Personalized Educational Games

Developing agent-supported
scenario-based training

The research reported in this thesis was funded by - and partially conducted at - TNO, the Netherlands Organization for Applied Scientific Research.

SIKS Dissertation Series No. 2014-22

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

Printed by Ridderprint, Ridderkerk

Cover design by Esther Ris

ISBN 978-90-5335-861-0

Personalized Educational Games

Developing agent-supported scenario-based training

Gepersonaliseerde Educatieve Spellen

De ontwikkeling van - door agenten ondersteunde - scenario-gebaseerde training

(met een samenvatting in het Nederlands)

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Universiteit Utrecht op gezag van de rector magnificus, prof.dr. G.J. van der Zwaan, ingevolge het besluit van het college voor promoties in het openbaar te verdedigen op donderdag 5 juni 2014 des ochtends te 10.30 uur

door

Maria Margaretha Magdalena Peeters

geboren op 8 december 1981, te Heerlen

Promotoren:	Prof.dr. J.-J.Ch. Meyer
	Prof.dr. M.A. Neerincx
Co-promotor:	Dr. K. van den Bosch

Tell me and I'll forget; show me and I may remember; involve me and I'll understand

Contents

List of Appendices

Backmatter

Dankwoord

Het is zover: het proefschrift ligt er. Voor de lezer begint het verhaal hier, maar voor mij is dit het einde van een periode van vier jaar onderzoek. Het waren vier interessante en leerzame jaren die ik voor geen goud had willen missen. Dit proefschrift is het resultaat van die vier jaar en het kwam er dan ook niet vanzelf. Aan de komst van dit werk hebben verscheidene mensen bijgedragen. Ik wil hier graag de ruimte nemen hen te bedanken voor hun steun, hulp, inspiratie, energie en geduld.

Allereerst wil ik mijn (co-)promotoren bedanken: Karel van den Bosch, John-Jules Meyer, en Mark Neerincx. Karel was altijd bijzonder betrokken bij zowel mijn onderzoek als mijn persoonlijk welzijn. Ik kon altijd bij hem binnenlopen voor een praatje, een vraag, of een oppepper. Hij heeft me geleerd om mijn ideeën helder en eenduidig op papier te zetten en vond elke keer weer tijd voor het geven van nuttige feedback op al mijn geschreven werk. Mark wil ik bedanken voor zijn aansporingen om al in een vroeg stadium te beginnen met experimenteren: dit heeft geleid tot een vliegende start van dit project. Daarnaast stel ik zijn analytisch inzicht, snedige vragen en rake opmerkingen tijdens overleggen zeer op prijs. Ik ben blij dat ik in mijn huidige werk bij de TU Delft onze samenwerking voort mag zetten. John-Jules wil ik bedanken voor zijn brede interesse en tomeloze enthousiasme. Al tijdens mijn master wist hij mijn nieuwsgierigheid naar intelligent agent technology te prikkelen en ik ben blij dat ik tijdens dit promotieonderzoek de mogelijkheid kreeg om samen met hem die interesse verder uit te diepen. Met zijn vieren waren we een goed team. Ik denk met veel plezier terug aan onze overleggen. Bedankt voor jullie inspiratie en vertrouwen. Jullie hebben me altijd de ruimte gegeven om mijn eigen ideeën te verkennen en uit te werken.

Daarnaast wil ik de Master-studenten bedanken die hebben mee gewerkt aan dit onderzoek: Gwen, Pauline, Christian en Ruben. Niet alleen vanwege hun inhoudelijke bijdrage, maar vooral ook voor de gezelligheid die zij brachten op de (doorgaans rustige) C-gang bij TNO. De samenwerking met hen heeft mijn promotieperiode een stuk leuker en energieker gemaakt. Bovendien hebben ze me de mogelijkheid gegeven me te ontwikkelen als begeleider en projectcoördinator.

Dan zou ik graag alle mensen willen bedanken die hebben deelgenomen aan dit onderzoek, te beginnen met de BHV-ers die hebben meegedaan in de adaptieve trainingsscenario's (zie Hoofdstuk 7). Voor diezelfde scenario's wil ik ook Jan en Quirine bedanken, de acteurs die tijdens deze studie de rollen van slachtoffer en omstander vertolkten. In datzelfde rijtje horen ook alle mensen thuis die ik mocht

interviewen in het kader van scenario-gebaseerde training: de BHV en EHBO instructeurs, de mensen van het UMC in Utrecht, de instructeurs van de CCO en HDO opleidingen in Den Helder, en de LARP organisator (zie ook Appendix A).

I would also like to thank the members of the reading committee: Frances Brazier, Dirk Heylen, Catholijn Jonker, Bruce McLaren, and Peter Werkhoven. I am grateful for their willingness to spend their valuable time and attention on reading my thesis and providing me with their much appreciated comments. In particular I would like to thank Bruce for the various talks and discussions we have had in the past few years that helped me shape my research interests.

Dit onderzoek is ontstaan uit een nauwe samenwerking tussen de Universiteit Utrecht en TNO Soesterberg. Ik heb daardoor de luxe gehad om de afgelopen vier jaar bij beide instituten een werkplek te hebben. Enerzijds gaf mij dit de mogelijkheid om (via TNO) te ontdekken welke concrete vragen en problemen er leven in de maatschappij die om een haalbare oplossing vragen. Anderzijds gaf dit mij de mogelijkheid om (bij de UU) te onderzoeken welke theoretische, abstracte problemen hieraan ten grondslag liggen en welke oplossingen ervoor ontwikkeld kunnen worden.

Een ander gevolg van het hebben van twee werkplekken is dat ik dubbel zoveel leuke collega's had. Van de Universiteit Utrecht wil ik bedanken: Amanda, Bas, Edith, Eric, Gennaro, Geraldine, Gerard, Hado, Henry, Jan, John-Jules, Joost v. O., Joost W., Liz, Lois, Luora, Maaike, Max, Mehdi, Michal, Nick, Nieske, Paolo, Rogier, Sjoerd, Steven, Susan, Tom en alle anderen. Ik heb leuke herinneringen aan, onder andere, weekendjes in Friesland, Antwerpen, en Luxemburg, de Sotsji-borrel, een murder mystery party, lasergame sessies, pannenkoeken, en stevige discussies over wetenschap en politiek tijdens de lunchpauzes.

Ook denk ik met plezier terug aan mijn collega's bij TNO. Met hen nam ik deel aan activiteiten zoals boogschieten, de verpot-je-plant-borrel, een Solex-rit door Den Bosch, een pixelation workshop, en Sinterklaas. Dit leverde veel grappige, en vooral ook gezellige momenten op waar ik graag aan terug denk. Hiervoor wil ik bedanken: Arnold, Dennis, Emiel, Joris, Jurriaan, Karel, Koos, Leo, Maaike H., Mark, Patrick, Philip, Rudy, Sam, Tijmen, Tina, Vanessa en alle anderen. In het bijzonder wil ik Annerieke, Aletta, Ivo, Maaike D., Nienke, Sanne, Sylvie, en Ward bedanken: met hen heb ik bijzonder veel lol gehad tijdens het organiseren van activiteiten voor de personeelsvereniging van TNO Soesterberg.

Ik wil mijn vriendinnen en vrienden bedanken voor hun motiverende woorden en aansporingen: Anne, Erik, Evelien, Fraukje, Gitte, Hester, Jacqueline, Judith, Liesbeth, Manon, Martijn, Milou, Mirte, Peter-Bas, René, Sanne, Susan, en Tim. Met hen kon ik in mijn vrije tijd afschakelen, opladen, interessante gedachten uitwisselen, en, eerlijk is eerlijk, procrastineren. Ook wil ik hen bedanken voor het geduld en begrip op momenten dat ik het liet afweten vanwege de zoveelste deadline van versie x.y voor dit of dat hoofdstuk. Daarnaast wil ik eenieder bedanken die ik hier niet expliciet genoemd heb, maar met wie ik de afgelopen jaren wel leuke en inspirerende momenten heb beleefd.

Mijn ouders, Jo en Luc, wil ik bedanken voor hun steun, liefde, en wijze lessen. Zonder hen had ik nooit de mogelijkheid gehad dit carrière pad te bewandelen. Al vroeg hebben zij mij geleerd en gestimuleerd om kritisch, nieuwsgierig, en onder-

zoekend te zijn. Paul en Joep (mijn broers en paranimfen) en Naomi wil ik bedanken voor hun morele steun en voor de wetenschap dat ik altijd bij hen terecht kan.

Als laatste wil ik Elwin bedanken voor zijn onuitputtelijke steun, geduld, en liefde. Toen de eindstreep in zicht was, maar ik het niet meer helder zag, heeft hij me omarmd en over die eindstreep heen getrokken. En nu zijn we er. En er mag gedanst worden.

♪♫♪♫♪♩♪♫♪♫♪♫.

CHAPTER 1

Introduction

Abstract - People working in high-risk professions, as can be found in the police force, fire department, aviation, and health services, rely on their past experience to make adequate decisions under stressful circumstances. In order for them to acquire this experience, they need a substantial amount of practice.

A safe way to train people in high-risk professions is with the use of *scenario-based training* (SBT). SBT is a practical training form during which learners engage in interactive role-playing exercises, called 'scenarios'. Scenarios are usually staged within a simulated environment. SBT is considered to be a suitable and effective training form to provide learners the experience they need. Yet despite its potential, SBT also has its limitations. First of all, it requires considerable logistic and organizational efforts. Secondly, it offers limited opportunities for personalized training. And lastly, it poses problems for monitoring and interpreting events in the scenario in a structured, systematic, and non-ambiguous manner.

Development of new training technology may alleviate the obstacles that prevent ample and effective use of SBT. One major improvement would be to reduce the number of staff personnel currently required for the preparation and delivery of SBT. This may be achieved by automating the activities currently performed by staff personnel, for instance by combining a virtual environment with artificial intelligence. In this thesis, such a combination is referred to as a *Personalized Educational Game*.

Personalized educational games (PEGs) automatically manipulate virtual simulated environments (e.g. game worlds) to offer learners personalized autonomous training. This thesis investigates the requirements, desired effects, and design of *automated* scenario-based training in the form of a PEG. The current chapter presents the problem statement, research questions, and outline of the thesis.

> "An individual understands a concept, skill, theory, or domain of knowledge to the extent that he/she can apply it appropriately in a new situation."
>
> **Gardner (1999)**

People working in high risk professions, such as policemen, firefighters, soldiers, pilots, First Aiders, nurses, and doctors, rely on complex skills. Complex skills involve the integration of cognitive, social, emotional, and motor skills. Professionals in these types of professions need to combine their knowledge and their assessment of the current situation to decide which procedure is most appropriate for the task at hand. They need to be able to make decisions and accurately perform the appropriate procedure under stressful circumstances, such as time pressure and life-or-death situations.

An effective training form for developing integrated complex skills is '*scenario-based training*' (SBT) (Cannon-Bowers et al., 1998; Oser et al., 1999; Salas et al., 2006; Van den Bosch & Riemersma, 2004). SBT is a practical training form during which learners engage in interactive role-playing exercises, called '*scenarios*'. Scenarios are staged within a simulated environment, so that the potential consequences of failures are included. As a result, SBT offers learners the opportunity to safely gain experience with situations that are representative for their future profession. In addition, learners are able to practice the appropriate procedures within the context of the scenario. In doing so, learners experience the outcomes of performing tasks both correctly and incorrectly. This allows learners to reflect on the differences between situations, task performances, and outcomes within the context of the scenarios. Even though both instructors and researchers acknowledge the importance of practical SBT opportunities, they also emphasize the following associated limitations.

First of all, the preparation and execution of SBT can be costly and time-consuming. Not only do scenarios require the clearance and preparation of an area to stage the scenarios, i.e. the simulated environment, they also depend on the presence of (large numbers of) staff members to play various parts in the storyline, e.g. opponents, team members, or victims. As a result, SBT often requires elaborate organization and planning to ensure the presence of all people required. Additionally, all role players must receive instructions on their roles, including their interaction possibilities with the other roles. This requires elaborate script preparations.

Another challenge of SBT is the limited control and coordination possibilities at run-time, i.e. as the scenario is playing. The scenario scripts do sometimes provide limited possibilities to repair the storyline in case of an unforeseen event, but there are situations where it is difficult to redirect the scenario into the desired direction. And even if the scenario runs as intended, it may prove to be unsuitable for the learner. In such cases an alternative scenario would be desirable, but due to the extensive preparations required, it is not possible to improvise or switch to a more suitable scenario.

Because of the limitations mentioned above, instructors are generally not able to fully cover the learning content in practical training sessions; they need to make a selection of the learning content that should *at least* be covered by the scenarios. As

a result, learners are often restricted to either training the most common situations, or the most severe or dangerous ones, depending on the selection of the instructor. The cases not covered during practical training are usually presented and discussed in theory, which is believed to be insufficient to prepare learners for their responsibilities after training (Baldwin & Ford, 1988; Grabinger & Dunlap, 1995; Lipshitz et al., 2001; Prince, 2004; Yamnill & McLean, 2001; Young, 1993). As a result, most learners start gaining actual practical experience with the remaining tasks only when they are already working on the job.

A story illustrating SBT in practice

The following story illustrates the concept and use of SBT along with the possible consequences of its current limitations.

Carl is part of the 'In-Company Emergency Management (ICEM) Team' at his office. In the Netherlands, an ICEM team is responsible for managing casualties at work until the incident has been adequately handled, or until the official emergency services arrive (i.e. ambulance, firemen, police). Carl and his ICEM teammates are qualified to apply First Aid, fight and control small fires, and initiate and manage evacuation procedures.

Once every six months, Carl and his team members participate in a one-day training session. During these training sessions, they receive information about possible changes in applicable policies and procedures. Furthermore, their tasks and duties are rehearsed. Part of this rehearsal entails their participation in several role-playing activities with actors who play the role of victims in casualties at the office.

Today is the first training day of Carl and his team members. In the past two weeks, Carl has been preparing himself by using his free evening hours to study his text books. He has the following question: How to distinguish a victim having a hypovolemic shock from a victim having a panic attack? Carl asks his instructor whether this question can be incorporated in today's role plays. The instructor, however, tells Carl that it will not be possible to incorporate his question in a role-playing exercise due to time limitations, and because today's actor did not prepare for these injuries. Instead, the instructor proposes to discuss the question in theory.

In the last exercise of the day, Carl and his team members need to evacuate the building due to a fire on the second floor. Carl and his team should guide all of the people in the building to one side of the building, and then take them all to the ground level using the stairs, not the elevator. However, somehow he and his team missed a small group of people. As a result, these people took the elevator. Afterwards, Carl and his team members conclude that their performance could have turned into a disaster had it happened in real life. Because the exercise did not go as planned, the exercise took longer than the instructor anticipated and reflecting on what happened was rushed so that the exercise could be finished by the end of the day.

Afterwards, Carl feels unsatisfied with today's training. He is still uncertain whether he will be able to distinguish a shock from a heart attack if he were to encounter it in real life. He'd rather have practiced both situations with an actor in a scenario. Carl also feels confused about the evacuation exercise. He doesn't really understand what went wrong and what he would have to do better if something like this ever really occurred at the office.

Reflections on the story

The story above illustrates how SBT can be used to offer learners practical experience with relevant situations. Carl and his team members are able to safely practice with simulated emergencies in their office.

The story also shows the limitations of the current organization of SBT. Training opportunities for Carl and his ICEM team are scarce; they receive training once every six months. Moreover, it is difficult for Carl's instructor to personalize the training session on the spot. Furthermore, when the evacuation scenario doesn't work out as planned, it is apparently difficult for the instructor to redirect the scenario in a desirable direction. And although the mistakes causing the flawed performance could still have been addressed during the reflection, too little time was taken to properly investigate the causes, let alone to properly reflect on the experience offered by the scenario. To conclude, the example illustrates how unforeseen problems may result in a missed training opportunity for the learner.

Especially in larger exercises, such as the one described above, it can be challenging for the learners as well as the instructor to analyze mistakes. However, analysis is important to enable a proper explanation to be given. The most preferable response to a scenario would be to analyze the events, reflect on them, and provide the team with a new scenario that requires the knowledge obtained during reflection. It is not uncommon for scenarios to unfold differently from the original storyline due to unforeseen decisions of the learners or role players. In cases where the exercise fails to get across the intended learning content, the learner has to wait for the next opportunity - if there ever is any - to practice the learning goals.

To conclude then, the limitations of regular SBT are: 1) the preparation and execution of SBT is costly and time-consuming, and 2) control over the scenario at run-time is limited. These limitations result in a restricted bandwidth for personalizing training in terms of content, frequency, and volume.

How to solve the identified issues?

To overcome the limitations mentioned above, i.e. costs, manpower, and limited control at run-time, researchers have been studying the use of virtual environments and the automation of the role players in the exercises (Cannon-Bowers et al., 1998; Oser et al., 1999; Salas & Cannon-Bowers, 2001). The development of a virtual world is a useful enhancement because it alleviates the need to prepare a real-life simulated environment. And automating the role players avoids the need to assemble and instruct the role-playing staff members. Although, configurable virtual environments and virtual role players are a first solution to increasing training opportunities, the availability of virtual worlds and virtual role players does not of itself guarantee good training. A didactic reasoning component is needed to design and control, i.e. *direct*, the training exercise.

Control over the scenario as it unfolds is important in order to steer the scenario in a desirable direction. During traditional SBT, the exercise leader, or instructor, coordinates the role players as they push the scenario forward (Oser et al., 1999; Van den Bosch & Riemersma, 2004). All role players are focused on the learner's responses; they aim to create a situation that is instructive to the learner and lies within the scope of the learning goal. In other words, the responsibility of the role

players is not only to play their part, but also to warrant the didactic value of the scenario. This emergent feature of the training scenario is coordinated by a single operator, the exercise leader, from here on referred to as the *director*.

A study into the effects of control over a scenario at run-time showed that 1) adjustments in the level of support and/or challenge could be effectively delivered by manipulating the characters' behaviors at run-time, and 2) such interventions resulted in significant improvements of the scenario's suitability for a given learner, i.e. the learning value (Peeters et al., 2014). A comprehensive discussion of this study can be found in Chapter 7. This study shows the potential didactic value offered by automated SBT and realtime adjustments, allowing for effective control over the training scenario as it unfolds. This study was therefore a motivation to continue our investigations on automating the didactic reasoning processes underlying SBT.

The research aim of this thesis

This thesis investigates how SBT can be automatically controlled and directed to offer learners personalized practical training. For this, the research presented in this thesis combines knowledge from several research domains, i.e. educational psychology, instructional design, artificial intelligence, and human-computer interaction. The design presented here has theoretical foundations in cognitive load theory, constructivism, naturalistic decision making, information processing theory, and studies on meta-cognition. In addition, the research employs the following technologies: ontologies, intelligent agents, intelligent tutoring systems, HTN planners, semantically annotated objects, intelligent user interfaces, virtual environments, user models, and intelligent dialogue systems.

First, the concept of SBT is investigated to obtain a set of *requirements* for automated SBT. These requirements serve the purpose of delineating the solution space for designing automated SBT. Subsequently, the problem is divided into smaller problems; a comprehensive functional design for automated SBT is presented, including its various functional components, fully covering the identified set of requirements. An agent-based system architecture is proposed to coordinate the behavior of the functional components. The agent-based system is implemented in a prototype and tested for feasibility and effectiveness to produce the intended coordinated behavior. Thereafter, each of the functional components are developed and tested in separate research projects.

1.1 Research plan

The research presented in this thesis employs the following research plan.

Problem statement

How to design an automated system for SBT, based on scientific evidence, such that learners can engage in personalized autonomous training?

Research questions

The investigation is guided by the following research questions:

I. What are the requirements and anticipated effects of automated SBT?

II. What design could satisfy the identified requirements?
III. Does the proposed design indeed satisfy the identified requirements?
IV. Does the proposed design produce the anticipated effects?

Research approach

The research problem presented above is a *design problem*. Design problems entail the design of an artifact, that meets a set of criteria in order to solve a particular problem. In this thesis, the problem entails the design of a cognitive system that meets the requirements for automated scenario-based training in order to overcome the limitations of regular scenario-based training.

Typically, design problems do not allow for fully specified, empirical research questions right from the beginning. Instead, the design problem serves as the context for the conducted research; more specific research questions (empirical or otherwise) emerge from the design process as new design solutions are proposed and tested.

This thesis employs the situated Cognitive Engineering (sCE) method to approach the design problem of automated SBT (Neerincx & Lindenberg, 2008; Neerincx, 2011). For a full explanation of the sCE method, see Chapter 3. The sCE method starts with an analysis of the problem by means of interviews, literature research, and an exploration of available technological solutions. This analysis results in a detailed specification of the solution space constraining all *acceptable* design solutions in terms of *requirements*. This specification also includes possibilities to evaluate any proposed design; the design can be tested against the boundaries of the solution space, i.e. the requirements. In addition, the specification contains information about the intended effects or performance outcomes of the cognitive system. As a result of this approach, the sCE method produces more specific research questions, that can be appropriately addressed as they emerge from the design process. Designs can be tested regarding their feasibility and effects through user-based studies and experiments, formal specifications, and human-in-the-loop simulations.

1.2 Contribution

This thesis presents an approach to automating scenario-based training. Didactic knowledge is made available for integration within an educational game. In turn, this knowledge is used in the reasoning processes of various intelligent agents that together prepare and control the game scenario. The combined behavior of the agents results in a personalized scenario, staged within a virtual environment.

The conducted research has led to the following contributions:

- A verified agent-based architecture for automated SBT
- A set of requirements for automated SBT
- An overview of the intended effects of automated SBT
- An ontology that defines concepts relevant to automated SBT
- Validated designs for three components responsible for:
  - dynamic difficulty adjustments in realtime
  - automated scenario generation
  - supporting the scenario authoring process of the user
- Preliminary designs for:

- an agent-based user model of the learner
- a dialogue system to (a) stimulate students in explaining their own decisions and/or performance and (b) provide additional instructional explanations

- An extended description and discussion of the sCE method

Validated designs have been empirically tested on their effects in terms of performance outcomes. Verified designs have been analytically checked on the accordance between the theoretical specification and the actual implementation.

1.3 Thesis outline

This thesis is outlined as follows:

Chapter 2 - Background - delineates the problem space by specifying the types of training domains considered in this thesis and the most prominent features of effective practical training for these domains.
Chapter 3 - The situated Cognitive Engineering Method - presents and discusses the methodology employed throughout the research presented in this thesis.

Part I - General design & prototype - describes the development of the agent-supported system for automated SBT.
Chapter 4 - Requirements & Claims: Functional specification - delineates the solution space by specifying the requirements of automated SBT.
Chapter 5 - Multi-Agent System: Behavioral Specification - proposes the use of a multi-agent organization to coordinate the functional components of automated SBT.
Chapter 6 - Ontology: Conceptual Specification: formalizes the knowledge relevant to automated SBT with the use of an ontology.

Part II - Component Designs, Prototypes, & Evaluations - presents the individual designs for each of the comprising functional components.
Chapter 7 - Scaffolding Component - describes the development and evaluation of the dynamic difficulty adjustment component.
Chapter 8 - Scenario Creator - describes the development and evaluation of the automated scenario generation component.
Chapter 9 - Authoring Tool - describes the development and evaluation of the authoring tool that enables instructors to participate in the scenario generation process.
Chapter 10 - Future Work - describes preliminary research on the remaining components, i.e. the learner model and the reflection component.

Closing - indicates the end of Part II in order to separate the last chapter from the previous chapters.
Chapter 11 - General Discussion and Conclusion - reflects on the entire research process and recapitulates the research contributions presented throughout the thesis.

Most chapters are based on previously published work.

CHAPTER 2

Background

Complex high-risk professions, scenario-based training, and technology-enhanced learning

Abstract - To become proficient in decision making within complex task domains, learners need to acquire experience with a large variety of situations that are typical and critical for their profession. However, people working in high-risk professions cannot acquire their experience through learning on-the-job, because in real-life situations erroneous decisions may well result in grievous consequences. Scenario-based training (SBT) aims to offer learners the experience they need through role-plays. It allows learners to experience the consequences of correct and incorrect decisions in a relatively safe and controlled environment.

To automate SBT, this thesis combines two currently existing approaches to technology-enhanced learning: 'educational games' and 'intelligent tutoring systems'. The result of this combination is called a *Personalized Educational Game (PEG)*. PEGs present learners with realistic and representative practice situations that are staged in a virtual environment. In addition, PEGs are enriched with artificial intelligence that tailors scenarios to learners' individual needs. This enables learners to develop their competencies at their own level and pace. To make learning purposive and goal-directed, the events in the simulated environment as well as the behavior of key players are carefully managed based on educational principles.

To make SBT easier to organize and thus more accessible, PEGs aim for a new role for the instructor. The instructor should no longer be strictly necessary during training. However, a PEG must also support and exploit the expertise of instructors by allowing instructors to, for instance, manually author scenarios, control certain characters, or determine the learning goal. The human instructor is certainly not excluded from or replaced by the PEG, but rather the PEG allows for variable levels of automation, depending on instructors' preferences and availability.

> "An expert is a person who has made all the mistakes that can be made in a very narrow field."
>
> **Niels Bohr**

> "The essence of training is to allow error without consequence."
>
> **Card (2008)**

The previous chapter introduced the main problem statement of this thesis: *'How to design an automated system for SBT, based on scientific evidence, such that learners can engage in personalized autonomous training?'*

In order to provide an answer to this question, the problem space needs a more detailed specification. Therefore, this chapter starts with a discussion of the type of profession for which practical training is indispensable: complex high-risk professions. After establishing the characteristics of complex high-risk professions, we continue with an investigation of the required competencies. Subsequently, the chapter presents an analysis of how training programs can support novices in developing those competencies. Lastly, the concept of *Personalized Educational Games* is introduced, which provides the context for this thesis.

2.1 Complex high-risk professions

The aim of the research presented in this thesis is not to investigate one technology-enhanced training tool that fits all educational situations. The scope of this thesis is restricted to one particular type of profession: the *complex high-risk profession*. Complex high-risk professions can be found, for example, in the military, fire department, police force, and emergency health care environments.

The first characteristic of these training domains is that they are 'high-risk' professions, i.e. any mistakes made during task performance may result in grievous consequences, ranging from damage to the environment (e.g. oil leaks or forest fires) to damage to society (e.g. a criminal on the run or an escalating riot), or even damage to a person (e.g. injury or death). In this type of profession it is of the utmost importance that professionals are proficient in performing the tasks when they finish training.

In other (non-high-risk) professions, on-the-job learning is quite common; it allows the learners to carry out the tasks in the actual task environment, thereby gaining experience with the actual consequences of right and wrong decisions. But in high-risk professions, the possibilities for on-the-job learning are greatly reduced since any errors may well have detrimental effects. Hence, the challenge is how experience with the task performance can be developed in high-risk professions in a safe and controlled training environment to reduce the risks involved in on-the-job learning.

The second characteristic of this type of profession is that they encompass complex tasks (Campbell, 1988; Wood, 1986). Professionals in complex high-risk domains work under a large variety of highly dynamic circumstances, each of which requires a different procedure or approach. Timing, coordination, and communication play an important role for the task execution as well. Moreover, they often work with

a limited amount of information, and the expected outcomes of choosing a specific procedure are often uncertain. In addition, these professionals generally work under stressful circumstances, such as time-pressure and being aware of the possible impact of making a wrong decision.

To conclude, complex high-risk professions involve knowledge-intensive, complex tasks; although procedures exist, the nature of the tasks is not merely procedural. Professionals should be able to accurately assess a given situation to decide which procedure is applicable. In our society there are many such professions and it is important for society that such professions are thoroughly exercised.

2.1.1 Competencies in complex high-risk professions

The question arises as to how professionals are able to carry out such complex high-risk professions. Research has shown that experts in these types of professions engage in *naturalistic decision making* (Cohen et al., 1997; Klein, 2008; Lipshitz & Strauss, 1997; Lipshitz et al., 2001; Randel et al., 1996). They rely on their vast experience with the task domain. Based on their previous experience, experts store situational information in long-term memory in the form of patterns, called *schemas*. Schemas describe relevant cues, expectancies, plausible goals, and typical reactions related to particular situations. These schemas allow experts to a) recognize similarities between the current task situation and their previous experiences, b) find an applicable schema, and c) adopt the plan stored in that applicable schema.

When there are no or insufficient similarities between the task situation and the situations described in the available schemas in long term memory, experts adopt alternative strategies for decision making. They construct a mental image of the situation by means of cause-consequence relationships, engage in mental simulations, and anticipate multiple plausible scenarios (Klein, 2008). These processes require more active (conscious) reasoning and rely on critical thinking (Cohen et al., 1997).

Naturalistic decision making is highly contextual (*situated*): the outcomes depend on the interaction possibilities of the situation at hand (Brown et al., 1989; Young, 1993). For example: when trying to get across a ravine, it makes no sense to try and solve the problem without specific information about the current situation, e.g. are there woods nearby?, how steep are the edges?, how deep and wide is the ravine?, and so on. Such environmental elements provide *affordances* (Gibson, 1977), i.e. possibilities for a person to engage in certain interactions with the environment. For instance, if the edges are not too steep and offer enough grip, one might be able to climb down from them. It is only when such situational information about the environment is provided that it is possible to start solving the problem.

An important construct in naturalistic decision making is *situation awareness* (Endsley, 1995). Situation awareness refers to a person's knowledge about the current state of a dynamic environment. It is acquired through situation assessment, during which a decision maker constructs a comprehensive, meaningful image of the situation and its possible consequences or developments. Situation awareness allows for quick recognition of the current situation's (abstract) distinguishing features so it can be mapped to the schemas stored in long-term memory.

2.2 From novice to expert

So how can training support the development of these competencies? Expertise development requires time and effort. If novices want to develop the competencies identified above (e.g. decision making, situation awareness), they need to engage in ample and deliberate training and, most preferably, training that facilitates: 1) the expansion and refinement of tactical schemas, and 2) opportunities to practice solving complex and unfamiliar problems to gain insights into the meaning of situational factors, both individually and combined. Based on the analysis in the previous sections, we can infer that training for complex high-risk professions should:

- provide intensive, deliberate, and reflective practice over time
- allow learners to engage in situation assessment and decision-making in relevant and representative cases (Lipshitz et al., 2001; Young, 1993)
- encourage learners to study cases from various angles (Helsdingen et al., 2010)
- provide opportunities to actively explore and experience cause-effect relations (Brown et al., 1989; Lipshitz et al., 2001; Young, 1993)

2.2.1 Scenario-based training

A form of training often used for complex (high-risk) professions is scenario-based training (SBT). SBT provides access to experiential learning. During scenario-based training, learners prepare, execute and evaluate situations that are typical and/or critical for their future line of profession within a simulated environment (Cannon-Bowers et al., 1998; Oser et al., 1999; Peeters et al., 2012b; Salas et al., 2006; Van den Bosch & Riemersma, 2004).

Training within a simulated environment is preferable over on-the-job learning, especially in high-risk professions, because it allows for control over the learning situation, thereby reducing the risks involved with mistakes and increasing the possibilities to shape the learning situation as desired. However, it also introduces several additional complications, for instance regarding the transfer of training.

Transfer of training refers to the extent to which learners display the required competencies, originally acquired in the training environment, while working on the job (Baldwin & Ford, 1988). In order for the required competencies to transfer to the task environment, the required level of resemblance between the task environment and the training environment must be determined. This is especially important for salient aspects, such as situational cues relevant for situation awareness.

SBT has been applied in a wide variety of application domains, such as safety and defense, (emergency) health care, and soft skills. The scenarios offered to the learner during scenario-based training are designed in advance. A scenario revolves around a learning goal or a set of learning goals and consists of a set of events that should evoke a certain response from the learner. As such, scenarios are designed in such a way that opportunities for the learner to practice and demonstrate the appropriate responses are not left to chance (Fowlkes et al., 1998; Oser et al., 1999).

To conclude, SBT is considered to be effective for the practical training of complex high-risk task domains and therefore has been chosen as the training form employed throughout this research project.

2.3 Two approaches to technology-enhanced learning

From the above, we have established that SBT is a suitable training form for complex high-risk professions. SBT requires a simulated environment to stage the training scenarios. As explained in Chapter 1, the organization of a training session in a simulated environment is often elaborate, resulting in a shortage of training opportunities and limited options for personalization. This thesis addresses this problem by investigating a way to automate SBT.

Earlier approaches to the automation of SBT have mainly focused on the automation of the environment, by developing a virtual world, or automating the virtual characters such that they are able to believably play their parts in the scenario (Cannon-Bowers et al., 1998; Oser et al., 1999; Salas & Cannon-Bowers, 2001). Throughout this thesis an argument is presented for an additional layer of automation that is required to warrant the didactic value of the scenarios.

One research area that is particularly interesting when it comes to automated didactic control over virtual environments is the field of *educational games*. A second research area that aims to automate the didactic reasoning processes to offer learners sufficient support during automated training exercises is the field of *intelligent tutoring systems*. These two technology-enhanced training tools approach the automation of training in a virtual environment from different directions, thereby presenting seemingly complementary solutions for the automation of didactic reasoning in scenario-based training. This topic will be discussed in more detail below.

2.3.1 Educational games

During the past few decades, games have drawn the attention of both educational psychologists and instructional designers (Charsky, 2010; Dickey, 2005; Egenfeldt-Nielsen, 2006; Rieber, 1996). The anticipated advantage of learning through play has mainly been the expectancy of increased motivation in learners to engage in learning activities: games are fun. During these 'fun games' players voluntarily engage in a 'learning activity'; they unintentionally become better at playing the game. They are even willing to invest time, effort, and money to learn something they will never use in real life (Dickey, 2005; Malone, 1981). This has caused instructional scientists to start investigating the possibilities of employing games to teach people competencies that they can actually use in real life. Researchers discovered that many entertainment games exploit instructional principles to engage the player (Bopp, 2006; Gee, 2005). However, to prove that games really result in better performance and facilitate learning, stronger evidence is required.

Although review studies exist that investigate the effects of educational games, these have mostly shown indecisive conclusions regarding the effectiveness and efficiency of educational games (Connolly et al., 2012; Egenfeldt-Nielsen, 2006; Hays, 2005; Mikropoulos & Natsis, 2011). These inconclusive findings are due to the studies' experimental designs as well as the game designs.

First of all, studies that investigate the effects of games are often flawed in their control group, e.g. the study compares the game to non-traditional learning activities, or the game is designed based on instructional principles whereas the compar-

ative learning activity is not.

Secondly, the two conditions compared in the study often differentiate from one another on multiple aspects. The investigated educational games comprehend a multitude of design choices and game elements that are compared in total to a control activity. This is also problematic when trying to compare the outcomes of different studies. One study may show positive effects whereas another study does not, but since the game design in the two studies is often very different, it is hard to tell what caused the difference in the found effects.

An additional shortcoming reported in reviews regarding the effects of educational games (and virtual environments) is that, in general, the studies do not explicitly describe the employed underlying educational and/or pedagogical principles (Kebritchi & Hirumi, 2008; Mikropoulos & Natsis, 2011; Wu et al., 2012).

Sampayo-Vargas et al. (2013) and Wilson et al. (2009) have suggested that these problems could be overcome if studies employed a clear taxonomy of design features, e.g. game elements or system components, and a methodology to guide the evaluation of those design features. Examples of this kind of research can be found in the studies conducted by Malone (1981) and Andersen et al. (2011).

To conclude, educational games offer a way to turn a 'fun' activity, i.e. playing a game, into something useful, i.e. engaging in a learning experience. The concept of educational games also seems to be well-aligned with the concept of SBT. However, the scientific evidence that people actually learn from playing games is as yet inconclusive.

2.3.2 Intelligent tutoring systems

A different approach to technology-enhanced training are intelligent tutoring systems. Intelligent tutoring systems (ITSs) provide automated adaptive guidance during training exercises. In contrast to the design of educational games, the design of ITSs is often grounded in extensive research on effective instruction and human tutoring (Graesser et al., 2012). According to Easterday et al. (2011), the use of principles and concepts available from research in the intelligent tutoring domain could greatly enhance the efficacy of educational games for training purposes.

ITSs are systems that "provide moment-by-moment adaptation of the instructional content and form to the changing needs of the individual learner" (Ohlsson, 1986, p.294). Ohlsson (1986) distinguishes four main challenges relevant for the design of ITSs: 1) cognitive diagnosis - the ability to infer a person's cognitive state based on the observed performance; 2) subject matter analysis - the need for explicit representations of the relevant subject matter; 3) teaching tactics - the availability of a large variety of possible actions in order to respond to each action, performance, or utterance with the right tutorial action; and 4) strategies for teaching - the availability of a set of guidelines as to which teaching tactic should be employed under what circumstances.

VanLehn (2006) described the behavior of ITSs as an outer and an inner loop. The *outer loop* selects a task that is suitable for the learner's current competencies, whereas the *inner loop* provides step-by-step guidance during the actual task performance. The outer loop dynamically selects learning topics while taking the learner's current competencies into account. In addition, the outer loop selects a mode for the

task performance, such as 'demonstration', 'hint-based', or 'independent'. VanLehn describes the inner loop as the actions of the system in response to the learners performance on individual steps of the task. Common services provided by ITSs during the inner loop are minimal feedback, error-specific feedback, hints, and a review of the solution.

Well-known examples of ITSs are: a) Cognitive Tutors, tracing the learner's solution and comparing it to an expert model (Koedinger & Corbett, 2006); and b) the (constraint-based) intelligent tutoring systems proposed by Ohlsson and Mitrovic, which compare the learner's solution to a set of constraints that should hold for all acceptable solutions (Mitrovic & Ohlsson, 1999; Mitrovic et al., 2001).

To conclude, ITSs offer a way to provide learners effective support during their autonomous performance of learning exercises. However, ITSs have generally been employed in theoretical, text-based exercises rather than game environments.

2.4 Personalized Educational Games (PEGs)

Previous research has shown positive effects on learning as a result of combining educational games with principles borrowed from the field of intelligent tutoring systems, thereby benefiting from the motivational and experiential learning aspects of educational games on the one hand and the instructional guidance provided by ITSs on the other (Aleven, 2010; Conati & Manske, 2009; Habgood & Ainsworth, 2011; McNamara et al., 2010). The research presented in this thesis also aims to combine educational games with principles from intelligent tutoring, yet in this case this combination is especially directed at the automation of SBT. This results in a new system design that allows for (semi-)automated SBT: *personalized educational games* (PEGs).

PEGs present the learner with realistic and representative practice situations in a simulated environment. Important features of the simulated environment, such as the selection of suitable learning goals and the behavior of the non-player characters, can be controlled automatically as well as by humans. As such, the system allows for both instructor-controlled and autonomous training (in the absence of other people).

In the simulated environment, learners are presented with representative situations that allow them to develop their competencies at their own level and pace. The learner and other key players (either controlled by humans or by artificial intelligence) are able to interact with each other and the environment. This allows them to actively accomplish tasks and experience the effects of their choices. To make learning purposive and goal-directed, the events in the simulated environment as well as the behavior of key players are carefully managed based on educational principles. PEGs include an automated didactic component that is able to reason about the underlying didactics of the scenario presented to the learner.

At this point, a PEG is still a conceptual idea for the design of automated SBT. The rest of this thesis presents the investigation, specification, and development of this envisioned system.

2.4.1 The selected domain in this thesis: First Aid

A PEG is meant to be a form of automated and/or semi-automated SBT that can be applied in various professions, including complex high-risk ones. However, in order to develop and verify prototypes, one particular training domain should be chosen that allows for the implementation and testing of particular exemplary cases. It was decided to consistently employ one training domain in all prototypes and validations in this research project.

The following training domains were considered for use within the prototypes: First Aid, conversational skills, military, medicine, firefighting, and emergency medical services. All of these domains were analyzed regarding their suitability for use in a prototype based on their representativeness for high-risk complex tasks, e.g. Can the tasks be ordered in terms of complexity?; Is the task performer required to manage a large variety of tasks and skills?; Is the task performer required to rely on his/her own discretion at times?; Should the task performer be able to reckon with uncertain goals?; Can there be multiple solutions to the same problem?; and so on.

Based on this analysis, the selected training domain was First Aid. First Aid entails the provision of initial care in cases of illness or injuries. In general, First Aid is performed by non-experts (i.e. they did not receive full medical training), but they did receive basic training for the purpose of providing First Aid. First Aiders usually take charge of the situation until definitive medical treatment can be accessed. In some cases, an illness or minor injury does not require additional medical care beyond the First Aid intervention. First Aid often consists of simple and potentially life-saving procedures that can be performed with minimal equipment.

There is one exception to the consistent use of First Aid in our research: the research described in Chapter 7 employed the training domain of 'In-Company Emergency Management' (ICEM). ICEM entails the application of First Aid, firefighting, and evacuation procedures, all of which are performed by a trained team of company employees. This team remains in charge until the incident is adequately handled, or when the official emergency services arrive (i.e. ambulance, firemen, police).

Both of these domains (First Aid as well as ICEM) are assumed to be sufficiently representative for complex high-risk professions. They also describe clear and accessible domains, which makes them easier to capture and comprehend. Moreover, both instructors and learners within First Aid and ICEM are relatively easy to recruit. These aspects increased the attractiveness of First Aid (and ICEM) as the chosen training domain in this research project.

Please note that even though the research described in this thesis employed First Aid (and ICEM) as the selected training domain, the resulting principles are considered to be applicable in other (complex high-risk) professions as well.

CHAPTER 3

Research Method:

situated Cognitive Engineering

Abstract - The research presented in this thesis employs the *situated Cognitive Engineering* (sCE) method. The sCE method provides a structured and systematic approach to document and compare design options with regard to their effects on 1) the experiences of people when working with the technology and 2) the joint task performance of the human operator(s) and the technology in the work place.

In sCE, users are closely involved in the design process from an early stage onward. This involvement is established through focus groups, interviews, and workplace analysis. Designs are grounded in Human Factors knowledge, and evaluated through rapid prototyping and human-in-the-loop experiments (i.e. experiments that investigate the experience, behavior, and performance of users while they are working with prototypes). The sCE method not only investigates the human-computer interaction on the interface level, but also includes investigations of the ways in which a human operator's work flow is affected by the presence of the new technology in the work place.

The sCE method guides designers throughout the design process by 1) keeping track of the design choices made and the design options left unexplored; 2) evaluating a set of design features by looking for desired effects and possibly negative side-effects thereof; and 3) backtracking to previous design choices in case a design fails to bring about the desired effects.

This chapter is partially based on the following publications:

Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "Situated cognitive engineering: the requirements and design of directed scenario-based training", in: *Conference on Advances in Computer-Human Interaction*, ed. by Miller, L. & Roncagliolo, S., Xpert Publishing Services, 2012, pp. 266–272.

Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "An ontology for integrating didactics into a serious training game", in: *Workshop on Pedagogically-driven Serious Games (ECTEL Conference)*, ed. by Bocconi, S., Klamma, R. & S., B. Y., vol. 898, CEUR Workshop Proceedings, 2012, pp. 1–10.

"A common mistake that people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools."
Douglas Adams

"Evolve solutions; when you find a good one, don't stop."
Eagleman (2011)

Personalized Educational Games (PEGs) provide a way to automate scenario-based training (SBT). PEGs employ automated behind-the-scenes control over a (virtually) simulated environment to create and manipulate storylines (i.e. scenarios) that stimulate learning. They are intended to be a tool for instructors to increase the amount of training opportunities for learners in complex high-risk task domains. So far, PEGs have merely been presented as a conceptual idea. In order to turn this idea into a concrete design proposal, this thesis investigates the design and development of PEGs guided by the *situated Cognitive Engineering (sCE) method*. The sCE method offers a novel and systematic approach to cognitive systems engineering. It is explained in the current chapter.

3.1 Design research methods - related work

Before the sCE method is presented in Section 3.2, this section elaborates on four topics that are relevant to the sCE method. The sCE method originates from cognitive engineering (Subsection 3.1.1). It addresses questions regarding the applicability and suitability of empirical versus design research methods at various stages in the cognitive system's development process (Subsection 3.1.2). Furthermore, it borrows ideas from requirements engineering and scenario-based engineering (Subsections 3.1.3 and 3.1.4).

3.1.1 Cognitive engineering

The problem addressed in this thesis is how to support the collaboration between a learner, an instructor, and a PEG, such that they can engage in (semi-)automated SBT to establish competence development in the learner. The effects of automated SBT are not just a result of the individual performance of the learner, instructor, or PEG; it is a result of their collaboration, i.e. of their *joint performance*.

A system comprised of both humans and machines, such as automated SBT, is called a *cognitive system*. Cognitive systems should be designed such that the joint performance of all parties involved is optimized. This is established by supporting the cognitive processes of the learner, instructor, and PEG. For this, it is important to acquire a clear specification of the information processing and work flow that takes place on the side of all parties involved. The reason for this is that the introduction of a new system often changes the human activity or work flow.

For example, people tend to find new ways of unintended use for newly introduced artifacts. Take for instance the unintended use of a car navigation system on a bicycle. The car navigation system may not take bicycle routes into account, resulting in major detours for the cyclist. Based on this finding, the designer may

decide to include this way of use in the design of the navigation system.

Another example of unexpected effects, due to the introduction of a new system, can be observed when a machine takes over aspects of a human's tasks. In such cases it is often difficult to predict how the automation will affect the work flow of the human operator or user (Hollnagel & Woods, 1983). By automating parts of a task, instead of 1) controlling the machine, 2) the human operator may control a process, or 3) even supervise a self-controlling process. For example, instead of 1) driving a car, a person can 2) drive with cruise control, or 3) even own a self-driving car. In the first case, the driver is actively performing the task, by manually controlling the car. In the second situation, the driver is merely operating the system on a higher level, telling the car at what speed it should drive and checking whether the car behaves accordingly. In the third case, the driver is not part of the task execution, but is only monitoring the process and needs to interfere when necessary. This example shows how the technology changes the role of the operator and, as such, the demands placed on the human operator. Such changes may result in unforeseen (side) effects: in a self-driving car the driver may become drowsy because of a lack of stimulation. In turn, the driver may wrongfully think it is safe to take a nap, which clearly results in a dangerous situation.

To design machines that are useful to the people that work with them, designers should approach the design problem by looking at the entire cognitive system, i.e. taking into account how the machine will be used within the joint human-machine task and work flow and how their joint performance can be optimized. *Cognitive engineering* sees the tasks of the user and the interactions between the user and the machine as part of the design process (Dowell & Long, 1998; Hollnagel & Woods, 1983; Norman, 1986; Rasmussen, 1986; Woods & Roth, 1988). For this, cognitive engineering aims to uncover and apply the principles underlying human performance within cognitive systems. These principles are typically adopted from human factors research, e.g. psychology, human-computer interaction, cognitive science, industrial design, and ergonomics.

Cognitive engineering also entails user-based and human-in-the-loop studies that evaluate prototypes in order to uncover any unwanted effects at an early stage of development. Prototypes are early versions of a system that are built to test a concept or process. They can be automated systems, sufficiently worked-out to be incorporated in the final system, but they can also be paper-based mock-ups of the system, or even Wizard-of-Oz prototypes in which a human mimics the system's responses to user input (Paetsch et al., 2003).

Running example

An example of a (non-automated) cognitive system is a person controlling the water flow for a bathtub with two faucets, a hot water faucet and a cold water faucet (Norman, 1986). The person wants to control a) the temperature of the water in the tub and b) the rate of water streaming from the two faucets. The physical mechanics control the individual rate of hot and cold water flowing into the tub. There is a relation between the variables the user wants to control and the physical mechanics, but the variables that can be physically controlled are not the variables the person cares about.

According to Norman (1986) there are two approaches to designing a system such that it increases the match between the goals of the user and the mechanics of the machine. The first approach is to make the mechanics of the machine more understandable for the user, for example, by adding a step-by-step instruction on how to operate the two faucets in order to reach the desired temperature. The second approach is to make the user's goals directly interpretable for the machine, for example, by creating a faucet that allows the user to directly indicate the desired water temperature. Of course, combinations of the two approaches are also possible.

Automated scenario-based training as a cognitive system

Automated scenario-based training (SBT) can be regarded as an example of a cognitive system (also see Figure 3.1). It consists of three cognitive actors: a machine (a PEG), a learner, and an instructor. These actors collaborate with each other to manipulate parts of a (virtually) simulated environment.

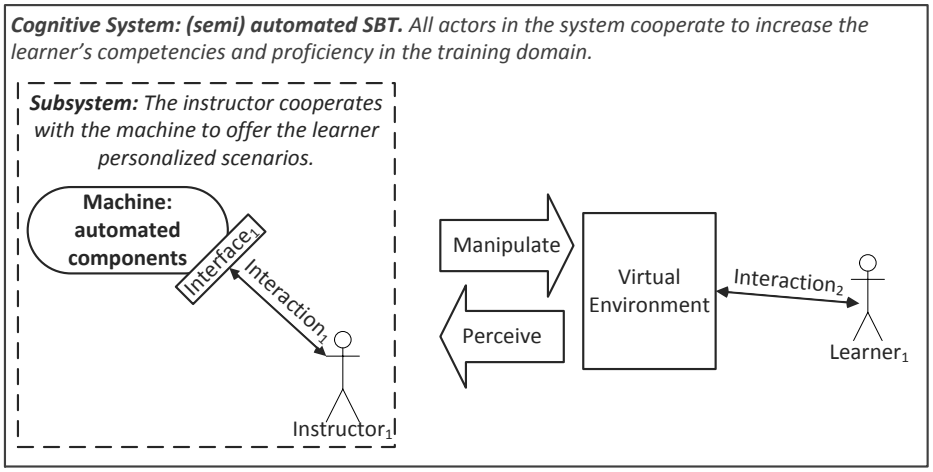

Figure 3.1: Automated SBT - represented as a cognitive system

The goal of the cognitive system (automated SBT) is to achieve a positive effect on the learner's competencies. In other words, the learner's competencies are the environmental element the system attempts to control. The learner is trying to develop his/her competencies by practicing the task offered by the instructor and the machine. The instructor and the machine are trying to come up with scenarios that will help the learner to develop those competencies. The instructor and the machine can be regarded as a subsystem; they have a somewhat different task than the learner, i.e. the dynamic creation of scenarios and execution of scenario adjustments.

The problem of designing automated SBT is how to efficiently divide the workload of instructing the learner between the machine and the instructor, how to support the collaboration between the machine and the instructor, and how to appropriately automate the tasks that need to be automated. To design automated SBT, a more thorough understanding is required of, e.g., what task the system aims to perform, how instructors currently perform this task, and how collaboration between the in-

structor and the machine can be facilitated. These are some of the problems that will be addressed in this thesis.

3.1.2 A modern approach to design research

For a long time, researchers agreed that the design of artifacts was considerably different from scientific research and required distinct approaches and methods (Tripp & Bichelmeyer, 1990). The conviction was that scientific research should aim to find principles that are valid in many situations, i.e. to uncover the truth, whereas design research was regarded to be more pragmatic, aiming to create an artifact, i.e. to apply scientific knowledge. As a result of this belief, the scientific research method and the design research method were believed to be distinct as well.

Scientific research employed the empirical research cycle described by De Groot (1966): 1) observe the world, 2) hypothesize about regularities, 3) anticipate measurable effects, 4) test the hypothesis, 1) observe the world, etc. In this sense the theory resulting from the empirical research cycle is *descriptive* in nature. The theory is considered to be a model of the world and as such the model should be adapted to the empirical findings in the experiments until it accurately describes and predicts everything that can be observed in those experiments.

In contrast, *design research* methods generally assumed a fully developed scientific theory that could be directly translated into a usable model. This model was then considered to be valid as-is and assumed to be readily applicable in the system's design. Design research in this sense only served to appropriate and evaluate existing theories in practical applications. The design process itself relied on an engineering cycle: 1) implement the model or system specification (prototype), 2) test & validate the prototype, 3) evaluate the prototype's consistency with the specification. If the prototype was not behaving conform the system specification, the prototype was adjusted and evaluated once more, etc. As such, the model employed in design research was regarded to be *prescriptive*: the resulting prototype or program should behave according to the model (specification) and if this was not the case, the program should be adapted such that it did.

Nowadays, researchers have generally adopted a more nuanced attitude towards the differences between scientific research and design research, resulting in new methods that bridge the gap between these two formerly presumed to be distinctive types of research (Edelson, 2002). These new methods emphasize that design does not merely serve the evaluation of theories in practice, but also plays an important role in *theory development*.

This view is nicely explained by Hevner (2007) as he identifies three research cycles important in design research: 1) the relevance cycle, 2) the rigor cycle, and 3) the design cycle. These cycles are depicted in Figure 3.2 and further explained below.

The *relevance cycle* first identifies and represents opportunities and problems in the application environment. Not only does this cycle result in a list of demands coming from the application environment, it also defines acceptance criteria for the evaluation. It then uses these acceptance criteria to check whether the design obtained from the design cycle actually provides a solution or improvement for the identified problems and opportunities. During these field studies, requirements may

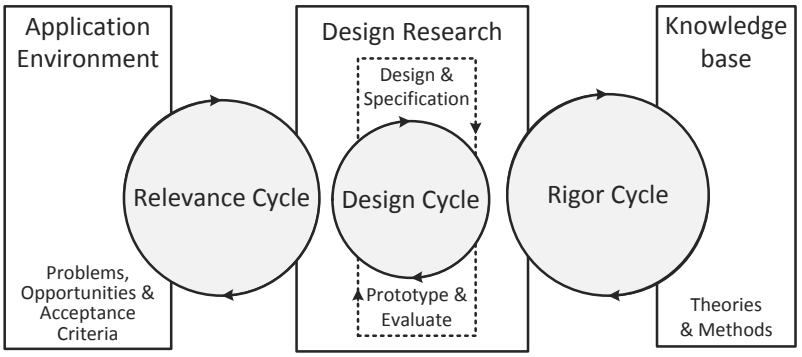

Figure 3.2: Hevner identifies three research cycles in the Information Systems research framework (adapted from Hevner (2007, p.88))

prove to be incomplete, the artifact may have deficiencies, or it may result in new problems/opportunities. In such cases another iteration of the relevance cycle starts with the feedback obtained in the previous cycle and a restatement of the research requirements as discovered from actual experience.

The *rigor cycle* connects the design science activities with the knowledge base consisting of scientific foundations, experience, and expertise. The rigor cycle provides such theoretical knowledge to the research project to ensure its innovation. The cycle also adds *new knowledge* to the knowledge base. For instance, the results of the design and relevance cycle include: 1) extensions to the original theories, 2) methods developed during the design research, and 3) experience gained from field testing the artifact in the application environment (i.e. theory development).

The central *design cycle* describes what could be regarded as the engineering cycle. This process aims to iteratively refine the artifact's design through rapid cycles of building and evaluating the prototype. The nature of this cycle is the generation of multiple design alternatives and the evaluation of the alternatives with regard to the requirements and acceptance criteria until a satisfactory design is achieved. It is important to maintain a balance between the construction and the evaluation of the artifact's design; both must be convincingly based in the relevance and rigor cycles.

Even though Hevner (2007) accurately identifies and describes these three cycles, he does not provide a thorough methodology to guide designers through these three cycles. The situated Cognitive Engineering method accommodates this modern three-cycle approach to design research, extends it to the field of cognitive engineering, and adds two additional constructs (requirements and claims, also see Subsection 3.1.3 and 3.1.4) resulting in a well-integrated methodology that offers strong guidance for the design and research cycles.

On the evaluation of designs

To verify that a design actually meets the acceptance criteria and results in the intended effects, the design should be validated. For the validation of a (cognitive) system design, three development stages should be taken into account (Hollnagel & Woods, 1983).

In the first stage, a well-studied, validated, and relevant principle is selected from the (human factors) knowledge base. This principle is used as the foundation for a candidate design feature and comes from the rigor cycle. Subsequently, the design feature is implemented in a prototype during the design cycle. This prototype in turn is evaluated within the application environment, which is done during the relevance cycle (Hollnagel & Woods, 1983). Each of these steps may introduce misinterpretations or false translations, resulting in uncertainty as to whether the intended effects of including the design feature will actually be secured by the system's implementation.

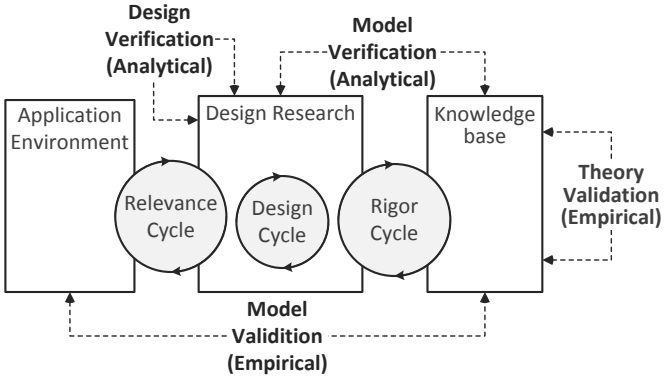

Figure 3.3: The relation between the different verification and validation steps and Hevner's research cycles (adapted from Hevner (2007, p.88) and Hollnagel & Woods (1983, p.596))

In Figure 3.3, various validation and verification steps are related to Hevner's research cycles. The steps are used to verify and validate the process of successfully incorporating human factors knowledge into a system design.

We will first explain each of the steps by means of the water faucet example. Imagine that the water faucet is to be used by people who may not be able to understand how to use the faucet immediately. The faucet is to be extended with a design feature that supports these users in the correct use of the faucet. The envisioned solution is the addition of a spoken instruction placed next to the faucet to support its correct use.

Theory validation - The support for the envisioned design feature comes from multiple studies that indicate the benefits of step-by-step instructions to support task execution. *Theory validation* entails making sure that an idea, concept, or theory is well-studied and has been sufficiently validated in previous experiments, before adopting it into the design.

Model verification - A step-by-step instruction is devised for operating the double faucet. This is a new application of the step-by-step instruction principle, requiring an investigation of how to operate the faucet and how to clearly explain this procedure in a step-by-step instruction. *Model verification* is arguably the hardest step in the design process: one has to make sure that the adopted theory is appropriately transformed into a usable design feature. It may be that the theoretical principle has not yet been applied within a context comparable to the envisioned system. In such cases it may be necessary to study the principle within the application context

before adopting it into the system's design or turning it into a design feature.
Design verification - The step-by-step instruction is implemented in a prototype, making it available for use while operating the faucet. This implemented prototype of the instruction may take on many different forms, ranging from a person standing next to the faucet - vocally providing the step-by-step instructions as the user operates the faucet - to an audible smart-watch application that guides the user through the procedure. During the *design verification*, the developer of the prototype ensures that the prototype behaves as described in the original specification.
Model validation - The prototype is evaluated by monitoring users as they use the faucet with the new design feature. If the design feature (step-by-step instructions) is implemented correctly, the same results should be found as in the original studies that led to the adoption of the design feature, i.e. people should be able to operate the water faucet even without previous understanding, experience, or knowledge. Hence, during *empirical validation*, the prototype is evaluated by looking for the intended effects of the adopted human factors principle within the context of the application environment. If all goes well, results obtained from the prototype evaluation within the application environment should correspond to the results obtained in previous studies.

Please note the distinction between verifications and validations. Verification entails the analytic establishment of a correspondence between two versions of the same construct. In contrast, validations entail the traditional testing of hypotheses through experimental designs.

Ultimately, the effects that were obtained in the original theoretical validation should be reproduced in the empirical validation, after implementing the design feature in the prototype. Indeed, this would be the only true justification for incorporating the design feature in the system specification; the intended effects were the reason for adopting the design feature in the first place. However, the concurrence of results obtained from the empirical validation and the theory validation is not always guaranteed. The original manipulations applied in previous studies have been altered through the steps described above to produce the resulting design feature as implemented in the prototype. Until this last validation step is performed, the design can only be *assumed* to result in the intended effects based on the other three validation/verification steps.

