Int J Artif Intell Educ (2015) 25:35–59
DOI 10.1007/s40593-014-0022-z

RESEARCH ARTICLE

Narrative Balance Management in an Intelligent Biosafety Training Application for Improving User Performance

Nahum Alvarez · Antonio Sanchez-Ruiz · Marc Cavazza · Mika Shigematsu · Helmut Prendinger

Published online: 19 August 2014

Abstract The use of three-dimensional virtual environments in training applications supports the simulation of complex scenarios and realistic object behaviour. While these environments have the potential to provide an advanced training experience to students, it is difficult to design and manage a training session in real time due to the number of parameters to pay attention to: timing of events, difficulty, user's actions and their consequences or eventualities are some examples. For that purpose, we have extended our virtual Bio-safety Laboratory application used for training biohazard procedures with a Narrative Manager. The Narrative Manager controls the simulation deciding which events will take place in the simulation, and when, by controlling the narrative balance of the session. Our hypothesis is that the Narrative Manager allows us to increase the number of tasks for the user to solve and, due to balancing difficulty and intensity, it keeps the user interested in training. When evaluating our system we observed that the Narrative Manager effectively introduces more tasks for the user to solve, and despite that, is accepted by the users as more interesting and not harder than

N. Alvarez (✉) · H. Prendinger
National Institute of Informatics, Tokyo, Japan
e-mail: nahum.alvarez.ayerza@gmail.com

H. Prendinger
e-mail: helmut@nii.ac.jp

A. Sanchez-Ruiz
Complutense University, Madrid, Spain
e-mail: antsanch@fdi.ucm.es

M. Cavazza
Teesside University, Middlesbrough, UK
e-mail: m.o.cavazza@tees.ac.uk

M. Shigematsu
National Institute of Infectious Diseases, Tokyo, Japan
e-mail: mikas@nih.go.jp

an identical system without a Narrative Manager. Also, a knowledge test demonstrated better results in users' interest and learning output in the narrative condition.

Keywords Intelligent tutoring systems · Interactive narrative · Drama managed environment · User interest

Introduction

Virtual intelligent training environments show great potential for teaching techniques and protocols in an affordable, effective and scalable way (Chen Y.F., et al. 2008), especially in cases where training in real life proves to be costly and difficult to manage on a large scale. They have been adapted to increasingly complex application scenarios, and are now able to simulate almost any kind of environment with realistic graphics and accurate object behaviour (Carpenter et al. 2006; van Wyk and de Villiers 2009; Belloc et al. 2012). Experiments with these applications show promising results regarding users' feeling of immersion and presence (Schifter et al. 2012; Wang et al. 2011) and engagement (Charles et al. 2011). In another work, (Reger et al. 2011) a virtual reality simulation is used to treat posttraumatic stress disorder in soldiers obtaining a reduction in their symptoms, showing that the virtual environment are effective in cases when real life rehearsals are not feasible. However, creating new training scenarios and managing the events that happen in such interactive environments is a complex task, because it is necessary to keep the difficulty balanced and pay attention to the user's engagement: if the training is too hard for the user, s/he won't be able to progress well. On the contrary, if the training is too easy or repetitive for her/him, the user may lose interest in the training, affecting the learning outcome.

Some researchers proposed to add a 'narrative layer' to training systems (Marsella et al. 2000). They hypothesized that a narrative centred learning environment will both increase the engagement of the user and the learning outcome.

We want to focus on this kind of integration of educational content and narrative, but with a more practical point of view: using narrative techniques in a control module for the training sessions will allow us to manage events in the session and show them in a more attractive form for the user. Systems with such direct control in the dramatic flow of the session are rare. Narrative flow has been formally defined as a sequence of events that generates an emotional response in the user, and the control of the flow is typically achieved by modelling the grade of conflict the user experiences in each moment (Crawford 2004; Ware and Young 2011).

Our contribution in this paper is to introduce a new drama manager that controls the dramatic flow in a virtual training scenario aimed at practicing biohazard procedures. We developed a virtual Bio-safety Lab (under submission in Prendinger et al. 2014), which serves as a virtual training environment for medical students. This virtual environment targets the training of protocols in response to an accident in the laboratory. These protocols are too dangerous to train in the real laboratory, and paper tests do not cover all the unexpected problems that can arise during the clearing of the accident. For this reason, a virtual application (in this case a virtual laboratory) can be a very good solution to this real need in medical teaching. Bio-safety Lab acts as a virtual tutoring system via user task recognition and has been tested already in two field experiments.

The remainder of the paper is organized as follows. The following section reviews previous work on virtual training applications and narrative control, and compares our system to that work. Thereafter, we will introduce our system, which uses a drama manager to control the events in a biohazard training environment. Here we first describe 3D interaction with the Bio-safety Lab and real-time task recognition, and then introduce the narrative techniques used in our training system. Next, we will present an experiment we conducted to investigate the engagement and training effects of our system, and discuss the results. Finally, we conclude our work and indicate future steps.

Related Work

Virtual training systems vary greatly in the method used. Testing systems in which users give answers and advance to the next step if they gave the correct response has been written about widely. For instance, in (van Wyk, and de Villiers 2009) an application designed to train accident prevention in a mine makes the user to check videos of a scenario and detect dangerous spots, only advancing when the user gives the correct answer. Other examples of these "testing systems" are a computer vision system checking user gestures following hand-washing procedures (Corato, Frucci, and di Baja 2012) or a tool designed for teaching how the modules of an energy plant work(Angelov and Styczynski 2007). These systems allow a high degree of immersion thanks to 3D graphics and videos, but lack flexibility regarding the user's participation in the training scenario.

More advanced training scenarios are the work of (Belloc et al. 2012) that implement intelligent objects with realistic behaviours, or the work of (Johnson and Rickel 1997), which incorporates an interactive tutor capable of correcting the user or explaining the rationale of certain steps. This additional functionality requires a more complex architecture. In the case of adding complex behaviour to scenario objects, a widely accepted strategy is to use a layered architecture that supports low, physics-related event translation into high-level events. For this translation process authors use knowledge-based techniques like rules or ontologies, such as in systems of (Lugrin, J.-L. and Cavazza 2007) or (Kallmann and Thalmann 2002) that include common sense behaviour for objects, which is difficult to implement with only a physical engine. Similarly to those works, our Bio-safety Lab includes both intelligent object behaviour and a real-time interactive tutoring system that corrects users' mistakes, in addition to a narrative layer controlling the simulation.

Virtual Training Systems Based on Narrative Control

Prevous works suggest that a narrative centred learning environment increases the engagement of the user (Gee 2003; Shaffer 2006), but testing if engagement is linked to a positive increment in learning outcome has shown more controversial results: There are findings for users participating in a narrative game-like application for learning French that showed no different scores compared with users using one without game mechanics (Hallinen et al. 2009), or proved that including off-task possibilities can even hinder user performance (Rowe, J.P. et al. 2009). Another experiment showed

that adding game mechanics and a narrative does not directly affect performance, yet can improve a user's attitude towards the exercise (Rai et al. 2009). However the opposite conclusion can be drawn from (Carini, R. M. et al. 2006) showing in a large-scale experiment that certainly engagement is positively linked to learning. The cause of this difference in the results of those works is shown in a subsequent experiment where Rowe et al. (2011) could show that engagement certainly can increase knowledge acquisition when the educational content and narrative are motivationally and tightly integrated. In that case, the system used mechanics similar to adventure games for teaching microbiology, in which it is necessary to investigate diverse places and objects and make notes related with the learning theme. Another example is the STAR framework (Molnar, A. et al. 2012), which using the same strategy showed good results in testing an application designed to teach health practices to students. In that application, the user takes on the role of a detective who has to solve the mystery of a poisoning crime, obtaining clues about bad habits that can result in getting sick.

In our Bio-safety Lab, the inclusion of narrative elements is not independent to the other mechanics of interactive training: we use narrative techniques to generate the events in the scenario, giving to the user new tasks to solve or helping him. These events directly affect the type and number of training contents the user will experience, so we can affirm that the narrative is tightly integrated with learning. Also, by balancing the difficulty and keeping the user interested we could make him experience psychological flow, as stated by Shaffer (2013). This concept describes a state in which the subject has a sense of engaging challenges at a difficulty level appropriate to his capabilities (Nakamura and Csikszentmihalyi 2002). The consequence of this effect is to persist or return to the activity because it is psychologically rewarding, which is in line with our hypothesis.

