Narrative Balance Management in an Intelligent Training Application with User Task Recognition for Enhancing User Interest and Performance

Nahum Alvarez Ayerza
Department of Informatics,
School of Multidisciplinary Sciences
The Graduate University for Advance Studies (SOKENDAI)

Doctoral Dissertation
Doctor of Philosophy

Dedicated to my Yaya, and to my Yayo

Acknowledgements

A doctoral thesis can be seen as the culmination of a long research process which spans along a few years. In my case I consider that this thesis is the final product of a longer period, a period which started way before, before even I came to Japan. Here I present the people I have to thank for backing me in this adventure.

I can remember clearly the moment I decided to study informatics. I was 13 and quickly my parents tried to put out that idea from my head, but it was to no avail. My initial motivation was no special: like many youngsters I was attracted to videogames and computers and inspired by them and as a result I ended in the computer science faculty, with my parents finally putting up with it. That's why firstly I want to thank them for all of these years supporting me. They were in part guilty of my avocation for research and books: my mother taught me to love books and knowledge and my father to fend for myself. I can remember countless times when I got a useful lesson and support from them. Some of my dearest memories are my mother reading me The Hobbit or my Father teaching my how to drive. I can still feel their support across the distance and even now, after my father died, I see him from time to times in dreams so I will feel always backed by both. So here are my deepest thanks.

They are not the only ones to influence me at different aspects in my way, of course: I remember still my walks with my late grandfather, Yayo, and I contemplate with great nostalgia my long holydays in the sea with him and my grandmother, Yaya, someone who has endured the toughest sorrows with the hardest will without surrender. So here are my deepest thanks for them. My other grandparents are in my thoughts too and maybe my character is closer to that branch of the family. I never had so much contact with my grandfather Alejandro, but one of my first memories is from him, and from the stories my family told me about him maybe I owe many of my traits to him. As for my grandmother Toñi, she is the greatest source of the family sense of humor, amusing coincidences, stories and song and maybe part of my creativity comes from there. She will keep singing and joking many, many years more. I love them all because thanks to them I am who I am, and I know that maybe I will be in the future as able as them, so here are my greatest thanks.

I was not the only child in my house. As the big brother I should be responsible and be an example to them but that's a difficult task and I only have done what I could, hoping it was enough. My brother Nestor is someone very different to me. That's why thanks to him I had the opportunity and the trouble at the same time of having a constant difference in opinions. Two aspects of him fascinate me greatly: a social skill I could not possibly match and something he has that I will never have: Faith. On the other hand, my sister Carla is probably my best female friend; we understand each other deeply, being very similar but at the same time very different. Maybe I spoil her a bit but I cannot avoid my fondness by her. She has assisted me whenever I needed, I have done the same for her, and without a doubt we will continue doing it. I sometimes question me if they aren't the ones giving me example. I have a third brother, also. It is smaller and furrier but as with the rest I love him, and he has given me also unconditional love: my dog Poker. Here are my humblest thanks to all of them, my siblings.

Finally, I also want to include some family members more, and thank one of my best friends in my family when I was a child: my cousin Ramón. I still remember our long talks about our dreams and hopes. Also thank my cousin Daniel, who is always a great source of

talks and wisdom. And finally one of my first role models: my uncle Abel, a continuous inspiration for seeking knowledge. Here are my most inspired thanks to both.

Returning to the moment when I decided to study informatics, in high school, I met with one of the best friends I have, Andrés. Since then we have passed for a lot, and got tons of memories. When we were teenagers we made a plan to go to Japan someday. In the end we did, but not exactly as we thought as we went in the same period but with other people, but it didn't matter, as we were together in spirit. Once I entered in the university I met another three friends I need to mention: Toni, Miguel y Edu. More than only friends, they have been for me models, brothers and teachers. We overcame our studies by supporting each other and we still do. We grew together and thanks to these four friends I grew into the man I am now. So here are my most kindred thanks.

After I graduated I started a normal job career as developer. At that moment I could not imagine I would be here now, but I already was aspiring to learn more about developing games so I registered in a game developing master. There I met more good friends; inspiring, kind, and capable, they are Jorge, Pablo, Borja, Juan and Sergio. Also I met Fede who would become my next master supervisor and a mentor figure for me. In that period I met also Nestor, and since then we have been sidekicks, having a complete and reciprocal sense of trust, support and rivalry, pushing each other to give the best of ourselves. Here are my most admiring thanks for them.

In my research I have been advised and assisted by many people: first of all I'd like to thank my supervisor Prof. Prendinger for his help, and also to Prof. Cavazza and Prof. Arturo Nakasone for their advices, without which it would have been impossible to me to progress. Next I want to thank to all of the internship students who have collaborated in my projects: Alex, Kwan, Joao C, Joao O, Fonseca and Hugo. Also, to all the people at the NII who have supported me in some way: Pascual, Andres, Isaac, Joan, Ferran, Vanessa, Mio and Fujino. Here are my professional thanks to them.

Almost as long as my stay in japan is the time I know my girlfriend Misumi. Being she from another culture, I still don't know how she endures my barbarian eccentricities, some of them difficult to endure even for a westerner, I'm afraid. However, she is always there for me to support me and help me, even when I don't deserve it. For that, here are my most loving thanks.

Summary

The use of three-dimensional virtual environments in training applications allows the simulation of complex scenarios and realistic object behavior. These environments offer an ideal setting to develop intelligent training applications; yet, their ability to support complex procedures depends on the appropriate integration of knowledge-based techniques and natural interaction. However, while these environments have the potential of providing 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: timing of the events, difficulty, user's actions and its consequences or eventualities are some examples.

In this work, we describe the implementation of a Narrative Manager system controlling automatically an intelligent training system for biohazard laboratory procedures, based on the real-time instantiation of task models from the trainee's actions. This project was made in collaboration with the National Institute of Infectious Diseases in Tokyo. A virtual biohazard laboratory has been recreated using the Unity3D engine, in which users interact with laboratory objects using keyboard/mouse input or hand gestures through a Kinect device. Realistic behavior for objects is supported by the implementation of a relevant subset of common sense and physics knowledge. In order to solve the issue of how maintain the user engaged, 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. The dynamics of this instantiation process with narrative control supports trainee evaluation, engagement management and real-time assistance. The Narrative Manager is an independent module and is generic so it could be applied to other training applications.

We present also the results of two experiment testing different aspects of our application. The first one with students from the Faculty of Medical Sciences at Kyushu University where we investigated the effect of real-time task recognition on recovery time after user mistakes and the usability aspect by comparing interaction with mouse and Kinect devices. In the second one, in the National Institute of informatics in Tokyo we finally tested if users using our Narrative Manager improve their learning output. Our hypothesis is that the Narrative Manager allows us to increase the number of tasks for the user to solve but keeps the user interested in the training due to balancing difficulty and intensity. When evaluating our system we observed that the Narrative Manager effectively introduces more tasks for the user to solve, and despite of that, is accepted by the users as more interesting and not harder than an identic system without Narrative Manager. Also, by observing the knowledge test results we saw that the learning output increases, being a consequence of solving more tasks.

Keywords

Virtual Training Applications, Narrative Structures, User Interest, Narrative Balancing, Narrative Authoring, User Task recognition, Gestural Interfaces, Computational Narratology

Contents

Figures Index

Tables Index

1. Introduction

1.1. INTRODUCTION

Intelligent training applications have gained considerable importance in recent years, as they can help people to learn handling situations in which on-site training is dangerous or impractical. Here, 3D environments offer an ideal setting to develop training applications because of their ability to convey a greater sense of realism than textbooks or 2D interfaces (Stocker et al. 2011). The sense of realism is generally attained through accurate visual representation and physical behavior of objects. These 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 at 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 behavior (Carpenter et al., 2006; van Wyk & de Villiers, 2009; Belloc & Ferraz, 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, T., Bustard, 2011). In another work, (Reger et al., 2011) use a virtual reality simulation to treat posttraumatic stress disorder in soldiers obtaining a reduction in the symptoms showing that the virtual environment are effective in cases when real life rehearsals are not feasible.

Furthermore, the success of training applications depends on the correct integration of knowledge based techniques and natural interaction. Knowledge-based techniques are used to support the training procedures that users have to execute. Natural interaction, on the other hand, is used to support execution of those training procedures in a way that mimics real-world execution as close as possible. However, the more complex the training scenario is, the more effort is required in implementing both knowledge and interaction aspects in a way that delivers an accurate and realistic training experience. Also, designing new training scenarios and managing the events that happen in such interactive environments is a complex task, because 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 hypothesize that a narrative driven learning environment will increase the engagement of the user and this will increase the learning outcome. Narrative is a very broad term, and it mostly has been defined as an ordered sequence of events (Szilas, 1999; Young, 1999), but other authors add that the narrative has to generate also an emotional reaction in the user (Bailey, 1999), or that has to be generated following a grammar given by the author (Lang, 1999), especially when we are dealing with automatically generated narrative. In our case we will use the concept of Narrative as an ordered sequence of events, generated by an algorithm and that has the goal of being interesting for the user. Narrative has been designed mostly with traditional techniques like branching, when variability is a need for the training scenario, or using a static script when the designer wants the user to immerse in a more engaging experience (Ponder et al., 2003; Gordon, 2004).

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 the events in the session and show them in a more attractive form for the user. Systems with such a 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 modeling the grade of conflict the user experiences in each moment (Crawford, 2004; Ware, 2010). By modulating the conflict, a system can analyze when is appropriate to trigger an event, paying attention to the difficulty of the session and the dramatic intensity that will have on the user, ultimately affecting the user's engagement. We hypothesize that applying such method to training applications not only allows us to optimize the triggering of events in a training session while keeping the user interested, but also results in better learning outcomes. With improving the learning outcome we refer concretely to improving user's Knowledge acquisition which is defined by the gale Encyclopedia of Education as *"the process of absorbing and storing new information in memory, the success of which is often gauged by how well the information can later be remembered (retrieved from memory)"*. With this idea in mind, we notice that recently, complex training scenarios involving the handling and management of pathogens have received great attention. Medical educational institutes are considering bio-risk training as a requirement for medical researchers to strengthen their response capability to bio-terror crises (Gaudioso et al. 2009; Rao 2011). However, real-world laboratory practice is impractical because of its extremely high risk and expense. Thus, such scenarios are good candidates to evaluate the potential of combining a knowledge-based procedure for training and a natural interaction paradigm.

In this work, we describe the implementation of an intelligent application for biohazard training: Bio-safety Lab, which serves as a virtual training environment for medical students (Prendinger et al., 2014), developed in collaboration with the National Institute of Infectious diseases in Tokyo. The application emulates a scenario in which users must handle a critical biohazard situation involving the spill of a contaminated blood sample from a broken bottle. The system has been tested 'in the field', i.e., with students from the medical faculty in Kyushu University, another collaborator, who are the target users of the system. In order to test our hypothesis we developed a novel Narrative manager system (Alvarez et al., 2014) that controls the dramatic flow in a virtual training scenario aimed at practicing biohazard procedures. The 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's task recognition and has been tested already in two field experiments.

1.2. CONTRIBUTION

The main contribution in this thesis is listed in the next points:

- First, we developed a novel control system which manages the events in a training session using narrative techniques with the goal of enhancing the user experience by balancing the difficulty of the session. This model is the first narrative system with this kind of goal. The system maximizes the number of events for the user to solve, but instead of becoming more

difficult of boring as a consequence, we obtain the opposite effect, creating an interesting experience for the users.

- Our system not only enhance the learning output and interest levels of the training subjects, but also saves time and efforts to the scenario design: is not necessary anymore to redesign again the whole session when we want to create a different arrangement of events, because the Narrative Manager generates a different session each time the user plays it.
- Finally, we test and confirm our hypothesis: our narrative management increases the learner interest in the training and as a consequence his learning output increases as well. The users of our system perceive it as more interesting and challenging than a system without narrative management, and at the same time is not perceived as more difficult. Also the learning output of the users improves in comparison with the system without Narrative Manager.

This work also presents a number of secondary contributions:

- Also, the real-time instantiation of hierarchical task models to prompt the user on incorrect actions using descriptions about high-level goals, rather than concrete actions. We confirm that this method diminish the recovery time of the subjects in correcting mistakes (but it doesn't have much impact in the knowledge acquisition compared with other methods).
- Next, we integrate in our system a gesture-based interaction and present an experimental comparison of two types of control interfaces: gestural based and a classic mouse and keyboard control, studying the benefits of each one.
- Finally, we present a high level authoring tool for designing the narrative flow of the session.

1.3. STRUCTURE

The remainder of the document is organized as follows. The following chapter reviews previous work on virtual training applications, with subsections analyzing gesture-based interfaces and training tools with narrative control, and compares our system to those works. Thereafter, in the chapter 3 we will introduce our system, which uses a drama manager to control the events in a biohazard training environment. The chapter is divided in three subsections: first we presents the system architecture and its components for natural gesture interaction, common sense reasoning, and task recognition, then the second subsection explains how user gestures are interpreted into high level actions, and finally we introduce the narrative techniques used in our training system, the core innovative contribution of the present work. Chapter 4 shows the experiments we carried out with our system evaluating different aspects: first, Kinect and mouse versions of our system are evaluated in terms of usability. Second, a 'dynamic' advice version based on real-time task recognition is compared to a 'static' advice version, where users can ask for text explaining the procedure. And finally, we present an experiment conducted to investigate the engagement and training effects of our system. In the chapter 5 we discuss the outcome of the experiments, especially in the case of the third one, analyzing extensively its results, as is the experiment which tests if our hypothesis about narrative management is correct. Finally, chapter 6 presents the conclusions of the document and indicates future steps.

2. Related Work

2.1. VIRTUAL TRAINING SYSTEMS

Virtual training systems vary greatly in the method used. Self-assessment testing systems appear frequently in the literature; in these systems the users are presented with questions and situations they have to solve, and only advance to the next step when they give the correct response. For instance, in (van Wyk, & de Villiers, 2009) an application designed to train accident prevention in a mine makes the user to check videos of a scenario and to detect dangerous spots, only advancing when the user give the correct answer. Other example of these self-assessment testing systems is the one in (Corato et al. 2012), a computer vision application for detecting whether the protocol for washing hands before an operation in an operations room is followed. It uses augmented reality technology but does not need markers for detecting actions. A limitation of the work is that even if the system recognizes the stages in the protocol, it does not have a provision for responding to user mistakes. In other words, that approach does not provide student monitoring or dynamic help or assistance. These systems allow a high degree of visual immersion thanks to 3D graphics and videos, but lack flexibility regarding the user's interaction with the training scenario.

To improve in those aspects, many works on training systems integrate interaction techniques with knowledge-based methods. They share the goal of improving the efficiency of knowledge transfer in situations where text and pictures are insufficient for training, and the conviction of the adequacy of the approach to better learning procedural tasks. Similar to us, some authors present systems with intelligent object behavior and manipulation capabilities. This additional functionality requires a more complex architecture. In the case of adding complex behavior to the 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 the system presented in (Kallmann & Thalmann, 2002) that include common sense behavior for objects which is difficult to implement with only a physical engine. In other example, (Angelov and Styczynski 2007) (see Figure 1, left) present a virtual 3D simulator that aims to teach the maintenance of a power plant. The simulator is web-based and allows interacting with a 3d model of the machinery, dismantling into its components or modifying its work parameters. The knowledge representation of the scenario is given by a specific language model allowing the designer to author different learning modules. In a scenario, the user has a number of goals and he has to switch the components of the machinery in the right order. The interaction level is high event if the user doesn't control an avatar neither manipulates directly the objects. However, the scenario is the same every time the user plays it, lacking flexibility. (Belloc et al. 2012) presents a generic system for scenario description, intelligent objects, and protocols. Similar to our system, the application is separated into three layers: graphics and physics, objects and behaviors, and plans. The system is designed for web platforms in order to allow max accessibility and uses petri nets as a model for the plan definition. They present two examples: the assembly of a hydroelectric generation unit and the assembly of a combustion engine. Once the objects are described, the training scenario is fairly simple: the user has to perform a series of correct steps and is able to proceed only when previous steps are completed (see Figure 1, right).

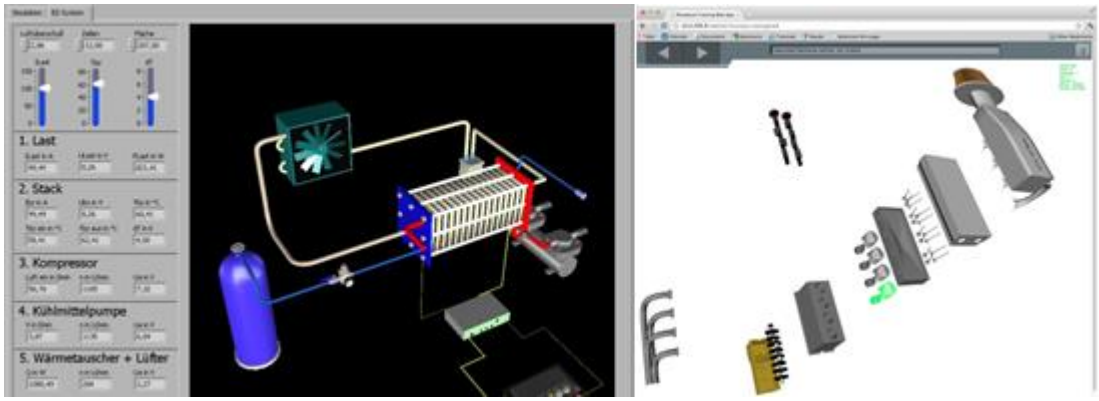


Figure 1: Two training system for learning machinery maintenance: The Virtual Circuit Breaking training module described in (Angelov and Styczynski 2007) (left) and the vehicle combustion engine depicted in X3D from (Belloc et al. 2012) (right).

There exist several works with virtual tutors, or online assistants. (Johnson and Rickel 1997) describe Steve, a virtual tutor system for teaching air compressor functions. A planner is used for executing scripted exercises and for explaining the rationale of the tutor's actions, but the system does not provide information about user mistakes. If the user makes a mistake, the tutor will correct the user telling the next step. Including a virtual tutor is not incompatible with other techniques, such as storytelling techniques. In Crisis Communication, (Carpenter et al. 2006) describe a virtual training system that teaches students how to manage the communication of crisis related information (see Figure 2). The system's knowledge is stored as an event tree that works as a storyboard with different choices. The users can take decisions that have consequences in the scenario, and use 3D vision glasses and six screens to facilitate immersion. However, the user does not affect the scenario or its objects directly, only giving directions from a predefined set of options to the non-player characters. This kind of scenario scripting is very costly: not only dialogue has to be recorded for every option, but when the designer wants to add content to it, he has to take in account all the different combinations of situations, growing in complexity exponentially; we will expand about the problem of branching in the next sections, but the reader can see that it's a common trait of training applications. (Cavazza and Simo 2003) uses a qualitative simulation model to train medical students to respond to patients with circulatory shock states. Like ours, this approach integrates knowledge-based techniques with a 3D environment in the health domain, but focused on the virtual patient rather than interacting with the environment. Similar to (Gordon 2003), we use a graph containing the description of the training scenario. That work describes a protocol of identifying and constructing decision structures in virtual environment scenarios. The result of this process is a decision graph divided into stages: training scenarios are constructed like storyboards, and branching trees are used to simulate training decisions. While our representation shares some similarities, training in that work is carried out by selecting text options in a web page, rather than by interacting with a complex 3D environment. Further, (Vidani and Chittaro 2009) proposes a task model (called concur task tree) to train emergency procedures that may assist disabled people. The model aims to be a complete representation including time relations between events and task difficulty management, but it is unclear to what extent features of the system are implemented. Also, the work does not separate difference knowledge layers, such as task definition and common sense knowledge base, which limits extensibility.

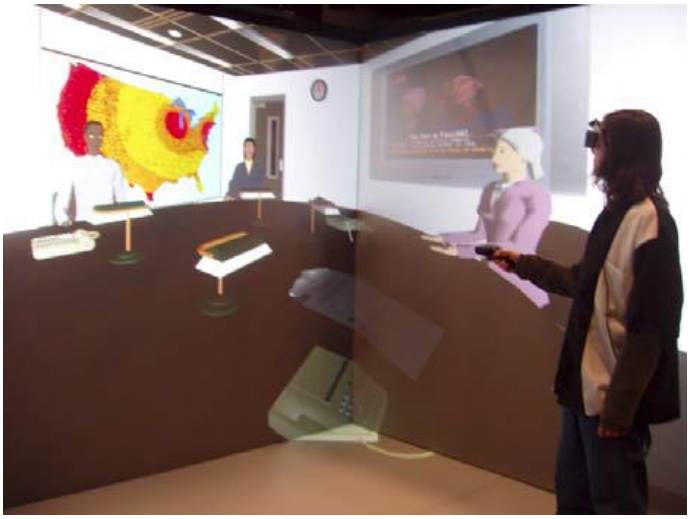


Figure 2: A user participating in Crisis Communication. The training system uses four screens and stereoscopic glasses to improve immersion.

More complete virtual tutoring systems, in which the tutor not only corrects, but also modulates the simulation depending of the user's actions, can be seen in (Lebeau et al., 2010). There, the authors present ASTUS, a learning system which uses plans with sub-goals that containing a list of checklists defining the required users' actions. The system allows making queries over the elements of the checklist (called knowledge elements) to see which ones are completed. It does not have a real-time 3D environment, but its knowledge model is similar to the one we use in Bio-Safety Lab, allowing also to define potential incorrect actions for assisting the users better. Similarly to some of those works, our Bio-Safety Lab includes both intelligent object behavior and a real-time interactive tutoring system that corrects user's mistakes, but on the top of that we add a narrative layer controlling the simulation. Also, like in the more complete works we reviewed in our system we separate the knowledge model from the simulation: in our case we use a task model for task recognition, a technique similar to plan recognition. Thus, we gain independence from the platform and scalability, but also we can perform advanced reasoning techniques on the knowledge. In the next section we will explain in detail the characteristics of this strategy and show representative examples in the virtual tutoring applications field.

2.2. SEMANTIC REPRESENTATION AND PLAN RECOGNITION IN TRAINING APPLICATIONS

Real-time interactive systems with semantic representation of events and plans need to be decoupled from the internals of the simulation engine; otherwise the semantic layer would be a static specification only working in one domain for some concrete cases. To achieve this goal, researchers have developed several solutions, generally based on the idea of using different layers to represent different types of knowledge (graphical, physical, common-sense, actions and goals, etc.) and some communication mechanism among them. In our system we include these capabilities in two modules:

one allows us to perform this decoupling between the native events triggered in the virtual environment and the semantic consequences of them, and the other is capable of recognize user actions and identify them as part of a plan. Although several general purpose architectures exist, there is no standard, so each system usually defines its own so in this section we review different approach to create such a system and compare it with ours.

For example, (Latoschik and Frohlich 2007) describe a solid architecture for interactive systems based on the concept of semantic reflection that uses monitors, filters and subscription lists for decoupling all the possible systems that can participate in a virtual environment. However the architecture does not specify a concrete semantic module or layer: each module can equally access the system's knowledge base. It is generic to the extent that implementing such a layer is possible, which is left to a future developer. In further research, (Wiebusch and Latoschik 2012) present an architecture targeting the use of common sense knowledge and grounding and also using a message dispatching system. It includes the OWL (Web Ontology Language) ontology as a central knowledge base to unify access to information from different modules. Our system also decouples different types of knowledge in semantic modules. In particular we developed a common sense database and a task recognition module, but communication is performed module-to-module instead of using a centralized database.

Taking into account common sense knowledge and information on the physical state of the environment, (Schneider 2010) proposed a plan recognition approach based on the user's plan selection and actions that are represented as a probabilistic automaton. This automaton is converted to a dynamic Bayesian network allowing the plan recognition process to occur at runtime. This approach is similar to ours because it also applies real-time monitoring, trying to infer the intended plan by observing the user actions and the states of the environment and it also tries to predict the user's probable next actions so that it can pro-actively provide additional instruction on how to currently execute the chosen plan. Our approach, on the other hand, relies on a hierarchical representation of tasks such that complex goals can be decomposed into simpler ones. This representation is more intuitive for the experts in our domain than a flat representation, because security procedures are usually represented this way. It also allows providing feedback to the user using abstract description of goals instead of concrete actions. Finally, we do not use probabilities to compute the next most probable task to complete, we use a simple heuristic based on hierarchical nature of our plans that seems sufficient for our purposes. The use of Dynamic Bayesian Networks to achieve intention recognition can also be applied to different scenarios: In (Tahboub 2006) the probabilistic intention inference is achieved by modifying the intention-action-state scenario and modeling it by Dynamic Bayesian Networks.

In the literature we can also find systems specifically designed for e-learning and training purposes and using plan recognition techniques: for example, a complete system (Franklin et al. 2002) designed to assist the user in real-time by recognizing activities that are sequences of processes represented using finite state automaton. The system contains features similar features to ours, such as backtracking and real-time monitoring, but it neither uses hierarchical plans nor has it been applied to a complex virtual environment. In (Amir and Gal 2011), the authors present a solid validation of task recognition in virtual laboratories with an interactive application that recognizes the user's current task. Task recognition is performed using grammar rules that represent tasks. They evaluated this method

and proved it to be effective. The main difference to our approach is how we represent the possible hierarchical plans: We use an explicit task network decomposed a priori instead of an implicit representation based on grammar productions. One advantage of our explicit representation is that it can be represented visually, thus easing the procedure management. Another important difference is that our system provides real-time assistance to the users during the simulation. Another application of these techniques is a framework built to assist in the development of systems for the recognition of high-level surgical tasks using analysis of videos that were captured in an operation room (Lalys et al. 2012). Here, the processing is based on dynamic time warping and hidden Markov models. As processing occurred after all the data was collected, it is not a real-time application. Also, plan recognition was used in a domain in which users engage in exploratory and error-prone behaviors (Gal et al. 2012). There, constraint satisfaction algorithms were used as a viable and practical approach for plan recognition in an educational software application similar to our intelligent biohazard training system.

2.3. VIRTUAL TRAINING SYSTEMS BASED ON NARRATIVE CONTROL

Systems that have architectures with plan recognition and semantic representation of events are able to generate complex and realistic training simulation, allowing us to define in some way the goals and rules of the training. We showed in the first section of this chapter that systems like those presented good results in user's learning acquisition and presence, but we think that that performance can be improved further. As we stated, our proposal is a narrative control layer in the system, deciding what events will happen and when, but narrative can be a viable strategy to improve a training application? This section explores deeper in this topic, surveying relevant research that have been done in the field.

Previous research proved that a narrative centered learning environment increases the engagement of the user (Gee, 2003; Shaffer, 2006), but testing if engagement is linked to a positive increment in the 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 than 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 the learning.

So, why are the results so different? The cause of this discrepancy in the results of those works is shown in a subsequent experiment where Rowe et al. (2009) 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 a mechanic 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. Other example is the STAR framework (Molnar, A. et al., 2012), that using the same strategy showed good results in testing an application designed to teach health practices to students. In that application, the user takes the role of a detective who has to solve the mystery of a poisoning crime, obtaining clues about bad habits that can end in getting sick.

We can find that the works that found narrative as a learning dampener included narrative elements only as an independent addition. This design flaw can be the reason for the poor results of the narrative strategy, because it was not affecting in any way the mechanics of the interactive learning; in sort, without the narrative elements the application would remain the same. In the French teaching system from (Hallinen et al., 2009) the narrative elements included are a score system in the form of earning virtual money due the user taking the role of a newspaper corrector, but this mechanic of obtaining money, doesn't affect or help the student in the application, so the narrative is independent of the user actions. Also, the math teaching application from (Rai et al., 2009) uses drawings and customization such as allowing users to have a virtual pet, but only as an interactive application, showing graphically math problems, but not affecting the exercises presented. In our Bio-safety Lab, we use narrative techniques to generate the events in the scenario, giving to the user new tasks to solve or helping him. These events affect directly the type and number of learning content the user will experience, so we can affirm that the narrative is tightly integrated with the 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 & Csikszentmihalyi, 2002). The consequence of this effect is to persist or return to the activity because is psychologically rewarding which is in line with our hypothesis.

Aside of 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 decision to manage a crisis. However, the user still has limited interaction possibilities only involving the user 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 in which training scenarios are constructed like storyboards, and applies it for 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 in 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. Similar to them, 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 scripts that control the virtual world happenings (Mollet and Arnaldi, 2006). The system allows virtual training for maintenance of industrial equipment and is being used, but the

narrative authoring and its impact in the training wasn't explored in their works. Other 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 brain-damaged people. The application uses IDtension, a drama manager engine that controls the simulation and does not rely on branching enabling a large number of choices. This method proved to be very immersive and engaging for the users, allowing them to find themselves in a realistic environment where unexpected things can happen, but still it wasn't evaluated in terms of learning, and it does not tutors the user at all: neither contains an environment with intelligent behaviors (everything is decided by the drama manager) nor warns the user about incorrect actions, being an experience only narrative without training elements in which the user interacts with characters and see what happens.

2.4. AUTHORING TOOLS FOR VIRTUAL TRAINING APPLICATIONS

As stated in the previous section, including a drama manager can be a good solution for the problem of authoring learning or training scenarios; however it adds another necessity of authoring. Instead of the scenario, in this case it is the tutor system the one that we would like to design in an easy way. By implementing a good authoring tool, the training expert can design directly the tutor system and the scenarios without having to know scripting or programming languages, and also the system will adapt better to its users (Sottilare & Holden, 2012). On the other hand, depending of the complexity of the application, the authoring tool may require some additional knowledge for use it. In simple learning applications that work as a series of self-assessment tests, the authoring tool consist in a simple to use graphic editor with drag and drop features. Interestingly, an Intelligent tutor authoring tool that uses crowdsourcing for creating its knowledge database showed that even if the results are not bad with this strategy and is faster to create a database this way than using a domain expert, has a low recall and is less reliable (Floryan & Woolf, 2013). It seems that the help of a domain expert is necessary for creating good knowledge bases, so it is important to have a complete but easy to use authoring tool in order to minimize the deploy of the training or learning system.

For example, (Tripoliti et al., 2013) presents a complete tool for creating tutors for medical diagnosis training. The tutor works as a series of steps that have to be completed, and the tool allows customizing the steps in a graphic editor with drag and dropping features. Another work, (Shaker et al., 2013) presents a tool for designing scenarios for a physics game with real time interaction in 2D scenarios, so is more complex than a set of self-assessment tests. The scenario editor goes further than the previous example in the usability, containing "wysiwyp" capabilities (what you see is what you play), allowing to play the scenario inside the editor in order to fine tuning its features. However, the increased complexity of the learning application requires a more careful design in the scenarios: impossible or unplayable scenarios can be created and even if the scenario is correct there is a possibility for being not satisfactory or boring (too easy, too difficult), so the designer has to test all the details and possible situations. Incidentally, this problem can be solved with the addition of a scenario director layer controlling the training, like our Narrative Manager, allowing the scenario designer to not to worry about if the training is too easy or not.