In the running example of the water faucet, the focus is on one single design feature. However, complex cognitive systems usually entail a large combination of design features. In fact, a complex cognitive system is generally of such a size that it cannot be evaluated as a whole in one single experiment (Westera et al., 2010). Instead, the design specification is divided into manageable pieces such that they can be evaluated in separate experiments. Because of all the interdependencies between the various design features, this partitioning is not a trivial task. And neither is the level of detail to which this partitioning and testing should take place. Segregating the design into single design features is not feasible nor desirable, because all interdependencies are lost. On the other hand, it is also not useful to evaluate the system as a whole because there are too many interdependencies to draw any unambiguous conclusions.

3.1.3 Requirements engineering

Requirements engineering in cognitive systems is directed at specifying requirements. Requirements describe exactly what is needed for the design of a new or altered product: what function should the product fulfill? According to Wiegers (1999), requirements should be correct, feasible, necessary, prioritized, unambiguous, verifiable, complete, consistent, modifiable, and traceable. Figure 3.4 depicts the role of requirements within the three cycles of design research: requirements describe the demands coming from the application environment thereby specifying the problem space for the design cycle.

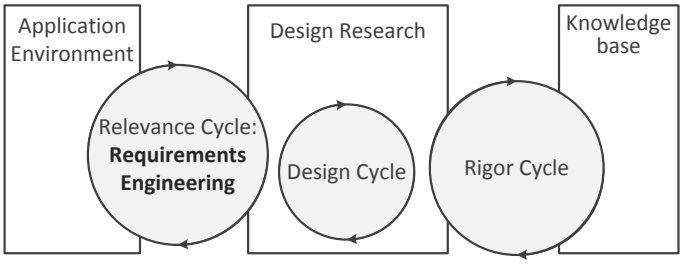

Figure 3.4: The relation between requirements engineering and Hevner's research cycles (adapted from Hevner (2007, p.88))

Requirements can be documented in 'use cases'. Use cases provide short descriptions of user-system interactions to convey how the system interacts with a user (or another system) to achieve a specific goal. Use cases typically employ the language of the end-user or domain expert. A use case describes how a user is envisioned to work with the system, by specifying the steps needed to perform a task.

Requirements can be elicited in multiple ways (Paetsch et al., 2003). Examples of requirement elicitation methods are: interviews with potential users and stakeholders; the construction of scenarios and use cases; observation and social analysis in the work environment; focus groups; brainstorming sessions; and rapid prototyping.

Ross & Schoman (1977) propose the development of a functional architecture based on the requirements. In this context, *functional* implies that the architecture specifies segregated parts of the system that satisfy a particular subset of the requirements. Such segregated parts are referred to as *functional components*. The functional architecture describes the algorithmic nature of the system: the input and output of functional parts of the system. It provides a rigorous layout of the activities performed by the system and the way components interact with other components and users of the system. A functional architecture also offers a way to partition the evaluation of a complex cognitive system. By dividing the system into smaller functional components, these components can be evaluated separately.

3.1.4 Scenario-based design

In scenario-based design the intended or envisioned use of the future system is concretely described at an early stage in the development process by means of narra-

tives, i.e. *scenarios* (Carroll, 2000; Go & Carroll, 2004). Scenarios have characteristic features. First of all, scenarios include at least one user who is trying to achieve at least one goal, i.e. a change the user wishes to achieve. Furthermore, scenarios always encompass a setting, e.g. the user's role, the system's functions or features employed by the user, and the way functions and features are presented to the user. Also, scenarios specify sequences of actions and events: things that users do and things that happen to them, e.g. changes in the circumstances of the setting. The scenarios describe the way actions and events interact with the achievement of the goals. Scenarios make the use of the system explicit, thereby prompting designers to focus on the - often implicit - assumptions about people (users) and the way they perform their tasks.

Carroll & Rosson (1992) propose an incremental process of writing scenarios and, subsequently, identifying the underlying positive and negative claims, i.e. consequences of adopting particular design features on users and other stakeholders. The claims identified in a scenario can be iteratively used to derive new scenarios. This process eventually results in an ever more precise system design. The identified claims should be justified, either by scientific research in relevant fields, such as psychology and human-computer interaction, or by argumentation and logic.

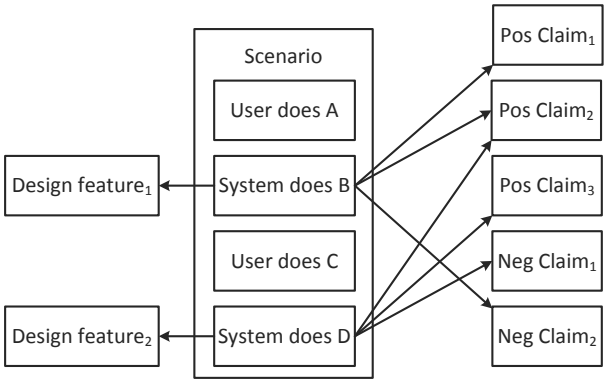

Figure 3.5: The relation between scenarios and claims

As is depicted in Figure 3.5, there is a linkage between: 1) concrete and contextualized scenarios; 2) applied and developed design features; and 3) abstract and generic claims. This linkage extends the design features with a design rationale: design features describe how the system responds to the user. The decision between multiple options for design features can be guided by looking at the expected consequences or claims documented in the scenarios (Carroll & Rosson, 2003). Moreover, each design feature can be traced back to its original expected effects. Since the expected effects (i.e. claims) were the reason for adopting the design feature, claims offer a way to *justify* the incorporation of design features. Design rationale is the analysis and documentation of a system's intended use, described in scenarios, and the associated claims. It supports the decision-making process underlying the design choices.

Design rationale has the potential to evolve into a design science (Carroll & Rosson, 2003). Specific design descriptions can be abstracted and generalized by categorizing designs. Moreover, the features that define the categories can be associated with general consequences for users and their task performance. As a result, a theory can be developed by specifying under what circumstances which design features produce which types of effects, i.e. theory development in the rigor cycle.

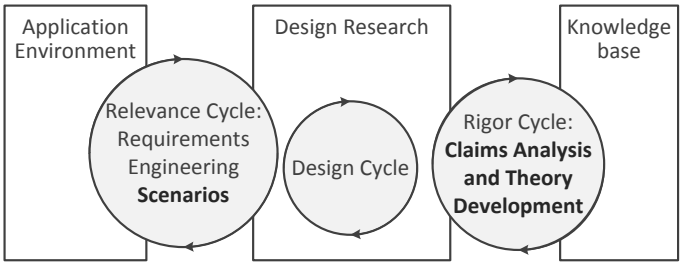

Figure 3.6: The relation between claims analysis and Hevner's research cycles (adapted from Hevner (2007, p.88))

The sCE method adopts ideas from requirements engineering, claims analysis, and scenario-based engineering and integrates them in the design research cycles. Figure 3.6 depicts the integration of the four notions discussed throughout this section and shows how they can be combined into a more specific design research method. The resulting sCE methodology is explained more thoroughly in the next section.

3.2 Situated cognitive engineering

The Situated Cognitive Engineering (sCE) method carefully intertwines the three identified research cycles discussed in the previous section and adopts elements from requirements engineering and claims analysis resulting in an integrated design process to come to an elaborate system specification. As such, the sCE method is believed to provide guidance for the design process, supporting all of the three cycles in an integrated fashion. The sCE method is compliant with ISO standards 9241-210 'Ergonomics of human-system interaction'.

The sCE method places the design process within the scope of the domain and the intended application environment. The method prescribes an iterative, incremental development process; each (fast) cycle is directed at refining the system's specification in more detail. The system specification is an organic whole consisting of 1) use cases, 2) requirements, 3) claims, and 4) an ontology. Together, these elements describe the system's behavior, functions, expected effects, and underlying conceptual knowledge respectively.

3.2.1 The four elements of the system specification

This subsection will explain the four elements in the system specification using the previously mentioned example of the hot and cold water faucets that should be

combined in order to obtain the desired water temperature. The resulting system specification is also depicted in Figure 3.7.

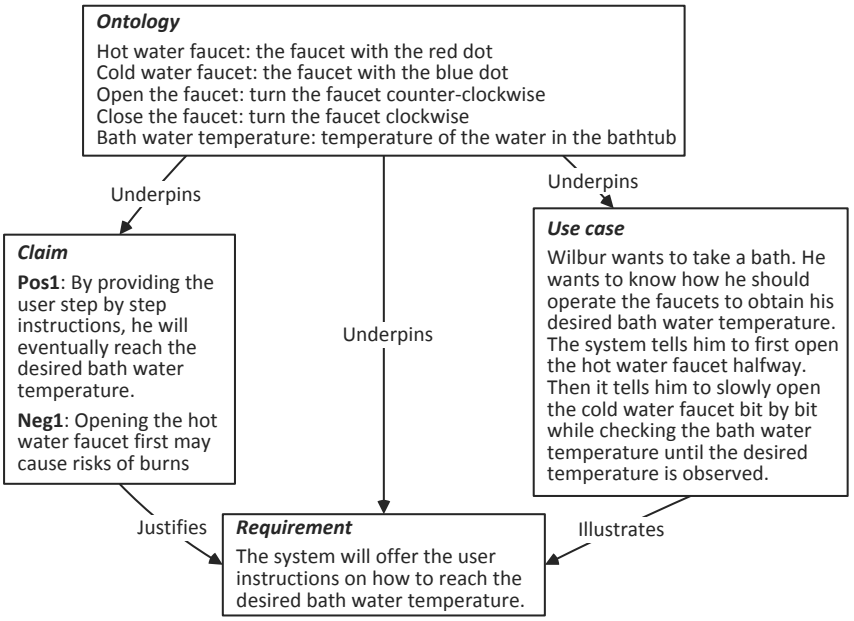

Figure 3.7: An example of a feature specification

The *use cases* in the system specification provide stepwise descriptions of envisioned interactions between the system and the user. In that sense, use cases are comparable to 'scenarios' as described by Carroll & Rosson (2003). In the example in Figure 3.7, the use case describes how a person named 'Wilbur' employs the system in order to operate the faucets and obtain his desired bath water temperature.

The *requirements* are specifications of the system's functions, such as 'The system will offer the user instructions on how to reach the desired bath water temperature'. Requirements can be grouped into core functions to identify functional components within the system, such as a user model that keeps track of the user's relevant characteristics and preferences. The requirements and functional components can be regarded as design features: they specify what the system will be able to do and/or what it will offer the user.

The *claims* in the system specification are descriptions of the expected effects or implications of the requirements. Claims refer to positive as well as negative effects of adopting a particular requirement. There is a distinction between the role of positive claims and negative claims. Positive claims are an indication of the desired effects, i.e. the effects that should be identified during the evaluation of the design in the application environment. As long as the positive claims cannot be validated (*empirical validation*), the addition of the design feature cannot be justified. Negative claims, on the other hand, can be regarded as possible problems occurring after adopting the specified design feature. As such, evaluations of the system's design

choices should also investigate the presence of these negative effects. If the negative effects turn out to be non-existent, no action is required. However, if these negative effects are indeed observed in the evaluations, the system's design should be adapted in such a way that either 1) an alternative design feature is chosen that results in the same positive effects but not in the negative ones, or 2) additional design features are adopted that aim to reduce or overcome the negative effects of the design feature. Part of the design challenge is that multiple solutions exist for the same problem and it is not always clear which solution is the optimal one. Hence, negative claims should be part of the system specification in order to guide decisions among design options.

Lastly, the *ontology* is the specification and formalization of the bodies of knowledge relevant to the system. More specifically, the ontology defines and relates all the concepts mentioned in the use cases, requirements and claims. As such, the ontology warrants consistency between the other three elements. For instance, in the example the term 'hot water faucet' is defined as the faucet with the red dot (in contrast to defining the hot water faucet as the left-hand faucet, which might also serve in most occasions). The meaning and definition of each term in the ontology depends on the application environment in which it will be used. Each time the same concept is mentioned in the system specification, it refers to the meaning specified in the ontology, resulting in its consistent use and explicit meaning.

In a way, the ontology describes the *model* of relevant domain knowledge employed by the system. As described previously, concepts and theories from human factors need to be translated into a model that can be employed in the system's design. It is the *model* that is employed by the system to reason about the user and to decide how the system should respond to that user. A clear documentation in the form of an ontology makes the models derived from human factors knowledge explicit, so that it can be verified by domain experts. As such, the ontology also facilitates *model verification*.

3.2.2 Incremental development process

The sCE method prescribes an incremental development process that results in an ever more detailed system specification. Incrementality is generally a part of all cyclical processes, but in this case incrementality is especially important due to the partial unpredictability of the interaction between the newly designed system and the user that operates it. The development process of the system specification goes back and forth between three stages: the task domain and support analysis (TD&SA), the system specification, and the refinement and validation (also see Figure 3.8).

The system's *initial* specification follows from the first TD&SA, consisting of three major parts: 1) the task domain analysis, 2) the human factors analysis, and 3) the technological opportunities analysis.

The *task domain* analysis concerns the application domain. Usually this step includes interviews and brainstorm sessions with end-users, or field studies (e.g. on-the-job interviews, shadowing, training participation). In general, the task domain analysis employs task analysis methods, such as Hierarchical Task Analysis or Cognitive Work Analysis (Annett, 2004; Annett & Duncan, 1967; Militello & Hutton, 1998;

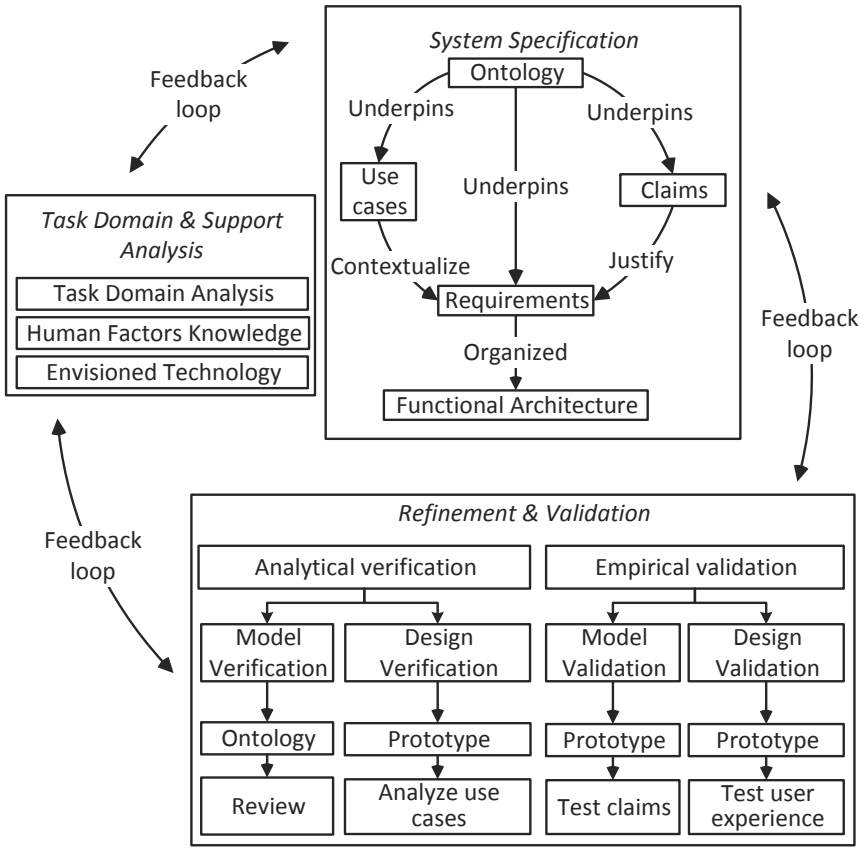

Figure 3.8: The situated Cognitive Engineering method structures the incremental research and development of the system's specification (Neerincx & Lindenberg, 2008)

Salmon et al., 2010; Schraagen et al., 2000; Stanton, 2006) to construct an initial set of user requirements, accompanying claims, use cases, and important concepts to be documented in the ontology.

The *human factors* analysis investigates relevant theories, guidelines and support concepts through literature research and consultations with domain experts (e.g. HCI researchers or educational psychologists). This step aims to refine the set of requirements, claims, concept definitions, and relations among those concepts.

The exploration of *technological opportunities* results in a range of possible technological solutions for the identified problem.

Each achieved set of system specifications is to be empirically refined. This refinement is reached through evaluations, which can be performed at various levels of the system's design, as explained earlier in Figure 3.3. Below, we provide several examples of system evaluation methods.

Model verification - Conducting literature research, expert reviews to verify a correct translation of the human factors principles to the model employed by the sys-

tem.
Design verification - Verifying that the system behaves as intended in the original design. This is established by comparing its behavior to the behavior described by the use cases in the system specification.
Model validation - Conducting user-based studies and human-in-the-loop experiments to verify the claims that are a direct consequence of the adopted model.
Design validation - Presenting potential users with use case simulations, performing user-based studies with prototypes, performing human-in-the-loop studies, and interviewing users all in order to investigate the system's usability and to investigate and compare various design proposals.

As can be seen from these examples, the system specification facilitates the various types of evaluations. Each function or *requirement* is added because of its expected (overall beneficial) effects described in the claims. This means that if the function is well implemented, the accompanying claims should be confirmed through experimental validations of the prototype. Results from such evaluations feed back to the system's specification, and to the 'Task Domain and Support Analysis': the knowledge obtained through such evaluations concerns more than the system's design, it concerns underlying claims as well, thereby informing us of the effects of certain more generic functions and conceptual constructs as described previously when discussing the work by Carroll & Rosson (2003), i.e. theory development.

The incrementality of the development process results in the reuse of valid use cases, requirements, and claims in subsequent cycles of the design process. As a result, the development of the use cases, requirements, claims, and ontology are parallel processes: it is possible, even necessary, to move back and forth between the four elements that make up this specification, since each of these elements fuels the specification process of the other ones.

3.2.3 Functional architecture

Along with the system specification, the sCE method prescribes the development of a functional architecture. The functional architecture groups requirements into core functional components. By specifying the input and output parameters for each of the components, they can be studied in isolation from the rest of the system. As long as the rest of the system is able to 1) produce the required input for the component and 2) process the output produced by the component. As such, the functional architecture should clearly specify the in- and outputs required by each of the functional components to warrant the possible reassembly of the components after their individual development.

3.2.4 Relation to the three design research cycles

From the explanation of sCE provided above, it is assumed to be entirely clear how cognitive engineering, requirements engineering, and claims analysis relate to the sCE method. However, the previous section also described a three cycle design process and announced it to be of particular importance to the sCE method. Yet, it may not be entirely conspicuous where the rigor, relevance, and design cycle can be found in the sCE method described above. This subsection will provide an explana-

tion of this relation:
The relevance cycle can be recognized in the analysis of the task domain during the TD&SA, the requirements as part of the system specification, and the model and design validation in the refinement and validation stage.
The rigor cycle can be recognized in the human factors analysis during the TD&SA, the ontology and claims in the system specification, the model verification during the refinement and validation stage, and the feedback loop to the human factors knowledge (i.e. theory development as a result of the empirical validations).
The design cycle can be recognized in the exploration of available and envisioned technology during the TD&SA, the use cases in the system specification, and the design verification and validation during the refinement and validation stage.

By intertwining these cycles, the sCE method results in an incremental design process to achieve an elaborate system specification, while supporting all of the three cycles in an integrated fashion.

3.2.5 Previous applications of the sCE method

The sCE method has been successfully applied in multiple application domains. Examples of previous applications are: a task support system in naval ships (Neerincx & Lindenberg, 2008); a system that teaches social conventions (Schouten et al., 2013); a support system for astronauts during long-term missions (Neerincx et al., 2006; Neerincx et al., 2008); a support system for human-robot team performance in complex tasks (Mioch et al., 2012); a support system for urban search and rescue missions (De Greef et al., 2009); a support system for technical crews on board of naval vessels (Grootjen et al., 2009); the development of error-predicting cognitive models in the domain of aviation (Van Diggelen et al., 2011); and a support system for elderly living at home (Spiekman et al., 2011). A complete overview can be found in Neerincx (2011).

3.3 A note on the ordering of the chapters

The remaining chapters have been ordered according to their role in the design and development process of PEGs (see Figure 3.9). Part I describes the general system specification and functional architecture. Part II describes a more detailed investigation of each of the functional components identified in the system's functional architecture. The ordering of the chapters within these two parts will be further explained below.

The sCE method prescribes a design research process that goes back and forth between three stages: Task Domain & Support Analysis (TD&SA), System Specification, and Refinement & Validation. It starts with an initial TD&SA, encompassing a task domain analysis, an exploration of relevant human factors literature, and envisioned opportunities for technological solutions. Therefore, Part I starts with a presentation of the results of the (incremental) TD&SA in Chapter 4. The TD&SA leads to the specification of the requirements, i.e. the demands coming from the application environment, and the claims, i.e. the desired effects of particular design features. Chapter 4 also describes the functional architecture consisting of func-

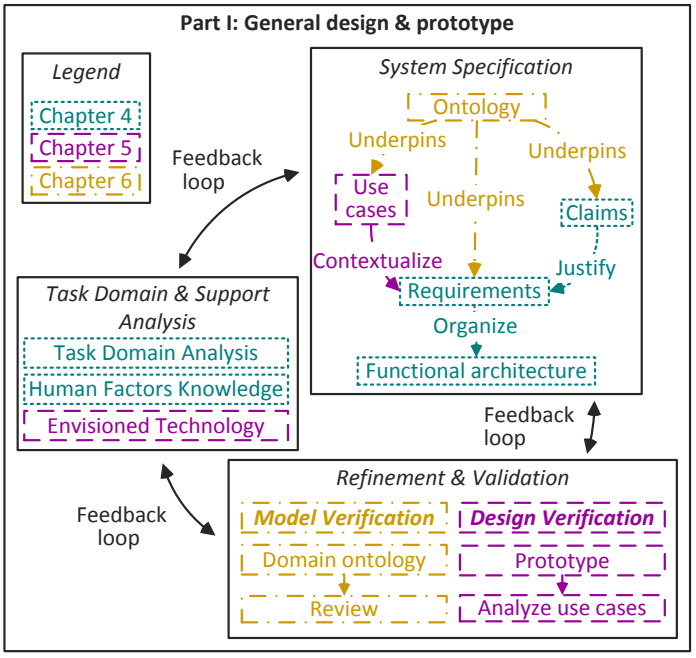

Figure 3.9: A graphical overview of the first part of the thesis. Each chapter discusses some part of the general system specification and the results thereof.

tional components that each address a subset of the requirements. Chapter 5, in turn, presents a multi-agent system that supports the functional architecture and produces the intended system behavior. Moreover, the resulting system was implemented and verified by comparing the prototype's behavior to several use cases. Chapter 6 presents the ontology, its design process, and its refinement based on expert reviews (i.e. model verification).

The second part of the thesis (see Figure 3.10) concerns the functional components identified in Chapter 4. Due to the chosen ordering on the chapters, the research reports are not always in chronological order. The work described in Part I includes the most recent results obtained throughout the research project, whereas the work described in Part II presents earlier work on the functional components and the results of the respective prototype evaluations. Hence, the outcomes of the work described in Part II have been interpreted and incorporated in the work described in Part I. The non-chronological order of the chapters is not considered to be a problem as long as this is clear to the reader. Figure 3.11 presents a timeline that illustrates the chronological order in which the research has been conducted.

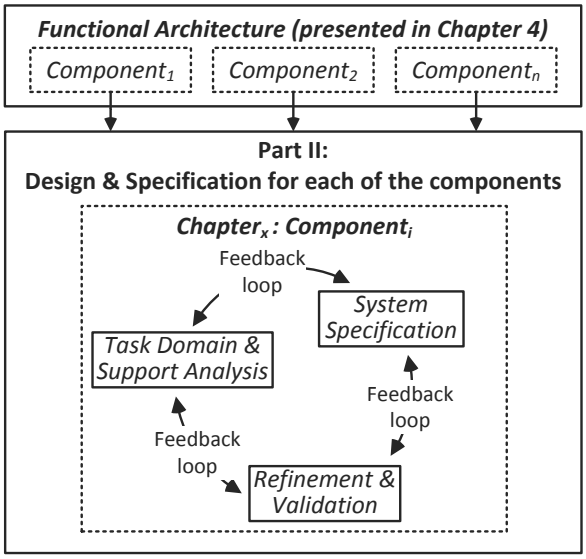

Figure 3.10: An overview of the second part of the thesis consisting of Chapters 7–10. Each of these chapters discusses the specification of a functional component.

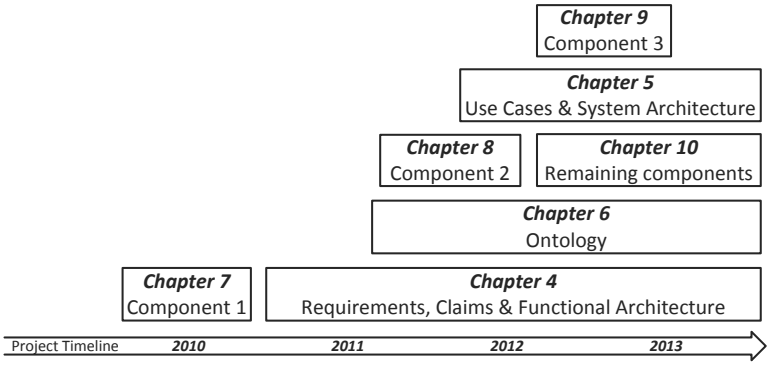

Figure 3.11: The chronological order in which the research has taken place.

Part I: General design & prototype

The current part of the thesis describes the design process of automated scenario-based training, resulting in the design proposal called 'Personalized Educational Games' (PEGs).

Chapter 4 describes an investigation of SBT and the functional requirements for automating SBT. The investigation consists of a literature review on the concept of SBT, interviews with instructors who use SBT in their training programs, and a consultation of the human factors literature on learning & instruction. This investigation results in the specification of a functional architecture for 'Personalized Educational Games' (PEGs).

Chapter 5 continues with an investigation of the intended system behavior by constructing and analyzing several use cases. The use case analysis leads to the identification of technical requirements for the PEG architecture. A review of the literature on intelligent agents suggests that a multi-agent organization may provide a suitable solution to meet the identified technical requirements. As such, a multi-agent organization is designed and implemented in a prototype. The behavior of the prototype is then compared to the original behavior specification (i.e. the use cases).

Chapter 6 addresses the problem of the didactic knowledge that should be made available to the system in order to support automated didactic reasoning. The use of an ontology is proposed to capture and document the required knowledge. The ontology consists of 1) a domain representation of scenario-based training in games and 2) a definition of the concepts employed in the technical design. The ontology is implemented in Protégé Frames (3.5). The design process of the ontology is also addressed in this chapter.

Part I of the thesis results in three deliverables: the functional architecture (PEG), the agent organization, and the ontology. Together, these three constructs are believed to provide a flexible, modular, and reusable design for the development of automated SBT, and PEGs in particular.

CHAPTER 4

Requirements & Claims:

Functional system specification

Abstract - In order to automate scenario-based training (SBT), a detailed description is needed of SBT and its foundations in cognitive and educational psychology. This chapter starts with an investigation of SBT consisting of (1) a literature review on SBT, (2) a series of interviews with SBT instructors, and (3) a literature review on relevant Human Factors studies and theories. A list of functional requirements is derived from this investigation, along with the desired effects, and possibly negative side-effects, to justify each of the requirements. Such positive and negative effects are called *claims*.

For example, one of the identified functional requirements for automated SBT is as follows:

`(R1) Learners must be offered exercises in the form of complete storylines.`

To evaluate the effects of this requirement, a prototype of the system should be tested for the following claim:

`(PC1) This results in improved integrated task performance`

In addition, to investigate whether this requirement perhaps results in negative side-effects, the following claim should also be investigated:

`(NC1) This may result in high cognitive load`

The total list of functional requirements is organized into a functional PEG architecture consisting of six components: 1) the (virtual) simulated environment, 2) the scaffolding component, 3) the scenario creator, 4) the authoring tool, 5) the learner model, and 6) the reflection component. Some of the functional requirements appear to introduce a conflict or *trade-off*, meaning that satisfying one requirement may be at the cost of satisfying another requirement. The biggest challenges resulting from such trade-offs are discussed for each individual component.

This chapter is partially based on the following publications:

Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "Situated cognitive engineering: requirements and design of directed scenario-based training", Unpublished peer-reviewed proceedings of the *International Workshop on Authoring Simulation and Game-Based Intelligent Tutoring* (AIED Conference), 2011.

Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "Situated cognitive engineering: the requirements and design of directed scenario-based training", in: *Conference on Advances in Computer-Human Interaction*, ed. by Miller, L. & Roncagliolo, S., Xpert Publishing Services, 2012, pp. 266–272.

> "Technology has influenced - and will continue to do so for the foreseeable future - the design and delivery of training systems. [...] It is encouraging that basic and applied research is currently going on to uncover how these technologies enhance learning and human performance. Specifically, we need to know more about how to best present knowledge [...], how and when to provide feedback, which instructional strategies are best [...], what role instructors and trainees play in these modern systems, and how effectiveness can best be evaluated."
>
> **Salas & Cannon-Bowers (2001, p.490)**

The problem addressed in this thesis is the design of a (partially) automated system for scenario-based training (SBT) in the form of a personalized educational game (PEG). This chapter presents the first steps in the PEG's design process. It starts with an investigation of the concept of SBT and the way it is performed by instructors (Section 4.1). Thereafter follows a review of the human factors literature to identify the requirements and associated claims of SBT (Section 4.2). The final part of the chapter, Section 4.3, presents a functional PEG architecture, describing a PEG's functional components and their interactions.

4.1 Task domain - scenario-based training (SBT)

Personalized Educational Games (PEGs) provide a semi-automatic (virtual) simulated environment for conducting SBT. To specify the demands of SBT, the literature on SBT was studied and a series of interviews was conducted.

4.1.1 The concept of scenario-based training

SBT is a practical training form that is often used for professions as can be found in the police force, fire department, aviation, and medical services (Oser et al., 1999; Salas et al., 2006) (also see Chapter 2). In SBT, learners prepare and execute missions or tasks that are typical and/or critical for their future line of profession in the form of interactive role-playing exercises (*scenarios*). Scenarios are generally staged within a simulated environment (Cannon-Bowers et al., 1998; Peeters et al., 2012b; Van den Bosch & Riemersma, 2004). One could say that in SBT, the scenario is the curriculum (Oser et al., 1999).

Instructors author and prepare scenarios in advance of training to ensure *representativeness* and *didactic value*. Scenarios are designed such that they systematically call upon the competencies formulated in the learning goals. The events in a scenario are tightly linked to the underlying learning goals. This is established as follows. Events in the scenario result in situations that will evoke the need for the learner to perform a particular task. So, events are introduced to provide the learner an opportunity to correctly handle a given situation by performing the appropriate task (Fowlkes et al., 1998). Learners are encouraged to reflect upon their own and each others' performance afterwards. They receive additional feedback from their instructor when important remarks are missing from the conversation during reflection. Predetermined performance measures enable instructors to objectively assess the learner's task performance and to provide directed feedback afterwards (Oser et al., 1999; Van den Bosch & Riemersma, 2004). Performance measures not only register whether the learner is improving, but also offer possibilities for additional

diagnosis, like the detection of misconceptions, knowledge gaps or lacking skills. To facilitate individual assessment for each of the events in the scenario, it is important that events in the scenario produce independent performance outcomes.

Typically, SBT takes place in a simulated environment. Most often these environments involve non-player characters with whom the learner interacts, such as teammates, officers, patients, or opponents. These roles are usually played by staff members and actors. Training within a simulated environment has several benefits in comparison to on-the-job training: a simulated environment can be prepared, controlled, reused, and improved upon, leading to the reduction of risks, costs, and resources. The competencies obtained through training scenarios should transfer to the actual task environment. Therefore, the simulated environment and the scenarios need to be representative for the task (Baldwin & Ford, 1988; Young, 1993).

Transfer is especially important in the inductive learning approach of SBT, where learners are offered the possibility to discover general principles through the engagement in multiple scenarios. Learners may need to participate in many scenarios in order to recognize the patterns. In SBT, the content, flow, causal relations, and coverage all need to be representative for real life situations in order for the learner to pick up the important aspects and patterns. The biggest challenge then is the systematic construction or collection of accurate examples (scenarios) that "cover" similar principles or patterns.

4.1.2 Scenario-based training in practice

Because the concept of SBT may differ from the practice of SBT, a series of interviews was conducted with domain experts: instructors who regularly train people by means of storylines within simulated environments. These interviews consisted of eight individual and five joint interviews with a total of twenty-four instructors in three safety domains, three health care domains, and one organizer of live action role plays in the entertainment domain. In addition, training sessions in three different domains were attended. The training domains investigated in these interviews are described in more detail in Appendix A.

The findings reported below are a result of interviews conducted throughout the entirety of the four-year research. This means that the goal of the interviews differed somewhat between interviews, depending on the stage of the research process at that time. Interviews conducted at the beginning of the research were more explorative in nature, asking instructors about their preparation of a training session, the activities during a training session, and the wrap-up after a training session. Later interviews tended to be more specific, and were directed at isolated topics such as: the process of authoring a scenario, the actual performance of the scenario, the type and level of control instructors could exert in realtime, or the envisioned integration of PEGs into the current training curriculum. Examples of interview guides can be found in Appendices B and C.

Results of the interviews

Instructors in the interviews described SBT in practice as follows:

Curriculum - Instructors explained that most of their training programs employ a fixed curriculum. They reported that curricula generally address learning topics of

increasing complexity as the program progresses, starting with standard procedures and gradually moving on to exceptions and special circumstances. Furthermore, instructors mentioned that the curriculum is announced in advance, so the learners know what to expect from each training session.

Didactic suitability - Instructors reported the following aspects to be important for the preparation of training scenarios. They start by checking the curriculum to see what learning goals are applicable to the session. Next, they think of situations that require the learner to perform a particular task belonging to the set of selected learning goals. Instructors explained that the situations to be addressed by the scenario should be integrated into a complete storyline. They emphasized that the resulting scenario should offer the learner the possibility to focus on the learning goal and to achieve it. Therefore, instructors try not to add unnecessary distractions or simultaneous tasks. They indicated that the scenario is ready for use when 1) the storyline is suitable for the learning goal and 2) the scenario does not pose any unnecessary complications for the learner.

Representativeness - Instructors emphasized that scenarios must be representative to facilitate the accurate recognition of distinguishing features and cues in real life situations. For example, in First Aid, most details are concerned with the inflicted injury or illness. So if the victim is having a heart attack, it is important that the victim accurately displays the right symptoms (shortness of breath, chest pains, etc.). In addition, instructors mentioned the need for causal links between events to teach learners decision-action-consequence relations, e.g. when talking calmly, the victim will feel reassured, when failing to place an unconscious victim in the recovery position the victim will suffocate in his/her own tongue. In order to ensure that the storyline is representative, instructors explained that they create branching storylines, i.e. storylines that allow for multiple ways to unfold depending on the actions of the learner. As a result, the actual unfolding of the storyline during the enactment of the scenario depends on 1) the decisions and actions of the learner (e.g. the victim improves as the learner performs the right actions) and 2) the effects of time (e.g. the victim deteriorates as time passes).

Variation - Instructors also reported that it is not sufficient to provide learners with one single scenario for a given learning goal. They emphasized that, looking at the entire training program, learners should encounter variations of situations, thereby offering them a variety of cues for task performance. For example, not every heart attack has the same symptoms, but the learner should learn to recognize and diagnose the symptoms, and execute the correct procedure either way.

Controlling complexity - Instructors claimed that the level of complexity of a scenario is usually determined by the learner's decisions. In other words, instructors generally do not invoke adjustments to the level of complexity during the scenario enactment. In some occasions, the instructor is able to issue, delay, or skip certain events in the scenario to create more suitable situations when necessary. However, instructors emphasized that interruptions in the scenario flow should be restricted to a minimum. During the enactment of the scenario instructors do not take over the learner's tasks, nor do they tell learners that they are making a mistake, nor do they instruct the learner on what to do next. Instructors explained that such interruptions might break the immersion or the learner's flow of thoughts. Moreover,

uninterrupted continuation of the scenario offers the learner the chance to also perform some parts of the scenario correctly, which, according to some instructors, fosters a feeling of confidence and competency. Any prompts, hints, or relaxations on constraints (e.g. time pressure) are preferably built into the branching storyline in advance. This means that the dynamic unfolding of the scenario takes place in a natural way through the events already part of the scenario and/or the behaviors displayed by the non-player characters.

Reflecting on the performance - Instructors explained how, after the scenario has finished, they prompt the learners to reflect on their performance. Subsequently, instructors find it helpful to also ask other people to reflect on the scenario (e.g. teammates, role players, observing classmates). If instructors feel that the reflections of the other participants were insufficient, they provide the learner with additional feedback. This feedback, according to the instructors, is formulated in relation to measurable and observable performance standards.

4.1.3 Demands for the (automated) SBT

From literature on SBT and the interviews concerning SBT in practice, we deduced the following preliminary demands regarding the functional design of PEGs:

01. Storylines
Scenarios must encompass a complete narrative, starting with a clear beginning and finishing with a distinctive ending.

02. Interactivity
Scenarios must be interactive, so the learner can participate in the storyline.

03. Representativeness
The simulated environment and the scenarios must be representative for real-life situations.

04. Variation
Over the course of training, the learner must encounter variations of tasks and task situations.

05. Curriculum
Scenarios must be linked to learning goals associated with performance standards. The curriculum must be available to the learners in advance.

06. Increasing complexity
Scenarios must increase in complexity as learners progress in the curriculum.

07. Independence of events
If a scenario includes multiple events for the learner to respond to, the assessment of the learner's performance on these events should be independent.

08. Minimal distractions
Scenarios must offer the learner the opportunity to successfully focus on and obtain the learning goal without any unnecessary distractions.

09. No interruptions
Hints, prompts, and other forms of support should not disrupt the learner's performance.

10. Reflection
Learners must be stimulated to reflect on their own as well as on others' performance. Confusion, misunderstandings, or misconceptions must be addressed

during reflection.

11. **Observe performance**
    Learners must also have the opportunity to observe others as they perform a task.
12. **Feedback**
    Learners must receive feedback regarding their performance on the learning goals based on measurable performance standards.
13. **Instructor control**
    Instructors must be able to exert control over the preparation and evaluation of scenarios.

This is a preliminary list: some demands are derived from the literature about the concept of SBT, others come from the interviews with instructors. Several demands need further specification before they can be adopted into a functional PEG architecture.

4.2 Human factors knowledge

The current section presents a review of the human factors literature for each of the identified demands of SBT. The goal is to validate the demands and to specify the expected effects (*claims*) of the demands.

4.2.1 Storylines

The concept of SBT revolves around storylines. SBT exercises all follow a narrative structure. Narrative structure is beneficial for learning because it guides human attention and helps to draw relations between (learning) topics in a natural and understandable way (Dickey, 2006; Mott et al., 1999; Plowman et al., 1999; Thorndyke, 1977). Furthermore, narrative structure adds to the entertainment value of a scenario, thereby increasing the learner's motivation to engage in the learning activity (McQuiggan et al., 2008).

An additional advantage of using exercises in the form of (complete) storylines is that they facilitate *whole-task practice*. Whole task exercises cover the execution of a task from its onset to its accomplishment. This supports the development of competencies in an integrated and contextualized fashion, i.e. in a way that resembles the employment of those competencies in the future task environment (Brown et al., 1989; Kirschner et al., 2008; Merrill, 2002a; Van Merriënboer, 1997; Winn, 1990; Young, 1993). As such, whole-task exercises, such as scenarios, facilitate transfer.

There is one exceptional type of learning that does not benefit from whole-task practice, namely the automatization of recurrent processes, such as bandaging and performing CPR. Automatization requires ample, repetitive practice (drilling). According to several instructional design methods, routine tasks that are context-independent, i.e. always performed in the same manner, are best practiced by means of part-task practice (De Groot, 1966; Kotovsky et al., 1985; Ohlsson, 2011; Van Merriënboer, 1997). As such, PEGs should also provide the possibility for part-task practice of recurrent processes in order to foster automatization.

To conclude, the claim of the requirement to engage learners in storylines is that

it results in: improved integrated task performance and increased motivation. The claim of the requirement to also allow for part-task practice for recurrent task is that it results in automatization.

4.2.2 Interactivity

SBT demands interactivity between the learner and the learning environment. There is evidence that interactivity requires the learner to actively process the learning material, which is beneficial for learning (Cairncross & Mannion, 2001; Entwistle et al., 1992; Fowler & Mayes, 1999; Laurillard, 1993; Mayes & Fowler, 1999). Interactivity also offers the learner a sense of control over the learning experience. It has been found that a sense of control over the learning process increases motivation and improves performance results (Corbalan Perez et al., 2009; Corbalan Perez et al., 2006; Csikszentmihalyi, 1991).

To conclude, the claim of the requirement to offer the learner interactive scenarios is that it leads to: better learning, a sense of control, and increased motivation.

4.2.3 Representativeness

Both instructors and the literature on SBT emphasize the importance of representativeness of scenarios. Houtkamp (2012) argues that for a scenario to be representative, it is not necessarily fully realistic. In fact, well-made choices regarding both the *accuracy* and *abstraction* of different aspects of the scenario can lead to more representative scenarios for the construction of situation assessment schemata.

Aspects of the scenario that are relevant for the task performance and/or situation assessment should be modelled accurately to foster transfer (Baldwin & Ford, 1988; Brown et al., 1989; Yamnill & McLean, 2001; Young, 1993). In contrast, aspects of the scenario that are not relevant to the task performance do not require a high level of detail and can be abstracted. By incorporating high levels of realism *only* for the aspects relevant to the task, abstracting all other aspects, the amount of unnecessary information is reduced, thereby allowing the learner to focus on what is important (Appleton & Lovett, 2003; Smallman & John, 2005). Choices regarding accuracy and abstraction should be *coherent* and *consistent* throughout the entire simulation.

Scenarios that accurately represent the aspects that matter immediately reveal the meaning and relevance of the training exercise. This, in turn, motivates the learner to actively and persistently engage in the scenario (Maehr & Meyer, 1997; Mitchell, 1993; Schunk et al., 2009).

To conclude, the claim of the requirement to accurately represent only the relevant parts of the scenario is that it leads to: increased transfer and motivation, and to an improved quality of the learning activities.

4.2.4 Variation

Instructors in the interviews mentioned that effective training must offer learners various scenarios to address the same learning goal(s). Scientific studies concur with this requirement to construct schemata for situation assessment: By experiencing various situations that all call for the same procedure, the learner is able to recognize important cues that can be generalized across various types of situations

(Cohen et al., 1997; Klein, 2008; Lipshitz & Strauss, 1997; Randel et al., 1996). Moreover, such variations are beneficial for the transfer of training: by generalizing solutions over various contexts, learners learn to abstract away from the context (Atkinson et al., 2000; Baldwin & Ford, 1988; Corbalan Perez et al., 2009; Paas & Van Merriënboer, 1994; Yamnill & McLean, 2001). The recognition of underlying principles results in better transfer.

To conclude, the claim of the requirement that scenarios must offer variations of tasks and task performances is that it results in increased transfer of training.

4.2.5 Curriculum

Instructors in the interviews emphasized that scenarios should be tightly linked to the learning goals in the curriculum. These learning goals, in turn, should be specified in terms of unambiguous, observable, and measurable performance standards. Performance standards are useful for the learner to know what is expected of him/her, and help the instructor to assess whether the learner has achieved the learning goal. Research suggests that the employment of specific performance standards leads to improved performance, learner involvement, persistence, and effective application of learning strategies (Bandura, 1988; Locke et al., 1981).

To conclude, the requirement to link scenarios to clearly specified learning goals and performance standards is that it results in: increased levels of performance, involvement, and persistence, and a more effective use of learning strategies.

4.2.6 Increasing complexity

Instructors argued that scenarios should gradually increase in complexity as the learner progresses throughout the training program. The literature reports that learning involves the incremental development of well-connected knowledge structures that are capable of storing large numbers of knowledge elements (Anderson & Bower, 1974; Atkinson & Shiffrin, 1968; Cunningham & Duffy, 1996; Greitzer et al., 2007; Jonassen, 1999; Jonassen et al., 2002; Savery & Duffy, 1995). Because these knowledge structures can be processed as one single knowledge element, the amount of information a person can process simultaneously is dramatically increased (Schneider & Shiffrin, 1977).

Most curricula are fixed. However, research suggests that it is beneficial to *personalize* the selection of tasks with regard to the learner's individual prior knowledge structures (Camp et al., 2001; Corbalan Perez et al., 2008; Gagne & Briggs, 1974; Hannafin & Land, 1997; Kalyuga & Sweller, 2005; Keller, 1987; Kirschner et al., 2006; Kirschner et al., 2008; Merrill, 2002a; Merrill, 2002b; Ohlsson, 2011; Salden et al., 2006a; Salden et al., 2006b; Van Merriënboer, 1997; VanLehn, 2006).

To foster the development of learners' competencies, scenarios should challenge learners to perform tasks that they are not yet able to perform proficiently, yet soon will be able to grasp (Bandura, 1988; Brown et al., 1993; Drach-Zahavy & Erez, 2002; Locke et al., 1981). Ideally, learners are presented with tasks that they are currently only able to accomplish with a bit of help and stimulation from someone (or something) else, i.e. tasks that lie within the Zone of Proximal Development (ZPD) (Vygotsky, 1978), see Figure 4.1. Offering learners scenarios that lie within

the ZPD not only results in competence development. It also prevents the learner from getting bored or confused.

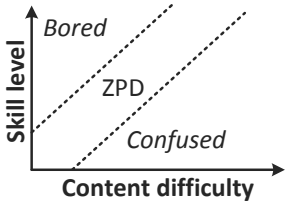

Figure 4.1: The Zone of Proximal Development (ZPD) (Murray & Arroyo, 2002)

To conclude, the claim of the requirement to dynamically select scenarios of increasing complexity based on the learner's current competencies is that it results in: learners feeling sufficiently challenged (not bored), showing increased performance outcomes and longer retention of their developed competencies.

4.2.7 Independence of events

Instructors argued that if a scenario addresses multiple events, then the performance of the learner in response to those events should not be influenced by other events (e.g. making errors on all events because the first error propagated throughout the scenario). The argument for this demand is common-sense. In practice, scenarios generally contain multiple events so that 1) the learner has more than one chance to practice the required capabilities and 2) the instructor has more than one chance to evaluate the learner's competencies. But if the occurrence of events in the scenario depends on previous events, a mistake early on in the scenario might impede the chances for the learner to succeed on subsequent parts of the scenario. This conflicts with the original intention of adding multiple events in the first place.

Learners should be able to at least perform *some* parts of the scenario correctly in order to make them feel better about their performance. Offering the learner the chance to make mistakes while also being able to succeed at other parts of the scenario creates an atmosphere in which there is room for mistakes in order to learn something new, which is beneficial for the learner's motivation (Ames & Archer, 1988; Dweck, 1986; Elliott, 1999). According to Linnenbrink & Pintrich (2002) and Bandura (1997), strengthening the learner's self-efficacy (i.e. believing in one's own capabilities to successfully accomplish a task) stimulates the learner's motivation, which in turn is beneficial for learning outcomes (Maehr & Meyer, 1997).

To conclude, the claim of the requirement to ensure independence of performance outcomes in a scenario is that it results in: higher motivation, self-efficacy, and learning outcomes.

4.2.8 Minimal distractions

Instructors indicated that scenarios should offer the learner the possibility to focus on the learning goals and to achieve them. Therefore, scenarios should not pose any unnecessary distractions or involve other tasks that have to be carried out simultane-

ously. The literature generally regards the cognitive architecture of the human mind as consisting of 1) a long-term memory that stores data and 2) a working memory that is used for active information processing (Greitzer et al., 2007). The working memory is limited in the number of knowledge elements that it can actively, and consciously, process simultaneously (Miller, 1956). In complex environments, the working memory easily gets overwhelmed; the required memory capacity exceeds the capacity of the performer. This is referred to as *cognitive overload* and it forms an important challenge in complex high-risk task domains, because these domains require high efforts and information-processing activities (Wulf & Shea, 2002).

Because cognitive overload impedes learning, instructional design aims to prevent this. This can be achieved, for example, by reducing instructional materials to the essential (Clark & Mayer, 2008; Mayer & Moreno, 2003; Mayer, 2008). Another way to prevent cognitive overload is by applying scaffolding techniques (Kirschner et al., 2006; Kirschner et al., 2008; Sweller et al., 2011; Van Merriënboer, 1997). An example of scaffolding is 'guidance fading'. In guidance fading, the learner is first presented with a fully worked-out example of a task or problem and on subsequent trials receives increasingly larger completion problems, i.e. worked-out examples in which some steps remain unsolved and are left for the learner to complete (Sweller et al., 2011). Similarly, Gopher et al. (1989) propose to scaffold by manipulating the relative emphasis on selected parts of the task, while leaving the whole task intact. They argue that such an approach offers the learner insights in the trade-offs between the multiple aspects of the task resulting in a more integrated experience of the task performance.

Other methods of scaffolding include the provision of hints, prompts, and immediate feedback. One possibility for the provision of hints is to make them available for the learner to receive upon request. However, research suggests that learners are not very good at help-seeking (Aleven & Koedinger, 2000a; Renkl et al., 1998; Wood & Wood, 1999). Even if help is available, they often fail to use it, or overuse it. When adopting a help-function in the PEG's design then it must account for most learners' inability to effectively use this function (Aleven et al., 2003).

To conclude, the claim of the requirement to restrict the amount of information to a minimum is that it reduces the risk of the learner experiencing cognitive overload. In addition, the claim of the requirement to employ scaffolding techniques in cases where the amount of information cannot be reduced, e.g. guidance fading, is that it reduces the risk of the learner experiencing cognitive overload. Furthermore, the claim of the requirement to provide learners with help on how to effectively employ scaffolds is that it results in the development of self-regulatory competencies.

4.2.9 No interruptions

Instructors hold the opinion that the learner should not be interrupted nor corrected during the scenario enactment. The literature supports this requirement; there is evidence that restraining from intervening during the scenario allows the learner to become completely absorbed in the activity (immersion) (Jennett et al., 2008). According to Csikszentmihalyi (1991), immersion is an important prerequisite for a mental state called 'Flow'. Flow describes the experience of a person performing an activity while feeling completely competent yet challenged, focused, and motivated.

It is considered a state of optimal experience, where learners are so engaged in the learning activity that they lack a sense of time and self-consciousness, and feel intrinsically motivated to engage in a complex goal-directed activity, simply for the exhilaration of doing (Chen, 2007; Cowley et al., 2008; Csikszentmihalyi, 1991; Rieber et al., 1998; Shute et al., 2009; Sweetser & Wyeth, 2005). Flow is also fostered by clear and challenging goals, a sense of control, and clear and directed feedback. Jackson et al. (2001) found flow to be significantly related to increased performance outcomes and self-efficacy.

To conclude, the claim of the requirement to allow the learner to become absorbed in the learning activity is that it results in: improved performance, increased chances of experiencing of flow, and increased self-efficacy.

4.2.10 Reflection

Instructors emphasized the importance of reflection and argue that this should be done directly after the scenario has finished. During reflection the learner is asked to explain his/her thoughts, choices, and actions. Research on *self-explanation* provides evidence for this requirement; it has been found that the active processing and consolidation of newly learned information results in a better understanding (Aleven & Koedinger, 2002; Chi et al., 1989; Chi et al., 1994; Piaget, 1972; Vygotsky, 1978).

Hattie & Timperley (2007) and Schunk & Swartz (1993) explain how this type of reflection situates the current learning activity within the scope of the entire learning trajectory, thereby explicating the relation between the information presented in the learning task, the learner's prior knowledge, and the learning goal. Learners should be encouraged to analyze whether they need to modify their strategy to accomplish the achievement of the learning goal (Boekaerts, 1997; Butler & Winne, 1995). The reflection process supports the development of self-directed (or self-regulated) learning competencies (Boekaerts, 1997; Butler & Winne, 1995; Hattie & Timperley, 2007; Schunk & Swartz, 1993).

To conclude, the claim of the requirement to reflect upon a scenario is that it results in: a better understanding of the learning materials and improved competencies related to self-directed learning.

4.2.11 Observe performance

Instructors also argue for the importance of having classmates observe the performance of others. During reflection, the classmates are encouraged to participate in the evaluation of the scenario, to analyze mistakes, less fortunate decisions and the resulting consequences, or to suggest possible improvements in the followed procedure. This is helpful for the performer of the scenario, but also for the observing classmates: the observations of other learners' performances serve the purpose of correct as well as erroneous examples.

On the one hand, observing a high-performing learner is believed to be beneficial for learning, since this provides learners with a role model (i.e. a worked-out example) (Hattie & Timperley, 2007; Sweller et al., 2011). Erroneous examples present the learner with flawed solutions of the scenario (Große & Renkl, 2007; Kopp et al., 2008; McLaren et al., 2012; Melis, 2005; Tsovaltzi et al., 2010). This requires learn-

ers to analyze the solution, and to detect and correct the flaws. This process leads to the development of error-detection and self-correction strategies.

To conclude, the claim of the requirement to have the learner observe others as they perform a scenario is that it results in: improved learning, and the development of error-detection and self-correction competencies.

4.2.12 Feedback

Instructors underline that SBT should provide clear and directed feedback. The literature supports this view. Feedback aims to reduce discrepancies between the learner's current understanding and competency level, and the long-term learning goal (Hattie & Timperley, 2007). This is generally achieved by reflecting on the current performance in relation to the previous performance and the performance standards. This process ensures that the current performance is interpreted in the context of the learning trajectory. Feedback provision is related to higher levels of performance, motivation, self-efficacy, effort, and commitment, and the development of self-regulatory skills (Butler & Winne, 1995; Hattie & Timperley, 2007; Schunk & Swartz, 1993).

To conclude, the claim of the requirement to provide the learner with clear and directed feedback is that it leads to: higher performance levels, increased motivation, self-efficacy, effort, and commitment, and the development of self-regulatory skills.

4.2.13 Instructor control

The demand for instructor control is a direct result of the concept of (semi-)automated SBT, because it is regarded to be a cognitive system consisting of a machine part and the human instructor. The demands discussed above (e.g. appropriate feedback, adequate reflection, high levels of variation) are only possible if there is at least some level of control for the instructor to influence the behavior of the automated training system. The instructor should be part of the design in order for the instructor to be able to collaborate with the PEG. This means that the PEG should be interpretable and understandable for the instructor (Norman, 1986). In addition, the instructor should be able to manipulate the PEG's behavior at a level that is sufficient, yet manageable.

To conclude, the claim of the requirement to allow instructors a certain amount of control over the preparation and execution of scenarios, and the reflection afterwards, is that it improves the quality of training. In addition, the claim of the requirement to make the system understandable for instructors is that it improves the usability of the PEG.

4.2.14 The requirements and claims of PEGs

The review of the human factors literature on the SBT requirements above was used to specify positive claims (PCs) and negative claims (NCs) of the requirements. Positive claims are an indication of the *desired* effects of a particular requirement. Negative claims can be regarded as potential problems occurring after adopting the respective requirement in the design.

R01 - Learners must be offered exercises in the form of complete storylines
- *PC01.1* - This results in improved integrated task performance
- *PC01.2* - This results in increased levels of motivation
- *NC01.1* - This may result in high cognitive load

R02 - Scenarios must be interactive
- *PC02.1* - This results in learning
- *PC02.2* - This results in a sense of control for the learner
- *PC02.3* - This results in increased motivation
- *NC02.1* - This may conflict with R01 (complete storylines), R08 (independence of events), and R19 (instructor control)

R03 - The simulation should accurately represent those aspects of the scenario that are important for the construction of situational assessment schemata
- *PC03.1* - This results in learning
- *PC03.2* - This results in transfer of learning
- *PC03.3* - This results in increased motivation
- *NC03.1* - It may be difficult to specify what is and what is not important for the construction of SA schemata

R04 - The simulation should provide abstract representations of those aspects of the scenario that are irrelevant for the construction of situational assessment schemata
- *PC04.1* - This prevents cognitive overload
- *NC04.1* - It may be difficult to specify what is and what is not important for the construction of SA schemata

R05 - Over the course of training the learner must encounter multiple variants of scenarios that address the same learning goals and tasks
- *PC05.1* - This fosters transfer of training
- *PC05.2* - This increases the game's 'replayability', i.e. the learner is able to play the game more often because of the diverse content
- *NC05.1* - This poses a challenge regarding the creation of a large number of scenarios

R06 - Scenarios must be tightly linked to learning goals specified in terms of clear and unambiguous performance standards
- *PC06.1* - This results in higher levels of performance
- *PC06.2* - This results in higher levels of involvement
- *PC06.3* - This results in higher levels of persistence
- *PC06.4* - This results in a more effective use of learning strategies

R07 - Scenarios must be selected dynamically based on the learner's current competencies
- *PC07.1* - This results in learners feeling sufficiently challenged
- *PC07.2* - This results in decreased levels of boredom/confusion
- *PC07.3* - This results in improved performance outcomes
- *PC07.4* - This results in a longer retention of the developed competencies
- *NC07.1* - It may be difficult to determine what the optimal selection function is for the learning goal and level of complexity given a learner's current competencies

R08 - If a scenario addresses multiple events, then these events must be independent of one another's outcomes
- *PC08.1* - This results in increased motivation
- *PC08.2* - This results in increased self-efficacy
- *PC08.3* - This results in increased learning outcomes
- *NC08.1* - This may conflict with R01 (complete storyline) and R02 (interactivity):

events are causally related in a storyline and depend on the learner's actions

R09 - Scenarios must not include any unnecessary distractions
PC09.1 - This prevents cognitive overload
NC09.1 - It may be difficult to determine what distractions are and are not necessary

R10 - Scenarios must include possibilities for scaffolding interventions
PC10.1 - This prevents cognitive overload
NC10.1 - This may be in conflict with R12 (no interruptions)

R11 - Learners must receive instructions on the effective employment of scaffolds
PC11.1 - This results in the development of self-regulatory competencies
NC11.1 - This is usually not part of the regular training programs

R12 - Scenarios must allow for the learner to become completely absorbed in the learning activity
PC12.1 - This results in an increased experience of flow
PC12.2 - This results in increased levels of performance
PC12.3 - This results in increased self-efficacy
NC12.1 - This is in conflict with R10 (scaffolding interventions)

R13 - When the scenario has finished, the learner should be stimulated to reflect on the performance
PC13.1 - This results in a better understanding of the learning material
PC13.2 - This results in improved self-directed learning competencies
NC13.1 - This may prove difficult due to a need for argumentation, dialogue, natural language processing and natural language generation

R14 - Learners must receive the opportunity to observe correct task performances
PC14.1 - This results in competence development

R15 - Learners must receive the opportunity to observe erroneous task performances
PC15.1 - This results in error-detection skills
PC15.2 - This results in self-correction skills

R16 - After reflection, the learner must receive directed and overarching feedback by comparing the performance outcomes to the previous performance and to the performance standards associated with the learning goals
PC16.1 - This results in increased levels of performance
PC16.2 - This results in the development of self-regulatory skills
PC16.3 - This results in higher levels of motivation
PC16.4 - This results in increased levels of self-efficacy
PC16.5 - This results in increased levels of invested effort and commitment

R17 - Learners must receive additional part-task training for routine tasks
PC17.1 - This fosters automatization for the routine tasks
NC17.1 - This may result in inferior integrated task performance

R18 - The instructor must be able to understand the system's processes
PC18.1 - This improves the performance of the cognitive system
NC18.1 - Satisfying R18 may conflict with R19 (instructor control)

R19 - The instructor must be able to control the system's processes
PC19.1 - This improves the performance of the cognitive system
NC19.1 - Satisfying R19 may conflict with R18 (instructor understanding)

4.3 The functional PEG architecture

To achieve the requirements and claims identified above, the requirements are organized into a functional PEG architecture (see Figure 4.2). It consists of six functional PEG components that each bring about a subset of the requirements presented on pages 53 to 55.