Aside from the learning benefits that narrative techniques have, the use of narrative authoring is an interesting strategy in creating training sessions and has been noted and used by a number of researchers. Examples are the works of Rizzo et al. (2010) where they present a clinical diagnosis training system including virtual agents with a scripted backstory and speech recognition for simulating patients or Carpenter et al. (2006) who proposed an approach that uses branching trees to advance in a story in which the user has to take decisions to manage a crisis. However, the user still has limited individual interaction possibilities to select the branch in the story tree, but there is no direct interaction with the environment. Another work (Gordon 2003) describes a protocol to create a branching structure where training scenarios are constructed like storyboards, and applies it to a web-based application. The approach of Ponder et al. (2003) uses a human as a scenario director or narrator. In this application the narrator controls part of a training session that aims to teach a user how to handle the information distribution in a crisis situation, and communicates with the user via voice. Another module uses a tree-based story model to drive the session forward. Similarly, Raybourn et al. (2005) describe a multiplayer simulation system for training army team leaders. The simulation is directed by a human instructor who controls events in the virtual environment. All of these systems present a well-known problem of the branching approach: it does not scale well when the number of events or possibilities in the story increases. Trying to give a solution for authoring educational interactive stories, (Silverman, B. G. et al. 2003) presented a generic tool for creating interactive drama for educational purposes but his conclusion was that there are no environments one can turn to for rapid

authoring of pedagogically-oriented interactive drama games. This issue has been confronted by incorporating a drama manager to control the events in the virtual world. For example, the commercial software Giat Virtual Training includes a scenario engine driven by a definition language that represents the training tasks as a series of steps, each step corresponding to an agent inside the system that has a birth, an activity and a death, and communicates with the rest of the agents (Mollet and Arnaldi 2006). The system allows connecting with a pedagogical module controlling the training and triggering events in a narrative way but neither its methods nor its impact on the training was explored in their works. Another drama manager can be found in the application presented by (Habonneau et al. 2012), an interactive drama application for teaching how to take care of people with brain damage. The application uses IDtension, a drama manager engine that controls the simulation and does not rely on branching so it does not have scalability issues. IDtension (Szilas 2007) contains a repository of goal tasks for the scenario characters and controls these characters performing actions that generate or resolve conflicts in the simulation. This solution proved to be very immersive and engaging for the users, allowing them to find themselves in a realistic environment where the unexpected can happen. However, IDtension was not evaluated in terms of learning, and it does not provide tutoring to the user: the system does not balance the difficulty of the session (it only cares for the dramatic impact in the user, not if he/she can resolve the situation or not) nor does it warn the user about incorrect actions. Simply put, it is a narrative-only experience without training elements in which the user interacts with characters, sees what happens and hopefully learns from his/her actions.

Introduction to Bio-Safety Lab

In Bio-safety Lab, the user takes the role of a bio-safety laboratory worker in first-person. The user can control his avatar's movements with the keyboard and with the mouse controls the right hand, allowing him to interact with the scenario's objects by doing a right or left click: possible actions are: taking or dropping objects, opening or closing doors, or using concrete objects. Figure 1 shows a typical system session

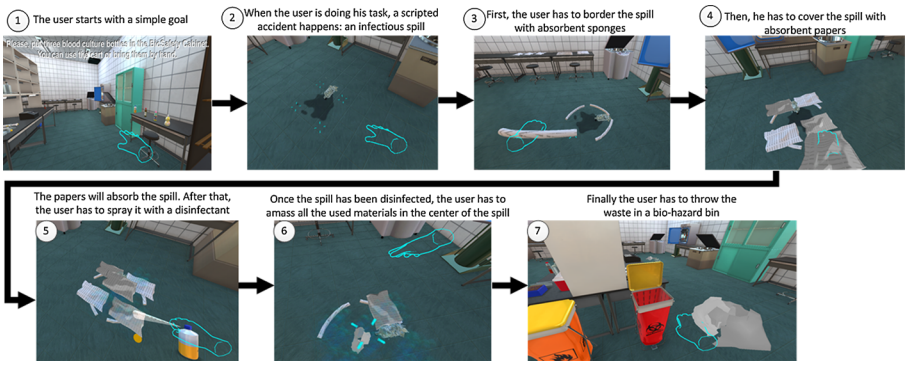

Fig. 1 Sample session and Spill cleaning in Bio-Safety Lab. The user controls the avatar's hand and performs the *different steps* (steps 3–7) in the protocol for cleaning the toxic spill

sample. After starting, the user is given a simple goal to perform: a routine laboratory procedure (see number 1 in Fig. 1). When the user begins to perform this initial goal, a scripted accident happens (number 2), involving the spill of a contaminated human blood sample, the user has to follow a certain protocol to clear it. This protocol contains the guidelines for cleaning a potentially infectious spill used in real-life and was given by our collaborators at the National Institute of Infectious Diseases in Japan. The protocol includes containing the spill by bordering it with special absorbent sponges, then absorbing the spill by putting absorbent papers on it, then disinfecting the spill with an antiseptic, and finally disposing the waste into a Bio-hazard bin (numbers 3–7).

The application implements a Task Recognition Engine, a module which acts as a virtual tutor, recognizing whether the user is doing properly the task of clearing the accident properly or not, and advises on it. This module stores the information concerning each task, including not only the correct way to act, but also common mistakes for each task, and is stored in a knowledge base. In this paper, we present the Narrative Manager, a control module inside the Task Recognition Engine that manages the scenario's events using a narrative model which optimally selects the timing and the event to trigger.

The training system in the Bio-safety Lab is inspired by Marc Cavazza's Death Kitchen (Lugrin, J.-L. and Cavazza 2006), which is an application that merges narrative management with complex scenario behaviour, where the system generates events dynamically. In Death Kitchen the user controls an avatar in a virtual kitchen while the system generates accidents and tries to "kill" him. It merges the narrative together with a layered behaviour intelligent system: the high level events that happen in the scenario are captured by the Causal Engine, an intelligent object's behaviour module that decides the consequences of the event and acts as a reasoner and drama manager. For example, if the user interacts with a water tap, opening it, the Causal Engine can make it work as expected, letting the water run normally from the tap, or decide that is more convenient to break the tap and spray the water all around.

This Causal Engine is similar to the reasoner engine contained in the Bio-safety Lab which captures high-level events and processes them. As in Death Kitchen, in Bio-safety Lab these events are used to decide to trigger consequences in a narrative fashion, but it also uses the events to perform the user's task recognition in giving real-time assistance (Death Kitchen does not have this feature). However, the way in which the consequences are decided differs. Death Kitchen uses a knowledge database called "Danger Matrix" (essentially a look-up table) containing the characteristics and requirements of each accident, and a planner selects the most dangerous available accident in the table and tries to generate it, creating new events if necessary. However, this Danger Matrix only allows one constraint (the partial order) for selecting the available events to trigger. In Bio-safety Lab, we select the event that maximizes the impact on the user using a knowledge structure known as Task Trees, allowing different constraints when deciding to select the event, which are also used when recognizing the user's task.

Another difference appears when deciding events to trigger an accident; Death Kitchen only uses two parameters: distance to the object and danger level of the accident. This means that the system can lead to an identical accident if the user goes to the same area in each session. The system works in this way because by design, Death Kitchen does not require more complex decision taking, but in our case we need

more complex behaviour in order to optimize the triggering of events. In order to achieve this optimization, we modulate the level of conflict between the user and the environment, creating difficult events for the user if the current task is too easy for him, and lowering the intensity of the events, providing assistance when the task is too difficult. For example, if the user is on the step containing the spill, the system could decide if the spill would overflow its boundaries, making the user hurry up and re-contain it. If the user is cleaning the spill too easily, the system can knock over another bottle, causing it to fall to the floor and break, creating a new spill (this of course can happen only if the bottle is available). On the other hand, if the user is having too much trouble solving his task, the system can help by hinting where the user should go and what he should do. This way we can create more events for the user to solve while at the same time keeping him engaged.

Research in the narrative field stated that creating a sense of conflict in users increases their engagement (Gerrig, R.J. 1993). Also, conflict has been well defined and modelled, as seen in (Ware and Young 2011), who created a narrative planner that generates stories with conflict, and defined a set of dimensions of the conflict in order to quantify it. However, the planner does not create conflict in the stories automatically, but only allows it as one of the possible outcomes: it is left to the player to encounter it. Also, they did not use the dimension of the conflict they defined in their work, admitting that was difficult to find a perfect solution of how to include them, and that they were somewhat subjective. Interestingly, they carried out an experiment (Ware et al. 2012) to validate these dimensions by contrasting how human subjects quantify conflict in stories with the algorithm they created (which includes giving values to some parameters manually) to measure them, and in the results there was agreement between the two quantifications, determining that the dimensions can be recognized as qualities of conflict.

Narrative Event Generation in Bio-Safety Lab

In Bio-safety Lab, the system places the user in a virtual laboratory and tells him to perform certain tasks. At some point when the user is handling lab materials, a bottle containing human blood sample contaminated with bacteria will fall, creating an infectious spill. The user must clean this potentially dangerous accident following a standard protocol.

The training system assists the user in the process of resolving the accident through real-time (user) task recognition. Our Narrative Manager decides how and when certain events will happen in the scenario, to either (a) decrease task difficulty if the user has trouble completing it, or (b) to increase task difficulty if the task is too easy for the user. The ultimate goal of the Narrative Manager is to engage the user in an interesting narrative entangled within the training process, maximizing the number of events for the user to solve, and thus increasing the possibility of learning acquisition. The architecture of the system is detailed in Fig. 2.

