a learning method targeted towards millenial students should rest on game-based learning. In this paper, we introduce such a method. First, game-based learning is explored in subsection 2.1. But what kind of teaching material would lend itself to such a method? What would be an environment conducive to student engagement and discovery? What form would an evaluation take? Those are the questions we investigated in a very specific context, namely, Second Life. More precisely, in the rest of this paper, we report on how Second Life was used for specifically teaching an undergraduate course on business strategies at UOIT. Thus, hereafter, we will use the term 'student' to refer to university students. We will first briefly summarize, in subsections 2.2 and 2.3 respectively, the key characteristics of Second Life and of the teaching material for the target course.

Then, in section 3, we will motivate and describe our teaching environment, and explain why it proceeds from the nature of the teaching material itself. We will also discuss what types of evaluations are supported. We conclude in section 4 with some final remarks about this whole experiment.

2. BACKGROUND

2.1 Beyond Traditional Teaching

Traditional teaching/learning environments (be they industrial or academic) are often viewed as 'boring' by millenial students (due to their typical lack of engagement from the students). As early adopters of technology, such students exhibit a new mindset towards learning: for them, *doing* is preferred to knowing, and achieving interactive, experiential learning is a necessity for their educational success. Furthermore, as previously mentioned, they appreciate the use of technology in learning.

In contrast to traditional teaching/learning environments in which control rests almost exclusively with the instructor, online video games offer a learner-centered approach to learning: the learner/player controls the learning process through interactions with the game. In particular, Kolb [8] who makes a most important remark when he states that games inherently support experiential learning by providing students with concrete experiences *and* active experimentation. For us, such an observation is especially important because, in our specific context of a university undergraduate course, insisting on experimentation indirectly emphasizes the necessity for an evaluation component in a game-based approach to learning. In turn, this suggests that one category of games, namely so-called 'serious' games, may be best suited for game-based learning in such a context.

Although there is no precise definition of what constitutes a 'serious game', this expression typically refers to online games that are used for training (e.g., medical, nursing), simulation or education purposes. Such games aim to provide highly sophisticated simulations for specific domains and their scenarios. They present such scenarios in a complex interactive narrative context that is coupled to interactive elements that are designed to engage the trainees. Generally, goals and challenges require trainees to solve domain-specific problems that the trainees are unlikely to have encountered, thus increasing both interest and immersion. In fact, the high level of sophistication offered by simulations in serious games allows trainees to be exposed to complex situations that would be difficult, if not impossible, to recreate in reality (due to a number of factors including cost, time, ethics, safety, etc.). Thus, beyond meaningful play for training purposes, serious games readily support the notion of evaluation through the use of challenges that confront trainees with possibly novel problematic scenarios that require solutions drawing on newly acquired skills, while being engaged in the game.

For example, Michael and Chen [14] discuss an experiment in which one group of students was taught in the traditional instructor-led way, while another was taught the same concepts through a serious video game. These authors first observe that students learning through the use of a serious game were found to be extremely engaged in the subject matter and much more attentive than those in the other group. Second, the work of these authors leads them to conclude that serious games offer not only improved self-monitoring, problem recognition and problem solving, but also improved short- and long-term memory, as well as increased social skills and increased self-efficacy! Other research on the role of serious games in education (e.g., [2, 16]) support similar conclusions: such rich simulation and gaming environments (such as Second Life) offer complex and diversified interactive approaches to learning and outcomes and, most importantly, foster *active* learning, which, in our opinion, is a fundamental expectation of millenial students.

The question then is how to exploit the potential of Second Life for teaching millenial students.

2.2 Second Life

Second Life [18] is a free online virtual world that is both mature and familiar to many of the students of the millennial generation. In addition, several major corporations (e.g. IBM) have chosen Second Life as their platform for the 3D Internet.

In essence, Second Life is a massive multiplayer online simulation environment (as opposed to a 'game' *per se*). Its nature lends itself to a multitude of immersive pedagogical usages [4, 19]. Moreover, the infrastructure behind Second Life eliminates many problems that less-used tools and environments face. In particular, having to support literally millions of simultaneous users, the server architecture of Second Life greatly contributes to ensuring availability and reliability of a learning environment (unlike 'home-made' and/or 'experimental' pedagogical tools and simulations). Moreover, because users can share in real-time a common interactive environment, Second Life fully supports user collaboration for shared tasks, a capability that is relevant to the design of assignments/projects.