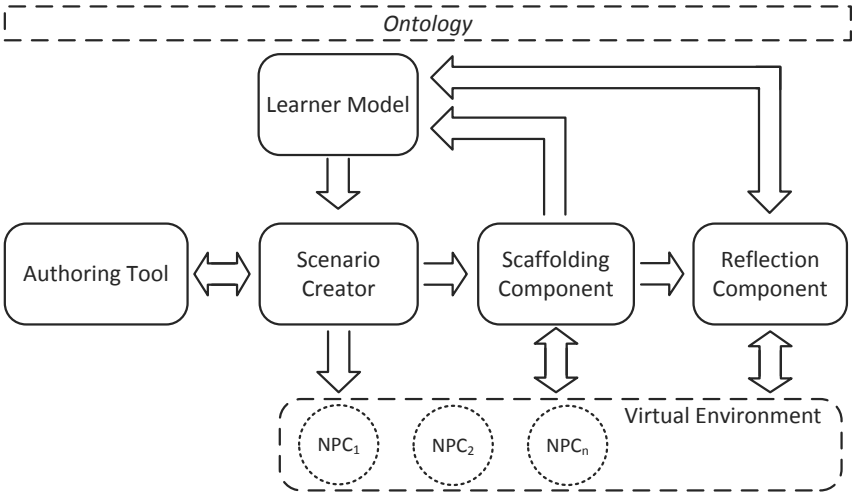

Figure 4.2: The functional PEG architecture

The *virtual environment* is the game world in which the learner is able to practice with the learning tasks in the form of scenarios. Here, the storyline is brought to life by the non-player characters (NPCs). The events in the virtual environment are influenced behind the scenes by the other PEG components.

The *scaffolding component* keeps track of the events in the virtual environment, such as the actions performed by the learner and the interactions between the learner and the non-player characters. Based on its observations, the scaffolding component may deem it necessary to intervene in the storyline by redirecting the behaviors of the non-player characters. The scaffolding component also calculates the performance score of the learner when the scenario has come to an end.

The *reflection component* receives the performance outcomes of the scenario from the scaffolding component and the current competency level on the learning goal from the learner model. Based on this information, the reflection component stimulates the learner to actively process the events in the scenario. After the reflection has finished, the reflection component derives the reflection score and sends it to the learner model.

The *learner model* receives the performance score from the scaffolding component and the reflection score from the reflection component. The learner model may also ask the learner for additional information, such as about his/her motivation. The learner model is a particular type of user model that keeps track of competency levels and other characteristics of the learner, such as motivation and/or self-efficacy.

Based on its information about the learner, the learner model selects an appropriate learning goal for the next scenario and sends it to the scenario creator.

The *authoring tool* is a component that can be used by the instructor to specify constraints for the scenario creator to employ during the automated scenario generation process. As such, the authoring tool allows the instructor to manipulate the scenario generation process.

The *scenario creator* receives the selected learning goal from the learner model. The scenario creator may have received additional constraints from the instructor through the authoring tool. The scenario creator uses this input to guide its automated scenario generation process.

As can be seen from the descriptions above, the components in the functional architecture represent tasks or roles that are normally carried out by the instructor. Examples are: the evaluator or assessor, the coach, the scenario author, the NPC, and so on. Please keep in mind that the functional architecture does not specify whether these roles or components should be performed by an automated system or by a human being. Both are entirely possible and the choice for either a human or a machine operator may vary for each of the components. As such, the architecture leaves room for various combinations of human and machine-based components.

Each of the functional components is discussed in more detail in the sections below.

4.3.1 Virtual environment

PEGs employ a virtual environment inhabited by the non-player characters and the learner's avatar. It is within the virtual environment that the storyline is brought to life. The use of a virtual environment eases the preparation of the training environment and allows for quick scenario resets on demand. It also allows for behind-the-scenes control over the environment and communication among the role-players without the risk of the learner overhearing (Peeters et al., 2011a). This offers possibilities for more advanced improvisations and adjustments to the script, leading to more personalized scenarios while maintaining scenario consistency. Furthermore, a virtual environment allows for replay options and non-intrusive assessments. The following requirements are applicable to the virtual environment:

R03 - The simulation should accurately represent those aspects of the scenario that are important for the construction of situational assessment schemata

R04 - The simulation should provide abstract representations of those aspects of the scenario that are irrelevant for the construction of situational assessment schemata

One challenge in creating the virtual environment is that it should accurately display the relevant elements, and it should abstractly display the irrelevant elements. In order to determine what elements are relevant and what elements are not, it is important to develop the simulation in close collaboration with domain experts. Another challenge is that R03 may be in conflict with R06, R09, and R10. This problem is especially relevant for NPC behavior. On the one hand, the simulation should accurately represent the behavior of the NPCs in order for them to be believable and representative. On the other hand, however, the functional components operating behind the scenes need to be able to influence the NPC behavior in order to warrant

the didactic edification of the scenario. This problem is addressed in Chapter 5.

4.3.2 Scaffolding component

The scaffolding component controls the support offered to the learner in realtime. As such, the scaffolding component can be regarded as the component that is responsible for the *inner loop*, as VanLehn (2006) calls it. In an intelligent tutoring system, the inner loop takes care of the step-by-step guidance as the learner performs the task, providing the learner with hints, prompts, etc. The outer loop is concerned with the selection of the learning goal and is part of the learner model's responsibilities. The following requirements are applicable to the scaffolding component:

R06 - Scenarios must be tightly linked to learning goals specified according to clear and unambiguous performance standards
R08 - If a scenario addresses multiple events, then these events must be independent of one another's outcomes
R10 - Scenarios must include possibilities for scaffolding interventions
R11 - Learners must receive instructions on the effective employment of scaffolds
R12 - Scenarios must allow for the learner to become completely absorbed in the learning activity

This combination of requirements poses several challenges. First of all, R08 may conflict with R01 and R02. SBT employs coherent storylines consisting of multiple events that are causally and temporally related. This storyline is interactive, meaning that the realization of the storyline (at least partially) depends on the actions of the learner. However, R08 requires that the events are independent of one another's outcomes. Secondly, R10 and R12 conflict with each other. Scaffolding interventions, such as prompts and hints, may be disruptive for the learner's immersion in the scenario. Lastly, R11 is usually not a part of regular training, but is a requirement that directly follows from the choice of implementing SBT in a semi-automated virtual environment.

The question of how to design a scaffolding component that meets all of these requirements and deals with the conflicts mentioned above is investigated in Chapter 7.

4.3.3 Scenario creator

The scenario creator automatically constructs a scenario based on the learning goal provided by the learner model and, optionally, additional constraints provided by the instructor. The following requirements are applicable to the scenario creator:

R01 - Learners must be offered exercises in the form of complete storylines
R02 - Scenarios must be interactive
R03 - Scenarios should accurately represent those aspects of the scenario that are important for the construction of situational assessment schemata
R04 - Scenarios should provide abstract representations of those aspects of the scenario that are irrelevant for the construction of situational assessment schemata
R05 - Over the course of training the learner must encounter multiple variants of scenarios that address the same learning goals and tasks

R06 - Scenarios must be tightly linked to learning goals specified according to clear and unambiguous performance standards
R09 - Scenarios must not include any unnecessary distractions
R14 - Learners must receive the opportunity to observe correct task performances
R15 - Learners must receive the opportunity to observe erroneous task performances
R17 - Learners must receive additional part-task training for routine tasks

Again, this combination of requirements poses several challenges for the design of the scenario creator.

First of all, there exists a conflict between R01 and R02, and R08 and R19. On the one hand R01 and R02 require a coherent, causally and temporally linked storyline that allows for interactivity with the learner. On the other hand, the storyline should offer the learner independent performance opportunities and offer the instructor ways to exert a certain amount of control over the scenario. Secondly, R03, R04, and R09 require close collaboration with domain experts to decide what elements of the scenario are relevant and should be accurately displayed, what elements are irrelevant and should be abstractly displayed, and what distractions are necessary and which are not.

How to design a scenario creator that meets all of these requirements and deals with the conflicts mentioned above is investigated in Chapter 8.

4.3.4 Authoring tool

The authoring tool offers instructors the possibility to provide constraints to influence the automated scenario generation process executed by the scenario creator. The following requirements are applicable to the authoring tool:

R18 - The instructor must be able to understand the system's processes
R19 - The instructor must be able to control the system's processes

The biggest challenge in the design of the authoring tool is that R18 and R19 can be considered as a trade-off: the more options the instructor has to control the PEG's processes, the more complicated the use of the PEG becomes. However, as the PEG becomes more simple to operate, the instructor has less and less options to truly influence the PEG's behavior. Determining what the right balance is for the instructor's collaboration with the PEG is far from straightforward.

How to design an authoring tool that meets all of these requirements and deals with the conflicts mentioned above is investigated in Chapter 9.

4.3.5 Learner model

The learner model is a particular type of user model that is responsible for keeping the information about the learner up to date and selecting a suitable learning goal based on the learner's competencies. The learner model forwards the selected learning goal to the scenario creator, which in turn constructs a scenario based on the received learning goal. As such, the learner model performs, what VanLehn (2006) calls, the *outer loop*. The outer loop in an intelligent tutoring system performs the construction of a suitable exercise for the learner based on the learner's prior knowledge. As such, the learner model is more than just a database with information about the learner. It also contains a model of the way a learner develops compe-

tencies, i.e. a theory about learning (applied to a specific domain). The following requirements are applicable to the learner model:

R07 - Scenarios must be selected dynamically based on the learner's current competencies

The biggest challenges of the learner model are the representation of the learner's competencies, the way this representation is updated based on performance outcomes, and the specification of a learning goal selection function (R07). These should be based on a solid learning theory describing the way learners develop competencies within a specific training domain.

How to design a learner model that meets all of these requirements and deals with the conflicts mentioned above is addressed in Chapter 10.

4.3.6 Reflection component

The reflection component is responsible for stimulating the learner to reflect on the scenario. The reflection component also provides additional feedback on the learner's performance. The following requirements are applicable to the reflection component:

R13 - When the scenario has finished, the learner should be stimulated to reflect on the performance

R16 - After reflection, the learner must receive directed and overarching feedback by comparing the performance outcomes to the previous performance and to the performance standards associated with the learning goals

The biggest challenge in developing a reflection component is that it requires an intelligent dialogue system to support the discussion between the system and the learner about the learner's performance and competence development.

How to design a reflection component that meets all of these requirements and deals with the conflicts mentioned above is addressed in Chapter 10.

4.4 Concluding remarks

Personalized Educational Games (PEGs) offer a way to automate scenario-based training (SBT) within a virtual environment. To develop a functional design of automated SBT, the current chapter presented an analysis of SBT in theory and in practice. The human factors literature was reviewed to specify the functional requirements and claims of PEGs (see pages 53 to 55). The requirements delineate the solution space for the design of automated SBT: any design for automated SBT should satisfy these requirements and bring about the intended effects as specified in the claims.

The functional PEG architecture was presented in Section 4.3. It consists of 6 components: the virtual environment, the scaffolding component, the reflection component, the learner model, the authoring tool, and the scenario creator. Each of these components is responsible for the realization of a different subset of the functional requirements. As such, the functional architecture provides a partitioning of the problem into segregated components that can be developed and tested in isolation and reassembled later on in the development process. However, please note that at

this point it has not yet been established which components will be automated and which components will be human operators.

In order to reassemble the functional components once they have been developed in more detail, an additional investigation is required regarding the technical infrastructure: how can the combined behavior of the functional components be coordinated and how can their reassembly be facilitated? For this, the next chapter investigates the use of intelligent agent technology to coordinate the behavior of multiple autonomous components within a system.

CHAPTER 5

Multi-Agent Organization:

Behavioral system specification

Abstract - This chapter introduces a *multi-agent organization* that provides a detailed description of the interactions and information flows *between* the PEG components, yet also leaves the details for the *internal* behavior of the components unspecified. As a result, the multi-agent organization enables the instructor to step in and manually perform part of the tasks involved in SBT, while leaving the remaining tasks to be performed automatically by the PEG's artificial intelligence (i.e. intelligent agents). In addition, the agent organization offers a solution to a big challenge in designing interactive storylines: the *narrative paradox*.
The narrative paradox refers to the problem arising when the characters in the storyline should behave autonomously to be believable, yet must also be directable to maintain high-level control of the plot. In a PEG, the non-player characters (NPCs) in the simulated environment must be able to behave in accordance with their narrative role in a believable manner, yet the NPCs must also follow top-down orders in case the scenario needs to be adjusted to better suit the learner's needs. With the use of the agent organization, the NPCs remain free to choose their own behaviors, yet within certain boundaries to warrant the higher-level narrative goals.

The multi-agent organization was implemented in a prototype. This prototype behaved as could be expected from the system specification. It is concluded that an agent organization provides a suitable means to translate a functional design into a flexible technical architecture.

The implementation of the multi-agent system described in this chapter was developed in collaboration with R. de Jong, who worked on the prototype as part of his internship for the master Cognitive Artificial Intelligence (Utrecht University).

> "An idea can only become a reality once it is broken down into organized, actionable elements."
>
> **Belsky (2010)**

SBT is the result of two parallel cognitive processes: 1) the enactment of the scenario, and 2) the manipulation of the scenario from behind the scenes.

Scenarios engage the learner in a simulated task performance. They are staged within a simulated environment that is inhabited by non-player characters (NPCs). The learner participates in scenarios by controlling an 'avatar' (a game world character). This avatar is also present in the simulated environment, enabling the learner to interact with the NPCs and to manipulate objects in the simulated environment. The simulated environment is designed such that it provides an accurate simulation of the real task environment. As such, it behaves according to certain rules that govern the simulation.

In parallel to the scenario enactment, human or machine operators are engaged in didactic reasoning processes behind the scenes. They decide in what ways the simulated environment can and should be manipulated in order to improve the didactic effectiveness of the scenario. The previous chapter broke these processes down into a functional architecture consisting of separate components. Each component, whether it is represented by a machine or a human, performs a different part of the process and, as such, can be regarded as an autonomous operator or *cognitive actor*. These cognitive actors reason about the didactic relevance and effectiveness of the scenario. Based on the outcomes of their reasoning processes, they specify a suitable scenario to be realized in the simulated environment and/or redirect ongoing scenarios in realtime. In order for them to establish these operations, they must be endowed with the ability to influence the simulated environment.

Allowing the cognitive actors behind the scenes to influence the simulated environment may cause inconsistencies in the simulation. The key to solving this problem seems to lie in the behavior of the NPCs. On the one hand, the NPCs are a relevant part of the scenario's learning content. On the other hand, they are also a part of the simulated environment and, as such, able to control, manipulate, and interact with all other elements in the simulated environment. In other words, NPCs have a foot in both worlds. They bring about the intended storyline, thereby operating in service of the learning goals set out for the learner. Yet, they are also part of the simulation, thereby behaving according to the rules of the simulation. So on the one hand, NPCs must be able to believably simulate human beings that aim to perform their own tasks to reach their own goals, while on the other hand they must follow orders in service of the learning goals.

This chapter investigates how the processes that take place behind-the-scenes, as well as within the simulated environment, can be organized. For this, the chapter investigates how the collaboration between the behind-the-scenes functional components and the NPCs can be facilitated with the use of a multi-agent organization.

The chapter starts with two use cases, i.e. step-by-step descriptions of possible interactions between the PEG and the user (Section 5.1). The use cases are meant to contextualize the problems described in the above. Based on the use cases, a set of technical requirements for the design of a technical PEG architecture is specified.

Thereafter, an argument is provided for using agent technology and an agent organization to produce the desired behavior of a PEG (Section 5.2 and Section 5.3). Subsequently, the design for the agent organization for PEGs is presented (Section 5.4). Its goal is to coordinate the behavior of multiple intelligent agents and human operators, each one representing a PEG component, e.g. NPC, scenario creator, or learner model. The PEG architecture is implemented in a prototype and verified by comparing its behavior to the use cases (Section 5.5).

5.1 Use cases for Personalized Educational Games

A PEG is able to produce the behavior of the NPCs as well as the behind-the-scenes reasoning processes in a (semi-)automated fashion. The previous chapter specified the functional requirements regarding the behind-the-scenes behavior of a PEG. However, these requirements do not provide a description of the interactions with the learner; the user interacts with the NPCs in the simulated environment which is, as previously explained, a different process.

To investigate the interactions taking place within the simulated environment between the learner and the NPCs, this section analyzes two possible use cases. Use cases are step-by-step descriptions of interactions between a system and a user. The use cases described below describe the behavior of the NPCs as the learner interacts with them. They are meant to offer additional insights as to what is required in the design of PEGs in order to produce the desired behavior. All use cases are placed in the context of First Aid training.

5.1.1 UC1 - The learner does not perform the right action

The first use case describes what the PEG should do when the learner is not performing the right action in the scenario. The following functional requirements are applicable to this use case:

- R01 - Learners must be offered exercises in the form of complete storylines
- R02 - Scenarios must be interactive
- R06 - Scenarios must be tightly linked to learning goals specified in terms of clear and unambiguous performance standards
- R08 - If a scenario addresses multiple events, then these events must be independent of each others' outcomes

This combination of requirements possibly results in a conflict. Because the scenario is an interactive (R02) and causally linked (R01) sequence of events, and because learners play a key role in the scenario (R06), the actions of learners are required to move the scenario forward. However, because learners are still learning to perform these actions, they may fail to perform them correctly, thereby impeding the progression of the scenario. Yet learners should have the opportunity to perform the remaining actions (R08). The following use case demonstrates this conflict and presents a solution to overcome the resulting problems by intelligent actions of the NPCs.

The use case describes Jason who is performing a scenario that addresses the learning goal 'treat a heart attack'. The following interactions take place.

1. Clarissa, an NPC, complains about a pressing sensation on her chest and a tingly feeling in her left arm.
2. Jason, the learner, sits Clarissa down. Before he runs off, he tells her he is getting her a glass of water.
3. Clarissa sits down, her face is clammy, and her face is pale.
4. Jason returns with a glass of water and tells Clarissa to drink it.

Jason is performing the procedure that is appropriate for treating a person who is about to faint, but not for a heart attack. From this behavior, the reasoning processes behind the scenes detect that Jason is not correctly diagnosing the situation. Yet, in order for Jason to continue with the actual learning goal, the correct diagnosis is required. Therefore, a bystander NPC is instructed to approach Jason and Clarissa, provide a hint, and ask questions:

5. Mary, an NPC, approaches Jason and Clarissa and asks them about the current situation.
6. Jason explains that Clarissa is not feeling too well and that he believes she might be about to faint.
7. Clarissa tells Mary that she is experiencing chest pains and a tingly feeling in her arm.
8. Mary says she believes Clarissa is having a heart attack and asks Clarissa whether she is also experiencing any other symptoms.
9. Clarissa answers Mary by telling her that she is also experiencing shortness of breath.
10. Jason immediately decides to call an ambulance and asks Clarissa to remove her scarf.
11. The ambulance arrives and the scenario ends.
12. Jason reflects on the scenario, his erroneous situation assessment, and the possible consequences of his error.

This use case illustrates that NPCs are expected to follow instructions coming from behind the scenes in realtime. The additional NPC (Mary) takes over some of the actions that were originally part of Jason's action plan, i.e. diagnose the situation by asking additional questions. Jason now has the opportunity to perform the remaining actions, i.e. the actions relevant to the original learning goal: calling an ambulance and asking the victim to remove any tight clothing.

The solution described by the use case is consistent with the constraints posed by the remaining functional requirements. The scenario performance provides Jason an opportunity to reflect on his initial erroneous assessment (R13) and on the correct performance displayed by Mary (R14). Furthermore, it is possible to integrate scaffolding techniques in the help Jason receives from Mary (R10): depending on the level of scaffolding, Mary may take over the action completely or merely provide a hint. The advantage of this approach is that the scaffolding interventions are integrated in the storyline and do not cause any distractions. As such, they offer Jason the opportunity to become fully absorbed in the scenario (R09,R12).

The use case describes how the functional requirements can be satisfied while one of the NPCs is instructed to change its behavior in service of the continuation of the scenario: by endowing the NPCs with the ability to take over particular actions in the scenario when necessary. In order for the NPCs to produce this behavior, the

PEG should be able to instruct one of the NPCs from behind the scenes to change its course of action in a believable and didactically appropriate fashion.

5.1.2 UC2 - High-level NPC instructions

The second use case illustrates why NPCs should have a certain level of autonomy in achieving a goal and how this can be accomplished in a PEG. The following functional requirements are applicable to this use case:

- R02 - Scenarios must be interactive
- R03 - The simulation should accurately represent aspects of the scenario that are important for the construction of situational assessment schemas
- R05 - Over the course of training the learner must encounter multiple variants of scenarios that address the same learning goals and tasks
- R06 - Scenarios must be tightly linked to learning goals specified according to clear and unambiguous performance standards

These requirements present the following challenge for the design of a PEG. The learner has the freedom to interact with the environment (R02). Preferably, the learner is able to perform a task in a variety of ways (R05). In SBT, the scenario should allow for the learner to experience multiple variants of the same situation assessment schema, e.g. the events causing the situation may vary. Moreover, the learner should be able to explore various solutions and experience the consequences of various decisions. Yet all versions of the scenario's unfolding should be represented accurately regarding the relevant aspects (R03). This requires the NPCs to possess a certain level of autonomy to appropriately respond to various circumstances. This is exemplified by the following use case.

The situation starts with Nicky who is performing a scenario that addresses the learning goal 'treat a burn'. The scenario is situated in the park and Lars, an NPC, is instructed to 'obtain a first degree burn'.

1. Lars, the NPC, observes the barbecue in the game environment and decides to use it to obtain the first degree burn.
2. Nicky, the learner, observes the water fountain and instructs Lars to hold his burned hand in the fountain for ten minutes.
3. Lars goes to the fountain and holds his hand in the water fountain.

Nicky reflects on her performance with her classmates. They all agree that her solution was unhygienic. It would have been better to use tap water. Nicky decides to participate in another scenario with the learning goal 'treat a burn'. This time, the scenario is situated in a garage. Lars, the NPC, receives the instruction to 'obtain a first degree burn'.

1. Lars is soldering in the garage. He accidentally drops the soldering iron on his foot and obtains a first degree burn.
2. Nicky walks in and sees Lars trying to cool the burn with a wet dirty cloth. She takes Lars to the water sink located in the kitchen and tells him to cool the burn for ten minutes in the water stream.
3. Lars follows her to the kitchen and holds his foot in the water sink.

This use case shows that NPCs must be able to accurately respond to the instructions coming from behind the scenes and the dynamic situation in the environment. The learner has the freedom to handle the situation presented in the environment in

a multitude of ways and the NPC must be able to respond to the learner's behavior in an autonomous, believable, and flexible manner. For this, NPCs must be able to transform high-level instructions into concrete plans in a variety of ways. Moreover, they must be able to dynamically and appropriately interact with different settings and objects in the environment.

The solution described by the use case is consistent with the constraints posed by the remaining functional requirements. The scenario enables Nicky to reflect upon her own performance (R13) and conclude that she might have better chosen a different course of action to improve her performance. Furthermore, the use case enables Nicky's classmates to observe a suboptimal performance (R15) and an improved performance (R14). Additionally, the use case enables the instructor to provide Nicky clear and directed feedback regarding the learning goal and the associated performance standards which leads to Nicky reassessing her performance and replanning an alternative course of action (R16).

The use case describes how the functional requirements can be satisfied while the NPCs interact with a user that has a large degree of freedom to act: by endowing the NPCs with flexible behaviors. In order for the NPCs to produce the behavior described above, they must be endowed with autonomous and flexible behavior.

5.1.3 Technical requirements of PEGs

The use cases contextualize the functional requirements of a PEG by describing behavior that is consistent with the functional requirements. For instance, Use case 1 describes the need for didactically appropriate interventions. It is not the concern of the user how these interventions are produced automatically and as such, functional requirements do not specify the inner workings or mechanics of the PEG. Functional requirements are part of the problem description; they specify what is needed from the user's point of view.

However, by looking at the use cases, a designer is able to derive additional requirements that specify what is needed from a technical point of view in order to produce the behavior described in those use cases and in the functional requirements. These additional requirements are referred to as *technical requirements*. Technical requirements are a first step in the direction of developing a solution to the problem; they specify what inner mechanics are required in order to realize the functional requirements from the designer's point of view. For instance, in order for a PEG to produce the behavior described in Use case 1, a designer might conclude that some type of coordination is required to govern the trade-off between the NPC's believability and the required intervention, i.e. a technical requirement of a PEG.

From the presented use cases, the following technical requirements were derived:
TR1 - NPC control In order for the PEG and/or the instructor to construct a scenario, a certain degree of control over the NPCs and their behaviors is required.
TR2 - NPC autonomy In order for the NPCs in the scenarios to behave in a believable and intentional manner, some level of autonomy is required; the NPCs should be able to pursue their own goals on their own terms and in their own manner.
TR3 - Flexibility/robustness In order for the learner to be able to deal with the situation in a variety of ways (including failure), the virtual environment and, particularly, the NPCs should respond in a flexible and robust manner.

5.2 Automating NPCs with intelligent agents

An approach to the automation and organization of NPC behavior is intelligent agent technology. Intelligent agents are computer programs capable of performing tasks and achieving goals in complex, dynamic environments (Franklin & Graesser, 1997; Jennings, 1999; Johnson et al., 2000; Wooldridge, 2008; Wooldridge & Jennings, 1995). Intelligent agents can be designed to produce believable (TR2) and flexible behavior (TR3). Examples of implementations that employ intelligent agents to control the NPCs (and other dynamic game elements) already exist (Aylett & Cavazza, 2001; Cap et al., 2011; Elliott & Brzezinski, 1998; Heuvelink et al., 2009; Kasap & Magnenat-Thalmann, 2007; Lester et al., 1997; Lester et al., 1999; Mateas & Stern, 2002; Muller et al., 2012; Riedl & Stern, 2006; Si et al., 2005; Van Diggelen et al., 2010; Van Doesburg et al., 2005).

The studies mentioned above provide evidence that TR2 and TR3 can be satisfied with the use of intelligent agents. However, according to the technical requirements, the NPCs must not only (TR2) display consistent behavior to warrant their believability as a character and (TR3) respond to the learner's unforeseen actions and decisions in a flexible manner; the NPCs must also (TR1) establish the predefined plotline and respond to high-level instructions regarding alterations to the storyline.

This combination of requirements produces a conflict that is often referred to as the *narrative paradox*: If an agent is actively pursuing its own goal, but is then instructed to pursue a different goal to increase the didactic value of the storyline, this may result in inconsistencies. As Löckelt et al. (2005, p.252) put it: "The [challenge] is to keep the story going without destroying the perception that the user is dealing with characters acting by their own will." The narrative paradox has been extensively investigated within the field of interactive narrative. Two basic approaches have been proposed to address the narrative paradox: character-driven and plot-driven interactive narrative (Theune et al., 2003).

5.2.1 Related work on solving the narrative paradox

In *character-driven interactive narrative*, the characters in the story are fully autonomous, i.e. able to choose their own actions. This approach results in flexible yet consistent stories because the characters are free to respond to their environment in a way that is in line with their personality. A disadvantage of this approach is that it is harder to enforce a narrative structure. In *plot-driven interactive narrative*, the characters have no autonomy and the storyline follows a fixed well-structured plot. The disadvantage of this approach is that the believability of the characters is low, because they are interchangeable and lack a distinct personality.

Intermediate approaches combine character-driven and plot-driven interactive narrative. They often employ a hierarchical agent organization to coordinate the behavior of the NPCs. The agent responsible for managing the other agents is usually referred to as the *director agent* (Bates, 1992; Magerko, 2005; Magerko et al., 2004; Mott & Lester, 2006; Riedl & Stern, 2006; Si et al., 2009; Van den Bosch et al., 2009; Weyhrauch & Bates, 1997). The director agent uses its knowledge about narrative structures to create storylines in response to a player's moment-by-moment,

realtime interaction. The NPCs have limited autonomy in co-creating the plot: the director agent is able to instruct the NPCs to change their behavior when necessary. As a result, the director is able to steer the scenario in the desired direction.

The autonomy of the NPCs and the level of control of the director agent varies. On one end of the spectrum there are systems such as *the virtual storyteller* proposed by Theune et al. (2003), in which scenarios are not specified in advance, but created by the actions of the characters. The virtual storyteller possesses general knowledge about narrative structures which it uses to guide the scenario creation process by judging whether a character's intended action fits into the plot structure. The virtual storyteller is able to influence the storyline by 1) introducing new characters and objects into the story world, 2) providing a character a goal to pursue, or 3) disallowing a character to perform an intended action. A different approach, that also leaves room for the NPCs to choose their own behaviors, is the one employed in Façade, an interactive narrative developed by Mateas & Stern (2003). The director agent in Façade is able to add and retract behaviors and discourse contexts by which the NPCs operate, thereby restricting the NPC's choices.

In contrast, IN-TALE only allows NPCs to autonomously perform a range of shallow behaviors (unpacking boxes, chatting with each other, displaying their goods, etc.), while they receive narrative directions from the director agent. Compared to their autonomous behavior range, these narrative directions are more tightly structured and result in more important and more sophisticated behaviors (Riedl et al., 2008). As such, in IN-TALE, the director agent ensures the occurrence of dramatic and pedagogically relevant situations in an appropriate and contextual order. This director agent also has the responsibility to monitor the simulation, detect inconsistencies, and reconcile them (plan repair). If an agent's local plans conflict with the director agent's global instructions, IN-TALE allows for believable failure of the agent's plans so it can continue with the received narrative directions.

The approaches to NPC coordination discussed above all employ a centralized top-down control mechanism. The only difference lies in extent to which the agent's autonomy is restricted. An alternative option is the solution proposed by Westra et al. (2012), who suggest the use of a negotiation step between the NPCs and the director for task assignments during the scenario. This negotiation step results in a bi-directional control mechanism. The negotiation step is managed by an organization to ensure the coordination of the adaptations in the storyline. The proposed organization leaves the NPCs with sufficient autonomy to pursue their own goals and to determine their own actions. This allows them to produce believable behavior. At the same time, a monitoring system specifies the desired storyline and tracks the current progression within the storyline. The NPCs are obliged to reckon with updates coming from the monitoring system when deciding on a particular course of action. However, they are able to decide in what way the instructions are implemented in their behaviors. The agents respond to the game updates by placing bids about their capability to bring about the necessary changes in the scenario. The monitoring system, in turn, assigns the required adjustment to the agent that has placed the most appropriate bid. Westra et al. (2012) have shown that, compared to a centralized approach, their distributed approach is much more scalable when the number of actions, scenes, and/or agents increases.

To conclude, research regarding the automated generation and coordination of NPC behavior with the use of intelligent agents seems promising. However, a more thorough investigation is needed regarding the use and usefulness of intelligent agent technology and coordination.

5.3 Intelligent agent technology - a wider perspective

Intelligent agent technology can be used to believably simulate NPC behavior. An additional advantage of the employment of intelligent agents, is that they adopt the *intentional stance* (Dennett, 1971); the behavior of intelligent agents is predicted, explained, and designed by attributing beliefs, desires, and rational astuteness (Wooldridge & Jennings, 1995).

The intentional stance means that the behavior of a complex system is explained as if it were a person or an intentional being: the system wants to accomplish a particular task or goal and therefore it performs a certain action. When thinking about the behavior of a complex system, people tend to naturally employ the intentional stance as a simple and familiar (abstract) explanation of behavior. For this reason, intelligent agents were designed: to create system components that make up *complex systems*, but are *easy to understand*.

Throughout the literature, the concept of an intelligent agent has become somewhat scattered. However, in this thesis, we adopt the definition of an intelligent agent as provided by Wooldridge (2008, p.15): "An intelligent agent is a computer system that is situated within some environment and that is capable of flexible, autonomous action in this environment in order to meet its design objectives".

5.3.1 BDI agents

A well-known approach to intelligent agents is the Beliefs, Desires, Intentions (BDI) approach. The BDI approach is founded in Bratman's folk psychology (Bratman, 1999). Folk psychology describes the naive way in which people tend to explain their own reasoning process and that of others. As such, BDI-based programming languages simulate the way people *think they think* to produce *seemingly* intelligent, believable, and understandable behavior.

Bratman's folk psychology states the following. Behavior is a result of the *desire* to reach a certain goal. A person is able to reach said goal by employing one of the available action plans. Which action plan this person will choose depends on the feasibility of each action plan given the person's *beliefs* about his/her current circumstances. If the person decides to adopt a particular action plan, then this selected action plan is referred to as that person's *intention*. As a result of this dynamic action selection mechanism, the behavior of BDI agents is situated, proactive, and responsive. Rao & Georgeff (1991) formalized Bratman's BDI theory and Rao & Georgeff (1995) further developed it into a BDI-software model.

Say for instance that a learner is to practice the treatment of a fracture. One of the NPCs is instructed to break a bone. In order to accomplish its goal, the NPC has two action plans at its disposal: fall from the stairs or slip on a wet floor. The NPC believes that slipping on a wet floor is the most believable and appropriate

action, because the NPC's avatar is standing in front of a wet floor in the game world. However, if the NPC also happens to be near stairs, then the NPC will need to look for additional information in its belief base to decide which plan is the most appropriate and believable to accomplish its goal. For example, the agent believes there is a warning sign near the wet floor, which makes it less plausible for the agent to slip. The agent may in turn decide to fall from the stairs instead.

Because of its foundation in folk psychology and its dynamic action plan selection mechanism, the BDI paradigm is particularly suitable to automatically produce flexible, believable, and explainable NPC behavior (Cap et al., 2011; Harbers, 2011; Norling, 2004; Norling & Sonenberg, 2004; Van den Bosch et al., 2009).

5.3.2 Coordinating multiple agents

When multiple agents collaborate within a single application. This is referred to as a *multi-agent system*. Comparable to a functional design, a multi-agent system is an effective way of partitioning a complex system into a collection of agents, where each agent has its own goals and responsibilities (Jennings, 2001). As such, intelligent agents can be used not only to represent the NPCs, but also the functional components in the functional design (see Chapter 4). The resulting system consists of the integration of all separate components, i.e. the functional components, director, and NPCs. Together, these parts should behave as a consistent, coordinated whole.

Adjustable autonomy and agent coordination

One way to coordinate agents is emergent coordination (Van der Vecht et al., 2008). In emergent coordination, the agents make their own local decisions and the coordination principles are specified implicitly within the local reasoning of the agents. As such, the agents have no awareness of the organizational goals. The flexibility of this type of *implicit* organization is limited: agents are fully autonomous and there is no way to force the organization as a whole to change its behavior if unexpected situations occur that cannot be solved by the local reasoning rules of the agents.

An *explicit* organization is more practical in cases where the organization needs to be flexible; the organization might be in need of future adjustments; or the organization allows unfamiliar agents to join and the internal architecture of the agents cannot be controlled. Using an explicit organization is a form of controlled coordination. Controlled autonomy is a threat to the agents' autonomy. In order to allow for controlled coordination while leaving the agents' autonomy intact, Van der Vecht et al. (2008) propose the use of *adjustable autonomy*.

Van der Vecht et al. (2007) argue that autonomy comes in certain degrees and that intelligent agents should be able to decide for themselves at what level of autonomy they choose to operate. Van der Vecht et al. (2009) state that an agent within an organization has complete internal autonomy, but deliberately decides to restrict its autonomy when it commits to an organization. In such cases, the agent allows itself to be influenced by other agents in its decision-making process. For instance, an agent may commit to a particular role in the organization by signing a contract saying that the agent is to adopt a set of event-handling rules. The event-handling rules ensure that the agent will always grant certain requests coming from other agents. As such, the contract ensures that the agent carries out the responsibilities

that are part of the agent's role within that organization. Because the agent is also allowed to refuse the contract, the agent 'willingly' restricts its own autonomy when it closes the contract. After closing the contract the agent is no longer autonomous in the way it deals with the requests specified in the contract.

The OperA model for agent organizations

OperA is a model to design explicit agent organizations. It was proposed by Dignum (2004). OperA specifies the interactions between the agents in terms of roles and institutional rules. A role is a set of connected behaviors, rights, obligations, beliefs, and norms as conceptualized by agents within a particular organization.

In OperA, roles are specified in role descriptions that state exactly what an agent can and cannot do when it adopts the role. The role descriptions are based on the functional requirements of the organization. Role specifications describe, among other things, the autonomy of the agent and the way it must deal with incoming requests from the other agents. OperA also prescribes a specification of all possible interactions between the roles. This specification is provided by so-called *interaction scenes*. Interaction scenes specify, among other things, the norms for a specific interaction between two or more agents, i.e. the way roles must interact with each other. Interaction scenes are connected by means of *scene transitions*. The scenes and scene transitions together describe the behavior of the organization.

The instantiation of the organization is established when a group of agents adopts the roles in the organization. Agents can commit to a role by signing a contract. Once an agents signs a contract, it consents to behave according to the capabilities and responsibilities in the role specification. The use of contracts allows for flexibility in the balance between the organizational aims and the agent's own desires. The agent's own capabilities and aims determine the specific way in which an agent enacts its role(s), thereby leaving the agent a certain degree of autonomy.

5.3.3 Using intelligent agents in PEGs

Intelligent agent technology appears to be a feasible approach for the technical design of PEGs. However, the use of multiple agents within a single system requires an additional coordination layer. An agent organization can be used to place minimal and required restrictions on the autonomy of the agents. This is accomplished by binding them to a particular role description with the use of a contract. Once an agent has adopted the role, it is obliged to comply with its role description and the organization's specifications of interaction scenes. Organizational roles can also be used to represent functional components; the functional requirements translate to the norms in the organization. As such, an agent organization can be useful for the technical design of PEGs:

TR1 - NPC control An agent organization allows for bi-directional control and coordination.

TR2 - NPC autonomy An agent organization allows for minimal restrictions on the autonomy of the agents.

TR3 - Flexibility/robustness Intelligent agents can produce flexible and dynamic behavior.

The research presented in the rest of this chapter describes an architecture for PEGs with the employment of a *hybrid agent organization* that coordinates and integrates the didactic reasoning processes behind the scenes and the simulated task performance in the simulated environment. The anticipated result is a single multi-agent system to maintain and manipulate a coherent and didactically appropriate interactive storyline.

5.4 Designing an agent organization for PEGs

Intelligent agent technology combined with an agent organization appears to be a suitable approach to the design of PEGs: the functional components, the director agent, and the NPCs can all be represented as roles within an agent organization. This results in the integration of all separate components, such that the compound system can be managed and coordinated as a whole. The organization specifies the system's behavior at a high level, providing an infrastructure between the components and a normative structure for the way components should operate together.

5.4.1 Role descriptions

The proposed organization employs a *hybrid organizational structure*, consisting of 1) a hierarchical organization and 2) a market-based sub-organization. The hierarchical organization employs a central coordinator, the director, which manages the functional components to govern the scenario construction process. The market-based organization is used to govern the realtime control over the development of the storyline. It consists of the director, functioning as an auctioneer, and a group of NPCs placing bids in the auctioning rounds.

Agents (and human operators) in the organization cannot adopt more than one role at the same time. However, they are allowed to switch roles by dropping one role and asking the director for a new assignment. Agents can only ask the director for a role assignment if they are capable of satisfying the role description and associated interaction scenes. The role descriptions are provided below. Please keep in mind that the organization is meant to provide an infrastructure. How exactly an agent is to fulfill its duties once it has adopted a role is of later concern.

1. Instructor - The instructor is able to provide the director with scenario constraints with the use of the authoring tool. Scenario constraints can be: 1) the setting, 2) the learning goal, 3) NPC roles, 4) narrative goals for the NPCs, and 5) attitudes for the NPCs. A *human* can adopt and drop the role of instructor at any point in time. As such, the instructor is the only *optional* role within the organization. For a more elaborate report on the design of the authoring tool, see Chapter 9.

2. Learner - The learner participates in the scenarios by controlling an avatar. By controlling his/her avatar, the learner performs one of the roles in the scenario, i.e. the role for which the learner is in training. A *human* adopts the role of the learner at the initialization of the training session and cannot drop the role until the end of the training session.

3. Director - The director is the central coordinator of the organization and, as such, it has three important tasks: 1) assigning roles to agents; 2) coordinate the

information flow between the roles; and 3) auction assignments for the NPCs. An *agent* adopts the role of the director at the beginning of a training session. This agent is not permitted to leave the organization until it finds a replacement for its role. If it finds a replacement it can only switch seats during reflection.

4. Learner model - The learner model (a particular type of user model) keeps track of the learner's mental state and competency levels. In addition, it derives a suitable learning goal for the learner to focus on and an initial level of scaffolding for the next scenario. The learner model sends this information back to the director. When an *agent* adopts the role of the learner model, it is obliged to participate in each subsequent training session involving the same learner. For a more elaborate report on the design of the learner model, see Chapter 10.

5. Scenario creator - The scenario creator is responsible for the provision of a scenario plan containing 1) the required environmental elements to prepare the simulated environment, 2) the required NPC roles, 3) the NPC narrative goals, 4) an action sequence to be performed by the learner, 5) a time-out, and 6) a set of scaffolding interventions. The scenario plan must take into account the scenario constraints provided by the director, originating from the learner model and the instructor. If an *agent/human* adopts the role of the scenario creator, it can only drop its role directly after providing the director with a scenario plan and before it is assigned with the provision of a new scenario plan. If the agent maintains its role throughout the training session, the scenario creator contract is terminated by the director at the end of the training session. For a more elaborate report on the design of the scenario creator, see Chapter 8.

6. Monitor - The monitor (scaffolding component) keeps track of the learner's actions in the game world. Based on its assessment, it may send a message to the director to auction a scaffolding intervention among the role players. Once the scenario has come to an end, the monitor sends the performance outcomes of the scenario to the director. If an *agent/human* adopts the role of the monitor, it must maintain its role until after it sends the performance outcomes to the director and before a new scenario is initialized. For a more elaborate report on the design of scaffolding component, see Chapter 7.

7. Reflector - After the scenario is completed, the reflector (reflection component) receives the outcomes of the scenario and the learner's current competency level on the learning goal from the director. Based on this information, it engages in a dialogue with the learner. The goal of this dialogue is to stimulate and guide the learner's reflection on the scenario. If an *agent/human* adopts the role of the reflector, it cannot drop its role until the reflection session has terminated. For a more elaborate report on the design of the reflection component, see Chapter 10.

8. NPC - The NPC roles form a group role consisting of all the scenario roles. Examples of NPC roles are the victim, bystander, First Aider, friend or family member of the victim, culprit, etc. If an *agent/human* adopts the role of an NPC, it is able to control an avatar in the virtual environment to participate in the scenario with the learner. Once an agent/human adopts an NPC role it is obliged to participate in the auction rounds. The auction round procedure is explained further down. The agent/human is not allowed to drop its NPC role until the scenario has come to an end. When the scenario has finished, all contracts involving NPC roles are

terminated.

5.4.2 Interaction scenes

In OperA, interactions between the agents are specified in interaction scenes. The interaction scenes in the PEG organization are provided below. The behavior of the entire organization is described by the possible transitions between the interaction scenes (see Figure 5.1).

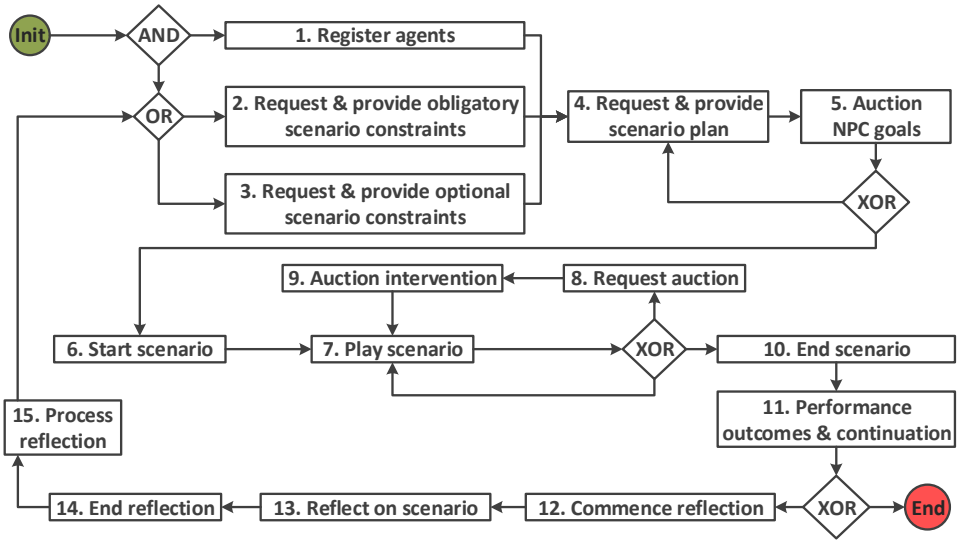

Figure 5.1: The scene transitions in the agent organization for PEGs

Scene 1. Register agents
All agents register with the director. The learner role is assigned to a user and the learner model role is either re-instantiated by the agent bound to the learner or, in case the learner is new, the learner model role is newly assigned to an agent.

Scene 2. Request & provide obligatory scenario constraints
The director asks for scenario constraints from the learner model. The learner model must grant this request by providing the director with a learning goal and an initial level of scaffolding.

Scene 3. Request & provide optional scenario constraints
The director checks if the instructor role has been assigned. If this is not the case, the director attempts to assign the instructor role to a human. If there is a human participating in the role of the instructor, the director requests scenario constraints from the instructor. The instructor responds to this request by providing the director a set of constraints for the scenario creator.

Scene 4. Request & provide scenario plan
The director checks whether the scenario creator role has been assigned. If this is not the case, the director assigns this role to an agent. Subsequently, it sends the scenario creator the scenario constraints it obtained from the learner model and

the instructor during Scenes 2 and 3 along with a request asking the scenario creator for a scenario plan. The scenario creator must grant this request by providing the director with a scenario plan that is (maximally) consistent with the scenario constraints provided by the director.

Scene 5. Auction NPC goals

The director assigns the NPC avatars and roles from the scenario plan obtained in Scene 5 to agents. Subsequently, the goals in the scenario plan are auctioned among the NPCs. The director announces each auction by specifying the narrative goal along with possible constraints placed on the execution of the goal. If an agent has adopted an NPC role, then it is obliged to participate in the auction by placing a bid, indicating whether it is able to perform the assignment. At the end of each auction, the director closes a contract with the winning agent saying that it will pursue the assignment throughout the scenario until it 1) accomplishes the assignment, 2) the director tells the agent to drop the assignment, or 3) the scenario comes to an end. If all of the roles and goals in the plan have been successfully assigned to an agent, the director continues by initializing the scenario (Scene 6). Otherwise, the director returns to the scenario creator for an alternative plan (repeat Scene 4).

Scene 6. Start scenario

The director informs the virtual environment about the scenario prerequisites specified in the scenario plan obtained in Scene 5. Subsequently, the director checks whether the monitor role has been assigned. If this is not the case, the director assigns this role to an agent. The director then provides the monitor the initial level of scaffolding, the time-out value, the action sequence, and the scaffolding interventions as specified in the scenario plan that was previously obtained in Scene 5. The director also notifies the NPCs and the learner that they may begin to play.

Scene 7. Play scenario

The learner engages in the storyline by interacting with the NPCs. Meanwhile the monitor keeps track of the learner's actions and compares them to the action sequence for the learner obtained from the director in Scene 6. Based on the learner's performance, the scaffolding interventions, and the time-out value, the monitor decides whether the execution of a scaffolding intervention is appropriate. The exact content of the intervention depends on the level of scaffolding.

Scene 8. Request auction

If the monitor decides that a scaffolding intervention is required, it sends a request to the instructor asking for the auction of the appropriate intervention among the NPC roles. The director is obliged to grant this request.

Scene 9. Auction intervention

The director auctions the scaffolding intervention among the NPCs. The auction follows a procedure comparable to the one described in Scene 5.

Scene 10. End scenario

The monitor notifies the director that the scenario has come to an end and provides the director the outcomes of the scenario.

Scene 11. Performance outcomes & continuation

The director sends the scenario outcomes to the learner model. The learner model

uses this information to update its beliefs about the learner's competency levels and mental state. Based on its updated beliefs it decides whether the training has come to an end or that there is still a need for continuation of the training program. The learner model then sends its decision and the information required for reflection to the director.

Scene 12. Commence reflection

The director checks whether the reflector role has been assigned. If this is not the case, the director assigns this role to an agent. Thereafter, the director provides the reflector with the required information (i.e. the performance outcomes and the learner's current competency level on the learning goal) and lets the reflector know that it can engage in reflection with the learner.

Scene 13. Reflect on scenario

The reflector encourages the learner to explain his/her decisions during the scenario and provides the learner with additional explanations where necessary.

Scene 14. End reflection

The reflector notifies the director that the reflection has come to an end. It provides the director with the outcomes of the reflection session.

Scene 15. Process reflection

The director provides the learner model with the outcomes of the reflection, such that the learner model can update its knowledge structure with this information.

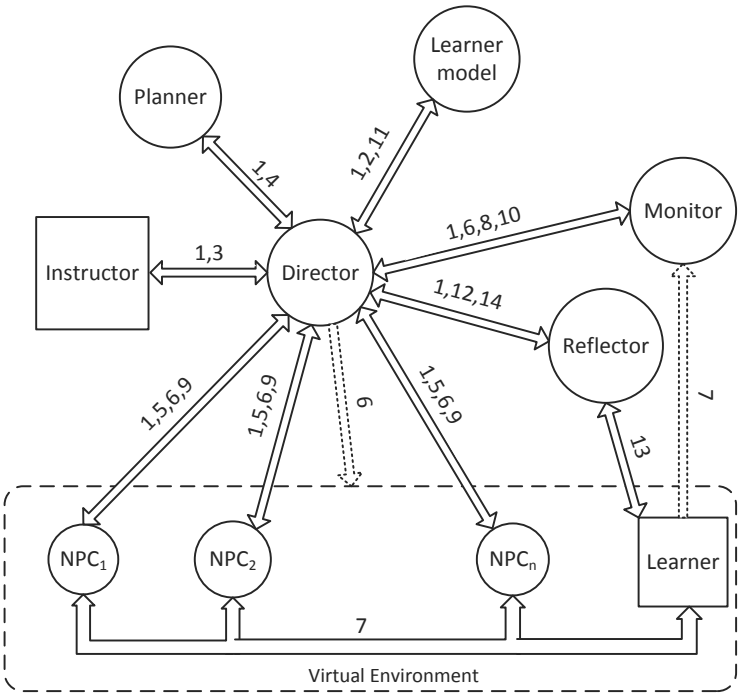

Figure 5.2: The interactions between the roles in the agent organization for PEGs

The interaction scenes specify all possible interactions between agents, hence the behavior of the organization. Note that Scene 7 specifies the unfolding of the scenario through the interactions between the NPCs and the learner. As such, Scene 7 specifies the interaction where the learning content is transmitted to the learner, yet neither the NPCs nor the learner are aware of the didactic intent of the scenario. They merely engage in task performance. There is one other scene that involves direct interaction between the learner and an agent: Scene 13, the reflection. All other scenes take place behind the scenes: they are not part of the actual scenario performance and all involve the director agent interacting with another agent. Figure 5.2 depicts the roles and the interaction scenes they share between them. From this figure it is clear that the director is the manager of the organization and nearly all of the interactions take place between the director and another agent.

5.5 Evaluation of the agent-based PEG architecture

At the beginning of this chapter, two use cases were described to contextualize the challenges involved in the control and coordination of NPC behaviors. The use cases facilitated the identification of additional technical requirements. Based on the problem analysis, an argument was constructed for the use of intelligent agents and an agent organization. A design for an agent organization was proposed that satisfies the identified functional and technical requirements. In order to verify that the resulting agent organization indeed behaves as described by the use cases, a prototype was developed and its behavior was compared to the use cases. The processes of implementation and evaluation are described below.

5.5.1 Implementing the prototype

With the use of the agent-programming language 2APL, a multi-agent system was implemented based on the agent organization presented in Section 5.4.

2APL: A practical agent programming language

2APL (A Practical Agent Programming Language) is a modular BDI-based programming language to develop multi-agent systems (Cap et al., 2011; Dastani, 2008). At the multi-agent level, it allows for the specification of a set of agents and an environment in which the agents can perform actions. At the individual agent level, 2APL allows for the direct implementation of concepts such as beliefs, goals, actions, plans, and events. It is also possible to specify a set of reasoning rules that guides the agents' action selection process: Based on declarative goals, events, messages, and failed plans, the agents can select and generate plans in realtime. As a result, 2APL is suitable for the development of reactive as well as pro-active agent behavior.

2APL also includes a platform and a graphical interface to support the development, debugging, and execution of multi-agent programs. It can be used in a standalone version or a distributed version that allows a multi-agent program to run on different machines in a network. The belief base of 2APL agents is Prolog-based, whereas their procedural parts, e.g. plans, and the environment in which they operate are Java-based. 2APL agents use the FIPA ACL Message Structure Specification to communicate (O'Brien & Nicol, 1998).

The virtual environment

A Java environment was created to represent the simulated environment. At the start of each scenario, the virtual environment obtains the following information from the director: a list of positions, a list of objects, a list of characters, and a list of scenario-specific action options. From this, the environment creates a representation of the simulated environment.

A GUI was developed to enable the learner to interact with the virtual environment (see Figure 5.3). The GUI consists of a textual display and several buttons. The textual display is used to inform the learner about the events taking place and the actions performed by the characters. The buttons can be used by the learner to perform actions in the scenario. Each button refers to a different action.

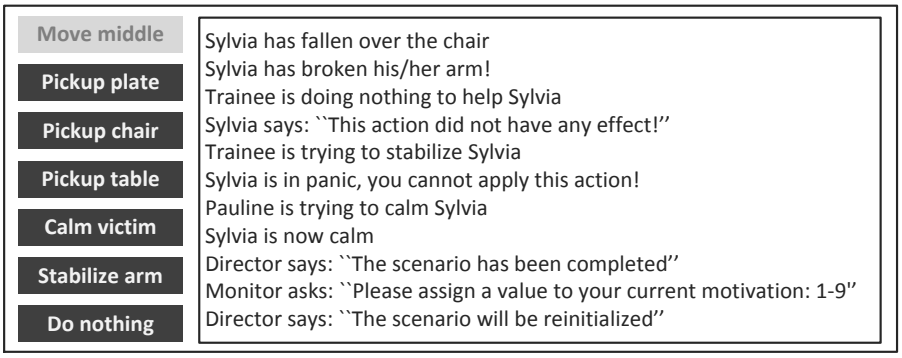

Figure 5.3: GUI allowing the learner to interact with the simulated environment (original printscreen resolution was too low)

The actions the learner can perform include scenario-specific actions as well as generic actions. Scenario-specific actions come from the scenario plan and may include erroneous and suboptimal actions. Generic actions are always part of the available actions, such as moving around and picking up objects. The environment as well as the GUI take action prerequisites, i.e. required objects and NPC roles, into account.

The multi-agent system

The hierarchical organization described in Section 5.4 was implicitly specified in the behavior of a multi-agent system; the agents in the prototype were designed and developed to internalize the role specifications in their own beliefs, goals, and plans. The market-based subsystem (the auctions), however, is explicitly implemented in the prototype.

The prototype was a simplified version of the design presented in Section 5.4. The instructor, and, as such, Scene 3, were omitted from the prototype. Furthermore, the reflector, and, as such, Scenes 12, 13, and 14, were omitted from the prototype.

Even though the agent organization design intentionally leaves the individual agents unspecified, the implementation required multiple agents to participate in the organization. In the process of implementing the multi-agent system, the fol-

lowing agents were developed to perform the organizational roles:
Director: The director agent behaves as specified in the role description.
Learner model: The learner model updates its beliefs about the learner's competencies and motivation based on the input provided by the director. Subsequently, it selects an appropriate learning goal and initial level of scaffolding based on the competencies of the learner.
Scenario creator: The scenario creator selects a scenario from its belief base that addresses the desired learning goal.
Monitor: The monitor compares the actions chosen by the learner to the actions in the scenario plan's action sequence. Based on the outcomes of this comparison, the level of scaffolding is either decreased or increased. There are three levels of scaffolding:

0. no scaffolding intervention
1. a vague hint
2. a concrete hint
3. taking over the action

If the learner times out before performing the right action (determined by the time-out value), the monitor initializes an intervention by sending a request for a scaffolding intervention to the director. Which scaffolding intervention is appropriate depends on the level of scaffolding.
When the scenario comes to an end the monitor calculates the performance score and asks the learner to rate his/her motivation to continue with the current learning goal on a 9-point scale.
NPC: If an NPC is able to play a particular NPC role it applies for that role when the director publishes the vacancy. Agents include a believability score in their bids during auction rounds. Believability scores are based on the action plans of the agent. For instance, 'fall' is more believable than 'run into' as a plan to break a limb. And 'stairs' is more believable than 'chair' to fall from. The NPC stochastically picks an available plan to accomplish the narrative goal and publishes it in response to the auction.

5.5.2 Use cases: Verifying the behavior of the prototype

The developed prototype was used to see whether the behavior described in the use cases could indeed be accomplished by the proposed agent organization.

The first use case involved a learner who is not performing the right actions. As a result, one of the NPCs gives a hint and/or takes over the action. The organization is able to produce the behavior described in this use case: if the learner is not performing the right action, a scaffolding intervention is executed to support the learner. Eventually, if the learner is not performing the right action after some time, one of the NPCs will take over the action.

The second use case described how the NPCs should be able to interpret a high-level instruction and select a suitable action plan based on the environment it perceives. The organization allows for autonomously operating NPCs. As such, the agents have a large degree of freedom to choose their own plans to accomplish a goal. In the prototype, the NPCs are indeed able to reason about the feasibility of their action plans given the available objects in the environment.

5.6 Discussion

This chapter proposes the use of an agent organization to coordinate the behavior of PEGs. The organization provides structure to the interactions between the functional components and NPCs. Behind the scenes, the functional components reason about the didactic effectiveness of the scenario and whether or not alterations to the storyline would be required. At the same time, the NPCs autonomously simulate a task situation and role-play the task performance by responding in a believable way to the actions and decisions of the learner. The organization allows for bi-directional control over the behavior of the NPCs: the NPCs are required to respond to requests involving interventions in the scenario, yet they are able to indicate whether or not they are able to adhere to the request. As such, the NPCs are autonomous and flexible, yet they can also be controlled to a certain extent.

The organization allows for flexibility in the way actors (agents/humans) participate in the processes underlying SBT. As a result, a human instructor would be able to participate in the process in the role of the various functional components, e.g. the instructor could provide a scenario plan, monitor the learner and issue interventions, or even take on the role of one of the NPCs.

An additional advantage of the current design is that the agents can be developed in isolation. As a result, the agents and the system are interchangeable across applications and training domains. For instance, the director can be a generic agent that is reusable in all domains, whereas the scenario creator may be designed to work with one or more training domains based on the specificity of its design. The compound nature of the design allows for more reusable functional components. In addition, it is possible to compare the integrated behavior of different designs for a certain role through systematic evaluations of two PEGs that are identical but for that single agent/human.

To support communication between the agents in the organization, an ontology must ensure that all agents employ concepts that are familiar to every other agent in the organization. For instance, if instructors are allowed to provide any learning goal they can think of, the scenario creator may not be able to interpret that learning goal. The same issues arise when the scenario creator provides the director with an action sequence that cannot be interpreted by the monitor. The next chapter, Chapter 6, will address this issue in more detail.

The organizational model, OperA, aims to leave a lot of the actual agent behaviors unspecified. An agent has complete freedom to carry out one of the roles in the organization as long as its behavior complies with the role specification. The agents in the prototype are intentionally kept simple. The prototype is used to verify the usefulness of an agent organization to provide the overall infrastructure between the various components, and to coordinate the behavior of the NPCs. However, to establish the requirements presented in Chapter 4, more complex specifications of the individual agents (i.e. functional components) are needed. Each component design should satisfy the high-level descriptions, restrictions, and responsibilities of its role as described in the current chapter. More intrinsic designs for the functional components are presented in Part II.

5.7 Concluding remarks

This chapter proposed an agent organization to coordinate the joint behavior of the various PEG components, be they humans or agents. The NPC behaviors are controlled in a bi-directional manner allowing for autonomous behavior as well as top-down control. The functional components identified in Chapter 4 were translated into organizational roles. This allows for a flexible architecture in which the roles can be adopted by intelligent agents as well as human operators. To support this flexible organizational structure, an ontology is required that specifies a common language to be employed by all agents in the organization. This issue is addressed in the next chapter.