The system has two separate parts: the interactive 3D Environment Module and the Task Recognition Engine. The first controls the virtual world where the user interacts and the second being independent of the simulation. This 3D Environment contains a Common-Sense Reasoner that translates Low-level events native to the 3D engine into

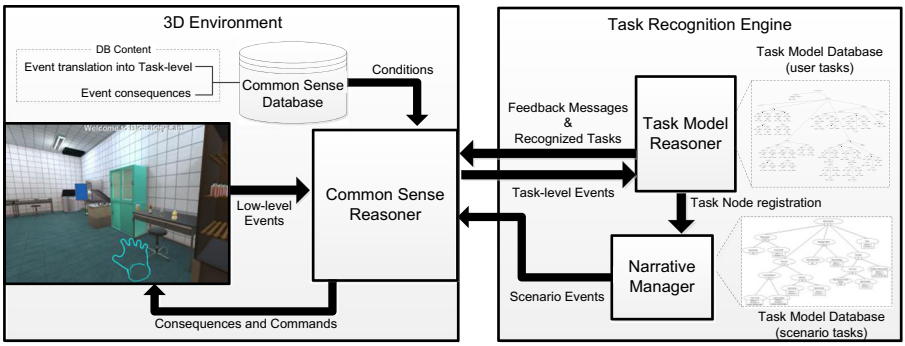

Fig. 2 System architecture in Bio-safety Lab. The Low-level events are translated into Task-level events and sent to the Task Model Reasoner where the state of task will be updated. Then the Task Model Reasoner will correct the user if he/she has made a mistake and the Narrative manager will decide if to trigger a Scenario Event

high-level ones called Task-level events and sends them to the Task Recognition Engine. This translation is done by consulting a common sense database that contains a set of rules relating the Low-level events with the Task-level ones, and also describing the consequences of each Task-level event if there are any. If the Reasoner decides that there is a consequence derived from the Task-level events it has an event dispatcher sub-module for triggering the desired effect. For example, the user can drop a bottle to the floor, generating a physical event (the collision between the object and the floor). Then the Common-Sense Reasoner checks into the database and finds that the colliding object was fragile so it should break. This break event will be sent to the Task Recognition Engine, and also will trigger a consequence in the environment: the breaking of the bottle into pieces.

The Task Recognition Engine contains two connected modules: the Task Model Reasoner and the Narrative Manager. The Task Model Reasoner is in charge of providing assistance to the user by supervising his actions and sending him messages whenever he makes a mistake during the current task, and is detailed fully in depth in Prendinger et al. 2014 (under submission). The Narrative Manager also monitors the user and triggers events in the 3D virtual world in order of keep the user engaged in his tasks. In the next subsections, we will describe them in depth. Finally, the Narrative Manager has been developed as an independent module and works by sending and receiving signals from the simulation engine, so it could be assembled in other training systems provided that a Task Model Database for that domain has been defined.

Task Model Reasoner

The Task Model Reasoner receives messages from the Common Sense Reasoner whenever a Task-level event occurs there. These messages are used in order to decide what kind of task the user is performing and if he is doing it correctly. We use a Knowledge-based representation in the form of Task Trees for identifying the different steps of the available tasks. This structure is able to deal with different levels of knowledge abstraction, and supports real-time monitoring and assistance, containing information about the possible mistakes that can be made and how to correct them.

A Task Tree is a hierarchical model which has a root node representing the whole task to perform on the top of the tree and child nodes being the subtasks in which it can be divided, down to the leaf nodes, which correspond to Task-Level Events in the virtual environment. A diagram of the task tree used from the spill accident is shown in Fig. 3. Hence the goal of the user will be to validate the root node of the tree by performing actions represented in the leaf nodes. In order to validate a parent node, the child nodes have to be validated following certain conditions: sibling nodes are related with each other in three different ways: AND (all subtasks must be completed in any order), OR (different ways to complete a task) and SEQ or sequences (all subtasks must be completed in order; these sequences do not require having all the included subtasks performed one immediately after the other, allowing separate and sequential actions with other different subtasks in between).

The system is able to monitor user actions by receiving task level (high-level) events from the virtual environment and matching them with the nodes of the Task Tree. In order to represent correct steps and mistakes, each of the leaf nodes is marked either as a correct step in the task or as an error. The OR groups contain the possible documented errors. Error nodes are still part of the same group as their sibling nodes, usually an OR group, due to it being one option more than performing the subtask. The rationale behind this is that it is a physically plausible way to perform a step of the task, even if it is a wrong one, and allows it to be recognized as an error while performing that step. Whenever a correct step is matched with a leaf node, it is instantiated in the task tree and the tree is updated from the bottom up, instantiating parent nodes if necessary and so on.

If the received event is considered a user mistake, the engine will not instantiate it but send back an assistance message in real-time to the virtual environment and display it to the user. There are two types of mistakes: actions that are explicitly represented in the task model as incorrect, and actions that are part of the correct procedures but that should not be executed yet. In the first case the system will explain to the user why he

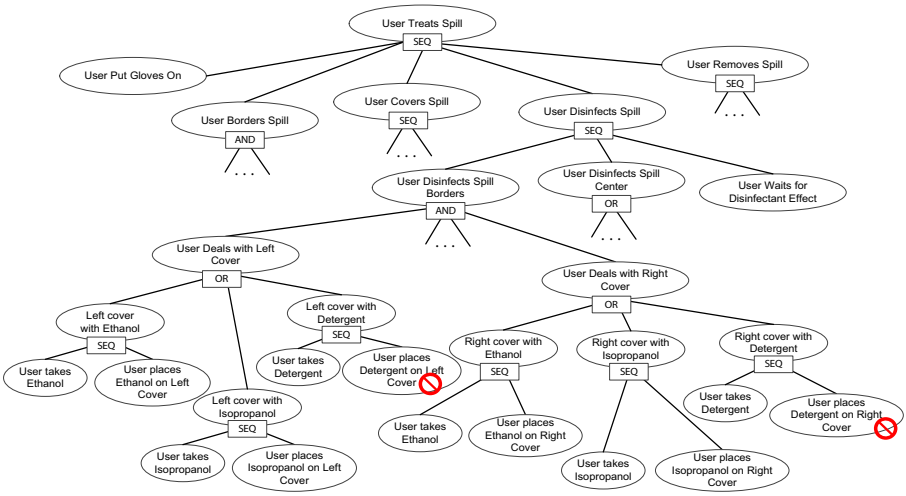

Fig. 3 Part of the Task Tree modelling the spill clearing protocol (complete Tree contains more than 250 nodes). The goal of the user is represented in the root node. Two nodes are marked as incorrect and if they are instantiated, the module will warn the user

should not perform that action, and in the second case the system will display which is the previous action the user has to perform before the one previously attempted. As an additional feature, the system allows information collection about user behavior and reaction to accidents and mistakes. This is a useful source of posterior analysis to identify more difficult parts of training or user profiling.

Narrative Manager

The Narrative Manager module is in charge of sending orders for triggering events back to the 3D Environment. These orders can be in response to the user's actions or proactive decisions from the module. The Narrative Manager manages the balance of difficulty in training sessions and the level of user interest in the training. Our hypothesis is that we can create more events for the user to solve by maintaining high levels of user interest. Like the Task Model Reasoner, the Narrative Manager also uses task trees as a knowledge database to decide what kind of events it will trigger. These task trees represent tasks for the system with the goal of hindering or helping user progress in a concrete task (described in another Task Tree) and contain two types of nodes: user nodes and system nodes. User nodes represent high level events performed by the user, and are instantiated by the Task Model Reasoner in the same way as described in the previous section. System nodes represent the events the Narrative Manager will trigger. The system nodes also contain an extra parameter called "Closure" indicating if that event can only be triggered before another concrete event takes place. We included this parameter because some events can only happen before the user completes a certain subtask. For example, a toxic spill can only overflow if has not been bordered with sponges before. The Task Tree containing the events related to the spill accident is shown in Fig. 4.

The Narrative Manager uses a set of parameters that describe the dramatic value of the events and the situation. These parameters are used to decide which event will be triggered next and are based on those of (Ware and Young 2011) described as the

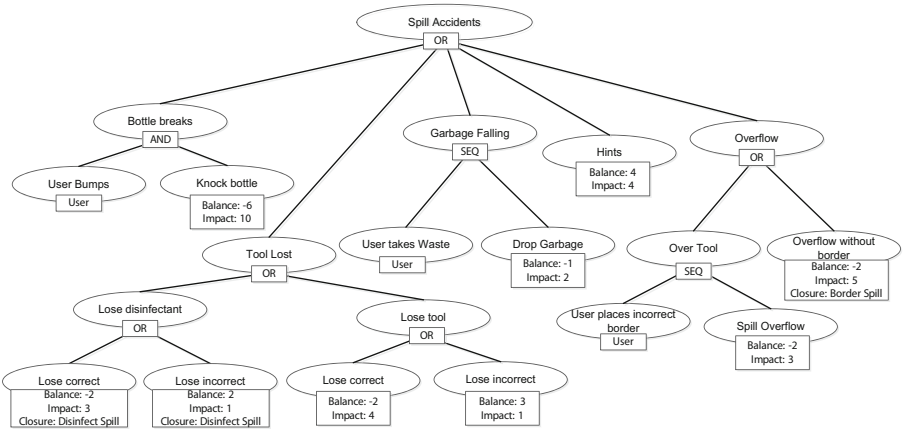

Fig. 4 System Task Tree for the spill accident. In each node's box its relevant information can be seen: the parent nodes show the group type (AND, OR or SEQ) and the leaf nodes inform if its type is User or not. Non-zero parameter values are shown in the leaf nodes: Balance, Impact and Closure

dimensions of conflict. These dimensions are four: balance, which measures the relative likelihood of each side in the conflict to succeed; directness, which is a measure of the closeness of participants including spatial, emotional and interpersonal distance; intensity, which is the difference between of how likely it will be for a participant to potentially succeed and how unlikely it will be for them to potentially fail; and lastly, resolution, which measures the difference in a participant's situation after a conflict event occurs.