More complex tutors can be designed in the Chocolato system (Isotani et al., 2010), using ontologies and allowing the selection of the type of pedagogic theory used and the user's goals. In this case, the authors confirmed than the authoring tool effectively improves the performance, saving

efforts and time to the domain experts and allowing a rapid deployment of the learning system. Other example can be found in the work of (Karampiperis et al., 2011) describing the ER designer toolkit which essentially is a trigger listener editor. Its graphic environment generates the event calculus code necessary to define the virtual tutor behavior. However, they don't complete the tool with a responsive training environment, so the human domain expert has to decide what to do when the virtual tutor detects the user's actions. In comparison with our system, this one would be similar to our Task Recognition system, but lack a layer controlling it (our Narrative Manager), with an authoring tool to define how this control layer will act.

We can find another advance tool with more common points with our system in (Cablé, 2013), which presents an authoring tool for an e-learning platform. Though the system presents a simple form of training (only present the user problems with only one answer en a series of self-assessment exercises), the customization capability is wide, allowing the selection of the resources used for the learning and how they will be used. This platform have several similarities to our Bio-Safety Lab, in the sense that they have different layers of knowledge, separating the raw data containing information about exercises, defined in a knowledge base by an expert, from the resources that define an scenario, describing what data is used and how they will be used in the learning. Aside this similarity in the architecture, the authoring tool allows a high grade of variability because the scenario designer does not specify all the details of the session, but only give a pattern of how the tutor will act. This is again a common point to us, because our Narrative Manager acts as an independent entity, only requiring general directions from the scenario designer to act (in our case, the description of the events and how in general the session have to go by). However, the system is not fully operational and it was not tested. Other interesting tool is seen in (Olsen et al., 2013, June) including a drag & drop interface and a knowledge model based in graphs, allowing editing the nodes and links graphically. This behavior graphs (as they are called in the paper) are similar to the ones we use. The graph traversing engine allows branching and collaborative work between users. However, our model is more sophisticated, containing different types of parameters and groupings. Other example of a training system with high level author capabilities can be seen in (Sottilare et al., 2012), but again, it is not fully operational yet, showing examples like a thermodynamics tutor for undergraduates (Amin-Naseri et al., 2013). The application allows designing virtual tutors by creating behavior rules (mainly rules of condition-consequence about user actions), customized questionnaires and profiling the users. The system can be plug to other simulation frameworks (Ray & Gilbert, 2013), in this case Unity 3D, like our Bio-Safety Lab.

Creating behavior rules is a good way to design the tutor behavior. We can see this strategy taken to a step further in the CTAT system (Aleven et al., 2006). This generic virtual tutor designer uses jess rules (a programming language for creating expert systems) and a database with "memory elements" for defining the goals of the users, and even the graphic interface of the learning application. Similar to the ASTUS system, allows designing almost any kind of tutor; in a sense both are equivalent, as seen in (Paquette et al., 2010). However, its authoring language is not accessible for non-programmers, having in this case the problem of needing an authoring tool for the authoring tool. In our case, the authoring tool contained in Bio-Safety Lab fits in that requirement, being an easy to access designer to model the tutor language.

Finally, when the training applications have special requirements i.e. a collaborative tool, the authoring system needs some additional features. An example can be again the CTAT system, which needs some adaptation to suit the requirements for collaborative learning applications (Olsen et al., 2013, January), or in the case of augmented reality learning applications, which add the tagging and optic recognition requirements of this kind of systems (Jee et al., 2014).

2.5. GESTURE RECOGNITION IN VIRTUAL TRAINING APPLICATIONS

The last feature of our Bio-Safety Lab is that, with the goal of improving the user's immersion, we decided to implement a gesture recognition interface aside of the classical keyboard and mouse one. Gestures are convenient for use in Virtual Reality environments such as a CAVE (Cabral et al. 2005). In training scenarios where physical interaction is important, gesture based interaction has a great potential as an object manipulation mechanism: not only does it allow for an optimal task execution (i.e. gestural interactions accurately mimic events in the real world), but also enhances the immersive attribute of the whole virtual environment. For example, a gesture-based immersive environment has also been shown in increase children's engagement with learning materials (Scarlatos and Friedman 2007). Under these considerations, we implemented a gesture recognition framework using Kinect[1] that would allow users to have a more natural interaction with the environment. By using gestures, users would be able to accurately mimic real-world actions, particularly in terms of taking, carrying, and releasing objects.

In the literature, there is a growing body of works dedicated to natural gesture based interaction (Kristensson et al. 2012; Song et al. 2012). Table 1 contains a proposed classification of the tutor system depending of its interaction capabilities (Sottilare & Holden 2013). Other example of an application with rich interaction is in (Gutierrez et al. 2010), where they use a dynamic setup of haptic devices and motion capture for training in industrial maintenance and assembly tasks. The tasks that they intend to teach to users involve high precision and coordination, which justifies their need for such a complex setup. Unfortunately such equipment is not only difficult to use, but also expensive and not available for many users. However, there are some proposals in the market much more accessible. This is the case of Microsoft Kinect and Asus Xtion which public SDKs are publicly available.

We use a Kinect sensor for detecting basic hand gestures and body position, because high precision is not required in our system and it makes easier to set up a medium scale experiment in money and equipment. For the same reason, it was not necessary to apply a complex technique for detecting the hands as the one used by (Oikonomidis et al. 2011), who investigated hand articulation tracking in near real-time using only visual information. This was achieved by formulating an optimization problem that minimizes the discrepancy between the 3D hand structure and the appearance of the hypothesized 3D hand model. While our gesture detection is not as precise, it is sufficient for our purpose and most importantly, it is in real-time, aiming for an interaction mode between limited kinetic and enhanced kinetic (see Table 1). Kinect incorporates a color camera and a depth sensor that theoretically provides complete body motion capture and face recognition. Kinect also have a microphone array and supports voice recognition, although in this work we are only focusing on the motion sensing capabilities. Kinect has its drawbacks when compared with more complex equipment. For instance, although Kinect is a very good "skeleton recognizer", it usually has problems detecting small object like fingers, due to the maximum resolution of the sensor (640 x 480 pixels), so if we want to do face or hand tracking, players would be very close to the sensor. Kinect can do head pose, hand position, and facial expression tracking, but the depth sensor is not very accurate for this goal (Zhang, 2014) and that means a lot of raw data has to be processed.

[1] http://www.microsoft.com/en-us/kinectforwindows/

Interaction Mode	Environment	Learner Position	Learner Motion	Sensors	Sensory Interaction	Individual/ Team
Static	Indoor	Only seated	Head motion, posture changes, gestures	Eye trackers, head pose estimation	Visual, aural, olfactory	Individuals and network enabled teams
Limited kinetic	Indoor	Standing and different positions	Same as static mode plus limited locomotion	Same as static plus motion capture	Visual, aural, olfactory, haptic	Individuals and co-located teams
Enhanced kinetic	Indoor	Standing and different positions	Same as static mode plus full locomotion	Same as static plus motion capture	Visual, aural, olfactory, haptic	Individuals and co-located teams
In the wild	Outdoor	Standing and different positions	Unrestricted natural movement	Portable sensor suits including motion capture	Visual, aural, olfactory, haptic	Individuals and co-located teams

Table 1: Interactive Tutoring Systems' interaction modes.

In our case, we also did not have to adapt the recognition system to different users, as done by (Hasanuzzaman et al. 2007). The authors recognize that there is an issue in the ambiguity when decoding gestures: the same one can mean different meaning in different cultures, or on the contrary, different gestures might have the same meaning – for instance, the gesture recognized as "OK" is not the same in every country – but the gestures we are recognizing are not attached to any cultural dependent meaning: everyone needs to close and open the hand when grabbing and dropping objects. The only gesture that could have a meaning is the one asking for help. We avoid this problem by giving specific instructions to the user on what gesture they should use. (Unzueta et al. 2008) proposed a way to distinguish between static and dynamic gestures, i.e., gestures that require one step and gestures that require more than one step, respectively. In our system there is only one case of a dynamic gesture: moving the hand back and forth for using disinfectant. However, only one of our static gestures might be confused with this – the help gesture. Because this a straightforward case we opted for a much simpler alternative to the method described by (Unzueta et al. 2008). We just consider that the user is performing the help gesture if he or she maintains it for more than two seconds.

3. System Architecture and Implementation

3.1. Introduction to Bio-safety Lab

We created Bio-Safety Lab as a collaboration with the National Institute of Infectious Diseases in Tokyo in order to response a need in the pedagogical field: students that had recently graduated have few or none practical experience in laboratory practices aside from memorizing manuals, especially regarding the problems that may arise when handling dangerous substances. Our system recreates a biohazard laboratory (Level 2) from a user-centered perspective and supports realistic gesture-based interaction in a way that reflects real-world procedures. The design of the graphical environment and the domain knowledge was done in stretch collaboration with experts from the National Institute of Infectious Diseases. In Bio-safety Lab, the user takes the role of a bio-safety laboratory worker from a first-person point of view. The user assumes the role of his avatar using two interface modes: gesture recognition with Microsoft Kinect, and with keyboard and mouse. The user directly controls with this interface the avatar's right hand (in the case of Kinect he can control both hands at the same time), allowing him to interact with the scenario's objects by doing right or left click: possible actions are: taking or dropping objects, open or closing doors, or use some concrete objects. The use of a 3D content development engine (Unity3D) provides a unified mechanism for visualization and interaction by taking advantage of the built-in mechanisms to define the physical behavior of objects.

The training system in Bio-Safety Lab assists the user in the process of resolving the accident through real-time (user) task recognition, consisting in a representation of biohazard training procedures as task models, which can be instantiated in real-time via the recognition of task-level events. Thus, the application implements a Task Recognition Engine, a module who acts as a virtual tutor, deciding whether the user is doing properly the task of clearing the accident or not, and gives advices about 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 the Kinect version of the application, task-level events can be derived from gestural interactions between the user hands and laboratory objects based on semantic properties attached to objects and common sense reasoning.

As the main contribution of this work, 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. Our Narrative Manager decides how and when certain events will happen in the scenario, to either (a) decrease task difficulty if the user has troubles to complete 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 an interesting narrative entangled with the training process, maximizing the number of events for the user to solve, and thus increasing the possibility of learning acquisition.

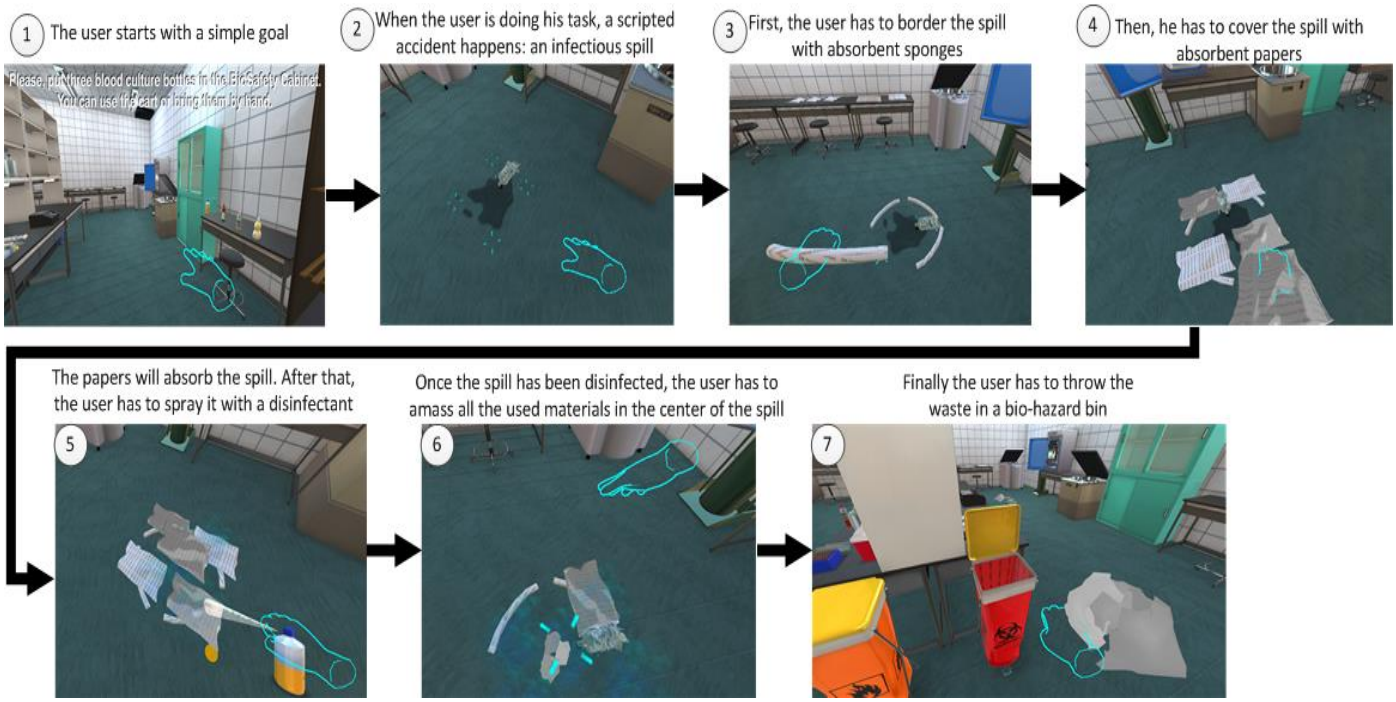


Figure 3 Sample session and Spill cleaning in Bio-Safety Lab. The user controls the avatar's hand and performs the different steps (steps 3-7) in the protocol for cleaning the toxic spill.

Figure 3 shows a sample of a typical session in the system. After start, the user is given a simple goal to perform: a routine laboratory procedure (see number 1 in Figure 3). 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 it was given by our collaborators in the National institute of Infectious Diseases of 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 who acts as a virtual tutor, recognizing whether the user is doing properly the task of clearing the accident or not, and gives advices about 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.

The training system in the Bio-safety Lab is inspired by Marc Cavazza's Death Kitchen (Lugrin & Cavazza, 2006), an application that merges narrative management with complex scenario behavior, in where the system generates events by itself. Death kitchen (depicted in Figure 4) is a narrative application where the user controls an avatar in a kitchen while the system generates accidents and tries to "kill" him. It merges together the narrative with a layered behavior intelligent system: the high level events that happen in the scenario are captured by the Causal Engine, an intelligent object's behavior module that decides the consequences of the event and acts as a reasoner and a 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.

Figure 4: Marc Cavazza's Death Kitchen. The user interacts with different objects and the system generates events in the form of accidents in order to get the avatar killed.

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 also it uses the events to perform user's task recognition for giving real-time assistance (Death Kitchen doesn't have it). 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 to the user using a knowledge structure known as Task Trees, allowing different constraints when deciding to select the event and they are used too when recognizing the user's task.

Other difference appears when deciding the 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 the identical accident if the user goes to the same area in each session. The system works in this way because by design, Death Kitchen doesn't require more complex decision taking but in our case we need more complex behavior in order to optimize the triggering of events. Looking for 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 relaxing the intensity of the events or event giving some help when the task is too hard for the user. For example, if the user is in the step of containing the spill, the system could decide if the spill will overflow its boundaries, making the user to hurry up and re-contain it. Or if the user is cleaning the spill too easily, the system can knock a culture bottle making it fall to the floor and break, creating a new spill (of course this can

happen only if the bottle is available). On the other hand, if the user is having too many troubles for solving his task, the system can help by hinting where should the user go and what he should do. This way we can introduce more events for the user to solve and at the same time keeping him engaged. Research in the narrative field states that creating a sense of conflict in users, increases their engagement (Gerrig, R.J., 1993), and also conflict has been well defined and modeled (Ware, S. and Young, R., 2010). They 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 doesn't introduce the conflict in the stories automatically but it only allows as one of the possible outcomes: it is left to the player to aim for it. Also they did not use the dimension of the conflict they defined in their work, and they admit that is difficult to find a perfect solution of how to include them and that they are 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 include to give manually values to some parameters) to measure them, and in the results there was agreement between the two quantifications so the dimensions can be recognized as qualities of conflict.

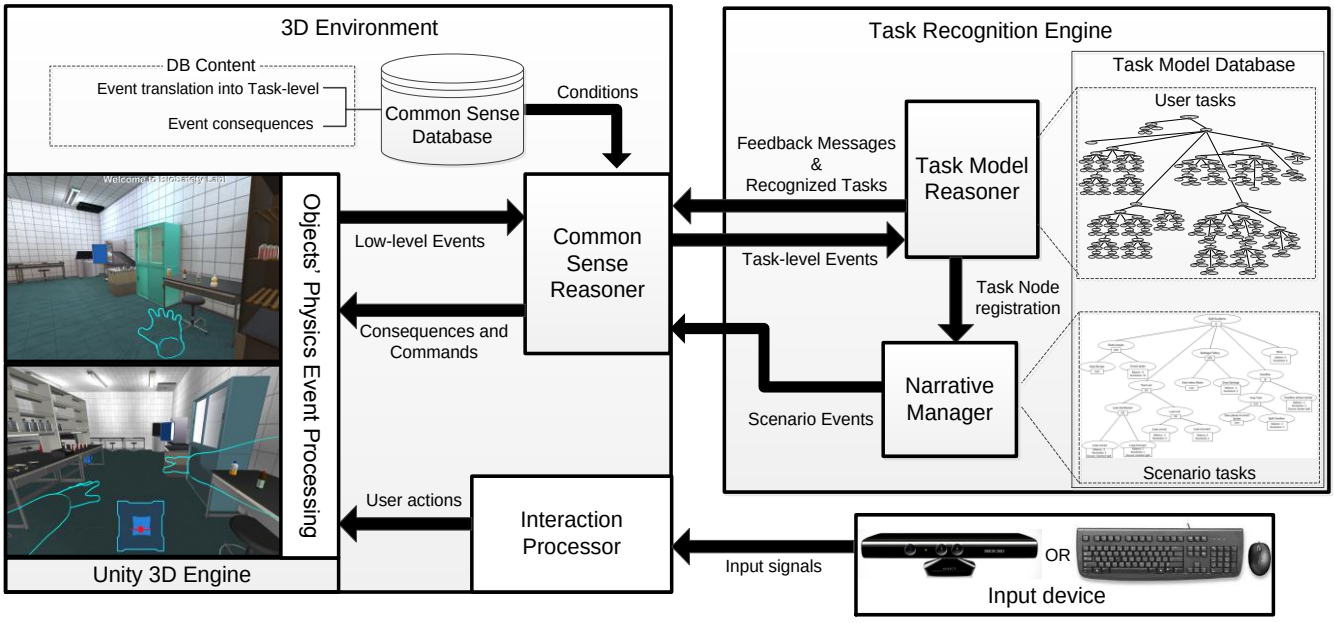


Figure 5 SystemArchitecture in Bio-safety Lab. The Low-level events generated in the environment are translated into Task-level events and sent to the Task Model Reasoner where the state of task will be updated. Then the Task Model Reasoner will correct the user if he/she made a mistake and the Narrative manager will decide if trigger a Scenario Event.

The architecture of the system is detailed in the Figure 5. The system has two separated parts: the interactive 3D Environment Module and the Task Recognition Engine. The first controls the virtual world where the user participates and the latter being independent of the simulation. This 3D Environment contains two modules: the Interaction Processor, in charge of managing the user interface and the Common-Sense Reasoner. The interaction Processor contains the mouse and keyboard interface as well as the Kinect interface. It transforms the user commands and gestures into native instructions for the 3D engine called Low-level events. A Low-level event is an atomic action

which has some immediate feedback in the graphical engine; for example, to move across the environment (a "move" Low-level event) is corresponded immediately by a displacement of the avatar, or taking and object (a "take" Low-level event) triggers an animation of the avatar's hand. Additionally, Low-level events can require some further processing to generate additional consequences or complex behavior. For example, the afore-mentioned "move" event across the room can make the avatar to bump into other object and knock it over, or the "take" event can trigger the attachment of an object to the avatar's hand, effectively taking it. In order to process this reasoning, the next module of the 3D Environment Module takes in action: the Common Sense Reasoner.

The Common Sense Reasoner translates Low-level events native to the 3D engine into high-level ones called Task-level events and send them to the Task Recognition Engine. This translation is done by looking in 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 is 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 find 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 in pieces.

The Task Recognition Engine is an independent system in charge of receiving Task-level events and evaluating the user performance and sends orders for generating events or messages back to the virtual environment in response. When it receives an event, it decides if it fits in the tasks stored in a Task Model database. If so, it can evaluate how the user is doing his task, send him warnings if he does a mistake and send back events in order to keep the user engaged. Also, if the system decides the user is having a difficult time it will send orders to help him; on the contrary, if it decides that the current task is too easy for the user, it will order for more tasks to be created. The engine contains two connected modules: the Task Model Reasoner and the Narrative Manager. The Task Model Reasoner is the module that receives the Task-level events from the aforementioned Common Sense Reasoner and is in charge of providing assistance to the user by supervising his actions and sending him messages whenever he makes a mistake in the task he is currently doing. Then it pass the received Task-level event to the Narrative Manager which also process it and sends command back to the Common Sense Reasoner to trigger events in the 3D virtual world in order of keep the user engaged in his tasks. This 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 different training systems provided that a Task Model Database for that domain has been defined. In the following subsections each module in the system will be described in depth.

3.2. THE INTERACTION PROCESSOR

The Interaction Processor provides two types of input methods: Keyboard and mouse device input as a traditional method and gesture based interaction using Kinect as an alternative method. These two modes of interaction answer a requirement of more realism from our collaborators in the Nation Institute of Infectious diseases, the potential users of the system: in training scenarios where physical interaction is important, gesture based interaction has great potential as an object manipulation

mechanism. It may support natural task execution as gestural interactions accurately mimic high-level events in the real world. Additionally, it may enhance the immersive attribute of the whole virtual environment (Sridhar and Sowmya 2008; Lu et al. 2012).

3.2.1. Gesture Recognition Interface

Based on these considerations, we implemented a gesture recognition framework using Kinect that allows users to interact naturally with the 3D environment. By using gestures, users would be able to accurately mimic real-world actions, particularly in terms of 'taking', 'carrying', and 'releasing' virtual objects. While gesture based interaction is used in our setup, it is not the focus of our research. Hence we had to compromise on some aspects of the interaction. First, the version of Kinect used was the Xbox version of the device that demonstrated some spatial constraints for users. For instance, the tracking of the user's body and hands cannot be closer than 1.6 meters. Second, if someone other than the user entered in Kinect's area of effect, which expands to 2.5 meters, Kinect started to detect both of them and malfunction until the other user left. Despite these issues, the device had a high degree of robustness when used in the appropriate (albeit slightly constrained) range of detection and environmental conditions.

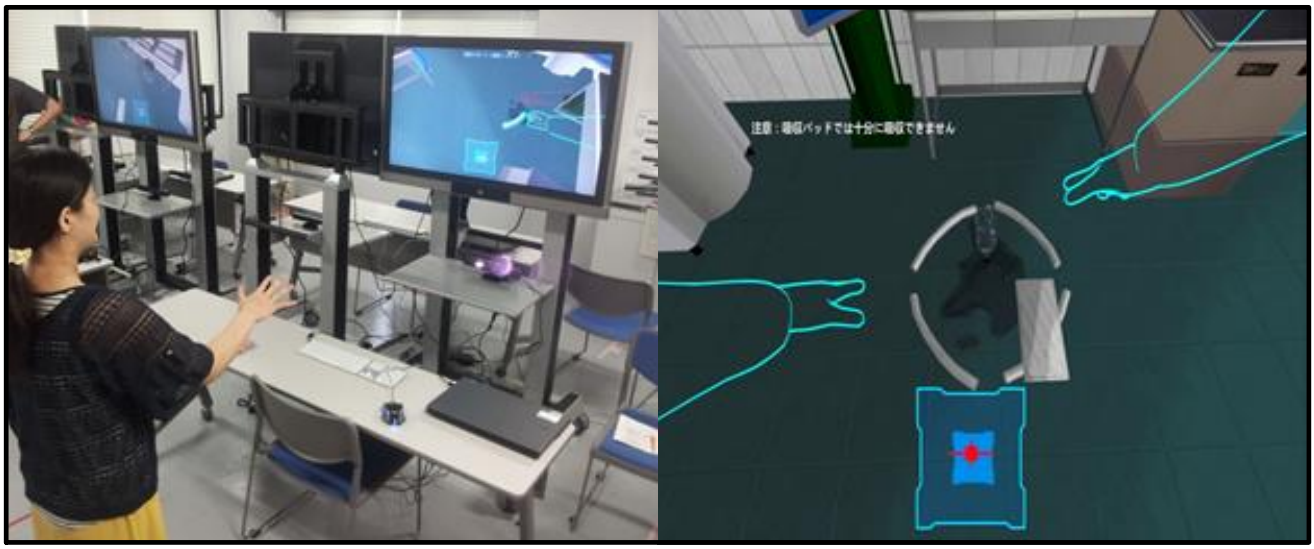


Figure 6: The Kinect control interface in Bio-Safety Lab. A user is standing in front of the Kinect device and controlling the avatar's actions (left). The user can see his/her position inside the blue square in the bottom part of the screen, represented by a red mark (right). The inner blue square represents the area where the user can move without move the avatar. The outer square is the area in where the user has to move for moving the avatar across the room. Outside of the square, user's movements are not detected.

There are two main types of gestures that needed to be defined in order to cover all operations in our Bio-Safety Lab: (1) gestures to navigate inside the 3D environment, and (2) gestures to manipulate objects, corresponding to the take object, drop object, use object, open object, and close object user actions. Execution of these gestures is indicated by showing two virtual hands and arms that mimic the detected gesture (see Figure 6). Additionally, if the system detects the user's hand is over an object, it will highlight it and show a hand suggesting the grab gesture (see Figure 7-a below). The goal of this feature is to better inform the user on the available actions at any given moment. The design of

navigation gestures was a challenge because we needed to define a natural way for users to indicate motion in the virtual world while remaining inside a limited area in the real world. Thus, for moving around the virtual world we decided to implement a control scheme based on the placement of the user within two squared areas (we can see it in Figure 6 in the bottom side of the screen).

While the user is inside the inner area, the avatar does not move. If the user wishes to move in any direction, he or she just needs to step outside of the inner area in that same direction. The further the user moves in that direction, the faster the avatar will move in the virtual world. The outer area is used both as a way to measure the velocity at which the avatar should move and as a way to inform the user of the limitations of his movements. If the user steps out of the outer area, the Kinect motion detection stops being accurate and for that reason, we stop all motion detection waiting for the user to step in the recognition area again.

Likewise, the gestures to manipulate objects were implemented trying to maximize the naturalness of the actions themselves. We are detecting two main gestures, closing hand and opening hand, along with a third one, moving a hand quickly back and forth. Based on these physical hand gestures, the following types of actions were implemented.

- A "take object" action where the user closes his/her palm over the object. The object will remain attached to the virtual hand as long as the user keeps his/her hand closed.
- A "drop object" action where the user opens his/her palm while holding an object.
- A "use object" action where the user moves his hand back and forth for a while over an object, for instance, when the user needs to use a disinfectant object to clean a toxic spill.
- An "open object" action (same user hand gesture as for "take object"). Object (e.g. refrigerator) opens when closed.
- A "close object" action (same user hand gesture as for "take object"). Object closes when open.

We note that in our system, hand gestures are contextual, i.e., the same physical gesture can refer to different actions in the 3D environment, depending on the context (the type or state of an object). For instance, the user may take an object by closing his/her hand while pointing at it; similarly, the user may use this gesture to open or close a door. The user may grab an object by maintaining a closed hand. The disinfectant (see list above) can be used by grabbing it and moving it above the area that has to be disinfected. Informal testing demonstrated that these methods are the most user-friendly for subjects operating inside the virtual environment through the exclusive use of their hands. As the majority of actions in the system can be carried out by closing, opening or moving a hand, users require only a little amount of time and effort to get used to this interface.

3.2.2. Mouse and Keyboard Interface

We also implemented a mouse and keyboard interface, to be able to compare it to the Kinect interface. We tried to reproduce as much of the Kinect interface as possible so that comparing both methods would be reliable. For this reason, all actions that the user can perform using Kinect can also be performed using mouse and keyboard. However, due to needing one hand to control the avatar's movement, the user can only control one of the avatar's hands. We considered this a possible disadvantage of this mode compared to the Kinect interface, and it will be discussed later in this work. Nonetheless, in the developed scenarios we did not designed situations in which the user needs both

hands for perform its actions (using both hands can enhance the user performance though). The controls in this interface mode are performed as follows:

- Pressing the arrow keys or "WASD" moves the avatar in the virtual world.
- Moving the mouse changes the position of the avatars hand. In this version of the interface the user controls only one hand as it would be impractical to control two hands with one mouse. As with the Kinect interface, if the hand reaches the limits of the screen, the view rotates in that direction.
- Left clicking the mouse is the equivalent to the close hand gesture. Therefore it can have the effect of taking, opening or closing an object, depending on the context.
- Right clicking the mouse is the equivalent to the open hand gesture. Therefore it has the effect of dropping an object if one is currently held.
- Moving the mouse back and forth when holding a disinfectant has the effect of using the disinfectant.

3.2.3. Low-Level Event Generation from User Actions

The gesture recognition and Low-level event generation processes are performed by the Interaction Processor module (see Fig. 1). It was implemented as an extension of the official Microsoft Kinect framework. Upon the recognition of a gesture, the Motion Preprocessing component determines the appropriate task-level event based on heuristics regarding the placement and intention of the user when he/she performs the detected gesture.

A Low-level event is an action that is indivisible from the virtual environment perspective and used to perform reasoning over the objects in the scenario, deciding its behavior and the event consequences. The Low-level event is then passed to the Common Sense Reasoner to analyze its effect on the virtual environment. In the case where a Low-level event is not generated from a recognized gesture (e.g. navigation gestures), the gesture is sent directly to the Objects' Physics Event Processing component as an animation procedure. Low-level events corresponding to the actions carried out by the user are formalized as combinations of gestures and objects in the virtual environment (that are targeted by the gestures). This approach supports a consistent definition of a set of user actions from primitive gestures and the classification of objects and their properties. In turn, the recognition of a low-level event serves the activation of physical behavior corresponding to the events' consequences, or modifications to the virtual environment state. Recognition of low-level events is based on predefined heuristics regarding:

- The type of gesture.
- The target object, i.e. object that will be manipulated.
- The surface object, i.e. object over which the target object is to be manipulated.
- The object already present in the gestured hand.
- The duration of the gesture: it can be a short duration (less than 1 second) or a long duration (2-3 seconds).

To determine the target and surface objects, we implemented a simple ray-casting technique in which a target or surface object is selected if it is close enough to the user. More advanced techniques for 3D object manipulation have been implemented e.g. in (Bowman and Hodges 1997). However, we

decided to use a simple method because (1) we wanted to emphasize the navigational aspect by letting users approach to the objects themselves, and (2) in-hand manipulation of objects was not a necessary feature in our environment. The user-generated Low-level events currently implemented in our system are:

- Take event: It occurs when the user closes his/her palm for a short time over an object ("take object" user action). The event is generated as "Take (object)".
- Open event: It occurs when the user closes his/her palm for a short time over an object that can be opened ("open object" user action). If this object is not opened yet, the event is generated as "Open (object)".
- Close event: Similar to the Open event, but the object needs to be opened this time ("close object" user action).
- Use event: It occurs when the user moves his/her hand for a long time over an object (the surface object), having another object in the gestured hand ("use object" user action). The event is generated as "Use (object, surface)".
- Drop event: It occurs when the user open his/her palm for a short time while having an object in the gestured hand ("drop object" user action). The event generated here is "Drop(object, surface)", with "surface" being the object over which the "objects" will be dropped.