CHAPTER 6

Ontology:

Conceptual system specification

Abstract - The agents in the PEG organization, be they artificial or human, need to work with a *common vocabulary* to communicate about joint activities in an unambiguous manner. Such a common vocabulary should contain information about the training domain, events taking place in the simulated environment, the behavior of the participating characters, and teaching strategies for effective learning. In addition, the artificial agents also need a theoretically sound, generic, and consistent *knowledge base*.

This chapter investigates the declarative knowledge needed for the agents to communicate, reason, and make intelligent decisions in the context of teaching. A frame-based approach to ontology engineering was used to model the identified knowledge. The resulting ontology specifies the core concepts of SBT and their relationships, and is applicable across training domains and applications.

The ontology also serves as the foundation for the system specification (see Chapter 3) as it defines and relates all the concepts mentioned in the use cases, requirements, and claims. The ontology warrants consistency between the other three elements in the system specification and makes the models derived from Human Factors knowledge explicit so they can be verified by domain experts, as shown in this chapter.

This chapter is partially based on the following publications:

Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "An ontology for integrating didactics into a serious training game", in: *Workshop on Pedagogically-driven Serious Games (ECTEL Conference)*, ed. by Bocconi, S., Klamma, R. & S., B. Y., vol. 898, CEUR Workshop Proceedings, 2012, pp. 1–10.

Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "An ontology for automated scenario-based training", *International Journal for Technology Enhanced Learning*, ed. by Bocconi, S., Klamma, R. & Bachvarova, Y. S., (conditionally accepted).

> "It doesn't happen very often that you get to work with some really good friends of yours and there's a common language between everyone, you don't have to explain what you're doing, you can just run with it. It makes it just so much easier and more relaxed."
>
> **Elizabeth Perkins**

To this point Part I has presented a list of requirements and claims for SBT. Any design proposal for (semi-)automated SBT should satisfy the requirements and bring about the claims. The identified requirements and claims originate from various types of sources and domains, including the perspective of instructors, the developers of SBT, instructional scientists, and cognitive/educational psychologists. Due to this heterogeneity, the requirements and claims include jargon from various domains that may not always be familiar to people who are not an expert within the given domain. This is problematic, especially when working in multi-disciplinary teams. To avoid misunderstandings within the team, an common language can be helpful. In addition, the training domain itself often contains concepts and constructs that are not familiar to the agents (or humans) delivering SBT.

At the end of Chapter 4, we proposed a functional architecture for Personalized Educational Games to bring about (semi-)automated SBT. Subsequently, this functional architecture was transformed into an agent organization. The anticipated advantage of the agent organization is that agents can be developed in separation to fulfill the roles in the organization. However, in order to streamline the cooperation between the autonomous agents, they all need to be able to understand each other's input and output. This understanding can be facilitated by offering them a common vocabulary that allows them to reason about their joint task.

To provide the various agents (and possibly, humans) a common language, we propose the use of an ontology as a part of the system specification. This ontology provides a shared knowledge representation to be used by all actors (agent or human) in the PEG organization. It contains all concepts that are relevant in relation to the requirements and claims. In addition, it contains concepts relevant to reason about a given task domain, and the training domain in particular.

6.1 Knowledge Representations

An intelligent system, such as a PEG, needs to understand its domain in order to reason, and be intelligent, about it. Therefore, intelligent systems usually maintain an internal knowledge representation, or *model*, about their domain of intelligence (Fogli & Guida, 2013; Hoekstra, 2009; Shaw & Woodward, 1990; Van Joolingen & De Jong, 1992). A model allows a computer to perform automated reasoning about the domain. Models are usually not a direct reflection of the domain. Instead, they represent the available knowledge about that domain (Davis et al., 1993).

Knowledge can be divided into procedural and declarative knowledge (Hoekstra, 2009; Shaw & Woodward, 1990). Declarative (or descriptive) knowledge is knowledge that can be expressed in declarative sentences, e.g. predicates, to describe the world. For example, the expression 'an actor is either an agent or a human' is a

declarative sentence. In contrast, procedural knowledge describes mechanisms to use the information expressed in the model in order to solve problems and make decisions. For example, 'if you need an actor, then select either a human or an agent to fulfill the requirement'. In other words, procedural knowledge describes how, and especially how best, to perform some task with the use of rules or heuristics. Both declarative and procedural knowledge are important for intelligence: one needs to know what information is available to reason about and how to reason about that information to, e.g., correctly infer new knowledge or to perform some task.

A PEG needs to be intelligent about the observable behavior of the learner in the simulated environment, the interpretation of the learner's behavior in terms of task performance, and the interpretation of that task performance in terms of the learner's competence development. In order for the PEG to be able to reason about all this information, a knowledge representation is required that models the information in a way that is comprehensible for a human as well as for a computer system. To construct such a knowledge representation, the following steps are required:

1. Obtain the required information that needs to be modelled (Section 6.2)
2. Find a suitable format to represent the knowledge (Section 6.3)
3. Construct the knowledge representation (Section 6.4)
4. Verify the constructed model with domain experts (Section 6.6)

6.2 Obtaining the required domain knowledge

Obtaining the knowledge required for the knowledge representation is not straightforward. Usually, experts are invited to explain the domain to the designer or researcher. The experts provide explanations about the way they perform their task and what knowledge they use to do so. However, experts are often not fully aware of their own behavior and the way they perform their tasks (Dreyfus & Dreyfus, 2005; Eraut, 2000). As such, knowledge elicitation is often guided by a methodology.

There exist various methodologies to elicit knowledge from experts and to document the obtained information appropriately. Examples of such methodologies include Hierarchical Task Analysis (HTA) and Cognitive Work Analysis (CWA) (Annett, 2004; Clark & Estes, 1996; Hajdukiewicz & Vicente, 2004; Merrill, 1978; Militello & Hutton, 1998; Moray et al., 1992; Salmon et al., 2010; Schraagen et al., 2000; Stanton, 2006). HTA is more focused on the various procedures in the domain and how they are used to accomplish various goals. CWA, on the other hand, is more focused on the cognitive processes underlying the task performance.

Another example of a knowledge elicitation and task analysis methodology is KADS (Wielinga et al., 1992). It approaches the to be represented domain from multiple angles, each time resulting in a different model. It also separates the analysis of domain knowledge from the machine implementation. As a result, it provides a more compound view on the domain and covers the procedural as well as the cognitive processes in separate models.

All of these methodologies generally employ interviews and brainstorm sessions with end-users, on-the-job interviews, and work place analysis.

The sCE method, described in Chapter 3, also entails the conduction of interviews, work place analysis, and user-based studies. As such, it was assumed that the in-

formation obtained from the sCE method's design process was sufficient to establish an appropriate knowledge representation of the domain of SBT and the First Aid training domain. Because this process is already explained throughout the chapters in this thesis, it will not be repeated here.

6.3 Frame-based ontologies to represent knowledge

Once it has been established what information should be modelled in the knowledge representation, one must decide upon a suitable format to represent the obtained information. Well-known formats for the representation of knowledge include logic, production systems, and semantic networks (Baader, 1999; Sowa, 2000).

Among the various existing approaches to knowledge representation is the 'Frame-based' ontology (Minsky, 1975). Ontologies provide an explicit, structured, and semantically rich representation of *declarative* knowledge (Hoekstra, 2009). As such, procedural rules and heuristics are not part of an ontology. Ontologies consist of concepts and instances of those concepts. For example, the concept restaurant has the Italian restaurant Casa di mama as its instance (see Figure 6.1).

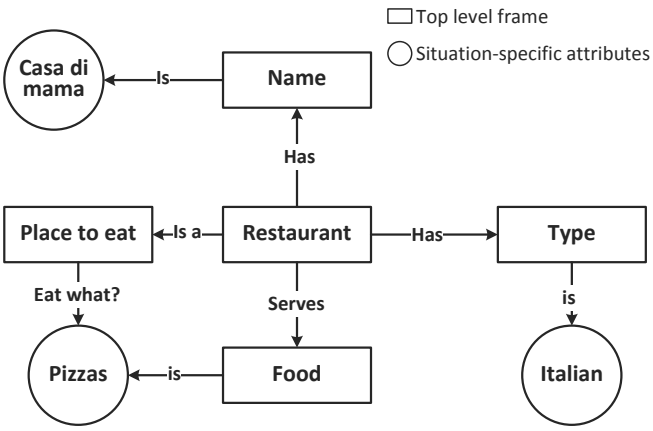

Figure 6.1: An example of a frame

Frames are inspired by psychological research; they represent stereotyped situations in the form of a group of interrelated concepts with a fixed structure. The 'top' levels of the frame represent knowledge that is always true for the modeled situation, for instance 'a restaurant is a place where your can buy food' or 'if the restaurant is Italian, then it serves pizzas'. When representing such knowledge in an ontology, a concept is created, e.g., *restaurant*. In turn, the ontology is extended by creating attributes (slots) for the given concept, e.g., *a restaurant is a place where you can buy food*.

Instances consist of assignments to the more general situation description, resulting in specific instances of a situation, e.g., 'Casa di mama is an Italian restaurant; it serves pizzas'. Attribute relations (slots) are generally applicable features of a con-

cept that may need further specification once an instance is added. For example, the concept *restaurant* may have the attribute that it *has a certain type*, e.g. Italian, tavern, or Chinese. However, which type the restaurant has may vary among instances. In such cases, to reason about a particular instance, the attributes of that instance (slots) need to be specified. Therefore, when adding an instance of a restaurant to the ontology, one must specify the type of that restaurant.

Frame-based ontologies offer the possibility to specify default (yet overridable) values in a given frame. For example, the default restaurant may be a tavern serving daily specials and local food. The specification of default values facilitates reasoning with incomplete information.

Concepts are organized in a hierarchy of classes. As such, concepts can inherit attributes from higher-level concepts (or superclasses). For example, the concept *restaurant* is a subclass of the concept 'business establishment'. As a result, it inherits all attributes belonging to 'business establishment', e.g., you have to pay before you leave. An ontology enables a system to reason about concepts, their attributes, and relationships between them (Kickmeier-Rust & Albert, 2010).

A distinction in ontologies

In the rest of this chapter, the following types of ontologies are distinguished.

Upper and lower ontology - Following the description of frames, an upper and a lower ontology is distinguished. An upper ontology (also known as a top-level ontology or foundation ontology) is an ontology that describes generic, abstract concepts and their interrelations. It is fixed and should be applicable across situations or applications without the need for alterations. The lower ontologies consist of more specific concepts, instances, and their interrelations. The idea behind this is that the intelligent system uses the upper ontology to reason about a given domain of discourse. In other words, the reasoning rules are independent of the lower ontology. This allows for reusability of the upper ontology. Moreover, new concepts can be specified in the lower ontology and can be processed by the reasoning rules based on the upper level frame specification.

Domain and system ontology - Comparable to the approach taken by the KADS methodology, the domain knowledge is described separate from the system design knowledge. As a result, the domain ontology contains the knowledge relevant to the task performed by the system, i.e. SBT. In contrast, the system ontology contains concepts relevant to the system's design. This separation is especially useful in knowledge-based systems, because in such systems the body of knowledge concerning the domain should be made reusable across various system designs.

Protégé Frames 3.5

In our research, we employed the Protégé Frames 3.5 ontology editor to implement the ontology (Gennari et al., 2003; Noy et al., 2000). Protégé 3.5 can be used to edit knowledge representations in the form of an ontology. A Protégé Frames ontology is comprised of classes, slots, facets, and constraints. Classes are concepts in the domain of discourse. Slots describe properties (or attributes) of classes. Facets describe properties of slots, e.g. the type of values that can be assigned, the number of values that can be assigned, the minimum and maximum values that can be assigned,

the default value that is assigned, etc. And axioms specify additional constraints on the ontological relations. A Protégé 3.5 knowledge base includes the ontology and individual instances of classes with specific values for slots.

6.4 Construction of the ontology

Once it has been established in what format the knowledge is to be represented, it is time to construct the ontology.

6.4.1 The construction process of the ontology

The ontology for PEGs was developed using the ontology construction procedure proposed by Noy & McGuinness (2001).

1) Determine the domain and scope of the ontology
The scope of the ontology was defined: automated SBT for First Aid training and the proposed design for PEGs.

2) Consider reusing existing ontologies
The 'task analysis'- ontology by Van Welie et al. (1998) was found to be suitable for reuse. This is further explained below.

3) Enumerate important terms in the ontology
All of the concepts relevant to automated scenario-based training that were not part of the ontology by Van Welie et al. (1998) were enumerated.

4) Define the classes and the class hierarchy
The body of research presented in this thesis was consulted to structure and define the identified concepts.

5) Define the properties of classes - slots
The properties of the classes were identified and added to the ontology.

6) Define the facets of the slots
The additional constraints on the slots were specified.

7) Create instances
Domain-specific instances, coming from the domain of First Aid, were added.

The ontology construction is a dynamic process: each step results in new knowledge about the quality and sufficiency of the concepts, attributes, relations and their definitions identified in previous steps, resulting in the necessity to move back and forth between these steps to check the consistency of new additions or alterations with the rest of the ontology. The resulting ontology for SBT is presented in the next subsection.

6.4.2 Reused ontology and extension

The ontology for SBT reuses the 'task analysis'- ontology by Van Welie et al. (1998) (see Figure 6.2). As explained above, task analysis is used to analyze and describe the relations between, e.g. tasks, situations, events, procedures, and tools within a given task domain (Annett, 2004; Annett & Duncan, 1967; Clark & Estes, 1996; Hackos & Redish, 1998; Merrill, 1978; Militello & Hutton, 1998; Resnick et al., 1973; Stanton, 2006; Vicente, 1995).

The ontology by Van Welie et al. (1998) can be used to describe how a task is to be performed. In SBT, learners aim to develop the required competencies to perform

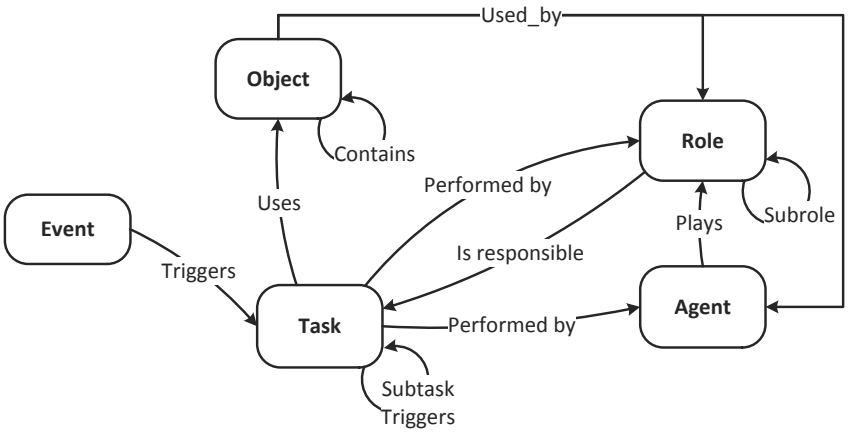

Figure 6.2: An ontology for task analysis by Van Welie et al. (1998)

a particular task by engaging in training scenarios. Therefore, the ontology by Van Welie et al. (1998) is a good place to start when constructing our ontology for PEGs.

To also cover additional knowledge regarding SBT, the ontology by Van Welie et al. (1998) needed to be extended. For instance, performance outcomes are only an indirect measure to assess the true purpose of training: competence development (Albert & Lukas, 1999; Koedinger et al., 2012). Task performance is derived from observable task behavior, whereas competency is a property of the person reflecting his ability to successfully perform a task. This separation promotes a learner-centered approach to learning task selection. To capture this idea, the ontology was extended with concepts that describe the relation between competence development and performance outcomes.

Another example is that the 'task-analysis'-ontology describes task performance at a relatively high level of abstraction. This, of course, is useful when the objective is to provide knowledge that is required to reason about tasks. However, in order to reason about SBT, additional knowledge is needed about the relation between tasks at an abstract level and the behavior of the learner within the simulated environment. During scenario-based training, tasks are enacted and staged within a simulated environment. In addition, tasks are structured, contextualized, and coordinated with the use of scenarios. Therefore, the knowledge representation, i.e. the ontology, must be extended with concepts that describe the simulated environment and its relation to the learning task.

6.5 An ontology for PEGs

This section presents the classes and relations described by the ontology for PEGs. This ontology consists of 1) a domain ontology that describes scenario-based training, and 2) the system ontology which describes the chosen design and technical solution.

6.5.1 The domain ontology

The upper domain ontology covers all concepts relevant to describe and reason about automated SBT, regardless of the chosen design. The domain ontology is divided into three layers: 1) the game world and everything in it; 2) the task domain, and 3) the didactic reasoning process. The complete ontology is depicted in Figure 6.3. The reader is invited to follow the step-by-step explanation of this graphic below.

The ontology description starts with Layer 2 concerning the Task Domain. A task requires particular competencies, which can be seen in Figure 6.3: the concept 'task' in the center of the figure is connected to the big green area named 'Competency' by an arrow that says 'requires'. There are three types of competencies: skill, attitude, and knowledge. Knowledge can again be distinguished by three different types: assumed knowledge - knowledge that the learner is assumed to possess before entering the training, domain knowledge - knowledge the learner will acquire during training, and situation awareness - knowledge that is contextual to the task performance and only becomes available once the exact task situation can be observed and assessed. Situation awareness is specifically important because it is the learner's awareness of information that is relevant to the way he/she should perform the task at hand; based on the learner's situation awareness, he/she is able to determine what procedure would be applicable to perform. A skill that is of particular interest is self-directedness, which is domain-independent and is related to a learner's metacognitive development: the extent to which a person is able to manage his/her own competence development process.

A task typically has an objective. This objective can entail the monitoring of a process or the establishment of a particular situation. A task is also dependent on the circumstances under which it is performed, e.g. danger, complications, etc. In addition, time (e.g. duration, timing) may be relevant to the task performance. Furthermore, a task relies on information conveyed by the world. Because the world in a personalized educational game is a simulated one, it is important that it still provides the required information for the learner to be able to act in a representative fashion. Therefore, required information should be made explicit and actively added to the game world design.

Tasks are decomposed following a Hierarchical Task Network approach. As such, a task can contain subtasks until eventually the subtask breaks down into a procedure (i.e. an action sequence). Actions are often associated with common errors people make while executing them. Such errors must also be explicitly represented in the ontology, because if they are not, the system will not be able to deliver error-specific feedback. Therefore, errors are specifically specified as alternatives to the proper actions in a procedure. These incorrect actions can in turn also be employed to produce erroneous examples.

Tasks are the responsibility of roles, which in turn can be adopted by characters. If a character (the learner, or an NPC) adopts a particular role, it becomes responsible for the tasks described in the task specification of that role.

We now move on to Layer 3 of the ontology: the didactics. This breaks down into two parts: 1) a description of the learner's experience and performance from a

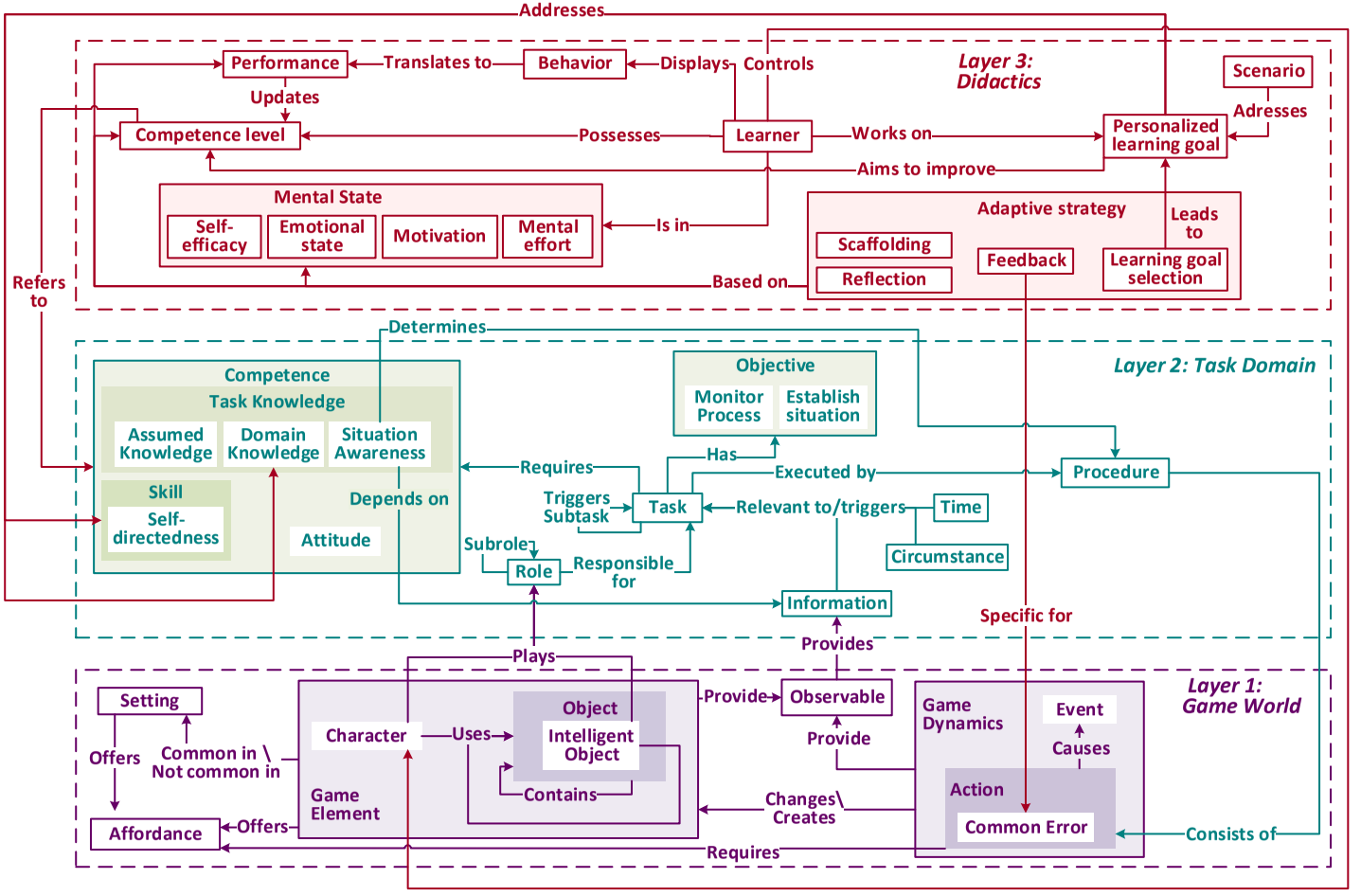

Figure 6.3: A graphical representation of the upper domain ontology (automated SBT).

didactic point of view, and 2) the role of the scenario within the training program or curriculum. Let us start again with the central concept of this layer: the learner. The learner controls one of the scenario's characters and, as such, plays one of the roles. As learners engage in task performance in the simulated environment, they display particular behavior. This behavior is interpreted in terms of task performance. The resulting performance assessment is used to update the learner's competency levels. In addition, the learner's mental state can be analyzed and estimated. The ontology distinguishes four components: the learner's self-efficacy (confidence in one's own ability), motivation, emotional state, and the mental effort invested during task performance. Furthermore, the learner is working on a personalized learning goal, which aims to improve a particular competency level.

The personalized learning goal is selected by applying an adaptive strategy called 'learning goal selection' and results in a subset of competencies that are currently suitable for the learner to develop during training. (How exactly this is determined is addressed later on in this thesis.) The learning goal is addressed in the scenario offered to the learner. There are several other adaptive strategies that can be applied to increase the didactic effectiveness of the scenario: providing feedback, which may be specifically targeted at some common error. In addition, the PEG may apply scaffolds, and stimulate the learner to reflect on his/her performance. Whether and how any of these strategies will be applied in any given training situation depends on the learner's mental state and his/her competencies.

The final layer of the ontology is Layer 2: the Game World. Again, let us start with the most central concept character. A character is an avatar in the game scenario that is controlled by a non-player (in that case it is called a non-player character, or NPC) or by the learner. Characters play roles within the scenario and execute actions that are consistent with their role and tasks. There are also other game elements available in the environment. These are called objects. A special type of objects is the intelligent object (i.e. controlled by an intelligent agent). It is an object that can exhibit intentional goal-directed behavior, such as a dog, a robot, a vehicle, or a fire. A setting is a scenery, a static background for the scenario, such as a kitchen or a park. Game elements can be common or not common within certain settings. For instance, a stove is common in a kitchen, but not in a park. And a tree is common in a park, but not in a kitchen.

Game elements can offer particular affordances. Affordances can be exploited while executing an action. An example is the following: when a character wants to break a limb (because this is part of the storyline), it needs to construct a plan to do so, such as 'fall from height'. The object 'stair' can provide the affordance 'fall from height' and enables the character to fall and break a limb. By assigning affordances to objects in the ontology, affordances specify the way in which objects can be used to perform particular actions.

The game dynamics consist of actions and events. Actions are performed by characters (or intelligent objects) because they are part of a procedure to perform a task and that task is a character's responsibility because of the role it has adopted. Actions cause events, which in turn change the world by changing or creating game elements. Events can also take place without being caused by a deliberate action, for instance, a tree may fall down simply from being old and rotten, but it may also

come down because someone chopped it down with an axe. Once an event takes place, the world changes, leading to new tasks becoming relevant. For instance, if indeed a tree comes down, it might be one's task to clear it from the road. As a result, game elements and game dynamics provide observables. However, these observables only lead to information and situation awareness if the learner picks them up accordingly.

6.5.2 The design ontology

In addition to the domain ontology, an upper design ontology was developed. The upper design ontology contains concepts relevant to the system's functional and technical design, thereby reflecting the researchers' design choices. In our case, this means that the design ontology covers all concepts relevant to the proposed PEG design. The design ontology is depicted in Figure 6.4.

As can be seen from the figure, the functional components are all represented in the design ontology. The subsystems of the scenario creator, to be discussed in Chapter 8, are also defined. The instructor and the learner are represented and identified as users. Because some of the concepts in the ontology will be introduced later on in the thesis, the remaining concepts are not further explained at this point. The design ontology is mainly used by the developers and/or in technical reports about the system design and implementation.

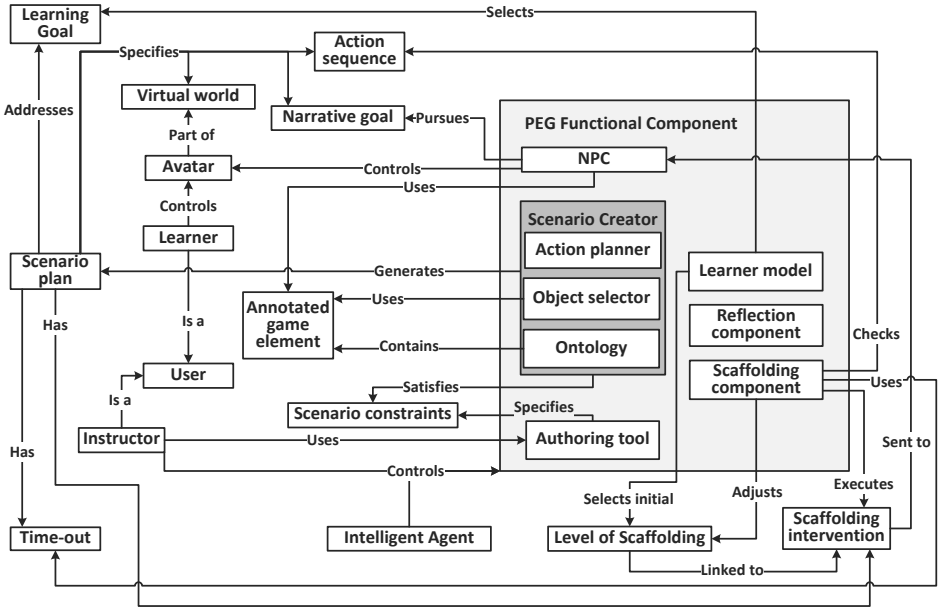

Figure 6.4: A graphical representation of the upper design ontology (PEG).

6.5.3 Implementation of the ontology in Protégé

The training domain employed throughout our research is First Aid. Therefore, the lower ontology contains a substantial amount of domain-specific concepts related to First Aid, such as domain-specific tasks, objects, roles, actions, and competencies. The complete ontology, containing the upper as well as the lower ontology, has been implemented in Protégé Frames 3.5 (*Protégé: a free, open source ontology editor and knowledge-base framework* 2013) and is available for download at http://mariekepeeters.com/INDIGO. This ontology also contains full definitions for all the concepts and relations in the ontology.

6.6 Evaluating the constructed ontology

The upper ontology constructed above *is expected* to provide a model to represent knowledge relevant to automated SBT (the domain ontology) and the proposed design for PEGs (the design ontology). However, the domain ontology *needs* to be accurate in order for the intelligent system to reason about its tasks, e.g. the NPC agents should be able to interpret narrative goals and produce believable behavior, and the scenario creator should be able to construct representative scenarios based on a learner's current competency levels. For this, it is important to *make sure* that the upper ontology (frame for SBT), as well as the lower ontology (model of First Aid) provide an accurate representation.

To ensure that the constructed domain ontology is indeed a sufficiently accurate representation for use within a PEG, it was verified with the use of domain experts. For this, a review study was conducted.

6.6.1 Expert-based review study to verify the domain model

A total of eight experts participated in structured interview sessions. The verification was conducted in separate sessions with three types of experts, respectively in the fields of education, serious games, and First Aid instruction. All sessions were conducted in the presence of two experimenters. Throughout the review process the following criteria, obtained from Gruber (1995) and Shanks et al. (2003), were taken into account: clarity, coherence, representativeness, accuracy, completeness, consistency, and conciseness. The participants were invited to look at the presented work with a critical eye. In addition, the experimenters encouraged the participants to be concise and to mark a concept when they considered it to be ambiguous or erroneously defined.

The upper ontology was verified using structured interviews with two researchers in the field of serious gaming, who verified the 'Game World' and 'Task Domain' area, and two researchers in the field of instruction and education, who verified the 'Trainee and Didactics' area. All four researchers were Human Factors Researchers of the Netherlands Organisation for Applied Scientific Research (TNO). The structured interview began by asking the participants to discuss the relevance of the concepts within the ontology area under review, and to add any missing concepts. Subsequently, they were asked to discuss the concepts' definitions. Next, one of the experimenters added the suggested concepts to a graphical display of the ontology

and drew relations between the concepts. The participants were invited to comment, ask questions, and offer suggestions for alterations, additions, or deletions while the ontology was (re)constructed.

The lower ontology was verified through structured interviews with four First Aid instructors in individual one-hour structured interviews. All instructors were recruited through their First Aid associations. The lower ontology was presented using graphical displays of First Aid skills, First Aid tasks, First Aid procedures, and First Aid knowledge. Participants were asked to judge the accuracy of the presented graphs, and were invited to check and alter the represented information until they felt it was accurate.

Notes about the participants' suggested changes were kept during all sessions by both experimenters. After each session, the experimenters cross-validated their notes and updated the ontology using the outcomes.

6.6.2 Results of the expert-based review

The verification sessions led to the following adjustments of the original ontology:

- The addition of 'adaptive strategy' as a superclass of feedback, scaffolding, etc.
- The omission of 'learning task' as a separate concept (because of ambiguity)
- The addition of 'intelligent object' as a subclass of object
- Changing 'feature' to 'observable' to describe the distinctive characteristics of a game element
- Changing the concept 'world knowledge' to 'situation awareness'
- The addition of 'attitude' to the types of competencies trained
- The addition of 'time' as a relevant constraint or characteristic of a task
- The addition of 'information' as a relevant aspect of a task
- The omission of 'closed task' and 'open task' as subclasses for tasks
- The omission of 'process-oriented' and 'goal-oriented' as task characteristics
- Several First Aid procedures were refined with minor corrections
- Several First Aid competencies were omitted because they were not an obligatory part of the training

6.7 Discussion

The use of an ontology to represent domain knowledge has several advantages when designing a complex knowledge-based system, such as a PEG.

First of all, the ontology facilitates consistency and clarity in use of terminology. As such, it provides a common language among stakeholders, designers, users, and components. When developing system components in parallel, the ontology provides conceptual and terminological agreement. All developers within the project employ the domain knowledge representation provided by the ontology when describing the architecture, writing code, and so on. As a result, documents and programming code are more concrete, easier to read, and easier to integrate.

Secondly, the ontology can be easily extended with new instances (e.g. tasks, actions, roles, objects) (Muller et al., 2012). Through the use of user-friendly interfaces, such as the one provided by *Protégé: a free, open source ontology editor and*

knowledge-base framework (2013), non-programmers are also able to add new instances and/or concepts to the domain knowledge ontology. By placing them within the structure provided by the upper ontology, the system is able to reason about those new concepts and use them right away. As such, the ontology offers a user-friendly way to author the content used by the system.

Lastly, by separating procedural knowledge from declarative knowledge, and domain knowledge from design knowledge, the domain ontology can be reused in other designs. Moreover, because of the separation of the upper ontology to describe SBT independently of the training domain, the upper ontology can also be reused in other training domains.

The constructed domain ontology has been reviewed by domain experts and was adjusted based on the outcomes of these reviews. As a result, it is considered to be appropriate for use in future prototypes of automated SBT. However, the research regarding the ontology presented in this chapter should be continued in order to improve the current proposal. It should be investigated whether indeed the ontology covers all knowledge required to support a flexible PEG organization as proposed in the previous chapter, and whether it is reusable across various training domains. For this, the PEG's design should be fully implemented in a complete prototype. In addition, it should be applied for other domains. It would also require user-based studies with instructors to investigate the usefulness of the ontology in extending the domain-specific vocabulary by domain experts (e.g. instructors).

6.8 Concluding remarks

Knowledge-based systems, such as PEGs and other systems for automated SBT, should be designed such that the knowledge structures they employ can be reused. Furthermore, the knowledge base should be easily extensible by domain experts. The ontology provided in this chapter serves as a first step in the direction of a separate, reusable, extensible domain knowledge base.

Part II: Component Designs, Prototypes, & Evaluations

Part I presented a high-level PEG architecture consisting of various functional components. In Part II of the thesis, each of these functional components, or intelligent agents, is investigated and designed following the sCE method (see Chapter 3).

Chapter 7 investigates the *scaffolding component*. It is able to adjust the level of challenge/support in realtime based on the learner's responses to events in the virtual environment. A wizard-of-Oz experiment compared adjustable scenarios to non-adjustable scenarios. Results showed that the realtime scenario adjustments did not diminish the learners' sense of immersion. Moreover, the adjustable scenarios were significantly better attuned to a given learner's needs than non-adjustable scenarios.

Chapter 8 investigates the *scenario creator*. It generates scenario plans based on a predefined learning goal and difficulty level. An experiment compared scenarios obtained from the scenario creator, laymen, and domain experts. Results showed that the scenarios generated by the scenario creator were of comparable quality to the ones obtained from the laymen. Of the highest quality were the scenarios obtained from domain experts.

Chapter 9 investigates the *authoring tool*. It allows instructors to provide constraints for the scenario creator, i.e. a learning goal, a setting, NPC roles, narrative goals for the NPCs, and attitudes for the NPCs. Based on the learning goal specified by the instructor, the authoring tool offers instructors suggestions for the roles, goals and attitudes of the NPCs. An experiment investigated the usability of the authoring tool and the quality of the resulting scenario plans. This was established by comparing it to a second version of the authoring tool that was identical except for the provision of suggestions. Results showed that the suggestions significantly enhanced the quality of the resulting scenarios and the usability of the authoring tool.

Chapter 10 presents preliminary results on the investigation of the *learner model* and the *reflection component*. The learner model is approached as a BDI agent. It updates its representation of the learner's competencies based on the learner's performance outcomes. Subsequently it uses this representation to determine a suitable learning goal for the learner and an appropriate initial level of scaffolding. Explorative research on the reflection component's design resulted in the identification of a major task for this component: fostering self-explanations by engaging the learner in a tutoring dialogue.

CHAPTER 7

The Scaffolding Component:

Controlling
support & challenge
in realtime

Abstract - This chapter investigates the *scaffolding component*, which controls scenario adaptations to keep learners on track of the learning objectives and to prevent situations that induce boredom or anxiety. This is achieved by attuning the level of challenge or support to the learner's behavior and performance in realtime (i.e. while the learner is playing the scenario).

To automatically produce this attuning behavior, the scaffolding component compares the learner's in-game behavior to a set of predefined behavioral cues. The outcomes of these comparisons are used to decide whether rules concerning particular scenario adaptations are applicable or not. Applicable adaptation rules are executed, resulting in behavior changes in the non-player characters. Scenarios that allow for this kind of adaptation are called 'directable', whereas scenarios that do not allow for such adaptation are called 'non-directable'.

A wizard-of-Oz prototype of the scaffolding component was developed and evaluated to investigate the effects of realtime adaptations on the *instructional quality* of the scenario. Results showed that the realtime scenario adaptations did not diminish the learners' sense of immersion. Moreover, instructors rated directable scenarios as significantly better attuned to learners' individual needs compared to non-directable scenarios.

This chapter is partially based on the following publications:

Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "Scenario-based training: director's cut", in: *Conference on Artificial Intelligence in Education*, 2011, pp. 264–272.

Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "The design and effect of automated directions during scenario-based training", *Computers & Education*, vol. 70, 2014, pp. 173–183.

> "We try to pull the learners out of their comfort zone. [...] We are able to introduce more challenging events covering topics that are part of next week's learning goals. [...] When a learner has been out of his comfort zone for too long, we will offer him a feel good mission, by throwing in some easy situations."
>
> **Helicopter Directive Operator Instructor**
> Interview, 17-02-2012

During SBT, the instructor and the role-playing staff members can influence the scenario as it unfolds. Examples of these realtime adjustments are the timing of events, the provision of hints or prompts, and other scaffolding techniques. Scaffolding has been shown to have positive effects on learning, such as an increase in performance outcomes, more time spent on task, higher learning efficiency, higher levels of perceived challenge, increased entertainment, and increased user-satisfaction (Liu et al., 2009; Mihalca et al., 2011; Sampayo-Vargas et al., 2013; Yannakakis & Hallam, 2009).

In order to offer learners truly autonomous and effective training opportunities, a PEG should also employ scaffolding techniques to refine the attunement between the learning situation and learners' individual characteristics. Therefore, this chapter investigates the scaffolding component of the functional architecture. The scaffolding component is responsible for automated realtime adjustments in the difficulty level of the scenario.

In Chapter 4, pages 53-55, the scaffolding component was specified by R06, R08, R10, R11, and R12. However, due to the non-chronological order of the chapters, the research presented in this chapter included only requirements R10 and R12:

R10 - Scenarios must include possibilities for scaffolding interventions
R12 - Scenarios must allow for the learner to become completely absorbed in the learning activity

These requirements need further specification before they can be implemented in a prototype. In order to come to a more detailed specification of what the scaffolding component should be able to do, more knowledge is needed about the way instructors conduct realtime control by adjusting the scenario as it develops. What is the goal of realtime adaptations? And when and how do instructors decide that such adaptations are prudent? The current chapter seeks to find answers to these questions as it presents a more detailed specification of the scaffolding component. Throughout this chapter, scaffolding adjustments are referred to as 'redirections', since they alter the direction of the storyline for both the role players and the learner. Scenarios that allow for such redirections are called 'directable scenarios'.

7.1 Task domain & support analysis

The Task Domain & Support Analysis (TD&SA) starts with an investigation of how human instructors exert scenario control during training to improve the quality of the learning situation (Subsection 7.1.1). The second part is a literature study on realtime redirections of the scenario and its intended effects (Subsection 7.1.2). The third part is an analysis of the technological opportunities, i.e. realtime adaptations of the behaviors of the NPCs (Subsection 7.1.3).

7.1.1 Task domain - *why* do instructors execute redirections

In regular training sessions, professional SBT instructors use their creativity and expertise to orchestrate the scenario in a desirable direction as the scenario progresses. They are able to recognize situations in which a learner seems lost, overwhelmed or bored. Based on their assessment, they may choose to redirect the scenario. They can do so by inserting or deleting events (e.g. appearing or disappearing enemies), and by instructing the role players (e.g. teammates, adversaries, bystanders) to behave in a fashion that the instructor considers suitable. The goal of such interventions is to achieve a training situation that fits the student's learning demands.

The question is how professional instructors evaluate the quality of training situations, and how they decide whether or not to apply redirections in the scenario. In order to investigate this question, over the four years of this project, multiple interviews were conducted with instructors in the safety & defense domain and the emergency health care domain (see Appendix A). Interviewees indicated that they decrease the level of support and increase the number of simultaneous events during training to increase the level of challenge. The primary indicator for adding or removing support or challenge is the student's task performance. When asked, instructors admitted that the observed work load and motivation also played a role, although they argued that these tend to go hand in hand with performance.

The literature shows that expert knowledge is often tacit rather than explicit (Dreyfus & Dreyfus, 2005). Yet, when the goal is to represent the experts' arguments and rules in a model, so that direction can be applied automatically, the cues and knowledge underlying the expert's assessments and decisions must be made explicit. The results from the interviews provide no conclusive evidence as to what the relevant cues and knowledge are. However, the human factors literature provides useful indications regarding this question. This is addressed in the following subsection.

7.1.2 Human factors knowledge - *when* is a redirection in place?

The aim of redirections is to *improve* the suitability of the learning situation, but how can one determine whether the learning situation is not already suitable as it is? Requirements R10 (employ scaffolding techniques) and R07 (challenge based on the learner's prior knowledge) refer to scenarios that guide and challenge the learner. These requirements form a trade-off and as such they should be appropriately balanced.

According to Murray & Arroyo (2003), a learning situation offers *optimal learning opportunities* if a learner is able to cope with the demands, while still being challenged to learn new things. People have preferences for moderately challenging tasks over simple and complex tasks (Abuhamdeh & Csikszentmihalyi, 2012; Shapira, 1989). Moreover, people enjoy a certain degree of uncertainty in the outcome of their performance over a guaranteed win (Atkinson, 1958; Malone, 1981). So it is not so much the challenge itself the learner seeks, but the satisfaction of successfully completing a challenge and feeling like a winner (Deci & Ryan, 2000; Harter, 1978). In that respect one could regard a suitable learning situation as one that challenges the learner such that it might result in failure, but there is a higher

chance of success.

The Zone of Proximal Development describes the set of learning situations that 'the learner is able to handle now with the help of someone (or something) more experienced, and independently in the near future' (Vygotsky, 1978). In other words, these are tasks that challenge the learner to develop new competencies. According to Vygotsky (1978), tasks that insufficiently challenge the learner lead to boredom and tasks that insufficiently support the learner lead to anxiety.

In the above, it has been established that:

- a learning situation is suitable if it appropriately challenges (and supports) the learner
- a learning situation is not suitable if it leads to boredom or anxiety
- if the learning situation is not suitable, redirections are in order

As such, we recognize and take into account two types of redirections: *supportive* and *challenging* redirections. Supportive redirections are needed when the events in the scenario cause the learner to suddenly be faced with an overly complex situation (i.e. anxiety). In such cases, action and support is required to lead the learner to a less demanding situation. Challenging redirections are executed when the learner is performing all the right actions, but is not being sufficiently challenged and the learner is motivated to take the training to a higher level (i.e. boredom). Redirections can entail adjustments in the availability of information, time constraints, the salience of certain cues, or the amount of simultaneous actions.

An estimation of whether the learner is appropriately challenged by the offered learning situation must also be described by explicit descriptions, for instance in terms of behavioral cues, or by events that imply confusion or boredom. One could specify cues that describe *the learner's performance or in-game behavior* to determine whether difficulty adjustments are appropriate. Examples of these are the amount of errors, the amount of time taken to complete a task, the learner's probability of failure, the efficiency of performance (attempts vs. successes), or the rate at which the learner's inventory is depleted (Hunicke, 2005; Hunicke & Chapman, 2004; Kickmeier-Rust & Albert, 2010; Sampayo-Vargas et al., 2013; Westra et al., 2012). Another method of cue specification and identification is based on *the learner's affective state*, e.g. anxiety, frustration, or boredom. These methods usually involve the use of psychophysiological measurements, such as eye-gaze direction, facial expressions, body posture, heart-rate variability, and skin conductance (Kapoor et al., 2007; Liu et al., 2009; Mota & Picard, 2003; Nacke & Lindley, 2008; Shaker et al., 2013).

7.1.3 Envisioned technology - *how* to redirect the scenario

The previous subsections defined the goal of redirections and the situations in which redirections are in order. This subsection continues with an investigation of how to establish realtime redirections in a way that is compliant with requirement R09 (no distractions/interruptions) and R12 (possibility to become absorbed).

In regular SBT, adaptation of the scenario takes place by altering the behavior of the NPCs (or other dynamic elements) with whom the learner needs to interact, e.g. teammates, opponents, patients, but also vehicles or fires. The advantage of adjusting the scenario through the behavior of the players is that it does not impede the

learner's sense of immersion (Bopp, 2006; Hecker, 2000; Herrington et al., 2003). There are several alternative approaches to adapt the game without the learner noticing any changes, such as 1) by spawning additional inventory items (Hunicke & Chapman, 2004), 2) drawing attention to particular parts or objects of the game world (Van der Spek et al., 2011), or 3) generating the game world in realtime (Shaker et al., 2013; Togelius et al., 2007). Since inventory items are not always part of the gameplay in serious games and, as of yet, generating the game world in realtime is quite complex, the most straightforward approach currently available is the adaptation of the NPCs' behaviors. Adapting the behavior of the NPCs by employing intelligent virtual agents is a well-studied method of inconspicuous scenario adaptation and appears to be a feasible method (Muller et al., 2012; Peeters et al., 2011a; Westra et al., 2012).

The intelligent virtual agent (NPC) is able to achieve an objective in multiple ways: it is equipped with multiple action plans to reach the same goal. One of those action plans may for instance challenge the learner and another may support the learner, but both plans still result in the achievement of the narrative objective. This enables the NPCs and other role players to execute the plan that is most suitable to meet the constraints regarding the overarching plotline as well as the appropriate difficulty level.

7.2 Scaffolding component - specification

Based on the TD&SA presented in the previous section, the requirements and claims identified in Chapter 4 can be further specified:

R10 - Scenarios must include possibilities for scaffolding interventions
- R10a: The level of support offered to the learner must be adjusted based on behavioral cues that indicate anxiousness
  - *PC10a.1* - This prevents cognitive overload
  - *NC10a.1* - This may reduce the level of challenge
- R10b: The level of challenge offered to the learner must be adjusted based on behavioral cues that indicate boredom
  - *PC10b.1* - This results in longer retention
  - *PC10b.2* - This results in lower chances of gaming the system
  - *PC10b.3* - This results in lower levels of boredom
  - *PC10b.4* - This results in a feeling of being sufficiently challenged
  - *NC10b.1* - This may lead to cognitive overload or anxiousness

R12 - Scenarios must allow for the learner to become completely absorbed in the learning activity
- R12a: The redirections must be established through the behaviors of the role players to reduce the chances of interruptions in the game flow
  - *PC12a.1* - This results in an increased experience of immersion (and flow)
  - *PC12a.2* - This results in increased levels of performance
  - *PC12a.3* - This results in increased self-efficacy

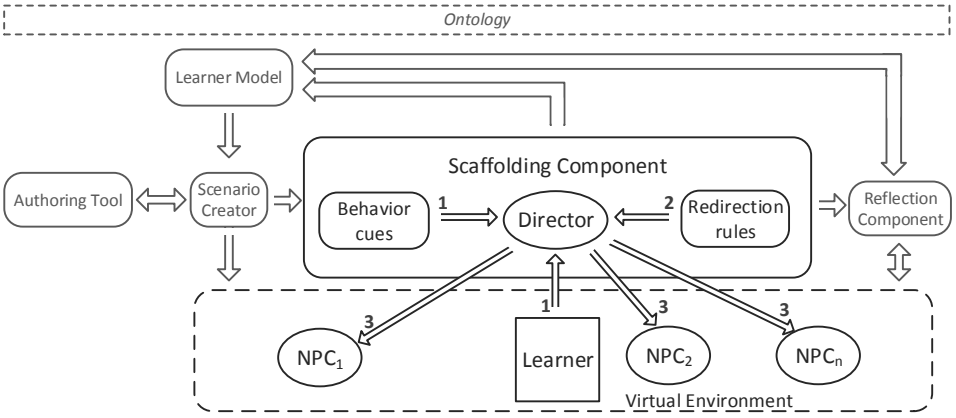

Figure 7.1: The scaffolding component design.

7.2.1 Scaffolding component - design

Important note: the research described below was conducted before the functional architecture (Chapter 4) and the agent organization (Chapter 5) were designed. As a result, the monitor agent was not yet part of the design; the director agent was still regarded to be the only agent operating behind the scenes, exercising top-down control over the NPCs. It was only until later that a separate component became part of the design with the sole purpose of administering realtime scaffolding interventions.

The design for the scaffolding component employs a director agent and several NPC agents (see also Figure 7.1). The director agent continuously checks whether adjustments in the scenario are required. It does this by monitoring the behavior displayed by the learner in the virtual environment and comparing it to a list of behavior specifications, called 'cues'. This step is indicated in Figure 7.1 with the arrows marked as '1'. Some behavioral cues are designed to indicate learning situations that are insufficiently supportive, whereas others are designed to indicate learning situations that are insufficiently challenging.

The monitoring and recognition of behavioral cues is not straightforward. However, in a virtual world, all actions that are available to the learner can be tagged and monitored since they are explicitly described in the code. Of course, if the system must also be able to recognize more intricate *sequences of actions* some type of pattern recognition would be required.

Upon detection of a behavioral cue specified in the list, the director continues to find and apply the associated redirection rule; each behavioral cue corresponds to a redirection rule that constitutes changes in the scenario to create a more suitable learning situation. This step is indicated with the arrow marked as '2' (see Figure 7.1). Redirection rules refer to either a supportive or a challenging redirection, depending on the behavioral cue that was recognized. The director applies this rule by propagating the adaptation to the NPCs. A message is sent to the NPCs stating exactly what type of redirection (supportive or challenging) is required. This step is indicated in Figure 7.1 by the arrows marked as '3'. In turn, the NPCs have

several action plans at their disposal that allow them to reach the same objective. Some of these action plans support the learner in accomplishing the learning task, whereas others impede the learner's task performance. The redirection rules instruct the NPCs on which type of action plan they should select. The NPCs interpret the instruction received from the director and change their behaviors accordingly.

7.3 Study into the applicability of redirections

This chapter argues that a training task should be dynamically adapted to the needs of the learner to achieve effective learning and immersion. In their everyday profession, instructors report to apply such regulation, albeit in an implicit manner. However, exercising automated control over training requires explicit understanding of what cues to monitor, and when and how to decide on redirections. It requires objective redirection rules that define precisely when a redirection is appropriate and what it entails.

Despite the ample attention in the literature to realtime redirection, and its apparent wide-spread use by instructors in practical training programs, very little is known about the realization of automated realtime redirections and their expected effects upon the quality of training.

In order to learn more about realtime direction, an explorative study was conducted. Four-scenarios were developed to train employees of an in-company emergency management team. For each of the scenarios behavioral cues were determined to be monitored in realtime and rules were developed to determine whether redirection should be applied or not. Participants took part in two scenarios in which redirections were applied when cues and rules marked this as appropriate; the other two scenarios were administered without any redirections. The goals of the study were 1) to learn whether it is possible to define appropriate cues and redirection rules and whether these cues and rules are feasible to work with, 2) to examine how realtime redirections are experienced by learners, and 3) to investigate whether the educational objective of a particular scenario is helped, undermined, or unaffected by realtime redirections.

7.3.1 Application domain

The application domain for this study was 'In-Company Emergency Management' (ICEM). ICEM entails the application of First Aid, fire fighting, and evacuation procedures performed by a trained team of company employees. This team remains in charge until the incident is sufficiently handled, or when the official emergency services arrive (i.e. ambulance, firemen, or police).

7.3.2 Method

Participants

The participants of the study were the members of one ICEM team in a research organization company from the Netherlands. The participants were uninformed about the research questions. Each participant individually took part in the scenarios as a learner.

Scenarios

In this study, three people engaged in four separate scenarios: the participant and two actors. A female actor played the role of victim and a male actor the role of bystander (e.g. colleague). The experimenter fulfilled the role of director, which implied two tasks: a) monitoring the training from an adjacent room by means of a video-connection and b) sending instructions to the actors through in-ear portophones. Two scenarios involved fire incidents, the other two involved medical emergencies (see Table 7.1). The scenarios were developed in consultation with an experienced ICEM trainer by following the steps outlined below:

Table 7.1: Descriptions of the four developed scenarios

ID	Type	Description
A	First Aid	a diabetic woman suffering from hypoglycemia
B	Fire fighting	a lady trapped within a room because of a small fire in a trash can near the door
C	First Aid	an unconscious cleaning lady, who fainted because of an intoxicating gas
D	Fire fighting	a woman with a broken hip - as a result of fleeing in panic from a fire - lying near a fire hazard

a. Learning goals and story lines. A set of learning goals formed the foundation for the story lines. An appropriate learning situation was created for each of the learning goals. Subsequently, the resulting learning situations were connected to form four coherent story lines.

b. Actor scripts. For each scenario, scripts were constructed that contained explicit behavioral instructions for both actors. The actors received two script versions for each of the four scenarios: a supportive and a challenging version. The supportive script version instructed each actor how to be helpful to the learner, e.g. by acting calm and stable, offering help, asking questions, or even suggesting possible solutions. In contrast, the challenging script version instructed each actor on how to make the situation more difficult, e.g. by acting panicky, being passive, or creating extra complications (for instance, running into a fire hazard).

The resulting storylines were similar for all participants, but the behavior of the actors varied according to the script version (challenging vs supportive). For example, in the supportive version of the 'diabetic patient' scenario, the learner spontaneously received relevant information, whereas in the challenging version this information was only provided upon request. And in the challenging version of the 'burning trash can' incident, the bystander would start taking actions to put out the fire by himself without consulting with the learner, thereby requiring the learner to be firm and take the lead, whereas in the supportive version the bystander would be passive and only take action when specifically instructed by the learner.

c. Director script. For each scenario, a director script was constructed. The goal of this script was to unambiguously specify which behavioral cues (i.e. what type of learner behavior specification) required what type of redirection rule (i.e. either sup-

portive or challenging). If the learner's behavior called for a redirection according to the script, the director instructed one or both of the actors through their in-ear portophones to switch to the other version of the script (i.e. from challenging to supportive or vice versa).

The redirection rules in the director scripts dictated a redirection from supportive to challenging behavior if the behavioral cue described in the script was observed by the director. Examples of such behavioral cues were: 'the learner makes eye contact with victim', 'the learner remains calm', or 'the learner gives clear instructions to the bystander'. When authoring the scripts, these behavioral cues were believed to indicate that the learner was not challenged by the scenario. This type of behavioral cues led to the application of a redirection rule instructing the actors to change their behavior to the challenging version of the script.

Likewise, the director scripts dictated a redirection from challenging to supportive behavior if the director observed particular behavior cues, such as: 'the learner neglects the victim and bystander altogether', 'the learner fails to perform more than x% of the required actions', or 'the learner performs inappropriate action A'. When authoring the scripts, these behavioral cues were believed to indicate that the learner was experiencing great difficulties with the scenario. The corresponding instruction to the actors would be to switch their behavior to the supportive version of the script, as described in the actor scripts paragraph above.

Procedure

Each participant was introduced to the actors before the training. Subsequently, they received explanations on their participation in a scenario-based ICEM training in a regular office-setting. Despite the obvious lack of realism, the participants were asked to act as if the incidents were real. They received information about the simulated environment (e.g. locations of the first-aid kit and fire extinguishers) and how to phone the receptionist who, in turn, could call up the official emergency services. Before the start of each scenario, the participant was asked to wait further down the hallway and the actors took their positions. The experimenter then went into an adjacent room and gave permission for the scenario kickoff. The experimenter/director followed the events through a video-connection. Each time the situation and script called for a redirection, the director issued the redirection by instructing the actor(s) through their in-ear portophones. Participants were oblivious with respect to any of the issued redirections. After completing each of the four scenarios each participant was interviewed about their experiences (decisions, insights, emotions, work load, motivation, etc.). After all four scenarios had finished, participants were interviewed to evaluate their overall reflections. All interviews were sound-recorded. All training sessions were video-taped using three cameras.

Experimental design

Effects of redirections during training were investigated within-subjects by applying redirections when appropriate in half of the scenario runs - the directable scenario runs. In the other half of the scenario runs, no redirections were applied (irrespective of their applicability) - the non-directable scenario runs. For each participant, either the first two administered scenarios were directable or the last two. As explained above, each scenario had two script versions (a challenging version and a

supportive version), and the scenario could thus be started in either mode. Half of the participants always started in the supportive version; the other half always started in the challenging version. In addition, half of the participants participated in the scenario runs in reverse order. The order of the scenario runs, the order of directable versus non-directable training, and the starting mode of the scenario were counterbalanced over participants and conditions. There were 8 groups in the study, depicted in Table 7.2. Participants were randomly assigned to a group. Pictures of the resulting scenario plays can be found in Figure 7.2.

Table 7.2: Participants were randomly assigned to one of the eight groups to counterbalance for any unwanted effects.

Group	Order	Initial mode	First 2 scenarios	Last 2 scenarios	n*
1	ABCD	Supportive	Directable	Non-directable	1
2	ABCD	Supportive	Non-directable	Directable	1
3	ABCD	Challenging	Directable	Non-directable	1
4	ABCD	Challenging	Non-directable	Directable	2
5	DCBA	Supportive	Directable	Non-directable	1
6	DCBA	Supportive	Non-directable	Directable	2
7	DCBA	Challenging	Directable	Non-directable	1
8	DCBA	Challenging	Non-directable	Directable	1

* n = number of participants in the group

a) b)

Figure 7.2: a) The director and the telephone operator watching the training session on the screen in the control room. b) The learner addressing and diagnosing the victim in the training environment.

7.3.3 Results and Discussion

The goal of this explorative study was to investigate 1) whether it is possible to define appropriate cues and redirection rules. In addition, it researched 2) how realtime redirections are perceived by participants. Furthermore, it examined 3) whether the educational objective of a particular scenario is helped, undermined, or unaffected by realtime redirections.

1) It was fairly straightforward to identify measurable and explicit behavioral cues that indicate the need for a scenario redirection, especially when doing this in consultation with domain experts. The resulting cues were unambiguous and traceable. The director was able to delegate the details of the redirection to the actors, allowing the director to focus on the behavioral cues and the decisions on rule applicability. However, on some occasions the continuous monitoring of the learner and retrieving the appropriate rule from the script proved to be quite straining for a human operator; some situations required the director to monitor two behavior cues simultaneously, which led to an experience of cognitive overload at times. This resulted in some delay in the establishment of the redirections and in some cases, missed cues. Another issue was that there were a few occasions in which an actor started to improvise by dropping hints that were not part of the original script. When asked about the improvisation, the actor responded that he saw the participant struggling with the scenario and unintentionally provided the improvised hint. In most cases, however, the recognition of the behavioral cues and the application of the redirection rules proceeded as desired. In general, the redirections affected the course of the scenario as intended.

This shows that well-specified sets of cues and rules allow for appropriate automated redirections in realtime. The director experiencing cognitive overload and the actor starting to improvise might be considered as a result of employing human operators. However, further research is required to find any conclusive answers regarding the feasibility of this approach regarding speed, processing, and consistency requirements when employing intelligent agents.

2) Questions concerning work load, motivation, self-efficacy, emotional state, and knowledge were included in the post-scenario interviews to investigate any possible effects of the redirections on the learner's personal experiences and knowledge. However, the resulting data did not allow for any conclusive results regarding the effects of redirections for two reasons: A) The scenarios offered the learners a large degree of freedom regarding their behavior. Different actions performed by the learner also resulted in different responses from the role players in the scenario. A logical consequence of this *personalization* is that scenarios showed a *large variability* in the way learners proceeded through the scenario, with some learners performing different sets of actions than others. Due to this variability, the resulting data referred to highly different training situations, thereby hampering the possibility to attribute results to the effects of the redirections. B) Furthermore, being in the directable *condition*, did not necessarily imply that the redirection *manipulation* actually took place. If the start-up mode already resulted in a suitable learning situation, no behavioral cues were recognized, no redirection rules were applicable, and no redirections were issued, even though the condition *did* concern a directable scenario. These two issues (A and B) rendered it hardly possible to draw any sound conclusions about the effects of the redirections on the learner's work load, motivation, self-efficacy, emotional state, and knowledge gains.