In their work they also recognize that there is no perfect way to give value to parameters and how to use them, and give directions on how to formally quantify them provided there is a component of decision from a domain expert. Thus, we decided to adapt them since they are designed to create conflict between characters and do not fit in a training domain which only has one participant and in which failure is not a desired outcome. For example, we do not need to know the sentimental closeness of the user to the scenario nor how will affect an event the user's long-term future. We use four parameters with our own nomenclature: Balance, Global Balance, Impact, and Intensity:

- Balance: an integer number that describes how good or bad an event is for the user (positive if it makes the task easier for the user and negative if it is difficult or hinders his or her performance). Each event contains its own balance value stored in the Task Tree, and the value is given by the domain expert that designs the tree. For example, breaking a bottle that contains an infectious sample would have a Balance of −6, and giving the user a hint has a Balance of +4. The Narrative Manager also maintains a global value for the accumulated balance of the whole training session by summing the balance of the triggered events; we call this parameter Global Balance. This Global Balance has a slow attenuating effect with time, tending towards 0. The rationale behind this attenuating effect is that receiving some hints or difficulties can affect the user in the short term but as time goes by this effect disappears: the hints are useful only for the immediate task and the problems can be resolved for the user if he/she takes some time to think.
- Impact: a positive value that represents the degree of dramatic load that an event has for the user when triggered. This value is also stored in each node of the Task Tree representing an event, and its value is given by the domain expert. An event with a Balance value very near to 0 can have high Impact if it has a strong visual or psychological effect. For example, when a toxic spill overflows its borders, it has a Balance of −2, being not very bad for the user, but very startling for him, having an Impact of 5.
- Intensity: a global positive value maintained by the system that describes how dramatic the current situation should be for the user. Intuitively, when the Intensity is high, the system will likely trigger some bad event soon. At the beginning its value is zero and it is updated dynamically by the Narrative Manager by adding or subtracting the Impact value of the triggered events to its previous value. The value is added or subtracted depending on its Balance value sign. For example, after the breaking of a bottle the system will decrease its Intensity value: the breaking of the bottle event has an Impact value of 10, and negative Balance so the Intensity will decrease by 10. This means that the system will not create large Impact events for some time. Conversely, after a sending a hint to the user, the Intensity of the system will increase 4 points, allowing for the triggering of larger events in the future.

As we can see, the values of Balance and Impact are static and quite subjective, since they are related to the dramatic quality of an event or the session. The domain expert has to define them by judging each event when he designs the scenario, and then the Narrative Manager will use these values to calculate the value of the related dynamic parameters (Global Balance and Intensity, that not only depend on the static value of the two static parameters, but also the timing and the type of user events). Attending to these parameters the Narrative Manager selects the next event to trigger and dynamically models the flow of dramatic intensity for each session, generating conflicts for the users and escalating the tension when necessary, and then relaxing the pace and the difficulty after that. The Narrative Manager is consulted whenever a high-level event is received by the Task Model Reasoner, or after a timer is reset (in our system the timer was set to 15 s). The rationale behind periodically calling the Narrative Manager even if the user is not performing actions is that in a narrative, it is meaningful even when nothing is happening, insofar as the system sends an empty event in order to build the intensity and decide if to trigger an event.

Whenever the Narrative Manager is called it selects an event from the Task Trees which maximizes Impact, with the current Intensity as the upper boundary, and with the Balance most similar to the current Global Balance, but with opposite value. We use Intensity as an upper boundary because it represents the maximum dramatic load we allow for the user at one time, thus limiting the events that can happen. Also, the system has to move the Global Balance towards zero, so the desired event should have a Balance similar to the current Global Balance, but with opposite value. Using both parameters allows maintaining a balanced difficulty while at the same time creating events that impact the user in a dramatic way. It is possible that an event does not meet the conditions above because the current Intensity is too low. In that case, no event will be triggered.

Figure 5 shows an example of how the Narrative manager selects events in a sample session of Bio-Safety Lab. In the example, the user starts bordering the spill (Number 1 in Fig. 5) and the actions increase Intensity as they are registered in the Task Model. Then the narrative Manager searches for a suitable event with an Intensity lower or equal

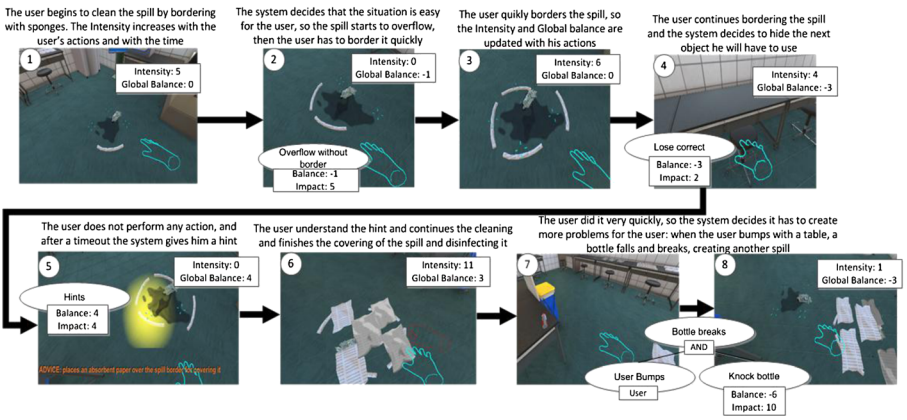

Fig. 5 Scenario Example showing the Narrative Manager triggering different events as the user performs his goal task. The Intensity and the Global Balance are updated as the user acts, the time passes, or scenario events are triggered. The Narrative Manager selects the action with the greatest Impact using the Intensity as an upper boundary and tries to choose the one with a Balance near the Global balance but with opposite value

than 5, the selected one being the one to cause the spill to overflow (number 2). Also, the Manager updates the Intensity and Global Balance values: it subtracts the Impact of the triggered event because it had a negative Balance, and adds the Balance to the Global Balance. Next, the user continues to border the spill and time passes, so the system updates the Intensity and Global Balance again. Again, in number 4 we can see that the system selected other events that hinder the user's task, this time the Impact and Balance values were not very close to the desired ones (6 and 0, respectively), probably due to not having other better events available at that moment. Number 5 shows that after some time of the user not performing any step of the main task, this time the system selected an event to help the user with a visual hint, increasing the Intensity and the Global Balance to positive values. These values increase even more (number 6) with the progressive completion of the task, reaching high values if the user is fast enough. This makes the Narrative Manager trigger a big problem to the user (see number 7 and 8): when the user bumps a table (it is very easy to bump lab tables, but the bottles only fall when the Narrative Manager decides so), a bottle falls and creates a new spill.

The event selection is carried out by an engine that traverses the scenario task tree and selects the nodes that maximize these values, and then selects one event from these candidates randomly. The selection of nodes follows the rules we stated for sibling nodes in the previous section, so, for example, the Narrative Manager will not execute a node in an OR branch if a sibling node has been executed already, and it respects the SEQ branches by executing nodes in order. Also, it takes into account the closures in the nodes, only selecting nodes if the closure has not been triggered. Finally, once the desired event is selected, the Narrative Manager updates the Utility and global balance again and sends back the order to execute the event to the virtual environment.

Finally, each time an event happens (whether it is an event performed by the user or triggered by the Narrative Manager) the Narrative Manager keeps the global value of the Balance and the Intensity updated. Intensity is calculated by adding or subtracting the Impact value of the event that just happened, depending on its Balance value sign (positive for adding, and negative for subtracting). The intuitive meaning behind this is that after an event with high intensity and negative balance (difficult for the user) is selected, the drama manager does not disturb the user more for some time. On the contrary, after a "good" event, it is more likely that something bad will happen. The current Global Balance is calculated by adding the Balance value of the event to it. If the Narrative Manager receives the empty event, it means that nothing relevant has happened, but a significant amount of time passed, so it will increase the Intensity value and the Global Balance will slowly converge to zero. We increase the Intensity because we consider that giving time helps the user to decide how to complete the task (hence it is beneficial to him/her), so after some time an event should trigger. Also, the Balance changes over time because the effects of an event diminish over time. In other words, the balance returns to zero as the situation becomes no more beneficial neither prejudicial to the user. For example, in the case of a negative event, if a bottle breaks the user could use some time to recall the steps of the protocol, or to look for materials to clean it. On the other hand, in case of positive events, if the user received a hint (a positive event that increases the balance) about what material to use for cleaning the spill, the effect of the hint only lasts until the user finds and uses the object. If after that the user does not know what to do, and spends some time thinking, it decreases the positive balance.