Please note that the consequences of the events are processed by the Common Sense Reasoner module. For example, when a "take" event happens, the Common Sense reasoner will check if the object is "takeable", or if the hand is already carrying an object. If the gestured hand is carrying another object, the component determines if the carried object can be used to carry a second object. Also, if a "use" event takes place, the Reasoner decides if the object can be used.

Figure 7 shows an example of the user's gestures processing. We can see in the left side of the picture the case when a "take object" action has been detected, then the Motion Preprocessing component receives the tuple ‹PALM CLOSED, SHORT DURATION, OBJECT ID, null›, and it determines that a Take task-level event has taken place. In the case of a "use object" action (Figure 6, right), it receives the tuple ‹PALM CLOSED, LONG DURATION, OBJECT ID, SURFACE ID›, and determines that a Use low-level event has taken place (see Fig. 6-b). In the case of the keyboard and mouse interface, the processing remains the same, replacing the gesture recognition for direct commands. For example, the "PALM CLOSED" condition will be "LEFT CLICK", and "PALM OPEN" will be "RIGHT CLICK".

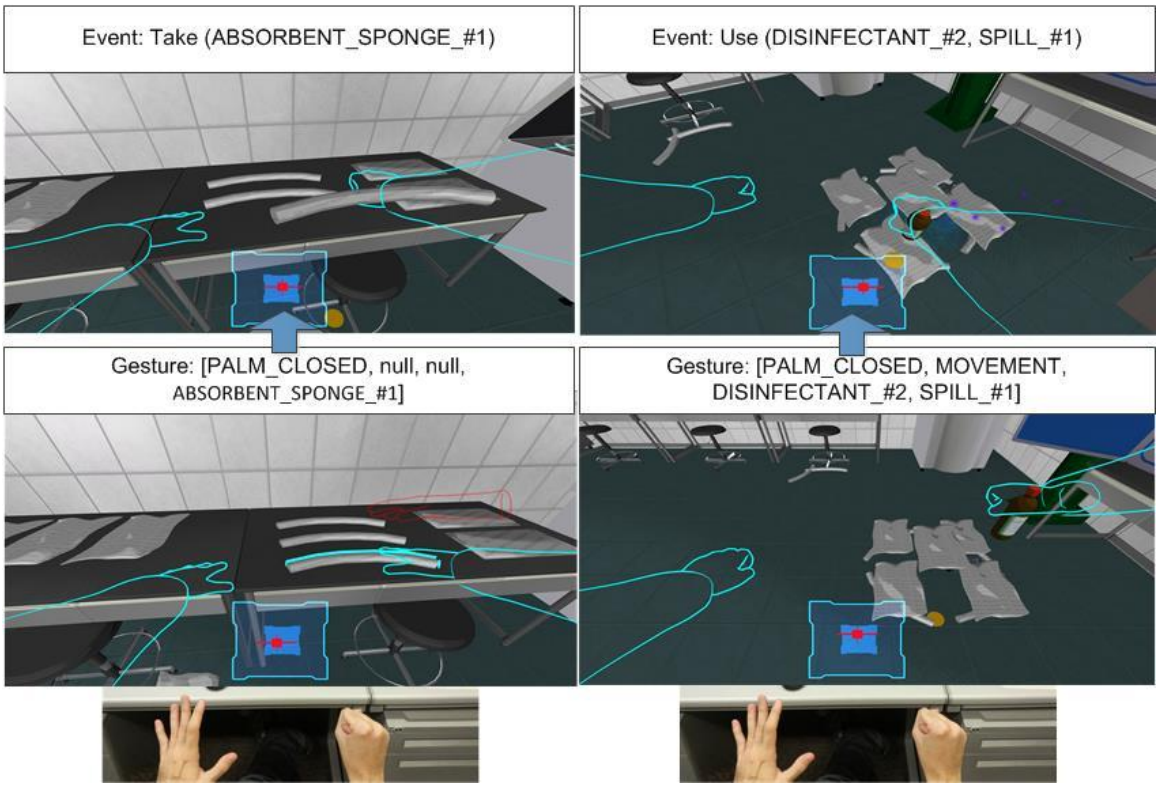


Figure 7: The "Take Object" and "Use Object" Action-Event Generation Sequence: (a) when the system detects the PALM CLOSED gesture over the ABSORBENT SPONGE object, the Take event is generated and, (b) when the system detects the PALM CLOSED gesture and is moving over the SPILL object using the DISINFECTANT object, the Use event is generated.

3.3. THE COMMON SENSE REASONER

When the Interaction Processor sends a Low-level event to the Common Sense Reasoner, this module determines the physical consequences that the aforementioned event generates inside the virtual environment and generates the Task-level event related to them. Task-level events are entities that have a semantic implication for recognizing and evaluating user's actions. For instance, if a user dropped a bottle on the floor, the Common Sense Reasoner would trigger a "break" Task-level event and generate the physical consequences of such an event based on the physical characteristics of the bottle and the substance it contains. The rationale for using a Common Sense approach is that the actual detailed simulation of some events is not always relevant (e.g. spill progression or actual shattering of a vial). On the contrary it is important to maintain an updated symbolic representation of the system (e.g. vials intact or broken, liquid inside or outside the vial).

This model is inspired in the qualitative physics model described in (Lugrin and Cavazza 2007). There the common sense reasoning relies on the physics engine's directives to perform the processing of physical consequences with semantic meaning called high-level events. In our case, the Common Sense Reasoner uses Low-level events as directives, and Task-level as semantic consequences. Since our system is focused on the achievement of realism from the perspective of training scenarios, the

implementation of a specialized physics engine was not necessary. However, our Common Sense Reasoner essentially implements a Qualitative Physics approach (Cavazza et al. 2004). This scheme supports better reasoning on the causal aspects and maintains an explicit representation of the action's consequences. As physical accuracy is of less importance in our application, the integration with Unity's native physics engine was not necessary.

In our Common Sense Reasoner, the Low-level events are processed through a series of cascading rules and generate Task-level ones. These rules are defined in the Common Sense Database (see Figure 8). As we can see in the figure, objects specified as parameters of those rules are matched to the parameters of the Low-level events and the semantic properties of those objects, which are also defined in the database. Once a Low-level event is completely processed, the Common Sense Reasoner generates a Task-level event (following the example in figure 8, a bottle breaking that generates a spill) and takes a series of actions:

- It processes the physical consequences of the Task-level event by interfacing to the primitives of the 3D environment engine, which supports the interactive visualization of the scenario state in real-time, in particular the creation or deletion of objects, or changes in their visual appearance.
- After the Task-level event has been triggered, the Common Sense Reasoner sends the event to the Task Model Reasoner located in the Task Recognition Engine for further process. This is done with the goal of evaluate the user's actions and decide which task is he/she performing, and how is doing it.
- It receives commands from the Task model Reasoner as answer to the previously sent Task-level event. These commands can be instruction for the user like hints or warnings or the generation of new events in the scenario.

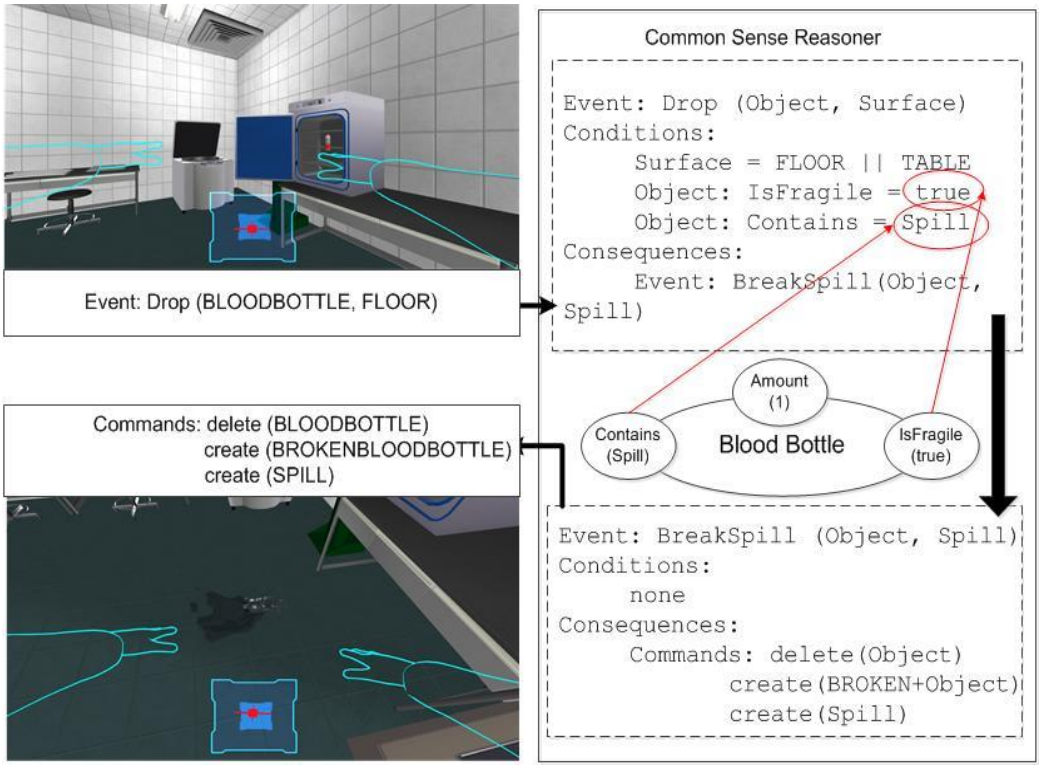


Figure 8: The Common Sense Reasoner: Detailed process for the "Drop (BLOODBOTTLE, FLOOR)" Low-level event. The Common Sense Reasoner match the event to a rule and generates a Break Task-level event and bypasses low-level physical simulation while updating the physical state of objects in the knowledge base. It interfaces to the Unity3D engine so as to update the physical appearance of objects following a change in state.

3.4. THE TASK MODEL REASONER

As we described in the previous section, the Common Sense reasoner sends Task-level events to the Task Recognition Engine for further processing.

The main purpose of the Task Recognition Engine is to provide task-related assistance to users during the simulation. In our context, students have been studying security procedures by reading a textbook and they are now prepared to rehearse them in the virtual laboratory. Hence the purpose of the system is to provide assistance to students where necessary by monitoring their actions. For that reason, we created a knowledge-based representation of biohazard training procedures called 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 from user actions and real-time assistance and guidance in case of incorrect execution, containing information about the possible mistakes that can be executed and how to correct them providing real-time feedback to the user. Furthermore, our approach is compatible with post procedure debriefing, by tracing the various errors and aborting the simulation when the situation cannot be corrected.

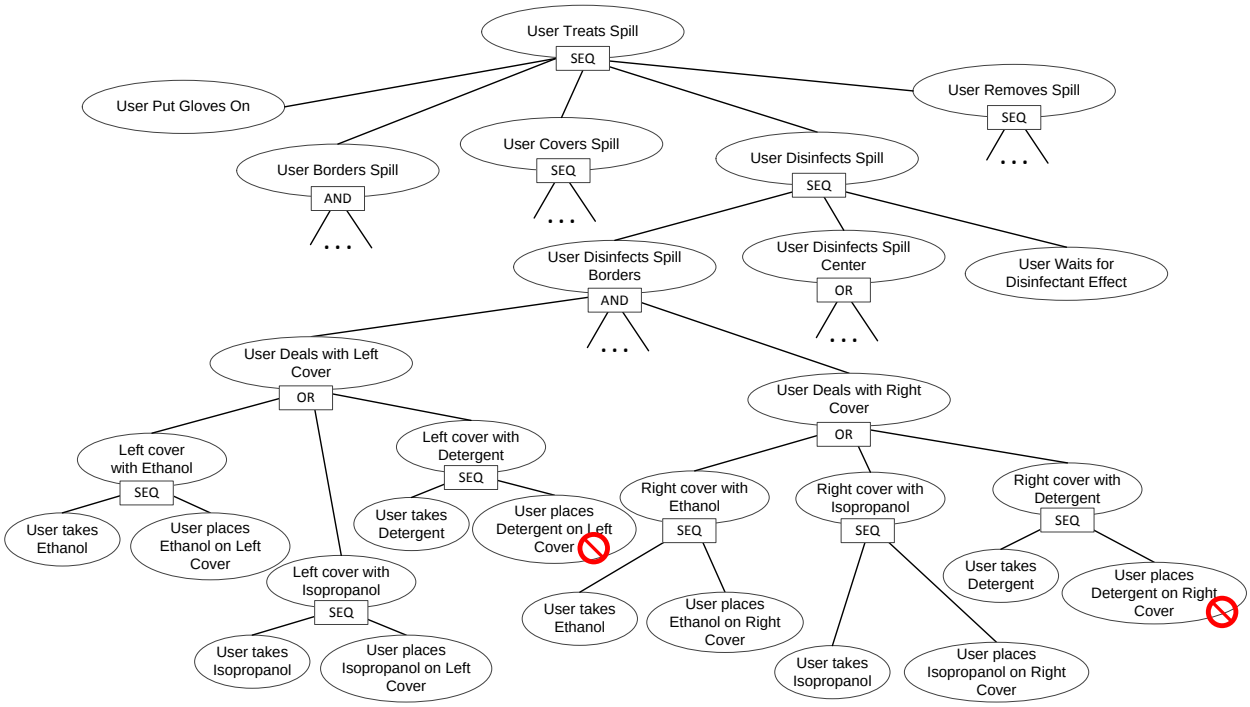


Figure 9: Part of the Task Tree modeling the spill clearing protocol (the complete tree contains more than 250 nodes). The goal of the user is represented in the root node. Two nodes are marked as incorrect and if they are instantiated, the module will warn the user.

A Task Tree is a hierarchical model which has in the top of the tree a root node representing the whole task to perform and the child nodes being the subtasks in which can be decomposed, until the leaf nodes, which correspond with Task-Level Events in the virtual environment that are performed by the user in the virtual environment. A diagram of part of the task tree used form the spill accident is shown in the Figure 9. 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 grouped with each other in three different ways:

- AND nodes: if a node is labeled with this property, all its immediate descendant nodes must be completed in any order.
- OR nodes: if a node has this property, it will be validates when any of its immediate descendant are validates, excepting nodes identified as mistakes (we will elaborate more about this later). The intuitive meaning is that this group represents different ways to do a task.
- SEQ nodes: when a node has this property all its subtasks must be completed in order: it represents a sequence of actions. The sequences do not require having all the subtasks performed one immediately after the other, allowing apart and sequential actions with other different subtasks in between.

This knowledge representation is inspired from a hierarchical planning formalism known as Hierarchical Task Networks (HTN) (Erol et al. 1994). Actually, the task tree is similar to an explicit

HTN in which the main task has been decomposed a priori and entirely, down to the level of elementary actions, rather than being dynamically refined using decomposition methods (Nau et al. 2004). Thus, instead of representing security protocols as a collection of refinement methods, we use an explicit task tree which can be represented visually, facilitating knowledge elicitation. Note that we do not use this task tree for plan generation and, in fact, we do not use any planning technique. We use a hierarchical task model (1) to intuitively represent training procedures as multi-step decomposable processes, and (2) to perform task recognition, which is achieved by traversing the tree each time the user performs an action in the virtual environment. In general, the idea of using planning-based formalisms to represent procedural knowledge is not unusual even in systems that do not use planning algorithms (Bradbrook et al. 2005; Shahar et al. 1998).

There are several benefits in using an approach inspired by HTNs to model complex procedures. First, HTNs have been shown as an effective way to encode domain knowledge and to restrict the order in which actions can be combined (Nau et al. 1998; Wilkins and desJardins 2001). Second, HTNs are intuitive enough for experts and allow them to work at different levels of abstraction, which decreases the effort required to model complex activities (Currie and Tate 1991; Muñoz-Avila et al. 2001). Along the same line, HTNs promote re-usability of abstract tasks among different protocols since subtrees can be shared in different branches of the tree.

The task trees in Bio-safety Lab were created during extended sessions with domain experts from the National Institute of Infectious Diseases in Tokyo who collaborated strongly in our project (see Appendix A for a manual defining the task trees and the complete task tree used). Note that some of the represented procedures are quite complex: the complete task tree that describes the protocol to treat the spill of a hazardous sub stance contains more than 250 nodes and requires at least 40 task-level user actions to be completed. In order to represent correct steps and mistakes, each one of the leaf nodes is marked either as a correct step in the task either as an error. The OR groups contain the possible documented errors. Error nodes are still part of the same group that their sibling nodes, usually an OR group, due to be just another option more to perform the subtask (albeit a mistaken one). The rationale behind this is that it's a physically plausible way to do a step of the task, even if it's a wrong one, and allows to be recognized as an error while performing that step. Whenever a correct step is matched with a leaf node, is instantiated in the task tree and the tree is updated in a bottom-up fashion, instantiating parent nodes if necessary and so on.

If the received event is considered a user's mistake, the engine won't instantiate it and sends back assistance messages in real-time to the virtual environment for displaying 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 should not be executed yet. In the first case the system will explain to the user why he should not do that action, and in the second case the system will display which is the previous action the user has to perform before the one he tried to do. Finally, in case of the user not knowing what to do next, the Task Recognition engine also is able to give to the user directions of what should be his next actions. As an additional feature, the system allows to collect information about the user behavior and how he reacted to the accidents and mistakes, being a useful source of later analysis in order to identify the harder parts of the training or profiling the user.

In the three following subsections the process of recognizing correct actions, detecting mistakes and giving real-time feedback will be discussed in detail.

3.4.1. Recognizing user's Actions

To monitor user activity, we need to identify the actions the user performs as the student progresses through the biohazard training protocol. In this subsection we describe how to identify actions that are correct according to the task model, and in the following subsection we will explain how to detect and help the user when the student makes a mistake.

The Task Model Reasoner receives messages with task-level events from the Common Sense Reasoner. These events in the message represent the actions that the user has performed in the virtual environment, and are used in order to decide what kind of task is performing the user and if he is doing it correctly. The Task Model Reasoner sends back new events indicating the tasks that have been recognized. The recognition is performed by matching the received events in the task trees. When one of the user task-level events matches the event contained in one leaf node of the tree, the reasoner instantiates the corresponding task and, probably, other higher-level tasks depending upon it. In this way, the task tree is instantiated from the leaves to the root as the user advances in the simulation. The matching between task-level events and leave nodes of the tree is quite straightforward. Task-level events describe specific actions performed in the virtual environment and hence they cannot contain variables. For example, the task-level event Take (ethanol #1) is received when the user takes a bottle of ethanol represented with the symbolic constant ethanol #1. Leaf nodes of the task tree, on the other hand, describe actions using typed variables. For example, the leaf node User takes ethanol contains the action Take (?x1 - Ethanol) that accepts any object of type Ethanol and therefore matches the previous user action.

This bottom-up instantiation of the task model from user actions supports the analysis of the user behavior from the perspective of the task to be learned, supports a unified mechanism to assess the user, and provides real-time feedback and assistance during the execution of the most complex procedures. For example, the tree in Figure 9 shows part of the training protocol that must be followed for the treatment of a toxic spill. In this case, the original task User Treats Spill is decomposed into five subtasks that must be performed in order: (i) Border the Spill, (ii) Cover the Spill, (iii) Disinfect the Spill, (iv)Merge the waste materials and (v) Dispose of the waste.

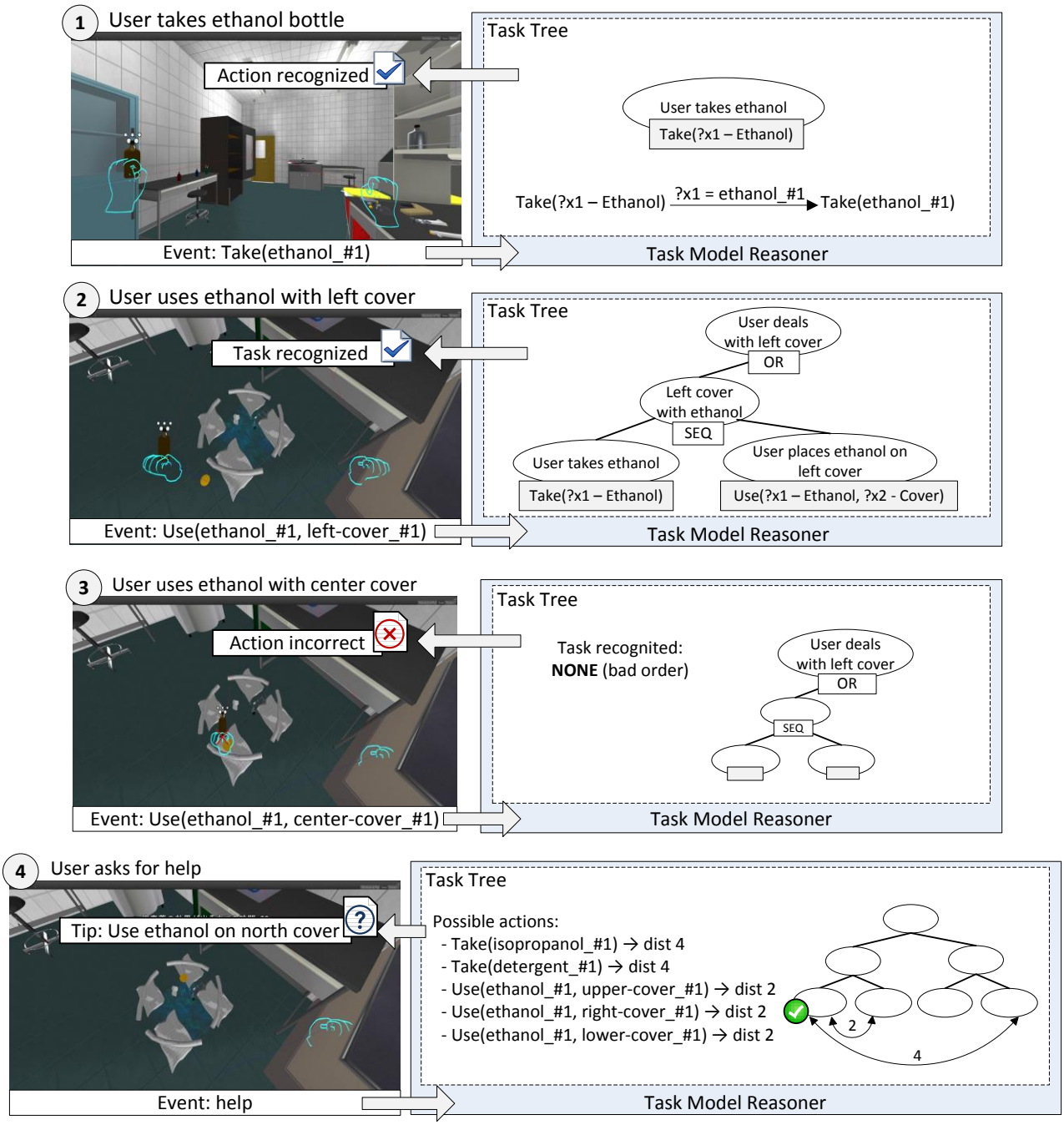


Figure 10: An example of task instantiation as the user progresses in the procedure of the spill cleaning. In Step 1 the user performs an action that is recognized as correct because it matches the task contained in a leaf node. In Step 2 the user performs another correct action that triggers the instantiation of several tasks at different levels in the tree. In Step 3, the user performs an (incorrect) action that should not be executed yet according to the procedure. In Step 4, the user asks for help; then, the reasoner computes the possible next actions, and selects which one to show using a heuristic approach based on distances in the task tree.

Fig. 10 shows an example of the interaction with the system and the task instantiation process that the reasoner performs. At this stage of the procedure, the user has to disinfect the spill that was previously covered and choose among different chemicals depending on the nature of the spill. In Step 1, the user chooses to take a bottle of ethanol (represented as the formal object ethanol #1 of type Ethanol), an appropriate disinfectant, so that the task reasoner receives a Take (ethanol #1) event. The reasoner traverses the current task tree looking for leaf nodes accessible in the current state which contains a compatible task. In this case, the reasoner finds a leaf node with the task Take (?x1 - Ethanol) that matches the user action. The reasoner instantiates the task with the corresponding variable binding and sends back an event indicating that the action was recognized. Next, in Step 2, the user uses the same bottle with ethanol to disinfect the left side of the spill. In this case, three different task nodes are instantiated as a result of the action, because this action completes two other higher tasks in the task tree. Note that one action might trigger the recognition of several tasks at different levels in the tree. We will return to the rest of the steps in Figure 10 in the next subsections, where we describe in detail how the system processes the user's actions that are recognized as mistakes and how it assists the user.

3.4.2. User mistakes and dynamic feedback

Deciding when and how to help the user is a very complex problem that is beyond the scope of this paper. However, the logical structure of the task representation provides a basic mechanism to reason on the user progression that can be used to detect incorrect actions and provide useful hints. In this sense, our system can be called a training or rehearsal system. We can detect two different types of incorrect actions:

- Actions that are explicitly represented in the task model as incorrect choices as decided by the domain expert.
- Actions that are part of the correct procedure but should not be executed yet due to be a part in a sequence of steps.

The first type corresponds to those errors that have been anticipated by the experts: for example, when the user utilizes a disinfectant that is not suitable for the kind of spill being treated. These errors are instances of task events received from the Common Sense Reasoner, e.g., using a wrong object like dropping an absorbent pad over the spill, or using the wrong instance of the correct object like using a wrong disinfectant when disinfecting the spill. The task nodes can contain specific messages to explain why the user action is not adequate. For example, if the user tries to use an inadequate disinfectant, the system can provide a warning and explain why the student should not use that chemical in this situation. The second type of mistakes comprises those actions that should not be executed at present, because they are part of a SEQ node with some previous subtask that is not achieved yet. For example, according to the Spill Disinfection procedure, the user should not try to disinfect the center of the spill until the borders have already been disinfected.

Let us take into consideration Step 3 in Figure 10, where the user decides to use ethanol on the center of the spill immediately after disinfecting the left border. In this case, the reasoner traverses the task tree, detecting that the action does not match any leaf node compatible with the current state, but a leaf node in a branch that should not be executed yet. In this way, the system may notify the user that he/she should complete three tasks first: User deals with upper cover, User deals with right cover, and

User deals with lower cover. Note that the system is able to use high-level tasks to describe what parts of the training procedure the user should complete without describing the specific actions the user has to do. In other words, the hierarchical task model provides the ability to interact with the user by using abstract concepts.

Finally, it may happen that the user does not know how to proceed next to deal with the remaining three border covers. In Figure 10, Step 4 shows what happens when the user asks for help. In this case, the task reasoner traverses the task tree looking for all the actions that can be executed next. Usually, there will be several different possible actions because the AND and OR nodes allow different execution paths. In our example, the user could either use the bottle with ethanol that he/she is holding in his/her hand with any of the three remaining borders, or he/she could take another disinfectant. Although all these actions are correct, some of them are more intuitive than others. In this case, the user is holding a bottle with ethanol that has already been used to disinfect a border, so we can presume he/she will want to use it again instead of taking another disinfectant.

We take advantage of the hierarchical structure of the task model to prioritize those possible actions in the tree that are closer to a task that has already been completed. The intuition beneath this heuristic is dual: (1) the task hierarchy groups tasks that are part of a whole, i.e., the task/subtask relation is a strong semantic constraint, and (2) if the user has performed a subtask, he/she will probably try to complete the sibling subtasks in order to complete the higher level task. In the example, when the system computes the distances of the five leaf nodes that contain the possible actions closer to the completed task (User takes ethanol), it decides to prioritize the User places ethanol on upper/right/lower cover tasks because they are siblings (distance 2 in the tree). So the system will advise the user to perform one of them.

As explained above, the hierarchical task model that we use to represent the biohazard training procedures let us to detect two types of errors in real-time: (1) errors that are labeled as incorrect (by the domain expert) and (2) errors that result from the execution of actions at the wrong time. In both cases the system is able to instruct the student showing descriptive help messages, but using different approaches.

- First case: the error and its associated help message are part of the domain model, and therefore they have been anticipated by experts. The system displays a message explaining why the user action is a mistake.
- Second case: the system displays a message compelling the user to complete another task before executing the current action.

It is important to remark that, although the help message is generic, the task to complete usually corresponds to an internal node of the tree, i.e., an abstract goal. In this way the system is able to describe abstract goals the user must achieve without indicating the specific actions to perform. Regarding the first type of errors, the task model is able to detect when the user is placed in the wrong position, for example between the spill and the air vent; when a wrong product is used to delimit or cover the spill, for example absorbent pads; when the user chooses a disinfectant not suitable for the type of substance being treated; or when a product is handled with the wrong tool, for example taking the waste with the gloves instead of the tweezers; among others incorrect actions. All of these errors

are nodes in the task tree that are identified with a task-level event and each one is associated with a specific help message. In the following, we describe some examples of them followed by the type of assistance message the system produces:

- When the student attempts to cover the center of the spill before covering all the borders the message is: "You should cover the spill borders before attempting to cover the spill center".
- When the student covers the left border and then attempts to disinfect it without covering the rest of the spill first the message is: "You should cover the spill before attempting to disinfect the spill".
- When the student attempts to dispose the covers and then the borders in two different steps the message is: "You should merge the waste before attempting to dispose the waste".

In summary, the hierarchical-task model chosen to represent bio-safety lab protocols provides the following benefits:

- It is an intuitive and explicit representation that can be used and revised by domain experts.
- It enables real-time detection of two types of mistakes: (1) mistakes that have been anticipated by the experts and included explicitly in the model and (2) mistakes that arise when the student forgets some step of the protocol.
- It supports interaction with the student using abstract concepts represented by the internal nodes of the network. Thus, the system can inform the student about the next to be completed high-level task without referring to the contained basic actions.
- It can anticipate the most likely next action the student will perform using a heuristic based on the hierarchical structure of the domain.

3.5. THE 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 of the user's actions or proactive decisions from the module. The Narrative Manager manages the balance of difficulty in the training sessions and the level of user's interest in the training. Our hypothesis is that we can create more events for the user to solve while maintaining high levels of user's interest. Like the Task Model Reasoner, the Narrative Manager also uses task trees as knowledge database to decide what kind of events will trigger. These task trees represent tasks for the system with the goal of hinder or helping the 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 we described in the previous section. System nodes represent the events the Narrative Manager will trigger. The system nodes also contain an additional property called "Closure" indicating if that event can only be triggered before another concrete event happens. The Task Tree containing the events related to the spill accident is shown in the Figure 11.