In the context of this paper, a crucial ability of Second Life is its support for the creation and experiencing of *scripts*. More precisely, Second Life comes with LSL, the Linden Scripting Language [10], an event-oriented programming language for specifying the behavior of objects in Second Life. It must be emphasized that LSL provides general purpose programming semantics comparable to those of programming languages such as Java (including type checking and casting!). In fact, it is possible to program extremely rich and diversified scripted *animations*.

Scripts are pervasive in Second Life as they capture how an object looks, moves, communicates and interacts with avatars. Second Life also allows the importation of images, videos and animations from external sources. Ultimately, the overall functionality of Second Life supports *experiential* learning, which is extremely arduous to achieve by hand or with slides, and is generally prohibitively time-consuming to attempt with stand-alone tools (whose learning curve may be very high). Furthermore, it must be emphasized that Second Life also provides a test harness [11] for LSL scripts, another capability that is relevant to the design of assignments/projects/exams.

2.3 The 36 Stratagems

The “Business Strategies for Professionals” course at UOIT examines the notion of strategy and its related concepts. The focus is on strategic management: choosing and/or designing a viable strategy, and monitoring strategic performance. The main characteristic of this course is that it discusses at length how to apply traditional Chinese military strategy to business. More specifically, the course investigates the relevance and applicability to the business world of the *Thirty-Six Stratagems* put forth by Sun Tzu in his famous book "The Art of War" [23]. The *Thirty-Six Stratagems* are divided into six groups containing six stratagems each.

The first three groups generally describe strategies for use in advantageous situations, whereas the last three groups contain stratagems that are more suitable for disadvantageous situations. These stratagems take the form of 36 Chinese proverbs related to 36 battle scenarios in Chinese history and folklore, predominantly of the Warring States Period and the Three Kingdoms Period.

Each stratagem (in bold below) is typically accompanied by a short domain-specific (e.g., war) description (in italics below). For example, stratagem 28 [21]:

Remove the Ladder when enemy has ascended to the roof

With baits and deceptions, lure your enemy into treacherous terrain. Then cut off his lines of communication and avenue of escape. To save himself, he must fight both your own forces and the elements of nature.

Most importantly, in the context of the target business course at UOIT that uses our learning environment, we remark that several authors have explored the relevance and applicability of the 36 stratagems to the business world (e.g., [9, 13]). On the one hand, some researchers have focused on producing specific examples (which we will call *instantiations*) of these stratagems in the domain of business. On the other hand, other authors [22] have aimed at replacing the short war-specific descriptions provided by Sun Tzu for each stratagem, with short descriptions specific to the domain of business.

As will be explained shortly, our proposal rests in part on this distinction between the a) a stratagem, b) its description in a particular domain, and c) specific examples of the use of this stratagem in a particular domain.

3. TEACHING PATTERNS WITH SECOND LIFE

3.1 About the Teaching Material

The design of a learning approach may proceed in many ways. The approach we propose stems from the analysis of the teaching material we carried out and now summarize.

Initially, as suggested in subsection 2.3, we can classify what we must teach into 5 categories:

- the names of the 36 stratagems
- a description of each stratagem in the domain of war
- a description of each one in the domain of business
- instantiations of each stratagem in the domain of war, and
- instantiations of each stratagem in the domain of business.

Semantically, the domain-specific description of each stratagem serves as a factory (or generator) for the production of one or more distinct specific examples (thereby explaining why we shall refer to such examples as instantiations of a stratagem in a domain).

From a pedagogical standpoint, the task at hand is one of learning by analogy [7]. Cognitively, analogy is the process of transferring concepts from one domain to another [*Ibid*]. So, in the context of our business strategies course, learning proceeds from first 'mastering' the 36 stratagems in one (source) domain (the war domain), and then being able to use this new knowledge to conceptualize the same stratagems in a target domain (the business domain). This process imposes a partial temporal flow to the order of presentation of the material: first a strategy's name and its source-domain description, then examples of this strategy in the source domain, then and only then the target-domain description and its corresponding examples (in no specific order). As will be explained later, this temporal flow also influences the nature of evaluation of the learning results in our proposal.