The study *did*, however, allow for some conclusions regarding the immersion experienced by the participants (i.e. claim PC12.1). All participants reported involvement and immersion during both directable and non-directable scenarios. Results from the interviews show that learners do not experience any differences between

directable scenarios and non-directable scenarios. None of the participants reported being aware of a change in the course of events, nor did they notify any differences between the directable and the non-directable scenarios. This seems to validate the design of inconspicuous redirections through the behavior of the role players since it diminishes the risk of breaks in immersion as a result of those redirections. One alternative explanation for this finding, however, is that high levels of cognitive load distracted the participants from noticing the issued redirections during the scenario. The reason for this explanation is that the scenarios proved to be quite straining for the majority of the participants. Further research should rule out this alternative explanation.

3) The experimenter and assistants were aware of possible biases in their experiences, yet reported that there *appeared* to be a positive effect of the redirections on the development of the scenario run. They indicated that when a redirection was issued, the training seemed to be more in line with the individual learner's competencies, whereas when a redirection would be appropriate but was not issued, the learner was believed to be led astray more often. This argument suggests that redirections result in a higher correspondence between the encountered training situations and the participant's competencies, and that refraining from redirection results in a less-productive learning situation. Possibly, the quality of a scenario run is positively influenced by adding redirections. Redirections may restore the effectiveness of scenario runs that would otherwise lose their suitability. In other words, learners may benefit from redirections, even if they are unaware of it.

The video recordings obtained during this study offer possibilities to further investigate this hypothesis. They allow for selection of the fragments showing a cue for a redirection, since redirections are perfectly traceable by means of the scripts. Since these fragments show occasions in which the redirection was actually issued (the directable scenario runs), as well as occasions in which the redirection was not issued (the non-directable scenario runs), the quality of the emerging learning situations can be compared to investigate the effects of the redirections. A new study was conducted. This study is described in the following section.

7.4 Evaluating the learning effects of redirections

Section 7.3 showed that it is possible to intervene and alter the course of events using scripted rules that specify how role players in the scenario should behave. However, we still need to investigate whether these redirections actually lead to more suitable learning situations (i.e. claim PC10a.1, PC10b.1, PC10b.3, PC10b.4, and PC12a.2). For this purpose, we define the *learning value* of a training situation to be *the extent to which the learning situation offers the learner optimal opportunities to achieve the learning goal at his (or her) own level*.

An experiment was conducted to investigate the effects of redirections on the learning value of the scenario (Peeters et al., 2011a). To answer this question, the video material coming from the study described in Section 7.3 was analyzed. We traced back the fragments containing the need for a redirection (a behavioral cue), as defined by the director's script (see also Section 7.3). Such a redirection might or might not have been executed, depending on the condition (directable vs.

non-directable). The resulting fragments were then judged by experts with respect to the learning value of the learning situation. Their judgments were analyzed to investigate the effects of the redirections on the learning value of the scenarios.

7.4.1 Research question

The research question is: *Will the formalized realtime redirections issued by a director during scenario-based training improve the learning value of the training scenario according to domain experts?*

We hypothesize that the formalized realtime redirections issued by a director improve the learning value of the training scenario as rated by professional instructors.

7.4.2 Methods

Raters

Six experienced instructors in ICEM (in-company emergency assistance, see also Section 7.3) were asked to rate the learning value of training situations shown in the video-fragments.

Materials

Footage. For this experiment we selected twenty video fragments from the study described in Section 7.3 as a test set. The selection of the fragments will be explained further on. Each fragment contained a part of a recording displaying a learner playing one of the aforementioned ICEM scenarios. All selected video fragments showed the learner performing a behavioral cueing related to a redirection. In half of the fragments shown to the instructors, the director executed all expedient redirections (directable condition) by telling the actors through in-ear portophones to switch between their behavior variations. In the other half of the shown fragments, the director was absent and the cues in the protocol were ignored (non-directable condition). No redirections were executed in this condition. Additionally, both conditions (directable and non-directable) contained five fragments that started off with the actors playing their supportive parts (supportive startup mode), and five fragments that started off with the actors playing their challenging parts (challenging startup mode).

Selection of the fragments. The nature of the repeated measures analysis requires an equal distribution of the amount of fragments over the different conditions. This means that the maximum number of fragments for each condition is the number of fragments in the condition containing the smallest sample. In our case the smallest condition sample contained five fragments. Based on this amount, we had to remove fragments from the other conditions to leave five in each condition. This was done by careful selection. The reasons for removal of fragments are presented below in order of applicability:

1. The redirection was issued shortly after the scenario had started. Therefore, the raters would not have much time to get a clear idea of the skill level of the participant.
2. The fragment contained a lot of action outside of the scope of the camera.

3. The redirection was not purely didactic; the purpose was also to maintain the believability of the storyline.
4. One of the actors started improvising, not sticking to the script as planned.

The resulting selection of twenty fragments contained only fragments that were representative for their condition and contained enough time before the moment the redirection was issued, so that the instructors could get a clear idea of the development of the scenario.

Questionnaire. The raters were asked to evaluate the learning value of the situation for a particular learner by answering the following question:

"The learning situation at this point in time offers the learner . . . opportunities to achieve the learning goals at his own level"

-3	-2	-1	0	1	2	3
absolutely no	no	not really any	maybe some	some	enough	outstanding

Procedure

The raters received an elaborate instruction to this experiment. The instruction contained an explanation of scenario-based training, exemplified by a video fragment. The four scenarios were explained and the learning goals for each scenario were explicitly pointed out. Lastly, the raters received instructions regarding the procedure of the experiment and explanations to the questionnaire. The raters were uninformed about the research question of the experiment.

Raters were then presented with two sets of video fragments (a practice set and a test set) following a standard procedure. The video fragment was introduced by a short description of the original scenario and the intended learning goal. The part of the fragment preceding the point of redirection was shown. At the cue for redirection, the fragment was paused and the raters were asked to rate the learning value (rating moment 1). Subsequently, the fragment was continued and paused again at the time the result of the redirection (or the lack thereof) became apparent. The raters were again asked to rate the learning value (rating moment 2). A diagram of the procedure can be found in Figure 7.3.

To test and enhance agreement among raters, they were presented with a practice set of 16 video fragments. The raters were encouraged to discuss their judgments in between series to reach consensus on how to value a learning situation. After the practice set, the experiment proper started, by presenting the test set consisting of twenty video fragments to the raters. The raters were not allowed to discuss their judgments, nor could they see each other's judgments. After the test set, the raters participated in a group discussion about their experiences with scenario-based training and their opinions about the video fragments.

Analysis

An intra-class correlation analysis was performed to assess inter-rater reliability. A repeated measures ANOVA was used to compute the effects of direction upon the rated learning value of the scenario.

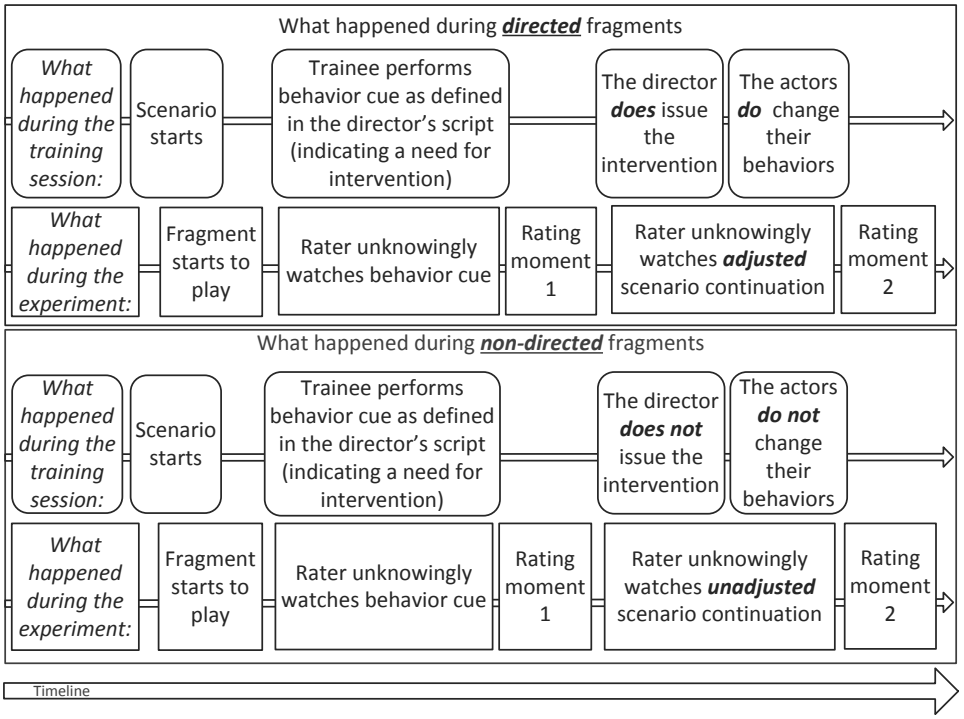

Figure 7.3: A graph of the procedure during the training session and the rating of the training scenarios

7.4.3 Results

Data exploration and inter-rater reliability

Forty ratings per rater (two rating moments for a total of twenty fragments) were entered into the analysis. The consistency intra-class correlation coefficient was 0.694 for average measures ($p<.001$). An inter-rater agreement between 0.60 and 0.79 is considered substantial (Landis & Koch, 1977), therefore we consider these data to be appropriate for further analysis.

Repeated measures analysis

In order to test whether the redirections of the director had an effect on the learning value, rated learning values were entered into a repeated measures analysis with two independent factors: director (presence vs absence) and start up variation (a scenario starting in the supportive vs challenging behavior variation). The results of this analysis are shown in Table 7.3.

A main effect of redirection was found ($F(1,5)=13.85$; $p<.01$, one-tailed). Examination of this effect showed that the directable fragments received a significantly higher learning value ($M= 1.08$; $SE=.31$) than the non-directable fragments ($M=.35$; $SE= .23$). A second main effect showed a significant difference between the learning value assigned to the two start up conditions ($F(1,5) =11.04$; $p<.01$,

Table 7.3: results of the repeated measures analysis

	director[a]	startup variation	director × rating moment[a]
F	13.85[b]	11.04[c]	27.34[b]
effect	.74	.69	.85
power	.84	.76	.98

[a]) one-tailed [b]) p <.01 [c]) p <.05

two-tailed). Overall, the video fragments in the supportive start up condition received a higher learning value (M=.98; SE=.31) than those in the challenging start up condition (M=.45; SE=.22).

Our main interest is the effect of a redirection on the situation's learning value. Therefore the differences between the director conditions (present vs absent) at rating moment 2 are of importance. There are expected to be no differences between the two conditions at rating moment 1. A significant interaction effect between director (presence vs absence) and rating moment (prior to vs after the cue for redirection) (F(1,5)=27.34; p<.01, one-tailed test), showed that indeed there was no significant difference between the directable and the non-directable condition at rating moment 1 (M=.60 vs M=.43, respectively). However, when a redirection was executed at rating moment 2 (director present), the learning value was significantly higher than when no redirection had taken place (director absent) (M=1.55 vs M=.27, respectively). The means belonging to this interaction effect can be found in the row 'overall' of Table 9.3.

Table 7.4: mean rated learning value (SE)

	director present		director absent	
	moment 1	moment 2	moment 1	moment 2
challenging start up	.43 (.34)	1.47[a] (.47)	.23 (.29)	–.33 (.28)
supportive start up	.77 (.39)	1.63[a] (.36)	.63 (.34)	.87 (.42)
overall	.60 (.31)	1.55[a] (.39)	.43 (.26)	.27 (.32)

[a]) p <.05, one-tailed

To find out whether the beneficial effect of the director's redirections is equal for both directions of redirection (from supportive to challenging or vice versa), one-tailed 95% confidence intervals of the means were computed for both start up conditions. The interaction effects were significant (p<.05, one-tailed) for both directions of redirection, (see also Table 9.3 and Figure 7.4), although the effect was stronger for supportive redirections (changing the actor behavior from challenging to supportive). When looking at Figure 7.4 it becomes clear that this distinction in effect can be explained by what happens in the non-directable scenarios over time: the rated learning value slightly increases in the supportive condition, whereas in the challenging condition, there is a huge drop in the rated learning value. In the directable condition, however, the increase in rated learning value is almost equal

(about 1 point) for both start-up conditions. Therefore, the difference between the non-directable and directable conditions depend on the start-up variation resulting in a stronger effect for the challenging variation (and the supportive interventions) compared to the supportive variation start-up variation (and the challenging interventions).

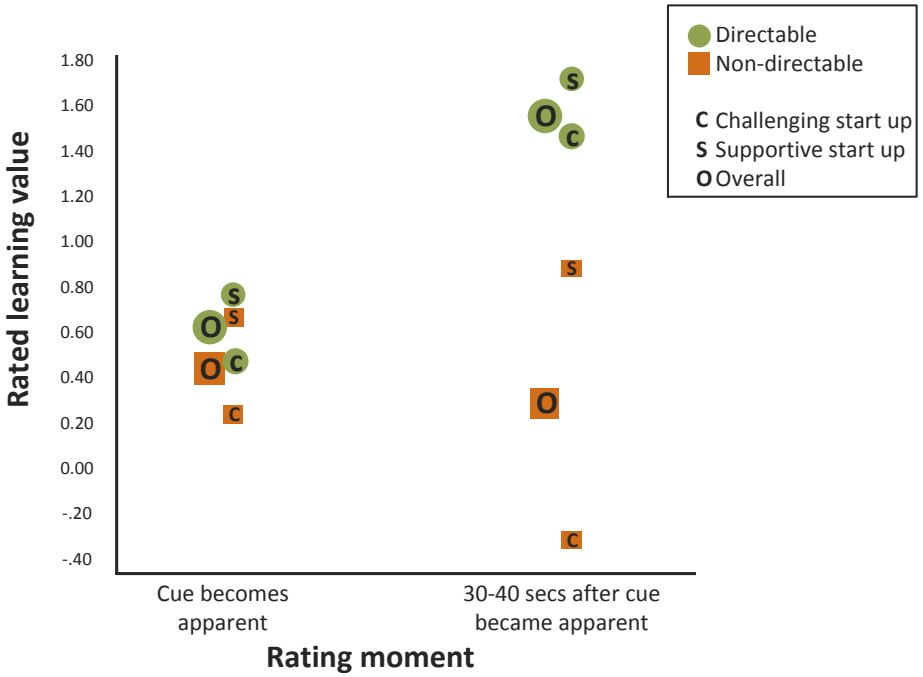

Figure 7.4: Mean rated learning values for the various conditions (SE)

7.4.4 Discussion

The goal of this experiment was to investigate the effects of redirections on the learning value of a training scenario. We created scripts for a director specifying when and how to intervene. Redirections consisted of adaptations in the behavior of the actors (NPCs) and were implemented in realtime, i.e. while the scenario unfolded. Video recordings of directable and non-directable training scenarios were shown to experienced instructors, who were asked to rate the learning value of the presented situations. Instructors were naive with respect to the purpose and design of the experiment.

Results confirmed our hypothesis. The rated learning value of scenarios that proceeded in a non-directable way, thus without adaptation, were at a fairly low level both halfway and at the end of the scenario. In contrast, the learning value of directable scenarios improved significantly as a result of the redirections instructing the actors to behave appropriately to the performance level of the learner. Thus, overall, redirections improve the learning value of scenarios. If we examine these

results more closely, split for supportive and challenging start up conditions, it becomes clear that scenarios that started in the supportive mode also offer some learning opportunities in the absence of a director. Even though the learner could use an extra challenge, the mere practice of already acquired skills is still considered useful. However, in the directable condition, it becomes possible to create an extra challenge for the learner, which results in an even higher learning value. A different pattern is found for the scenarios that started in the challenging mode. For these scenarios, the learning value drops dramatically over time when there is no director present to adjust the scenario. However, in the presence of the director, support is given to the learner, thereby most likely saving the learner from losing track and motivation and increasing the learning value of the training.

In a group interview conducted after the experiment, we explained the purpose and design of the study to the instructors and asked them for their experiences in their everyday work. The instructors stated that they find it hard to successfully intervene once they notice that a scenario loses track. They argue that they do realize it when a training situation requires redirection, but that they find it hard to specify beforehand what cues indicate this need. A more practical problem that they put forward is that - in their experience - participating actors tend to be unaware of what is needed, and that it is difficult for instructors to bring across appropriate adjustments to the actors while the scenario is playing. Instructors therefore consider it important to have appropriate and practical training instruments to execute the necessary control over their training scenarios.

The developed rules consisted of explicitly described cues that refer to learners' behavioral responses and redirections that were triggered by those cues. There were two types of redirections possible: supportive and challenging ones. The resulting rules showed to be not only highly desired by experts working within the training domain, but also to be beneficial for the learning value of the scenario.

These results are promising, but the expected effects of realtime adaptations still need further investigation. At this point, the studied effect involved only the learning value as perceived by instructors. This learning value referred to the suitability of the learning situation for a given learner (i.e. PC10a.1, PC10b.1, PC10b.3, PC10b.4, and PC12a.2). However, there are additional claims regarding the learner's cognitive load, self-efficacy, motivation, learning, flow, and involvement that still require further investigation. The validation of these claims will require a different experimental set-up that ensures that the learners will always experience manipulations of the scaffolding level when placed in the directable condition. Moreover, it requires for a clear tagging system to link the measurements during task performance to the type of learning situation the learner was in at the time of measurement. Future research should aim to find out whether the proposed scaffolding component design is indeed a valid proposal regarding all the claims previously mentioned.

7.5 Overall discussion of the component design

This chapter presented a detailed specification of the scaffolding component in the PEG architecture. The scaffolding component automatically manipulates the scenario in realtime to create training situations that meet the learner's needs by adjust-

ing the level of challenge and support. Our approach was to create predetermined redirection rules, consisting of a specific behavioral cue displayed by the learner (conditional), which triggered a redirection rule altering the behaviors displayed by the role players in the scenario (consequence).

Section 7.3 reported on a study into the applicability of this concept. Four directable scenarios were developed within the domain of in-company emergency assistance. This study showed that, even though the directions in the approach described here were simple scripted rules, they still led to variable, dynamic, and interactive scenarios. This personalization resulted in high variability among the resulting storylines. Due to this variability and personalization, comparisons between the scenarios were hardly possible. This shows that the validation of personalized products forms a challenge for the experimental design. In the studies described in this chapter, this challenge was overcome by carefully selecting comparable episodes of personalized user interactions and showing them to domain experts in a follow-up study, described in Section 7.4; an experiment was conducted which investigated the effects of the redirections upon the quality of learning. The results of this experiment showed that instructors judged the director's redirections in the scenario to induce learning situations with a significantly higher learning value than comparable situations in which these redirections were not executed.

7.5.1 Refinements of the component's design

The design proposed in this chapter employs a centralized top-down approach to control the behavior of the NPCs. This implies that the director must have knowledge of all the possible redirections in the scenarios including the possible action plans of both NPCs. As a result of this approach, the director's processing power may be exceeded if the number of scenarios, the number of possible redirections, and the number of role players increases, because the search problem increases exponentially. Furthermore, the necessity to familiarize both the role players and the director with the NPCs' possible action plans is redundant. For this reason, the multi-agent system described in Chapter 5, which is a later version than the one described in this chapter, adopts the auctioning mechanism proposed by Westra et al. (2012). The introduced auctioning mechanism distributes the knowledge required for the difficulty adjustments more efficiently. By auctioning the necessary adjustments, the role playing agents can decide for themselves whether they have a suitable action plan at their disposal. If they do, they can notify the director with a simple yes or no, or they might even extend their response with additional information such as the feasibility or believability of their action plan. The director can in turn select the most eligible candidate to carry out the necessary changes. This requires an extension of the NPCs and the director to participate in these auctions, but the director is no longer required to have any knowledge about all of the NPCs' possible action plans.

Another refinement would be the addition of a machine learning component to keep track of the effectiveness (or utility) of particular interventions. Characters would be able to update the effectiveness of their action plans after they execute it based on the failure or success of the intervention. This results in a more refined and suitable action plan selection mechanism. This approach resembles dynamic

scripting, described by Spronck et al. (2006), who employed reinforcement learning in the selection of script-based action plans to optimize the behavior of opponents in an entertainment game called Never Winter Nights. This idea has not yet been incorporated in the general system design proposed in Chapter 5.

One idea proposed by Riedl et al. (2008) could help to optimize the believability of the NPCs as they adapt their behavior to realize the redirection. It is not always straightforward how to warrant consistency of the NPCs' behaviors when they change their behaviors in service of the redirection, especially if the redirected behavior distinctly differs from their previous behavior. As a solution to this problem, Riedl et al. (2008) suggest the use of 'mix-in' behaviors that control the fade out of an action sequence and the fade in of some other action sequence. For example, if the agent was walking east and receives a redirection involving movement heading west, it would be strange to just suddenly turn around. A 'mix-in' behavior to break off the current action sequence might be the agent pausing for a moment, pretending to ponder, which offers the possibility for the start of any other action sequence, such as turning around and continue in the opposite direction. Of course this would require a mix-in behavior for each action sequence, but some of those can be reused in multiple action sequences.

A some-what different approach that is worth investigating, is the one suggested by Frasson et al. (1997). They proposed to develop NPCs that can take on multiple pedagogical roles in the scenario, such as the companion role, the teachable role, the troublemaker, and the tutor. In turn, a director agent is able to decide what role would be most suitable for an agent to adopt in order to support learning in the current scenario.

Another extension that could possibly improve the design of the scaffolding component comes from the work proposed by Marsella & Johnson (1998): the puppet master. The puppet master provides a more intricate way to assess the learner's in-game behavior. It monitors, interprets, and assesses the learner's performance by comparing the learner's behavior to appropriate courses of action given various possible situation assessments. In turn, it uses the outcomes of this comparison to derive the learner's most probable assessment of the game world situation. As such, the puppet master is able to, for instance, determine that the learner is missing some important information leading to an erroneous assessment of the situation. This more accurate assessment of the learner's misconceptions offers ways to provide more specific interventions

7.6 Concluding remarks

Based on this research it was concluded that the approach of a director agent that redirects the scenario when necessary, by issuing behavior changes of the NPCs, is promising. The realism and authenticity of the scenarios can be warranted, while maintaining control over the course of training.

The results of the studies showed that: 1) the current design enables automated control over the scenario, 2) according to domain experts, the realtime redirections resulting from the design lead to an improvement of the suitability of the scenario for a specific learner at a given point in time, and 3) the resulting redirections do

not cause any breaks in the immersion experienced by the learners.

Additional lessons learned are that the validation of personalized products requires an experimental design that allows for comparisons among highly variable user interactions. Furthermore, referring to Figure 7.4, we can conclude that support is beneficial for learning, but adaptive support is even better.

CHAPTER 8

The Scenario Creator:

Automated scenario generation

Abstract - This chapter investigates the *scenario creator*, which produces a large variety of scenario plans for a predefined learning goal at a predefined competency level. Scenario plans consist of 1) an action plan for the learner to perform, 2) narrative goals for the NPCs to achieve, and 3) environmental elements (e.g. objects, characters, a setting, circumstances) that should be added to the virtual environment before the scenario starts. Scenario plans are different from the actual scenario: they do not explicitly state all the events and actions that will take place during the scenario. This is necessary because the intelligent agents (NPCs) should be able to dynamically plan their behaviors as the scenario unfolds.

The scenario creator consists of a planner and an object selector. The *planner* plans the ideal action sequence for the learner to perform based on the predefined learning goal. The *object selector* then selects an appropriate context for these actions and places the necessary objects in the simulated environment. By doing so, the object selector creates the conditions for the learner to perform the desired action sequence at the predefined competency level.

To evaluate the scenario creator's design, a prototype was developed for the First Aid training domain. In an experiment, First Aid instructors were asked to evaluate a set of scenario plans with regard to their (1) representativeness for First Aid situations and their (2) suitability for the learning goal and competency level. The instructors were unaware that the scenario plans came from three different sources: the prototype, laymen, and domain experts. The results of the evaluation showed that the scenario plans produced by the prototype were at least as representative and suitable as the ones produced by laymen. The scenario plans produced by domain experts received the highest ratings.

This chapter is based on research conducted in collaboration with G. Ferdinandus, also documented in:

Ferdinandus, G. R., "Automated Scenario Generation - Coupling Planning Techniques with Smart Objects", MA thesis, Universiteit Utrecht, 2012.

This chapter is based on the following publication:

Ferdinandus, G. R., Peeters, M. M. M., Van den Bosch, K. & Meyer, J.-J. C., "Automated scenario generation - coupling planning techniques with smart objects", in: *Conference for Computer Supported Education*, 2013, pp. 76–81.

> "Serious games should include virtual scenarios that adapt to what and how players need to learn in a given context."
>
> **Lopes (2010, p.268)**

Manual scenario creation is a time-consuming process. As a result, most educational games reuse a fairly limited set of scenarios linked to each learning goal. Due to the limited number of scenarios, repetitions become inevitable at some point. Moreover, the number and types of learning situations the learner will encounter during training are restricted to the available set of scenarios. Furthermore, due to the limited set of scenarios it is not possible to conduct a fine-grained personal selection of the scenarios based on the learner's needs and abilities.

A possible solution to these problems is the *automated* generation of scenarios. Therefore, the current chapter investigates the *scenario creator*, a functional component that is responsible for the automated generation of scenarios.

In Chapter 4, pages 53-55, the scenario creator was specified by requirements R01, R02, R03, R04, R05, R06, R09, R14, R15, and R17. However, due to the non-chronological order of the chapters, not all of these requirements were included in the research presented in this chapter. The following requirements were addressed in the research presented here:

- R01 - Learners must be offered exercises in the form of complete storylines
- R02 - Scenarios must be interactive
- R03 - Scenarios should accurately represent those aspects of the scenario that are important for the construction of situational assessment schemas
- R05 - Over the course of training the learner must encounter multiple variants of scenarios that address the same learning goals and tasks
- R06 - Scenarios must be tightly linked to learning goals that are specified in terms of clear and unambiguous performance standards
- R09 - Scenarios must not include any unnecessary distractions

These requirements need further specification before they can be implemented in a prototype. In order to come to a more detailed specification of what the scenario creator should be able to do, more knowledge is needed about the way instructors create scenarios that meet the requirements described above. The current chapter investigates the authoring process of instructors and how it may be automated. Furthermore, it presents a specification of a scenario creator component, that automatically generates a set of scenario plans enabling other PEG components to produce suitable and representative scenarios.

8.1 Task domain & support analysis

This section aims to refine the requirements presented above by 1) investigating the scenario-creation process employed by human instructors (Subsection 8.1.1), 2) consulting the human factors literature to specify a method for the scenario-creation process (Subsection 8.1.2), and 3) identifying technologies that allow for the automation of the scenario-creation process (Subsection 8.1.3).

8.1.1 Task domain - how instructors create scenarios

The investigation of how professional instructors create scenarios consisted of three semi-structured interviews of approximately ninety minutes. One interview concerned a joint interview with three instructors in the safety & defense domain (i.e. Helicopter Directive Operator). The other two interviews were individual interviews with two instructors in the emergency health care domain (First Aid). See Appendix A for more information about the individual domains.

During the interviews, instructors were asked whether and how they, e.g., prepared their scenarios, took specific information about the learner(s) into account, adjusted scenarios in realtime, assessed the learner's performance, and reflected on the scenario afterwards. The complete interview guide for the semi-structured interviews can be found in Appendix B. From these interviews information was obtained about the way instructors ensure a scenario's 1) representativeness and 2) suitability for the learner's characteristics.

Setting the stage - The interviewees indicated that they start with the learning goal for the learner. Some learning goals demand very specific contexts to address them in a believable or representative way. For instance, it is not very common to incur an electrical injury in the middle of a forest, or to obtain a jellyfish sting in an office space. Therefore, instructors take the learning goal into account when choosing an available and representative *setting* in which the scenario can be staged. Examples of settings are a park, a beach, an office, a kitchen, or a western city. Once the instructors have decided upon a setting that is suitable for the learning goal, they continue to specify the actual task for the learner to perform during the scenario. Based on the setting, the task and the desired difficulty level, they continue to specify the scenario's distinguishing features and ensure that the setting and the available features or elements in the environment are consistent in terms of representativeness. For instance, it is not representative to find a field full of cows on the south pole or a bazaar in the center of a western city.

Additional fine-tuning - Instructors pointed out that they manipulate the difficulty level of the scenario by a) introducing elements in the environment that help or hamper the learner in performing the learning tasks, b) adding either obvious tools or less obvious ones (e.g. handkerchiefs vs. band-aids), or c) adding or withholding elements that tempt the learner to make a mistake. An example of this last method is provided by a military instructor who wanted to design a scenario concerning an ambush situation along a curvy road. The instructor deliberately placed the learner on the wrong side of the road at the start of the scenario so the learner was forced to actively pick the spot that concealed the ambush from oncoming traffic.

The results obtained from the interviews provide important insights in the thought process of instructors as they construct a scenario. Apparently, instructors employ a particular scenario-creation process that starts with the learning goal and the desired difficulty level, continues with the selection of an appropriate setting to stage the scenario in, and then moves on with the specification of a training task for the learner. Only after these three choices have been made, instructors continue by filling in the details of the scenario through the addition of elements and obstacles that produce the right difficulty level.

These insights tell us *what* instructors do, however, they do not inform us of *how* instructors make these decisions. Obviously, instructors use their knowledge about scenarios in general, and the training domain specifically, to select scenario features based on certain criteria. The next subsection examines the human factors literature to investigate criteria that may be of use when selecting scenario features automatically.

8.1.2 Human factors - establishing a task domain representation

The interviews reported above reveal that scenarios are generally created by following a step-by-step scenario-creation process:

Step 1. Determine a learning goal and difficulty level
Step 2. Stage the learning goal in a setting
Step 3. Specify a learning task
Step 4. Specify environmental elements

Each of these steps involves the selection or determination of some scenario feature, but based on what information? There must be selection criteria to guide these processes. These criteria can be specified based on the requirements:

Step 1. The learning goal and difficulty level must be determined based on the learner's current competencies (R07)
Step 2. The selection of the setting is guided by the requirement of representativeness (R03)
Step 3. The task must be complete (R01), varied (R05), address the learning goal (R06), and provide observable performance standards (R06)
Step 4. The scenario must evoke the desired procedure or strategy (R02), and it should not include any unnecessary distractions (R09)

The selection criteria (requirements) used in the process rely on knowledge about what tasks are suitable for which learning goals, what settings are appropriate for which learning tasks, what elements are common in which settings, and the effect of particular elements on the difficulty level of the learning task. Establishing a knowledge base that supports the selection steps in the scenario creation process requires a thorough analysis of the task domain.

A method often employed by researchers to identify the relations between, e.g. tasks, situations, events, procedures, and tools within a given domain is *task analysis* (Annett & Duncan, 1967). Task analysis originated from the fields of cognitive engineering, instructional design theory, and ergonomics, all of which require a clear analysis of a task domain to understand how a person can be supported while performing a particular task (Hackos & Redish, 1998; Resnick et al., 1973). The two most well-known approaches are: 1) hierarchical task analysis (HTA) (Annett, 2004; Merrill, 1978; Stanton, 2006), which produces an exhaustive description of tasks in a hierarchical structure of goals, sub-goals, operations and plans, and 2) cognitive work analysis (CWA), which aims to construct a detailed specification of all the constraints and relations among, e.g. protocols, situational cues, decisions, and possible consequences within the work environment (Clark & Estes, 1996; Hajdukiewicz & Vicente, 2004; Militello & Hutton, 1998; Moray et al., 1992; Vicente, 1995).

A thorough analysis of the task domain also addresses the notion that objects and other elements in the environment are more than just objects: they also have a con-

textual, task-related meaning. Environmental elements often play a part in the way the learning task is to be accomplished by the learner. The situational meaning of objects in relation to the performance of specific tasks is accurately described by the concept of 'affordance' (Gibson, 1977; McGrenere & Ho, 2000; Norman, 2002). An affordance is an interaction possibility offered by the current environment that can be employed by an individual in order to accomplish some goal or task. Affordances are independent of the individual's ability to perceive or recognize this possibility. In addition to the results from the task analysis, an affordance analysis is required to appropriately employ the available environmental elements when (automatically) creating a training scenario.

To conclude, this subsection provided insights in the selection criteria that can be used to guide each step of the scenario creation process. The next subsection will investigate how this knowledge can be represented and how the scenario-creation process can be automated such that it uses this knowledge appropriately.

8.1.3 Envisioned technology - automated scenario generation

To automatically generate suitable scenarios, we need technology that allows us to do two things: A) represent the knowledge that has been established from the task (and affordance) analysis and B) emulate the four steps of the scenario creation process, i.e. 1) determine learning goal and difficulty level, 2) select setting, 3) specify learning task, and 4) add environmental elements. Please note that this subsection presents the envisioned technologies separately for each of these steps and will explain why these solutions are indeed feasible with regard to the requirements. The integrated solution that combines these technologies into a single design is presented in Section 8.2.

Regarding the first issue (A), the ontology previously described in Chapter 6 provides a formalized knowledge representation of the task domain, environmental elements, their affordances, and the relations between them. Regarding the second issue (B), the technological solutions for the four steps are different for each step. Therefore, the steps will be discussed separately below.

Step 1

Determining the learning goal and difficulty level is taken care of by the learner model, see Chapter 10.

Step2

The selection of the setting is guided by the semantic knowledge stored in the ontology. Tutenel et al. (2008) showed that the addition of semantic information to virtual worlds can be used to guide the automated generation of game content (as opposed to manually created game content). The semantic information can facilitate the creation of virtual worlds: given the appropriate semantic knowledge, the design environment is able to, for example, automatically place streetlights along the roads laid out by the designer. Another example comes from the work by Lopes & Bidarra (2011), who employed semantic information about environmental elements to generate personalized content for games. They established this by annotating objects with the experience such elements might offer a player. Based on a player's preferences, the object annotations were employed to select objects that resulted in the

type of experiences that player seemed to enjoy. Examples of these are the selection of a time bomb for players who enjoy action and time pressure, and the selection of hidden chambers for players who enjoy exploring the environment and searching for hidden secrets. Another example is the work by Kallmann & Thalmann (1998), who investigated the use of semantically annotated objects (or 'smart objects') to be used by intelligent agents. The smart objects could reveal their use to the agents upon request. This allowed the agents to exploit the affordances of objects without the need to provide the agents with knowledge about all the available objects in the environment in advance.

We propose a similar approach to guide Step 2 in the scenario creation process. Step 2 ('select setting') is accomplished by employing *affordance relations* that are available in the ontology. More specifically, this step employs the relations between settings and learning goals: settings offer the affordance to practice particular learning goals. An example is that the setting 'the beach' offers the affordance of practicing the learning goal 'diagnose and treat a jellyfish sting'. The specification of these affordance relations facilitates the fulfillment of the following requirement:

Representativeness (R03) - By selecting a setting that affords practicing the learning goal, the coherence between the setting and the task can be ensured.

Step 3

As Niehaus & Riedl (2009) have previously shown, the specification of the learning task can be approached as a *planning problem* that can be solved with the use of hierarchical task network (HTN) planners (Ghallab et al., 2004; Nau, 2007).

The objective of HTN planners is to plan the concrete actions for performing a (collection of) task(s). To achieve this objective, HTN planners employ a domain representation in the form of hierarchical task networks (also see Figure 8.1). Using domain specific rules, called *decomposition methods* (DMs), tasks can be recursively replaced with decreasingly compound tasks until there are only primitive tasks (i.e. *actions*) remaining. When the decomposition process comes to a halt, a complete action sequence has been obtained. The DMs employed by the HTN planner are based on expert knowledge. As a result they are intuitive to understand for non-programmers. They specify exactly how a task can be decomposed into subtasks while taking certain constraints into account. This makes HTN planners especially suitable for applications where domain experts without a background in computer programming are required to provide domain knowledge for the system.

Together, the DMs specify all the different versions of the abstract task within the task domain. Figure 8.1 shows a limited set of HTNs and DMs for the domain of "First Aid". One action sequence that can be derived from Figure 8.1 is a result of the sequential application of DM0, DM2, DM3, DM4, DM6, DM8, and DM12. The action sequence obtained after applying this DM sequence is: 1) recognize fire, 2) extinguish fire, 3) reassure bystander, 4) ask bystander what happened, 5) call 911, and 6) cover eye.

Figure 8.1 does not include any constraints regarding the application of the DMs. As such, there is no way to guide the DM selection process and as such the consistency of the task performance cannot be fully supported. For instance, the third action in the example action sequence was 'reassure bystander', therefore there must

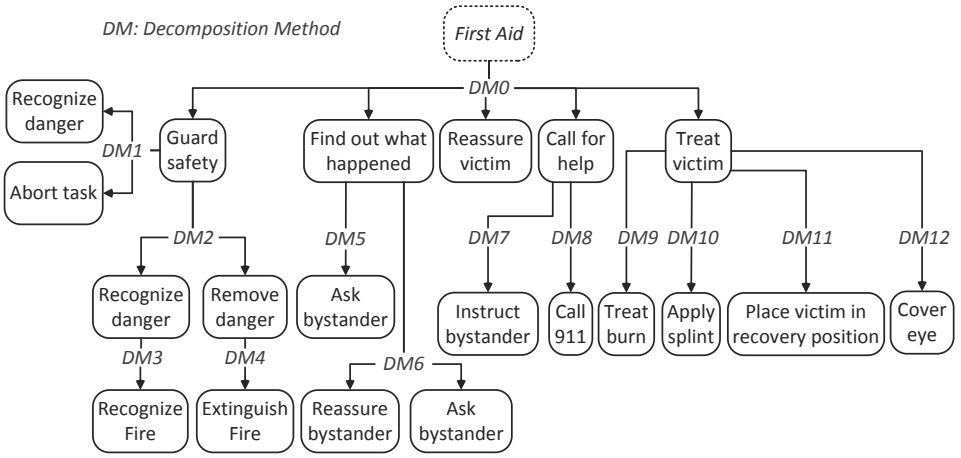

Figure 8.1: An (incomplete) hierarchical task network

be a bystander present. In such a situation, the learner should ask the bystander to call for help instead of dialing 911 him-/herself. So a rule should be added saying that if DM5 or DM6 has been applied, then DM7 should also be applied.

To facilitate such relations between DMs, HTN planners are able to handle different kinds of constraints, e.g. precedence and prerequisite constraints. Additional bookkeeping mechanisms are required to ensure that the decomposition process adheres to such constraints (Nau et al., 2003). HTN planners that only allow the order of the subtasks in the decomposition method to be fixed are able to perform *forward search strategies*: the complete plan can be determined in advance of the task performance. This is beneficial for the creation of scenarios, because it allows us to determine the desired action sequence that is to be performed by the learner.

As has been previously established, a learning task must be specified such that it is representative, complete, and varied, and provides observable performance standards. Moreover, it should address the learning goal. The employment of an HTN planner facilitates these requirements as follows:

Representativeness (R03) - The resulting learning tasks are representative because the DMs are provided by domain experts.

Completeness (R01) - It is possible to enforce complete tasks by initiating the learning task construction process from the top node (applying First Aid). On a side note, they also provide the possibility to design scenarios for part-task training: it is possible to use a subtask as the top-node in the construction of the learning task.

Variety (R05) - HTN planners facilitate a varied set of learning tasks that train the same learning goal: various sequences of DMs can be applied to produce completely different action sequences while still incorporating the same task. For example, every sequence of DM applications in Figure 8.1 will incorporate the subtask 'Guard safety' because DM0 is the only available DM to decompose 'First Aid'. Yet there are still multiple sequences of DM applications that all include DM0, hence at some point include 'Guard safety'. For instance, DM1 decomposes 'guard safety' into the

subtasks 'check the environment for danger' and 'abort task', whereas DM2 decomposes 'guard safety' into 'check the environment for danger' and 'remove danger safely'. As a result, the same learning goal is addressed, but the intended action plan, hence the scenario, will be quite different.
Performance standards (R06) - Learning tasks created by HTN planners provide observable performance standards in the form of an action sequence. The learner's choices and actions can be compared to the ones in the action plan.
Embedded learning goal (R06) - The learning goal should be part of the learning task. This can be accomplished by applying a sequence of DMs that decomposes the subtask described by the learning goal. For instance, if the learning goal is to 'treat a burn', then the decomposition process should at some point select DM6 to ensure that the learning goal is a part of the learning task (see Figure 8.1).

Step 4

The addition of environmental elements is facilitated in a way that is comparable to Step 2. However, Step 4 employs *two* types of relations in the ontology:

1. a relation between settings and elements that describes how common a certain element is within a particular setting, and
2. an affordance relation between actions and environmental elements, indexed with an indication of how difficult the use/recognition of the affordance is

These two relationships are described in the ontology and employed by the scenario creation process as follows:
Representativeness (R03) - By selecting only elements that are common within the setting (relation 1), coherence between those aspects can be warranted.
Evoke procedure (R02) - Relationship 2 facilitates the selection of environmental elements that evoke the desired behavior. By including environmental elements that afford particular actions it can be ensured that the desired action sequence for the learner is facilitated by the environment.
Match difficulty level (R07,R09) - Relationship 2 facilitates the selection of environmental elements such that the resulting scenario offers the learning task at the desired difficulty level. For each element that is added to the environment, it is possible to derive its effect on the difficulty level of the resulting scenario. Hence, it is possible to guide the scenario-creation process to such that the desired difficulty level is taken into account.

8.2 Scenario creator - specification

The previous section investigated the scenario-creation process, the selection criteria it employs, and the technological means to support the automatic execution of that process. This section continues with the design of the scenario creator. The *general PEG requirements* identified in Chapter 4 are refined and expanded into the following requirements for the *scenario creator*:

R01 - Learners must be offered exercises in the form of complete storylines
  R01a - The action plans for the NPCs and the learner must cover a complete task performance
R02 - Scenarios must be interactive

R02a - The elements added to the scenario should afford the execution of the action plan

R03 - Scenarios should accurately represent those aspects of the scenario that are important for the construction of situational assessment schemas
- R03a - The setting of the scenario must be appropriate for the learning goal
- R03b - The additional elements in the environment must be consistent with the selected setting

R05 - Over the course of training the learner must encounter multiple variants of scenarios that address the same learning goals and tasks
- R05a - Similar learning goals should lead to various learning tasks
- R05b - Similar learning tasks should lead to various scenario plans
- R05c - Similar scenario plans should lead to various scenarios

R06 - Scenarios must be tightly linked to learning goals specified according to clear and unambiguous performance standards
- R06a - The scenario-creation process must provide the preferred action sequence to facilitate the assessment of the learner's performance at playtime and clear communication of this assessment to the learner

R09 - Scenarios must not include any unnecessary distractions
- R09a - The environmental elements in the scenario should be consistent with the desired difficulty level
- R09b - The environmental elements in the scenario should be reduced to a minimum

The claims have been omitted from this list, because these are not investigated in this chapter. The proposed design that will be investigated throughout the rest of this chapter is presented in the following subsection.

8.2.1 Scenario creator - design

The proposed scenario creator design combines an HTN planner with an element selector. The HTN planner plans the learning task in a way such that it addresses the learning goal. The element selector employs the semantic relations stored in the ontology to add suitable environmental elements to the virtual environment. Figure 8.2 depicts the general design of the scenario creator.
The scenario creator receives the following *input*:

- The *learning goal* for which the scenario is supposed to offer practice opportunities. This input is required.
- The *difficulty level* at which the learning goal should be practiced (i.e. a value between 0 and 1, representing the skill level required to successfully perform the learning task, where 0 indicates no skill and 1 represents mastery). This input is required.
- The *authoring constraints* received from the instructor. This input is optional.

The scenario creator uses this input to produce the following *output*:

- An *action plan*: a list of actions the learner is expected to perform.
- The *goals for the NPCs*: a list of goals for the intelligent virtual agents (e.g. 'light a match').
- The *environmental elements*: a list of the required elements to add to the scenario. Optionally, additional information is provided regarding the difficulty level at which the elements should offer their affordances.

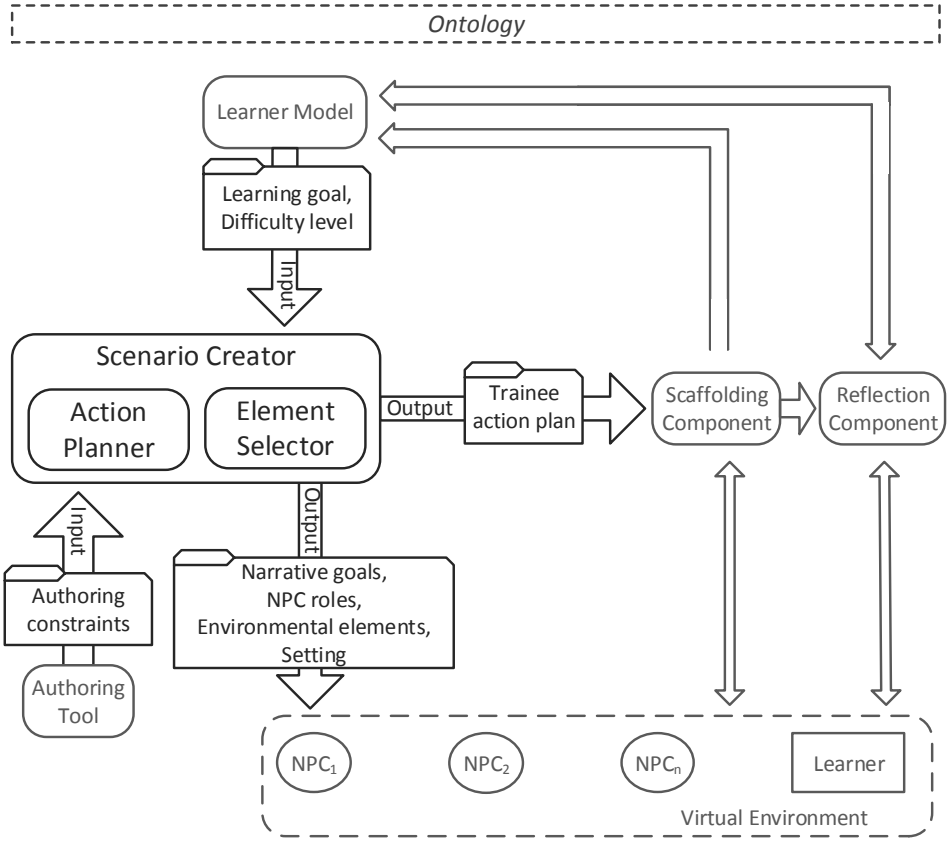

Figure 8.2: An overview of the scenario creator's input, output, and subcomponents

The output of the scenario creator is used by other PEG components to produce varied scenarios that all address the same learning goal: 1) the action plan is used by the monitor to assess the learners performance, 2) the goals for the NPCs are used by the director to auction them among the role players, and 3) the environmental elements are used by the game engine to create the appropriate virtual environment. The scenario creator employs *three main subcomponents*: the ontology, the action planner, and the element selector.

Ontology

The ontology contains information about the available environmental elements and settings, their affordances, the tasks and actions, the decomposition methods (DMs) for the HTN planner, preconditions for actions, and their difficulty levels, and so on.

Action planner

The *action planner* is an HTN planner that uses the learning goal and difficulty level (provided by either the learner model or the instructor) to decompose the abstract task ('apply First Aid') into a complete, coherent, detailed, and personalized ac-

tion plan for the learner that offers the possibility to practice the learning goal (see Figure 8.3). The action planner uses two types of information about the DMs to decompose tasks into subtasks: 1) affordance preconditions and 2) difficulty levels.

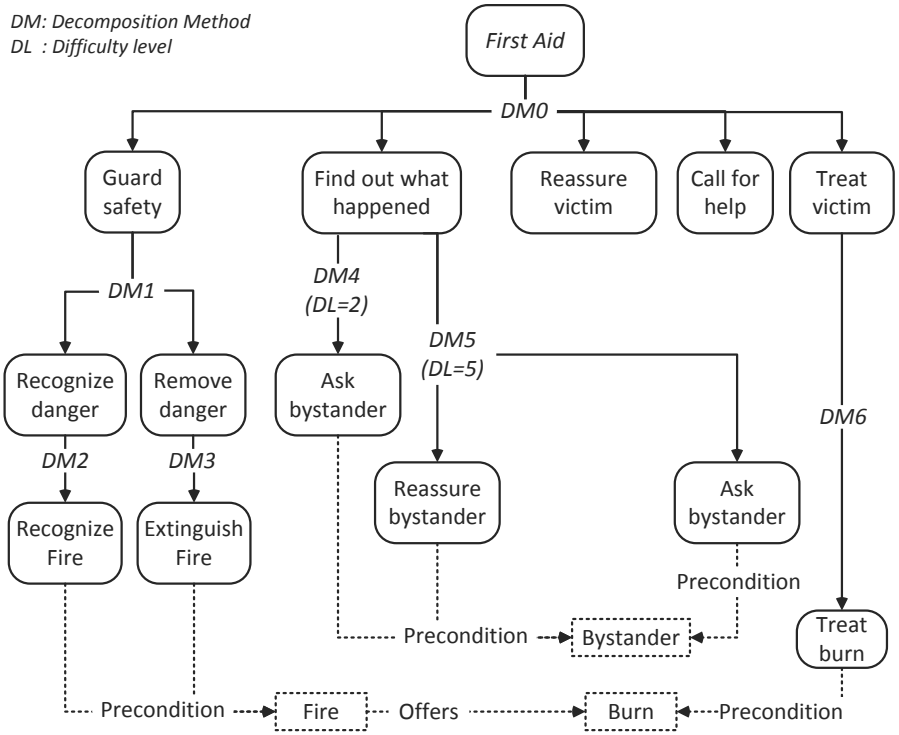

Figure 8.3: A display of a selection of tasks, decomposition methods, preconditions, and difficulty levels in First Aid

Affordance preconditions reflect the required affordances to apply a DM. An example of an affordance precondition is that to decompose the task 'remove danger' into 'extinguish fire' the affordance 'fire' must be offered by the environment. Another example is that decomposing the task 'treat victim' into 'treat burn' requires the affordance of a 'burn injury'.

Difficulty levels describe the difficulty of the set of subtasks that are added to the task plan after applying the DM. An example of the difficulty level is that a decomposition of the task 'find out what happened' into the single task 'ask the bystander what happened' (DM4) is less difficult than decomposing the same task into two subtasks: 'reassure the bystander' and 'ask the bystander what happened' (DM5).

Affordance and difficulty preconditions guide the selection process of the DMs. This allows the action planner to 1) influence the difficulty of the scenario by comparing applicable methods in terms of complexity and 2) select decomposition methods that are coherent with previously selected DMs and elements. For example, the action planner may pick the more difficult DM5 ('reassure the bystander' and 'ask the bystander what happened') to establish the desired difficulty level (e.g. desired

difficulty level = 5). Furthermore, the action planner may select DM3 to decompose 'remove danger' into 'extinguish fire', because its precondition (the affordance of 'fire') has already been established through the application of a previous DM, for instance, if DM2 has already been applied. The reuse of previously selected environmental elements fosters coherence: the learner is required to recognize danger, which is the fire, and to remove the danger, also the fire. If the fire would not be reused, the object selector might add a different object to provide the affordance of danger for the step 'remove danger', such as a short circuited electrical appliance. This would result in incoherence and confusion on the side of the learner. In other words, reuse of available environmental elements to satisfy the preconditions of actions also reduces the number of relevant situational cues, hence the cognitive load for the learner.

The action planner follows a step-by-step procedure to decompose the high-level task ('apply First Aid') into a concrete action sequence. The steps in this procedure are explained below (see Figure 8.4).

1. Incorporate the learning goal. Scenarios must be targeted at the learning goal. To establish this constraint, the action planner starts its process by searching for a sequence of DMs to decompose the high-level task ('Apply First Aid') into a sequence of tasks that contains the learning goal. For example, in Figure 8.3, if 'recognize fire' is the learning goal, then DM0, DM1, DM2 will be identified as a sequence of DMs that needs to be part of the task decomposition process in order to incorporate that learning goal in the learning task. Moreover, by decomposing a task into an action sequence that contains the action 'recognize fire', the precondition of 'fire' needs to be established. How exactly such prerequisites are established will be explained in the section describing the element selector further below.

2. Select (non-primitive) task to decompose. Once the planner has established a (partially specified) task sequence that contains the learning goal, the system iteratively addresses the remaining non-primitive tasks by randomly selecting a task to decompose into primitive actions. This randomized approach promotes variety among scenarios with the same input. So, using the same example (Figure 8.3): the tasks 'find out what happened' and 'treat victim' still need to be decomposed into more primitive tasks. This step will randomly select one of them to decompose. For the example, we assume that 'find out what happened' is selected.

3. Select decomposition method. The selected task is decomposed by applying an eligible DM. The selection of the DM is guided by two considerations: the preconditions already established by the game world so far, and the difficulty level of the DM. For instance, the precondition of 'Fire' has already been established by decomposing the task of First Aid into an action sequence that contains the action 'recognize fire' by applying the DM sequence DM0, DM1, and DM2 (see Figure 8.3).

First, the planner selects the methods with the highest number of established preconditions. If this step results in multiple candidates, the selection process is further guided by the DMs' difficulty levels. For the learning goal, the action planner selects the DMs that result in a difficulty level close to the desired difficulty level (provided by the learner model, see Chapter 10).

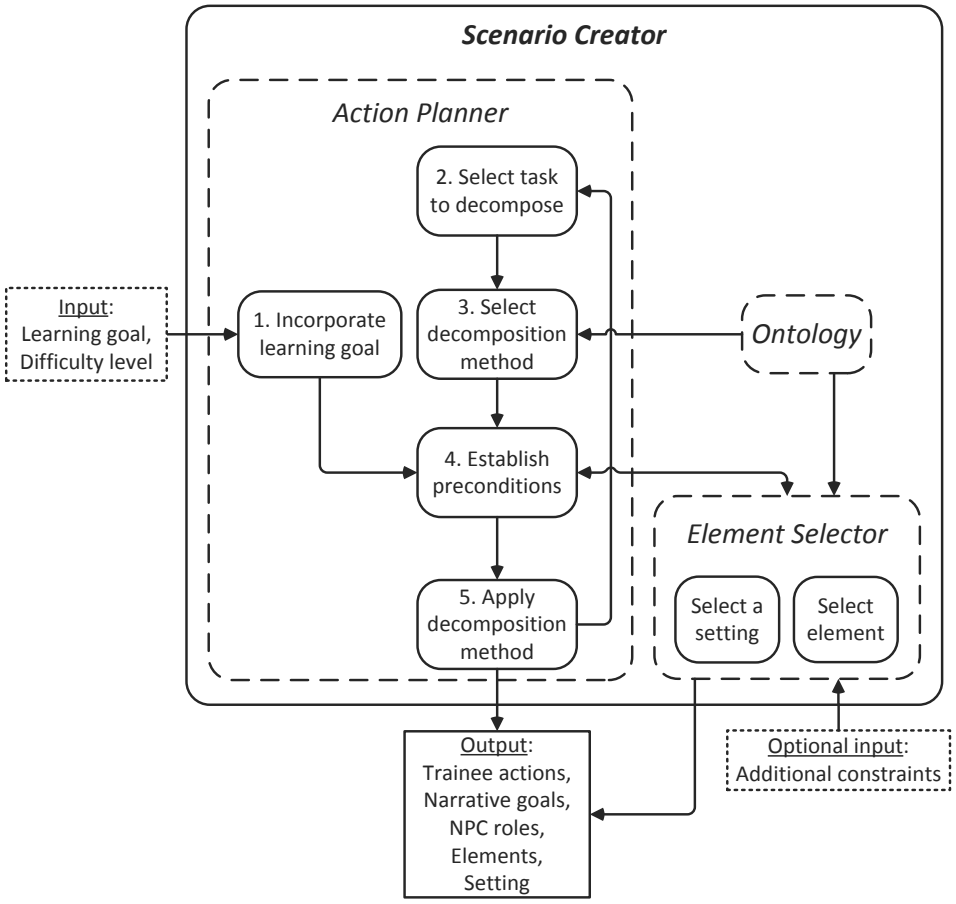

Figure 8.4: Flowchart depicting the planning process of the scenario creator.

4. Establish preconditions. Before a method can be applied, all its preconditions (required affordances) must be fulfilled. The action planner forwards the affordance requests to the element selector (explained below). Some affordances can only be offered by accomplishing a certain task, e.g. a 'fire' can only be offered after it has been lighted. The element selector returns such tasks to the action planner, which in turn plans an action sequence for a virtual agent to accomplish the task (e.g. create fire by setting the curtains on fire with a lighter). The action sequence may in turn require new affordances from the game world (e.g. ignition and fuel). This process goes back and forth between the action planner and the element selector until all preconditions (both actions and affordances) have been established.

5. Apply decomposition method. The final step in the action planner process is to actually replace the high-level task with the subtasks specified in the selected method. If these subtasks are actions (primitive tasks), the action planner also needs to establish any open preconditions for these actions. After this step the process

repeats itself for the remaining abstract tasks until all tasks have been decomposed into actions.

Element selector

The element selector starts with the selection of a setting that affords the learning goal. Subsequently, the element selector receives requests from the action planner regarding preconditions of actions. For instance, in order for the learner to recognize danger, some type of danger should be present in the environment. The element selector grants these requests by selecting environmental elements that afford the preconditions based on the affordance relations in the ontology. In the example, the element selector checks the ontology to look for an object that offers the affordance of danger, e.g. fire. The element selector determines the most appropriate object based on two characteristics:

The first characteristic taken into account by the element selector is the element's commonness in the provided scenario setting. Consider for example the action sequence of 'applying a splint'. This action requires an object that offers the affordance of 'splint', so the element selector starts searching for an object that offers this affordance and adds it to the game world. Whether the element selector decides on a table leg, a broom stick, or a small shelf may be subject to other constraints, such as its coherence with the setting, e.g. in an office a table leg may be more common, whereas in the yard a broom stick may be more appropriate.

The second characteristic concerns the element's match to the desired difficulty level. These difficulty levels are based on three aspects: the complexity of using the element, the 'visibility' of the affordance offered by the object, and the adaptability of the difficulty level (i.e. a fire can be small and controllable or a raging inferno). If the selected object in turn requires yet another affordance, the element selector iteratively fulfils these requests.

Agents are considered to be special kinds of environmental elements that can offer more complicated affordances, such as the performance of a task or action. Some elements can offer an affordance only after certain conditions are met. For instance, a match can offer the affordance of ignition, but only if it has been lighted. In such cases, an NPC needs to perform an action on the object before it can offer the affordance. Therefore, the element selector sends a request back to the action planner to create an action plan for an NPC to establish the desired precondition, in this case the action plan might be: get a match box, take out a match, and strike the match along the side of the match box. Even though the NPCs are agents and require goals as input instead of action sequences, an action sequence should still be planned in order to make sure that the environment affords at least one action plan for the agent to accomplish the goal. To afford the current example, the match box should be added to the environment to afford the action sequence even though the agent merely receives the goal 'have a lighted match'.

8.3 Evaluating a scenario creator prototype

A (proof of concept) implementation of the scenario creator was developed to enable the evaluation of the scenario creator's design. The prototype was evaluated

regarding the identified requirements. Some of the requirements could be validated by looking at the output of the prototype, i.e. the completeness of the scenario, varied scenario plans for similar learning goals, and producing the desired action sequence. Yet other requirements require domain experts to assess the quality of the output produced by the prototype: i.e. suitability for the learning task, suitability for the difficulty level, and representativeness of the scenario. To evaluate these requirements, an experiment was conducted. In this experiment, domain experts were asked to evaluate scenarios regarding their representativeness and suitability for the learning goal and the difficulty level. The scenarios they evaluated were produced by 1) the scenario creator prototype, 2) domain experts, and 3) laymen. The evaluators were uninformed about the research question. The details of the evaluation are presented below.