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. These parameters are inspired by the ones of (Ware, S. and Young, R., 2011) described as the dimensions of

conflict. These dimensions are four: balance, that measures the relative likelihood of each side in the conflict to succeed; directness, which it is a measure of the closeness of the participants including spatial, emotional and interpersonal distance; intensity, which is the difference between of high will be a participant potential success and how low the potential failure; and lastly resolution which measures the difference in a participant situation after a conflict event happens.

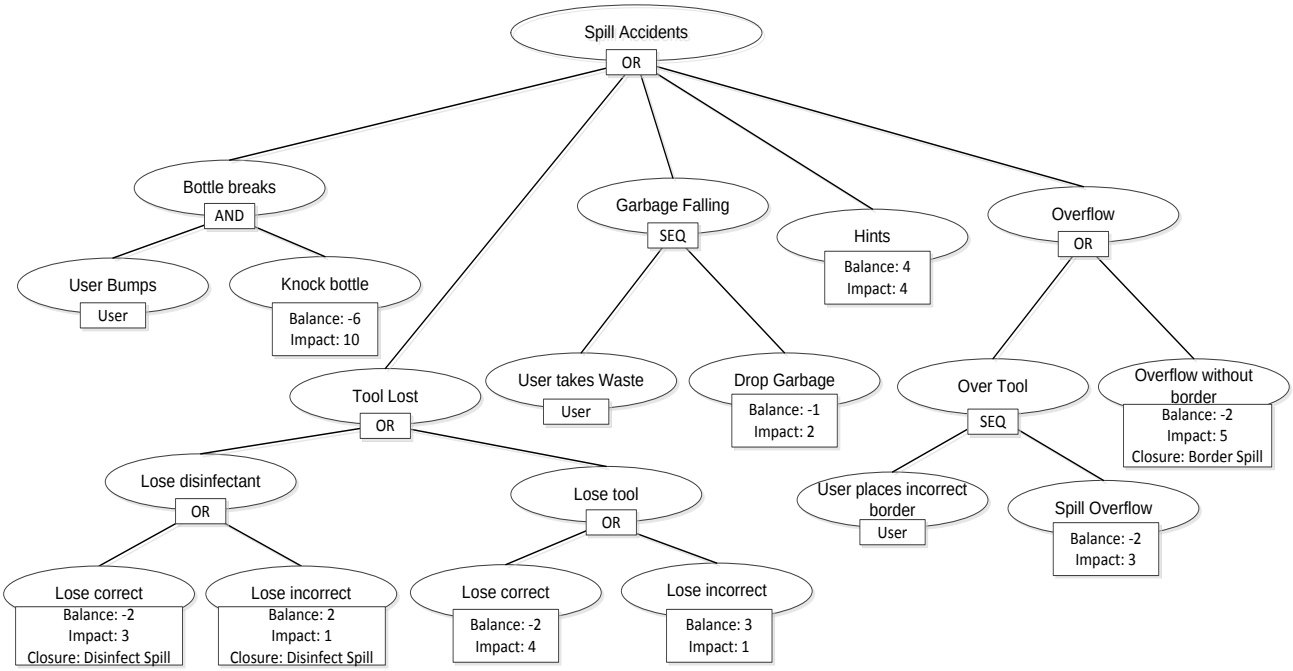


Figure 11: System Task Tree for the spill accident. In each node's box it can be seen the information relevant for it: the parent nodes show the type of the group (AND, OR or SEQ) and the leaf nodes inform if its type is User or not. Non-zero parameter values are shown in the leaf nodes: Balance, Impact and Closure.

In their work they also recognize that there is no perfect way to give value to the parameters and how to use it and give directions of how quantify them in a formal way, but always with a component of decision from a domain expert. Thus, we decided to adapt them because they are designed for creating conflict between characters and don't fit in a training domain which only has one participant and his failure is not a desired outcome. For example, we don't need to store the sentimental closeness of the user with the scenario neither know how bad will be if the user fails. We use 4 parameters with our own nomenclature: Balance, Global Balance, Impact and Intensity:

- Balance: It's an integer number that describes how good or bad for the user is an event (positive if it makes the task easier for the user or its beneficial for him, and negative if it's hard or hinders his performance). Each event contains its own balance value stored in the Task Tree, and the value is given by the domain expert who 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 maintains also 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 to 0. The rationale behind this attenuating effect is that receiving some hints or difficulties can affect the user in the short term but as the 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: It is a positive value that represents the degree of dramatic impact that an event has for the user when is 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 is very startling for him, having an Impact of 5.
- Intensity: It is a global value maintained by the system that describes how dramatic is the current situation for the user. 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 of its Balance value's 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 big Impact events for some time. The Narrative Manager also increases the Intensity value slowly with time, being able to trigger events even if the user is not doing anything if given enough time.

As we can see, the values of the Balance, 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 for calculating the value of the related dynamic parameters (Global Balance and Intensity, that not only depend of the static value of the two static parameters but also the timing and the kind of the user events).

3.5.1. Selecting the most appropriate event for the user

Attending to the parameters we described, the Narrative Manager will select the next event to trigger and model dynamically 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 in two cases:

- Whenever a high-level event is received: when the Common Sense Reasoner sends a Task-level Event to the Task model Reasoner, this event will be registered into the Task tree (or it won't, if it's a mistake or it doesn't fit in the task), and after this, the Reasoner passes the event to the Narrative Manager for further processing.
- After a time interval passes, specified by a timeout (in our system the timer was set to 15 seconds), the Narrative Manager is called with an empty event. The rationale behind periodically calling the Narrative Manager even if the user is not doing actions is that in a narrative when nothing happens is also meaningful, so the system sends an empty event in order to build the intensity and decide if it trigger an event.

Each time the Narrative Manager is called, it selects the event from the Task Trees which maximizes Impact, with the current Intensity as upper bound and with the Balance most similar to the

current Global Balance but with opposite sign. We use Intensity as an upper bound because it represents the maximum dramatic load we allow for the user at a time, thus limiting the events that can happen. This means that a big Impact level like creating an extra spill cannot happen unless the user has done many actions before and the situation for him can be considered easy. Also, the system tries to move the Global Balance value towards zero, so the desired event should have a Balance similar to the current Global Balance, but with opposite sign. Using both parameters allows maintaining a balanced difficulty and at the same time creating events that impacts the user in a dramatic way. It is possible that no event meets the conditions above because the current Intensity is too low. In that case, no event will be triggered, meaning this that the Intensity is still too low for the system to act.

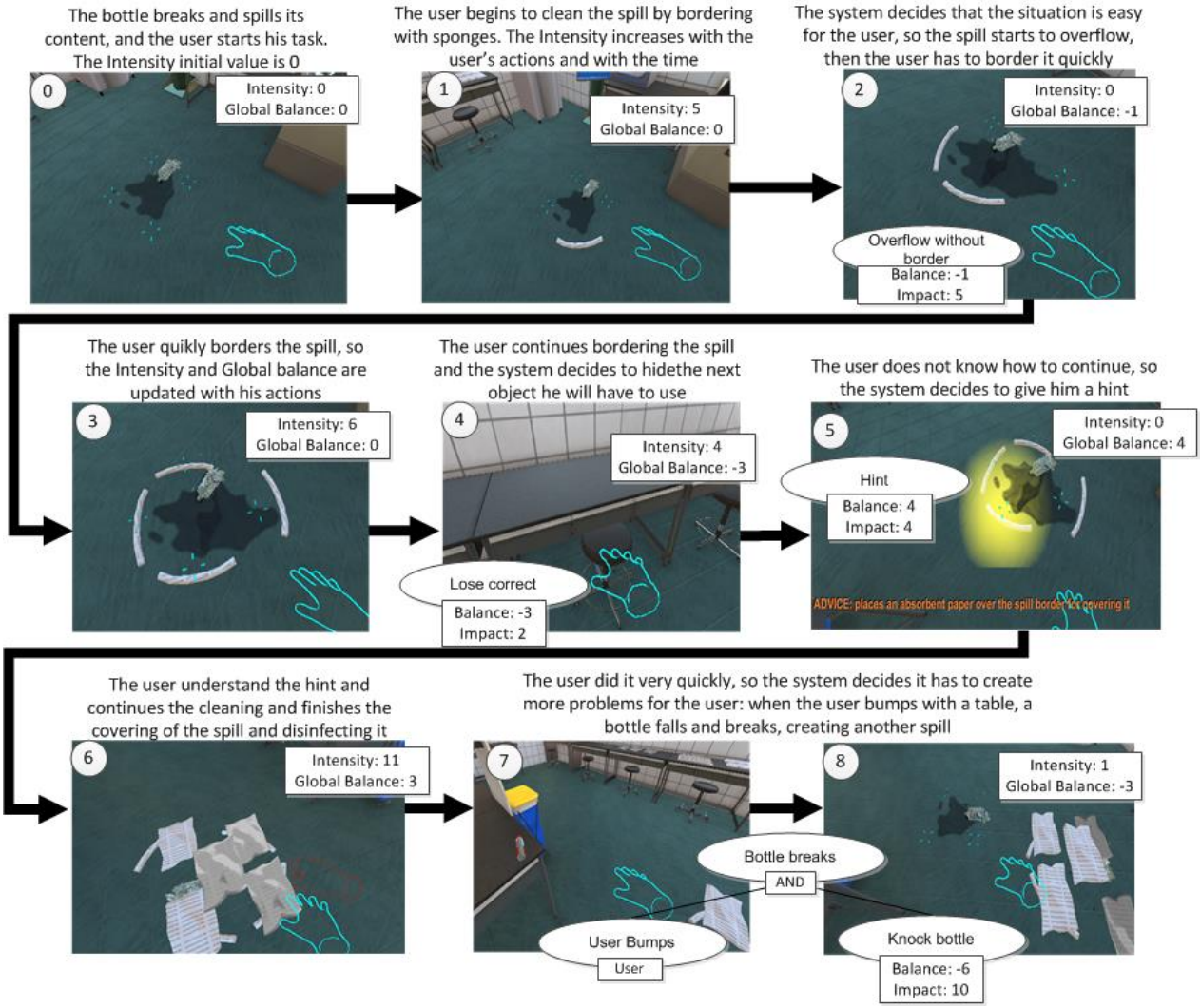


Figure 12: Scenario Example showing the Narrative Manager triggering different events as the user is performing his goal task. The Intensity and the Global Balance are updated as the user acts, the time passes, or scenario events are triggered. The Narrative Manager selects the action with the greatest Impact using the Intensity as upper bound and tries to choose the one with a Balance near to the Global balance but with opposite sign in order to keep the user interested and challenged.

Figure 12 shows an example how the Narrative manager selects the events in a sample session of Bio-Safety Lab. In the example, the user starts bordering the spill (Number 1 in the figure 12) and that actions increase the Intensity as they are registered in the Task Model. Then the narrative Manager searches for a suitable event with an Intensity lower or equal than 5, being the selected one making the spill overflows (number 2). Also, the Manager updates the Intensity and Global Balance values: subtracts the Impact of the triggered event because it had negative Balance, and adds the Balance to the Global Balance. Next the user continues bordering the spill and the time passes, so the system updates the Intensity and Global Balance again. Again, in number 4 we can see that the system selected another events that hinders the user's task, this time the Impact and Balance values were not very close to the desired ones (values of 6 and 0, respectively), probably due to not having other better events available at that moment. Number 5 shows that this time the system selected an event for helping the user with a visual hint, increasing the Intensity and the Global Balance to positive values. This values increase even more (number 6) with the progressive completion of the task reaching high values if the user is enough fast. This makes the Narrative Manager to trigger one big problem to the user (see number 7 and 8): when the user bumps with a table (it is very easy to bump with the 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 maximizes these values, and then selecting from these candidates one event randomly. The selection of the nodes follows the rules we stated for sibling nodes in the previous section, so for example, the Narrative Manager won't execute a node in an OR branch if a sibling node has been executed already, and respect the SEQ branches by executing nodes in order. Also it takes in 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 again the Utility and the global balance and sends back the order of executing the event to the virtual environment.

Finally, each time an event happens (whether is an event performed by the user or triggered by the Narrative Manager) the Narrative Manager keeps updated the global value of the Balance and the Intensity. Intensity is calculated by adding or subtracting the Impact value of the event that just happened, depending of its Balance value's sign (positive for adding, and negative for subtracting). The current Global Balance is calculated by adding to it the Balance value of the event. If the Narrative Manager decides that no event has to be triggered and some time (defined by a timer) happened from the previous event, it will simply increase the Intensity value and the Global Balance will slowly converge to zero. We increase the Intensity because we consider that giving time to the user is good for him, so after some time an event should trigger. Also, the Balance changes with the time because if the previous event was bad for the user (negative balance), giving him time can be considered as something positive for him.

For example, if a bottle breaks, the user could use some time to recall the steps of the protocol, or for looking for the materials for cleaning it. We decided also that if the previous event was positive, its effect will diminish with the time, so the balance should return to zero as the situation becomes no more beneficial to the user; for example, if the user received some hint (a positive event that increase the balance) about what material use for cleaning the spill, the effect of the hint only last until the user

finds and use the object, If after that the user does not know what to do, and will spent some time thinking, decreasing the positive balance.

3.5.2. Drama manager customization: authoring an Intensity curve

Aside of the user experience enhancing capabilities of the Narrative Manager, using this system has a secondary advantage, as we hinted in the related works section. Having an engine controlling the details of the training session (in our case, the events that happen in the virtual environment), makes the task of authoring the scenario easier; the knowledge experts simply has to define the possible events in a scenario and specify their narrative parameters of Impact and Balance. However, this feature has a downside: it is difficult to control or predict how the session will develop. It seems to be counterintuitive to want to control the events when we just have stated that not having to do so is an advantage, but is desirable to give to the domain experts some degree of control over the details of the training, especially if they want to reinforce certain behavior in the users or to teach them some concrete practice.

In Bio-Safety Lab, the Narrative Manager manages different parameters in real time to select the event to trigger, and that makes it very difficult to predict when the domain expert is defining the scenario events and their Impact and Balance. Making simulations of the system can give a general idea of how the session will be, and calculating how the narrative parameters will vary and giving careful values to the events' parameters can increment the probability that some concrete events will happen. However the session always will try to maximize the user engagement by keeping him challenged continuously. This can be sometimes contrary to the desires of the session designer: maybe is better for the user to put him in a more forgiving session because he previously finished a more difficult one and he has to reinforce some procedures. Other example would be when the experiment designer wants to limit some difficult events for some concrete moments: by giving a high impact value the designer only will know that the system will trigger that event later, but that is the only control he will have.

In order to answer this need, we have implemented an additional feature in Bio-Safety Lab: the Intensity curve authoring tool. This application allows drawing a curve that will be used as a reference for the Narrative manager when it selects the events in the session. A screen capture of the authoring tool can be seen in Figure 13. The graphic interface consists in a screen in which the scenario designer can model a curve that represents the Intensity over the time in the training session. The curve contains five dots called milestones that can be dragged up and down with the mouse. While moving a milestone, its numerical value will appear in the textbox related to that milestone. This value represents a percentage of how near can be the intensity in that relative moment from the maximum possible Impact from the scenario's events. As we see, there is no time specified in the graph. The reason is because each session will likely have a different duration, and the only significant variable that we can detect is the events themselves. Using as an example a session controlled by the curve depicted in Figure 13, when the system selects an event with an Impact near to the half of the maximum Impact event available, the system will suppose that has passed the second milestone (with a value of 51%), then the system will try to decrease the intensity to a third of that maximum value, reaching the third milestone (value of 31%) when doing that. In the example we decided using only five milestones, but we can add more dynamically. However, we think five milestone is enough, because this implies that we model the tendency of the Intensity of each fifth of the session, and we

consider that an enough degree of control for the scenario designer. By adding more milestones the designer would have too much control in the scenario, effectively scripting the whole session and not allowing the Narrative Manager to control freely the Intensity curve.

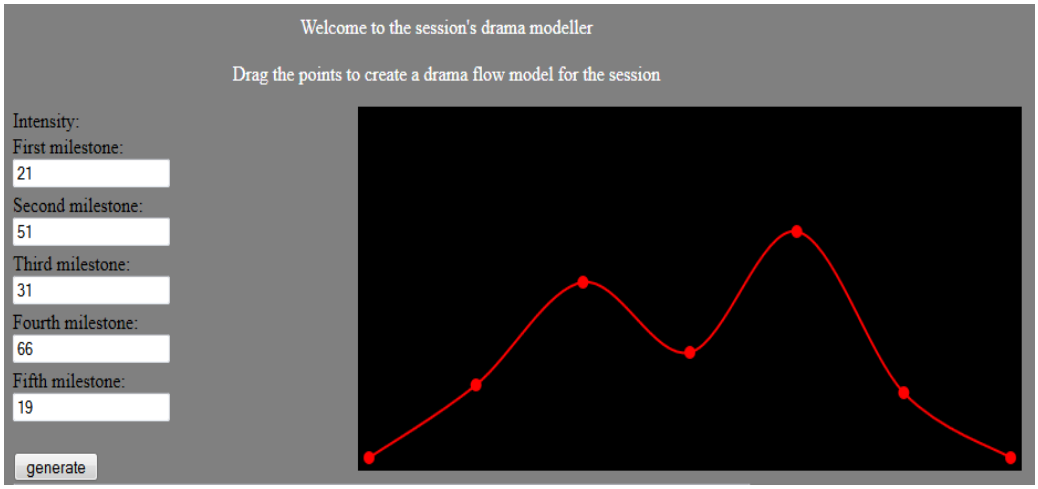


Figure 13: The Intensity Curve modeler tool for Bio-safety Lab. Moving the red dots in the curve the scenario designer can model how the Narrative Manager will behave in the sessions: the module will try to model over the time an Intensity curve similar to the depicted one.

The Narrative Manager can work with or without using a predefined Intensity curve. If the scenario designer does not create one, the Narrative Manager will proceed as we described in the previous section, but when using one as a model, there are two changes in the process of event management: when the Intensity value is increased over the time and when a scenario event is selected by the Narrative Manager. First, the Narrative Manager will use the next milestone value as a modifier for increasing the Intensity gradually. For example, if the milestone has a value of 100 (the maximum possible), the Intensity will increase with the time a 110% more than the usual value, and with a value of 0, the Intensity at only a 10% of its rate, and finally if the next milestone has a value lesser than the current Intensity, it won't increase with the time until its value reach the milestone. The rationale behind this variation in the rate of increasing the intensity is done because we want to increase it faster if the slope of the curve is very pronounced. There are other methods for varying the increment over time of the Intensity, but we decided use this one because its simplicity: a milestone in the top of the graph doubles the rate, and one in the bottom almost negates it, while a milestone in the middle, does not modify it.

Then, when selecting the next event to trigger we can observe the second change in the processing: as we described, a milestone's value is mapped to the maximum Impact of the available events in the scenario and the system uses it as a threshold. If the next milestone has a value bigger than the current Intensity, the Intensity will increase until an event with this value is triggered. Until that instant, other events can be triggered by the system, but if those events would reduce the Intensity value (due to having a Balance with negative sign), the Narrative Manager won't decrease it. When the triggered event has an Impact greater than the milestone value, the system will start using the next milestone's value (if there is no next milestone, it simply doesn't change). In the case of a milestone

that is lesser than the current Intensity, then it means that the curve as a descending slope, so the system will not increase the intensity over time until an event with Impact greater than the milestone's value is triggered. The events triggered until then can only decrease the Intensity, and in case it is an event that should increase it, the system won't do it.

With this two differences in the event management, the Narrative Manager can use the Narrative curve to create a similar Intensity curve for the session. The resultant curve won't be the same; for example, it is possible that even if the model curve has a descending slope the Narrative Manager decides to create events that increase the Intensity: the parameter value may not increase but the event is triggered anyway, so the "narrative intensity" for the user will increase nonetheless. However, the session will behave using the guidelines of the predefined curve: in the previous example, the intensity will end with a value lesser than the milestone, having the desired descending slope. This effect is intended: it is not our intention to give orders to the Narrative Manager of how has to act exactly, but only give it a general direction of how we want the training session. This way, each session will be still different even if we use the same Intensity curve to define it.

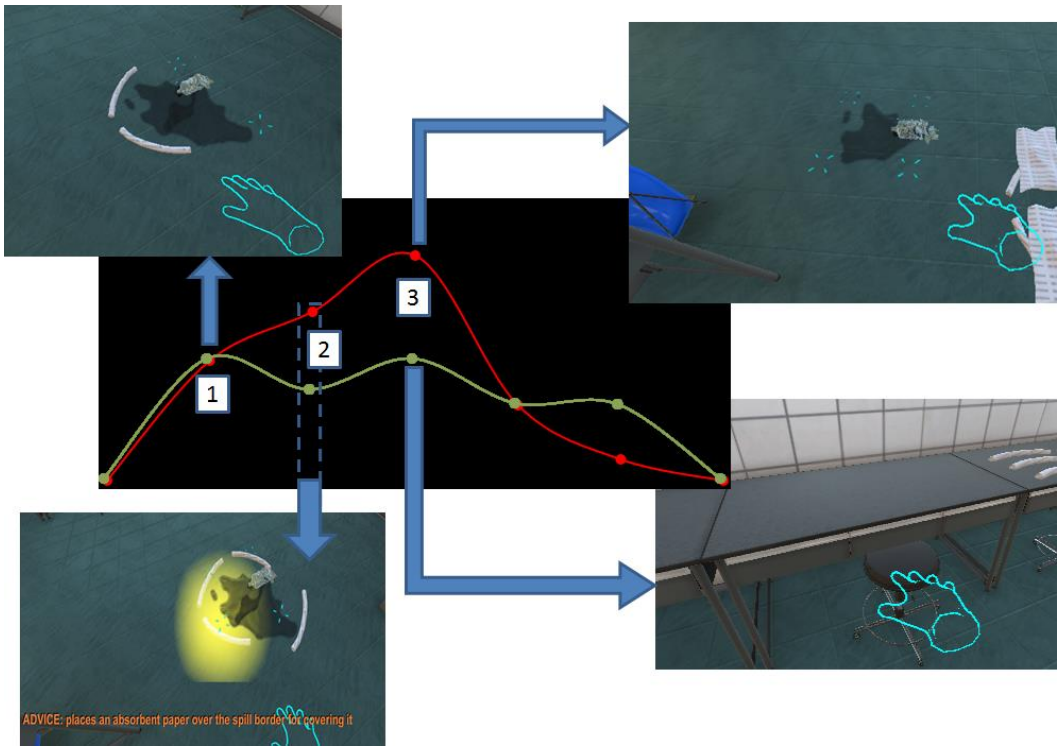


Figure 14: An example of the different behaviors of a scenario from using two different Intensity curves (red and green). In the first segment of the curves, the scenario would generate (roughly) the same events for both of the sessions, but then the curves separate (1). In the second milestone (2), the Narrative Manager decides to give a hint to the user allowing the red curve to rise in both sessions, in order to increase the intensity, however as the next milestone in both curves has a very different height, the instant when reaching it or the events that will happen in both sessions will be different. Finally, in the third milestone (3), the red curve allows for high impact events to happen, triggering a new spill. On the other hand, the green curve doesn't allow that and only generate events with medium impact, like hiding necessary tools from the user.

Figure 14 contains an example of how two different curves would affect the curse of the events in one hypothetical training scenario creating two different sessions as a result (given by the red and green curves; we will refer to the two different generated sessions as red and green respectively). Given that the user acts exactly the same way in both sessions, the events until the first milestone (Number 1 in the figure) of the curves would be roughly the same, raising the Intensity of the session until intermediate values (represented by triggering a spill overflow, pictured in the image). Then the two curves separate and the red one starts to rise whilst the green one decreases. In the red session this represent that the user likely will receive more high impact events, allowing a fast increase in the Intensity. In the green one, the Narrative Manager will trigger only some low impact events, lowering the Intensity. After the session reaches the second milestone (Number 2), both curves rises, so the Narrative Manager decides to give a hint to the user in both sessions, but given that the next milestone in the green curve is lower, probably it will be reached sooner than in the red session. In that one the Intensity will increase over the time at a faster rate and also will trigger more events in order to reach the higher milestone. The effect of the different Intensity values being bounded by the curve can be seen clearly in the third milestone of the curves (Number 3), where the red one has a high value and allows the system to trigger a new spill (an event with very high impact). On the other hand, the green curve will trigger a more moderate event, like hiding one of the spill cleaning tools. Also is interesting to note again that the time is not specified in the curve so the instant when reaching a milestone in two different curves would be only dependent of the Intensity value, being most likely at different times; in the example, the session using the green curve would reach sooner the third milestone because generating a hint As we see, two different curves would condition how the Narrative Manager select the events (even with exactly the same user actions) by limiting the maximum impact that the selected events can have and by modifying the rate at the Intensity rises.

Our first experiments with the Intensity Curve Modeler application in the laboratory have allowed us to validate the architecture and the requirements of the authoring tool, but more experimenting would be needed, especially in real conditions with real subjects and for comparing with other strategies for drama modeling. However, in this work is not our purpose to give the best solution for authoring the session Narrative; our goal is to design and develop a functional and effective way for authoring it.

4. System evaluation

4.1. FIELD EXPERIMENTS

We conducted two experiments in order to assess different aspects of the Bio-Safety Lab system. The first one was carried out with an initial version of the application without Narrative Manager. In that occasion we were looking for usability data in order to select the interface that better suits the users in our training environment, and to see to what degree the task recognition system improved the experience by itself. We have called this experiment "Usability and Feedback study". With the results of the experiment we could remove the majority of the problems we identified and created a baseline for the next study: we chose the control interface with better usability and presence results, and we established that using our real-time feedback improved users' results.

Next, for the second experiment we polished the application using only the best control method and comparing one version with drama manager and another without it. We called this last experiment "User's Interest and Performance study". The goal of this last experiment is to confirm the hypothesis of this work: that we can increase the number of tasks for the user to solve without dropping the user's interest in the training due to balancing difficulty and intensity, and that this effect has the consequence of improving user's knowledge acquisition (defined as how well the users absorb, store and remember new information). In the next subsections we will discuss both of the experiments.

4.2. USABILITY AND FEEDBACK STUDY

This field study was conducted with students from the Medical campus of Kyushu University to answer three questions. First, we wanted to investigate whether real-time (dynamic) feedback allows for faster recovery from mistakes than static feedback, like traditional methods like reading manuals. In our case, the static feedback used in Bio-Safety Lab consists in opening a screen in the application which presents the protocol in text form. When opened, users cannot perform other actions until they close this screen. We hypothesize that dynamic feedback is more effective than static feedback. Second, we wanted to know whether dynamic feedback or static feedback is better regarding the recall of bio-safety lab protocols. And finally third, we wanted to assess whether a Kinect device or mouse device is more intuitive and easy to use by this user group.

The study had twenty-eight subjects, who were senior students of the Medical laboratory technologist course, and a few freshmen from the graduate course. Thus, our study well exceeds the number of required subjects for a usability study (Hwang and Salvendy 2010). The subjects were recruited since all had completed the clinical microbiology lecture and practice, so we assume they understand basic facility of a laboratory and good microbial practice protocols in general. None had experience using 3D virtual world technology or motion capture applications such as Kinect. In the study, we had twenty females and eight males with an average age of 21.6 years. The study was assisted by four non-technical assistants, three technical assistants and two experimenters. Finally, the subjects were paid an equivalent of JPY 1000 in the form of gift coupons for participation. We

prepared four conditions, designed for comparing the two different control interfaces and the two feedback systems:

1. "Kinect tutorial" condition: subjects use the Kinect device to perform pre-defined tasks in the 3D environment, such as grabbing an object and bring it to another location, opening the door of a container, etc. Those simple tasks are aimed at practicing the operation of the interaction device and are not related to resolving an accident in the biosafety lab.
2. "Mouse tutorial" condition: subjects use the mouse device (and keyboard) to perform predefined tasks in the 3D environment (the same that are in the Kinect tutorial).
3. "Dynamic Feedback" condition: Using the Kinect control interface, the user solves the spill accident scenario using the system we developed (but without Narrative Manager). The user thus receives real-time feedback (in the form of warnings when he makes mistakes) when making a mistake in the application regarding the bio-safety protocol.
4. "Static Feedback" condition: Using Kinect, the user solves the spill accident scenario without using the system we developed, but they can access a text manual upon request when getting stuck. This manual is accessed by doing a specific gesture in front of the gesture capturing device and by doing this the simulation is paused and the manual is the only thing that can be seen in the screen, allowing the user to read it but not to act at the same time.

Note that we use a within-subject design when comparing the Kinect interface to the mouse interface, and a between-subject design when comparing dynamic and static feedback conditions. Four subjects did the experiment in parallel. They are assigned to groups A, B, C, and D, each group containing a different combination of the order using the control interface with one of the feedback systems. Figure 15 shows how we organized the subjects' groups and the stages of the experiment, which was divided into six parts:

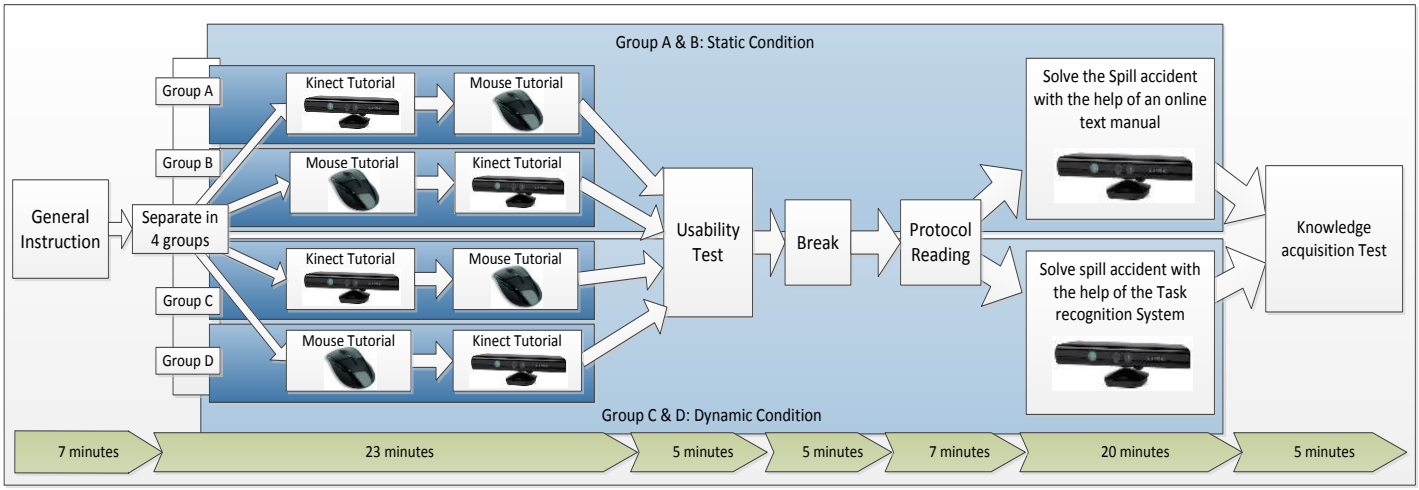


Figure 15: Design for the first experiment showing its stages. The users were distributed in four groups for the tutorials, and then half of the groups run the scenario receiving help from the Task Model Reasoner in the form of warnings. The other half used an online text manual which they could read at any time.