For now, it is important to understand that adopting “learning by analogy” as the pedagogical foundation of our approach does impose restrictions on how the stratagems/strategies are to be presented. More precisely, we are constraining all descriptions and instantiations of the stratagems to be expressed with respect to either the source or the target domain. In other words, it will not be acceptable to have some of the 36 stratagems be illustrated using one source domain (e.g., armies at war) and others using a distinct source domain (e.g., courtiers intriguing). Analogy is about a conceptual transfer from a *single* source domain to a *single* target domain. Thus, all descriptions and examples must be consistent in their commitment to a same (source or target) domain.

“Learning by analogy” also leads to a key aspect of our proposal: the idea of applying a set of strategies to a source and then to a target domain suggests that this set of strategies forms what Christopher Alexander has called a *pattern language* in his seminal work in architecture [1]. There are two consequences to thinking of Sun Tzu’s stratagems as *patterns*:

First, the notion of patterns has been researched in several different fields. In software engineering, the work of Gamma et al. [6] has emphasized the need for the standardization in the presentation of patterns. This is highly desirable from a pedagogical standpoint: using a specific format for the description of the stratagems (e.g., name, short description, motivation, structure, participants, interactions, and tradeoff analysis, as in [6]) should improve memorization, as well as ease comparison between these patterns.

Second, given a domain, the description of each pattern defines *de facto* a domain-specific *script* in the sense used by Schank and Abelson [17], namely: a generalized episode (in this case of a stratagem). Semantically, for these authors, a script defines some *roles* (i.e., participants) and a sequence of *actions*. Consider again, for example, the previously mentioned stratagem 5: **Loot a burning house**. In the domain of war, all instantiations of this stratagem would follow a script such as:

Identify roles 'looter' and 'enemy'
Enemy is in weak state
Looter attacks enemy
Looter eliminates enemy

Script 5W: Loot a Burning House

This script should be viewed as a blueprint (or factory [6]) from which all examples in this domain must be instantiated. From this viewpoint, scripts indeed fit well in Alexander's [1] conceptualization of patterns as *generators* (of instances/instantiations in a domain). In fact, scripts correspond to what Gamma *et al*. [6] capture in an interaction diagram between the different participants of a pattern.

We suggest that conceptualizing a strategy (or stratagem) as a script allows us to refine what we mean by 'learning by analogy'. In the context of our target course, understanding any one of the 36 analogies will consist in 'transferring' the script associated with one of the 36 stratagems from the domain of war to the domain of business. It is possible that the transfer reduces to the complete reuse of the source script (i.e., the names of roles and actions remain unchanged). Most often, however, while the ordering of the steps of the script remains the same, roles and actions will be relabeled from the source domain to the target one. For example, for stratagem 5:

- The 'looter' becomes the 'hostile company' and the 'enemy', the 'competitor'.
- Step 2 merely requires the target competitor to be in a weak state (i.e., no need to refer to a military state).
- The action 'attack the enemy' in the source script can remain the same, or can be associated with one or more refinements (i.e., specializations) of itself in the domain of business (e.g., 'perform hostile take-over of competitor', 'seek bankruptcy of competitor', etc.).
- Finally, 'Looter eliminates enemy' becomes 'Hostile company puts competitor out of business'.

The key requirement is that the steps of the script in the source domain remain the same in order for the analogy to be easily conceptualized. Without such a constraint, that is, if we allow the source script and the target script to be quite different structurally, then the notion of *analogical transfer* is considerably blurred and, in fact, the crucial point that the two scripts are for a same strategy may be lost!

It must be emphasized that the onus is on the instructor to develop, in both the source and the target domain, scripts and examples derived from them. These should be conducive to learning, that is, they should highlight the commonalities *and* the analogical transfer between such scripts (and their corresponding examples).

Furthermore, in the learning approach we propose, scripts constitute the ideal vehicle for immersion through animations. We explore this idea next.

3.2 Animated Scripts for Immersion

Our pedagogical proposal for the game-based analogical learning of strategies rests on engaging students through the use of animations illustrating the scripts associated with such strategies. This is not a trivial task. In particular, the description, scripts and examples of the strategies to teach must be domain specific. Since animations are to be associated with scripts, it follows that animations too must be domain specific. We take this requirement to entail that all animations in a domain must take place in the same animation context. For example, for the domain of war, we require that all animations be set in the context of a *battlefield*.