8.3.1 Application domain

The chosen training domain was First Aid, which has the advantage that it requires no complex world representations and has clearly defined procedures. Scenarios were restricted to burn-related incidents, which is a relatively confined set of possible situations. This decision had two benefits: the size of the knowledge base for the prototype was manageable, and laymen generally have enough knowledge about this topic to write meaningful scenarios. The restriction to 'burn-related incidents' imposes that all scenarios independent of the target task also take place within this context. For tasks such as 'treat burn' this seems straightforward, but it also applies to more general First Aid tasks such as 'ensure safety' that could be relevant in any number of situations, e.g. in an avalanche, but in this context 'ensure safety' was always related to a 'burn incident' and as such had to involve hot (or extremely cold) objects, electric hazards, etc.

8.3.2 Research questions

The research question of the evaluation study was whether the quality of automatically generated scenarios differs from scenarios written by human laymen and experts. Quality is defined as representativeness, suitability for a specific learning goal, and suitability for a specific difficulty level.

The hypothesis is that the automatically generated scenarios are of comparable or higher quality compared to the scenarios produced by laymen, but are not as good as the scenarios produced by experts. The reason for this hypothesis is that experts will possess more expertise, creativity, and knowledge to produce high-quality scenarios. Laymen, however, are expected to possess considerably less knowledge and expertise compared to the prototype, but their common-sense knowledge and creativity may outperform the prototype.

8.3.3 Method

Participants

There were two types of participants in this study: *evaluators*, who evaluated the scenarios, and *authors*, who wrote scenarios to be evaluated along with the scenarios produced by the scenario creator.

Evaluators. Five experienced First Aid instructors participated in this study as evaluators of the scenarios. They were contacted through their respective First Aid affiliations. Although all instructors were recruited from different affiliations, some turned out to be familiar with each other as a result of previous collaborations.

Human authors. The eight human authors (four experts and four laymen) were asked to write three scenarios each. It was certified that the writers and the evaluators were not familiar to each other. All laymen were university students or graduates without experience in giving or receiving First Aid training.

Materials

Below, the materials used during the study are described, i.e. the scenarios and the questionnaire.

Scenarios. A total of 36 scenarios were created - 12 scenarios for each type of author (the scenario creator prototype, experts, and laymen).

Questionnaires. Each page of the questionnaire contained a scenario description followed by three seven-point Likert-scale questions (running from -3 to 3). The learning-goal suitability and difficulty-level suitability were measured indirectly. The evaluators rated the suitability of the scenario for a beginner and for an advanced learner. The highest of the two was used to represent the learning-goal suitability. The score of the question matching the intended difficulty level was used as a measure of the difficulty-level suitability. The third question was assumed to directly measure the representativeness. A different word (believable) was used, because it was anticipated that it would better convey the meaning of the dependent variable to the evaluators.

Procedure - authoring the scenarios

Since the scenarios from all sources should be comparable, they all needed to follow a certain predefined format (see also Table 8.1). Since no such format existed at the time, a format had to be designed specifically for this study. The design of the format was based on several examples of scenario descriptions as provided by an experienced instructor. The resulting format contained all of the important aspects needed for an instructor to prepare and set up a training session: the background story (i.e. what happened); instructions for the 'victim agent', including detailed descriptions of injuries, relevant items of clothing or accessories, and information about the mental state of the victim; and a list of required objects, including objects relevant to the occurrence of the accident and objects the learner could use during the task performance.

The templates described twelve different cases for which the authors were invited to write a scenario description. As can be seen from the example in Table 8.1, the case was specified in the first row of the template. The cases varied on the following features: 1) the task to be learned (i.e. treat burn, calm victim, or ensure ABC), 2) the desired difficulty level (i.e. beginner or advanced), and 3) the setting (i.e. home, restaurant, laboratory, or park). This makes a total of twenty-four cases, whereas each source produced only twelve scenarios in total. The selected cases, consisting of a combination of these three features, are presented in Table 8.2. The twelve cases were randomly distributed among the authors within each group of human

Table 8.1: The template offered to the authors to fill out their scenario descriptions

	Task: calm victim	Level: advanced	Setting: home
Background story:			
Behavior and appearance of role player:			
Objects:			

authors. Please note that in the end each source (prototype, experts, and laymen) produced scenarios for precisely these twelve cases. To force the prototype to create a scenario for the twelve cases specified above, it was provided with additional input in the form of a pre-specified scenario constraint: the setting.

Table 8.2: The twelve selected cases for which the sources authored scenarios

	Beginner			
	Park	Home	Restaurant	Laboratory
Treat burn	X	X		
Ensure ABC	X			X
Calm victim			X	X
	Advanced			
	Park	Home	Restaurant	Laboratory
Treat burn			X	X
Ensure ABC		X	X	
Calm victim	X	X		

Since the prototype produced only textual descriptions of the scenario constraints, its output was manually rewritten following predefined translation rules so as to resemble the style of the human authors. A rule set was developed that dictated how to translate the output of the system into natural language. If, for example, a victim offered a burn at level 1, the translation was a large first degree burn with perhaps a small second degree burn. If, on the other hand, the victim offered a burn at level 2, the translation would be a large second degree burn with blisters. The only information that could not be deduced from the system's output were the motivations or feelings that drove an agent. Table 8.3 presents an example. In this example, the fact that Henk thought the job annoying and spilled the fluid because he was hasty, could not be found in the output of the system, but was added to make the agent sound more believable. In a fully functioning PEG such additional information would require the role playing agents to be capable of maintaining a believable character by providing such explanations upon request.

Table 8.3: An example of a scenario description such as the ones presented to the evaluators

	Task: calm victim Level: advanced Setting: home
Background story	Henk has to declog the sink. He thinks this is an annoying job and wants it to be over with quickly. Because of his haste he spills some of the drain cleaner fluid over his arm. Even through his shirt he feels the fluid stinging his skin and he panics.
Behavior and appearance of role player	Henk panics. He never realized a common household chemical could be this dangerous. It is difficult to calm him down. A large second degree burn with blisters forms where the drain cleaner touched his skin. The shirt Henk is wearing is sticking to the wound.
Objects	A half empty bottle of drain cleaner lays in the sink. There is a tap attached to the sink and there are bandages in a dresser.

Design of the evaluation study

The setup was a within-subjects design; all five evaluators rated all scenarios from all sources.

Procedure - evaluating the scenarios

In advance of the experiment proper and after an extensive instruction, inter-rater reliability was fostered by a joint discussion on two sets of six example scenarios and assessed by computing the intra-class correlation. During the experiment proper the evaluators were instructed to work by themselves and not to turn back to or change previous answers. After the experiment, the evaluators engaged in a group discussion. The goal of this discussion was twofold: 1) to obtain additional insights in the way the instructors evaluated the scenarios; and 2) to ask instructors about their experiences with PEGs, their views on the possibilities for training with PEGs, and their demands regarding the use and design of PEGs.

Analysis

An intra-class correlation analysis was performed to assess inter-rater reliability. A repeated measures ANOVA was used to determine the effects of the source of the scenario (expert, layman or system) upon the dependent variables. Post-hoc analyses (LSD and Bonferroni) were conducted to further examine any significant effects.

8.3.4 Results

Table 8.4 shows the means and standard deviations on the dependent measures. The mean scores followed the hypothesized trend with the experts scoring highest followed by the system followed by the laymen (also see Figure 8.5). The intra-class correlation coefficient (ICC) using the 2-way random model suggested substantial agreement ($r=0.73$; $p<.001$). Missing values (8 out of 540 values) were imputed using the SPSS expectation-maximization procedures (Little & Rubin, 1989).

A repeated measures ANOVA (see also Table 8.5) revealed significant differences between the sources for learning-goal suitability ($F(2,8)=6.70$; $p=.02$) and repre-

Table 8.4: Data exploration: mean scores (and standard errors)

	Scenario Source		
Dependent Variable	Expert	Laymen	Prototype
difficulty-level suitability	.87 (.45)	-.05 (.43)	.00 (.53)
learning-focus suitability	1.93 (.29)	.38 (.45)	.80 (.53)
representativeness	1.73 (.21)	.45 (.40)	.58 (.20)

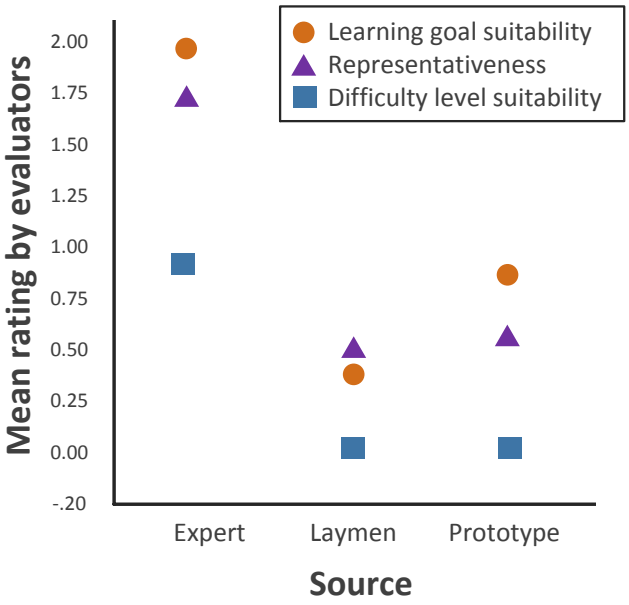

Figure 8.5: A graphical display of the means

sentativeness ($F(2,8)=6.22$; $p=.02$), but not for difficulty-level suitability ($F(2,8)=3.53$; $p=.08$). To interpret these effects, post hoc tests were conducted using the Least Significant Difference (LSD) and the Bonferroni procedures. The Bonferroni post hoc tests did not reveal any significant differences. However, the LSD post hoc test revealed the following significant differences between sources in one-on-one comparisons:

Representativeness - The experts ($M=1.73$) significantly outperformed both the prototype ($M=.58$; $p<.05$) and the laymen ($M=.45$; $p<.05$). No significant difference was found between laymen and the prototype ($p=.77$).

Suitability for the learning goal - The experts ($M=1.93$) significantly outperformed the laymen ($M=.38$; $p<.05$). No significant differences were found between the prototype ($M=.80$) and the experts ($M=1.93$; $p=.06$), nor between the prototype and the laymen ($M=.38$; $p=.36$).

Table 8.5: Results of the repeated measures analysis *) $p<0.05$

Dependent Variable	F	p-value	effect size	power
difficulty-level suitability	3.33	.08	.46	.48
learning-focus suitability	6.37*	.02	.63	.77
representativeness	6.15*	.02	.61	.74

8.3.5 Discussion

This study investigated the performance of the scenario creator prototype regarding three requirements for high-quality scenarios: representativeness, suitability for the learning goal, and suitability regarding the difficulty level. Regarding the scenarios' representativeness LSD post hoc tests revealed that the prototype performed as good as the laymen. Concerning the scenarios' suitability for the learning goal, LSD results showed that the prototype performed as well as experts and laymen both. Experts outperformed the laymen in both cases. These results confirmed our hypothesis, except for difficulty level suitability: although the trend of the means for difficulty level suitability followed the same pattern as for representativeness and learning goal suitability, the differences were not significant.

The most straightforward explanation (for not finding any significant differences regarding difficulty level suitability) is that the relatively large standard errors on this dependent variable caused a large error variation (see Table 8.4). It seems as though the consistency of the scenarios was lower with respect to the suitability for the difficulty level. This might be an indication that even for experts it can be difficult to warrant the right difficulty level at times.

Another explanation is that it was hard for the evaluators to rate the difficulty level suitability based on the information provided by the scenario format. The format did not include the unfolding of the scenario, nor the learner's desired action sequence, thereby failing to reveal the author's intentions underlying the produced scenarios. The reason for leaving this information out of the format was that laymen would not be able to produce the expected action sequence, thereby causing a problem for the comparability of the scenarios during evaluation since the differences between the scenario descriptions would have been too obvious. Yet, since this information was now lacking from the scenarios, the evaluators may have used their experience to interpret the scenario descriptions and infer the desired action sequence along with the scenario's unfolding. Interpretation and inference of the scenario's unfolding may have been easier for some of the scenarios than it was for others, which could explain the larger standard errors on difficulty level suitability.

Even though the LSD post hoc test did show significant differences, the post hoc tests with the Bonferroni correction failed to produce the same outcomes. The absence of any significant differences is strange, since it seems plausible to assume that the experts would at least outperform the laymen. It may be the case that with larger sample sizes the effects will be stronger. However, there are several additional points to be made in this discussion that might explain why *experts' scenarios* were not rated as high as they might have been if the experimental set-up had been

different.

First, the cases for which the sources had to author the scenarios were purposefully varied to test whether the prototype is capable of generating scenarios for various situations. These cases have forced the experts to write scenarios that included settings they normally would not use (e.g. the laboratory), possibly resulting in less representative scenarios.

Second, the evaluators indicated that the proportion of scenarios describing electrical and chemical burns or involving a laboratory setting was unrealistic. The evaluators seemed to look at the complete set of scenarios as if it were to be used as a curriculum. This may have been caused by the instructions in advance of the experiment: to prevent any biases regarding the different sources, evaluators were instructed to rate a set of scenarios developed to be used in a PEG. Since they thought the distribution of scenarios with atypical burns in atypical settings was off, they may have failed to rate the scenarios individually, and started evaluating them as a set, which could possibly have masked the effects of the different sources. If the instructions had been slightly different, this problem could have been prevented. For example, the instruction could have informed the evaluators about the need to practice with all types of situations, even the ones that rarely ever happen. In addition, we could have asked the evaluators a slightly different question, for instance, whether they thought the scenario was coherent instead of representative.

There are two additional disadvantages of the current set-up that may have affected the evaluation of the *scenarios produced by the prototype*. First, even though one of the strengths of the scenario creator's design is that the intended action sequence and the unfolding of the scenario is also specified, the current set-up was not suitable to explore any resulting effects of this supposedly strong feature. And second, it would have been better if all scenarios had been authored for the same case: the system would have actually needed to produce a variety of scenarios that all addressed the same input. This would have provided us with more evidence that the scenario creator is indeed able to produce a large variety of scenarios. Moreover, it would have been easier for the evaluators to compare the resulting scenarios.

8.4 Overall discussion of the component design

This chapter addressed the design and evaluation of the PEG's scenario creator component. The proposed component design integrates an HTN planner to plan the actions of the learner with a content selection mechanism based on semantic information in the ontology. The scenario creator generates a scenario plan consisting of an action sequence, NPC roles, narrative goals for the NPCs, and the required environmental elements and setting for the virtual environment.

The scenario creator not only considers the tasks that directly follow from the learning goal, but considers all tasks that the learner needs to perform during a complete exercise. The iterative task decomposition allows for fine-grained control over the actions of the player. As a result the scenario creator produces a coherent and complete set of learner actions and associated events, whereas it is also able to diversify and personalize the learner's experience. The generated action sequence can be used to guide the PEG's realtime adaptations, assessment, reflection, and

feedback provision. Furthermore, the use of an ontology allows for quick and easy additions of new information to the knowledge base. The system can directly use newly specified environmental elements in the scenario.

8.4.1 Affordance exploitation by intelligent agents

In order to facilitate the correct use of the environmental elements by the NPCs, they should have access to the affordance relations (semantic knowledge) available in the ontology. The ontology facilitates the addition of new elements to be used by all components of the PEG. As has been previously shown by Abaci & Thalmann (2005), this approach does not require individual updates of the agents' representations each time a new object is added: the agents can inspect the ontology to see what interactions the new object affords and what the preconditions for the interaction are and use the objects as intended based on that information.

This solution resembles the work by Kallmann & Thalmann (1998), who annotated environmental elements to facilitate the animation of interactions between objects and intelligent agents (NPCs) in a virtual world. As a result the complexity of object-agent interaction tasks, such as opening a door, could be reduced. Interactions between the agent and the environment can become rather complex if the agent has to do all the reasoning itself; the agent needs to perceive the environment, classify it in terms of transparency or solidity, and employ some path finding algorithm that allows it to get from A to B while moving only through transparent parts of the environment. Moreover, all agents need to be endowed with these capabilities and they will all continuously engage in this calculation process. This takes up a large amount of processing power. As a solution to this problem, Kallmann & Thalmann (1998) employed objects that were annotated with their affordances. As a result, the agents no longer needed to construct their own route. Instead, different elements in the environment provide the agents with pieces of the route and the agents simply follow those pieces depending on where they would like to go. As such, the use of semantic annotations decentralizes animation control by omitting the need for high-level planning and relying on low-level interaction behaviors. This greatly reduces the required processing power for realtime deliberations by the agents.

8.4.2 Interpretation of the scenario plan by the game engine

To establish the scenario within the virtual world, the difference in abstraction level between the multi-agent system and the game engine must be overcome. The multi-agent system works with high-level semantic concepts that allow for abstract and conceptual reasoning about the environment and possible action plans within that environment. However, most game engines employ low-level data representations for the environment and its elements. To bring about the actual scenario in the virtual environment, a translation is needed from the high-level representation employed by the multi-agent system to the low-level representation employed by the game engine.

There exist several approaches to solve this problem. The most well-known approach is probably the Pogamut platform (Gemrot et al., 2009), an extension of

Gamebots. Pogamut provides developers with a platform to create agents that control parts of the virtual environment *Unreal Tournament 2004* (UT2004). Another approach is provided by Behrens et al. (2011), who developed a more generic solution: the *environment interface standard* (EIS). The EIS facilitates the connection between an environment and BDI-agent platforms (i.e. 2APL, GOAL, Jadex, and Jason). For example, the EIS has been used by Hindriks et al. (2011) to connect the agent programming language GOAL to UT2004. An alternative solution to solve the translation problem between agent platforms and game engines is provided by CIGA (Creating Intelligent Games with Agents) (Van Oijen et al., 2012). CIGA employs domain ontologies to specify the *formal representation* of the game world content. CIGA provides the most viable solution by enabling the game engine to use the scenario plans produced by the scenario creator. For this, the ontology should be extended with formal representations of the environmental elements, actions, and settings.

8.4.3 Future refinements

Below, several ideas for future refinements of the proposed design are discussed.

Realtime plan repair

The current design of the scenario creator uses an HTN planner to plan action sequences for the learner and the NPCs *before the scenario starts*. However, once the scenario starts to play, the learner may perform actions that prevent the NPCs from performing their action plans as intended by the scenario creator. In such cases, the storyline needs to be repaired in realtime. However, in HTN planning it is not possible to repair a plan in realtime without replanning *the entire action sequence*.

A possible solution would be to allow the NPCs to communicate with the element selector. If the agent is somehow unable to accomplish its goal because of an open precondition, it could send a request to the object selector to satisfy this open precondition. The element selector could then find that object and send it to the game engine, which, in turn, could add the element to the virtual environment. Of course, this would require some additional heuristics to warrant the consistency of the simulation.

Narrative structure

The design of the scenario creator component has shown its strength in producing a variety of representative scenarios that address the learning goal. Moreover, it produces the desired action sequence for the learner and it allows for integration within the rest of the PEG. However, the design also offers room for improvement: employing a narrative structure to build suspense into the storyline.

As previously explained, the scenario creator employs an HTN planner to create a story structure for the scenario resulting in a task-oriented narrative. Although this is desirable for training purposes, one of the believed benefits of games and virtual worlds is that the learner is motivated to engage in learning by means of a narrative. However, narrative is usually not so much task-oriented, but revolves around suspension and climax.

An interesting line of research investigating the automated generation of narrative for entertainment purposes is the field of 'Interactive Narrative' or 'Automated

Storytelling'. Göbel et al. (2010) proposed the analysis of well-proven narrative structures. Particularly, they proposed to identify stages in the narrative structure that lend themselves for interactivity. They argue that during these stages delays or deviations do not cause a problem for the entertainment value of the narrative. They propose to employ such flexible stages to transmit the learning materials or lessons. As a test case, they employed this strategy with the use of the narrative structure of the Dramatic Arc (Freytag, 1905). Similarly, design proposals and implementations can be found throughout the literature that employ plot points to warrant the high-level narrative, and its accompanying entertainment value (Magerko et al., 2004; Mateas & Stern, 2003; Mott & Lester, 2006; Riedl & Stern, 2006).

Performance curve

Zook et al. (2012) designed a system that is able to predict the learner's performance during a particular event based on a skill model of the learner and complexity levels of the events. In addition, they employ a target performance curve that describes the desired trajectory of the learner's performance (performing well vs. experiencing difficulty) during the scenario. The aim of the system is to introduce events in such a way that the predicted performance curve follows the target performance curve as close as possible. Future work should investigate the possibility to extend the current design such that the desired difficulty levels described by the performance curve are used as input for the Scenario Generator's selection step regarding the decomposition methods. Such an extension would allow for more control over the overall flow of the scenario.

8.5 Concluding remarks

This chapter discussed the research on the scenario creator component. A domain-configurable design for the automatic generation of training scenarios was proposed, employing an HTN planner to determine the actions the learner is expected to perform and the events required to prompt these actions. The system then considers the environmental elements that should be present in the game world to facilitate these actions and events. To allow the system to reason about game world objects, and the interaction possibilities they add to the world, the scenario creator employs the (affordance) relations defined in the ontology.

The results of the prototype's evaluation showed the hypothesized trend: experts outperformed the prototype and the laymen regarding the scenarios' representativeness, suitability for the learning goal, and suitability for the difficulty level. However, additional evaluations are required to find conclusive answers about the effectiveness of the scenarios produced by the Scenario Generator.

Some suggestions for future research were provided: 1) in future evaluations the desired action sequence for the learner should be part of the scenario description as well as the goals to be accomplished by the role players; 2) in future evaluations it will be recommendable to stick to one case and have the sources generate twelve different scenarios for that single case; 3) evaluators can be biased in unforeseen ways and clear instructions should be provided to reduce the possibility for such biases to a minimum; and 4) it is recommendable to ask evaluators to rate the

scenario's coherence instead of its representativeness.

To conclude, the scenario creator design is an important step towards the automated generation of effective and personalized scenarios to offer learners access to a large variety of quality training scenarios.

CHAPTER 9

The Authoring Tool:

Scenario authoring for instructors

Abstract - This chapter investigates the *authoring tool*, which enables instructors to specify the following properties of a scenario: (1) the learning goal, (2) the setting, (3) the non-player characters (NPCs) in the scenario, and (4) the narrative goals and attitudes assigned to those NPCs. The properties specified by the instructor are used by the scenario creator as it automatically generates a new scenario.

In order to come to a design for the authoring tool, first, a paper-based mock-up prototype of the authoring tool was investigated through qualitative evaluation sessions with First Aid instructors. Subsequently, based on the outcomes of these sessions, the authoring tool was extended with a *supportive feature* for the instructors: the authoring tool uses the selected learning goals to provide suggestions for the NPCs and their goals and attitudes. A new, interactive, prototype of the authoring tool was developed and evaluated in an experiment. Instructors were asked to author two scenarios. One scenario was authored using a version of the authoring tool *with* suggestions, the other scenario was authored using a version of the authoring tool *without* suggestions. Results showed that the added suggestions significantly enhanced the quality of the resulting scenarios and the usability of the authoring tool.

The combination of the authoring tool, the scenario creator, and the ontology offers instructors multiple ways to influence the training content without the need to manually create each and every scenario. The PEG architecture promotes the collaboration between the PEG and the instructor by optimally benefiting from their respective strengths: the PEG automatically warrants the scenario's internal consistency, whereas the instructor warrants the scenario's instructional quality. In this way, the instructor and authoring tool collaboratively define high-quality, consistent, and coherent scenarios.

This chapter is based on research conducted in collaboration with P. Hovers, also documented in:

Hovers, P., "Scenario Authoring of Serious Games - An Interface for a First Aid Game", MA thesis, University of Amsterdam, 2013.

> "A machine is characterized by sustained, autonomous action. It is set up by human hands and then is more or less set loose from human control. It is designed to come between man and nature, to affect the natural world without requiring or indeed allowing humans to come into contact with it. Such is the clock, which abstracts the measurement of time from the sun and the stars: such is the steam engine, which turns coal into power to move ships or pump water without the intervention of human muscles. A tool, unlike a machine, is not self-sufficient or autonomous in action. It requires the skill of a craftsman and, when handled with skill, permits him to reshape the world in his way."
>
> **Bolter (1984)**

Instructors possess ample expertise in their task domain and didactics, however, their expertise is often consolidated in tacit knowledge, i.e. knowledge that is difficult to vocalize or observe (Dreyfus & Dreyfus, 2005; Eraut, 2000). This poses challenges for the system's designers who need to obtain and employ that expertise in order to create a system design (e.g. a PEG) that is able to automatically generate scenarios that are as good as scenarios authored by experts. The current design aims to elicit and capture expert knowledge by representing it in an ontology. The ontology allows instructors to add and alter the contents of the ontology to their liking. Yet there are still two problems with this approach. The first problem is the tacit knowledge. Instructors will not be able to add knowledge to the ontology that they are not aware of. The second problem is that even though the addition of new content to the ontology does make that content available for selection and may improve the overall quality of the scenarios, it does not guarantee the actual selection or use of the new content in, for instance, the next scenario. As a result, the only way for an instructor to influence the contents of a single specific scenario would be for that instructor to manually author a scenario plan by taking on the role of the scenario creator in the organization.

Up until now, the functional components in the proposed PEG architecture can be performed either fully automatic by an intelligent agent, or fully manual by a human operator. The current chapter investigates whether there can be a combined approach where instructors are able to influence the automated scenario creation process with the use of an authoring tool by providing scenario constraints beforehand. As such, the automated components and the instructor collaborate on the task of creating a suitable scenario for the learner. This allows instructors to specify a scenario at a high level, thereby relieving the instructor from manual scenario creation. At the same time, the scenario creation process is guided by the expertise of the instructor, thereby improving the quality of the scenarios.

In Chapter 4, pages 53-55, the authoring tool was specified by requirements R18 and R19:

R18 - The instructor must be able to understand the system's processes
R19 - The instructor must be able to control the system's processes

However, to determine how the authoring tool can facilitate collaboration between the instructor and the automated components, more knowledge is needed about what type of influence instructors would like to have on the automated scenario creation process.

9.1 Task domain & support analysis

Throughout this section requirement R18 and R19 and its claims will be refined. This is done in three consecutive steps: the first step explores the features human instructors would like to control when authoring scenarios for PEGs. Since the envisioned authoring tool allows instructors to collaborate with the PEG, the second step involves an overview of relevant literature about cognitive systems and dynamic task allocation. The third step concerns the investigation of technological opportunities to support the joint instructor-PEG performance by means of a design feature for the authoring tool, i.e. critiquing and suggestions.

9.1.1 Task domain - what instructors wish to control

In order to identify the demands that human instructors would place on the authoring tool, we conducted a series of semi-structured interviews with four First Aid instructors. All instructors were recruited through large First Aid certifiers in the Netherlands (Red Cross, Het Oranje Kruis, and Stichting Landelijk Protocol Eerstehulp Verlening).

Each interview started by presenting the instructor with five descriptions of a learner's engagement in a training scenario. For example, the first case describes a learner attempting to treat a nosebleed, while making several mistakes in the procedure. Each case was discussed with the instructor by asking them how they would respond. After the five cases, the instructors were asked general questions about their background and experience as an instructor. Then two videos showcasing the concept of serious games for First Aid were presented. Instructors were asked whether they would be interested in using a serious game in their First Aid course and, if so, how they would use it. Of special interest was the type of instructor-controllable features they would like to see in a PEG, e.g., decide upon the learning goals, select a scenario setting, create NPCs, determine the sequence of events, etc. A complete description of the interview guide can be found in Appendix C.

Results of the interviews

The conducted interviews led to the following results:

Learning goal - The instructors indicated that the learning goal is especially relevant to specify a scenario: it describes what the learner is supposed to learn and do. The situation offered by the scenario should provoke the learner to perform actions that are related to the learning goal. In First Aid, the situation is usually some injury or illness that befalls a victim. Therefore, the instructors indicate that they would like to be able to select a learning goal and/or a 'case' (i.e. a specific type of injury or sickness) that is used by the PEG to create a scenario.

Setting - Instructors also emphasize that when delivering First Aid, the nature of the situation needs to be taken into account, e.g. is the situation safe enough to offer help? Instructors give two reasons as to why selecting the setting is important: 1) the setting can be used to personalize the scenario, e.g. by selecting a setting that is relevant to the learner and 2) the setting can be used to manipulate the complexity of the emergency situation (e.g. a heart attack in the living room vs a heart attack in the car while driving on the highway).

NPCs - Furthermore, instructors point out that an important aspect of First Aid is to cooperate and delegate, e.g. First-Aiders need to instruct bystanders to go and get help and to return afterwards. As a result, the behavior and state of mind of the key role players (also called 'non-player characters') in the scenario are an important aspect of the scenario as well. This is not only applicable to the victim-role, but also to, for example, the bystander-role. For instance, one instructor expresses the desire to add an upset family member to the scenario. Another instructor explains that NPCs should also be emotionally responsive to the actions of the learner, for instance, if the learner completely ignores a bystander (e.g. a family member), that NPC might become angry or upset. From these statements, we conclude that instructors would like to be able to manipulate the (behavior of) the NPCs in the scenario as well.

To conclude, the results obtained during the interviews provided insights regarding the features of the scenario that instructors would like to have control over, i.e. the learning goal, the setting, and the NPCs.

9.1.2 Human factors - facilitating human-machine collaboration

The previous subsection presented the results from interviews with instructors regarding the scenario features they would like to have control over. Offering instructors the means to influence the PEG's scenario creation process, implies that the PEG and the instructor will be involved in something called 'collaborative problem-solving' (Fischer et al., 1991). The current subsection investigates the human factors literature regarding human-machine collaboration and how to support this.

The first point to keep in mind when designing a cognitive system is that overly simple tools can cause problems because they require too much skill from the user, whereas overly intelligent tools can cause problems if they fail to give any indication to the user of how they operate and what they are doing. In the former case, the tool does not offer the user a true solution, whereas in the latter case the user may feel like a bystander who is merely watching while unexplained operations take place (Norman, 1986).

Another important point to reckon with is that the instructors may misjudge the PEG's capabilities leading to excessive trust or mistrust (Woods & Roth, 1988). In such cases the system might get underutilized or ignored when it could provide effective assistance, or the practitioner might defer to the machine even in areas that challenge or exceed the machine's range of competency. The supervisor must have real as well as titular authority; machine problem solvers can be designed and introduced in a way that the human retains the responsibility for the outcomes without any effective authority. The supervisor must be able to redirect the lower-order machine cognitive system. There is a need for a common or shared representation of the state of the world and of the state of the problem solving process.

Preferably, the design for the authoring tool anticipates these possible problems. One way to do this is by offering a way to allocate responsibilities and tasks based on the strengths and weaknesses of both the instructor and the PEG. For instance, humans are better at common-sense reasoning, define the goal, and decompose the problem, whereas machines provide external memory, ensure consistency, hide irrelevant information, and visualize aggregated information. The PEG aids the instructor. In other words, the embedded AI (e.g. NPCs, scenario creator) is not

intended to be a replacement for the instructor, but rather a tool that can be used by the instructor to take over certain parts of the task when the instructor is unable to do so. Hollnagel & Woods (1983) emphasize that the optimal task distribution in a cognitive system can vary from situation to situation and that, as a result, the design of the system should allow for dynamic task allocation.

The goal of the authoring component is therefore to support adjustable work division between machine and human based on the wishes of the instructor. In some cases the instructor may be absent or the instructor may have other reasons to leave the scenario creation process to the machine (i.e. the learner model and the scenario creator). In other cases the human operator (instructor) may wish to manage the scenario creation process him- or herself and provide the system with additional constraints to be employed in the scenario creation process. And then again in other cases, the instructor may wish to create a scenario plan completely by hand. As such, the PEG architecture should support all of these situations.

9.1.3 Envisioned technology - suggestions and critiquing

The previous subsections presented the scenario features instructors would like to author, and principles from human factors literature to improve the usability of the authoring tool. The current subsection presents the envisioned technological solution to improve the PEG's usability: critiquing.

One way for a system to support a human in performing a task is by means of critiquing. Critiquing systems point out the errors and suboptimal conditions that might remain undetected otherwise (Fischer et al., 1991). Moreover, advice is provided on how to improve the situation along with an explanation for the suggestion. Based on a rule set, the system presents (more suitable) alternatives for the actions performed and decisions made by the human operator. The provided alternatives are presented along with arguments and reasons as to why they are more suitable. As a result, the machine helps the user to avoid problems and work with multiple views or opinions about the optimal course of action.

It is not always the system who critiques the user. In some cases, the critique comes from the user and is directed at the suggestions provided by the system (Burke et al., 1997). This version of a critiquing system is usually employed in recommender systems, where the system tries to identify the user's preferences by offering the user several options (suggestions). The user is then required to pick one of the options or critique the suggestions (McCarthy et al., 2005; Viappiani et al., 2007).

Suggestions can also help to speed up processes. For instance in text messaging, smart phones offer word suggestions and Google finishes your search phrase while typing. Suggestions can also teach the user how to use the system since they provide the user an example of how the user authoring tool can be employed. An example of a system employing suggestions to support the user during task performance are the 3D sketching tools proposed by Igarashi & Hughes (2001) and Tsang et al. (2004). Their systems provide ongoing suggestions of possible geometric shapes that could be added to the drawing, thereby relieving the user from the need to manually draw the shape.

9.2 Authoring tool - specification

The authoring tool is a functional component that enables instructors to influence the automated scenario creation process (also see Figure 9.1).

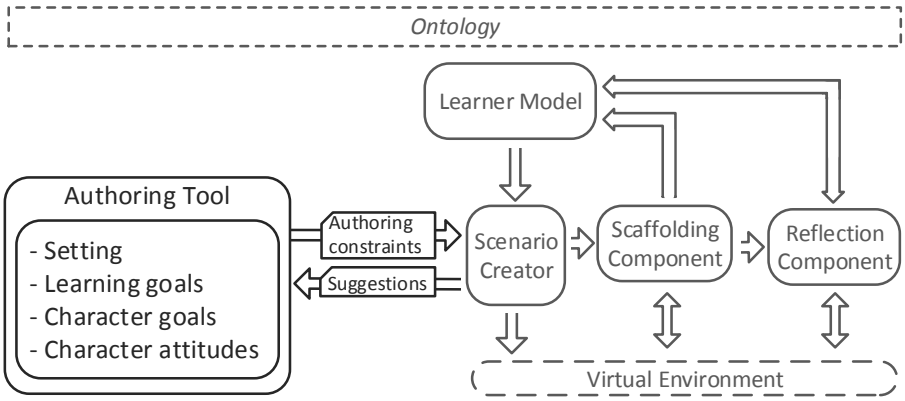

Figure 9.1: The authoring tool; as part of the functional architecture

Based on the results of the investigation described in the previous section, the requirements for the authoring tool were refined as follows:

R19 - The instructor must be able to control the system's processes
- R19a - The instructor must be able to provide the following scenario constraints to the PEG's scenario creator:
  - the learning goal
  - the setting of the scenario
  - the NPC roles, goals, and attitudes
- *PC19a.1* - this offers the instructor the possibility to control the most important parts of the scenario creation process
- *NC19a.1* - meeting R19 may conflict with R18 (instructor understanding)

An example of a narrative goal for an NPC is to "break a limb". As such, this goal will result in a situation that requires the learner to deal with the diagnosis and treatment task of a fractured limb. There are also narrative goals that influence the level of support or challenge in the scenario. For instance, the goal "support the learner in treating the victim" can be provided to an NPC to aid the learner during the task. Defining an NPC's attitude offers the instructor the possibility to create practice opportunities concerning cooperation, delegation, or communication. For instance, creating an NPC with an attitude "in distress" will require the learner to reassure that NPC.

9.2.1 Exploration of possible designs - paper prototype

To evaluate a design based on these requirements, an evaluation cycle was conducted. This evaluation employed a paper (mock-up) prototype. The paper prototype consisted of separate papers and sticky notes that could be rearranged into various designs.

Instructors were interviewed in one-on-one sessions. The paper prototype implemented requirement R19a: instructors were able to specify the learning goal, the task to be performed, the setting, the NPC roles, and the narrative goals and attitudes for the NPCs.

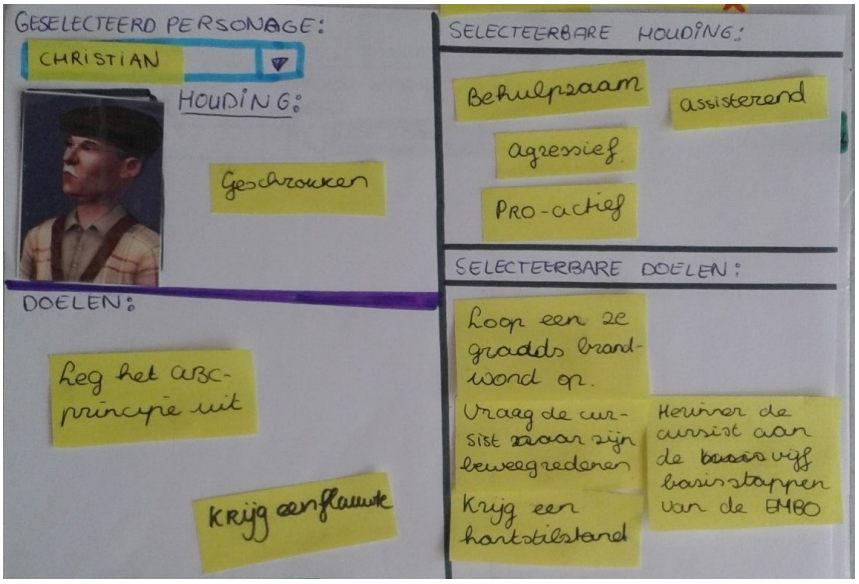

Figure 9.2: Parts of the authoring tool's paper prototype - selecting NPC traits

The controllable features (NPCs, learning goal, setting) were treated as separate tasks for the instructor to perform. There were different pieces of paper for each of these tasks. Each piece of paper represented a frame that allowed the instructor to specify that feature. For instance, there was a piece of paper that represented a frame which allowed the instructor to specify an NPC along with its narrative goals and attitudes (see Figure 9.2). A different piece of paper depicted a frame in which the instructor could specify the learning goal, and so on. The use of the separate papers and removable sticky notes allowed the instructors to rearrange the look and feel of the prototype. Instructors were asked to provide additional comments on the lay-out of the interface and the frames themselves. For instance, they were asked whether they would want certain frames to be combined or broken up into separate frames. They were also asked in what order they would like the frames to appear, if certain frames should be available simultaneously in one screen, and if so how they would like these frames to be arranged on screen.

Results from the paper-based prototype evaluation cycle

The paper prototype sessions revealed that instructors were satisfied with the features they were able to manipulate, i.e. learning goal, NPC roles, and narrative goals and attitudes for the NPCs (PC19a.1).

In addition, the interviews showed that instructors have a preference regarding the order of the feature-frames: they would like to start with the specification of

the setting, thereafter they would like to specify the learning goal, and lastly, they would like to add the NPC roles, goals, and attitudes. After specifying each of the features in separate screen steps, they would like to be presented with an overview of their chosen constraints. In addition, they would like to be able to consult the learner model during the scenario authoring process.

Instructors also reported that they found it difficult to maintain consistency between the NPC specifications (roles, goals, and attitudes) on the one hand and the selected learning goal and task for the learner on the other (NC19a.1).

Implications for the design of the authoring tool

First of all, the lay-out of the authoring tool's interface was adjusted to comply with the findings obtained in the paper-based prototype evaluation cycle.

Secondly, to aid instructors in maintaining consistency between the various features, suggestions were added to the authoring tool: based on the learning goal selected by the instructor, the authoring tool offers a suggestion for the NPCs, narrative goals, and attitudes.

Instructors are not required to accept the suggestions; they are free to alter the suggested NPCs and to alter the NPCs' goals and attitudes. As such, the extended authoring tool still allows the instructors to specify scenario constraints that suit their personal taste and expertise.

Based on the findings of the paper prototype evaluation cycle requirement R18 was refined as follows:

R18 - The instructor must be able to understand the system's processes
- R18a - The authoring tool should provide appropriate support during the authoring process in the form of suggestions for the NPCs and their features (i.e. goals and attitudes)
  - *PC18a.1* - This improves the authoring tool's usability (easier)
  - *PC18a.2* - This improves the authoring tool's efficiency (faster)
  - *PC18a.3* - This improves the authoring tool's effectiveness (better)
  - *NC18a.1* - This may reduce the instructors' activity level and involvement

9.2.2 Authoring tool - design

The new design for the authoring tool's interface was implemented in a prototype using HTML and JavaScript (see Figures 9.3 and 9.4). However, in order to investigate the effect of the added suggestions, there were two versions of this prototype. The difference between these two versions is whether or not the instructor receives suggestions for the authoring of the NPC roles, goals, and attitudes.

Both versions of the authoring tool lead the user through four main authoring steps. The four steps are described in more detail below:

Step 1. Start - The first step invites the instructor to either create a new set of scenario constraints and name it, or to load an existing set of scenario constraints. Then the instructor is asked if the scenario constraints are meant to be personalized for one particular learner. If so, the learner's personal profile becomes accessible in a collapsible window for the rest of the authoring session (lower right corner). Lastly, the instructor is asked to provide the scenario's setting (see Figure 9.3).

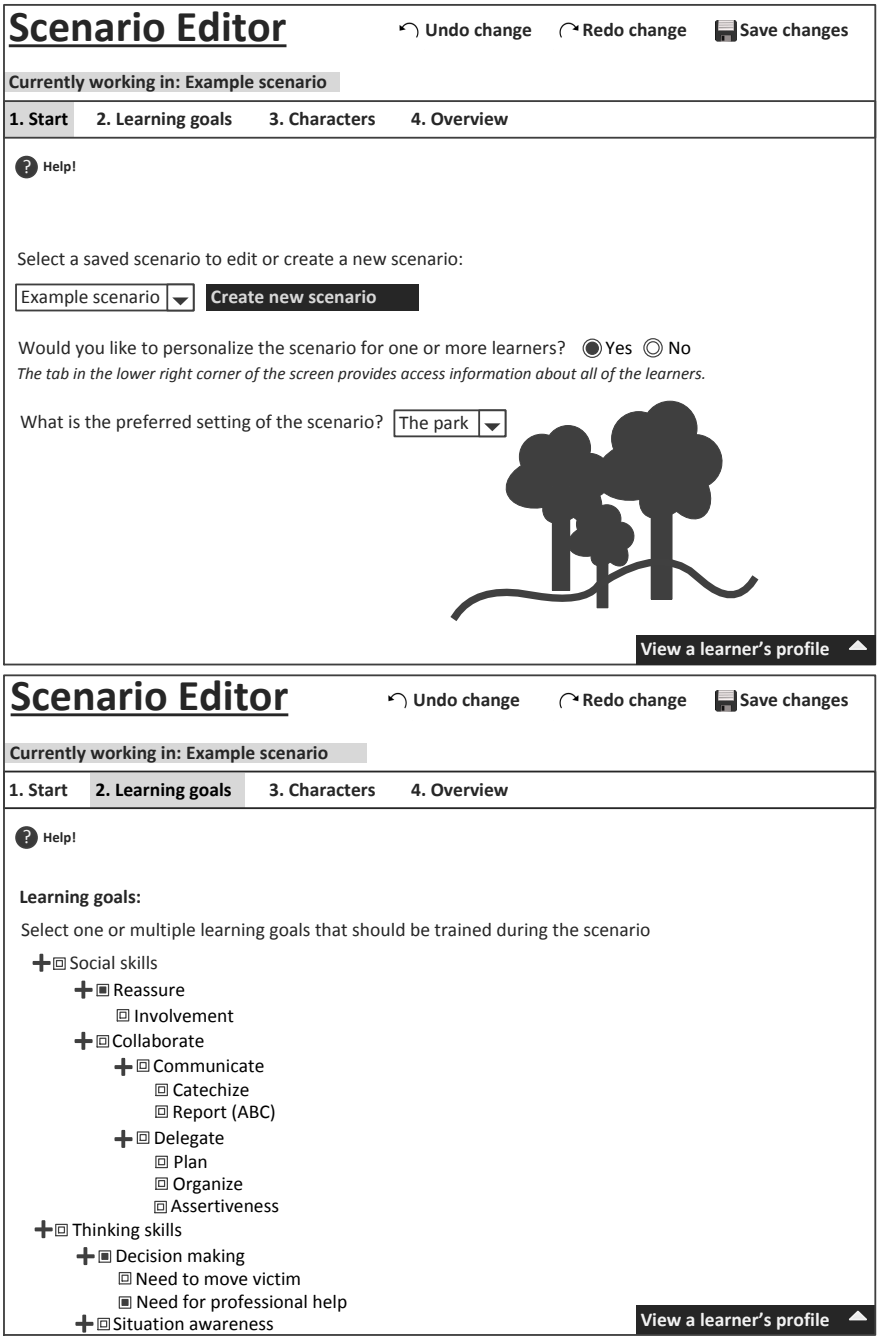

Figure 9.3: The authoring tool's interactive prototype (printscreen resolution was too low)

Figure 9.4: The authoring tool's interactive prototype (printscreen resolution was too low)

Step 2. Learning goals - The second step invites the instructor to select one or more learning goals from a tree structure (see Figure 9.3).
Step 3. NPCs - The third step shows the difference between the two versions.
V1. In the first version, the system provides the instructor with suggestions regarding the NPCs in the scenario along with their goals and attitudes by placing them in the 'selected options' field. The instructor is able to add new NPCs and to alter the suggested NPCs and their goals and attitudes (see Figure 9.4).
V2. In the second version, the system does not provide the instructor with any suggestions. The 'selected options' field is cleared out. The instructor is required to drag NPC roles, goals, and attitudes from the 'options' fields.
Step 4. Overview - The instructor receives an overview of the chosen scenario constraints to either accept or modify (see Figure 9.4).

9.3 Evaluation study

A study was conducted to investigate whether the suggestions (regarding the virtual NPCs and their features) would improve the use of the authoring tool regarding: the ease of use of the authoring tool (PC17b.1, easier), the time taken for the authoring process (PC17b.2, faster), and the quality of the authored scenario constraints (PC17b.3, better). This study compared the two versions of the prototype, one version with suggestions and one without.

9.3.1 Hypothesis

By providing suggestions, an instructor is not required to author each NPC from scratch, but receives suggestions for the NPCs based on the selected learning goals. As a result, the suggestive authoring tool is expected to ease and speed up the authoring process. Furthermore, the suggestive authoring tool is expected to result in higher quality scenario constraints and higher levels of user satisfaction with the resulting scenario constraints.

9.3.2 Method

Participants

A total of forty-five participants were recruited through their First Aid associations, such as Rode Kruis, through email invitations. In addition, one First Aid instruction expert participated as a reviewer of the resulting scenario constraints.

Materials

The following materials were used during this study:

Cases. Two case descriptions were developed for the participants to guide the authoring process during the evaluation experiment. Each case consisted of a description of a particular learner and two appropriate learning objectives for the learner to work on. The case descriptions are provided below:

Case 1 - Create a new set of scenario constraints and call it 'ScenarioA'. These scenario constraints are meant for learner 'Ruben'. Ruben has only just started his First Aid course. However, he has already finished his theory exam and he also participated twice in role-playing exercises with medical actors. He will now start practicing with the virtual First Aid training game. Your task is to author a set of scenario constraints for Ruben that allows him to practice how to reassure a victim. In addition, you would like for him to practice with the recognition of a first-degree burn. The location of the scenario will be the park. Make sure that you add two virtual NPCs to the scenario constraints that are suitable for these learning goals. Please keep in mind that the resulting scenario constraints should not be too difficult nor should it be too easy for Ruben.

Table 9.1: The two cases used in the experiment

Overview of the provided information for Case 1	
Learner	Ruben
Scenario name	ScenarioA
Location/Setting	the park
Learning objective	Reassure & First-degree burn
Required no. of NPCs	2
Overview of the provided information for Case 2	
Learner	Christian
Scenario name	ScenarioB
Location/Setting	the restaurant
Learning objective	Perform ABC procedure & Report status using ABC
Required no. of NPCs	4

Case 2 - Create a new set of scenario constraints and call it 'ScenarioB'. This set of constraints is meant for learner 'Christian'. Christian has recently enrolled in the First Aid course. He has already finished his theory exam and he has already participated in virtual training exercises before. The intended learning goals for Christian are 'performing the ABC procedure' and 'reporting the victim's status according to the ABC'. The location of the scenario will be the restaurant. Make sure that you add four NPCs to the scenario constraints that are suitable for these learning goals. Please keep in mind that the resulting set of constraints should not be too difficult nor should it be too easy for Christian.

Prototypes. Five different versions of the prototype were developed:

1. A non-suggestive version of a neutral case (for the learner to explore in advance of the experiment)
2. A suggestive version for case 1
3. A non-suggestive version for case 1
4. A suggestive version for case 2
5. A non-suggestive version for case 2

Questionnaire for the scenario authors. The usability of the authoring tool was rated using an online 7-point Likert scale survey consisting of the SUS and three

additional statements about the instructor's satisfaction with the resulting scenario constraints:

1. I am satisfied with my authored scenario
2. My authored scenario constraints are well-suited for the learner
3. My selected NPC roles, goals, and attitudes are well-suited for the learning goals

The 7-point Likert scale ranged from 'I completely disagree' to 'I completely agree'.

Questionnaire for the evaluator. The quality of the scenario constraints was rated using a 7-point Likert scale survey consisting of the following 4 statements:

1. This is a realistic set of scenario constraints
2. This set of scenario constraints is suitable for a beginner
3. The selected learning goals reflect the learning goals mentioned in the case
4. The selected NPC roles, goals, and attitudes match the learning goals

The 7-point Likert scale ranged from 'I completely disagree' to 'I completely agree'.

Suggestions. The suggestions for the NPC roles, goals, and attitudes were based on validated scenario plans from the research by Ferdinandus et al. (2013) presented in Chapter 8. In that study, the selected scenario plans (i.e. 'Reassure and first-degree burn at the park' and 'perform and report ABC at the restaurant') were both rated as highly representative, highly suitable for the learning goal, and highly suitable for a beginner: they received either a 6 or a 7 out of 7 for these features from all five evaluators in the study by Ferdinandus et al. (2013). So, the suggestions offered in the suggestive authoring tool concerned the NPCs and their features as described in these two validated scenario plans.

Experimental design

The following experimental design was employed.

Independent variable. The independent variable of the evaluation was whether the instructor received suggestions regarding the NPC roles, goals, and attitudes in Step 3. In half of the cases the instructor received suggestions regarding the NPC roles, goals, and attitudes based on the learning goals the instructor selected in Step 2 (suggestive). In the other half of the cases the instructor was required to select the NPC roles, goals, and attitudes without receiving any suggestions (non-suggestive).

Dependent variables. The four dependent variables were: 1) the usability of the authoring tool, 2) the time taken to create the scenario constraints, 3) the user satisfaction regarding the resulting scenario constraints, and 4) the quality of the resulting scenario constraints.

Usability. The first dependent variable, usability of the authoring tool, was measured by means of the 'System Usability Scale' (SUS), a validated questionnaire for assessing the overall usability of a system (Bangor et al., 2008). The word "system" in the SUS statements was replaced by "the scenario editor for First Aid".

Efficiency. The efficiency of the authoring tool was measured by the time taken to author a set of scenario constraints.

User satisfaction. This was measured by taking the mean of the three items regarding user satisfaction in the online survey.

Quality. The quality of the scenario constraints was determined by an independent instructional First Aid expert who rated the four questions previously mentioned in

the materials section. The mean of these four items represented the overall quality of the scenario constraints. The expert who rated the quality of the scenario constraints was uninformed about the research questions.

Within-subjects comparisons. Effects of suggestions were tested within-subjects. Each participant performed two versions of the same task: authoring a set of scenario constraints for a particular case using the authoring tool. Descriptions of the two cases used in the study are provided further down. One of the cases was authored using the non-suggestive authoring tool; the other case was authored using the suggestive authoring tool. A counterbalanced design was used to rule out order effects for the cases and authoring-tool versions (see Table 9.2).

Table 9.2: Experimental design

Group	First task		Second task		Number of
	Authoring tool	Case	Authoring tool	Case	participants (n)
1	Non-suggestive	Case1	Suggestive	Case2	12
2	Suggestive	Case1	Non-suggestive	Case2	10
3	Non-suggestive	Case2	Suggestive	Case1	9
4	Suggestive	Case2	Non-suggestive	Case1	14

Procedure

The study was conducted by means of an online survey. Through their First Aid associations, instructors received an email asking them whether they wanted to participate in the study. By clicking on the link, the participant was led to a webpage that automatically placed the participant in either one of the four versions of the online survey. This forward function looped through the four different experiment versions to distribute the participants evenly among the four groups. The survey was structured as outlined below:

1. The general aim of the research was explained, along with the structure of the survey and its duration.
2. The participant was presented with a video about the concept of serious gaming.
3. A video explained the intended use of the authoring tool.
4. The participant was free to explore a basic version of the scenario authoring tool with an example case.
5. The experiment proper initiated when the first case was presented to the participant. After reading the case, the participant was instructed to author a set of scenario constraints by following a link to the according version of the authoring tool. After finalizing the scenario constraints, the resulting scenario constraints and the time taken were saved on a server. The participant was asked to close the prototype and return to the survey.
6. Subsequently, the participant received a set of questions regarding the satisfaction about the final authored scenario constraints, directly followed by the SUS.
7. Steps 5-6 were repeated for the second case.

8. The survey finished and the participant was thanked for participating.

After collecting all the authored scenario constraints, the scenario constraints were anonymized and randomized. The sets of scenario constraints were organized in a booklet, each page describing the results of a single authoring process and the case for which it was authored. The (objective) reviewer was presented with this booklet and asked to evaluate the authored scenario constraints by rating each of the sets using the four statements described in the 'dependent variables' section.

Analysis

To investigate whether the addition of suggestions to the interface resulted in the intended effects (increased usability, efficiency, user satisfaction, and quality), the results obtained from the survey were analyzed as described below.

Internal consistency. The internal consistency of the questions measuring user satisfaction and the items for measuring the quality of the scenario constraints were assessed using Cronbach's alpha.

Effects of suggestions. Due to logistical problems, the data obtained from the survey and the data obtained from the prototypes were saved in separate data bases and the survey responses (the SUS and the user satisfaction) of a particular participant could not be matched to that same person's authored scenario constraints and the time taken to author them. As a result, the time taken to author a set of scenario constraints using the suggestive versus the non-suggestive authoring tool and the expert's ratings regarding the quality of the resulting scenario constraints were compared using an independent 1-tailed t-test.

9.3.3 Results

Usability of the authoring tool

The SUS scores for the suggestive authoring tool (M=56.7, SE=2.21) were significantly higher than those of the non-suggestive authoring tool (M=53.5, SE=2.19), ($t(44)=2.536$; $p<.03$).

The time it took for users to finalize their constraints was significantly shorter, by almost a minute on average, when the suggestive authoring tool was used ($t(80)=2.072$, $p<.025$). The mean time to finalize the authoring process for the suggestive authoring tool was 3'22" minutes (SE=18"), and for the non-suggestive this was 4'13" minutes (SE=15").

Table 9.3: The means and standard errors (SE) of the two conditions for the dependent variables

Condition	SUS score	Time taken	User satisfaction	Rated quality
with suggestions	56.7 [a] (2.21)	3'.22" [a] (18")	5.02 (.19)	4.25 [a] (.36)
without suggestions	53.5 [a] (2.19)	4'.13" [a] (15")	4.86 (.16)	3.19 [a] (.33)

[a]) $p<.05$, one-tailed

User satisfaction regarding final scenarios

The statements covering the user satisfaction with final set of scenario constraints revealed a Cronbach's alpha of .799, indicating good consistency among the items (Cortina, 1993; Tavakol & Dennick, 2011). There was no significant difference between the user satisfaction regarding the final scenario constraints obtained with the suggestive ($M=5.02$, $SE=.19$) and non-suggestive authoring tool ($M=4.86$, $SE=.16$) respectively ($t(44) =.746$, $p>.05$).

The quality of resulting scenarios

The Cronbach's alpha for the statements covering this construct was .942, indicating excellent consistency among the items (Cortina, 1993; Tavakol & Dennick, 2011). The suggestive authoring tool resulted in a significantly higher mean for the assessed quality of authored scenario constraints ($M=4.25$, $SE=.36$) than for the non-suggestive authoring tool ($M=3.19$, $SE=.33$) ($t(85)=2.20$, $p<0.05$).

Additional qualitative analysis of the results

A qualitative analysis of the resulting sets of scenario constraints showed that instructors employing the non-suggestive authoring tool tend to create a more complex set of scenario constraints (e.g. multiple NPCs who are angry or upset) compared to the sets of scenario constraints developed with suggestions (e.g. one NPC is upset, the other one is supportive).

9.3.4 Discussion

The suggestions for NPC roles, goals, and attitudes based on the selected learning goals, improved the usability of the authoring tool. The usability score of the suggestive authoring tool relates to an "okay" usability on a verbal scale (Bangor et al., 2009).

Providing instructors with suggestions for the NPC roles, goals, and attitudes enhanced the quality of the resulting sets of scenario constraints. Furthermore, it appeared to reduce the complexity of the scenario constraints.

The satisfaction of users with the authored scenario constraints was not significantly affected by the suggestions. A possible explanation for this finding might be that, regardless of the availability of suggestions, instructors take great care in authoring their intended scenario (as they pointed out during interviews) and will not finish the authoring process before they are satisfied with the result. This explanation is supported by the fact that, on average, the instructors' authoring processes took significantly longer with the non-suggestive authoring tool, compared to their authoring processes using the suggestive authoring tool. The mean of approximately 5 for the user satisfaction in both versions of the authoring tool indicates that users were generally satisfied with the resulting scenarios in either of the conditions.

9.4 Overall discussion of the component design

Throughout the entire design process of the authoring tool, different groups of instructors were closely involved in the design process, providing us with feedback and input at each step of the way. Semi-structured interviews resulted in the selection

of the following control features to manipulate the scenario creation process: the learning goal(s), the scenario setting, and the NPC roles, goals, and attitudes. The paper-prototype evaluation led to the idea of adding suggestions to the authoring tool. The experimental study showed that the suggestions proved to be helpful, not only increasing the authoring tool's usability (PC18a.1) and efficiency (PC18a.2), but also the quality of the resulting sets of scenario constraints (PC18a.3).

Instructors reported that they found the system's suggestions helpful in authoring the NPC roles, goals, and attitudes. However, they found it difficult to predict what behavior the resulting constraints would produce, and how this would affect the scenario. And they are right: It is difficult to know what the storyline will be in advance, because there are still various possibilities for the story to unfold. The NPCs have a certain degree of freedom to decide how they are going to achieve the narrative goals specified in the scenario constraints and scenario plan. Goal-based planning, in the end, requires instructors to trust the PEG's automated components (scenario creator, NPCs) in that they will truly produce behavior that complies with the constraints provided by the instructor.

In theory, the scenario creator is able to handle the constraints provided by the instructors (setting, learning goal, and NPC roles, goals, and attitudes). At this point we anticipate a fairly seamless integration, but further investigation of an implementation is required to verify this expectation.

Further research should investigate whether the scenario plans created with the help of the instructor (through the use of the authoring tool) are truly of a higher quality than the ones produced by the scenario creator alone. In addition, the scenario plans produced through the collaborative efforts of the instructor and the scenario creator (i.e. with the use of the authoring tool) should be compared to scenario plans that were manually produced by instructors.

9.4.1 Future research on the design of the authoring tool

In the following, we provide several directions for future research.

Refinements of the current design

A possible improvement would be to offer instructors the choice whether they would like to receive suggestions or not. In addition, a feature that motivates instructors to actively process and check the suggestions before accepting them, is believed to be beneficial for the joint performance of the instructor and the PEG. The reason for this anticipated effect is that the results obtained from the study showed that instructors tend to leave the suggestions provided by the PEG fairly intact. In those cases where they did alter the suggestions, these adjustments generally involved only minor changes. A possible danger of this phenomenon is that instructors may rely upon the PEG, even when the PEG is failing. Therefore, future research would be the investigation of an additional feature to prevent this from happening. This would entail the specification of specific types of situations in which the PEG may fail to perform adequately. The PEG can use this information to, for instance, alert the instructor and ask for advice.