System Evaluation

We tested in an experiment if our Narrative Manager could generate more events for the user to solve while at the same time maintaining them interested. If this is proven true, then as the user resolved more situations without being uninterested, the learning outcome should increase. The experiment was conducted at the National Institute of Informatics in Tokyo, confirmed by its ethical board. A total of 30 subjects participated, 17 male and 13 female, all informatics students. The experiment consisted of 3 stages: first the user does a stage to learn about the domain and controls of the application, and then in the second and third stages the users run two training sessions in the Bio-safety Lab application with and without Narrative Manager. The experiment lasted approximately 75 min. Figure 6 shows a diagram of the design of the experiment. The three stages are detailed next.

First Stage The participants read a manual with the protocols for clearing certain accidents in a bio-safety laboratory. The manual contains a brief description of how to clear the accident in the application and also a guide with pictures of laboratory material and instruments for the training session. The subjects can consult this manual at any moment they want in this stage. After reading the manual, they run a tutorial in order to learn how to use the application. By doing the tutorial the subjects learn how to control their avatar inside the application and also familiarize themselves with the environment and learn the use of certain objects that they will have to use later. The goal of this stage is eliminate possible usability problems and provide domain related knowledge necessary to perform the tasks in the virtual environment.

Second Stage Then, the participants are separated randomly in two groups: Script Condition and Narrative Condition, each one running two sessions of the scenario. The participants do not know in which group they are, and they cannot see the others' screen, so they are blind to the condition. Then, the participants in the Script condition run the training scenario twice in a version of the Bio-safety Lab without the Narrative Manager and the participants in the Narrative condition run the training scenario in the Bio-safety Lab with the Narrative Manager twice, which will dynamically trigger events in real time. Both versions contain one pre-scripted accident (spilling human blood infected with bacteria, see Fig. 1 to see a graphical depiction of how a sample session without Narrative manager is conducted) soon after starting the application;

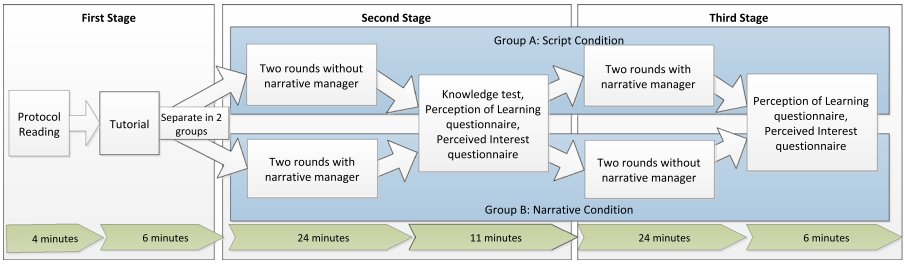

Fig. 6 Experiment design showing the three sequential stages of the experiment with the duration of each one. The users were distributed evenly in two groups, without knowing to which one they were assigned

until then the two scenarios contain no differences, and we consider this concrete instant as the experiment start point. It finishes when the user runs out of time (12 min) or when he cleans all the existing spills. Both versions also use the Task Model Reasoner to recognize the user's tasks and give a warning when the user makes mistakes. In summary, the differences between the two systems are that in the version without Narrative Manager the only events from the system are the initial scripted spill accident and the Task Reasoner Module warnings reactive to user's mistakes, whereas in the version with Narrative Manager dynamically generated system events and hints are also included. After the subjects finished, they filled in a Knowledge Test about the spill cleaning protocol, containing questions like "*Is there an incorrect position to handle a spill? Where and why?*", "*What is the correct order to cover the spill?*" or "*Which material should be used for bordering the spill?*" among others. Then the subjects filled in a Perception of Learning questionnaire used to capture user impressions of learning from the simulation. This information is important and one of the sources for evaluating training tools (Schumann et al. 2001). After the Perception of Learning questions, they filled in an adapted Perceived Interest Questionnaire (Schraw 1997) to evaluate how interesting the system they ran was. We do not use all the questions of the Perceived Interest Questionnaire because not all were applicable to the system, as it was a generic test for narrative works.

Third Stage Once the participants completed the questionnaires, they twice ran the scenario again, but this time the one in the other condition. The reason to have another run with the other version of Bio-safety Lab is to give the participants the opportunity to compare which one is more interesting. Finally, the participants are given the Interest questionnaire again and three additional comparison questions to evaluate the system they have just run. However, we did not make the user perform the Knowledge Test because in our experiment the scenario was too simple, so after all runs of it, the users would remember everything easily.

The questions from the questionnaires were in the form of a Likert scale from 0 to 10, being 0 "Completely in disagree" and 10 "Completely agree", and are contained in Table 1.

Results and Discussion

We extracted the behaviour information of the subjects from the system logs that Bio-safety Lab creates, in order to represent on a chart how they proceeded to task completion. We chose the burn-down charts due to it being widely used as a graphical representation of work left to do versus time (Mike Cohn 2005). In the vertical axis we used the distance to the goal as heuristic range, which is the number of steps the user has to take to clean a spill, starting at 23 and finishing at 0. Each one of the dots on the graph symbolizes one action from the user or one event from the system, and when the value reaches zero on the Heuristic range axis it means that the subject has cleared the scenario.

Figure 7 shows the charts created for all the subjects in the Narrative Condition, distributed in two groups (the Scripted Condition subjects follow a homogeneous

Table 1 Questionnaire used in the experiment, containing questions about recall and perceived interest

Questions
I remember the session in the virtual environment completely
I knew what to do next at all times
I noticed warning messages from the system
When reading a warning message from the system, I understood why
When reading a warning or an advice message from the system, I knew what to do next
I noticed differences in the application between sessions
I thought the scenario tried to help me while using the application
I thought the scenario tried to obstruct me while using the application
I remember how to treat a spill of infected blood
I understand the problems that can arise solving a spill of infected blood
I think this application would be very useful in training laboratory procedures
The events in the system were different each time I played
The session was suspenseful
The scenario grabbed my attention
I would like to play again if I had the chance
I thought the session was very interesting
I got caught up in the session without trying to
I would like to know more about the scenario
I liked this session a lot
I thought the first system was more amusing than the other
I thought the first system was more difficult than the other
I thought the first system was more challenging than the other

tendency and will be discussed at the end of this section in order to compare it with the Narrative Condition). We can observe two tendencies in the Narrative Condition Group, so we divided the chart in two in order to make it easier to read. In one of the groups (upper graph), the users were advancing steadily to the task completion, so the Narrative Manager decided to create one extra spill for them to clean. This event is represented by a big peak in the graph, widening the distance to the goal by 23 points (we limited this event to a maximum of one per session, due to it taking a long time to be dealt with). On the other hand, the second group (bottom graph) was considered by the system to have enough trouble with the initial spill and the extra events it created for them, so the subjects did not have a second spill. In these cases, the graphs have hills representing small problems for the users, like spill overflows or missing tools.

We can further analyze the behaviour graphs and identify the impact that each Scenario Event has in task completion. Figure 8 shows the task progression graph of one subject of the experiment, and how they behave when certain events happened. As we can see, when an event which is difficult for the user happens, the curve rises; for example, when the spill overflows, the curve rises a bit because is a light inconvenience that requires some actions to be redone. Also, when a bottle breaks and a new spill is created, we can see a big peak in the function, because the user will have to complete another 23 actions to clear the scenario. Some of the "bad" events do not create a rise,

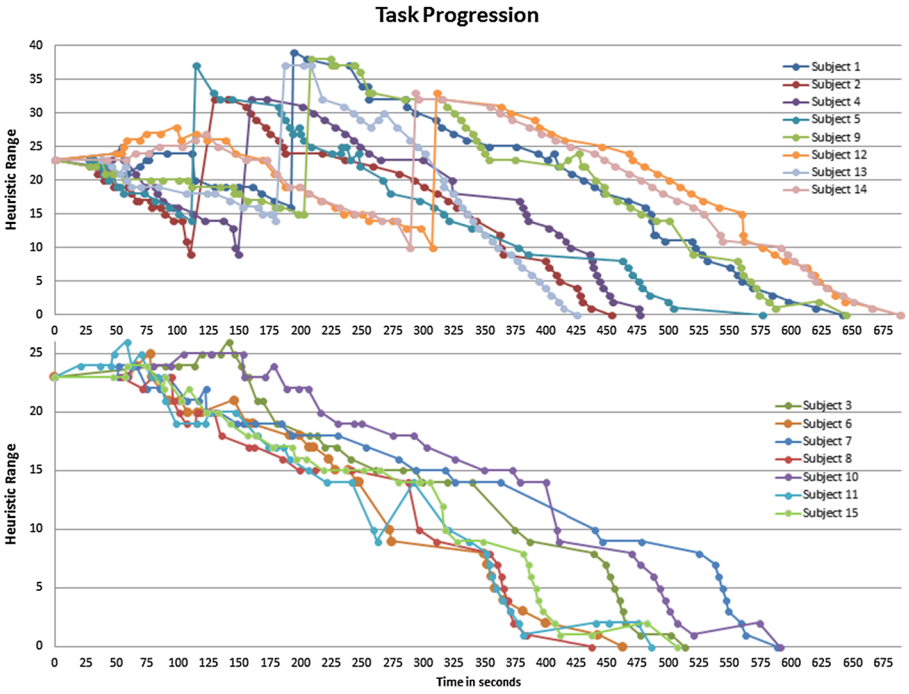

Fig. 7 Task Progression charts for Narrative Condition Subjects. The *upper chart* shows the users that advanced quickly to the goal and as a result the Narrative Manager decided to create an additional spill (represented by the big peak of each graph). The *bottom chart* shows the users that advanced slowly in their task and as a result the Narrative Manager only presented them small difficulties