Deciding on a domain-wide animation context is a difficult choice that an instructor (as subject expert) faces. In essence, the challenge is to have all scripts of a domain 'fit' this shared animation context. For example, in the domain of war, all scripts must have their roles and actions expressed in terms of two (or more) armies in battle. This can require a significant amount of creativity for some stratagems. Consider, for example, 'chaos' stratagems such as: 19: **Remove the firewood from under the pot** (i.e., steal someone's thunder) and 20: **Catch a fish while the water is disturbed** (i.e., create confusion and exploit it to your enemy's detriment). In practice, once the domain-wide animation context is chosen, then all roles and actions of all scripts must be 'fitted' to that context. Moreover, this is not a purely conceptual task: each action of a script must allow for one or more visualizations.

To illustrate this discussion, let us go back to the script given in 3.1 for stratagem 5. Step 2 states that the enemy reaches a weak military state. But, as is obvious from the short description for this stratagem given in 2.3, 'famine', 'corruption' and 'crime' may be causes for the weak state of the enemy. A script being a generator of examples, it is important that it is written with sufficient abstraction to carry out its purpose as a factory of examples. In other words, replacing step 2 in Script 5W with something such as:

Enemy is decimated by famine

is not desirable as it is too specific. More generally, the *causes* of actions are best left out of scripts and instead be used to generate specific instantiations of a script.

Most importantly, 'fitting' each role and action in a common context is not sufficient: the visualization of all such roles and actions must be addressed. This also requires creativity. For example, soldiers that are either starved, or corrupted, or murdered can use colors distinct from the one used to denote 'normal' soldiers in animations. More generally, the use of color to distinguish the possible *states* of some entity is widely used in games, software modeling tools, etc.

The visualization of actions is typically more challenging. Consider, for example, the following specification for an action of statagem 20:

***Spies from army on left of battlefield confuse soldiers on the front line of army on right of battlefield*.**

Such a specification illustrates several of the representational pitfalls to avoid in writing actions:

1) There should not be an overabundance of roles across the domain (if spies are one of the 50 roles used in the domain of war, and each role has a color associated with it, it is likely the user will not recognize the role from the color...).

2) Explicit references (such as 'army on left' of the battlefield) should be avoided: instead the names of the participants in the strategy should be used to improve learnability.

3) The expressions 'soldiers on the front line' is a role in disguise. It is overspecific: a script is a generator of examples and thus should not overcommit in terms of its visualization. For this stratagem, the action of 'confusing' the enemy should not be limited only to the 'front line' of the enemy army.

4) The action 'to confuse' is meaningless unless it is associated with some kind of animation. It is quite easy to capture (and remember) soldiers fighting and soldiers dying. The action of 'spies confusing enemy soldiers' is definitely less obvious to visualize, though always possible. Perhaps the spies could do 'jumping jacks' or make tables spin.

In the end, the instructor who creates a script must produce one that allows several possible instantiations. Consequently, assuming that the role named 'strategist' refers to the army that applies stratagem 20, the following wording for steps 2 and 3 of this stratagem:

2) Strategist creates confusion in enemy's army
3) Enemy is in weak state

is more desirable because step 2 can be associated with several animations (e.g., inducing enemy soldiers into a confused state[1] by having the strategist carry out confusing maneuvers such as going two steps forward and then two steps backwards repeatedly, or detonating one or more bombs in the enemy's positions) and also because step 3 is being reused over several scripts.

At this point of the discussion, we remark that generic steps such as 3) above (which we have already encountered in script 5W), are ideally transferred to equivalently generic steps in the target domain. So, in the business domain, stratagem 20 could be:

1) Identify roles 'strategist' and 'competitor'
2) Strategist creates confusion in competitor's company
3) Competitor is in weak state
4) Strategist attacks competitor
5) Strategist hurts competitor's business

Script 20B: Catch a fish while the water is disturbed

In the domain of business, a multitude of tactics could be used to generate confusion (e.g., misinformation). Similarly, a company attacking and hurting another can take many forms. Thus, the proposed script retains the generative nature of its equivalent in the domain of war, which makes the conceptual transfer quite obvious.