1. Welcome: subjects were welcome and receive general introduction (7 min, see Appendix B).
2. Tutorial and Usability Assessment: subjects learn how to control the environment with Kinect (13 min) and mouse and keyboard (10 min) in tutorial style. Groups A and C had first the Kinect tutorial and the other two had first the mouse tutorial.
3. After the subjects finished the tutorial, they filled usability questionnaires (5 min).
4. Protocol Reading: subjects are presented a protocol (standard guidelines, see Appendix C) about the treatment of a toxic spill (5 min).
5. Spill Treatment and Knowledge Acquisition: subjects solve the spill accident with Kinect interface (20 min). Groups A and B played the scenario in the Static feedback condition and C and D groups did it in the dynamic feedback condition.
6. Afterwards, the subjects did the knowledge acquisition test (eight multiple-choice and writing questions), and then answer the questions from Table 2 about presence, application experience and perception of learning (5 min). When they finished, they received the reward.

When interacting with Kinect, subjects were standing at 1.6 meter distance from a 42 inch monitor. When interacting with the mouse device, subjects were seated at a table in front of the monitor. In the Spill treatment stage of the experiment the subjects played a spill accident scenario, where they were told to perform a simple routine called "Isolation of infectious bacteria in human blood sample" in the virtual environment. While the subject was carrying out this task, the system automatically created the accident of a hazardous spill when the user opened the incubator, which was a mandatory action for the initial task. In accordance with the protocol, the "User Treats Spill" task was triggered. The experiment lasted for 75 minutes with a five minute break after 35 minutes.

Questions
U1 I think I would like to use this system frequently
U2 I find the system to be unnecessarily complex
U3 I think the system is easy to use
U4 I think the support of a technical person is needed in order to be able to use this system
U5 I think the various functions in this system are well integrated
U6 I think there was too much inconsistency in this system
U7 I think most people will learn to use this system very quickly
U8 I find the system to be very cumbersome to use
U9 I felt very confident while using the system
U10 I got tired because of the application's control device
U11 In general I think the controls were natural and intuitive
K1 The gestures for moving across the scenario were intuitive
K2 The gestures for controlling the camera view point were intuitive
K3 The gestures for taking and releasing objects were intuitive
C1 I think the system used with Kinect is more usable than with mouse and keyboard
C2 I think the system used with mouse and keyboard is more useable than with Kinect

Table 2: Questionnaire used in the first experiment, containing questions about usability.

Regarding subjects' subjective experience about usability, we prepared questions with answers in a Likert Scale (from "1: Strongly disagree" to "5: Strongly agree"). The list of questions is composed of:

1. Nine questions about the degree of usability of the application, using a very extensively used questionnaire created by (Brooke 1996) with two additional questions (see Table 2, questions from U1 to U11).
2. Three questions to obtain a measure of the usability of the gestural interface, and additional two comparative questions (see Table 2, questions from K1 to C2).
3. Eight questions to assess the level of user's presence, using a subset of the questionnaire created by (Witmer and Singer 1998).We considered only using the questions linked to the control factors, since this was the most prominent feature we wanted to analyze in our application (see Table 3, questions from P1 to P8). We added seven more application-specific questions about users' experience and perception of learning (see Table 3, questions from A1 to A7).

Questions
P1 I was able to completely control events in the world
P2 The environment was very responsive to actions that I initiated or performed
P3 My interactions with the environment seemed very natural
P4 The mechanism which controlled the movement through the environment was very natural
P5 My experiences in the virtual environment seemed very consistent with my real world experiences
P6 I was able to move and manipulate objects in the virtual world very well
P7 I adapted to the virtual environment experience very quickly
P8 I experienced a lot of delay between my actions and the expected outcomes
A1 I remember completely the session in the virtual environment
A2 I knew what to do next at any time
A3 I noticed warning messages from the system
A4 When reading a warning message from the system, I understood why
A5 When reading a warning message from the system, I knew what to do next
A6 I understand all the problems that can arise solving a spill of infected blood
A7 I think this application would be very useful to laboratory training procedures

Table 3: Questionnaire used in the first experiment, containing questions about presence and perception of learning.

The subjects were also tested on their knowledge of the spill cleaning protocol and their recall of the correct order of actions to be carried out of the spill treatment. There were eight multiple-choice questions with three multiple choice options, such as "What is the correct spill treatment order?", "Which is the correct spill disposal procedure?", etc.

4.3. USER'S INTEREST AND PERFORMANCE STUDY

We tested in an experiment if our Narrative Manager can generate more events for the user's to solve and 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 in 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 graduated students and everyone had experience with navigation in 3D virtual environments. The experiment was carried out by one experimenter and one non-technical assistant and lasted approximately 75 minutes. Finally, the subjects were paid JPY 1000 in cash for participation.

The experiment consisted of 3 stages: first the user does a tutorial in order to learn about the domain and the 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. Figure 16 shows a diagram of the design of the experiment. The three stages are detailed next.

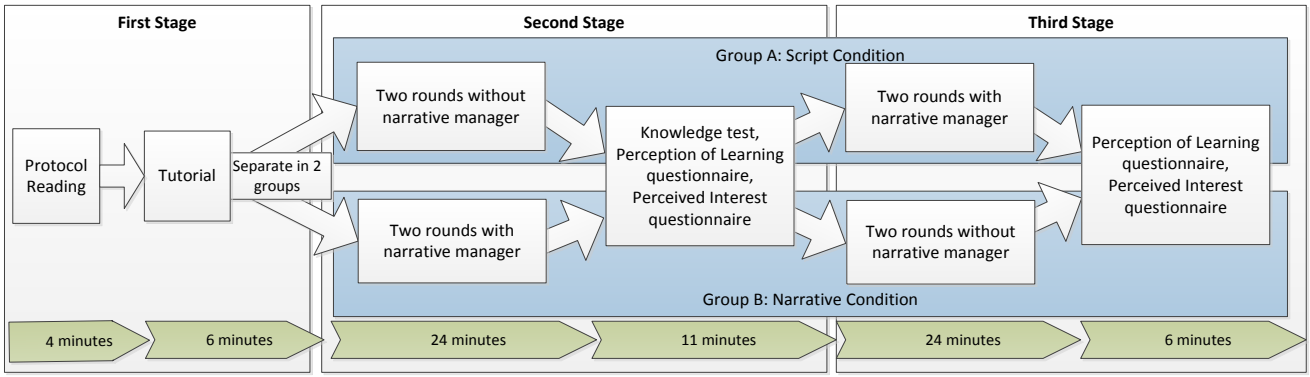


Figure 16: Design for the second experiment showing its three sequential stages with the duration of each one. The users were distributed evenly to the two groups, but they didn't know to which one they were assigned.

First stage: The participants read an information sheet and a manual with the protocols for clearing certain accidents in a bio-safety laboratory (see Appendix B and C). The manual contains a brief description of how to clear the accident in the application but also a guide with pictures of laboratory material and instrumental for the training session. The subjects can consult this manual in 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 they also familiarize 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 were separated randomly in two groups: Script condition and Narrative condition, running two sessions of the scenario each one. 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 twice the training scenario in a version of the Bio-safety Lab without the Narrative Manager. The participants in the Narrative condition run twice the training scenario in the Bio-safety Lab with the Narrative Manager, which will trigger dynamically events in

real time. Both versions contain one pre-scripted accident (spilling human blood infected with bacteria, see Figure 3 to a graphical depiction of how a sample session without Narrative manager is conducted) soon after starting the application; until then the two scenarios contain no differences, and we consider this concrete instant the experiment start point and it finishes when the user runs out of time (12 minutes) or when he clean all the existing spills. Both versions also use the Task Model Reasoner for recognizing user's tasks and giving warning when the user makes mistakes. Thus, in the version without Narrative Manager, the only events that do not come from the user actions are the spill accident as it is scripted and the Task Reasoner Module warnings. After the subjects finished, they filled first a Knowledge Test about the spill cleaning protocol, containing questions like "*There is an incorrect position to handle a spill? Where and why?*", "*What is the correct order for covering the spill?*" or "*Which material should be used for bordering the spill?*" among others. Then the subject filled a Perception of Learning questionnaire (see Table 4, questions from P1 to P12) used to capture the user impression of learning from the simulation. This information is important and one of the sources for evaluating training tools (Schuman, P. L., et al., 2001). After the Perception of Learning questions, they filled an adapted Perceived Interest Questionnaire (Schraw, 1997) to evaluate how interesting was the system they run. We don't use all the questions of the Perceived Interest Questionnaire because not all were applicable to the system, being a generic test for narrative works so we selected only questions applicable to our domain (see Table 4, questions from I1 to I7).

Questions
P1 I remember completely the session in the virtual environment
P2 I knew what to do next at any time
P3 I noticed warning messages from the system
P4 When reading a warning message from the system, I understood why
P5 When reading a warning or an advice message from the system, I knew what to do next
P6 I noticed differences in the application between sessions
P7 I think while using the application the scenario tried to help me
P8 I think while using the application the scenario tried to obstruct me
P9 I remember how to treat a spill of infected blood
P10 I understand the problems that can arise solving a spill of infected blood
P11 I think this application would be very useful to training laboratory procedures
P12 The events in the system were different each time I played
I1 The session was suspenseful
I2 The scenario grabbed my attention
I3 I would like to play again if I had the chance
I4 I thought the session was very interesting
I5 I got caught-up in the session without trying to
I6 I would like to know more about the scenario
I7 I liked this session a lot

Table 4: Questionnaire used in the second experiment containing questions about users' perception of learning and perceived interest.

Third stage: Once the participants complete the questionnaires, they run again twice the scenario, but this time they do it in the other condition. The reason for having another run with the other version of Bio-safety Lab is giving the participants the opportunity to compare which one is more interesting. Finally, the participants are given again the Perceived Interest questionnaire (again, questions in Table 4 from I1 to I7) and three additional comparison questions (see Table 5) for evaluating the system they have just run. However we did not make the user to pass the Knowledge Test because in our experiment the scenario too simple, so after all the runs of it, the users would remember everything easily. The questions from the questionnaires were in the form of a Likert scale and are contained in the Table 3. We labeled the Narrative Condition as NC system in order to not giving too much information to the user.

Questions
C1 I think the first system was more amusing than the other
C2 I think the first system was more difficult than the other
C3 I think the first system was more challenging than the other

Table 5: Questionnaire used in the second experiment containing questions about comparisons of the system used. Please note that depending of the group the subject was in, the first system would be the one with Narrative Manager or without it.

5. Results and discussion

5.1. Usability and feedback Results

In this subsection we will discuss the results obtained in the first experiment. These results are related mainly to the usability of the control interface and the benefits of having a system with dynamic feedback. In order to measure the performance of the subjects, we divided the "Spill Treatment" task in five ordered steps, or subtasks: (1) Bordering the spill, (2) covering the spill, (3) disinfecting the spill, (4) merging the waste, and (5) disposing the waste. Thus, our metrics are the number of steps one user could complete in the session, and the time to complete each one of them. In the dynamic feedback version, subjects could complete 4.6 steps on average during the allotted 20 minutes, with ten subjects completing all five steps. In the static feedback version, subjects could complete 3.3 steps on average, with four subjects completing all five steps. This shows that in the dynamic feedback version, more users were able to complete the spill treatment, and in general they completed more steps of the task. The average duration for the tasks is shown in Table 6.

Condition	Step 1	Step 2	Step 3	Step 4	Step 5
Dynamic Feedback	197.3	308.8	218.7	135.4	52.6
Static Feedback	231.2	449.5	257.4	139.7	47.7

Table 6: Average time consumed to complete each one of the five steps.

Type of mistake	Dynamic feedback		Static feedback	
	Average	Std.dev	Average	Std.dev
Interposing between the spill and the air vent	5.66	8.65	11.75	19.97
Cover spill with sponge	91.99	63.55	188.92	208.53
Cover spill with pad	43.44	45.23	271.20	199.20
Cover center of spill before outsides	22.34	24.97	54.30	61.31
Disinfect the center of the spill before the outsides	0.35	0.07	NM	NM
Throw waste without merging	97.18	58.74	116.00	30.61
Throw waste in wrong bin	49.00	N/A	22.00	N/A

Table 7: Average time consumed for correcting mistakes and standard deviation (in seconds) in dynamic and static feedback groups. "NM" abbreviates "no mistake" and N/A means that there was only one mistake, so standard deviation is not computed.

We now turn to the general analysis of user mistakes that occurred during the interaction with the system. Table 7 reports on the average time taken by subjects to correct one instance of a mistake in the dynamic feedback version and static feedback version, respectively, and Table I in the appendix D shows the t-test results for comparing the two groups. The results indicate that dynamic feedback is significantly more effective than static feedback, i.e., users receiving real-time assistance could

recover from errors significantly faster. This finding supports our claim of the importance of real-time task recognition.

Question	Kinect		Mouse	
	Average	Std.dev	Average	Std.dev
(U1) I would like to use this system frequently	3.18	0.77	3.46	0.96
(U2) The system was unnecessarily complex	2.75	0.89	2.18	0.90
(U3) The system was easy to use	2.93	0.90	4.00	0.77
(U4) The support of technical person is needed	3.93	0.68	2.86	0.85
(U5) The functions of the system are well integrated	4.04	0.79	4.18	0.55
(U6) There is too much inconsistency in the system	2.43	0.74	2.22	0.75
(U7) People will learn how to use the system quickly	3.71	1.05	4.21	0.67
(U8) The system is very cumbersome to use	2.82	0.86	2.21	0.92
(U9) I felt confident while using the system	2.90	0.96	3.68	0.90
(U10) I got tired because the control device	3.18	0.98	2.07	0.90
(U11) The controls were natural and intuitive	3.21	0.92	3.71	1.05

Table 8: Average and standard deviation values for the results of the usability questionnaire. The numerical values range from "1: Strongly disagree" to "5: Strongly agree".

Question	Average	Std.dev
(K1) Moving across scenario intuitive	3.29	1.01
(K2) Controlling camera viewpoint intuitive	3.04	1.07
(K3) Taking and releasing objects intuitive	3.00	0.98
(C1) Kinect more usable than mouse/keyboard	2.39	0.83
(C2) Mouse/keyboard more usable than Kinect	4.11	0.63

Table 9: Average and standard deviation values for the additional questions on Kinect and comparative questions between the two control interfaces. The values range from "1: Strongly disagree" to "5: Strongly agree".

Table 8 presents the results of the usability of Kinect and mouse/keyboard (for more details about the statistics comparison results see table II in appendix D). The usability results, specifically U3 and U10, indicate that the subjects preferred the mouse interface over the Kinect interface with statistical significance. Table 9 shows the results for the questions specific to Kinect and the comparative questions. Judging from the numbers, the perception of operating Kinect is neutral. Informal comments on usability of Kinect also provide some valuable insights. The advantage of Kinect is only felt after some practice time. Still, Kinect is seen as more tiring than the mouse interface also in the free-format feedback; the mouse/keyboard device is seen as more usable than the Kinect interface. Table 10 presents the results on the presence and application experience questions in both dynamic and static feedback groups. Table III in Appendix D contains the details of the t-test results for the two feedback groups. In terms of presence (questions P1 to P8), both interfaces ranked low. One explanation is that the application has some design issues that lower immersion related values, e.g., the

user can easily ‘undo’ already successfully performed actions, and consequently has to redo those actions. The majority of these ‘undoings’ are due to lacking precision with the gesture control. This issue could be overcome by ‘locking’ objects in place when correctly used in order to prevent the user from move them and thus undoing correct actions by mistake. Importantly, the medical students considered the application as very useful for laboratory training procedures (Question A7).

Question	Dynamic		Static	
	Average	Std.dev	Average	Std.dev
(P1) I was able to completely control events	2.22	0.97	1.71	0.61
(P2) Environment was very responsive to my actions	3.07	0.99	2.36	0.74
(P3) My actions within the environment seemed natural	2.21	0.89	2.00	0.55
(P4) My movement through environment was very natural	2.93	0.99	2.86	0.66
(P5) The Virtual environment seemed consistent to the real world	2.86	1.09	2.86	0.86
(P6) I was able to move/manipulate objects in the environment very well	2.43	0.94	1.93	0.73
(P7) I adapted to the virtual environment very quickly	2.86	1.23	2.57	0.65
(P8) I experimented much delay between an action and its outcome	3.21	0.97	2.86	0.95
(A1) I remember the session in the virtual environment	3.64	1.08	3.14	0.66
(A2) I always knew what to do next	3.14	1.03	3.07	0.83
(A3) I noticed the system’s warning messages	3.79	0.97	2.43	1.02
(A4) I understood why I received a warning	4.07	0.73	2.50	0.65
(A5) After received warning, I understood what to do	4.00	0.88	2.64	0.74
(A6) I understand the problems I can have when cleaning a toxic spill	4.14	0.66	3.71	0.61
(A7) The Application is very useful for lab training	3.93	0.83	3.71	0.83

Table 10: Averages and standard deviation of the results for presence and application experience questionnaire, comparing dynamic and static feedback. The values range from “1: Strongly disagree” to “5: Strongly agree”.

The final question was whether dynamic feedback or static feedback is superior as to the knowledge acquisition of bio-safety lab protocols. Regarding the seven multiple-choice questions, in the dynamic version (14 subjects), there were 7.64 (of 10) correct answers on average and in the static version (14 subjects) there were 7.5 correct answers on average, not being statistically different. This indicates that the type of feedback has no specific impact on knowledge acquisition of the protocol. We speculate that the subjects remembered the protocol, which they read just before the spill cleaning task, equally well. It is interesting to note is that the students using the dynamic feedback version had the impression of better recall of the session and better understanding of spill handling and mistakes (Questions A1 to A6).

In summary, seeing all of these results a number of conclusions can be extracted: First, seeing the comparison between Kinect and mouse, we can state that contrarily to what we thought, the gesture interface does not enhance the user immersion, but the contrary: it needs some practice and assistance to be used to, and is not as accurate and natural as the mouse interface. Thus we decided to continue the development of our tool and the rest of the experiments only with the mouse interface. Second,

receiving dynamic feedback from the system improves the user's performance and perception of learning, so we were in the right track for the next experiment, in which we focused in the learning features of Narrative Manager using as a baseline the system with dynamic feedback from this first study. In the next subsection we will analyze the results of the second experiment and discuss them.

5.2. NARRATIVE MANAGER AND USER'S INTEREST RESULTS

In our second experiment we search for a confirmation of our main hypothesis: does the narrative Manager's event generation helps in improving the user engagement by balancing the difficulty of the session and thus keeping him interested? If our hypothesis is true, this effect should be reflected in the questionnaire results as well as in the users' performance and in their knowledge acquisition. Also, by analyzing the training session we should be able to identify how the difficulty balance was carried out and if it follows a narrative pattern.

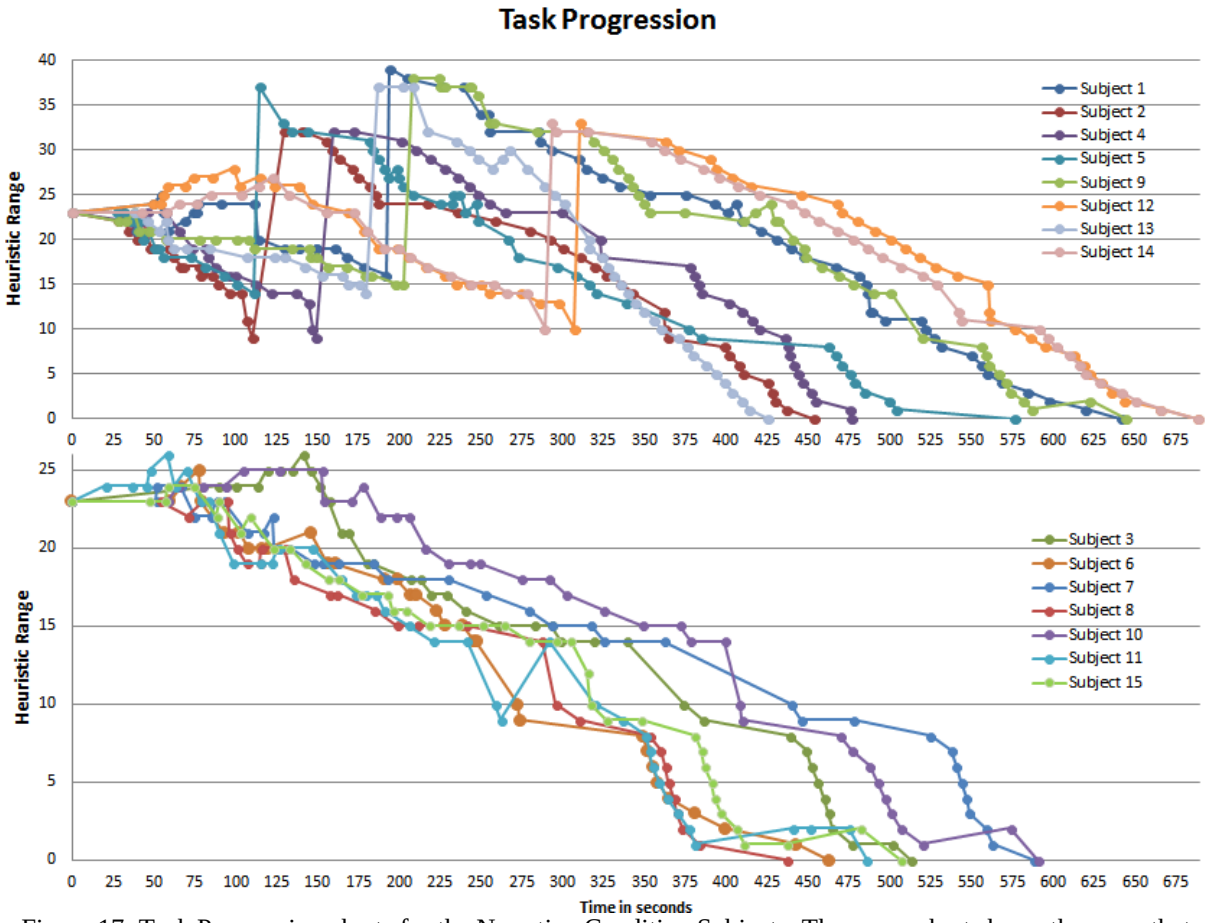


Figure 17: Task Progression charts for the Narrative Condition Subjects. The upper chart shows the users that advanced quickly to the goal and the Narrative Manager decided to create an additional spill (represented by the big peak of each graph). The bottom chart shows the users that advanced slowly in their task so the Narrative Manager only presented them small difficulties.

In order to do this, we extracted the behavior information of the subjects from the system logs that Bio-safety Lab creates, in order to represent in a chart how they proceeded to the task completion. We choose the burn-down charts due to being widely used as a graphical representation of work left to do versus time (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 do for clearing a spill, starting at 23 and finishing at 0. Each one of the dots in the graph symbolizes one action from the user or one event from the system, and when the value reach zero in the Heuristic range axis it means that the subject cleared the scenario.

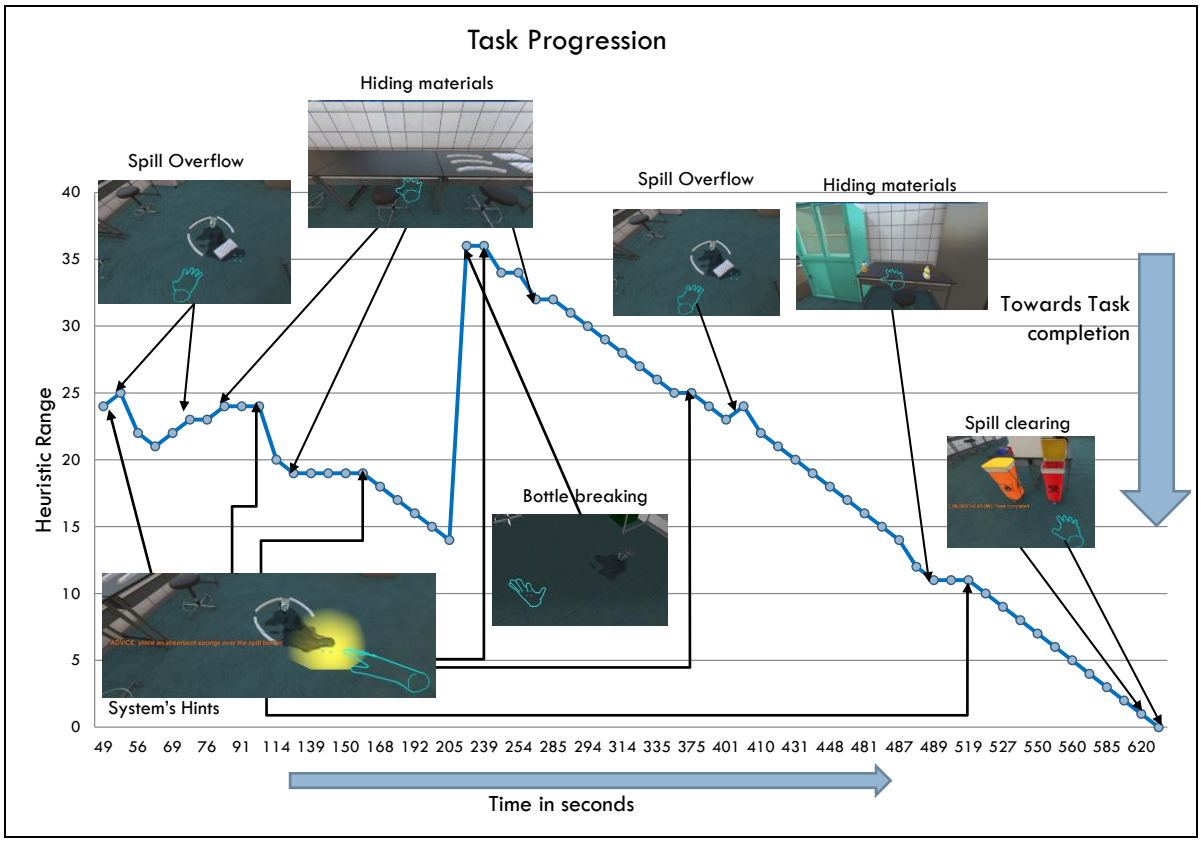


Figure 18: Task Progression Graph for the subject 1 in the Narrative Condition. We can observe how the task advances or goes back depending of the event ordered by the Narrative Manager

Figure 17 shows the charts created for all the subjects in the Narrative Condition, distributed in two groups (the Scripted Condition subjects follow a homogeneous tendency and will be discussed in 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 making 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 max of one per session, due to take 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 didn't have a second spill. In these cases, the graphs have hills representing small problems for the users, like spill overflowings or missing tools.

We can further analyze the behavior graphs and identifying the impact that each Scenario Event has in the task completion. Figure 18 shows the task progression graph of one subject of the experiment, and how behaves when certain events happened. As we can see, when an event difficult for the user happens, the curve rises; for example, when the spill overflows, the curve rises a bit because is a light inconvenient that requires to redo some actions. 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 do another 23 actions for clearing the scenario. Some of the "bad" events does not create a rise, but creates a plane zone. This is the case of hiding materials from the user, which causes the user to lose some time in doing other actions and searching for the materials, stopping 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 the task completion. Finally, the two spills were cleared at the end, being 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 again clearing the situation, and except for some minor problems that he could solve, he finish the scenario.

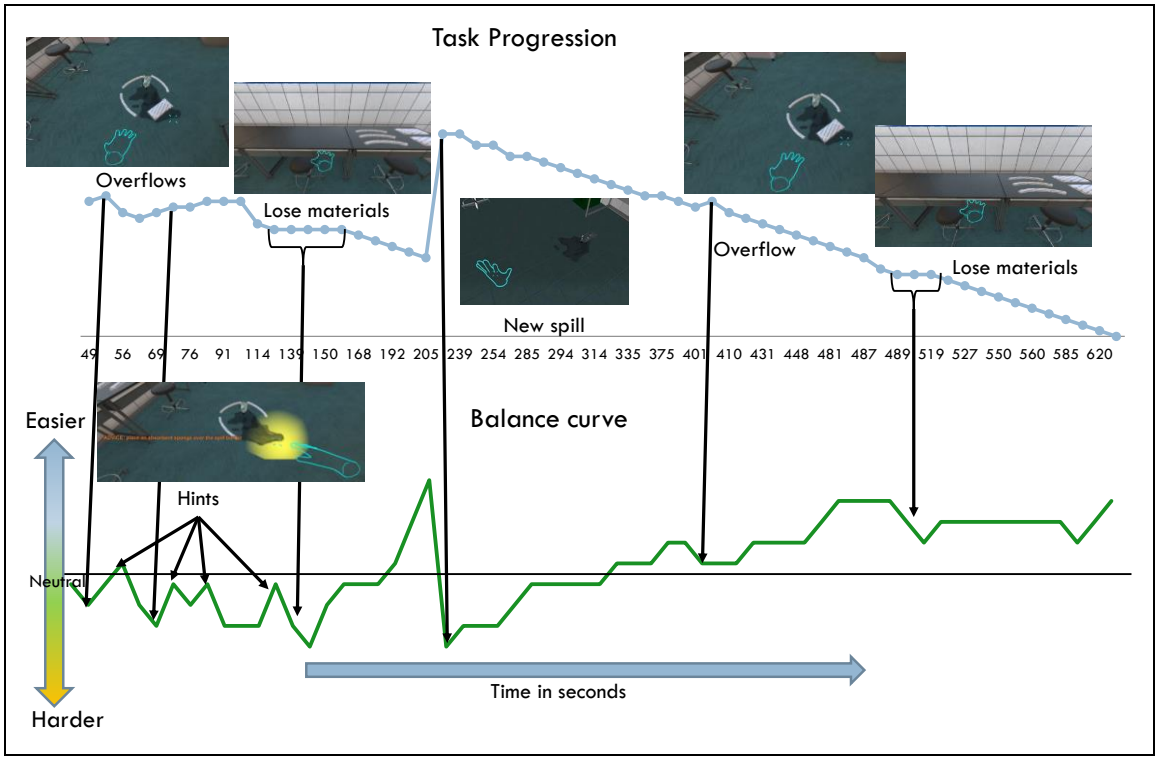


Figure 19: Balance graph related to the Task Progression for the subject 1 in the Narrative Condition. The events sent by the Narrative Manager can be directly related to increasings or decreasings in the balance levels.

By analyzing the events that happened in the session, we can also calculate the Balance parameters for each instant in the session. So we can create a Balance curve that represents how well the situation developed for the subject as he proceeds towards the task completion. If we relate the task progression graph with the balance curve for that session (see Figure 19) we observe that these events are translated directly in increasings and decreasings in the session's balance. For example, systems 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 and we can see it happened just after the subject was advancing steadily to the task completion. The Narrative Manager decided that the task became too easy for the user at this point and it should stress him more so it created a new spill. We can see that as the user is completing the task, the balance curve goes up again, and the system keeps oscillating the balance, and towards the end, when the user is solving quickly the situation, the system tried to decrease the balance but the user managed to clear the accidents.