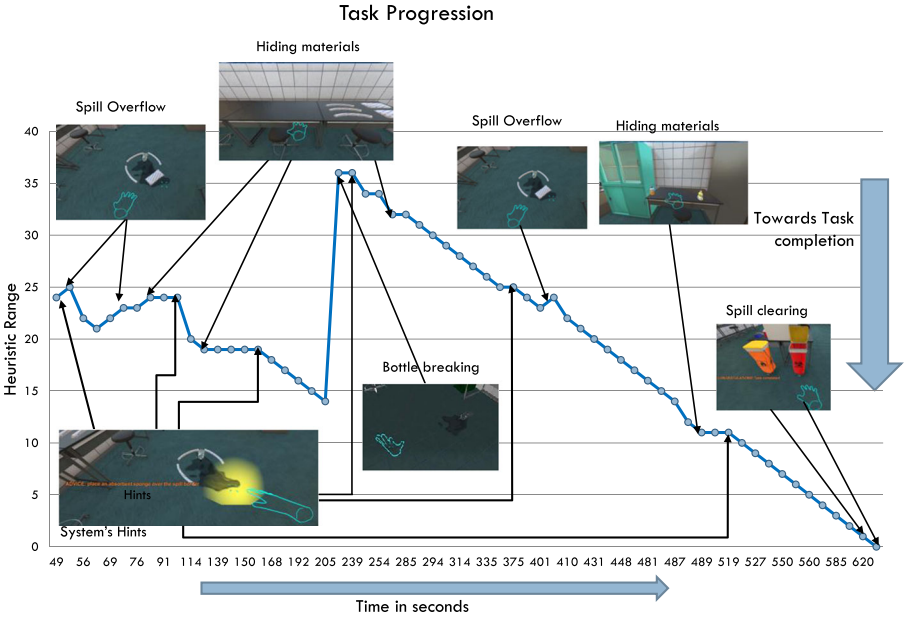

Fig. 8 Task Progression Graph for subject one in the Narrative Condition. We can observe how the task advances or goes back depending on the event ordered by the Narrative Manager

but a flat zone. This is the case of hiding materials from the user, which causes the user to lose time doing other actions and searching for materials, preventing him from solving the accident. On the other hand, the system sometimes helped the user by giving hints. We can see this in different points of the graph when after receiving a hint from the system there is an advance towards task completion. Finally, the two spills were cleared at the end, the two last steps before clearing the scenario. In total we can perceive the session as a narrative work, seeing how the user had some small problems at the beginning, started to solve the situation, and then something very bad happened. Finally, after some time the user starts to clear the situation again and except for some minor problems that were able to be solved, he finished the scenario.

By analyzing the events that happened in the session, we can also calculate the Balance parameters for each instance in the session. We can create a Balance curve that represents how well the situation developed for the subject as he proceeded towards task completion. If we relate the task progression graph to the balance curve for that session (see Fig. 9) we observe that these events are translated directly in an increase and decrease in the session balance. For example, system hints create small peaks and "bad" events are valleys in the balance curve. Moreover, when the subject is advancing through the task, the graph rises; this means that the current task is easy for him/her. The new spill (created by breaking a bottle) is a big cliff. We can see that it occurred just after the subject was advancing steadily to task completion. The Narrative Manager decided that the task became too easy for the user at this point and created stress causing a new spill. We can see that as the user is completing the task, the balance curve goes up again, the system continues oscillate the balance, and towards the end, when the user is quickly solving the situation, the system tries to decrease the balance. Nevertheless, the user managed to clear the accidents.

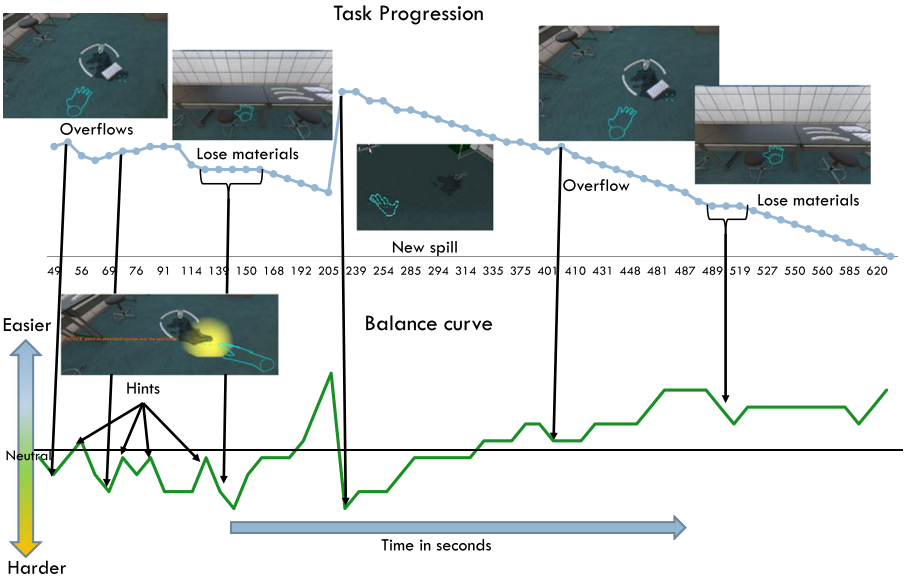

Fig. 9 Balance Graph related to Task Progression for subject one in the Narrative Condition. The events sent by the Narrative Manager can be directly related to the increase or decrease in balance levels

Finally, when comparing the Task Progression graphs from the Narrative condition and the Scripted condition we see that in the scripted condition the curve proceeds directly towards the end and contains many less events (counting User and Scenario events, including warnings and hints) and takes less time to complete. On average, the Narrative Manager driven sessions contained 154.29 events, and the other without Narrative Manager 76.36. Of course, if the scripted version contained an equivalent number of system events it would be interesting to compare. In that case, if the events are scripted, the version without Narrative Manager would contain actual narrative, so we decided to only maintain the necessary minimum of scripted events. However, it would be interesting to test the Narrative system against one with randomly generated events. In the next section we will elaborate more on this idea.

In Fig. 10 we can compare two subjects from the two groups and observe the difference in event density. The subject from the Narrative condition in the figure is the same as analyzed in this section (Figs. 8 and 9), containing 204 events, and the one from the session without Narrative Manager contains 76 events. Intuitively, we could think that having more events in the scenario for the user to solve would mean that the scenario is more difficult for the user to solve, but when reading the results of the questionnaires (see Table 2) the subject's perception is different. Please note that the questionnaires do not mention the name of the compared systems, addressing them as "first" and "second". We have combined the answers of the two groups for easier reading (reversing the answers of the group that started without Narrative Manager). Seeing the results, not only does the user not think the Narrative condition is easier than the scripted one, but he/she also perceives that the Narrative condition is more engaging and interesting.