Furthermore, emphasizing the conceptualizations of scripts as generators clarifies the relationship between scripts and animations: an animation always corresponds to a particular instantiation of a script within a domain. For example, consider script 20W and its equivalent 20B (given above):

1) Identify roles 'strategist' and 'enemy'
2) Strategist creates confusion in enemy's army
3) Enemy is in weak state
4) Strategist attacks enemy
5) Strategist defeats enemy

Script 20W: Catch a fish while the water is disturbed

In the context of a battlefield, as previously stated, creating confusion and having the enemy confused can both be animated in many different ways. But step 3 is highly reusable across the 36 stratagems. Therefore, in order to emphasize this common step across several stratagems, a 'standard' animation could be associated with it. The same guideline can apply to steps 4 and 5, which are also highly reusable step across several scripts (e.g., all soldiers combat one against one for step 4, and all enemy soldiers lie on the battlefield for step 5). One advantage to having a 'standard' animation for a step reusable across a domain is that it allows the instructor to assign as an exercise the development of more specific animations for the same step in different strategies (as will be further discussed in 3.4)

[1] that can be represented as a color or an action (e.g., shaking head, going in circles) of the confused soldier.

Once animations have been associated with scripts in the source domain, the same must be done in the target domain, which presupposes: i) all source scripts have been transferred/adapted to the target domain, and ii) a single animation content has been selected for that target domain. For example, in the business domain, this shared animation could consist of:

1) two (or more) boardrooms (one per company), each with several screens monitoring share prices, sales, productivity, etc. , as well as news (in order to visualize misinformation), and

2) a pool of employees (e.g., in a cafeteria) of each company (in order to visualize employee movement across companies).

Returning to stratagem 20, step 2 could again be animated in several ways (e.g., feed contradictory information about the strategist's company to the newwire of the competitor's company, send employees to misinform). However, as in the source domain, steps 3, 4 and 5 of script 20B would be reusable and thus each should be associated with a single animation across the domain. The point to be grasped is that, ideally, the organization of animations in the target domain would be highly similar to the one in the source domain (in order to facilitate the learning of the analogy between the two domains).

Ultimately, the combined usage of scripts and animations a) favors the immersion of students in the teaching material and b) leads to a systematic student evaluation method. Both of these topics will be addressed in 3.4. First, we will briefly overview the proposed teaching environment. Please use a 9-point Times Roman font, or other Roman font with serifs, as close as possible in appearance to Times Roman in which these guidelines have been set. The goal is to have a 9-point text, as you see here. Please use sans-serif or non-proportional fonts only for special purposes, such as distinguishing source code text. If Times Roman is not available, try the font named Computer Modern Roman. On a Macintosh, use the font named Times. Right margins should be justified, not ragged.

3.3 The Teaching Environment

Given the teaching material for our experiment originates in Sun Tzu's well-known Art of War, we implemented our virtual learning environment in the form of a small village in ancient Asia.

At the entrance of this village, the user finds a configuration we will reuse systematically: 2 screens, the left one offering a textual presentation and the right one a video synchronized with this presentation and explaining it. We will refer to this configuration as a slides/video synchronized pair (hereafter SVSP). At the entrance of the village, the SVSP addresses the organization of the course (i.e., outline, logistics, organization of the village). To the right we find 2 teleport posts. The red one carries the user back to the university grounds from which this course was accessed. The blue post brings the user to the shared battlefield, which will be discussed shortly.

Once through the gate of the course, the user can circulate within the village.

The village consists of 36 traditional Chinese buildings: one for each of Sun Tzu's 36 stratagems. Each building clearly identifies the name of the stratagem it corresponds to. The motivation for such an organization is simple: each stratagem deserves a separate building because, as previously mentioned, each stratagem is associated with a distinct script in the domain. However, inside each building, the user will find the same learning configuration, namely an SVSP.

The decision to a homogeneous approach to the presentation of the teaching material proceeds from the nature of the latter: we are dealing with 36 members of a same domain, not 36 unrelated topics. Consequently, all stratagems are presented in the same way.

At this point, it must be emphasized how SL makes it easy to construct the village, to set up an SVSP at the entrance of the village and in each building, to upload the presentation and video for each SVSP, and to synchronize them. Furthermore, given it was observed that millenial students have a strong preference for learning through searching Internet, this functionality is offered via the left screen of any SVSP.