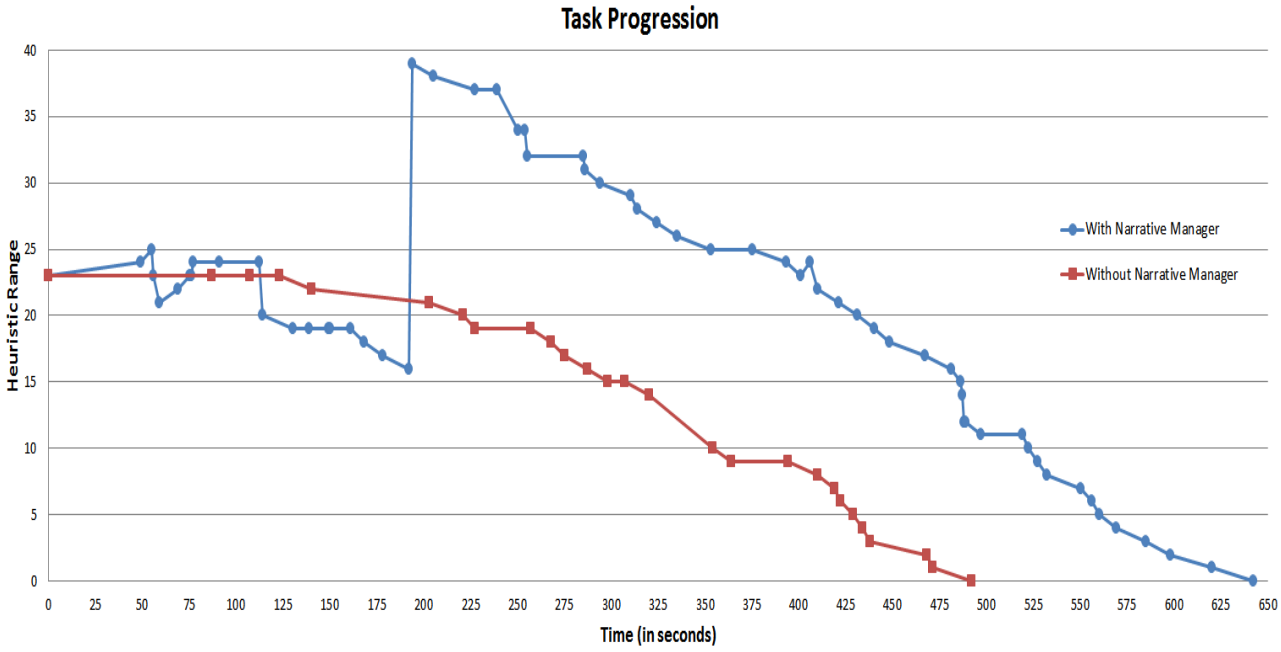


Figure 20: Comparison between two sessions from the two conditions: The upper graph (blue, circle dots) represent the session of a subject in the group With Narrative Manager, and contains more events and takes more time than the bottom graph (red, square dots), from a subject in the group Without Narrative Manager.

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 much 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 would contained a number equivalent of system events it would be interesting to compare, but in that case, if the events are scripted the version without Narrative Manager would contain actual narrative, so we decided to maintain only 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 about this idea. In the Figure 20 we compare two subjects from the two groups, and we can observe the difference in event density. The subject from the Narrative condition in the figure is the same we have analyzed in this section (Figures 17 and 18), containing 204 events, and the one from the session without Narrative Manager contains 76 events.

Question	Average
(C1) I think the NC system was more amusing than the other	7.16
(C2) I think the NC system was more difficult than the other	5.08
(C3) I think the NC system was more challenging than the other	6.75

Table 11: Average results of the comparison questions. The average values are in the range from "0: Strongly disagree" to "10: Strongly agree".

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 comparison questions in the questionnaire (see Table 11) 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 the user doesn't think the Narrative condition is not harder than the scripted one, but he/she also perceives that the Narrative condition is more engaging and interesting.

Question	Narrative Manager	Without Narrative Manager	T test value
(I1) The session was suspenseful	5.66	5.0	0.177
(I2) The scenario grabbed my attention	7.66	6.0	0.005
(I3) I would like to play again if I had the chance	6.66	5.83	0.178
(I4) I thought the session was very interesting	8.33	6.16	0.005
(I5) I got caught-up in the session without trying to	6.83	4.83	0.015
(I6) I would like to know more about the scenario	7.33	5.66	0.015
(I7) I liked this session a lot	6.83	5.5	0.036

Table 12: Average results of the perceived Interest questions. The average values are in the range from "0: Strongly disagree" to "10: Strongly agree".

For the next questionnaires results (see Tables 12 and 13), we conducted a T test for two populations assuming equal variances with an alpha value of 0'05 in order to compare how different are they (for more information about the T test values, see Table IV in Appendix B). The results of the interest questions are also positive (see Table 12), indicating that the Narrative Manager system was more interesting than the other one, maintaining the user engaged in the training session, and backing 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 this in a better learning outcome. We are aware that this good results can due to the scenario was a single accident with a straight forward protocol, but we think that the simplicity of our scenario helps more to the Scripted Condition users than to 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 test we used for measuring the learning outcome could not be enough for confirming absolutely how much our system improves learning. However, as the focus of this work is analyzing how our drama manager improves the user's interest in the training session, we think the knowledge questionnaire was enough for confirming if the Narrative Manager system has better learning values compared to the scripted system and if it can be a consequence of the improving in the interest (as it is the only difference).

Question	Narrative Manager	Without Narrative Manager	T test value
I remember completely the session in the virtual environment	9.16	7.33	1e-3
I knew what to do next at any time	8.5	6	8e-4
I noticed warning messages from the system	8.16	7	0.03
When reading a warning message from the system, I understood why	8.16	6.5	0.01
When reading a warning or an advice message from the system, I knew what to do next	8.33	5.0	1e-6
I noticed differences in the application between sessions	7.66	4.33	7e-4
I think while using the application the scenario tried to help me	8.16	6.16	4e-3
I think while using the application the scenario tried to obstruct me	5	3.33	0.02
I remember how to treat a spill of infected blood	9.33	7.83	0.01
I understand the problems that can arise solving a spill of infected blood	8.66	7	0.02
I think this application would be very useful to training laboratory procedures	9.33	8	0.01
The events in the system were different each time I played	7.5	2.83	3e-5

Table 13: Average results in the Perception of Learning questions. The average values are in the range from "0: Strongly disagree" to "10: Strongly agree".

Moreover, in the subjective Perception of Learning questions shown in Table 13, we can observe better values too. We only interpret these questions as an indicative of the better training experience for the subjects, as we have already the knowledge test as an objective metric for learning improvement. The results are better in the Narrative Manager system in all the questions. The subjects perceived subjectively that they remembered better training session with the Narrative Manager. Also they perceive more help from the system and remember better the messages they received, something that is logic, as the Narrative Condition effectively helps the user and has more warning messages, including the proactive hints for the user (the version whiteout 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 run the scenario, and even if both of the systems use the same warning messages to correct user's mistakes and use them in the same way, in the Narrative Manager version they are better perceived.

5.3. DISCUSSION

After analyzing the results of the experiments we can discuss further their significance and implications. Certainly the results confirmed our hypothesis: the Narrative Manager improves the knowledge acquisition of the users due to making them interested in the training by balancing the difficulty. Also, Bio-Safety Lab is the only system with this kind of proved model. However, other training applications also demonstrated improving in user's results so we have to compare them with ours. In the related work section we surveyed different training tools in the literature, and explained their similarities and differences with Bio-safety Lab, highlighting the features of our system that solved their deficiencies when necessary and we showed that there are not many narrative management system in training applications and even those that have it neither they make a formal model of the narrative system nor evaluates it comparing it with other options.

System	Type of training	Narrative	Interaction	Scalability	Variability
VRML (Angelov, 2007)	Validation Test	No	High	Good	Low
VR training for Mining (Van Wyck, 2009)	Validation Test	Script	Low	Bad	Very Low
Crisis Communication (Carpenter, 2006)	Interactive training with assistance	Script with Human Director	Low	Bad	High
X3D (Belloc, 2009)	Validation Test	No	High	Good	Low
STAR framework (2012)	Interactive training with assistance	Narrative Management	High	Bad	Medium
Raybourn et al. (2005)	Interactive training with assistance	Human Director	High	Bad	High
Rowe (2009)	Interactive training with assistance	Script	High	Good	Low
Bio-Safety Lab	Interactive training with assistance	Narrative Management	High	Good	High

Table 14: General comparison between different training systems and Bio-safety Lab. We compare five dimensions indicating the type of training, if has Narrative control and what kind of it, the level of interaction, how scalable the system is and how easy is to create different training sessions. Bio-Safety Lab has high scores in these last three dimensions.

Table 14 contains a comparison of the general features between the most representative training systems we mentioned in the related work section with Bio-safety Lab. We have established five dimensions to compare, the first and the second being descriptive of the type of training and the last three qualitative dimensions of different features. First, in the Type of training, we can divide all the systems in two groups: validation test systems and Interactive training with assistance ones. The first group consist in applications which present a problem to the user and only advance to the next step of the training when the user gives the correct answer. This answer can be textual or a more complex one, like selecting an object from a scenario (in VR training for Mining) or putting a piece in the correct position and order in a machine (in VRML). The second group consists in applications that let the users to interact freely with the environment, solving their goals in the order and way they want, being this factor an improve in user immersion in the scenario. Of course, applications in both groups obtain good knowledge acquisition results when confronted with traditional methods like books and

manuals, but experiments given in the literature do not compare them with more advanced applications or even other interactive options.

The next dimension we examine is the Narrative capability. As we defined, a narrative implies having some rules for generating the scenario events in order to reach some goal. We have showed in the related works section how a well implemented narrative management well-aligned with the training improves the levels of user's presence and interest. Thus, we can find systems with no narrative, or systems using a script for the training session. Some systems use a Human director who decides the events to trigger, and finally others use a Narrative Manager controlling the virtual scenario. The systems that use some kind of narrative control benefit from it by having a high presence and interest values, and if those systems have a Narrative Management they also lack the constraint of having to script a new session each time we want a different scenario. The third dimension, Interaction, indicates how wide the user's possibilities when participating in the training are. In systems with low Interaction, the user can only select actions from a restricted list of possible ones. On the other hand, systems with high Interaction, allow the users to freely manipulate al the objects or options available in the scenario. High Interaction translates directly in high user's presence. Next, the Scalability dimension indicates how easy is to create new objects or events in the training scenario. In systems with bad Scalability, creating a new object requires modifying the core of the application with new code, and generating a new version of it, while in systems with good scalability new objects or events can be introduced without changing the code if certain conditions are met. Systems with high Scalability contain heavily data driven engines, allowing the real time loading of scenario data like objects and events: Usually, if similar objects or events are needed, it only will require some changes in the configuration files, or in some cases if new models are needed, just adding the model new files will work. It is interesting to note that in Bio-Safety Lab, to add new events is only necessary to add them to the knowledge database (if the event is something completely different and involving new effects and objects maybe it will be needed to add new models and logic to the system though), but also the narrative manager can be assembled to another completely different training system due to its generic interface, and we would only need to create the Task Trees for the new domain.

The last dimension of the table, Variability, shows the how many changes are need to be done in the system for making different training sessions (always inside of its domain). In this dimension we observe a wide range of results. Systems with low variability require making a new script or even new logic rules in the application engine in order to change the events in the scenario session. The most extreme case would be VR training for Mining, which is video based, so it requires to change the whole video in order to creating a different scenario, so even with slight changes a high effort has to be done. The STAR framework has a medium value due to have a structure similar to an adventure game, is necessary to change the game data in order to change the scenario and if the changes include the events that happens in the session, the game logic needs to be modified as well. Finally, systems with high Variability allow changing the scenario events, and their ordering with minimal changes or even without changing anything. In the table, those applications have a human director for the scenario, so they can narrate spontaneously and trigger manually the events, or have a narrative manager controlling the session, like our Bio-safety Lab, allowing automatic event generation, and making different each session.

We can see from the table that Bio-safety Lab has high values in these last three dimensions, but those are only technical features of the systems so we should compare as well the learning features. It turns out that all the training applications we reviewed gave good results in terms of user's presence, user's interest and knowledge acquisition, but it is also true that in the experiments they conducted for obtaining those results they only compared the system with a traditional method like memorizing texts (as we said before) or event they didn't compared with anything. Then, the results of those other systems only shows that are a good training tool in general, not being an absolute measure and cannot be addressed as a consequence to any concrete feature of the system. Also it can be argue that comparing an interactive system with memorizing a text is not fair at best, being obviously better, and the results trivial. On the other hand, we made a two evaluations with a comparison with another training system each: first, we compared a system without narrative manager but with real-time assistance against a system without this assistance, but with a readable manual; then, a second evaluation with a comparison between our Narrative Manager and the first system, being the only difference between the two the addition of the Narrative Manager. Our first comparison is already better than the best evaluations we can find in the literature, in the terms that the results won't be trivial. For example, we didn't expect that the results in knowledge acquisition were so similar. In a pilot experiment previous to the first one we conducted, we compared the system (without narrative Manager) with only memorizing a manual, obtaining good but trivial results. Once we proved that the system with real-time assistance was effective, we can compare it with a system that was identical except in one feature: our Narrative Manager. As we have seen, it gave good results in terms of user's interest and knowledge acquisition, and we are able to say that these results are directly related to the Narrative Manager. In the future we would like to compare the system with other different training methods, including a completely scripted scenario designed for being very interesting and challenging. Theoretically, a scripted scenario like that would have better or at least equal results than Bio-safety Lab, but precisely one of the advantages of our system is that it makes unnecessary to make such complex scripts which are very costly in terms of effort and is only effective a short number of times because the user can memorize it after playing it repeatedly, losing quickly its training value. It would be interesting to compare how the user interest is in both systems after running the training scenario once and after running it a number of times. We will extend this idea further in the next chapter of the document so next we will discuss the narrative method we developed.

We proved our hypothesis and showed that our Narrative manager obtains its goal with very good results in the aspect of user interest and knowledge acquisition, but it can be argued that the model behind it has some seemingly arbitrary aspects. The parameters' values in the Scenario Task Tree are given by the scenario designer and there is no formal metric to calculate them; certainly, this fact is acknowledged also by Ware & Young in their work but we see this fact as a feature of any system that requires domain knowledge (i.e. our Task Trees) to work. In the development of Bio-safety Lab we had extensive meetings with our domain expert collaborators from the National Institute of Infectious diseases, where we obtained from them all the knowledge necessary to construct the trees and the get the values of parameters we needed. On the other hand, the election of the parameters themselves and its processing could seem arbitrary too. But this is not true; the election of the parameters follows the conditions to make the user to enter in a psychological state of flow (Nakamura & Csikszentmihalyi, 2002): to have a good balance between the perceived challenges of the training task and their own perceived skills. Concretely, we use the Balance and Impact parameters to give a value for the challenge that entail one event (Balance would be the real challenge level and Impact the perceived

one). With the Global Balance and the Intensity we evaluate the current levels of challenge and the system knows how the user is performing (and therefore his skills) by analyzing how he is advancing trough the task completion. Thus, even if our chosen parameters may not be the best solution, this model has a theoretical background (the psychological flow theory) and furthermore, by seeing the experiment results we proved that it worked, so the parameters and their values worked well. In the future we would like to experiment more in this aspect, comparing the use of different parameters or giving different values; even comparing it with random values would be interesting. In the next section we will elaborate more about this matter.

Our system's narrative parameters are different of the ones given by Ware & Young due to be in a very different domain: the main difference between them is that in our training application there is a very clear goal (to balance the training keeping the user interested), and is the first narrative system with that kind of goal, whilst in Ware & Young's system there is no goal, simply generating any possible story. Additionally we limited our narrative system to have only one antagonist to the user: the environment; thus, some of the conflict dimensions are not needed, like the emotional distance. Also, the parameters' processing is completely novel: in the original work from Ware & Young there is not any formal method or algorithm to use them. Moreover, their application just generates possible stories but it does not generate explicitly conflict and in their evaluation they only assessed their conflict model, not the system.

Finally we would like to discuss some minor but interesting findings about the gestural interface. Initially we designed this interface following a requirement of realism, in order to make the control the most intuitive we could. In the time we designed it, we perceived the Microsoft Kinect device as a very good option to do it, because it is affordable and easy to port to a large scale experiment, with the possibility of implant it in real training facilities for students. The interface allowed the user to use both hands at the same time and even to move across the environment while performing other actions with their hands. However, the results when comparing this interface with a traditional one (using mouse and keyboard) were poor, even with user not used to play 3D games (who are already used to the traditional interface and maybe biased to favor it). Certainly the gestural interface was perceived as very amusing but also as very difficult to control and not accurate. We concluded that affordable motion capture technologies were still too limited and if a more accurate system would be needed, the training system would not be scalable at all. Other possibility was that our interface approach was not enough complete: experiments using two Kinects at the same time showed promising results (Caon et al., 2011), or even other recent devices like Kinect v2 or Leap Motion[2] could be explored as well. Nevertheless, the gestural interface was not the focus of our system and it could add too much noise to the experiment results because the user is more concentrated in controlling the system than in reaching the training goal, so we decided to remove it from application for the second experiment.

In the next section, the final one of this work, we will present our conclusions, starting with a summary of our contribution and findings, and also extending some of the ideas of this discussion and presenting other news.

[2] https://live.leapmotion.com/

6. Conclusions

In this work we have presented Bio-Safety Lab, an intelligent training or rehearsal system for biohazard laboratory procedures, based on the real-time user‘s task recognition with mouse and keyboard input or gestural interaction and a new method for controlling the events in a training session, using narrative techniques adapted to a training domain. These narrative techniques are integrated in a Narrative Manager module in the Bio-Safety Lab system. The Narrative Manager module communicates with the task recognition system and uses its same knowledge structure for modeling scenario tasks and decides when and which event will trigger, depending on the user’s performance and the difficulty of the situation, balancing 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 him to solve while keeping the user engaged, leading this into more knowledge acquisition. An experiment confirmed this hypothesis, and even if our subjects were computer science students, we believe the results here will generalize well to other populations, especially those interested in Bio-safety procedures.

The main contribution of this work is the design and integration of the Narrative Manager within a virtual training system, and the novel method it uses for controlling the session. This method is inspired by a narrative conflict model and the theory of psychological flow and we designed it for training scenarios, being the first system in including this kind of narrative processing. Other narrative training systems are only deployed for improving user’s immersion and avoid scripting the session every time, whilst our system has a concrete goal: balancing the scenario difficulty and improving user interest and knowledge acquisition. Also, we use a knowledge structure called Task Trees that enable 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 we can modulate narrative intensity to make him interested. Secondary contributions are the integration of real-time task recognition, error recognition strategies and gestural interaction in the training system and its evaluation in a field study. Finally, as we discussed in the previous section, the main limitation of the system is that requires of some degree of domain knowledge to define all the necessary data to work, but aside of that, the system is generic and could be applied to any kind of domain, provided that a domain expert creates this knowledge database: the Narrative Manager is an independent module and it communicates with the virtual environment engine through messages with a documented API, so with the correct domain definition in the knowledge base, can be potentially applied to other types of tutors or training scenarios, i.e. evacuation training, law enforcement, manual assembly or machinery work. Moreover, this only has to be done once, unlike in other scripted systems where the designer has to remake parts of the application data whenever the application runs in order to create a different session.

A first field study demonstrated the robustness of the system and the usefulness of corrective feedback messages for fast recovery in order to. Users would easily see and understand the message when performing incorrect actions. Importantly, errors are more easily described as part of a dynamic task representation than in a manual, which would require repeating error descriptions in every single context in which they can occur. For instance, subjects confused the absorbent paper with the absorbent pad because of their similar look. One subject (in the static feedback condition) even completed the first and second stages of the spill handling with this wrong tool without noticing at all

(he consulted many times the manual, but he didn't notice of his mistake). In this way, using a system with task recognition and responsive warnings users can enhance their knowledge of the training procedure by getting acquainted with the experience of "how-not-to-do-it". With these results we could state that our system could be used for effective training and we could continue to add the Narrative manager, which is the key contribution of this work.

An additional and interesting finding of the first study was that quantitatively the mouse and keyboard interface was clearly superior to the Kinect interface for our target user group, as judged from usability indicators. Still, the Kinect interface did not have negative ratings, and received some supportive comments in the free text part of the questionnaire (see Appendix F). We think that gestural interfaces will keep growing with the time, creating more powerful devices with better results, but our conclusion was that without a state of the art device with very accurate capturing capabilities the users feel that the learning curve of the controls is too difficult. For this reason we decided to keep aside the gestural interface and focus in the traditional controls, removing a potential source of noise in the results for the next experiment: the users could be distracted by the control interface and its problems and this can affect their judgment of the system.

In a second study we tested our Narrative Manager in order to see if it improves the users' experience of the training, keeping them interested by the balancing of the event's difficulty. There, we demonstrated that the Narrative Manager system keeps the users effectively interested and shown how it models the training session by giving significantly better results in the questionnaires. The users are able to train more time when the session is interesting for them, something that can be seen in the session graphs we extracted from the experiment. 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. By training more, but not being bored by the training, the users recall better the trained procedures, and produced better results in the knowledge test. As a consequence of these facts, we can confirm that the Narrative Manager has a positive impact in the learning outcome of the training session. We also analyzed a sample of the subject's session and shown how the Narrative Manager models the session and the balance over the time, creating a narrative curve.

We have opened a number of paths that we want to research as a future work. We are planning to apply this method to different domains to demonstrate its versatility. This will be a good opportunity to run a larger scale experiment with a more complex training procedure in order to analyze better the learning outcome degree of our system, because the simpler nature of the experiment we run doesn't allow us to measure with accuracy how improved the subjects learning. Also, in this future experiment we want to use two control conditions: one with only the necessary minimum of scripted events, as we did before, but also another with randomly generated events. It would be very interesting to analyze the results of that kind of system: Theoretically, if the events are random, there is a possibility that a course of generated events creates a narrative session. In that case, the results could be similar to the Narrative system so we should analyze with what probability or under what conditions can we obtain these results. Other interesting comparison would be creating a scripted training session with the goal of being interesting and challenging, and compare it with our system. Theoretically the results of such a script would be good, but it has some disadvantages: first, even if they provide good results, the efforts to create highly interesting scripts are not needed with our system, and second, a script cannot

adapt its difficulty to the user's skills so in cases with users too skilled or too poor skilled it would not generate good results, contrasting with the our system which is adaptive to the user.

Finally, we would like to continue testing and comparing different uses for the parameters we defined for the Narrative Manager as well as other different strategies in order to choose the best interpretation of a narrative model. By seeing the results of the second experiment we conducted, we proved that the parameters and its values worked well in obtaining our goal, but maybe there are better parameter sets or better ways to give them value. In the virtual narrative field there is a lack of benchmarking or comparison studies due to the difficulty to establish a good metric. However, with the infrastructure we created we made possible the narrative analysis of the training sessions like we did by generating Balance curves and Task Progression curves in our second experiment, so we can compare different approaches in future experiments. Similarly, the algorithm used for our authoring tool when the Narrative manager is fed with an Intensity Curve is one interpretation from a number of possible ones, so more testing would be needed.

In conclusion, we developed a generic system for managing training scenarios using a new narrative method inspired in a conflict model, effectively balancing the training session with the goal of maximizing users' interest and improve their knowledge acquisition. We demonstrated our hypothesis in the presented experiments and we intend to improve the techniques more in the future. Using a Narrative Manager allows enhancing the user experience by keeping a high interest in the session, allowing at the same time for more tasks to train, with the consequence of improving the learning outcome. We think that the use of narrative technologies is going to be applied to more and more fields in the near future and we hope this work will help to drive more research into this direction.

7. Published Papers

7.1. Journals as First Author

Alvarez, N., Prendinger, H., Cavazza, M. Shigematsu, M., Sanchez, A., "Narrative Balance Management in an Intelligent Biosafety Training Application for Improving User Interest." *International Journal of Artificial Intelligence in Education* (2014).

7.2. Journals as Co-Author

Prendinger, H., Alvarez, N., Sanchez-Ruiz, A., Cavazza, Catarino, J., Oliveira, J., Prada, R., Fujimoto, S., Shigematsu, M., "Intelligent Biohazard Training based on Real-time Task Recognition". Under re-submission in *ACM Transactions on Intelligent Systems and Technology (TIST)* (2014).

7.3. Conferences as First Author

Alvarez, N., and Federico P., "Exploring body language as narrative interface." Interactive Storytelling. Springer Berlin Heidelberg, 2012. 196-201.

Alvarez, N. and Prendinger H., "An Authoring Tool for modeling the Narrative Flow in a Virtual training Application". CoSeCiVi (Videogames Sciences Spanish Society Congress), 2014. http://gaia.fdi.ucm.es/sites/cosecivi14/es/papers/08.pdf.

References

Aleven, V., Sewall, J., McLaren, B. M., & Koedinger, K. R. (2006, July). Rapid authoring of intelligent tutors for real-world and experimental use. In Advanced Learning Technologies, 2006. Sixth International Conference on (pp. 847-851). IEEE.

Amin-Naseri, M., Guo, E., Gilbert, S., Jackman, J., Hagge, M., Starns, G., & Faidly, L. (2013, July). Authoring a Thermodynamics Cycle Tutor Using GIFT. In AIED 2013 Workshops Proceedings Volume 7 (p. 45).

Amir, O. and Gal, Y. 2011. Plan recognition in virtual laboratories. In Proceedings Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI'11). 2392–2397.

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.

Bailey, P. (1999). Searching for storiness: Story-generation from a reader's perspective. In Working notes of the Narrative Intelligence Symposium (pp. 157-164).

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.

Bowman, D. A. and Hodges, L. F. (1997). An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In Proceedings of the 1997 symposium on Interactive 3D graphics (I3D '97). 35–38.

Bradbrook, K., Winstanley, G., Glasspool, D., Fox, J. and Griffiths, R. (2005). AI planning technology as a component of computerised clinical practice guidelines. In Proceedings of the 10th conference on Artificial Intelligence in Medicine (AIME'05). Springer-Verlag, Berlin, Heidelberg, 171–180.

Brooke, J. (1996). SUS: A quick and dirty usability scale. In Usability evaluation in industry, P. W. Jordan, B. Weerdmeester, A. Thomas, and I. L. Mclelland (Eds.). Taylor and Francis, London.

Cablé, B., Guin, N., & Lefevre, M. (2013). An Authoring Tool for Semi-automatic Generation of Self-assessment Exercises. In Artificial Intelligence in Education (pp. 679-682). Springer Berlin Heidelberg.

Cabral, M. C., Morimoto, C. H. and Zuffo, M. K. (2005). On the usability of gesture interfaces in virtual environments. In Proceedings of the 2005 Latin American Conference on Human-Computer Interaction (CLIHC'05). ACM Press, 100–108.

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.

Caon, M., Yue, Y., Tscherrig, J., Mugellini, E., & Abou Khaled, O. (2011). Context-aware 3d gesture interaction based on multiple kinects. In AMBIENT 2011, The First International Conference on Ambient Computing, Applications, Services and Technologies (pp. 7-12).

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.

Cavazza, M., Hartley, S., Lugrin, J.-L., Bras, and Le, M. (2004). Qualitative physics in virtual environments. In Proceedings of the 2004 International Conference on Intelligent User Interfaces (Virtual environments & stories). 54–61. http://doi.acm.org/10.1145/964442.964454

Cavazza, M. and Simo, A. (2003). A virtual patient based on qualitative simulation. In IUI 2003. ACM, 19–25. http://doi.acm.org/10.1145/604045.604053

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.

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

Cohn, M. (2005). Agile Estimating and Planning. Prentice Hall PTR, Upper Saddle River, NJ, USA.
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.
Currie, K. and Tate, A. (1991). O-Plan: The open Planning Architecture. Artif. Intell. 52, 1 (1991), 49–86.
Floryan, M., & Woolf, B. P. (2013, January). Authoring Expert Knowledge Bases for Intelligent Tutors through Crowdsourcing. In Artificial Intelligence in Education (pp. 640-643). Springer Berlin Heidelberg.
Erol, K., Hendler, J. A., and Nau, D. S. (1994). UMCP: A Sound and Complete Procedure for Hierarchical Task-Network Planning. In AIPS. 249–254.
Floryan, M., & Woolf, B. P. (2013, January). Authoring Expert Knowledge Bases for Intelligent Tutors through Crowdsourcing. In Artificial Intelligence in Education (pp. 640-643). Springer Berlin Heidelberg.
Franklin, D., Budzik, J., and Hammond, K. J. (2002). Plan-based interfaces: Keeping track of user tasks and acting to cooperate. In IUI. 79–86.
Gal, Y., Reddy, S., Shieber, S. M., Rubin, A., and Grosz, B. J. (2012). Plan recognition in exploratory domains. Artif. Intell. 176, 1 (2012), 2270–2290.
Gaudioso, J., Gribble, L. A., and Salerno, R. M. (2009). Biosecurity: progress and challenges. Journal of the Association for Laboratory Automation 14, 3 (2009), 141–147.
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. (2004). Authoring branching storylines for training applications. In Proceedings of the 6th international conference on Learning sciences (pp. 230-237). International Society of the Learning Sciences.
Gutierrez, T., Rodriguez, J., Velaz, Y., Casado, S., Cruces, A. S., and Sanchez, E. J. (2010). IMA-VR: A multimodal virtual training system for skills transfer in Industrial Maintenance and Assembly tasks.. In RO-MAN, Carlo Alberto Avizzano and Emanuele Ruffaldi (Eds.). IEEE, 428–433.
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 Interactive Storytelling. Springer Berlin Heidelberg.
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-Verlag.
Hasanuzzaman, M., Zhang, T., Ampornaramveth V., Gotoda, H., Shirai, Y., and Ueno, H.. (2007). Adaptive visual gesture recognition for human-robot interaction using a knowledge-based software platform. Robot. Auton. Syst. 55, 8 (Aug. 2007), 643–657.
Hwang, W. and Salvendy, G. (2010). Number of People Required for Usability Evaluation: The 10 _ 2 Rule. Commun. ACM 53, 5 (2010), 130–133.
Isotani, S., Mizoguchi, R., Isotani, S., Capeli, O. M., Isotani, N., & de Albuquerque, A. R. (2010, January). An authoring tool to support the design and use of theory-based collaborative learning activities. In Intelligent Tutoring Systems (pp. 92-102). Springer Berlin Heidelberg.
Jee, H. K., Lim, S., Youn, J., & Lee, J. (2014). An augmented reality-based authoring tool for E-learning applications. Multimedia Tools and Applications, 68(2), 225-235.
Johnson, W. L., & Rickel, J. (1997). Steve: an animated pedagogical agent for procedural training in virtual environments. SIGART Bull. 8, 1-4, 16–21.
Kallmann, M. & Thalmann, D. (2002). Modeling Behaviors of Interactive Objects for Real-Time Virtual Environments. J. Vis. Lang. Comput., 13(2):177-195.
Karampiperis, P., Mouchakis, G., Paliouras, G. and Karkaletsis, V. 2011. ER designer toolkit: a graphical event definition authoring tool. In Proceedings of the 4th International Conference on PErvasive Technologies Related to Assistive Environments (PETRA '11). ACM, New York, NY, USA, , Article 34 , 5 pages.