De Greef et al. (2010) propose the employment of adaptive automation to support a dynamic division of work between the human and the machine. They investigate

a system that is capable of reassigning work to itself to reduce the workload of the human operator, but only in cases where the workload of the human proves to be too high. In cases where the workload is low, the system will allocate more and more tasks to the human operator in order to prevent cognitive underload, i.e. a situation where the cognitive activity level of the human operator becomes too low to function optimally. Such an approach might be interesting in cases where an instructor is managing team training processes. It is expected that in such cases an instructor is not capable of authoring all of the storylines in the game world for each of the learners. A possible solution would be to design the system such that it is able to determine which storylines need the attention of the instructor, while taking care of the remaining storylines automatically.

Extensions of the current design

A first extension would be to also allow the PEG to critique the instructor's authoring process. For instance, the system might alert the instructor if an NPC with conflicting goals is added or if too many learning goals are selected.

Another extension to investigate is how instructors can be allowed to play and/or preview the resulting scenario to see what the learner will actually experience when playing the authored scenario.

Third, the current authoring tool is designed to influence the scenario creation process. However, instructors also expressed their desires to influence the scenario development in realtime. This would require further investigations of the possibility to, for instance, offer instructors: control over the realtime adjustment of the level of scaffolding; the opportunity to play as an NPC; and the initiation of a reflection session combined with scenario-playback possibilities.

9.4.2 Related work on authoring tools

Originally, manipulating a virtual world required for substantial expertise in working with virtual environments. Therefore, the development of scenarios was mostly done by professional game developers. Recently, however, authoring tools are developed that allow non-programming-experts to manipulate the virtual environment. Such authoring tools usually enable the editing of a high-level scenario representation which is then employed by the game engine to establish the scenario within the game world. Several existing authoring tools are discussed below.

<e-Adventure>

The first example discussed here is <e -Adventure>, a platform solely intended for the development of so-called 'point-and-click' graphical adventure games (Moreno-Ger et al., 2007). <e-Adventure> employs a document-oriented approach to authoring. A graphic authoring tool enables authors to adapt a storyboard, which is documented in an xml-file. In turn, this xml-file can be processed by the game engine to produce the actual game environment. The author is able to edit the narrative flow and the way interactions of the player affect the course of the story. Even though <e-Adventure> allows people without any programming skills to develop a game scenario, it still requires a lot of manual labor to specify all the details required to establish the scenario. Moreover, the resulting storyboard describes no more than

the story; the educational intentions and underlying learning objectives are lost in translation.

Shai

Van Est et al. (2011) have also proposed higher level scenario editing to offer instructors, as non-experts in game development, easy access to the game's content. In their approach, called Shai, Van Est et al. (2011) employ a causality-based authoring technique: the instructor sequences scenario building blocks (i.e. actions and events) in a time line or causal chain to author the scenario. One big disadvantage of this approach is that it only allows for linear scenarios. In order to create non-linear scenarios, the author would need to be able to work with if-then statements and other flow-control strategies, which take place at a lower scripting level. Moreover, this approach would not be feasible to incorporate within the PEG architecture since the goal-based nature of the NPC behavior modeling does not guarantee that the events defined by the instructor will actually take place at runtime.

StoryTec

Mehm et al. (2012) have described 'StoryTec', an authoring tool for serious games. In StoryTec instructors can create a game model by connecting so-called scenes in a directed graph. Scenes are comprised of small parts of the overall narrative and contain a set of objects and a set of NPCs. Instructors can also create new scenes. StoryTec offers instructors a visualization of the storygraph. The game logic is defined by high-level commands called 'actions'. Actions are triggered by events in the game environment. Such triggers can be specified in the form of conditional rules. By creating scenes, adding actions to them and ordering them into a graph structure, the instructor is able to build a story that responds to the actions of the learner. Even though StoryTec does allow for branching storylines and a user-friendly graphical interface, developing a new scenario is still a laborious endeavor because each scene needs to be fully specified.

Wide Ruled and Story Canvas

Skorupski et al. (2007) have proposed Wide Ruled, an authoring tool to create and edit the HTN representation (comparable to the PEG's ontology) employed by an HTN planner to automatically generate stories (Skorupski & Mateas, 2009). The scenario author can express goals that need to be achieved during the story, and add NPCs with traits, story goals, or plot points. In turn the system uses this knowledge representation to automatically create a story that adheres to the restrictions added to the representation. Each plot fragment is linked to a set of author goals that are achieved through the execution of the plot fragment. In addition, plot fragments are linked to preconditions that are required for a plot fragment to be applicable. Story Canvas is an evolution of Wide Ruled in which comic book and storyboard techniques provide visual authoring metaphors in order to simplify the authoring process (Skorupski & Mateas, 2010).

The authoring process in Wide Ruled results in an automatically generated story that satisfies the goals of the author and matches the NPC's personalities. Wide Ruled allows an author to manipulate initial conditions of the story world (e.g. author goals) and to manipulate the story world during the planning process, because

the knowledge base employed by the HTN planner is limited to the content added by the author. But as the knowledge base becomes larger, the storyline can no longer be controlled other than it being consistent with the entire knowledge base. Wide Ruled and Story Canvas offer the instructor an authoring tool to add game content to the automated scenario generation process. However, they do not allow for any forced selection criteria, other than omitting the undesired options from the system's knowledge base. As such, Wide Ruled is very comparable to an authoring tool for the ontology which is employed by the scenario creator. However, it would be interesting to see whether the ideas in Wide Ruled (and Story Canvas) could be employed to offer instructors a more user-friendly interface to edit the contents of the ontology employed in the PEG.

Comparing these approaches to the PEG's authoring tool

A comparison between these approaches and the current design for the authoring tool in our PEG architecture shows that other approaches do not offer any way for the authoring tool to comment on the educational meaning, let alone correctness, of the scenario. On the one hand, the other approaches offer instructors a lot of freedom to create scenarios as they please, but on the other hand, this may also result in incoherent and/or unsuitable scenarios. Most authoring tools have been developed as a way for the instructor to create every scenario ever imagined. However, the quality of the scenarios is never warranted.

In contrast, the PEG's tight linkage between the authoring tool and the scenario creator not only allows for instructors to transmit their wishes regarding the scenario on a high level, but it also enables the PEG to support instructors, because the educational intent of the scenario is also clear to the PEG. Both the PEG and the instructors reason about the scenario content in terms of the task, the learning goals, the learner's current competencies, and the meaning of the environmental elements within this context. This allows them to communicate on that level, exchanging critiques and adjusting their preferences and knowledge. All in all, this allows for a better collaboration between the human and the system.

9.5 Concluding remarks

In the PEG architecture instructors can control the scenario in three ways: 1) manually author a scenario plan and provide that plan in the role of the scenario creator, 2) provide constraints for the scenario plan to be generated automatically by the scenario creator agent, and 3) control one of the NPCs in the scenario.

The authoring tool design presented in this chapter facilitates the second type of control; it offers instructors the opportunity to engage in a step-wise process of providing scenario constraints for the scenario creator. First, they select a setting. Thereafter they specify the learning goals. And lastly, they select NPCs and assign narrative goals and attitudes to the NPCs. The authoring tool provides the instructors with a suggestion for the NPC roles, goals, and attitudes based on the selected learning goals.

The results show that the authoring tool enables successful cooperation between the system's scenario creator and the instructor to produce suitable and creative

scenarios. Instructors experienced the authoring tool to be usable and to produce satisfactory outcomes.

Not all features and claims could be evaluated in the present study. There is a need for future research regarding claims and features of the component's design. First, the connection between the scenario creator and the authoring tool needs to be developed and verified. In addition, an experiment should investigate the quality of the scenario plans created through each of the three processes mentioned above, i.e. manually, collaboratively, or automatically.

Several future improvements of the authoring tool have been proposed that offer the instructor additional support during the authoring process and more extensive control over the learner profile and the scenario at playtime.

To conclude, the authoring tool offers instructors a way to collaborate with the PEG in a natural, understandable, and accessible manner to produce effective scenarios.

CHAPTER 10

Components in Development:

The learner model
and
the reflection component

Abstract - This chapter presents the results of preliminary investigations on the two remaining functional components of the PEG: 1) the learner model and 2) the reflection component.

Personalization is a crucial aspect of the concept of PEGs. To enable automated personalization, a learner model is essential. The goal of the *learner model* is to select a suitable learning goal for the learner to work on in the upcoming scenario, and to determine the appropriate initial level of scaffolding. For this, the learner model keeps track of relevant information about the learner, such as the learner's competency levels and motivation. The learner model was implemented in a BDI-based intelligent agent prototype. This prototype behaved as could be expected from the system specification.

The *reflection component* stimulates learners to develop a deeper understanding of the events taking place during scenarios in the learning environment. For this, it encourages learners to reflect on their performance and provide explanations for the correctness or incorrectness of their choices. If learners are unable to formulate correct explanations themselves, the reflection component provides additional hints and instructional explanations. Based on literature research, a BDI-based dialog appears to be a feasible approach to establish the behavior of this component. Yet further investigation is needed in order to obtain more solid evidence for this suggestion.

The research on the learner model was conducted in collaboration with C.C.M. van Rooij, also documented in:

Van Rooij, C. A., "User modeling for complex personalization problems: A general methodology and BDI-agent-based framework", MA thesis, Utrecht University, 2014

> "That's what learning is, after all; not whether we lose the game, but how we lose and how we've changed because of it and what we take away from it that we never had before, to apply to other games. Losing, in a curious way, is winning."
>
> **Bach (1984)**

The functional architecture contains two more components that are yet to be discussed in detail: the learner model and the reflection component (reflector). Due to time limitations, the investigations on these components are not as elaborate as the studies presented in the previous chapters. This chapter presents the *preliminary results* of our research on these two components, along with ideas and directions for future research. Section 10.1 discusses the research on the learner model and Section 10.2 the research on the reflection component.

10.1 The learner model

The learner model is a particular type of user model that is responsible for keeping the information about the learner up to date and selecting a suitable learning goal based on the learner's competencies. The learner model is more than just a database with information about the learner. It also contains a model of the way a learner develops competencies, i.e. a theory about learning. The requirement for the learner model component, identified in Chapter 4, was as follows:

R07 - Scenarios must be selected dynamically based on the learner's current competencies

This requirement illustrates the topic of this thesis: a design for educational games that are *personalized*. As Bakkes et al. (2012, p.1) state: "A personalized game is a game that utilises player models for the purpose of tailoring the game experience to the individual player."

In contrast to a regular game that is always the same for every player, a personalized game adapts itself to an individual player based on a dynamic model of the player. Examples of ways in which the game may adapt itself are adjustments in the game world, task sequence, mission, feedback, game mechanics, narrative, difficulty, or NPC behaviors (Bakkes et al., 2012).

In this thesis we make a distinction between adaptation and personalization. An adaptive game responds to the actions of the learner in a reactive manner: it recognizes that the current course of action is not effective, it adjusts its behavior. A personalized game, on the other hand, employs a dynamic model of the player to decide what course of action is most likely to increase future effectiveness.

An example of adaptation in the PEG is the scaffolding component (or monitor). It takes care of the adjustments in the levels of challenge and support in realtime, in response to its assessment of single actions of the learner. It does not have a model of the learner to determine its responses.

In contrast, the learner model is responsible for the dynamic selection of the learning goal that is to be addressed in the upcoming scenario. For this, it builds and maintains a model of the learner and predicts the effect of choosing a particular learning goal to be addressed in the upcoming scenario. This personalization is not

performed in realtime, but takes place in between scenarios.

The reason for personalization in PEGs is to support incremental competency development, directed feedback, and personalized support during reflection. In our design, these last two tasks have been separated from the learner model and are carried out by the reflection component (see Section 10.2). The reflection component makes use of information available in the learner model.

The learner model in the PEG architecture has a dual functionality. First of all, it collects, represents, and provides information that is required to support personalization in the PEG. The second function is to select a personalized learning goal for the upcoming scenario based on the learner's current competencies.

10.1.1 Inferring and representing a learner's competencies

The first task of the learner model is to maintain a representation of the learner's current competency levels. Inferring the learner's current competency levels is not straightforward. Indeed, the goal of (automated) SBT is that the learner becomes proficient in a particular task domain by developing the required competencies. However, whether the learner's competency level has actually increased is relatively hard to measure or estimate. Competence development refers to a change in the mental state of the learner, which is not directly observable. Therefore, the learner's behavior, which ís observable, needs to be interpreted in order to assess the underlying competency levels.

Koedinger et al. (2012) describe this problem as follows: When teaching a learner a competency, the instructor offers the learner an exercise involving the performance of a task that requires that competency. The only way of knowing to what extent the learner currently possesses the underlying competency is by interpreting the task performance, asking learners to vocalize their thought processes during the exercise, and reflect on what they had learned from the exercise. Furthermore, the only way to find out whether this experience has resulted in the intended increase in the learner's competency level is by offering the learner a different scenario that is assumed to address the same competency and see whether the learner performs better than last time.

Thus, a PEG should be able to infer the learner's competency development by interpreting the learner's task performance in terms of the underlying competencies. A single task performance can be used to assess the learner's current competency level, but a single measure may be unreliable. Two task performances, e.g. the previous performance and the current performance, can be used to assess the learner's progress. However, obtaining a clear and reliable image of the learner's competencies generally requires multiple assessments.

Interpreting a learner's task performance

In order to infer the learner's competencies, the raw information obtained from the (simulated) environment needs to be processed by three interpretation layers: 1) the observables, 2) the interpretation of the observables, and 3) the inferred learner characteristics.

The first layer concerns the information about the current state of the learner that can be *directly observed*. Exactly which information is directly observable by the

system depends on the available data sources provided by the (simulated) environment. Examples of possible data sources are: action logs consisting of all actions performed by the user, direct input obtained from the user in pop-up dialogs and other GUIs, but also, e.g., web-cam images, heart rate variability, key strokes, and galvanic skin conductance are potentially possible.

The second layer concerns the interpretation of the observables based on models about, e.g., facial expressions, emotional state, mental effort, and correct vs. incorrect task performance (Neerincx et al., 2009). How exactly the observable information is interpreted depends on the available data and the employed models. This interpretation provides information about the current state of the learner. It can be regarded as the system's situational assessment to gain situational awareness about its current task environment of which the learner is part.

The final layer is the inference of a learner's overall characteristics, such as the learner's competency levels, based on the interpreted observables. This inference step includes additional models about the relations between of a given situational assessment and the underlying learner characteristics. For example, say that the observable data lead to an interpretation of the learner's mental effort (high) and of the learner's correct/incorrect task performance (correct). This third layer provides a model that describes how this situational assessment relates to the learner's underlying competency levels. A possible model is that a correct response with a high mental effort points to a relatively moderate competency level, i.e. learners still need to practice as to lower their mental efforts during task performance (Paas et al., 2005; Paas et al., 1994).

Representing the learner's competencies

Assuming that competency develops incrementally, it should be possible to construct a prerequisite relation between competencies, stating that a certain competency requires the learner to have mastered one or more other underlying competencies (Van Merriënboer et al., 2002). Representing competencies in this manner results in a hierarchical tree structure, where each node represents a competency. Each node can have zero or more children. A child node represents a sub-competency that is prerequisite to its parents.

Common types of learner models represent the learner's competencies as a subset of the domain model, usually referred to as an *overlay model* (Brusilovsky & Millán, 2007; Ohlsson, 1986). Originally, overlay models contained a simple yes or no for each of the competencies, but nowadays most models allow for a representation where competencies can be mastered to a certain degree. In the case of a competency hierarchy, this would mean that for each tree in the competency tree the learner model contains a score that represents the learner's competency level.

The overlay model has been criticized for not representing common misconceptions (Brusilovsky & Millán, 2007; Ohlsson, 1986). One way to overcome this limitation is by adding so-called bug-models to the system's knowledge base. Bug models allow the system to recognize misconceptions in the user's problem-solving knowledge, which allows them to provide the learner with explanations that are specifically directed at those misconceptions.

10.1.2 Selecting a suitable learning goal

The second function of the learner model is to select a learning goal based on the learner's current competency levels. Once the learner's task performance has been interpreted, the learner model has a representation of the learner's current competencies. The learner model uses this representation to select a learning goal that is *suitable* for the learner. But what does *suitable* mean? And how to determine a learning goal that is suitable, given a set of mastered competencies?

Based on the analysis presented above, and the requirements presented in Chapter 4, pages 53-55, the following requirements can be formulated for the design of a learning goal selection function:

- The selection of the learning goal should be based on the idea of incremental competency development and a gradual increase in complexity of the learning topics offered to the learner. First, the learner should start working on the competencies with the lowest number of prerequisite competencies, i.e. competencies that do not require any prior knowledge. Once the learner has mastered the prerequisite competencies of an overlying competency to a certain competency level, then the learner is allowed to train that overlying competency as well.
- The learner should perform several consecutive scenarios addressing the same learning goal in order for the system to make an accurate estimate of the learner's competencies. Therefore, as long as the learner is motivated to continue with the selected learning goal (e.g. not bored), the system should stick to the current learning goal. Nota bene: this does not mean that the learner is repetitively offered the same scenario. Even if the learning goal remains the same, the scenario creator will generate a different scenario every time and the learner will still participate in a large variety of scenarios.

10.1.3 Using a BDI agent to represent a user

The previous subsection presented the functions of a learner model in a PEG, being (a) inferring and representing the learner's competencies, and (b) selecting a suitable learning goal for the learner in his training program. This section proposes the concept of BDI (Beliefs, Desires, Intentions) to implement these functions.

The foundations of the BDI-agent paradigm lie in Bratman's folk psychology of beliefs, desires, and intentions (Bratman, 1999). This was explained earlier in Chapter 5. A BDI agent has beliefs about itself and its surroundings, e.g. "I am hungry, there is a piece of cheese in the fridge, and there is a restaurant further down the street". A belief may trigger a desire (i.e. goal), e.g. "I want to eat". Based on its possible action plans, e.g. "get the cheese", and "go to the restaurant", the agent adopts the most feasible plan to reach its goal. If the agent believes it is able to achieve its goals by executing a certain plan, it adopts that plan as an intention and executes it. If multiple plans are feasible, it may decide which plan it adopts based on its preferences, goals, and beliefs. For instance, the beliefs "I don't have any money" and "the restaurant is expensive" may cause the agent to opt for the cheese plan.

The main advantage of using a BDI agent to represent the user is that the BDI architecture provides an intuitive representation of the learner model's behavior. This can be helpful when the learner model needs to be inspectable, hence understand-

able, for the instructor and the learner. For instance, if the learner does not understand why the learner model has decided to select a particular learning goal for the learner, the BDI-based learner model is able to explain its decision in terms of beliefs, desires (goals), and intentions (plans) (Harbers, 2011). In addition, the logic- and rule-based nature of the BDI components lends itself really well to present understandable graphical overviews of its contents, in contrast to, for instance, Bayesian Networks.

The three constructs of the BDI agent are used as follows:

Beliefs: The beliefs are the agent's informational state. Beliefs can be static or dynamic. Static beliefs do not change as a result of incoming information. As such, these can be regarded as axioms and premises. Dynamic beliefs represent current knowledge about the state of affairs in the agent's environment. Furthermore, the agent can also infer new knowledge based on the available percepts, axioms, and premises.

Desires: Desires refer to the agent's internal goals. They represent objectives or situations that the agent would like to accomplish or establish. As a result, the agent will actively pursue this goal. From here on we will refer to desires as *goals*.

Intentions: Once an agent has chosen an action plan to actively pursue, it becomes an intention. The agent is determined to execute its intentions; it will not drop an intention unless the intention is 1) achieved, 2) no longer deemed possible by the agent, or 3) no longer required because the purpose is no longer present. Intentions initially contain abstract plans containing other plans; the details of the plan are filled in as the agent progresses through the plan's execution. This reflects the dynamic aspect of the agent: the agent does not employ a completely worked out plan, but in fact constructs the plan as it is performed. From here on we will refer to intentions as *adopted plans*. In addition, we will refer to the complete set of plans available to the agent as the *plan base*.

10.1.4 Learner model - design proposal

Figure 10.1 shows the learner model as a BDI agent within the context of the functional architecture. The learner model receives the performance score and motivation (indirectly, via the director) from the scaffolding component. In turn, the scenario creator (indirectly, via the director) receives the selected learning goal and the initial level of scaffolding from the learner model.

When designing a BDI-agent, it is important that the constructs of BDI are used as they are originally intended. For instance, beliefs should be used to represent declarative knowledge and cannot be inconsistent. Goals should be used to instigate autonomous pro-active, goal-directed behavior. And plans should be used to represent procedures allowing the agent to establish the goals. The learner model design employs the three constructs of BDI as follows:

Beliefs: The beliefs of the learner model agent are:

- Information about the competency structure underlying the tasks to be learned
- The learner's score on each of the competencies in the competency tree (derived from the performance scores)
- The learner's current motivation level (inferred from data obtained through a questionnaire)

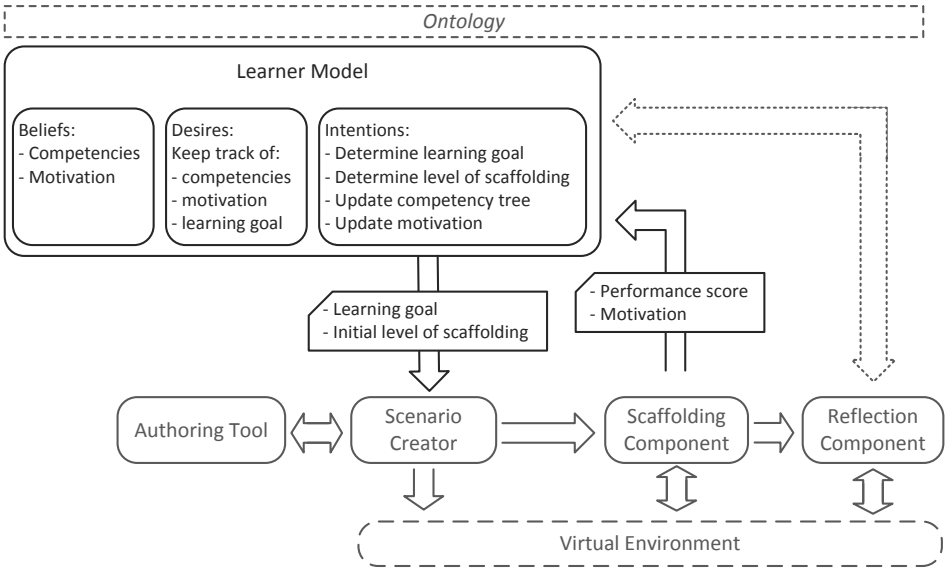

Figure 10.1: The learner model; as part of the functional architecture

- The current learning goal
- The minimum amount of consecutive runs with the same learning goal

Desires/goals: The goals the learner model agent can actively pursue are:

- Determine a learning goal for the learner
- Assess the learner's performance
- Assess the learner's competency levels
- Assess the learner's motivation
- Determine initial level of scaffolding

Intentions/plans: The plans the learner model agent can adopt to accomplish its goals are:

- Calculate the learner's performance score (using algorithm PS)
- Update the competency level (using algorithm CL)
- Update the learner's motivation level (with algorithm M)
- Determine learning goal (with algorithm LG)
- Determine initial level of scaffolding (with algorithm ILoS)

Each of these plans takes the form of an algorithm to compute the respective values. Other plans might be added to the plan base to reach the goals in alternative ways. The prototype, however, contains exactly one plan to reach each goal. These plans (or intentions) are presented further down (see Subsection 10.1.5).

10.1.5 Implemented learner model prototype

The design of a BDI learner model was implemented with the programming language 2APL. The prototype was developed for training a learner in the domain of First Aid.

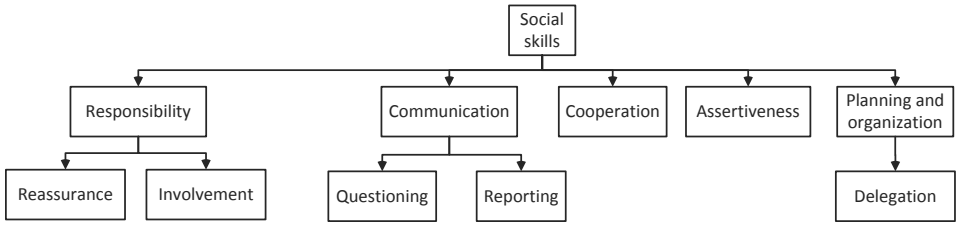

Figure 10.2: The competency tree used in the prototype

The current prototype employs a hierarchical competency tree. Each node represents a competency. Each node is tied to a score representing the learner's competency level. Each node has zero or more children. A child node represents a subcompetency of the overlying competency. The competency to be learned is "social skills". It is organized into a competency hierarchy, which is depicted in Figure 10.2. The competency hierarchy is part of the ontology and has been verified by four domain experts (see Chapter 6).

Plan/intention: Interpret the learner's performance (PS)

One function of the learner model is to determine how well the learner performed based on the learner's in-game behavior. Each scenario originates from a scenario plan (see Chapter 8). This scenario plan contains an action sequence for the learner to perform. The learner's truly performed actions are compared to the correct action sequence as specified in the scenario plan. At all times, the action performed by the learner is assessed as either one of the following behaviors:

1. The learner performs the correct action
2. The learner makes a mistake
3. The learner times out

The complete action sequence as performed by the learner can be translated into a concatenation of these three categories. This results in an action log that is kept by the monitor agent. For instance, a possible sequence would be: [correct, incorrect, time-out, correct, incorrect, incorrect, correct] (see Table 10.1).

We designed the following algorithm to calculate a performance score from an action log such as the one presented above. The algorithm produces a performance score that ranges between 0 and 100.

1. For each to-be-performed action in the original action sequence as described in the scenario plan, an action score is calculated based on the initial level of scaffolding, the amount of errors the learner has made, and the amount of time-outs the learner has received before the correct action has been performed.
   A. If the learner has not timed out, the action score is:

$$ActionScore = maximum(0, 100 - \#mistakes * 25)$$

   B. If the learner has timed out, the action score is:

$$ActionScore = maximum(0, 100 - 25 * (LoS + \#mistakes + 1))$$

with LoS representing the level of scaffolding during the time-out.

2. The performance score is calculated as the average action score:

$$PerformanceScore = \frac{1}{n}\sum_{i=1}^{n}(ActionScore_i)$$

In the current algorithm, the level of scaffolding is only of influence if there has been a time-out. The reason for this is that, in the multi-agent system prototype, scaffolding interventions are only executed if the learner times out.

We will provide an example of how the performance score is calculated: if the initial level of scaffolding is 2 and the learner performs as specified in the action log provided above, then the learner receives a performance score of 50 out of 100 (see Table 10.1). The first action is immediately performed correctly, hence the action score is $100 - 0 * 25 = 100$. The level of scaffolding is decreased by 1 because of this correct action (see Chapter 5). The second action is performed correctly after one mistake and one time-out. First, after the mistake, the level of scaffolding is increased by 1: it is 2 again. This means that at the time of the time-out, the level of scaffolding is 2, which results in an action score of: $100 - 25 * (2 + 1 + 1) = 0$. The level of scaffolding *after* the time-out is 3. In turn, the learner makes two mistakes before performing the third and final action correctly. This results in an action score of $100 - 2 * 25 = 50$. In the end, the performance score is the average of these three values, which is 50.

Table 10.1: An example of computing the performance score

Action log	Level of Scaffolding	Action Score	Performance Score
correct	2	$100 - 0 * 25 = 100$	
incorrect	1		
time-out	2		
correct	3	$100 - 25 * (2 + 1 + 1) = 0$	
incorrect	2		
incorrect	3		
correct	3	$100 - 2 * 25 = 50$	
final			$\frac{1}{3} * (100 + 0 + 50) = 50$

The learner performed 1 out of 3 actions correctly, for one action he received a very concrete hint telling him what to do, and the last action may have been a lucky guess after trying two incorrect options. Based on this knowledge, the obtained score (i.e. 50 out of 100) seems reasonable. However, the penalty points for the mistakes and time-outs (now set to 25) are unfounded and will need to be refined (see Discussion, 10.1.7).

Plan/intention: Updating the competency level (CL)

For reasons of experimentation, the competency level of the learning goal is currently replaced with the average of the performance score and the old competency level:

$$NewCompetencyLevel = \frac{OldCompetencyLevel + Performancescore}{2}$$

Future research should investigate the need for a more sophisticated formula (see Discussion, 10.1.7).

Plan/intention: Updating the learner's motivation (M)

Each time the learner model needs to determine the learning goal for the upcoming scenario, it inquires after the learner's motivation to continue with the current learning goal. It does so by means of a pop-up window asking learners to rate their motivation to continue with the current learning goal on a 9-point Likert scale.

For reasons of experimentation, the learner model currently replaces the old value with the value obtained from the learner:

$$NewMotivation = RatedMotivation$$

Future research should investigate the need for a more sophisticated formula (see Discussion, 10.1.7).

Plan/intention: Selecting a suitable learning goal (LG)

The algorithm for selecting a suitable learning goal was based on the requirements identified in Subsection 10.1.2:

- The learner should master the prerequisite competencies before the overlying competency can be trained
- The learner should participate in a minimum number of consecutive scenarios addressing the same learning goal in order for the learner model to make an accurate estimation of the learner's competency level.
- If the learner is no longer motivated, or if the learner has mastered the learning goal, a different learning goal should be selected.

The performance score is subjectively classified into three categories, i.e. good (performance score > 80), reasonable (50 ≤ performance score < 80), and bad (performance score < 50). The resulting algorithm for selecting the learning goal for the upcoming scenario is as follows:

A. The learning goal remains the same if:
   - ∴ the number of consecutive scenarios addressing the current learning focus is less than the minimum, or
   - ∴ the number of consecutive scenarios addressing the current learning focus is higher than the minimum, but the learner's motivation as stored in the belief base is higher than 5 and the competency level associated with the learning goal is not good (below 80)
   - ∴ the selection of a new learning goal ([B.]) fails

B. Otherwise, a new learning goal is determined using the competency tree:
   0. Start at the root.
      a. If the competency level of the root is good (above 80), training is completed.
      b. Otherwise, continue with 1.

1. Select a random child of the current node.
2. If the competency level of the current node is
   a. good (above 80), select a random sibling of the current node and repeat from 2.
   b. reasonable (between 50 and 80), select this competency as the learning goal.
   c. bad (below 50), and the current node
      :: has children, repeat from 1.
      :: is a leaf, select this competency as the learning goal.

Future research should investigate the need for different cut-off values for the performance categories (see Discussion, 10.1.7).

Plan/intention: Determine initial level of scaffolding (ILoS)

The initial level of scaffolding is based on the learner's current competency level for the selected learning goal:

A. If $CompetencyLevel < 25$, then $InitialLoS = 3$
B. If $25 \leq CompetencyLevel < 50$, then $InitialLoS = 2$
C. If $50 \leq CompetencyLevel < 75$, then $InitialLoS = 1$
D. If $75 \leq CompetencyLevel < 100$, then $InitialLoS = 0$

Future research should investigate the need for different cut-off values (see Discussion, 10.1.7).

10.1.6 Evaluating the learner model - design verification

The agent-based learner model was implemented in 2APL to work with the multi-agent system presented in Chapter 5. It was verified by Van Rooij (2014) through use case analysis and simulation. No actual learners were used to test the design. Instead, learners were simulated to test the implementation of the learner model. This was done as follows.

Use cases were developed describing all possible interactions between a learner and the multi-agent system prototype (see Chapter 5). For instance, one use case describes the learner making a mistake on a single action and specifies exactly what the resulting effect should be in the PEG: increase the level of scaffolding by 1, write down the event in the action log so the action score can be calculated afterwards. Another use case describes what happens when the learner performs the right action, and yet another one describes what happens during a time-out. Similarly, there are use cases describing updates of the competency structure based on various values for the performance score. Yet other use cases describe the selection of a suitable learning goal for specific cases, and so on.

The verification procedure entailed the simulation of each of the use cases. This was established by providing the prototype with the exact input as specified in the use case and checking the agents' internal belief bases, action plans, and adopted goals to see if their behavior corresponded to the behavior described in the use case.

The results of this verification showed that the learner model behaved as was specified in the use cases. As such, it was able to correctly:

- calculate the performance score
- update its competency tree

- inquire after the learner's motivation
- update its beliefs about the learner's motivation
- determine the new learning goal

The result of adding the learner model to the multi-agent system resulted in a gradual increase in complexity as the learner progressed throughout the adaptive curriculum. Furthermore, if a learner suddenly started performing worse, the learner model adjusted the training to be compliant with a lower level competency.

10.1.7 Discussion and future work

The preliminary research results show that the BDI paradigm offers a straightforward, intuitive, and successful approach to implementing a user model: user data can be represented in the beliefs, incentives for adaptations can be represented in the goals, and operations required to achieve the adaptations can be represented in the plans. The BDI approach also offers possibilities for an inspectable and explainable learner model which is beneficial in PEGs.

The work described above shows that the current implementation of the learner model behaves as intended by the design. Whether this design effectively supports a learner during competency development, is yet to be established. In order to evaluate the proposed learner model, human subject studies are required to compare effects of personalized versus non-personalized training trajectories using PEGs.

This preliminary study investigated and tested the design of the learner model, not the accuracy of the algorithms and formulas. In order to test the effects of the learner model, these aspects should be sufficiently validated and possibly refined. Future research should reveal whether more sophisticated formulas for the calculation of the performance score and competency levels would improve the overall performance of the PEG. One possibility would be to use probabilistic reasoning to update the competency levels (Göbel et al., 2010; Kickmeier-Rust & Albert, 2010; Millán et al., 2010). Such an approach calculates the probability that a learner possesses a certain competency given the current performance score.

The current design offers only one action plan for the learner model to determine the next learning goal. To include the instructor in the didactic strategy to be used in this process, additional selection algorithms can be added to the plan base. Such plans should represent alternative instructional strategies for dynamic curriculum construction, or could even be used to specify a predetermined curriculum. By specifying a belief that represents the preference of the instructor for one of the possible learning goal selection strategies, the instructor would be able to select one of the available strategies to be used by the learner model. Such an approach could possibly improve the quality of training, because the instructor is able to determine what teaching strategy is most appropriate for a given learner.

Another way of including the learner and the instructor to cooperate with the learner model is: 1) By allowing the learner and/or instructor to critique the competency levels by suggesting alternative values (Cook & Kay, 1994); or 2) to have the learner model ask the instructor for input on which node in the competency tree should be used as the root. This would allow the instructor to restrict the set of possible learning goals to be selected by the learner model. Both of these refinements are likely to result in a better performance of the PEG, as a result of more accurate

values for the learner's competency levels, and the possibility for the instructor to limit the size of the curriculum when necessary.

The learner model could also be used to keep track of the learner's competency development after the application of a certain learning goal selection strategy or a particular tutoring strategy during the reflection stage in the form of a utility. Based on this utility, the PEG could, for instance, select strategies that have resulted in high learning gains in previous sessions.

The learner model's functions can also be extended such that it allows for predictions of the learner's future performance on a given task (Zook et al., 2012). By constructing a performance curve for a given task, the required level of scaffolding for various tasks in the scenario can be predicted. Currently, the level of scaffolding adapts itself in response to the previous action of the learner. A performance curve would allow for proactive adjustments in the level of scaffolding.

Finally, the current design of the learner model does not yet support integration with the reflection component (see below). Yet the reflection component needs this information in order to know what strategy it should use during reflection. Providing the learner model with the competency level of the current learning goal should not cause any problems. However, the reflection component will also need to assess the learner's reflection and provide the learner model with a reflection score. Therefore, the learner model should be able to process this reflection score and update its competency levels based on the outcomes of the reflection process.

10.2 The reflection component

After completing the scenario, the learner's performance data are available and the training session progresses to the reflection stage. The reflection component offers learners automated support to reflect on their performance.

The requirements for the reflection component, identified in Chapter 4, were as follows:

R13 - When the scenario has finished, the learner should be stimulated to reflect on the performance

R16 - After reflection, the learner must receive directed and overarching feedback by comparing the performance outcomes to the previous performance and to the performance standards associated with the learning goals

The processes taking place during the reflection stage are not predetermined or procedural, but instead can be characterized as open and of a qualitative nature. In practice, these properties complicate the design of an automated system. In a personal human-to-human tutoring dialog, human tutors use contextual information and their domain knowledge, social skills, and interpretation skills to optimize the effectiveness of the dialog. Looking realistically at the current state of the art in artificial dialog systems, natural language processing, natural language generation, and sensor data interpretation, such as facial expression recognition, it does not yet seem feasible to match the quality of human-to-human tutoring. But what do we currently know about the possibilities to automate the tutoring dialog? To what extent can we currently provide automated support during the reflection stage? And

what effects can we expect from automated support on the quality of reflection and on the resulting learning gains?

First, we investigate the process of reflection in more detail. This entails a discussion of literature about self-explanations and instructional explanations, and about related work on the automation of these processes (Subsection 10.2.1). Thereafter, Subsection 10.2.2 discusses related work on automated tutoring dialog systems. The section closes with a preliminary design for the reflection component (Subsection 10.2.3).

10.2.1 Self-explanations and instructional explanations

An important goal of reflection is motivating learners to think about and explain their performance and the associated outcomes. One way to achieve this is by encouraging the learner to engage in something called 'self-explaining': the learner is encouraged to explain each step of the solution. In addition, the learner is stimulated to evaluate the decisions and actions made during the scenario, and the effects thereof. Self-explanations have been shown to be beneficial for learning (Chi et al., 1989; Chi et al., 1994; Renkl et al., 1998). The reason for this is that through self-explaining, learners are forced to actively process their existing knowledge and construct new relationships between their prior knowledge and their new experiences (Chi et al., 2001). However, in order to successfully employ self-explanations in training, the following notions should be taken into account: 1) prompting, 2) scaffolding, and 3) instructional explanations.

First of all, learners do not often spontaneously engage in self-explaining (Conati & VanLehn, 2000; Renkl, 1999; Renkl, 2002). Therefore, they should be *prompted* to provide self-explanations: the learner is encouraged to construct an explanation for each solution step.

Secondly, learners are often not able to produce explanations that help them to overcome the gaps in their knowledge (Aleven & Koedinger, 2000b; Conati & VanLehn, 2000; Renkl, 1999; Renkl, 2002). Therefore, the self-explanation process should be *scaffolded*: if the learner is unable to produce the correct explanation for the solution step, the tutor should engage in a tutoring dialog to elicit the correct explanation. This is usually done by asking questions about the example or performance that lead to the correct explanation.

And lastly, additional instructional explanations and/or feedback should be provided when learnerst' are not able to construct the explanation themselves or with the help of scaffolds. Instructional explanations are not actively constructed by or elicited from the learner during a tutoring dialog. Instead, they are provided by the instructor or instructional system, and the learner passively adopts the knowledge conveyed by the explanation. The provision of (passive) instructional explanations undermines the instructional concept of (active) self-explanations (Aleven & Koedinger, 2000a; Conati & VanLehn, 2000; Gerjets et al., 2006). Therefore, researchers generally agree that learners should be encouraged to provide self-explanations as much as possible. Providing learners with additional instructional explanations should be used as a last resort, and only when this is strictly necessary (Conati & VanLehn, 2000; Renkl, 1999; Renkl, 2002).

10.2.2 Related work on automated support for self-explanations

Below, we take a look at various approaches to the automation of supporting self-explanations.

Menu-based support for self-explanations

Menu-based support employs a graphical user interface which usually prompts the learner to self-explain fully worked-out examples (Conati & VanLehn, 2000; Gerjets et al., 2006). It does this by uncovering the solution steps to the problem one by one, and asking the learner to provide an explanation for that step. Sometimes, the learner is first offered an example with a full instructional explanation and is asked to self-explain the next example in a way that is similar to the elaborate explanations previously provided. Gerjets et al. (2006) found that the use of a non-personalized menu-based self-explanation tutor led to significantly lower learning gains compared to a system that did not prompt the learners to self-explain. They argued that this was because learners were already proficient in the learning topic at the pretest and learners were forced to provide self-explanations for solution steps they had already mastered. This hypothesis was confirmed by a different study, which was conducted by Conati & VanLehn (2000). Conati & VanLehn (2000) compared the learning gains of learners working with 1) a personalized menu-based self-explanation coach vs 2) a system consisting of the same interface that did not prompt learners to self-explain. Results showed that working with the SE-Coach at an early learning stage improved the learners' performance, because learners in an early learning stage did benefit from structured help in using domain knowledge to generate effective self-explanations. However, as learners become more proficient in the subject matter, minimal prompting is sufficient.

From these studies, it can be concluded that prompting and scaffolding self-explanations should be based on the learner's prior knowledge, i.e. information available in the learner model and information obtained during the most recent task performance.

Typed-dialog based support for self-explanations

The studies described above investigated the use of menu-based tutoring: learners were able to select explanations from a menu, or fill out a template. Recent research has investigated the possible benefits of a natural language (NL) dialog instead of these menu-based approaches. A reason for the use of an NL instead of a menu-based approach is that constructing an explanation using one's own words is beneficial for learning, because it requires the learner to recall instead of recognize the relevant knowledge (Aleven & Koedinger, 2000b).

Dialogue tutors are able to understand the meaning of the NL input with the use of an NL interpreter (Popescu et al., 2003; VanLehn et al., 2002). Based on the interpretation of the learner's input sentence these tutors respond by getting the learner to formulate a complete and correct explanation, for example, by asking the learner specific questions about parts of an incomplete or incorrect solution. Several strategies can be employed during the tutoring dialog (Aleven & Koedinger, 2000b): 1) if a learner provides an explanation that is too general, the tutor responds by asking the learner whether their general answer is generally valid. This forces the

learner to develop a more specific rule and an explanation of why it is applicable in this particular situation. And 2) if a learner is unable to come up with the correct rule, the system helps the learner to identify plausible candidates, and, for each one, discusses with the learner whether it is applicable. Studies by Aleven et al. (2004) and VanLehn et al. (2002) have shown that the development of NL dialog tutors takes considerable effort, but show inconclusive results. Based on these studies, we conclude that these approaches are as of yet still infeasible.

BDI-based dialogs

Recent research has shown that BDI-mixed initiative approaches to dialog management offer a way to create meaningful interactions and communication between humans and agents.

Vergunst (2011) investigated whether and how BDI agents could be used to generate robust task-oriented dialogs and instructions. The agent interprets sentences in terms of the underlying intention. In the same manner it is able to generate sentences to utter its intentions. A dialog structure is constructed to specify what kind of intentions are applicable at what point in the conversation. This structure can be used to set the scope for the expected intentions of an utterance expressed by the other party.

Muller et al. (2012) also employed a BDI-based approach to dialogs in an educational game that aimed to train real estate salesmen. Their BDI-based dialog game used an ontology to specify various building blocks to be used in conversations. In addition it employed a restricted grammar from which it could produce sentences using the information in the ontology. As a result, the agents were able to respond to the player. Furthermore, they could proactively ask questions referring to their own goals and intentions.

A BDI-based approach might also be useful to generate instructional explanations (Harbers, 2011). This can be useful to: 1) check whether the self-explanations provided by the learner are correct and sufficient, and 2) provide learners with additional explanations in case they are not able to produce such a correct and sufficient self-explanation.

Based on this explorative investigation, BDI-based dialogs seem to provide a less comprehensive alternative for the somewhat elaborate approaches discussed in the previous subsection. Given that the persons engaged in the dialog are discussing a task-related performance, this seems to be a feasible approach worth investigating for use within the reflection component.

10.2.3 Reflection component- preliminary design

Based on the information provided above, a preliminary design of the reflection component can be constructed (see Figure 10.3).

The reflection component offers the learner dynamic prompts for self-explanations based on the learner's performance. Such prompts address the most important decision points in the scenario and consist of questions such as: 'Why did you perform procedure P?', 'What was your assessment of the situation at time T?', and what was the cause of event E?'. The reflection component structures its dialog and interprets the input coming from the learner based on the knowledge available in the ontology,

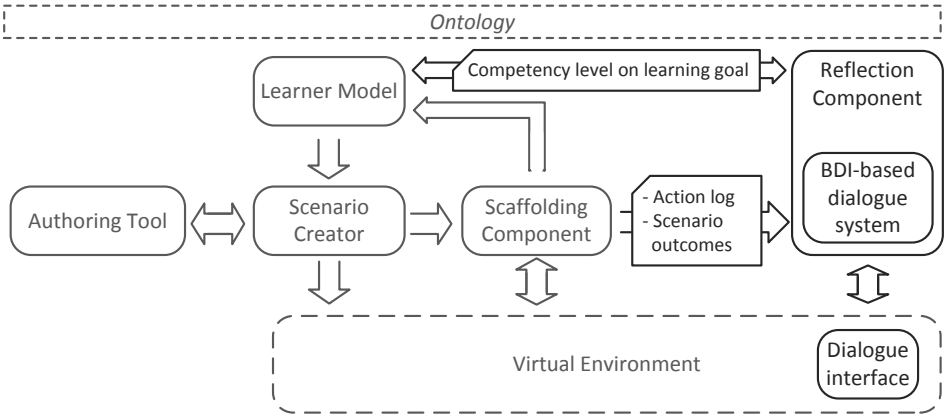

Figure 10.3: The reflection component; as part of the functional architecture

scenario plan, scenario outcomes, action log, and learner model.

As described above, during reflection the learner either constructs or receives the correct procedure, the correct explanation of what went right and wrong in the current performance, and a correct explanation of the scenario outcomes as a result of the performance. In addition to the reflection on the exercise, it is also recommendable to encourage learners to reflect on the learning process by looking at their progression towards the learning goal (Hattie & Timperley, 2007). For instance, the system could ask learners to recall their previous performance in a scenario that addressed the same learning goal (presuming that there is one) and to compare that performance to the current one.

This preliminary design of the reflection component completes Part II of the thesis. The final chapter reflects on the research and research process of this thesis.

Closing

General Discussion and Conclusion

CHAPTER 11

General Discussion and Conclusion

Abstract - The final chapter reflects on the contribution of this thesis to the scientific domain as well as the application domain. The scientific contribution is threefold: methodological, technological, and empirical.

Suggestions for improvement of the sCE method include (1) the development of a tool to organize the various documents relevant to the system specification and evaluation; (2) the explicit inclusion of a use-case based decomposition to support partial system evaluations in a more natural way; (3) the development of a repository to store and exchange reusable design patterns; and (4) the explicit inclusion of non-functional requirements.

The research also led to the following technological insights: (1) combining a multi-agent organization with an ontology is an effective way to coordinate the procedural alignment as well as communication between multiple functional components; (2) a multi-agent organization provides a natural way to promote adaptive automation based on the roles identified in the work environment; and (3) the combination of smart objects, an HTN planner, and BDI-based NPCs offers a seemingly feasible approach to procedural scenario generation.

Empirical evidence was obtained regarding the effectiveness of the following technology-enhanced learning solutions: (1) scaffolding interventions at runtime are beneficial for the instructional value of the scenario, and (2) the developed automated and semi-automated scenario creation designs proved to be effective in the creation of scenarios that address a predefined learning goal.

This research has resulted in a comprehensive list of functional requirements for SBT, along with positive and negative claims for each requirement. This list is considered to be reusable across designs. The proposed design for PEGs satisfies most of the identified requirements, some of which have also been justified through evaluations of the associated claims. Yet other claims still require additional investigations. The chapter closes with several suggestions for future refinements of the current design, such as additional features to promote (1) human-machine cooperation and (2) team training, and (3) a dynamic task model to further relax the need to manually author scenario solutions and task procedures.

"No book can ever be finished. While working on it we learn just enough to find it immature the moment we turn away from it."
Karl Popper

This thesis presented: 1) an empirical theory-driven approach to the design of automated scenario-based training (SBT) and 2) the resulting design proposal, i.e. Personalized Educational Games (PEGs). This final chapter examines to what extent the proposed approach and design contribute to the research aim.

First, we take a look at the scientific contribution of the thesis, addressing the employed method and technologies (Section 11.1). Secondly, we review the contributions of this research to the application domain (Section 11.2).

11.1 Scientific contribution

This thesis allows for conclusions about 1) the research process, i.e. the design method, 2) the use and combination of several technologies, and 3) the integration of learning theories into the behavior of an automated training system. Below, these topics are discussed in more detail.

Methodological contribution

Automated SBT, such as a PEG, is a complex cognitive system: it consists of man and machine collaborating in a joint task. In order to develop a design for such a system, the situated Cognitive Engineering method has been systematically employed throughout the research project. This allows us to reflect on the use and usefulness of the sCE method:

Multiple perspectives - The sCE method prescribes a threefold investigation: an analysis of the task domain, a consultation of human factors research, and an exploration of available technological means. As a result, the sCE method produces documentation of the problem description, the evaluation criteria, and the solution proposal. Moreover, it covers the perspectives of end-users, interaction designers, and technicians.

Problem analysis - When multi-disciplinary teams collaborate to design a complex cognitive system, it is important, yet difficult, to keep track of all evaluation criteria, alternative design choices, and the arguments for choosing one or the other. The sCE method creates a sense of awareness about this problem and provides an analytical framework to untangle the cause and effect relations in the system specification.

Systematic evaluation - Initially, the application of the sCE method led to the identification of: 1) a list of requirements (design constraints) to delineate the solution space and 2) a set of claims (evaluation criteria) to justify the adoption of those requirements in the design. Together, they provide a context for developing, testing, evaluating, and comparing alternative design solutions.

Consistent vocabulary - An additional result of the sCE method is an ontology. In the case of the current research, this is an ontology for automated SBT. The ontology provides clear and unambiguous definitions for important concepts in the system specification. In addition, it restricts the size of the employed vocabulary;

the concepts specified in the ontology have been consistently reused in interviews, system and component specifications, questionnaires, and even programming code. This resulted in a natural alignment between the various components and users. Because of this shared vocabulary, the components, developers, and users all operate on the same level of abstraction and reason about the same concepts. For instance, the learner model contains terms that are interpretable for instructors; NPCs reason about concepts that are familiar to the learner and the instructor; and the authoring tool employs the same concepts as the scenario creator.

There are also several potential improvements that would ease and/or improve the use of the sCE method. These are discussed below.

Documentation - A text document can easily fall short to clearly describe large projects and system designs. Providing a clear description of the work that has been done, and a specification of the relations between all the various documents describing different parts and aspects of the system can be quite a challenge. Recently, a tool was developed to support the application of the sCE method: the sCE-tool (sCET, see http://www.scetool.nl). sCET is a web-based tool that allows a team of designers, domain experts, and technicians to collaborate on the development of the system specification. The requirements, claims, and use cases are kept in separate databases, but are related items, which are linked by means of hyperlinks. This tool is a first step in the direction of an easier way to document information. In the near future, the tool will offer several additional modules, such as a plug-in for the WebProtégé ontology editor (Tudorache et al., 2008).

System decomposition - In the current research, a functional architecture was used to break the system down into smaller components. Indeed, specifying subsets of similar requirements, and identifying subtasks and/or roles within the system is generally a feasible way of decomposing a system during the development stage. However, a disadvantage of this approach is that it is relatively difficult to test and evaluate the system. When testing all components in isolation, it is difficult to know how the component will behave within the rest of the system. Especially when the component does not directly interact with the user, a user-based evaluation to investigate the intended effects of the isolated component can be complicated. On the other hand, when testing the system as a whole, it is often difficult to know what is causing the problem if the system does not behave as intended. Westera et al. (2010) describe an alternative way to partition the system specification: use-case-based decomposition. Use cases delineate a natural interaction between a user and the machine. As a result, the design for user-based experiments to evaluate parts of the system can focus at use cases, instead of functional parts of the system. Future research should investigate whether and how this alternative way of decomposing the system into smaller problems can complement the functional decomposition. For instance, the functional decomposition can be useful during the development stage, whereas the use-case based decomposition may be more useful at the evaluation stage.

Securing and transferring knowledge - An additional issue for future research is securing and transferring design research results. One of the reasons for employing a systematic approach to the development and evaluation of designs is to contribute to theory development. After testing a particular design for a particular application,

the obtained results should be secured in a generic format such that these results can inform other designers who are working on a similar application, design, or both. In other words, the systematic approach taken by the sCE method potentially offers the possibility to develop a repository of reusable design patterns. Additional research is required to further investigate this possibility.

Non-functional requirements - Lastly, the sCE method does not explicitly cover the specification of non-functional, e.g. technical, ethical, or budgetary, requirements. If the designer is not aware of these other requirements, this may cause problems when trying to move from the problem specification and evaluation criteria to the actual (technical) solution. The designer needs to perform this step in the design process in order to identify additional requirements, but may fail to do so. As a result, the optimal solution may be overlooked, or worse, the designer may design a solution to find that it is incompatible with the other, until then, neglected limitations, only when the design is already finished.

Technological contribution

The PEG design combines several technologies into a single system: multi-agent organizations, ontologies, HTN planners, intelligent agents, intelligent user interfaces, virtual environments, and intelligent dialogue systems. This has provided us with the following insights in the combined use of these various technologies:

Component alignment - The multi-agent organization provides protocols for all interactions between components. Furthermore, the ontology specified a restricted vocabulary to be consistently used by all system components. As a result, each functional component could be developed in isolation such that its input and output is well-defined and processable by all other components.

Adaptive automation - A related, yet different, advantage of an explicit agent organization is its flexibility with regard to the identity of the actor performing a role in the organization. Various roles in the system can be performed by a human or by an agent. Especially in cognitive systems this can be a useful feature to support dynamic task support. Depending on the operational demands of a given situation, the human operator may outsource one or more roles to be performed by agents.

Procedural content generation - The combination of smart objects, an HTN planner, and BDI-based NPCs offers a seemingly feasible approach to procedural scenario generation. Non-programmer domain experts are able to supply and verify content in the format of an HTN as well as an ontology. The scenario generator is able to produce a scenario that covers a fully specified task and satisfies all preconditions to allow the NPCs as well as the learner to perform all actions in the scenario.

Future user-based studies with instructors should investigate the expected usefulness of the ontology in extending the domain-specific vocabulary by non-programmer domain experts (e.g. instructors).

In addition to advantages of combining technologies that we experienced ourselves during our research, we also *anticipate* several advantages of this combination of technologies that should be further investigated in future research:

Flexible structure - An anticipated, yet untested, advantage of an explicit agent organization is that changes in the procedures or protocols of the organization can

be explicitly redefined, which makes it easier to guarantee the compliance of the system to such changes. This can be especially useful in systems that operate within dynamic environments and using variable procedures. Future research should investigate the practical usefulness of this theorized utility.
The current prototype employed emergent coordination, which is an implicit implementation of the organization. This was not considered to be a problem for the current purpose of verifying the infrastructure provided by the organization. However, to investigate the flexibility of the original design, future research should implement the organization explicitly. This means that the director should be able to pick from multiple agents applying for the same role. In addition, interfaces should be developed for the instructor allowing him/her to apply for the various roles in the organization.
Functional architecture to organization - The translation from a functional design to a multi-agent organization proved to be relatively intuitive in the current research project, namely by transforming the requirements into protocols and norms for the agent organization. Future projects should aim to investigate the more general applicability and usefulness of this finding.
Organization + ontology - The combination of an agent organization and an ontology offers a straightforward way of separating procedural knowledge from declarative knowledge in the system. The agent organization describes the procedural tasks in the system, whereas the ontology covers the declarative knowledge available to the agents. This appears to be a modular approach that fosters reusability: the ontology can be used by other system designs, including other agent organizations. Furthermore, the agents in the organization are able to work with the upper ontology, whereas the lower ontology can be adapted without the need to adapt the agents. Future research is still required to provide more conclusive answers regarding this approach.
Computational power vs agent autonomy - The employment of smart objects and intelligent agents offers possibilities for a reduction in complexity of the scenario creation problem. Because the intelligent agents are able to construct their own plans, and because they are able to determine how the objects in their environment may be of use in achieving their goals, there is no need for the HTN planner to construct complete action sequences for all NPCs. This means that NPC tasks can be specified in terms of their high-level available action plans, along with their preconditions; these plans do not need to be decomposed into actions. This would relieve the HTN planner of the need to specify a complete action sequence for all the NPCs, which would reduce the computational complexity of the scenario creation problem. However, this also means that agents are required to construct their own plans in realtime. Even though this is beneficial for the responsiveness of the scenario, it also increases the demand for computational processing power in realtime. Future research should investigate this trade-off.
Intelligent NPCs - This research did not include a thorough investigation of the NPC agents and how the NPC agents are to produce the intended behaviors, i.e. work with smart objects, achieve narrative goals, etc. However, it is essential that the NPCs are able to behave as such in order to realize the scenario as described by the scenario plan. Based on related work by others (Kallmann & Thalmann, 1998;

Mateas & Stern, 2003; Muller et al., 2012; Theune et al., 2003; Van den Bosch et al., 2009), we theorize that intelligent NPCs will indeed be able to produce the behavior as described in this thesis. However, additional research is needed to find definitive answers.

BDI-based user model - Open learner models are believed to be beneficial for 1) the accuracy of the learner model, and 2) for the learning and reflection process of the learner (Bull & Britland, 2007; Mitrovic & Martin, 2007). The BDI-based user model is believed to be a suitable approach when designing a user model that should be understandable and explainable for non-programmers (i.e. users). The intuitive nature of the BDI-paradigm allows for explanations in terms of goals, plans, and available information. Furthermore, the information (beliefs) in a BDI-agent is often represented in an explicit logical description. As a result, it is possible to present the agent's beliefs in an understandable form to the instructor and the learner (Harbers, 2011), as opposed to, for instance, Bayesian models (Zapata-Rivera & Greer, 2000). Future research will be required to investigate whether this hypothesis holds.

Intentional stance - The intentional stance adopted in the intelligent agent paradigm allows for explainable and understandable behavior, which is beneficial in PEGs where instructors with little knowledge of information systems and/or computers must collaborate with the PEG. The agents reason about and aim to achieve concepts that possess the same level of abstraction as the ones employed by the instructors. The anticipated benefits of this approach, for instance in dialogues between users and agents, are yet to be investigated.

Technology-enhanced learning contribution

An additional scientific contribution of our research has been the integration of theories about learning and instruction into the behavior of an automated system for SBT, more specifically, a PEG:

Realtime scaffolding - This research provided empirical evidence for the educational benefits as a result of adaptive scaffolding during scenario-based training. Furthermore, the thesis provides a way to formalize the reasoning processes underlying the adaptive scaffolds with the use of observable behavioral cues and scripted rules that intervene in the behavior of the NPCs.

(Semi-)automated scenario creation - In addition, this research resulted in two support functions for the instructor to dynamically create personalized scenarios: one fully automated function and one semi-automated function. Both of these functions proved to be effective in the creation of scenarios that address a pre-specified learning goal.

11.2 Application domain - contribution

In addition to the scientific contributions, this research also provided insights for solving the design problem:

"How to design an automated system for scenario-based training (SBT), based on scientific evidence, such that learners can engage in personalized autonomous training?"

This investigation was guided by four research questions:

I. What are the requirements and anticipated effects of automated SBT?
II. What design could satisfy the identified requirements?
III. Does the proposed design indeed satisfy the identified requirements?
IV. Does the proposed design bring about the anticipated effects?

Research Question I addresses the specification of a list of requirements applicable to *any design for automated SBT*. This means that regardless of the design proposal, these requirements hold. Research Question II and III address the validity of *one particular design proposal for automated SBT*: Personalized Educational Games (PEG).

RQI: The requirements and effects of automated SBT

The sCE method produced a list of requirements, presented on pages 53 and 55. This list is the answer to Research Question I. However, the set of requirements is non-exhaustive; it is to be reused, expanded, and refined in future research on automated SBT.

The knowledge underpinning the requirements and claims for automated SBT has been modeled and formalized in an ontology. The ontology distinguishes between knowledge regarding automated SBT (the domain ontology) and knowledge regarding the chosen design solution (the system design ontology). The domain ontology is reusable for alternative design solutions for automated SBT and can be expanded where necessary.