For the next two questionnaires results (see Tables 3 and 4), we conducted a T test for two populations assuming equal variances with an alpha value of 0'05 in order to compare how different they are. We also conducted a Holm-Bonferroni correction in order to counteract the multiple comparisons problem for each test, showing that some of the results could be false positives, having T test values greater than the corrected α

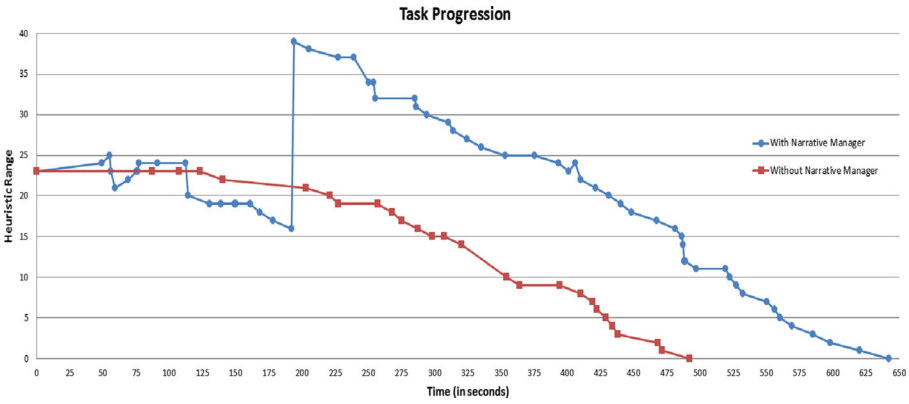

Fig. 10 Comparison between two sessions from the two conditions: The upper graph (*blue, circle dots*) represents the session of a subject in the group With Narrative Manager, and contains more events and takes more time than the bottom graph (*red, square dots*), from a subject in the group Without Narrative Manager

Table 2 Average results of the comparison questions

Question	Average (out of 10)
I think the NC system was more amusing than the other	7.16
I think the NC system was more difficult than the other	5.08
I think the NC system was more challenging than the other	6.75

value; concretely, the three last questions in Table 3, and questions 4, 10 and 11 in Table 4 are false positives. In order to obtain more decisive results we are planning to enact future experiments (for more details, see the next section of the paper). The remaining statistically significant results of the interest questions (see Table 3, questions 2 and 4) indicate that the Narrative Manager system was more interesting than the other one, maintaining the user engaged in the training session, and supporting our hypothesis. The consequence of being more interested is that the system with more tasks to solve is not harder for the subjects, allowing them to learn more. This consequence appears in the results of the Knowledge Test containing 8 questions about the protocol. The subjects who used the Narrative Manager system had an average score of 9.48 out of 10 in the Test, while the subjects without the Narrative Manager had an average score of 7.03; the T test value (alpha value of 0'05) when comparing the two groups results is 7.48e-8. From these results we can show that the narrative control of the training session improves the user's interest, and as a consequence he can train more tasks, resulting in a better learning outcome. We are aware that these good results can be due to the scenario being a single accident with a straight forward protocol, but we think that the simplicity of our scenario helps the Scripted Condition users more than the Narrative Condition ones (the protocol is easier to remember). So, with more complex scenarios, the difference between the two conditions would be more pronounced. Also, it can be argued that the knowledge

Table 3 Average results of the perceived interest questions

Question	With narrative manager	Without narrative manager	T test value	Holm-Bonferroni correction α
1.- The session was suspenseful	5.66	5.0	0.177	0.007
2.- The scenario grabbed my attention	7.66	6.0	0.005	0.008
3.- I'd like to play again if I had the chance	6.66	5.83	0.178	0.01
4.- I thought the session was very interesting	8.33	6.16	0.005	0.012
5.- I got caught in the session without trying to	6.83	4.83	0.015	0.016
6.- I 'd like to know more about the scenario	7.33	5.66	0.015	0.025
7.- I liked this session a lot	6.83	5.5	0.036	0.05

Table 4 Average results in the perception of learning questions

Question	With narrative manager	Without narrative manager	T test value	Holm-Bonferroni correction α
1.- I completely remember the session in the virtual environment	9.16	7.33	1e–3	0.004
2.- I knew what to do next at all times	8.5	6	8e–4	0.004
3.- I noticed warning messages from the system	8.16	7	0.03	0.005
4.- When reading a warning message from the system, I understood why	8.16	6.5	0.01	0.005
5.- When reading a warning or an advice message from the system, I knew what to do next	8.33	5.0	1e–6	0.006
6.- I noticed differences in the application between sessions	7.66	4.33	7e–4	0.007
7.- I thought the scenario tried to help me while using the application	8.16	6.16	4e–3	0.008
8.- I thought the scenario tried to obstruct me while using the application	5	3.33	0.02	0.01
9.- I remembered how to treat a spill of infected blood	9.33	7.83	0.01	0.012
10.- I understood the problems that can arise when cleaning a spill of infected blood	8.66	7	0.02	0.016
11.- I thought this application would be very useful in training laboratory procedures	9.33	8	0.01	0.025
12.- The events in the system were different each time I played	7.5	2.83	3e-5	0.05

test we used for measuring the learning outcome might not be enough to absolutely confirm how much our system improves learning. However, as the focus of this work is to analyze how our drama manager improves the user's interest in the training session, we think that the knowledge questionnaire was enough to confirm if the Narrative Manager system has better learning values compared to the scripted system.

Moreover, in the subjective Perception of Learning questions shown in Table 4, we can observe better values too. We only interpret these questions as indicative of the training experience for the subjects, as we already have the knowledge test as an objective metric for learning improvement. The results are better in the Narrative Manager system in all questions. The subjects subjectively perceived that they remembered training session better with the Narrative Manager. Also, they perceive more help from the system and remember the messages they received better, as the Narrative Condition effectively helps the user and has more warning messages, including proactive hints for the user (the version without Narrative Manager only has reactive warning messages). However and interestingly, the obstruction from the system is not very strongly perceived. The users perceived differences between the two rounds they ran the scenario, and even if both systems use the same warning messages to correct user's mistakes in the same way, they are better perceived.in the Narrative Manager version.

Conclusions

In this paper we have presented a new method for controlling events in a training session, using narrative techniques adapted to a training domain. We applied this method in an intelligent training system for biohazard laboratory procedures based on users' real-time task recognition. In this application we integrated a Narrative Manager module which communicates with the task recognition system and uses its same knowledge structure for modeling scenario tasks. The Narrative Manager decides when and which event will be triggered, depending on the user's performance and the difficulty of the situation, which in turn balances the training session. Our hypothesis is that we can use this technique to keep the user interested in the training. As a consequence, the Narrative Manager is able to maximize the number of events and tasks for the user to solve while keeping him engaged, leading to more knowledge acquisition. Even if our subjects were computer science students, we believe the results here will generalize well to other groups, especially those interested in Bio-safety procedures.

Our main contribution is the design and integration of the Narrative Manager within a virtual training program, and the method it uses to control the session. This method is inspired on a narrative conflict model, adapted for training scenarios, and uses a knowledge structure called Task Trees that enables us to apply this work to different training domains. By controlling the level of conflict in the training session, we keep it dynamically balanced for the user and modulate narrative intensity to make him interested. The Narrative Manager is an independent module, so with the correct domain definition in the knowledge base, it can be potentially applied to other types of tutors or training scenarios, i.e. evacuation training, law enforcement, manual assembly, or machinery work.

We tested our system in a pilot study where we showed that the Narrative Manager system keeps the users effectively interested and depicted how it models the training sessions. The users are able to train more time when the session is interesting for them and they remember better the training procedures. Even though the Narrative Manager presents more tasks for the user to solve, it is not perceived as more difficult, and the users have a better experience from the session. As a consequence of these facts, we can confirm that the Narrative Manager has a positive impact on the learning outcome of the training session. We also analyzed a sample of the subjects' sessions and showed how the Narrative Manager models the session and balance over time, resulting in a narrative curve.

As a future work, we are planning to improve the Narrative manager to make it able to model a training session using a predefined narrative curve as an input template, improving the authoring capacity of the application. With this new capability, the training designer not only models the system knowledge, but can propose how the training should proceed by giving high-level directions. The narrative Manager will still decide events and timing, but it will adjust itself to the input template, creating different sessions each time, but with similar outcomes. We also want to apply our narrative method to different domains in order to demonstrate its versatility, and we would like to run a larger scale experiment with a more complex training procedure. Such experiment will allow us to better analyze the learning outcome degree of our system and to obtain more definite results, in order to confirm or refute the results we obtained. For that

experiment we want to use more control conditions too: one with only the necessary minimum of scripted events, as we did before, another one with randomly generated events and a third one without proactive hints. It would be very interesting to compare the results of those systems: theoretically, if the events are random, there is a possibility that a course of generated events could create a narrative session. Also we would be able to identify the concrete impact of the hints in users' performance. In that case, the results could be similar to the Narrative system, so we should analyze with what probability or under what conditions we can obtain these results.

Acknowledgments This work was supported partly by (1) the Health and Labour Science Research Grants, Research on Emerging and Reemerging Infectious Diseases (Research on development of teaching materials and methodology to evaluate performance to strengthen biorisk management [H20-Shinko-Ippan-009]) from the Ministry of Health, Labour and Welfare of Japan as a contract research, (2) the 'Global Lab' Grand Challenge grant from the National Institute of Informatics, (3) Fundaçao para a Ciencia e a Tecnologia, under project PEst-OE/EEI/LA0021/2013, and (4) the Spanish Ministry of Economy and Competitiveness under grant TIN2009-13692-C03-03. The authors would like to thank A. Nakasone and H. Damas for their help with the system development, and E. Gray for his help with the graphical assets.

References

Angelov, A. N., & Styczynski, Z. A. (2007). Computer-aided 3D virtual training in power system education. In IEEE General Meeting of the Power Engineering Society, IEEE, 1–4.

Belloc, O. D. R., Ferraz, R. B. D., Cabral, M. C., de Deus Lopes, R., & Zuffo, M. K. (2012). Virtual reality procedure training simulators in X3D. In Web3D, C. Mouton, J. Posada, Y. Jung, and M. Cabral, Eds. ACM, 153–160.

Carini, R. M., Kuh, G. D., & Kleint, S. P. (2006). Student engagement and student learning: testing the linkages. *Research in Higher Education, 47*(1), 1–32.