Kristensson, P. O., Nicholson, T., and Quigley, A. (2012). Continuous recognition of one-handed and two-handed gestures using 3D full-body motion tracking sensors. In Proceedings of the 2012 ACM international conference on Intelligent User Interfaces (IUI '12). 89–92.

Lalys, F., Riffaud, L., Bouget, D., and Jannin, P. (2012). A Framework for the Recognition of High-Level Surgical Tasks From Video Images for Cataract Surgeries. IEEE Trans. Biomed. Engineering 59, 4 (2012), 966–976.

Latoschik, M. E., and Fröhlich, C. (2007). Towards intelligent VR - multi-layered semantic reflection for intelligent virtual environments. In GRAPP (AS/IE). 249–260.

Lang, R. (1999). A declarative model for simple narratives. In Proceedings of the AAAI fall symposium on narrative intelligence (pp. 134-141).

Lebeau, J. F., Paquette, L., Fortin, M., & Mayers, A. (2010, January). An authoring language as a key to usability in a problem-solving ITS framework. In Intelligent Tutoring Systems (pp. 236-238). Springer Berlin Heidelberg.

Lu, G., Shark, L.-K., Hall, G., and Zeshan, U. (2012). Immersive manipulation of virtual objects through glove-based hand gesture interaction. Virtual Reality 16, 3 (2012), 243–252.

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.

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.

Mollet, N., Arnaldi, B. (2006). Storytelling in Virtual Reality for Training. Edutainment 2006:334-347.

Muñoz-Avila, H., Aha, D. W., Nau, D. S., Weber, R., Breslow, L., and Yaman, F. (2001). SiN: Integrating Case-based Reasoning with Task Decomposition. In IJCAI, Bernhard Nebel (Ed.). Morgan Kaufmann, 999–1004.

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

Nau, D. S., Ghallab, M., and Traverso, P. (2004). Automated Planning: Theory & Practice. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Nau, D. S., Smith, S. J. J., and Erol, K. (1998). Control Strategies in HTN Planning: Theory Versus Practice. In AAAI/IAAI, Jack Mostow and Chuck Rich (Eds.). AAAI Press / The MIT Press, 1127–1133.

Oikonomidis, I., Kyriazis, N., and Argyros, A. (2011). Efficient model-based 3D tracking of hand articulations using Kinect. In Proceedings of the British Machine Vision Conference. BMVA Press, 101.1–101.11.

Olsen, J. K., Belenky, D. M., Aleven, V., & Rummel, N. (2013, January). Intelligent Tutoring Systems for Collaborative Learning: Enhancements to Authoring Tools. In Artificial Intelligence in Education (pp. 900-903). Springer Berlin Heidelberg.

Olsen, J. K., Belenky, D. M., Aleven, V., Rummel, N., & Ringenberg, M. (2013, June). 1 Authoring Collaborative Intelligent Tutoring Systems. In AIED 2013 Workshops Proceedings (p. 1).

Paquette, L., Lebeau, J. F., & Mayers, A. (2010). Authoring Problem-Solving Tutors: A Comparison between ASTUS and CTAT. In Advances in Intelligent Tutoring Systems (pp. 377-405). Springer Berlin Heidelberg.

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 7th International Workshop on Immersive Projection Technology, 9th Eurographics Workshop on Virtual Enviroments, A. Kunz and J. Deisinger, Eds., Eurographics Association (Zurich, Switzerland), 097–106.

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-Verlag.

Rao, J. E. (2011). United States Department of State's Biosecurity Engagement Program: Bio Threat Reduction Through International Partnerships. Encyclopedia of Bioterrorism Defense (2011).

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. Orlando, FL.

Ray, C., & Gilbert, S. (2013, July). Bringing Authoring Tools for Intelligent Tutoring Systems and Serious Games Closer Together: Integrating GIFT with the Unity Game Engine. In AIED 2013 Workshops Proceedings Volume 7 (p. 37).

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

Rizzo, A., Parsons, T., Buckwalter, G. and 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-Verlag.

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. J. 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 & 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.

Scarlatos, L. and Friedman, R. (2007). On developing user interfaces for children in educational virtual reality systems. Technical Report. CUNY Graduate Center Technical Report TR-2007001.

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.

Shaker, N.; Shaker, M.; Togelius, J.. Ropossum: An Authoring Tool for Designing, Optimizing and Solving Cut the Rope Levels. AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, North America, nov. 2013.

Schneider, M. (2010). Resource-aware plan recognition in instrumented environments. Ph.D. Dissertation.

Shahar, Y., Miksch, S. and Johnson, P. D. (1998). The Asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines. Artificial Intelligence in Medicine 14, 1-2 (1998), 29–51.

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 (ed.), International Conference on Virtual Storytelling, Vol. 2897 (pp. 65-73). Springer. ISBN: 3-540-20535-7.

Song, Y, Demirdjian, D. and Davis, R. (2012). Continuous body and hand gesture recognition for natural human-computer interaction. ACM Transactions on Interactive Intelligent Systems 2, 1 (2012), 1–28.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S., & and Holden, H.K. (2012). The Generalized Intelligent Framework for Tutoring (GIFT). Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

Sottilare, R. A., & Holden, H. K. (2013, July). Motivations for a Generalized Intelligent Framework for Tutoring (GIFT) for Authoring, Instruction and Analysis. In AIED 2013 Workshops Proceedings Volume 7 (p. 1).

Sridhar, A. and Sowmya, A. (2008). Multiple Camera, Multiple Person Tracking with Pointing Gesture Recognition in Immersive Environments. In Advances in Visual Computing, George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Paolo Remagnino, Fatih Porikli, Jrg Peters, James Klosowski, Laura Arns, YuKa Chun, Theresa-Marie Rhyne, and Laura Monroe (Eds.). Lecture Notes in Computer Science, Vol. 5358. Springer Berlin Heidelberg, 508–519.

Stocker, C., Sunshine-Hill, B., Drake, J., Perera, I., Kider, J. J. T. and Badler, N. I. (2011). CRAM It! A Comparison of Virtual, Live-Action and Written Training Systems for Preparing Personnel to Work in Hazardous Environments. (2011), 95–102.

Szilas, N. (1999, November). Interactive drama on computer: beyond linear narrative. In Proceedings of the AAAI fall symposium on narrative intelligence (pp. 150-156).

Tahboub, K. A. (2006). Intelligent Human-Machine Interaction Based on Dynamic Bayesian Networks Probabilistic Intention Recognition. Journal of Intelligent and Robotic Systems 45, 1 (2006), 31–52.

Tripoliti, E.E.; Pappas, I.G.; Petrakis, E.G.M.; Sans, J.M., "Crafting vascular medicine training scenarios: The RT3S authoring tool," Bioinformatics and Bioengineering (BIBE), 2013 IEEE 13th International Conference on , vol., no., pp.1,4, 10-13 Nov. 2013

Unzueta, L., Mena, O., Sierra, B. and Suescun, A. (2008). Kinetic Pseudo-energy History for Human Dynamic Gestures Recognition.. In AMDO (2008-07-10) (Lecture Notes in Computer Science), Francisco J. Perales Lpez and Robert B. Fisher (Eds.), Vol. 5098. Springer, 390–399.

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.

Vidani, A. C. and Chittaro, L. (2009). Using a Task Modeling Formalism in the Design of Serious Games for Emergency Medical Procedures. In VS-GAMES, Genaro Rebolledo-Mendez, Fotis Liarokapis, and Sara de Freitas (Eds.). IEEE Computer Society, 95–102.

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 (ed.), ISAmI , Vol. 92 (pp. 59-67). Springer. ISBN: 978-3-642-19936-3.

Ware, S. & Young, R. (2010). Modeling Narrative Conflict to Generate Interesting Stories. Artificial Intelligence and Interactive Digital Entertainment Conference, North America.

Wiebusch, D. and Latoschik, M. E. (2012). Enhanced decoupling of components in intelligent realtime interactive systems using ontologies. In SEARIS. 43–51.

Wilkins, D. and desJardins, M. (2001). A Call for Knowledge-Based Planning. AI Magazine 22, 1 (2001), 99–115.

Witmer, B. G. and Singer, M. J. (1998). Measuring Presence in Virtual Environments: A Presence Questionnaire. Presence 7, 3 (1998), 225–240.

Young, R. M. (1999). Notes on the use of plan structures in the creation of interactive plot. In AAAI Fall Symposium on Narrative Intelligence (pp. 164-167).

Zhang, Z.: Microsoft Kinect Sensor and Its Effect. IEEE Multimedia 19 (2), pp. 4-10 (2012)

Appendix A

A.1 DESCRIPTION OF THE XML NOTATION USED FOR THE TASK TREES

Element <model>:

- Description: Declares a model for using a set of task trees.
- Parameters:
  - Id: identifier of the model.
  - Description: description of the model.
- Child elements: <task>.
- Parent elements: none.

Element <task>:

- Description: Declares a task tree.
- Parameters:
  - Id: identifier of the task.
  - Description: description of the task.
  - Warning: if the task is an error task, this parameter contains a warning message.
  - Status: for intern use, its value has to be set to "not".
  - Balance: Narrative parameter. Value of the difficulty of solving this task.
  - Resolution: Narrative parameter. Value of the narrative impact of this event.
  - Closure: If this event cannot happen after some other event is done, this parameter contains the id value of that event.
  - Type: if this event can be done by the user, it has the "user" value, if not, it has the "system" value.
- Child elements: <and>, <or>, <seq>.
- Parent elements: <model>, <and>, <or>, <seq>.

Element <and>:

- Description: Declares an "and" subgroup of tasks.
- Parameters: none.
- Child elements: <task>.
- Parent elements: <task>.

Element <or>:

- Description: Declares an "or" subgroup of tasks.
- Parameters: none.
- Child elements: <task>.
- Parent elements: <task>.

Element <seq>:

- Description: Declares an "seq" subgroup of tasks.
- Parameters: none.
- Child elements: <task>.
- Parent elements: <task>.

A.2 TASK TREES USED IN BIO-SAFETY LAB

File 1: Task Tree used for the Spill Cleaning protocol (XML format).

```
<model id="SPILL" description="Contaminated Blood Spill Treatment">
  <task id="CONTAMINATION" description="Contaminated Blood Spill Treatment" warning="" status="not"
  balance="0" resolution="0" closure="" type="user">
    <and>
      <task id="USERPOSITION" description="User's Correct Position" warning="" status="not" balance="0"
      resolution="0" closure="" type="user">
        <or>
          <task id="INTERPOSING" description="User is Interposing" warning="Don't interpose between
          the spill and the air vents, is dangerous if you breath the gases" status="not" balance="0"
          resolution="0" closure="" type="user"/>
          <task id="NOTINTERPOSING" description="Don't interpose between the spill and the vents"
          warning="" status="not" balance="0" resolution="0" closure="" type="user"/>
        </or>
      </task>
      <task id="TREATSPILL" description="clean the Spill" warning="" status="not" balance="0" resolution="0"
      closure="" type="user">
        <seq>
          <task id="BORDERSPILL" description="contain the spill in its borders" warning="" status="not"
          balance="0" resolution="2" closure="" type="user">
            <and>
              <task id="FIRSTBORDER" description="contain the spill with something in the first border"
              warning="" status="not" balance="1" resolution="0" closure="" type="user">
                <or>
                  <task id="PLACESPONGEBORDER" description="place an absorbent sponge over
                  the spill border" warning="" status="not" balance="0" resolution="0" closure=""
                  type="user"/>
                  <task id="PLACEPADBORDER" description="user places absorbent pad over the
                  spill" warning="Using an absorbent pad for border won't be able to retain the spill"
                  status="not" balance="-1" resolution="0" closure="" type="user"/>
                  <task id="PLACEPAPERBORDER" description="user places absorbent paper over
                  the spill" warning="Using an absorbent paper for border won't be able to retain the
                  spill" status="not" balance="-1" resolution="0" closure="" type="user"/>
                  <task id="PLACEOBJECTBORDER" description="user places object over the spill"
                  warning="Using a non absorbent object for border won't be able to retain the spill"
                  status="not" balance="-1" resolution="0" closure="" type="user"/>
                </or>
              </task>
              <task id="SECONDBORDER" description="contain the spill with something in the second
              border" warning="" status="not" balance="0" resolution="0" closure="" type="user">
                <or>
                  <task id="PLACESPONGEBORDER" description="place an absorbent sponge over
                  the spill border" warning="" status="not" balance="0" resolution="0" closure=""
                  type="user"/>
                  <task id="PLACEPADBORDER" description="user places absorbent pad over the
                  spill" warning="Using an absorbent pad for border won't be able to retain the spill"
                  status="not" balance="-2" resolution="0" closure="" type="user"/>
                  <task id="PLACEPAPERBORDER" description="user places absorbent paper over the
                  spill" warning="Using an absorbent paper for border won't be able to retain the spill"
                  status="not" balance="-2" resolution="0" closure="" type="user"/>
```

```
                    <task id="PLACEOBJECTBORDER" description="user places object over the spill"
                    warning="Using a non absorbent object for border won't be able to retain the spill"
                    status="not" balance="-2" resolution="0" closure="" type="user"/>
                  </or>
                </task>
                <task id="THIRDBORDER" description="contain the spill with something in the third border"
              warning="" status="not" balance="0" resolution="0" closure="" type="user">
                  <or>
                    <task id="PLACESPONGEBORDER" description="place an absorbent sponge over
                    the spill border" warning="" status="not" balance="0" resolution="0" closure=""
                    type="user"/>
                    <task id="PLACEPADBORDER" description="user places absorbent pad over the
                    spill" warning="Using an absorbent pad for border won't be able to retain the spill"
                    status="not" balance="-2" resolution="0" closure="" type="user"/>
                    <task id="PLACEPAPERBORDER" description="user places absorbent paper over
                    the spill" warning="Using an absorbent paper for border won't be able to retain the
                    spill" status="not" balance="-2" resolution="0" closure="" type="user"/>
                    <task id="PLACEOBJECTBORDER" description="user places object over the spill"
                    warning="Using a non absorbent object for border won't be able to retain the spill"
                    status="not" balance="-3" resolution="0" closure="" type="user"/>
                  </or>
                </task>
                <task id="FOURTHBORDER" description="contain the spill with something in the fourth
              border" warning="" status="not" balance="0" resolution="0" closure="" type="user">
                    <or>
                    <task id="PLACESPONGEBORDER" description="place an absorbent sponge over
                    the spill border" warning="" status="not" balance="0" resolution="0" closure=""
                    type="user"/>
                    <task id="PLACEPADBORDER" description="user places absorbent pad over the
                    spill" warning="Using an absorbent pad for border won't be able to retain the spill"
                    status="not" balance="-2" resolution="0" closure="" type="user"/>
                    <task id="PLACEPAPERBORDER" description="user places absorbent paper over
                    the spill" warning="Using an absorbent paper for border won't be able to retain the
                    spill" status="not" balance="-2" resolution="0" closure="" type="user"/>
                    <task id="PLACEOBJECTBORDER" description="user places object over the spill"
                    warning="Using a non absorbent object for border won't be able to retain the spill"
                    status="not" balance="-3" resolution="0" closure="" type="user"/>
                  </or>
                </task>
              </and>
            </task>
            <task id="COVERSPILL" description="cover the spill for absorbing" warning="" status="not"
          balance="0" resolution="0" closure="" type="user">
              <seq>
                <task id="COVERBORDERS" description="cover the spill borders" warning="" status="not"
                balance="0" resolution="2" closure="" type="user">
                  <and>
                    <task id="COVERFIRSTBORDER" description="User deals with the first border"
                    warning="" status="not" balance="0" resolution="1" closure="" type="user">
                      <or>
                        <task id="PLACESPONGEBORDERFORCOVER" description="user places
                        absorbent sponge over the spill for cover" warning="Using an absorbent
                        sponge won't absorb well the spill" status="not" balance="-1" resolution="0"
                        closure="" type="user"/>
                        <task id="PLACEPADBORDERFORCOVER" description="user places
                        absorbent pad over the spill border for cover" warning="Using an absorbent
```

```
                pad won't absorb well the spill" status="not" balance="-1" resolution="0"
                closure="" type="user"/>
                <task id="PLACEPAPERBORDERFORCOVER" description="places an
                absorbent paper over the spill border for covering it" warning="" status="not"
                balance="0" resolution="0" closure="" type="user"/>
                <task id="PLACEOBJECTBORDERFORCOVER" description="user places
                object over the spill for cover" warning="Using an non absorbent object won't
                absorb the spill" status="not" balance="-2" resolution="0" closure=""
                type="user"/>
              </or>
    </task>
    <task id="COVERSECONDBORDER" description="User deals with the second
border" warning="" status="not" balance="0" resolution="0" closure="" type="user">
          <or>
            <task id="PLACESPONGEBORDERFORCOVER" description="user places
            absorbent sponge over the spill for cover" warning="Using an absorbent
            sponge won't absorb well the spill" status="not" balance="-1" resolution="0"
            closure="" type="user"/>
            <task id="PLACEPADBORDERFORCOVER" description="user places
            absorbent pad over the spill for cover" warning="Using an absorbent pad
            won't absorb well the spill" status="not" balance="-1" resolution="0"
            closure="" type="user"/>
            <task id="PLACEPAPERBORDERFORCOVER" description="places an
            absorbent paper over the spill border for covering it" warning="" status="not"
            balance="0" resolution="0" closure="" type="user"/>
            <task id="PLACEOBJECTBORDERFORCOVER" description="user places
            object over the spill for cover" warning="Using an non absorbent object won't
            absorb the spill" status="not" balance="-2" resolution="0" closure=""
            type="user"/>
          </or>
    </task>
    <task id="COVERTHIRDBORDER" description="User deals with the third border"
warning="" status="not" balance="0" resolution="0" closure="" type="user">
          <or>
            <task id="PLACESPONGEBORDERFORCOVER" description="user places
            absorbent sponge over the spill for cover" warning="Using an absorbent
            sponge won't absorb well the spill" status="not" balance="-1" resolution="0"
            closure="" type="user"/>
            <task id="PLACEPADBORDERFORCOVER" description="user places
            absorbent pad over the spill for cover" warning="Using an absorbent pad
            won't absorb well the spill" status="not" balance="-1" resolution="0"
            closure="" type="user"/>
            <task id="PLACEPAPERBORDERFORCOVER" description="places an
            absorbent paper over the spill border for covering it" warning="" status="not"
            balance="0" resolution="0" closure="" type="user"/>
            <task id="PLACEOBJECTBORDERFORCOVER" description="user places
            object over the spill for cover" warning="Using an non absorbent object won't
            absorb the spill" status="not" balance="-2" resolution="0" closure=""
            type="user"/>
          </or>
    </task>
    <task id="COVERFOURTHBORDER" description="User deals with the fourth
border" warning="" status="not" balance="0" resolution="0" closure="" type="user">
          <or>
            <task id="PLACESPONGEBORDERFORCOVER" description="user places
            absorbent sponge over the spill for cover" warning="Using an absorbent
```

```
                                sponge won't absorb well the spill" status="not" balance="-1" resolution="0"
                                closure="" type="user"/>
                                <task id="PLACEPADBORDERFORCOVER" description="user places
                                absorbent pad over the spill for cover" warning="Using an absorbent pad
                                won't absorb well the spill" status="not" balance="-1" resolution="0"
                                closure="" type="user"/>
                                <task id="PLACEPAPERBORDERFORCOVER" description="places an
                                absorbent paper over the spill border for covering it" warning="" status="not"
                                balance="0" resolution="0" closure="" type="user"/>
                                <task id="PLACEOBJECTBORDERFORCOVER" description="user places
                                object over the spill for cover" warning="Using an non absorbent object won't
                                absorb the spill" status="not" balance="-2" resolution="0" closure=""
                                type="user"/>
                            </or>
                        </task>
                    </and>
                </task>
                <task id="COVERCENTER" description="cover the spill center" warning="" status="not"
            balance="0" resolution="1" closure="" type="user">
                    <or>
                        <task id="PLACESPONGECENTER" description="user places absorbent sponge
                        over the spill" warning="Using an absorbent sponge won't absorb well the spill"
                        status="not" balance="-1" resolution="0" closure="" type="user"/>
                        <task id="PLACEPADCENTER" description="user places absorbent pad over the
                        spill" warning="Using an absorbent pad won't absorb well the spill" status="not"
                        balance="-1" resolution="0" closure="" type="user"/>
                        <task id="PLACEPAPERCENTER" description="places an absorbent paper over the
                        spill center for covering it" warning="" status="not" balance="0" resolution="0"
                        closure="" type="user"/>
                        <task id="PLACEOBJECTCENTER" description="user places object over the spill"
                        warning="Using an non absorbent object won't absorb the spill" status="not"
                        balance="-2" resolution="0" closure="" type="user"/>
                    </or>
                </task>
            </seq>
        </task>
        <task id="DISINFECTSPILL" description="disinfect the spill" warning="" status="not" balance="0"
    resolution="0" closure="" type="user">
            <seq>
                <task id="DISINFECTBORDERS" description="disinfect the spill covers in the border"
                warning="" status="not" balance="0" resolution="0" closure="" type="user">
                    <and>
                        <task id="FIRSTCOVER" description="User disinfects the fist spill cover" warning=""
                        status="not" balance="1" resolution="0" closure="" type="user">
                            <or>
                                <task id="PLACEETHANOLBORDER" description="spray ethanol over the
                                spill in circles" warning="" status="not" balance="0" resolution="0" closure=""
                                type="user"/>
                                <task id="PLACEISOPROPANOLBORDER" description="spray isopropanol
                                over the spill in circles" warning="" status="not" balance="0" resolution="0"
                                closure="" type="user"/>
                                <task id="PLACEBLEACHBORDER" description="user places bleach over the
                                cover" warning="Using bleach won't disinfect the spill" status="not" balance="-
                                1" resolution="0" closure="" type="user"/>
```

```
            <task id="PLACEDETERGENTBORDER" description="user places detergent
            over the cover" warning="Using detergent won't disinfect the spill" status="not"
            balance="-1" resolution="0" closure="" type="user"/>
            <task id="PLACEOBJECTBORDERDISINF" description="user places object
            over the cover" warning="Putting an object over the spill won't disinfect it"
            status="not" balance="-2" resolution="0" closure="" type="user"/>
        </or>
    </task>
    <task id="SECONDCOVER" description="User disinfects the second spill cover"
    warning="" status="not" balance="0" resolution="0" closure="" type="user">
        <or>
            <task id="PLACEETHANOLBORDER" description="spray ethanol over the
            spill in circles" warning="" status="not" balance="0" resolution="0" closure=""
            type="user"/>
            <task id="PLACEISOPROPANOLBORDER" description="spray isopropanol
            over the spill in circles" warning="" status="not" balance="0" resolution="0"
            closure="" type="user"/>
            <task id="PLACEBLEACHBORDER" description="user places bleach over the
            cover" warning="Using bleach won't disinfect the spill" status="not"
            balance="0" resolution="0" closure="" type="user"/>
            <task id="PLACEDETERGENTBORDER" description="user places detergent
            over the cover" warning="Using detergent won't disinfect the spill" status="not"
            balance="0" resolution="0" closure="" type="user"/>
            <task id="PLACEOBJECTBORDERDISINF" description="user places object
            over the cover" warning="Putting an object over the spill won't disinfect it"
            status="not" balance="0" resolution="0" closure="" type="user"/>
        </or>
    </task>
    <task id="THIRDCOVER" description="User disinfects the third spill cover" warning=""
    status="not" balance="0" resolution="0" closure="" type="user">
        <or>
            <task id="PLACEETHANOLBORDER" description="spray ethanol over the spill
            in circles" warning="" status="not" balance="0" resolution="0" closure=""
            type="user"/>
            <task id="PLACEISOPROPANOLBORDER" description="spray isopropanol
            over the spill in circles" warning="" status="not" balance="0" resolution="0"
            closure="" type="user"/>
            <task id="PLACEBLEACHBORDER" description="user places bleach over the
            cover" warning="Using bleach won't disinfect the spill" status="not"
            balance="0" resolution="0" closure="" type="user"/>
            <task id="PLACEDETERGENTBORDER" description="user places detergent
            over the cover" warning="Using detergent won't disinfect the spill" status="not"
            balance="0" resolution="0" closure="" type="user"/>
            <task id="PLACEOBJECTBORDERDISINF" description="user places object
            over the cover" warning="Putting an object over the spill won't disinfect it"
            status="not" balance="0" resolution="0" closure="" type="user"/>
        </or>
    </task>
    <task id="FOURTHCOVER" description="User disinfects the fourth spill cover"
    warning="" status="not" balance="0" resolution="0" closure="" type="user">
        <or>
            <task id="PLACEETHANOLBORDER" description="spray ethanol over the spill
            in circles" warning="" status="not" balance="0" resolution="0" closure=""
            type="user"/>
```

```
                        <task id="PLACEISOPROPANOLBORDER" description="spray isopropanol
                        over the spill in circles" warning="" status="not" balance="0" resolution="0"
                        closure="" type="user"/>
                        <task id="PLACEBLEACHBORDER" description="user places bleach over the
                        cover" warning="Using bleach won't disinfect the spill" status="not"
                        balance="0" resolution="0" closure="" type="user"/>
                        <task id="PLACEDETERGENTBORDER" description="user places detergent
                        over the cover" warning="Using detergent won't disinfect the spill" status="not"
                        balance="0" resolution="0" closure="" type="user"/>
                        <task id="PLACEOBJECTBORDERDISINF" description="user places object
                        over the cover" warning="Putting an object over the spill won't disinfect it"
                        status="not" balance="0" resolution="0" closure="" type="user"/>
                    </or>
                </task>
            </and>
        </task>
        <task id="DISINFECTCENTER" description="User disinfects the spill cover in the center"
        warning="" status="not" balance="0" resolution="2" closure="" type="user">
            <or>
                <task id="PLACEETHANOLCENTER" description="spray ethanol over the spill center"
                warning="" status="not" balance="0" resolution="0" closure="" type="user"/>
                <task id="PLACEISOPROPANOLCENTER" description="spray isopropanol over the
                spill center" warning="" status="not" balance="0" resolution="0" closure=""
                type="user"/>
                <task id="PLACEBLEACHCENTER" description="user places bleach over the cover"
                warning="Using bleach won't disinfect the spill" status="not" balance="0"
                resolution="0" closure="" type="user"/>
                <task id="PLACEDETERGENTCENTER" description="user places detergent over the
                cover" warning="Using detergent won't disinfect the spill" status="not" balance="0"
                resolution="0" closure="" type="user"/>
                <task id="PLACEOBJECTCENTERDISINF" description="user places object over the
                cover" warning="Putting an object over the spill won't disinfect it" status="not"
                balance="0" resolution="0" closure="" type="user"/>
            </or>
        </task>
        <task id="WAITEFFECT" description="wait for the disinfectant to take effect" warning=""
        status="not" balance="0" resolution="1" closure="" type="user"/>
    </seq>
</task>
<task id="REMOVESPILL" description="User removes the spill" warning="" status="not" balance="0"
resolution="0" closure="" type="user">
    <seq>
        <task id="MERGEWASTE" description="merge the waste" warning="" status="not"
        balance="0" resolution="2" closure="" type="user">
            <and>
                <task id="MERGEBORDERS" description="User merges the borders" warning=""
                status="not" balance="0" resolution="0" closure="" type="user">
                    <and>
                        <task id="MERGEFIRST" description="User merges the first border"
                        warning="" status="not" balance="2" resolution="0" closure="" type="user">
                            <or>
                                <task id="DROPCOVERBORDERCENTER" description="drop one
                                border in the center of the spill" warning="" status="not" balance="0"
                                resolution="0" closure="" type="user"/>
                                <task id="DROPCOVERBORDERELSE" description="User drops the
                                border elsewhere" warning="Dispersing the waste can disperse part of
```

```
                    the spill" status="not" balance="-1" resolution="0" closure=""
                    type="user"/>
                </or>
            </task>
            <task id="MERGESECOND" description="User merges the second border"
            warning="" status="not" balance="0" resolution="0" closure="" type="user">
                <or>
                    <task id="DROPCOVERBORDERCENTER" description="drop other
                    border in the center of the spill" warning="" status="not" balance="0"
                    resolution="0" closure="" type="user"/>
                    <task id="DROPCOVERBORDERELSE" description="User drops the
                    border elsewhere" warning="Dispersing the waste can disperse part of
                    the spill" status="not" balance="-1" resolution="0" closure=""
                    type="user"/>
                </or>
            </task>
            <task id="MERGETHIRD" description="User merges the third border"
            warning="" status="not" balance="0" resolution="0" closure="" type="user">
                <or>
                    <task id="DROPCOVERBORDERCENTER" description="drop other
                    border in the center of the spill" warning="" status="not" balance="0"
                    resolution="0" closure="" type="user"/>
                    <task id="DROPCOVERBORDERELSE" description="User drops the
                    border elsewhere" warning="Dispersing the waste can disperse part of
                    the spill" status="not" balance="-1" resolution="0" closure=""
                    type="user"/>
                </or>
            </task>
        <task id="MERGEFOURTH" description="User merges the first border" warning=""
        status="not" balance="0" resolution="0" closure="" type="user">
            <or>
                <task id="DROPCOVERBORDERCENTER" description="drop the last
                border in the center of the spill" warning="" status="not" balance="0"
                resolution="0" closure="" type="user"/>
                <task id="DROPCOVERBORDERELSE" description="User drops the
                border elsewhere" warning="Dispersing the waste can disperse part of the
                spill" status="not" balance="-1" resolution="0" closure="" type="user"/>
            </or>
        </task>
    </and>
</task>
<task id="MERGECOVERS" description="User merges the covers" warning="" status="not"
balance="0" resolution="1" closure="" type="user">
    <and>
        <task id="MERGEFIRST" description="User merges the first cover" warning=""
        status="not" balance="1" resolution="0" closure="" type="user">
            <or>
                <task id="DROPCOVERBORDERCENTER" description="drop one cover
                from the border in the center of the spill" warning="" status="not" balance="0"
                resolution="0" closure="" type="user"/>
                <task id="DROPCOVERBORDERELSE" description="User drops the cover
                from the border elsewhere" warning="Dispersing the waste can disperse part
                of the spill" status="not" balance="-1" resolution="0" closure="" type="user"/>
            </or>
        </task>
```

```
                    <task id="MERGESECOND" description="User merges the second cover"
                warning="" status="not" balance="0" resolution="0" closure="" type="user">
                        <or>
                            <task id="DROPCOVERBORDERCENTER" description="drop another
                            cover from the border in the center of the spill" warning="" status="not"
                            balance="0" resolution="0" closure="" type="user"/>
                            <task id="DROPCOVERBORDERELSE" description="User drops the cover
                            from the border elsewhere" warning="Dispersing the waste can disperse part
                            of the spill" status="not" balance="-1" resolution="0" closure="" type="user"/>
                        </or>
                    </task>
                    <task id="MERGETHIRD" description="User merges the third cover" warning=""
                status="not" balance="0" resolution="0" closure="" type="user">
                        <or>
                            <task id="DROPCOVERBORDERCENTER" description="drop another
                            cover from the border in the center of the spill" warning="" status="not"
                            balance="0" resolution="0" closure="" type="user"/>
                            <task id="DROPCOVERBORDERELSE" description="User drops the cover
                            from the border elsewhere" warning="Dispersing the waste can disperse part
                            of the spill" status="not" balance="-1" resolution="0" closure="" type="user"/>
                        </or>
                    </task>
                    <task id="MERGEFOURTH" description="User merges the fourth cover" warning=""
                status="not" balance="0" resolution="0" closure="" type="user">
                        <or>
                            <task id="DROPCOVERBORDERCENTER" description="drop the last
                            cover from the border in the center of the spill" warning="" status="not"
                            balance="0" resolution="0" closure="" type="user"/>
                            <task id="DROPCOVERBORDERELSE" description="User drops the
                            cover from the border elsewhere" warning="Dispersing the waste can
                            disperse part of the spill" status="not" balance="-1" resolution="0" closure=""
                            type="user"/>
                        </or>
                    </task>
                        <task id="LEAVESCENTER" description="User leaves the center cover"
                        warning="" status="not" balance="0" resolution="0" closure="" type="user"/>
                    </and>
                </task>
            </and>
        </task>
        <task id="DISPOSEWASTE" description="User disposes the waste" warning="" status="not"
        balance="0" resolution="0" closure="" type="user">
            <or>
                <task id="INBIOHAZARDBIN" description="dispose the waste in the bio hazard waste bin"
                warning="" status="not" balance="0" resolution="0" closure="" type="user"/>
                <task id="INFLAMMABLEDBIN" description="User disposes the waste in the flammable
                waste bin" warning="Infectious waste should not be thrown in the flammable bin"
                status="not" balance="-1" resolution="0" closure="" type="user"/>
                <task id="INNORMALBIN" description="User disposes the waste in the normal waste bin"
                warning="Infectious waste should not be thrown in a normal bin" status="not" balance="-
                1" resolution="0" closure="" type="user"/>
            </or>
        </task>
    </seq>
</task>
</seq>
```

```
        </task>
      </and>
    </task>
</model>
```