Completion of the presentation found in a building gives access to two teleports. The first one is to the *prespecified* animation of the stratagem (associated with that building) in the source domain; the second, for the target domain. In the context of the Business course at hand, recall that our source animation context is a *battlefield*.

Our battlefield involves two (or more) armies, each with its own distinct color. The battlefield can be visualized from different perspectives. In fact, the avatar of a student can be placed anywhere on this battlefield.

Each army is minimally composed of soldiers and a general. An army may also include special 'role' units, such as the spies.

Furthermore, it is straightforward to associate different states and actions with units of an army. While different graphical representations (e.g., soldier versus general) and colors can be used to denote roles and states, actions in animations must be specified as specific movements. Consider, for example, fighting. In its simplest form, this consists in having a soldier of one army 'bumping' into a soldier of another army.

More generally, as will explained shortly, animating all 36 strategies in a single domain consists in i) establishing all roles, states, and actions used by the scripts of this domain, ii) associating some graphical representation and/or animation snippet to each of these, iii) developing an animation for each strategy and iv) uploading the relevant animation for each one of the 36 buildings of our village. This process, which is discussed in the next subsection, amounts to defining what we will call a *domain specific animation language* (hereafter DSAL).

For now, the point to be understood is that the semantic richness of such a DSAL is independent of the level of sophistication of the graphics used in animations. In other words, we insist, it is entirely feasible to develop adequate animations for different domains even using simplistic graphics. This is important because simpler graphics entail less LSL programming and faster course set up. But two crucial questions must be immediately addressed:

First, do simpler graphics increase the possibility of students developing their own animations? The answer is no. The development of domain-specific animations by an instructor or a student is *independent* of the level of sophistication of such graphics. In other words, as will be explained below, the amount of work is the same, whether using simplistic or complex graphics. Conversely, the amount of work devoluted to the LSL programmer (required to implement the DSAL) is directly proportional to the complexity of such graphics. In other words, it is the course instructor who must define the semantics of a DSAL, but it is left to an LSL expert to make this DSAL operational.

Second, do simpler graphics decrease student immersion? The short answer is 'yes somewhat'. Consider, for example, the action of fighting. Peg-soldiers bumping in one another is a simple (esthetically rather unsatisfying) way of denoting this action. Having soldiers wielding swords is much more captivating. However, in the context of a university course, students feedback indicates they understand that the graphics are not the focus of the learning process; scripts are. In other words, students pay attention to the animation of the steps of a script, in the source domain, then in the target domain. In our current implementation, such an animation is uninterruptable, and thus non-interactive. That is, while the animation can optionally display the number of each step (in the top right corner of the animation), it runs without any possibility for a student to intervene. Some students complain about this lack of interaction: they would like to have an animation environment in which they could interactively control the behavior of the participants of this animation. While this is technically feasible, we have purposely ruled it out for now. The reason is simple: a script is a pre-established sequence of actions and states across a set of participants; it is immutable. Put another way, from a pedagogical viewpoint, there is no room for interaction in the animation of a script. From our viewpoint, should we, for example, let a student control the sword of a soldier or arbitrarily move the latter, we would lose both the notions of a script and of a DSAL, and, ultimately, the whole learning approach we are proposing. Put simply, a truly interactive animation would not be constrained to respect its script, which is totally unacceptable in the context of our learning approach! Consequently, in summary, we advocate the use of graphics that are capable of offering all the semantics of a DSAL. Any further sophistication to these graphics may represent a significant investment of time and money (in LSL development) while not significantly improving the learning of the relevant scripts and analogies. And, in the context of learning strategies via scripts, interactive animations of such scripts are ruled out *a priori.*

To conclude this overview of our learning method, we must now address the issue of the implementation of a domain specific animation language and how this affects our approach to student.[2]

3.4 Creating Domain Specific Animations in Second Life

As mentioned earlier, the set of scripts of a domain inherently define the roles/participants, states, and actions relevant to the animation language of this domain, a DSAL. In other words, a visualization convention must be created for each such role, state, and action. This task is carried out by the domain expert, namely the teacher.