Carpenter, E., Kim, I., Arns, L. L., Dutta-Berman, M. J., & Madhavan, K. P. C. (2006). Developing a 3D simulated bio-terror crises communication training module. In VRST, M. Slater, Y. Kitamura, A. Tal, A. Amditis, and Y. Chrysanthou, Eds., ACM, 342–345.

Charles, T., Bustard, D. W. & Black, M. M. (2011). Experiences of promoting student engagement through game-enhanced learning. In M. Ma, A. Oikonomou & L. C. Jain (ed.), Serious Games and Edutainment Applications (pp. 425–445). Springer . ISBN: 978-1-447-12160-2.

Chen,Y.F., Rebolledo-Mendez,G., Liarokapis,F., de Freitas,S., Parker,E. (2008). The use of virtual world platforms for supporting an emergency response training exercise. International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games.

Corato, F., Frucci, M., & di Baja, G. S. (2012). Virtual training of surgery staff for hand washing procedure. In AVI, G. Tortora, S. Levialdi, and M. Tucci, Eds., ACM, 274–277.

Crawford, C. (2004). *Chris Crawford on Interactive Storytelling*. Indianapolis: New Riders Publishing.

Gee, J. P. (2003). *What video games have to teach us about learning and literacy*. New York: Palgrave Macmillan.

Gerrig, R.J. (1993). Experiencing narrative worlds: On the psychological activities of reading. Yale U. Pr.

Gordon, A.S. (2003). Authoring branching storylines for training applications.

Habonneau, N., Richle, U., Szilas, N., & Dumas, J. E. (2012). 3D Simulated Interactive Drama for Teenagers Coping with a Traumatic Brain Injury in a Parent. In LCNS (Ed.), *Interactive Storytelling*. Berlin Heidelberg: Springer.

Hallinen, N., Walker, E., Wylie, R., Ogan, A., & Jones, C. (2009). I was playing when I learned: A narrative game for French aspectual distinctions. In S. Craig & D. Dicheva (Eds.), *Proceedings of the Workshop on Intelligent Educational Games at the 14th International Conference on Artificial Intelligence in Education* (pp. 117–120). Berlin: Springer.

Johnson, W. L., & Rickel, J. (1997). Steve: an animated pedagogical agent for procedural training in virtual environments. *SIGART Bulletin, 8*(1–4), 16–21.

Kallmann, M., & Thalmann, D. (2002). Modeling Behaviors of Interactive Objects for Real-Time Virtual Environments. *Journal of Visual Languages and Computing, 13*(2), 177–195.

Lugrin, J.-L. & Cavazza, M. (2006). AI-based world behaviour for emergent narratives. In H. Ishii, N. Lee, S. Natkin & K. Tsushima (ed.), Advances in Computer Entertainment Technology (pp. 25). ACM . ISBN: 1-59593-380-8.

Lugrin, J.-L. & Cavazza, M. (2007). Making sense of virtual environments: action representation, grounding and common sense. In IUI '07: Proceedings of the 12th international conference on Intelligent user interfaces (pp. 225–234). ACM Press. ISBN: 1-59593-481-2.

Marsella, S., Johnson, W.L., & LaBore, C. (2000). Interactive pedagogical drama. In Proceedings of the Fourth International Conference on Autonomous Agents, 301–308.

Mike, C. (2005). *Agile Estimating and Planning. Prentice Hall PTR*. NJ, USA: Upper Saddle River.

Mollet, N., Arnaldi, B. (2006). Storytelling in virtual reality for training. edutainmen. 334–347.

Molnar, A., Farrell, D. & Kostkova, P. (2012). Who poisoned Hugh? - The STAR framework: integrating learning objectives with storytelling. In D. Oyarzun, F. Peinado, R. M. Young, A. Elizalde & G. Méndez (ed.), ICIDS, Vol. 7648 (pp. 60–71). Springer. ISBN: 978-3-642-34850-1.

Nakamura, J., & Csikszentmihalyi, M. (2002). The concept of flow. Handbook of positive psychology, 89–105.

Ponder, M., Herbelin, B., Molet, T., Schertenlieb, S., Ulicny, B., Papagiannakis, G., Magnenat-Thalmann, N., & Thalmann, D. (2003). Immersive VR decision training: Telling interactive stories featuring advanced virtual human simulation technologies. In A. Kunz & J. Deisinger (Eds.), *7th International Workshop on Immersive Projection Technology, 9th Eurographics Workshop on Virtual Enviroments* (pp. 097–106). Zurich: Eurographics Association.

Prendinger, H, Alvarez, N., Sanchez-Ruiz, A., Cavazza, M. Catarino, J., Oliveira, J., Prada, R., Fujimoto, S., Shigematsu, M. (2014). Under submission in ACM Transactions on Intelligent Systems and Technology (TIST).

Rai, D., Heffernan, N. T., Gobert, J. D., & Beck, J. E. (2009). Mily's World: Math game involving authentic activities in visual cover story. In S. Craig & D. Dicheva (Eds.), *Proceedings of the Workshop on Intelligent Educational Games at the 14th International Conference on Artificial Intelligence in Education* (pp. 125–128). Berlin: Springer.

Raybourn, E. M., Deagle, E., Mendina, K., & Heneghan, J. (2005). *Adaptive thinking & leadership simulation game training for special forces officers* (Proceedings of Interservice/Industry Training, Simulation, and Education Conference). FL: Orlando.

Reger, G. M., Holloway, K. M., Candy, C., Rothbaum, B. O., Difede, J., Rizzo, A. A., & Gahm, G. A. (2011). Effectiveness of virtual reality exposure therapy for active duty soldiers in a military mental health clinic. *Journal of Trauma Stress, 24*, 93–96. doi:10.1002/jts.20574.

Rizzo, A., Parsons, T., Buckwalter, G., & Kenny, P. (2010). *A New Generation of Intelligent Virtual Patients for Clinical Training. In IEEE Virtual Reality Conference*. Waltham: USA.

Rowe, J. P., McQuiggan, S. W., Robison, J. L., & Lester, J. C. (2009). Off-task behavior in narrative-centered learning environments. In S. Craig & D. Dicheva (Eds.), *Proceedings of the 14th International Conference on Artificial Intelligence in Education* (pp. 99–106). Berlin: Springer.

Rowe, J. P., Shores, L. R., Mott, B. W., & Lester, J. C. (2011). Integrating Learning, Problem Solving, and Engagement in Narrative-Centered Learning Environments. *I. Journal of Artificial Intelligence in Education, 21*(1–2), 115–133.

Schifter, C. C., Ketelhut, D. J., & Nelson, B. C. (2012). Presence and Middle School Students' Parti-cipation in a Virtual Game Environment to Assess Science Inquiry. *Educational Technology and Society, 15*(1), 53–63.

Schraw, G. (1997). Situational Interest in Literary Text. *Contemporary Educational Psychology, 22*(4), 436–456.

Schumann, P. L., Anderson, P. H., Scott, T. W., Lawton, L. (2001). A Framework for Evaluating Simulations as Educational Tools. Developments in Business Simulation and Experimental Learning, Vol. 28, 215–220. Reprinted in the Bernie Keys Library, 8th edition.

Shaffer, D. W. (2006). *How computer games help children learn*. New York: Palgrave Macmillan.

Shaffer, O. (2013). Crafting Fun User Experiences: A Method to Facilitate Flow. Human Factors International. Online White paper. http://www.humanfactors.com/FunExperiences.asp.

Silverman, B. G., Johns, M., Weaver, R., & Mosley, J. (2003). Authoring Edutainment Stories for Online Players (AESOP): Introducing Gameplay into Interactive Dramas. In O. Balet, G. Subsol, & P. Torguet (Eds.), *International Conference on Virtual Storytelling, Vol. 2897 (pp. 65–73)*. Berlin, Heidelberg: Springer. ISBN 3-540-20535-7.

Szilas, N. (2007). A computational model of an intelligent narrator for interactive narratives. *Applied Artificial Intelligence, 21*(8), 753–801.

Van Wyk, E., & de Villiers, R. (2009). Virtual reality training applications for the mining industry. In Afrigraph, A. Hardy, P. Marais, S. N. Spencer, J. E. Gain, and W. Straßer, Eds., ACM, 53–63.

Wang, M., Lawless-Reljic, S., Davies, M., & Callaghan, V. (2011). Social Presence in Immersive 3D Virtual Learning Environments. In P. Novais, D. Preuveneers, & J. M. Corchado (Eds.), *ISAmI, Vol. 92 (pp. 59–67)*. Heidelberg: Springer. ISBN 978-3-642-19936-3.

Ware, S.G., & Young, R.M., (2011). Toward a Computational Model of Narrative Conflict. Technical Report DGRC-2011-01, Digital Games Research Center, North Carolina State University, Raleigh, NC, USA. http://dgrc.ncsu.edu/pubs/dgrc-2011-01.pdf.

Ware, S.G., Young, R.M., Harrison, B., & Roberts, D.L. (2012). Four quantitative metrics describing narrative conflict. In Interactive Storytelling (pp. 18–29). Springer Berlin Heidelberg.