File 2: Task Tree used for the Scenario events (XML format).

```
<model id="SCENARIO" description="Scenario tasks">
    <task id="EVENTS" description="system events" warning="" status="not" balance="0" resolution="0"
    closure="" type="system">
        <and>
            <task id="KNOCKBOTTLE" description="" warning="" status="not" balance="-6" resolution="12"
        closure="" type="system"/>
            <task id="LOSE" description="lose an important object" warning="" status="not" balance="0"
        resolution="0" closure="" type="system">
                <or>
                    <task id="LOSEDISINFECTANT" description="user loses the disinfectant" warning=""
                status="not" balance="0" resolution="0" closure="" type="system">
                        <or>
                            <task id="LOSEETHANOL" description="user loses the ethanol" warning=""
                        status="not" balance="-1" resolution="3" closure="PLACEETHANOLBORDER"
                        type="system"/>
                            <task id="LOSEISOPROPANOL" description="user loses the isopropanol"
                        warning="" status="not" balance="-1" resolution="3"
                        closure="PLACEISOPROPANOLBORDER" type="system"/>
                            <task id="LOSEDETERGENT" description="user loses the detergent" warning=""
                        status="not" balance="1" resolution="1" closure="FIRSTCOVER" type="system"/>
                            <task id="LOSEBLEACH" description="user loses the bleach" warning=""
                        status="not" balance="1" resolution="1" closure="FIRSTCOVER" type="system"/>
                            <task id="LOSEETHANOLFAR" description="user loses the ethanol" warning=""
                        status="not" balance="-2" resolution="3" closure="PLACEETHANOLBORDER"
                        type="system"/>
                            <task id="LOSEISOPROPANOLFAR" description="user loses the isopropanol"
                        warning="" status="not" balance="-2" resolution="3"
                        closure="PLACEISOPROPANOLBORDER" type="system"/>
                            <task id="LOSEDETERGENTFAR" description="user loses the detergent"
                        warning="" status="not" balance="2" resolution="1" closure="FIRSTCOVER"
                        type="system"/>
                            <task id="LOSEBLEACHFAR" description="user loses the bleach" warning=""
                        status="not" balance="2" resolution="1" closure="FIRSTCOVER" type="system"/>
                        </or>
                    </task>
                    <task id="LOSETOOL" description="user loses the disinfectant" warning="" status="not"
                balance="0" resolution="0" closure="" type="system">
                        <or>
                            <task id="LOSEPAPER" description="user loses the absorvent paper" warning=""
                        status="not" balance="-1" resolution="4" closure="COVERSPILL" type="system"/>
                            <task id="LOSESPONGE" description="user loses the absorvent sponge"
                        warning="" status="not" balance="-1" resolution="4" closure="BORDERSPILL"
                        type="system"/>
                            <task id="LOSEPAD" description="user loses the absorvent pad" warning=""
                        status="not" balance="2" resolution="1" closure="COVERSPILL" type="system"/>
                            <task id="LOSEPAPERFAR" description="user loses the absorvent paper"
                        warning="" status="not" balance="-2" resolution="4" closure="COVERSPILL"
                        type="system"/>
                            <task id="LOSESPONGEFAR" description="user loses the absorvent sponge"
                        warning="" status="not" balance="-2" resolution="4" closure="BORDERSPILL"
                        type="system"/>
                            <task id="LOSEPADFAR" description="user loses the absorvent pad" warning=""
                        status="not" balance="3" resolution="1" closure="COVERSPILL" type="system"/>
```

```
                    </or>
                   </task>
                </or>
              </task>
               <task id="OVERFLOWSPILL" description="overflow when bordering" warning="" status="not"
            balance="0" resolution="0" closure="" type="system">
                    <or>
                  <task id="PADOVERFLOW" description="lose an important object" warning="" status="not"
                  balance="0" resolution="0" closure="" type="system">
                      <seq>
                            <task id="PLACEPADBORDER" description="user places absorbent pad over the spill"
                            warning="" status="not" balance="0" resolution="0" closure="" type="user"/>
                            <task id="SPILLOVERFLOW" description="overflow the border" warning="" status="not"
                            balance="-2" resolution="5" closure="BORDERSPILL" type="system"/>
                      </seq>
                  </task>
                  <task id="PAPEROVERFLOW" description="lose an important object" warning="" status="not"
                  balance="0" resolution="0" closure="" type="system">
                      <seq>
                            <task id="PLACEPAPERBORDER" description="user places absorbent paper over the
                            spill" warning="" status="not" balance="0" resolution="0" closure="" type="user"/>
                            <task id="SPILLOVERFLOW" description="overflow the border" warning="" status="not"
                            balance="-2" resolution="5" closure="BORDERSPILL" type="system"/>
                      </seq>
                 </task>
                 <task id="EMPTYOVERFLOW" description="the spill overflows" warning="" status="not" balance="-2"
            resolution="7" closure="BORDERSPILL" type="system"/>
            </or>
     </task>
     <task id="HINTS" description="system help" warning="" status="not" balance="0" resolution="0" closure=""
type="system">
            <and>
                <task id="ARROWHINT" description="show graphic help" warning="" status="not" balance="2"
                resolution="4" closure="" type="system"/>
                <task id="ARROWHINT - EXPLICIT" description="show graphic help (and glow)" warning=""
                status="not" balance="2" resolution="6" closure="" type="system"/>
                <task id="ARROWHINT - TOTAL" description="show graphic help (and glow and arrow)" warning=""
                status="not" balance="2" resolution="8" closure="" type="system"/>
            </and>
       </task>
       <task id="GARBAGEFALLING" description="a piece of waste falls from the garbage heap" warning=""
status="not" balance="0" resolution="0" closure="" type="system">
            <seq>
                <task id="TAKESWASTE" description="User takes the waste" warning="" status="not" balance="0"
            resolution="0" closure="" type="user"/>
                <task id="DROPGARBAGE" description="infect a place with bacteria" warning="" status="not"
            balance="-1" resolution="2" closure="DISPOSEWASTE" type="system"/>
            </seq>
          </task>
      </and>
   </task>
</model>
```

File 3: Knowledge database used in the Common Sense Reasoner (XML format).

```
<commonsense>
	<properties>
		<property name="fragile" type="bool">
		</property>
		<property name="surface" type="bool">
		</property>
		<property name="absorbent" type="bool">
		</property>
		<property name="wears" type="object">
		</property>
		<property name="wearable" type="object">
		</property>
		<property name="container_quantity" type="int">
		</property>
		<property name="contains" type="object">
		</property>
		<property name="infectious_type" type="int">
		</property>
		<property name="disinfectant_type" type="int">
		</property>
		<property name="absorbable" type="int">
		</property>
		<property name="physic_presence" type="bool">
		</property>
		<property name="vapors" type="bool">
		</property>
		<property name="area_of_effect" type="bool">
		</property>
	</properties>
	<objects>
		<object name="table02">
			<properties>
				<property name="surface" value="true">
				</property>
			</properties>
		</object>
		<object name="Bottle 0">
			<properties>
				<property name="fragile" value="true">
				</property>
			</properties>
		</object>
		<object name="BLOODBOTTLE">
			<properties>
				<property name="fragile" value="true">
				</property>
				<property name="container_quantity" value="1">
				</property>
				<property name="contains" value="SPILL">
				</property>
			</properties>
		</object>
		<object name="SPILL">
```

```
		<properties>
			<property name="infectious" value="1">
			</property>
			<property name="absorbable" value="30">
			</property>
			<property name="vapors" type="true">
			</property>
			<property name="area_of_effect" type="true">
			</property>
		</properties>
</object>
<object name="AVATAR">
		<properties>
			<property name="wears" value="nothing">
			</property>
		</properties>
</object>
<object name="CART">
		<properties>
			<property name="physic_presence" value="true">
			</property>
		</properties>
</object>
<object name="ABSORBENTPAPER">
		<properties>
			<property name="absorbent" value="true">
			</property>
		</properties>
</object>
<object name="ABSORBENTPAD">
		<properties>
			<property name="absorbent" value="true">
			</property>
		</properties>
</object>
<object name="ABSORBENTSPONGE">
		<properties>
			<property name="absorbent" value="true">
			</property>
		</properties>
</object>
<object name="ETHANOL">
		<properties>
			<property name="disinfectant_type" value="1">
			</property>
		</properties>
</object>
			<object name="ISOPROPANOL">
		<properties>
			<property name="disinfectant_type" value="1">
			</property>
		</properties>
</object>
<object name="BLEACH">
		<properties>
			<property name="disinfectant_type" value="2">
```

```
                                        </property>
                                </properties>
                        </object>
                        <object name="DETERGENT">
                                <properties>
                                        <property name="disinfectant_type" value="3">
                                        </property>
                                </properties>
                        </object>
                </objects>
</commonsense>
```

Appendix B

INFORMATION SHEETS USED IN THE EXPERIMENTS

First page: Information Sheet used in the first experiment in the Medical Faculty in Kyushu University (in Japanese).

Second page: Information sheet used in the second experiment in the National Institute of Informatics in Tokyo (in English).

実験の流れについての説明

大学共同利用機関法人 情報・システム研究機構
国立情報学研究所
National Institute of Informatics

キネクト装置を用いたバーチャル・バイオセーフティ実験室プログラムの研修効果についての検討　(General Explanation)

本日は、私たちの実験に協力して頂きありがとうございます。
この実験の：

実験者は？: 国立情報学研究所プレンディンガー研究室　Nahum Alvarez, PhD Student

実験の目的は？：

今回の実験の目的は、安全な実験をするためのバイオセーフティの学習にバーチャル空間での実習が役に立つかを検討することにあります。皆さんの協力で以下の２つの質問への答えを見つけたいと考えています。

（１）3D バーチャル環境が、学習過程の促進に役に立つかどうか

（２）訓練をする時に、動きを伴った双方向性の活動が役に立つかどうか

実験の進め方は？：

(1) この説明が終了したら、皆さんには先ず、実験室の中で事故が起こった時の対応の仕方の手順を読んでもらいます。この時、メモをしても、写真を撮ってもいけません。

(2) 次に、ＡとＢの２つのグループに分かれて、実験装置の使い方の練習として、別々の作業をやってもらいます。

Ａの人は、キネクト装置を使い、自分の手の動きでバーチャル実験室の中で作業をします。

Ｂの人は、同じ作業をマウスとキーボードを使ってバーチャル実験室の中でします。

おおよそ 25 分程度かかります。

(3) それぞれが使った装置のバーチャル実験室での使い勝手について、質問票に回答してください。

(4) 事故対応の手順書を読んでもらいます。このときも、メモも写真も禁止です。バーチャル実験室内で発生した事故へ、バイオセーフティを守った手順で対応します。装置の使い方の練習通りに、キネクトの人は手の動きで、マウスとキーボードの人はそれらを使って、実践してください。

おおよそ 20 分程度かかります。

(4) この練習で学んだことと、経験したことについて質問票に答えてください。

(5) 質問票を回収後、最後に出席確認のサインをしていただき、わずかですが、記念品を差し上げます。途中に休憩をはさみ、全体でおよそ 75 分かかります。

＊英語での説明には、アシスタントの通訳が付きますので、質問等は遠慮なくしてください。時間のルールがありますので、移動や、止めなど、指示は守ってください。

＊コンピュータ画面を見るのは、連続して 20～25 分程度で、健康に影響は無い範囲内ですが、画面を見つめていて気分が悪くなるなどあったら、その場ですぐに申し出てください。いつでも中止できます。

＊この文章の説明を受け、読んだ上で、次の実験へ進んだ方は、記載している事項を理解し、実験への協力を承諾したものとみなさせていただきます。疑問がある場合には、その時点で速やかに申し出てください。

ご協力感謝します！　　では、開始します。

大学共同利用機関法人 情報・システム研究機構
国立情報学研究所
National Institute of Informatics

PARTICIPATE IN RESEARCH

Information for Prospective Participants

Biosafety Experiment (16th floor, Room 1608)

Research Team Contact

Researcher: Nahum Alvarez, PhD Student, National Institute of Informatics, The Graduate University for Advanced Studies, Japan. (16th Floor, Room 1608)

What is the purpose of the research?

The purpose of this research is to investigate: learning methods in the context of biosafety.

Are you looking for people like me?

The research team is looking for students, staff members and professionals, or anyone who has an interest in 3D virtual interactive applications. Note: Previous exposure to computer games is welcome, due to the nature of the simulation.

What will you ask me to do?

Your participation will involve:

(1) Reading a document about an accident solving protocol in a biosafety laboratory.

(2) Running a tutorial in the virtual environment. This will take around 5 minutes.

(3) Solving a number of dangerous situations in the virtual laboratory following a specific biosafety procedure. This will take around 50 minutes.

(6) You will also be asked to answer a questionnaire about your experience after certain stages.

In total it will last around 70 minutes and there will be short breaks between each stage of the experiment.

Are there any risks for me in taking part?

The research team does not believe there are any risks beyond normal day-to-day living associated with your participation in this research.

It should be noted that if you do agree to participate, you can withdraw from participation at any time during the project without comment.

Are there any benefits for me in taking part?

It is expected that this project will not benefit you directly if you are not a medical student/professional.

Will I be compensated for my time?

1000 yen will be rewarded to the participants who complete the experiment and answer the questionnaires.

Please note that whole experiment lasts around 70 minutes.

I am interested – what should I do next?

If you have any questions or would like to participate in this study, please contact the research team for details at nahum.alvarez.ayerza@gmail.com .

You will be provided with further details to ensure that your decision and consent to participate is fully informed.

Thank you!

Appendix C

PROTOCOL SHEETS GIVEN IN THE EXPERIMENTS

First page: Protocol Sheet used in the first experiment in the Medical Faculty in Kyushu University (in Japanese).

Second page: Protocol sheet used in the second experiment in the National Institute of Informatics in Tokyo (in English).

スピル事故対応の手順 (Protocol)

感染性のある危険な液体が床にこぼれて飛散した場合、これをスピルと呼びます。スピルは実験室にいる人みんなを感染の危険にさらすので、ただちに正しくその処理をしなければなりません。今日の実験では、すでに必要な個人防衣（ガウン、実験室用スリッパ、マスク、手袋）は正しく身につけていると仮定して、すぐにスピル処理を進めます。

スピルの処理の手順は：

1- 最初に、飛散した感染性のスピルの周囲を吸収材のスポンジ（ソックス）で囲って、液体による汚染が拡大しないように土手を作る

2- スピルを吸収シート（マット）で外側からスピルの中央へ向かっておおって行く。これは、かぶせたシートの重さで、外側へスピルが広がらないように、外から内へ向かってかぶせて行く。重ねながら中央がおおわれるまで、くり返す。

3- おおった吸収材の上から、消毒薬（細菌なので、今日は効く消毒薬はエタノールかイソプロパノールとする）をたっぷりかける。この時も、感染性の残るスピルを外へ広げないように、外側から内側へとらせん状にかけて、外側から順に消毒して行き、最後に一番たくさんのスピルがあるはずの中央にもたっぷりかける。

4- 消毒薬は直ぐには効かないので、効果が出るまでそのまま待つ。

5- スピルが消毒されたことを確認して、使った吸収材も一緒に廃棄物を捨てます。汚染地域が広がらないように、スピルをおおった場所の、外側から内側へ向かって集めて行き、それを取って、感染性物質用のゴミ箱へ捨てる。

6- この後に、個人防衣を脱いで、手洗いをして退出しますが、この実験では省略します。

＜注意＞

スピル処理をする時は、風上、風下を常に考慮して、排気口とスピルの間に立つことが無いようにすること。風下に立つと、処理をしている時に、自分に向って感染性物質の飛沫が飛んできて吸い込んで、感染することがあり危険です。もちろん、スピルに踏み込んではいけません。

How to clean an infectious spill:

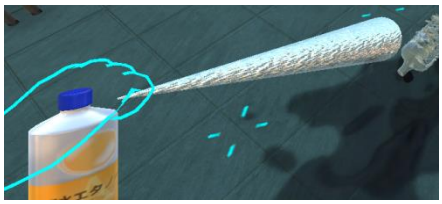
This protocol applies when a dangerous infectious liquid spills and disperses to the floor.

Those who are in the laboratory - since everybody is exposed to the danger of infection -have to carry out the processing correctly immediately.

Procedure of cleaning the spill:

1- Enclose the spill with absorbent sponges, in order to contain the spill and avoid its expansion. There are marks (see in the next figure) in the spill to help you placing the absorbents.

2- Cover the spill corners and then its center with absorbent sheets of paper. Is not necessary to cover the same zone of the spill with more than one sheet.
3- When the spill is covered it will begin to be absorbed but is still infectious, so spray it with an antiseptic (ethanol or isopropanol). Spray with spiral movements from the outsides to the center. For using the spray (see in the next figure), simply move it over the spill.



4- Since an antiseptic is not immediately effective, wait until it makes effect.
5- When the spill has been disinfected, you have to dispose the generated waste. Amass all the waste in one heap in the center of the spill.
6- When all the waste materials are merged in one, throw it in a biohazard bin.

<Caution and notes>

When you work around a spill, don't stand between an exhaust vent and the spill because you can inhale infectious vapors.

When you use all the absorbents from a table, they will appear again in the same place when you come back later.

Appendix D

STATISTIC TESTS FOR THE QUESTIONNAIRES RESULTS

Table I: t-test results of the comparison of recovery time results between dynamic feedback and static feedback versions from the first experiment.

Type of mistake	df, t-value	One-tailed	Two-tailed
Interposing between the spill and the air vent	t(102) = -2.42	p < 0.01	p < 0.02
Cover spill with sponge	t(21) = -1.86	p < 0.05	p = 0.07
Cover spill with pad	t(42) = -6.17	p < 0.01	p < 0.01
Cover center of spill before outsides	t(44) = -2.51	p < 0.01	p < 0.02
Disinfect the center of the spill before the outsides [a]	-	-	-
Throw waste without merging	t(7) = -0.66	p = 0.26	p = 0.52
Throw waste in wrong bin [b]	-	-	-

[a] There is no sample in this group.
[b] There is only one sample.

Table II: t-test results of the comparison between Kinect and mouse usability from the first experiment.

Question	df, t-value	One-tailed	Two-tailed
(U1) I would like to use this system frequently	t(52) = -1.23	p = 0.11	p = 0.22
(U2) The system was unnecessarily complex	t(54) = -2.39	p < 0.02	p < 0.03
(U3) The system was easy to use	t(53) = -4.79	p < 0.01	p < 0.01
(U4) The support of technical person is needed	t(51) = 5.18	p < 0.01	p < 0.01
(U5) The functions of the system are well integrated	t(48) = -0.78	p = 0.22	p = 0.44
(U6) There is too much inconsistency in the system	t(53) = 1.02	p = 0.16	p = 0.31
(U7) People will learn how to use the system quickly	t(47) = -2.11	p < 0.03	p < 0.05
(U8) The system is very cumbersome to use	t(54) = -2.55	p < 0.01	p < 0.02
(U9) I felt confident while using the system	t(54) = -3.16	p < 0.01	p < 0.01
(U10) I got tired because the control device	t(54) = 4.4	p < 0.01	p < 0.01
(U11) The controls were natural and intuitive	t(53) = -1.89	p < 0.05	p = 0.06

Table III: t-test results of the comparison between dynamic feedback and static feedback version regarding presence, perception of learning and application-specific feedback from the first experiment.

Question	df, t-value	One-tailed	Two-tailed
(P1) I was able to completely control events	t(22) = 1.63	p = 0.06	p = 0.12
(P2) Environment was very responsive to my actions	t(24) = 2.15	p < 0.03	p < 0.05
(P3) My actions within the environment seemed natural	t(22) = 0.76	p = 0.23	p = 0.45
(P4) My movement through environment was very natural	t(23) = 0.22	p = 0.41	p = 0.83
(P5) The Virtual environment seemed consistent to the real world	t(25) = 0	p = 0.5	p = 1
(P6) I was able to move/manipulate objects in the environment very well	t(25) = 1.57	p = 0.06	p = 0.13
(P7) I adapted to the virtual environment very quickly	t(20) = 0.77	p = 0.23	p = 0.45
(P8) I experimented much delay between an action and its outcome	t(26) = 0.98	p = 0.17	p = 0.34
(A1) I remember the session in the virtual environment	t(22) = 1.47	p = 0.08	p < 0.15
(A2) I always knew what to do next	t(25) = 0.2	p = 0.42	p = 0.84
(A3) I noticed the system's warning messages	t(26) = 3.6	p < 0.01	p < 0.01
(A4) I understood why I received a warning	t(26) = 6.01	p < 0.01	p < 0.01
(A5) After received warning, I understood what to do	t(25) = 4.41	p < 0.01	p < 0.01
(A6) I understand the problems I can have when cleaning a toxic spill	t(26) = 1.78	p < 0.05	p = 0.08
(A7) The Application is very useful for lab training	t(26) = 0.69	p < 0.25	p = 0.5

Table IV: t-test results of the comparison between using the system with Narrative manager and without it from the second experiment.

Question	df, t-value	One-tailed	Two-tailed
(P1) I remember completely the session in the virtual environment	t(28) = 3.31	p = 0.001	p = 0.002
(P2) I knew what to do next at any time	t(28) = 3.49	p < 0.001	p = 0.001
(P3) I noticed warning messages from the system	t(28) = 1.85	p = 0.037	p = 0.074
(P4) When reading a warning message from the system, I understood why	t(28) = 2.37	p = 0.01	p = 0.02
(P5) When reading a warning or an advice message from the system, I knew what to do next	t(28) = 5.73	p < 0.001	p < 0.001
(P6) I noticed differences in the application between sessions	t(28) = 3.53	p < 0.001	p = 0.001
(P7) I think while using the application the scenario tried to help me	t(28) = 2.83	p = 0.004	p = 0.008
(P8) I think while using the application the scenario tried to obstruct me	t(28) = 2.09	p = 0.002	p = 0.004
(P9) I remember how to treat a spill of infected blood	t(28) = 2.44	p = 0.01	p = 0.021
(P10) I understand the problems that can arise solving a spill of infected blood	t(28) = 2.05	p = 0.024	p = 0.049
(P11) I think this application would be very useful to training laboratory procedures	t(28) = 2.29	p = 0.014	p = 0.029
(P12) The events in the system were different each time I played	t(28) = 4.65	p < 0.001	p < 0.001
(I1) The session was suspenseful	t(28) = 0.93	p = 0.177	p = 0.355
(I2) The scenario grabbed my attention	t(28) = 2.72	p = 0.005	p = 0.01
(I3) I would like to play again if I had the chance	t(28) = 0.93	p = 0.178	p = 0.357
(I4) I thought the session was very interesting	t(28) = 2.73	p = 0.005	p = 0.01
(I5) I got caught-up in the session without trying to	t(28) = 2.27	p = 0.015	p = 0.03
(I6) I would like to know more about the scenario	t(28) = 2.28	p = 0.01	p = 0.03
(I7) I liked this session a lot	t(28) = 1.85	p = 0.036	p = 0.073

Appendix E

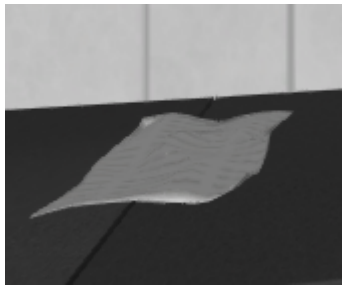
KNOWLEDGE TESTS USED IN THE EXPERIMENTS

実験を通じて学んだ内容についてたずねます：　以下の質問に、それぞれの指示に従って解答してください。選択肢があるときは、解答を丸で囲んでください。答えが複数ある場合もあります。

1. スピルを処理するときに立ってはいけない場所がある。それは、どこで？
   なぜ立ってはいけないのかを答えなさい。

2. スピルが拡大しないように周囲をかこむために使うものはどれか？

(a)



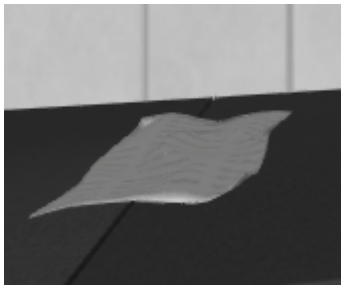
(b)

(c)

3. スピルをおおうために使うのはどれか？

(a)

(b)



(c)

4. スピルを処理した後の廃棄物（汚染された吸収材など）を捨てるゴミ箱はどれか？

(a)

(b)

(c)

5. 正しいスピル処理の順番の並びはどれか？
   (a) スピルの境界を囲む→スピルをおおう→スピル領域を消毒→汚染物を中央へ集める→廃棄する
   (b) スピルをおおう→スピルの境界を囲む→スピル領域を消毒→汚染物を中央へ集める→廃棄する
   (c) スピルの境界を囲む→スピルをおおう→汚染物を中央へ集める→スピル領域を消毒→廃棄する
   (d) スピルをおおう→スピルの境界を囲む→汚染物を中央へ集める→スピル領域を消毒→廃棄する

6. スピル処理をした後の、廃棄物の捨て方で正しいのはどれか？
   (a) 手で廃棄物をつかみ、一般ごみのゴミ袋へ捨てる
   (b) トング（炭バサミ、氷バサミ）で廃棄物をつかみ、可燃性ごみのゴミ箱へ捨てる
   (c) 手で廃棄物をつかみ、バイオハザードごみのゴミ箱へ捨てる
   (d) トング（炭バサミ、氷バサミ）で廃棄物をつかみ、バイオハザードごみのゴミ箱へ捨てる

7. スピルを吸収材で被う手順はどれが正しいか？
   (a) 中央を被って、それから辺縁を被う
   (b) 先ず周りから被って、中央を被う
   (c) 特に順番は無い

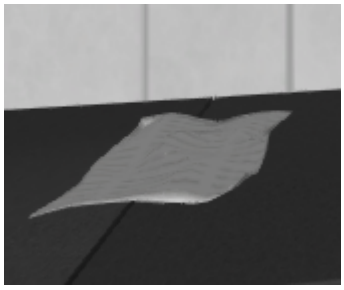
8. 今回のスピルの消毒に使う組み合わせで正しいのはどれか？
   (a) エタノールか、イソプロパノール
   (b) 洗剤（海面活性剤）か、漂白剤（ブリーチ）
   (c) 漂白剤（ブリーチ）か、エタノール

9. バーチャル実験室でスピルの処理をした時に、手順を間違った人は、最低 2 つの間違った例をあげて、正しい手順は何だったかを書いてください。

Learning Experience: Please answer each question and if necessary mark the correct answers (there may be more than one) with a circle.

1. There is an incorrect position to handle a spill? Where and why?

2. Which ones should be used for bordering the spill?

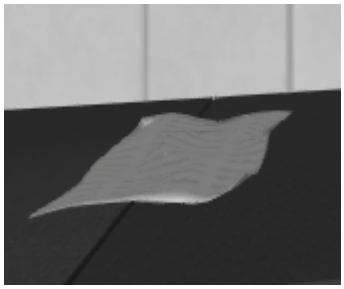
1 

2 

3 

3. Which ones should be used for covering the spill?

1 

2 

3 

4. Which ones should be used for disposing the spill waste?

1 

2 

3 

5. What is the correct spill treatment order?
    (a) Border the spill, cover the spill, disinfect the spill, merge papers, dispose waste
    (b) Cover the spill, border the spill, disinfect the spill, merge papers, dispose waste
    (c) Border the spill, cover the spill, merge papers, disinfect the spill, dispose waste

6. Spill must be covered in the following order:
    (a) Center, then Borders
    (b) Borders, then Center
    (c) Any order

7. Disinfection of the spill must be done with:
    (a) Ethanol or Isopropanol
    (b) Detergent or Bleach
    (c) Bleach or Ethanol

If you made mistakes, please describe two, and what should be the correct behavior

Appendix F

USERS' COMMENTS ON KINECT USABILITY:

- The system with Kinect was more difficult, but I felt that it have more capabilities.
- The movements for walking forward and backwards should be easier than making one step with the leg.
- It was difficult. If I become used to it, I feel I can handle it well.
- If the bottles and others could be broken, it would feel more realistic. In general it was fun.
- With Kinect you use your real body and movements so if I use the system, I prefer Kinect.
- It was fun to use it.
- My arms got tired with the Kinect system. If it's for using it for a long time, I prefer the mouse system. However, it was a bit tiring to have to look upward when seating using the mouse.
- It's necessary to get used to the controls due to move the camera or taking things by accidentally moving the hands.
- In the beginning the Kinect controls were more difficult, but once I got used to them, it became easier.
- It would be good it there was a pause button. I think learning while having fun is a good mechanism.
- It was tiring to move the hands.
- It was well done and it was interesting. My eyes got a bit tired.
- I think the Kinect system is more usable, but the some operations are difficult and I got tired.
- Using Kinect was more difficult than I thought. The mouse system was easier.
- It was fun to use it but it was difficult to carry out the actions I wanted to do.