Now recall that Second Life offers LSL, a comprehensive *scripting* language. Thus, in Second Life, the task of animating scripts ultimately consists in programming them in LSL. To illustrate this point, consider the LSL code for having a single soldier die:

[2] If necessary, you may place some address information in a footnote, or in a named section at the end of your paper.

```
//- UNIT DIE procedure
unitDie()
{
 speed = 0;
 float rotAmount = llSin(50 * DEG_TO_RAD);
 float num = llFrand(rotAmount);
 float num_2 = rotAmount - num;

 // Create a random rotation that
 // lies on the X-Y plane
 rotation dieRot = <num, num_2, 0, rotAmount>;
 llSleep(0.25);
 llSetPos(<currentPosition.x,
   currentPosition.y, deadZ>);
 llSleep(0.25);
 llSetLocalRot(dieRot);
}
```

This procedure is called in a multitude of scripts. For example, in a script in which several groups of soldiers have been defined, we find the following partial code:

```
// Large force of the red army
group 9;
position 113, 34, facing 270; colour red; move forward 104,
speed 0.5; fight 55;
// Large force of the blue army
group 8;
position 4, 37 facing 90; colour blue; move forward 103,
speed 0.5; fight 54; die;
```

This is not LSL code; this is code expressed in the DSAL for the war domain. For example, ***fight*** and ***die*** are procedures of that DSAL that have LSL definitions. It must be emphasized that each DSAL comes with its features and restrictions. For example, in the war domain, our DSAL has positioning use absolutes coordinates (while Second Life offers a third axis). On the other hand, timing delays, number of soldiers, of spies, grouping of units, etc. are specifiable by the user.

Interestingly enough, as discussed in 3.2, DSALs for war and business share some common vocabulary (e.g., creating confusion, being in a weak state, attacking someone). From a pedagogical viewpoint, this is important as it helps reducing the learning curve for a DSAL. But recall that the animations for these common states and actions will likely be totally different from one domain to another.

From the above LSL and DSAL samples, it is clear that significant programming expertise is required to use the former but not the latter. And it is unlikely a course instructor will be an LSL programmer. Consequently, as previously mentioned, it will be up to an instructor to i) define the scripts for both source and target domains, and then, for each domain, to ii) specify a corresponding DSAL (which not only identifies roles, states and actions but also describes their visualization). Then the LSL programmer will have to implement this DSAL (e.g., coding for war, that spies are in black, that 'to confuse' consists in starting to fight and then quickly slightly retreat, that being weak corresponds to having a low number of soldiers, or encircled, etc.). It is important to remark that, in practice, the DSAL will have essentially closed semantics: no new action, state or participant can be added to the DSAL and immediately used in the animations once these animations have been uploaded. In other words, fine-tuning of and possible additions to the DSAL are not to be planned frequently unless the LSL programmer is readily available. Ultimately, the onus is on the course instructor to define a DSAL that not only handles this instructor's animations in a domain, but also leaves rooms for students to contribute their own animations in that domain. This idea leads us to now consider how students can be evaluated using the proposed method. Assuming students are not fluent in LSL programming, two forms are evaluations are possible.

First, traditional evaluations consist in questions asked of the students. These can take the form of questionaires to be filled in each building. For example, once and only once a student has listened to the teaching material in a building and looked at the animation for the source domain *and* the one for the target domain, this student could be asked questions via the slides/video synchronized pair of that building. The performance of the student on such exams could even determine whether exiting from the current building is allowed! Alternatively, throughout the village, non-player characters could challenge an avatar to provide an answer to such questions, in exchange for some collectable valuable [14] (such as those required to be admitted in a guild of Sun Tzu's disciples). This second approach has the merit of presenting millenial students with a challenge, which is typically preferred to exams per se. We did not develop such traditional evaluation schemes in our current prototype and thus will not discuss them further.

Instead, the evaluation approach we have adopted requires that students develop scripts and/or animations. More specifically, we have asked students to develop a) alternative animations for stratagems in the domain of war and b) missing scripts and animations in the domain of business.

Finally, it should be emphasized that the functionality offered by Second Life readily allows all students to share the same learning environment. More precisely, all students visit the same village and several can observe simultaneously the same animation. Cooperation between students is also entirely supported in Second Life (in which communication between avatars is the central socializing concept).

4. CONCLUSIONS

In this paper, we report on how Second Life was used to create a teaching center for a university course on business strategies. More specifically, this course systematically develops analogies between the military stratagems found in Sun Tzu's "Art of War" and modern-day business strategies. Learning by analogy suggests having these teaching objects conceptualized as scripts applicable to different domains (such as war and business). The hands-on learning approach we propose rests on having *both* teacher and students capable of producing animations for each strategy. To do so, for each domain, a simple animation language targeting non-programmers is developed (and 'game-play' consists in developing animations). Having students challenged to demonstrate their understanding of the teaching material through the creation of animations constitutes a novel approach to the evaluation of students, one that favors student engagement and collaboration. Also, having students create animations ultimately verifies the completeness and correctness of the Domain Specific Animation Language these students use.

5. ACKNOWLEDGMENTS

Support from the Natural Science and Engineering Research Council of Canada is gratefully acknowledged. Wealso thank the UOIT Teaching Innovation Center, IBM's Global Business Services (GBS) and Centre For Advanced Studies (CAS) for their co-operation and financial support.

6. REFERENCES

[1] Alexander, C. *The Timeless Way of Building*, Oxford University Press, 1979.

[2] Annetta, L., Murray, M., Laird, S., Bohr, S. and Park, J.. Serious games: Incorporating video games in the classroom, *EDUCAUSE Quarterly Magazine*, Vol. 29, No.3, 2006.

[3] Dali, H., Design and Implementation of E-learning Performance Evaluation System, *International Conference on Computer Science and Software Engineering*, Volume 5:376-380, December 2008.

[4] Delwiche, A.,Massively multiplayer online games (MMOs) in the new media classroom. *Educational Technology Society*, 9 (3):160-172, 2006.

[5] ECAR 2009 report , DOI= http://net.educause.edu/ir/library/pdf/ers0906/rs/ERS0906w.pdf

[6] Gamma, E., Helm, R. Johnson R., and Vlissides, J., Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, Reading, 1994.

[7] Gentner, D., Holyoak, K. J. and Kokinov, B. K., *The Analogical Mind: Perspectives from Cognitive Science*, MIT Press, 2001.

[8] Kolb, D., Experiential learning: experience as the source of learning and development. Englewood Cliffs, New Jersey: Prentice-Hall, 1984.

[9] Lee, S.F., Roberts, P., Lau, W.S. and Bhattacharyya, S.K., Sun Tzu's as business and management strategies for world class business excellence evaluation under QFD methodology, *Business Process Management Journal*, Volume: 4(2) 96 - 113, 1998.

[10] Linden Scripting Language Portal, DOI= http://wiki.secondlife.com/wiki/LSL_Portal

[11] Linden Scripting Language Test Harness, DOI= http://wiki.secondlife.com/wiki/LSL_Test_Harness

[12] Mangold, K., Educating a new generation: Teaching baby boomer faculty about millenial students, *Nurse Educator*, 32(1):2123, 2007.

[13] McNeilly, M., Sun Tzu and the Art of Business: Six Strategic Principles for Managers, Oxford University Press, 2000.

[14] Michael D. and Chen, S., Serious Games: Games That Educate, *Train and Inform*. Thomson Course Technology, Boston, MA. USA, 2006.

[15] Monaco, M. and Martin, M., The Millennial Student: A New Generation of Learners. *Athletic Training Education Journal*, 2:42-46, 2007.

[16] Ruben, D., Simulations, games, and experience-based learning: The quest for a new paradigm for teaching and learning. *Health Education Research*, Theory and Practice, 30(4): 498505, 1999.

[17] Schank R.C. and Abelson, R., *Scripts, Plans, Goals, and Understanding*. Earlbaum Assoc., Hillsdale, NJ, 1977.

[18] Second Life, an Online 3D Virtual World. Linden Lab. DOI= http://secondlife.com

[19] Second Life in Education, Exploring the Educational Uses of Second Life. DOI= http://sleducation.wikispaces.com/educationaluses

[20] Science Notes 09 DOI = http://scicom.ucsc.edu/SciNotes/0901/pages/avatar/avatar.html

[21] Thirty six stratagems DOI = http://en.wikipedia.org/wiki/Thirty-Six_Stratagems

[22] Thirty Six Stratagems as business strategies DOI= http://www.businesscoachingexecutive.com/36-stratagems.htm

[23] Wing, R.L., The Art of Strategy: A New Translation of Sun Tzu's Classic The Art of War, Main Street Books, 1988.