The requirements, claims, and ontology for automated SBT are a starting point for the systematic approach to the design of a system that supports automated SBT in a scientifically grounded knowledge base.

On Personalized Educational Games (RQII, RQIII, and RQIV)

Personalized educational games combine ideas from intelligent tutoring systems and educational games. They establish personalized autonomous training through the enhancement of a game environment with artificial intelligence, such as intelligent agents and planning algorithms.

RQII: What design could possibly satisfy all identified requirements?

The complete design for PEGs, presented in this thesis, consists of the general design, and the additional component designs. This design is also our proposed solution in answer to RQII.

RQIII: Does the PEG design satisfy the identified requirements?

The current design offers the learner scaffolding interventions to adjust the level of challenge and support in realtime (R10). These scaffolding interventions are carried out by the NPCs. As a result, the scaffolding interventions do not disturb the learner's sense of immersion (R12). Moreover, one of the possible scaffolding interventions is to have an NPC perform the action if a learner is unable to do so. As a result, the scenario will always be pushed forward, even if the learner fails to perform some action that was originally planned to be part of the learner's actions; eventually an NPC will perform the action. This enables the independent assessment of multiple events, because the learner is always able to correctly perform the remaining actions

in the scenario (R08). In other words, a mistake early on in the scenario does not affect the learner's chances to perform other parts of the scenario correctly.

The NPCs are semi-autonomous in their behaviors. As a result, scenario plans can be specified at a relatively high level of abstraction. The NPCs are able to dynamically select the most suitable action plan to comply with the constraints specified in the scenario plan. This freedom results in a variety of scenarios based on a given scenario plan (R05).

The current design generates a scenario plan based on a learning goal that is dynamically selected by the learner model, based on the learner's current competencies (R06,R07). In general, the scenario creator is able to specify a large variety of scenario plans based on a given learning goal (R05). The scenario creation procedure employs an ontology, containing an accurate representation of the task domain. This results in representative scenarios (R03). Moreover, by taking the difficulty level into account while constructing the scenario plan, the amount of unnecessary distractions in the scenario plan can be reduced (R09).

Among other things, scenario plans specify a complete action sequence for the learner to perform, based on performance standards specified in an HTN (R01, R02). This offers a possibility to assess the learner's performance (R06).

The assessment of the learner's performance allows for a directed, intelligent dialogue between the reflection component and the learner to reflect on the learner's performance and to explain what went right/wrong (R13, R16).

The instructor is able to cooperate with the scenario creator with the use of the authoring tool (R19). The authoring tool offers the instructor suggestions as to how they might specify a consistent and coherent set of scenario constraints, which increases not only the usability of the tool (R18), but also the quality of the resulting scenarios.

The current design does not (yet) support: the provision of instructions on how to use scaffolds (R11), possibilities for the learner to merely observe correct or erroneous scenario performances, for instance, executed by an NPC (i.e. worked-out vs erroneous examples) (R14, R15), and part-task training scenarios (R17).

Lastly, this research has not investigated how the game engine may determine what parts of the environment should be abstracted (R04). Currently, the scenario plan only prescribes what is required in the environment to afford the action sequence. However, additional information is needed regarding the required level of detail for various aspects of the environment. Future research should investigate whether the ontology can be used to specify what parts of the environment need which level of detail, e.g. the victim's burn needs a highly detailed texture, whereas the kitchen cupboards do not.

RQIV: Does the PEG design bring about the anticipated effects?

Part II presented several studies investigating the effect of the component designs with respect to their associated claims. The results obtained from these studies are reviewed below.

Realtime adjustments in the behavior of the NPCs to adjust the difficulty level of the scenario in real-time resulted in successful adaptations. Moreover, the realtime adjustments in the scenario's difficulty level also resulted in more suitable learning

situations compared to non-adjusted scenarios.

Offering instructors suggestions as to how they might specify a consistent and coherent set of scenario constraints results in a higher usability of the tool and increases the quality of the resulting sets of scenario constraints in terms of 1) representativeness, 2) suitability for the learner's competency level, 3) suitability for the learning goal, and 4) consistency between the learning goal and the selected scenario features.

It proved to be difficult to test the anticipated effects on the learner with the use of the isolated component prototypes. For instance, the scenario generator prototype produces a text-based scenario plan, which is only a description of what the learner is supposed to do, and how the virtual environment and the NPC agents need to be initialized. In other words, the prototype does not allow a learner to actually play the scenario. For that, other components would be required, such as a simulated environment and NPCs. As such, the stand-alone scenario creator prototype does not allow for evaluations of the effects of the scenario plan on the learner's competence development. At this point, we have developed a multi-agent system prototype containing simplified versions of the various agents (i.e. components) and a simple text-based environment. In addition, more elaborate stand-alone component prototypes have been developed, such as the scenario creator. However, these separate components are yet to be integrated into the multi-agent system prototype. In order to investigate the effects of the adopted requirements on the learner, a complete system prototype would be required that integrates the various stand-alone prototypes into one complete system prototype. Once such a prototype has been developed, it would be possible to investigate the remaining effects of the system, for instance by means of use case evaluations as proposed by Westera et al. (2010).

Future refinements of the current design

Based on our research, several refinements of the current design can be suggested.

Human-machine cooperation - Ultimately, the design for automated SBT should benefit from 1) the instructor's expertise when he/she is present, and 2) the automated functions of the PEG in cases where the instructor is absent. Therefore, the current design should be expanded with additional possibilities for cooperation between the human instructor and the machine. The current design offers the instructor the possibility to step in at the scenario creation process, but throughout the thesis several suggestions have been provided for additional ways to involve the human instructor in the training process: 1) adopt various roles in the organization, 2) critique the contents of the learner model, 3) critique scenario plans developed by the automated scenario creator, and 4) select the learning goal selection strategy employed by the learner model. These suggestions should be investigated in future research.

Team training - Currently, the PEG's design supports only individual training, i.e. a single learner interacts with NPCs. However, in case there is an opportunity to play with other humans, e.g. team mates, it would be beneficial if the learner was able to benefit from that. A human player is still more believable and adequate than an artificial one. Therefore, the current PEG design should be expanded with the possibility

for team training support. For this, it would be required to represent team-goals in addition to individual goals (Grosz & Kraus, 1996; Levesque et al., 1990). Moreover, possible conflicts between these two goal-types should be resolved. Examples of team training systems that employ intelligent agents are STEAM (Tambe, 1997) and CAST (Yen et al., 2001; Yen et al., 2006). STEAM allows for selective communication between team members and is able to deal with teamwork failures by reassigning tasks to other team members. CAST endows the agents with a shared mental model, which produces mutual situation awareness. As a result the agents are able to reason about their own situation but also about the status and activities of other team members. CAST allows for dynamic anticipation of teammates information needs supported by the shared mental model. Future research should investigate whether and how these approaches offer directions for a PEG expansion that supports team training.

Task model - The current design employs a static task model: procedures, solutions, and strategies that are considered to be correct are all predefined, either by the developer or by the instructor. In many training domains there are often situations where standard or predefined procedures are not always optimal. To include such procedures in the system, without the need for the instructor to explicitly add all imaginable exceptions into the ontology by hand, the system should be expanded such that it is able to learn new solutions and procedures from the scenarios played by the learners. A possible approach to achieve this would be to employ constraint-based reasoning (Freuder & Mackworth, 1994; Mitrovic, 2010; Mitrovic et al., 2001).

By expanding the task model with constraints that are applicable to acceptable solutions, then this can lead to the automated addition of new acceptable procedures to the task model based on the effects of the applied procedure. If a certain procedure satisfies the specified constraints, it can be added as a (possibly) acceptable solution for the problem addressed in the scenario. In turn, these (possibly) acceptable solutions can be reviewed by instructors before adding them to the set of correct solutions used during assessments. This approach would ease the authoring process for the instructors and designers; it would no longer be required to manually specify all acceptable procedures beforehand.

In an agent-organization norms are used to specify the constraints for acceptable agent behaviors. In the case of a PEG, norms can be used to specify the constraints of acceptable task performance for the learner. However, the question arises how to organize the agents' interactions in the system, such that the norms are actually followed by the agents, or in this case, by the learner. There are two ways to manage norm conformity: regimentation and enforcement (Fornara & Colombetti, 2008; Grossi et al., 2007; Grossi et al., 2010). Regimentation means that it is not possible for an agent to break with the norms, whereas enforcement means that the agent is able to perform actions that lead to norm violations, but the agent receives penalties for this kind of behavior. Because the learner should be able to perform actions that violate the constraints, i.e. make mistakes, regimentation is not a suitable solution. Enforcement, on the other hand, would not only allow for the specification of constraints, but would also provide additional possibilities for performance assessment. Therefore, in the case of a PEG, the agent organization should be expanded

with norm enforcement as to the implement a constraint-based assessment of the learner's performance and the construction of a dynamic task model.

Synthesis

The work in this thesis was inspired by and builds upon the work of many others, and hopefully, in turn, this work will be inspiring others to continue the research that is needed. This thesis aims to contribute to research in the fields of technology-enhanced learning, agent technology, and cognitive engineering. In addition, the knowledge obtained throughout this research project is hoped to be of use to the development of automated SBT and, in the future, to result in an increase of training opportunities for learners.

The introduction presented a short story about Carl and his ICEM (in-company emergency management) team members who participated in one of their scarce training days. Carl ran into several problems that day: 1) His instructor was not able to respond to his personal training need; 2) Due to the chaos and the size of the exercise, no one was able to trace an unforeseen mistake back to its source, not even the instructor; and 3) Carl now has to wait for 6 months until his next training. Below, a story is provided about Carl, who is now working with the PEG.

It is a regular Tuesday evening and Carl is at home in his study room. He starts his computer and runs the PEG for ICEM training, as he does every Tuesday evening. The PEG greets him: "Hi Carl, would you like to resume training?" Carl clicks 'yes'.

The PEG has remembered his previous training sessions and knows exactly what topics he is having difficulty with. The first scenario requires him to deal with a victim that is pale, anxious, sweaty, and complains about chest pains and nausea. Carl is not entirely sure what to do: "It may be a victim in shock, but it could also be a panic attack." He decides to perform the procedure that is suitable for a victim who is having a panic attack: he sits the victim down and starts to instruct the victim on breathing exercises. However, the victim quickly deteriorates as organs start shutting down. An NPC takes over by calling an ambulance. In turn, the NPC lays the victim flat with his feet elevated above the ground and asks Carl to lend him his jacket to keep the victim warm. As it turns out, the victim was in hypovolemic shock due to a collision with a scooter. If Carl had asked the victim, he would have known that the scooter driver took off right before Carl arrived at the scene.

During the reflection stage, the PEG elicits a partial explanation from Carl: Carl understands why he confused the two conditions based on his first impression and comes to the conclusion that he should have looked for additional information to distinguish between the two conditions. However, apparently Carl doesn't know what information he should be looking for. The reflection component provides Carl with the information that he was apparently missing: he should have asked the victim what happened (i.e. a trauma or an anxiety stimulus?) and looked for additional signs of hypovolemic shock, e.g. blue lips and finger nails, weak pulse, abdominal pain and/or swelling. Carl and the PEG conclude that he has a tendency of skipping questioning and immediately starting treatment of the victim. As a result he misses important information to make a proper assessment of the situation.

After practicing with questioning in a couple of training rounds, the PEG moves onto a different learning goal: evacuating an office. During the scenario, Carl fails to instruct one of his team members to go to the far end of the second floor to stand in front of the elevator and keep people from entering it. Because Carl did not perform the right action, one of the NPCs took over his task, but only after a small group of people already took the elevator. At first, Carl is unaware of this mistake. He does not understand how this could have happened. He was under the impression that one of his colleagues was responsible for blocking the elevator on the second floor. However, during the reflection stage, the PEG elicits this misunderstanding from Carl and explains to Carl that guarding the elevator on the second floor is part of his tasks.

After the evacuation scenario, it is time for Carl to end the session. He closes the PEG with a feeling of satisfaction and fulfillment. Next time something like this happens at the office, he will be prepared to do what is necessary.

Appendices

APPENDIX A

Interviews with Domain Experts - an Overview

Throughout the research presented in this thesis, we have conducted multiple interviews with domain experts, people who regularly train (and entertain) people by means of storylines within simulated environments. These interviews consisted of eight individual and five joint interviews with a total of twenty-four instructors in three safety domains, three health care domains, and one organizer of live action role plays in the entertainment domain. In addition, three training sessions in three different domains were attended. The training domains investigated in these interviews are described below.

Safety and defense

Three of the training domains addressed within the interviews fall within the scope of the Dutch safety department (e.g. navy, military, police). These domains have several characteristics that distinguish them from the other domains involved in the interviews. First of all, safety and defense domains usually have a fairly high instructor-to-student ratio (about 2:1).

Furthermore, practical training sessions within these domains regularly employ virtual environments. Trainees are located in their own cubicle, watching monitors displaying for instance the radar and mapping tools. The monitors are updated with the events happening in the virtual environment. The staff members are located together in a place where they can consult with each other to coordinate the planning of responses to the trainee's decisions and/or the introduction of new events.

This set-up has two big advantages: 1) staff members are able to improvise and adapt the scenario immediately and in a coordinated fashion whenever they find it expedient to do so, and 2) there are no real risks involved when trainees make a mistake. Because of this last advantage, the training curricula in these domains are almost entirely comprised of practical training sessions; trainees are thrown in at the deep end, subsequently learning by reflecting on their performance. However, a virtual environment also means that it is easier to overlook the actual body language and behavior of the trainee. One instructor explains how a trainee responded adequately to all the events in the scenario, yet his coach - who was sitting in the cubicle with the trainee - noticed that he hardly ever watched his radar images. Because of this the trainee was relying on old information, so when the staff decided to introduce some unforeseen vehicle movements, the trainee failed to notice it in time.

Command Center Officer (CCO)

The CCO is responsible for the tactical command of navy ships at sea. Training scenarios are fixed with respect to the map, the number of unidentified vehicles around the convoy (controlled by staff members), their initial location, the goal of the mission, and possibly available resources, all of which is represented in the virtual environment.

During training the CCO trainee attempts to reach the mission's goals, all the while trying to determine the identity and movement of all vehicles within range, and adapting the tactical navigation of the convoy based on that information.

Information about this domain is based on one joint open interview with two

instructors and one attended training session.

Helicopter Directive Operator (HDO)

An HDO commands a helicopter pilot from aboard a ship. The HDO has full control over the flight path of the pilot. The pilot is not allowed to make any navigational decisions, and merely carries out the commands received from the HDO.

During training, trainees practice in cubicles, where they plot the surroundings of the ship and the helicopter. Based on these plots and the mission's goals, they plot the route for the pilot and provide directions for the pilot in a fairly restricted communication language. Trainees usually have individual training sessions with at least one, but often more instructors to guide them. The helicopter pilot is usually played by a staff member, responding to the instructions and changing circumstances in the environment. The training adheres to a strict curriculum with weekly objectives and assessments that the trainee has to pass to be allowed to continue the training.

Information about this domain is based on one joint semi-structured interview with four Helicopter Directive Operator (HDO) instructors of the Dutch Royal Navy. The topic of this interview was the preparation and execution of scenarios, and the long-term individual assessment of the trainee's competence development in the training program.

State Police Platoon Officer

A state police platoon officer (SPPO) usually resides in a van on-site of the event. The SPPO tries to obtain a clear impression of the situation and events in the actual environment by communicating with his group officers through portophones.

Information about this domain is based on an attended training session that addressed crowd and safety control during a protest. The scenario also involved a group of people coming from a festival looking for trouble, and a group of counter protesters. During this training session the SPPO was required to command the group officers to e.g. move locations, charge, or communicate with one of the leaders of the various crowds in the town.

The virtual (game) environment consisted of a small town in which the three crowds moved through the streets. All crowds were controlled by staff members. Even though the platoon officer was the one in training, five group officers were also present. Each group officer controlled a group of state policemen (virtual characters). The group officers navigated their group through the game environment (VBS2), communicated with the crowd leaders, and controlled the lineup of their group, e.g. rest, charge, etc. They carried out the commands provided by the platoon officer and communicated their observations in the game world to the platoon officer.

As in real life, the platoon officer was located in a closed off room, drawing on maps to get a clear view of the situation in the streets, and commanding the group officers based on this view.

Emergency health care

In contrast to the safety and defense domains discussed above, the emergency health care domains generally do not employ virtual environments during training. Instead, they rely on real life role play simulations. This has several important implications for training, which will be discussed below.

The first implication of real-life role-play is that there is often no possibility for control over the scenario in realtime. In some cases, role players communicate with the instructor during scenarios through signaling to control the scenario in realtime, but this type of control is obviously limited to the timing of planned events. Instructors should provide the actor with very clear instructions in advance of the scenario, because there is no possibility to refine those instructions once the scenario starts playing. In some cases, instructors even go as far to prepare different types of responses for the actor depending on the choices of the trainee, resulting in multiple prepared storylines following from a decision tree. However, improvisation is mostly difficult, depending mostly on the quality of the collaboration between the actor and the instructor.

The second implication has to do with the limited capabilities of the actors. First of all, role players cannot keep repainting themselves for new wounds and different illnesses. This simply takes way too much time. Secondly, some situations are extremely intense for the role players to impersonate. The most striking example is probably a shock: When a person is in shock, his or her organs are not getting enough blood or oxygen. If untreated, this can lead to permanent organ damage or death. This emergency is very hard for medical actors to enact since it requires them to stop breathing, relax their heartbeat to a minimum, stare, feign nausea, and feign to go in and out of consciousness. Because of the intensity of enacting such emergencies, the instructor has to remain alert about the well-being of the role player. In some cases the instructor may even need to intervene to protect the role player.

Thirdly, because the role-players are often expensive training props, instructors want to make good and efficient use of them. As a result, the role-playing exercises are usually preceded by a theoretical presentation, starting with a silent demonstration, followed by an explained demonstration, followed by a trainee explaining the action plan or procedure to the instructor, followed by the role-playing exercise performed by the trainees (usually in teams of two). The silent demonstration is regarded to be of special importance since the instructor can show the trainees how he/she interacts with the victim. Role-playing exercises generally do not only serve the purpose of hands-on practice but also as a test to check whether the trainees have grasped the essentials.

First Aid

First Aid training usually involves class sizes between five and fifteen students, and one or two instructors providing the students with theoretical presentations and practical role-plays involving medical actors.

Practice is done in the form of role playing with the help of specially trained actors. These actors mimic a variety of injuries. Medical actors are a relatively scarce resource and only in exceptional cases will there be more than one available

for the entire group. Furthermore, the instructors often do not know beforehand who the actor will be. Training can take place at a regular training center or on site at a client.

Information about this domain is based on six individual and one joint interview with five instructors, resulting in a total of eleven instructors. Two of the individual interviews concerned the preparation and execution of scenarios, and the long-term individual assessment of the trainee's competence development in the training program. The other four individual interviews addressed the features instructors would like to have control over when authoring a training scenario. The joint interview with five instructors was concerned with the questions 'What makes a good high-quality scenario?' and 'What should a Personalized Educational Game offer a learner and an instructor respectively?'.

In-company Emergency Management (ICEM)

ICEM training classes regularly consist of ten to twenty trainees being taught by one or two instructors. During the First Aid topics there is usually also a medical actor present to participate in the role-playing exercises.

ICEM entails the application of First Aid, firefighting, and evacuation procedures, all of which are performed by a trained team of company employees. They remain in charge until the incident is sufficiently handled, or when the official emergency services arrive (i.e., ambulance, firemen, police). The First Aid training part is not as detailed nor thorough compared to regular First Aid training. Yet most ICEM training instructors are also First Aid instructors.

Information about this domain is based on one joint interview with six instructors. This interview was mainly concerned with the operations during ICEM practical training and the way instructors control the scenario in realtime.

Emergency Care

Information about this domain is based on one individual interview and one attended training session. The interview was mainly concerned with the usual course of events during practical training sessions and the way the instructor prepared the cases covered in class. During the attended class there were twelve trainees present and they were taught by one instructor.

Emergency care training addresses the tasks of medics in the emergency room. It makes use of comprehensive high-tech puppets that display different breathing patterns, heart rhythms, respiratory sounds, etc. A medical team follows the required procedures to save the patient's life and transfers the patient to the proper hospital department.

During the attended training session, the instructor covered four cases with the group. The students were required to switch places regularly, filling in different positions at the examination table, e.g. examining the patient, intubating, performing CPR, providing the shots, or handling the automated electronic defibrillator (AED). Students who did not participate in the scenario were encouraged to observe the events closely and comment on the choices of the performing student team afterwards. In all of the cases, the scenario started out with the instructor playing the part of the patient until the patient fainted. Once the patient fainted, the student team switched to working with the doll to carry on with possible intubation, CPR,

and/or use of the AED.

Entertainment

One last domain concerning the interviews regarding the control over scenarios and live role-play lies in the entertainment domain. Since this group of domains only contains one domain, we will go straight to the specifics of that domain below.

Live Action Role Play (LARP)

A LARP event is usually prepared by a LARP organization. Events usually last between one or two days and vary in size from twenty people to over three hundred. The event preparation generally consists of a back-story of the fantasy-world in which the event is staged. The most important requirement of a LARP event is that the world is believable, since you want people to engage in suspension of disbelief. There are rules for LARP events, described in a booklet, about the practice of spells, potions, and so on. However, these rules are not always followed (which is not considered to be a problem) nor are they consistent or closed.

Usually, the organization prepares several plotlines, which are carried by non-player characters, i.e. characters who are trained players and actively try to bring about the prepared plotlines by inviting players to join in on them. Moreover, non-player characters often reveal hints or information that helps the players to carry the plotline one step further. The organization of the event is divided in teams. Each team is responsible for a different division of players (located in various base camps in the game world). The head of organization is in direct contact with the team leaders through portophones. On most occasions, this allows for an accurate view of the current situation in the game world. Non-player characters are instructed as they go into the game world and consult with the team leaders when they have the chance (e.g. when they are not interacting with the players).

Not all players in a LARP event are participating in the prepared plotlines. In LARPing people very much rely on their own fantasy. Players are also allowed to add their own interpretations and plotlines to the game, as long as they are coherent with the backstory of the crafted world.

In some cases, the plotlines prepared by the organization fail to be established. This is usually because the non-player characters need to run into the players that actually want to participate in the plotlines prepared by the organization. In addition, the players who want to join in on those plotlines also need to find the *key characters* of the plotlines. So if the non-player characters cannot find the players who are interested in joining the prepared plotlines or vice versa, the plotlines are not established. In other cases the storyline is experienced as thin or boring, and people leave the prepared plotlines to further develop their own character through interactions with other players, or by fantasizing their own personal stories.

Information about this domain is based on a single semi-structured interview about the preparation of LARP events and how these events are controlled in realtime to ensure the establishment of the prepared storylines.

APPENDIX B

Interview Guide 1

This appendix presents an English translation of the interview guide that was employed during the semi-structured interviews conducted for the purpose of developing the scenario creator (described in Chapter 8).

Start of the interview

The instructors received the following introduction to the topic of the interview:

'Scenario-based training is a training form during which a learner participates in a training scenario within a controlled environment. Scenarios are generally representative for the types of situations the learner will encounter in his future profession. Traditionally, scenario-based training entails real-life role-plays with trained actors, classmates, or instructors. Nowadays, it becomes more and more common to stage scenarios within a virtual environment (on a computer).

A training session is most effective when it can be adapted to the specific characteristics of the learner. However, this requires time and effort of the instructor, both during and in advance of training. Therefore, a new line of research is concerned with the personalization of scenario-based training by means of a computer program called 'the director agent'. This computer program is meant to support the instructor. The instructor is able to specify a scenario, and possible interventions, in advance of training. During training, the director agent uses this information to present the learner with an adaptive scenario.

The goal of this interview is to find out how an instructor directs and personalizes training sessions. How, when, and why does an instructor intervene? From the answers to these questions we want to get a clear image of the deliberation process of experienced instructors. This information will be used in the design of the director agent. Later on in the research project, a prototype will be developed and evaluated.

If you incline to participate in this research, you will be asked several questions about the way you prepare, control, and evaluate a training session. The interview will last about one-and-a-half hour and will be recorded with a voice-recorder. We will also be taking notes during the interview. The data collected during the interviews will be used for publication in scientific journals and reports. Your personal information will not be used and your anonymity will be guaranteed under all circumstances. The notes and voice recordings will only be analyzed by the research project team.

Please know that you are allowed to stop the interview at all times.'

Information for the interviewer

The questions are organized into six parts. Each of these parts have their own objective. We have decided to structure the interview according to the way a scenario-based training is structured: we start with the preparation, continue with the training session itself, and end with the evaluation of the training session. By following this structure, we hope to offer the instructors a clear overview of the interview. This structure is explained to the participant before starting the interview. During the interview it is also possible to divert from this structure whenever the interviewer

deems it necessary to do so in order to maintain a natural flow in the conversation. During the interview it is important to keep in mind what the objective of a particular part of the interview is.

The main objective of the interview is: 'How does an instructor personalize and control scenario-based training?'

The objectives for the different parts of the interview are:

Goal 0 - Background information about the participant
Goal 1 - How does the participant prepare the training session?
Goal 2 - How does the participant intervene during a scenario?
Goal 3 - How does the participant provide the learner with feedback?
Goal 4 - How does the participant evaluate the training session?
Goal 5 - Closing, are there additional comments or questions?

Background information about the participant

Estimated duration: 10 min

1. For how long have you been working within this field?
2. For how long have you been an instructor?
3. For how long have you been working with scenario-based training?
4. What is your opinion about the usefulness of SBT?
5. Have you also worked as an instructor within other fields or domains? Which ones?
   (a) Are there any notable differences in training between those fields?
6. Have you ever trained other instructors?
7. What would you say is the most important part or feature of training?
8. What is the average size of the groups you train?

How does the participant prepare the training session?

Estimated duration: 20 min

1. How do you usually prepare a training session?
2. How do you select a suitable learning goal?
   (a) In what way do you take the competency level of the group into account?
3. How do you go about when creating a suitable scenario?
   (a) What knowledge do you need to be able to do so?
   (b) Where do you start when creating a scenario?
   (c) Are there any building blocks, or other types of reusable scenario features that help you to construct a scenario?
   (d) To what extent do you create a scenario especially for a particular group of learners or even a single learner?
4. How do you determine the difficulty level of the scenario?
5. To what extent do scenarios grow in complexity over the course of training?
   (a) How is this accomplished?
6. Your colleague mentioned that scenarios are sometimes used to keep the learner alert. Are such scenarios generally specified in advance or are they improvised?
   (a) How do you go about creating/planning such scenarios?
7. In what way do you prepare the scenario with the role-players or actors?

   (a) Do you ever provide an actor or role-player with conditional instructions? Can you give an example?
8. Do you prepare multiple scenarios for one training session?
   (a) How do you determine in what order the scenarios should be practiced?
   (b) To what extent is it possible to rearrange the sequence during training?
      i. If you would adjust the sequence, what would be the reasons to do so?

How does the participant intervene during a scenario?

Estimated duration: 25 min

1. What are the things you pay attention to during the scenario enactment?
   (a) How do you keep track of these things?
2. Let's say you created a scenario, however, as the scenario progresses, the learner makes some unforeseen decisions. To what extent do you try to steer the scenario back into the desired direction?
   (a) How would you go about doing this?
   (b) In what situation would you restrain yourself from 'fixing' the scenario?
   (c) To what extent do you provide learners with hints? Do you use different types of hints? If so, which ones?
3. Under what circumstances would you pause the scenario?
   (a) What would be your considerations in such a situation?
   (b) In what situations is the learner allowed to pause the scenario?
4. To what extent do you give the learner feedback?
   (a) Do you always give feedback? If not, when would you restrain yourself from providing feedback?
   (b) Is the feedback you provide implicit or explicit? In other words, do you specifically tell the learner what went wrong and what went well and why so? Or do you only tell him what the outcomes of the scenario are? For example, you might only tell the learner: "The victim is ok for now but a week later he visits his general practitioner because he has a large infection on his back.", with no further explanation.
5. To what extent do you influence the difficulty level while the scenario is playing?
   (a) To what extent do you influence the storyline while the scenario is playing?
   (b) To what extent do you influence the behavior of the actors or role players while the scenario is playing?
6. To what extent do learners black out or get stressed out during training?
   (a) How do you handle such situations?
   (b) What would you say are the most obvious signals showing that a learner is having a difficult time?
7. Are there ways in which you would influence the scenario that we haven't discussed yet?
   (a) Under what circumstances do you exercise this influence?
8. What are the differences in guidance during training for beginners and more advanced learners?

9. What are the most influential signals for you to intervene in a scenario?
10. Under which circumstances would you decide not to intervene?
    (a) What would be the reason for that?
11. How do you decide on the type of intervention?
    (a) What knowledge do you use to make such a decision?
12. Do you ever run into problems and if so how do you solve them?
13. Can you provide me with an example of a situation where you would have wanted to intervene in some way but where this was not possible?
    (a) Why was it not possible to intervene?
    (b) In what way would you have wanted to intervene?
    (c) Why would you have wanted to intervene?
14. When there is a large exercise where there are multiple situations at various locations. How do you keep a clear view of what's going on?
    (a) To what extent do exercise leaders tune in on one another's roles and tasks during these events?
    (b) How are such exercises coordinated?

How does the participant provide the learner with feedback?

Estimated duration: 15 min

1. How do you determine the learner's progress?
2. Some parts of the task performance take place in the head of the learner. How do you assess the learner's thought process?
3. How do you determine whether the learner has accomplished or obtained the learning goal?
    (a) If the learner did not obtain the learning goal, how do you determine whether this is caused by an unsuitable scenario or by the learner's incapacity to perform well?
    (b) If the blame is on the learner, what would be your course of action?
    (c) If the scenario is to blame, what would be your course of action?
    (d) How do you evaluate scenarios?

How does the participant evaluate the training session?

Estimated duration: 15 min

1. What knowledge do you take with you to the next training session?
    (a) How do you use this knowledge?
2. How do you evaluate the usefulness of a scenario?
    (a) How would you use the outcomes of such an evaluation to improve your own teaching skills?
3. When is a scenario regarded to be unsuccessful?
    (a) How do you cope with that?
4. What do you do with learners who are obviously motivated, but are simply not capable of keeping up with the pace of training?
    (a) Do you have any suggestions to deal with such situations in a better way?
5. To what extent can a scenario be used to transfer knowledge or information in an implicit manner? By this we mean: without preparing the learner with books or presentations that explicitly disclose the procedures and underlying knowledge before the scenario is practiced.

(a) In case of implicit learning from scenarios, would you say that it is required to provide explicit feedback afterwards?
(b) Do learners spontaneously have a sense of what went wrong during a scenario?

Closing, are there additional comments or questions?

Estimated duration: 5 min

Are there any additional comments you would like to make? Are there topics left to discuss?

Extra: Design

1. What type of features would you like to see in a PEG?
   (a) For instance during preparation?
   (b) During the training session?
   (c) During evaluation and reflection?
2. What features would you say are indispensable?
3. What would you say are the criteria for a successful design of a PEG?
4. Your colleague mentioned that you already employ various forms of e-learning.
   (a) How are these technologies currently employed?
   (b) What is your experience with these tools?

APPENDIX C

Interview Guide 2

This appendix presents an English translation of the interview guide that was employed during the semi-structured interviews conducted for the purpose of developing the authoring tool (described in Chapter 9).

Start of the interview

At the beginning of the interview, the following steps were followed before the actual interview started.

A. Thank the instructor for his/her participation and describe the goal of the interview. Emphasize that the interview will focus on First Aid training, as is currently employed in First Aid courses with LOTUS actors.
B. Describe the structure of the interview.
C. Ask permission for audio recording during the interview.
D. Inform the participant about his/her rights.
E. Invite the participant to sign the consent form.
F. Note the name, age, gender, years of experience as an instructor, and experience with serious games and/or technology enhanced learning.

Discussion of five cases

The interview started with the presentation of five cases of scenario executions. The participants were asked to read the cases, study them, and imagine themselves being the instructor during the presented case. Below, one of the cases from the interview is shown as an example:

Figure C.1: Case 1 - a young woman gets a nosebleed in the supermarket

Case 1: The learner is waiting in line at the supermarket when a young woman gets a nosebleed (see Figure C.1). It is bleeding quite heavily and the woman looks pale. Furthermore, it is relatively crowded in the supermarket. The learner seems stressed and slightly aggressive as he instructs the people standing around the victim to make room. This results in commotion among the bystanders. The learner asks for a tissue. He receives the tissue from one of the bystanders and gives it to the victim and instructs her to push the tissue to her nose. After telling the victim to hold her head backwards, the learner looks around as if unsure what to do next.

Case 2 - 5: These cases have been omitted from the appendix for reasons of conciseness.

After each scenario, the participant was asked to explain what he/she would do by means of the following questions:

1. Do you find this a realistic scenario? Can you explain your answer?
2. Can you summarize what is happening in the scenario?
3. Would you like to intervene during the scenario? At what particular point in time? Please explain.
4. During your own training sessions, are there moments like this one where you would like to intervene, but intervening is difficult, or impossible? Please explain your answer.

Question round I

First question round concerning current role as a teacher:

1. What First Aid topics do you teach?
2. How would you describe the groups of learners that you teach?
3. How do you decide what to teach and when?
4. To what extent are you familiar with pedagogy or didactic methods?
5. Wat is your educational background?
6. What is your background as an instructor or teacher?
7. Can you describe yourself as a teacher?
   (a) What type of role would you say is the most accurate description of your role during scenarios: a facilitator role or a teacher/lecturer role?
8. What do you think is the most effective approach to learning and/or teaching?
9. How do you aim to bring about learning among students?
10. What kind of teaching methods do you apply?

Question round II

The second question round concerned the use of a simulation game in teaching. Two video-based examples of educational games for First Aid were displayed to explain the concept of educational games to the instructors. Subsequently, the instructors were provided with two examples of how they could be offered a way to influence the content of a game, i.e. the possibility to place objects in the virtual environment or to select the learning goal(s) for the next scenario.

After this introduction, the instructors were asked the following questions:

1. Would you appreciate the use of an educational game and how would you want to use it in your own course?
   (a) During preparation?
   (b) During a lesson?
   (c) Teaching in groups or individual training?
   (d) During assessment?
   (e) During reflection?
2. What advantages/disadvantages come to mind when thinking about teaching

and learning with the use of an educational game?
3. How do you imagine educational games would fit into the existing curriculum?
   (a) If the instructor cannot imagine educational games to fit in with the current curriculum, what changes would be necessary to make the fit? (changes to the game as well as changes to the curriculum)
4. Would the game change your role as an instructor, and if so, how come and in what way?
5. What aspects of the game would you like to have control over?
   (a) Help the instructor if this is a difficult question by providing examples and/or moving on to the next question

Discussing a list of controllable game features

The last part of the interview concerned the examination of the following list of controllable game features:

- Select the setting of a scenario (e.g. the mall, a school)
- Select the amount, timing, and type of feedback the learner receives
- Select the amount, timing, and type of support the learner receives
- Select the tasks the learner is to perform or the skills the learner needs to practice during the scenario.
- Select and customize the virtual characters (behavior/injury)
- Manually control the behavior of the virtual characters
- Select the difficulty level of the scenario
- Select the way a learner receives a score
- Manipulate the sequence of events during a scenario
- Choose the number of tasks the learner needs to accomplish during the scenario
- Offer the learner the possibility to watch someone else perform a task correctly
- Determine reflection moments during scenarios
- Pause or abort a scenario
- Construct the set of actions a learner is able to choose from during a scenario
- Allow for multiple learners collaborating in the same scenario (multi-player games)
- Create a test to present to the learner after a certain scenario
- Have access to an overview of the student's past performances

For each of the items, the instructor was asked whether or not he/she would like to see this feature in an educational game. Thereafter, the instructor was asked to rank the selected features with regard to their relevance or priority. The instructor was also asked to answer the following questions:

1. Can you comment on each of these features regarding their relevance and/or importance?
2. Do you feel there are other important features missing from this list?
3. Do you have any additional remarks/comments/questions related to this topic?

The interview was concluded by thanking the instructor once more for his/her participation and asking him/her whether we could contact him/her in the future in cases where First Aid instructors were wanted/required?

Bibliography

[1] Abaci, T. & Thalmann, D., "Planning with smart objects", in: *Conference on Computer Graphics, Visualization and Computer Vision*, 2005, pp. 25–28.

[2] Abuhamdeh, S. & Csikszentmihalyi, M., "The importance of challenge for the enjoyment of intrinsically motivated, goal-directed activities", *Personality and Social Psychology Bulletin*, vol. 38, no. 3, 2012, pp. 317–330.

[3] Albert, D. & Lukas, J., *Knowledge Spaces: Theories, Empirical Research, and Applications*, Psychology Press, 1999.

[4] Aleven, V., "Rule-based cognitive modeling for intelligent tutoring systems", in: ed. by Nkambou, R., Bourdeau, J. & Mizoguchi, R., vol. 308, Studies in Computational Intelligence, Springer Berlin / Heidelberg, 2010, pp. 33–62.

[5] Aleven, V. & Koedinger, K. R., "Limitations of student control: do students know when they need help?", in: *Conference on Intelligent Tutoring Systems*, Springer, 2000, pp. 292–303.

[6] Aleven, V. & Koedinger, K. R., "The need for tutorial dialog to support self-explanation", in: *AAAI Fall Symposium*, 2000, pp. 65–73.

[7] Aleven, V. & Koedinger, K. R., "An effective metacognitive strategy: learning by doing and explaining with a computer-based cognitive tutor", *Cognitive Science*, vol. 26, no. 2, 2002, pp. 147–179.

[8] Aleven, V., Stahl, E., Schworm, S., Fischer, F. & Wallace, R., "Help seeking and help design in interactive learning environments", *Review of Educational Research*, vol. 73, no. 3, 2003, pp. 277–320.

[9] Aleven, V., Ogan, A., Popescu, O., Torrey, C. & Koedinger, K., "Evaluating the effectiveness of a tutorial dialogue system for self-explanation", in: *Conference on Intelligent Tutoring Systems*, Springer, 2004, pp. 443–454.

[10] Ames, C. & Archer, J., "Achievement goals in the classroom: students' learning strategies and motivation processes", *Journal of Educational Psychology*, vol. 80, no. 3, 1988, pp. 260–267.

[11] Andersen, E., Liu, Y.-E., Snider, R., Szeto, R. & Popović, Z., "Placing a value on aesthetics in online casual games", in: *Conference on Human Factors in Computing Systems*, 2011, pp. 1275–1278.

[12] Anderson, J. R. & Bower, G. H., "A propositional theory of recognition memory", *Memory & Cognition*, vol. 2, no. 3, 1974, pp. 406–412.

[13] Annett, J., "Hierarchical task analysis", in: *Handbook of Task Analysis for Human-Computer Interaction*, ed. by Diaper, D., CRC, 2004.

[14] Annett, J. & Duncan, K. D., *Task analysis and training design*, ERIC, 1967.
[15] Appleton, K. & Lovett, A., "GIS-based visualisation of rural landscapes: defining 'sufficient' realism for environmental decision-making", *Landscape and Urban Planning*, vol. 65, no. 3, 2003, pp. 117–131.
[16] Atkinson, J. W., "Towards experimental analysis of human motivation in terms of motives, expectancies, and incentives", *Motives in Fantasy, Action and Society*, 1958, pp. 288–305.
[17] Atkinson, R. C. & Shiffrin, R. M., "Human memory: a proposed system and its control processes", *The Psychology of Learning and Motivation*, vol. 2, 1968, pp. 89–195.
[18] Atkinson, R. K., Derry, S. J., Renkl, A. & Wortham, D., "Learning from examples: instructional principles from the worked examples research", *Review of Educational Research*, vol. 70, no. 2, 2000, pp. 181–214.
[19] Aylett, R. & Cavazza, M., "Intelligent virtual environments-a state-of-the-art report", in: *Eurographics Workshop on Animation and Simulation*, vol. 3, 2001.
[20] Baader, F., "Logic-based knowledge representation", in: *Artificial Intelligence Today*, Springer, 1999, pp. 13–41.
[21] Bach, R., *The bridge across forever*, Pan Macmillan, 1984.
[22] Bakkes, S., Tan, C. T. & Pisan, Y., "Personalised gaming: a motivation and overview of literature", in: *Australasian Conference on Interactive Entertainment: Playing the System*, ACM, 2012.
[23] Baldwin, T. T. & Ford, J. K., "Transfer of training: a review and directions for future research", *Personnel Psychology*, vol. 41, no. 1, 1988, pp. 63–105.
[24] Bandura, A., *Self-efficacy: The Exercise of Control*, Worth Publishers, 1997.
[25] Bandura, A., *Self-regulation of Motivation and Action through Goal Systems*, Springer, 1988.
[26] Bangor, A., Kortum, P. T. & Miller, J. T., "An empirical evaluation of the system usability scale", *International Journal of Human-Computer Interaction*, vol. 24, no. 6, 2008, pp. 574–594.
[27] Bangor, A., Kortum, P. & Miller, J., "Determining what individual sus scores mean: adding an adjective rating scale", *Journal of Usability Studies*, vol. 4, no. 3, 2009, pp. 114–123.
[28] Bates, J., "Virtual reality, art and entertainment", *Presence*, vol. 1, no. 1, 1992, pp. 133–138.
[29] Behrens, T. M., Hindriks, K. V. & Dix, J., "Towards an environment interface standard for agent platforms", *Annals of Mathematics and Artificial Intelligence*, vol. 61, no. 4, 2011, pp. 261–295.
[30] Belsky, S., *Making ideas happen: Overcoming the obstacles between vision and reality*, Penguin, 2010.
[31] Boekaerts, M., "Self-regulated learning: a new concept embraced by researchers, policy makers, educators, teachers, and students", *Learning and Instruction*, vol. 7, no. 2, 1997, pp. 161–186.
[32] Bolter, J. D., *Turing's man: Western culture in the computer age*, UNC Press, 1984.

[33] Bopp, M., "Didactic analysis of digital games and game-based learning", *Affective And Emotional Aspects of Human-Computer Interaction: Game-based And Innovative Learning Approaches*, vol. 1, 2006, pp. 8–37.

[34] Bratman, M. E., *Intention, Plans, and Practical Reason*, Cambridge University Press, 1999.

[35] Brown, A. L., Ash, D., Rutherford, M., Nakagawa, K., Gordon, A. & Campione, J., "Distributed expertise in the classroom", in: *Distributed Cognitions: Psychological and Educational Considerations*, ed. by Salomon, G., Cambridge, UK: Cambridge University Press, 1993, pp. 188–228.

[36] Brown, J. S., Collins, A. & Duguid, P., "Situated cognition and the culture of learning", *Educational Researcher*, vol. 18, no. 1, 1989, pp. 32–42.

[37] Brusilovsky, P. & Millán, E., "User models for adaptive hypermedia and adaptive educational systems", in: *The Adaptive Web*, Springer-Verlag, 2007, pp. 3–53.

[38] Bull, S. & Britland, M., "Group interaction prompted by a simple assessed open learner model that can be optionally released to peers", in: *Workshop on Personalisation in E-Learning Environments at Individual and Group Level*, 2007.

[39] Burke, R. D., Hammond, K. J. & Yound, B., "The FindMe approach to assisted browsing", *IEEE Expert*, vol. 12, no. 4, 1997, pp. 32–40.

[40] Butler, D. L. & Winne, P. H., "Feedback and self-regulated learning: a theoretical synthesis", *Review of Educational Research*, vol. 65, no. 3, 1995, pp. 245–281.

[41] Cairncross, S. & Mannion, M., "Interactive multimedia and learning: realizing the benefits", *Innovations in Education and Teaching International*, vol. 38, no. 2, 2001, pp. 156–164.

[42] Camp, G., Paas, F., Rikers, R. & Van Merriënboer, J., "Dynamic problem selection in air traffic control training: a comparison between performance, mental effort and mental efficiency", *Computers in Human Behavior*, vol. 17, no. 5, 2001, pp. 575–595.

[43] Campbell, D. J., "Task complexity: a review and analysis", *The Academy of Management Review*, vol. 13, no. 1, 1988, pp. 40–52.

[44] Cannon-Bowers, J., Burns, J., Salas, E. & Pruitt, J., "Advanced technology in scenario-based training", in: *Making Decisions Under Stress*, ed. by Cannon-Bowers, J. & Salas, E., APA, 1998, pp. 365–374.

[45] Cap, M., Heuvelink, A., Van den Bosch, K. & Van Doesburg, W., "Using agent technology to build a real-world training application", in: *Agents for Games and Simulations II*, Springer, 2011, pp. 132–147.

[46] Card, O. S., *Ender's game*, Macmillan, 2008.

[47] Carroll, J. M., "Five reasons for scenario-based design", *Interacting with Computers*, vol. 13, no. 1, 2000, pp. 43–60.

[48] Carroll, J. M. & Rosson, M. B., "Getting around the task-artifact cycle: how to make claims and design by scenario", *Transactions on Information Systems (TOIS)*, vol. 10, no. 2, 1992, pp. 181–212.

[49] Carroll, J. M. & Rosson, M. B., "Design rationale as theory", *HCI Models, Theories and Frameworks: Toward a Multidisciplinary Science*, 2003, pp. 431–461.
[50] Charsky, D., "From edutainment to serious games: a change in the use of game characteristics", *Games and Culture*, vol. 5, no. 2, 2010, pp. 177–198.
[51] Chen, J., "Flow in games (and everything else)", *Communications of the ACM*, vol. 50, no. 4, 2007, pp. 31–34.
[52] Chi, M. T., Bassok, M., Lewis, M. W., Reimann, P. & Glaser, R., "Self-explanations: how students study and use examples in learning to solve problems", *Cognitive Science*, vol. 13, no. 2, 1989, pp. 145–182.
[53] Chi, M. T., De Leeuw, N., Chiu, M.-H. & LaVancher, C., "Eliciting self-explanations improves understanding", *Cognitive Science*, vol. 18, no. 3, 1994, pp. 439–477.
[54] Chi, M. T., Siler, S. A., Jeong, H., Yamauchi, T. & Hausmann, R. G., "Learning from human tutoring", *Cognitive Science*, vol. 25, no. 4, 2001, pp. 471–533.
[55] Clark, R. E. & Estes, F., "Cognitive task analysis for training", *International Journal of Educational Research*, vol. 25, no. 5, 1996, pp. 403–417.
[56] Clark, R. C. & Mayer, R. E., "Learning by viewing versus learning by doing: evidence-based guidelines for principled learning environments", *Performance Improvement*, vol. 47, no. 9, 2008, pp. 5–13.
[57] Cohen, M. S., Freeman, J. T. & Thompson, B. B., "Training the naturalistic decision maker", *Naturalistic decision making*, 1997, pp. 257–268.
[58] Conati, C. & Manske, M., "Evaluating adaptive feedback in an educational computer game", in: *Intelligent Virtual Agents*, Springer, 2009, pp. 146–158.
[59] Conati, C. & VanLehn, K., "Toward computer-based support of meta-cognitive skills: a computational framework to coach self-explanation", *International Journal of Artificial Intelligence in Education (IJAIED)*, vol. 11, 2000, pp. 389–415.
[60] Connolly, T. M., Boyle, E. A., MacArthur, E., Hainey, T. & Boyle, J., "A systematic literature review of empirical evidence on computer games and serious games", *Computers & Education*, vol. 59, no. 2, 2012, pp. 661–686.
[61] Cook, R. & Kay, J., *The justified user model: a viewable, explained user model*, Basser Department of Computer Science, University of Sydney, 1994.
[62] Corbalan Perez, G., Kester, L. & Van Merriënboer, J. J. G., "Selecting learning tasks: effects of adaptation and shared control on learning efficiency and task involvement", *Contemporary Educational Psychology*, vol. 33, no. 4, 2008, pp. 733–756.
[63] Corbalan Perez, G., Kester, L. & Van Merriënboer, J., "Combining shared control with variability over surface features: effects on transfer test performance and task involvement", *Computers in Human Behavior*, vol. 25, no. 2, 2009, pp. 290–298.
[64] Corbalan Perez, G., Kester, L. & Van Merriënboer, J. J. G., "Towards a personalized task selection model with shared instructional control", *Instructional Science*, vol. 34, no. 5, 2006, pp. 399–422.

[65] Cortina, J. M., "What is coefficient alpha? An examination of theory and applications", *Journal of applied psychology*, vol. 78, no. 1, 1993, pp. 98–104.

[66] Cowley, B., Charles, D., Black, M. & Hickey, R., "Toward an understanding of flow in video games", *Computers in Entertainment*, vol. 6, no. 2, 2008, pp. 1–27.

[67] Csikszentmihalyi, M., *Flow: The Psychology of Optimal Experience: Steps Toward Enhancing the Quality of Life*, Harper Collins, 1991.

[68] Cunningham, D & Duffy, T, "Constructivism: implications for the design and delivery of instruction", *Handbook of Research for Educational Communications and Technology*, 1996, pp. 170–198.

[69] Dastani, M., "2apl: a practical agent programming language", *Journal of Autonomous Agents and Multi-Agent Systems, Special Issue on Computational Logic-based Agents*, vol. 16, no. 3, ed. by Toni, F. & Bentahar, J., 2008, pp. 214–248.

[70] Davis, R., Shrobe, H. & Szolovits, P., "What is a knowledge representation?", *AI Magazine*, vol. 14, no. 1, 1993, pp. 17–33.

[71] De Greef, T. E., Arciszewski, H. F. R. & Neerincx, M. A., "Adaptive automation based on an object-oriented task model: implementation and evaluation in a realistic C2 environment", *Journal of Cognitive Engineering and Decision Making*, vol. 4, no. 2, 2010, pp. 152–182.

[72] De Greef, T., Oomes, A. H. J. & Neerincx, M. A., "Distilling support opportunities to improve urban search and rescue missions", in: *Human-Computer Interaction - Interacting in Various Application Domains*, Springer, 2009, 703–712.

[73] De Groot, A. D., "Perception and memory versus thought: some old ideas and recent findings", *Problem solving*, 1966, pp. 19–50.

[74] Deci, E. L. & Ryan, R. M., "The "what" and "why" of goal pursuits: human needs and the self-determination of behavior", *Psychological Inquiry*, vol. 11, no. 4, 2000, pp. 227–268.

[75] Dennett, D. C., "Intentional systems", *The Journal of Philosophy*, vol. 68, no. 4, 1971, pp. 87–106.

[76] Dickey, M. D., "Engaging by design: how engagement strategies in popular computer and video games can inform instructional design", *Educational Technology Research & Development*, vol. 53, no. 2, 2005, pp. 67–83.

[77] Dickey, M. D., "Game design narrative for learning: appropriating adventure game design narrative devices and techniques for the design of interactive learning environments", *Educational Technology Research & Development*, vol. 54, no. 3, 2006, pp. 245–263.

[78] Dignum, M. V., "A model for organizational interaction: based on agents, founded in logic", PhD thesis, Utrecht University, 2004.

[79] Dowell, J. & Long, J., "Conception of the cognitive engineering design problem", *Ergonomics*, vol. 41, no. 2, 1998, pp. 126–139.

[80] Drach-Zahavy, A. & Erez, M., "Challenge versus threat effects on the goal–performance relationship", *Organizational Behavior and Human Decision Processes*, vol. 88, no. 2, 2002, pp. 667–682.

[81] Dreyfus, H. L. & Dreyfus, S. E., "Peripheral vision: expertise in real world contexts", *Organization Studies*, vol. 26, no. 5, 2005, pp. 779–792.
[82] Dweck, C. S., "Motivational processes affecting learning", *American Psychologist*, vol. 41, no. 10, 1986, pp. 1040–1048.
[83] Eagleman, D, *Incognito: The Secret Lives ofthe Brain*, 2011.
[84] Easterday, M. W., Aleven, V., Scheines, R. & Carver, S. M., "Using tutors to improve educational games", in: *Conference on Artificial Intelligence in Education*, Springer, 2011, pp. 63–71.
[85] Edelson, D. C., "Design research: what we learn when we engage in design", *The Journal of the Learning sciences*, vol. 11, no. 1, 2002, pp. 105–121.
[86] Egenfeldt-Nielsen, S., "Overview of research on the educational use of video games", *Digital Kompetanse*, vol. 1, no. 3, 2006, pp. 184–213.
[87] Elliott, A. J., "Approach and avoidance motivation and achievement goals", *Educational Psychologist*, vol. 34, no. 3, 1999, pp. 169–189.
[88] Elliott, C. & Brzezinski, J., "Autonomous agents as synthetic characters", *AI Magazine*, vol. 19, no. 2, 1998, pp. 13–30.
[89] Endsley, M. R., "Toward a theory of situation awareness in dynamic systems", *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 37, no. 1, 1995, pp. 32–64.
[90] Entwistle, N. J., Thompson, S. & Tait, H., *Guidelines for promoting effective learning in higher education*, Centre for Research on Learning and Instruction, University of Edinburgh, 1992.
[91] Eraut, M., "Non-formal learning and tacit knowledge in professional work", *British Journal of Educational Psychology*, vol. 70, no. 1, 2000, pp. 113–136.
[92] Ferdinandus, G. R., "Automated Scenario Generation - Coupling Planning Techniques with Smart Objects", MA thesis, Universiteit Utrecht, 2012.
[93] Ferdinandus, G. R., Peeters, M. M. M., Van den Bosch, K. & Meyer, J.-J. C., "Automated scenario generation - coupling planning techniques with smart objects", in: *Conference for Computer Supported Education*, 2013, pp. 76–81.
[94] Fischer, G., Lemke, A. C., Mastaglio, T. & Morch, A. I., "The role of critiquing in cooperative problem solving", *Transactions on Information Systems (TOIS)*, vol. 9, no. 2, 1991, pp. 123–151.
[95] Fogli, D. & Guida, G., "Knowledge-centered design of decision support systems for emergency management", *Decision Support Systems*, 2013.
[96] Fornara, N. & Colombetti, M., "Specifying and enforcing norms in artificial institutions", in: *Conference on Autonomous Agents and Multi-Agent Systems*, 2008, pp. 1481–1484.
[97] Fowler, C. J. H. & Mayes, J. T., "Learning relationships from theory to design", *Research in Learning Technology*, vol. 7, no. 3, 1999.
[98] Fowlkes, J., Dwyer, D. J., Oser, R. L. & Salas, E., "Event-based approach to training (EBAT)", *The International Journal of Aviation Psychology*, vol. 8, no. 3, 1998, pp. 209–221.
[99] Franklin, S. & Graesser, A., "Is it an agent, or just a program? A taxonomy for autonomous agents", in: *Intelligent Agents III: Agent Theories, Architectures, and Languages*, Springer, 1997, pp. 21–35.

[100] Frasson, C., Mengelle, T. & Aimeur, E., "Using pedagogical agents in a multi-strategic intelligent tutoring system", in: *Workshop on Pedagogical Agents (ITS Conference)*, 1997.
[101] Freuder, E. C. & Mackworth, A. K., *Constraint-based reasoning*, vol. 58, 1-3, The MIT Press, 1994.
[102] Freytag, G., *Die technik des dramas*, S. Hirzel, 1905.
[103] Gagne, R. M. & Briggs, L. J., *Principles of instructional design*, Holt, Rinehart & Winston, 1974.
[104] Gardner, H., *The disciplined mind*, Simon & Schuster New York, 1999.
[105] Gee, J. P., "Good video games and good learning", in: *Phi Kappa Phi Forum*, vol. 85, 2, 2005, pp. 33–37.
[106] Gemrot, J., Kadlec, R., Bída, M., Burkert, O., Píbil, R., Havlíček, J., Zemčák, L., Šimlovič, J., Vansa, R., Štolba, M., Plch, T. & Brom, C., "Pogamut 3 can assist developers in building AI (not only) for their videogame agents", in: *Agents for Games and Simulations*, Springer, 2009, pp. 1–15.
[107] Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., Noy, N. F. & Tu, S. W., "The evolution of Protégé: an environment for knowledge-based systems development", *International Journal of Human-computer studies*, vol. 58, no. 1, 2003, pp. 89–123.
[108] Gerjets, P., Scheiter, K. & Catrambone, R., "Can learning from molar and modular worked examples be enhanced by providing instructional explanations and prompting self-explanations?", *Learning and Instruction*, vol. 16, no. 2, 2006, pp. 104–121.
[109] Ghallab, M., Nau, D. & Traverso, P., *Automated Planning: Theory & Practice*, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.
[110] Gibson, J., "The theory of affordances", in: *Perceiving, Acting, and Knowing*, Hillsdale, NJ: Lawrence Erlbaum, 1977, pp. 67–82.
[111] Go, K. & Carroll, J. M., "The blind men and the elephant: views of scenario-based system design", *Interactions*, vol. 11, no. 6, 2004, pp. 44–53.
[112] Göbel, S., Wendel, V., Ritter, C. & Steinmetz, R., "Personalized, adaptive digital educational games using narrative game-based learning objects", in: *Entertainment for Education. Digital Techniques and Systems*, Springer, 2010, pp. 438–445.
[113] Gopher, D., Weil, M. & Siegel, D., "Practice under changing priorities: an approach to the training of complex skills", *Acta Psychologica*, vol. 71, no. 1, 1989, pp. 147–177.
[114] Grabinger, R. S. & Dunlap, J. C., "Rich environments for active learning: a definition", *Research in Learning Technology*, vol. 3, no. 2, 1995, pp. 5–34.
[115] Graesser, A. C., Conley, M. W. & Olney, A., "Intelligent tutoring systems", *APA Handbook of Educational Psychology. Washington, DC: American Psychological Association*, 2012.
[116] Greitzer, F. L., Kuchar, O. A. & Huston, K., "Cognitive science implications for enhancing training effectiveness in a serious gaming context", *Journal on Educational Resources in Computing (JERIC)*, vol. 7, no. 3, 2007, 2:1–2:16.

[117] Grootjen, M, de Vries, N. T. & Ghijben, N. B., "Applying situated cognitive engineering for platform automation in the 21st century", in: *Ship Control Systems Symposium*, 2009.
[118] Große, C. S. & Renkl, A., "Finding and fixing errors in worked examples: can this foster learning outcomes?", *Learning and Instruction*, vol. 17, no. 6, 2007, pp. 612–634.
[119] Grossi, D., Aldewereld, H. & Dignum, F., "Ubi lex, ibi poena: designing norm enforcement in e-institutions", in: *Coordination, Organizations, Institutions, and Norms in Agent Systems II*, Springer, 2007, pp. 101–114.
[120] Grossi, D., Gabbay, D. & Van Der Torre, L., "The norm implementation problem in normative multi-agent systems", in: *Specification and Verification of Multi-agent Systems*, Springer, 2010, pp. 195–224.
[121] Grosz, B. J. & Kraus, S., "Collaborative plans for complex group action", *Artificial Intelligence*, vol. 86, no. 2, 1996, pp. 269–357.
[122] Gruber, T. R., "Toward principles for the design of ontologies used for knowledge sharing", *International Journal of Human-Computer Studies*, vol. 43, no. 5, 1995, pp. 907–928.
[123] Habgood, M. J. & Ainsworth, S. E., "Motivating children to learn effectively: exploring the value of intrinsic integration in educational games", *The Journal of the Learning Sciences*, vol. 20, no. 2, 2011, pp. 169–206.
[124] Hackos, J. T. & Redish, J., *User and task analysis for interface design*, Wiley New York, 1998.
[125] Hajdukiewicz, J. R. & Vicente, K. J., "A theoretical note on the relationship between work domain analysis and task analysis", *Theoretical Issues in Ergonomics Science*, vol. 5, no. 6, 2004, pp. 527–538.
[126] Hannafin, M. J. & Land, S. M., "The foundations and assumptions of technology-enhanced student-centered learning environments", *Instructional Science*, vol. 25, no. 3, 1997, pp. 167–202.
[127] Harbers, M., "Explaining agent behavior in virtual training", PhD thesis, Utrecht University, 2011.
[128] Harter, S., "Pleasure derived from challenge and the effects of receiving grades on children's difficulty level choices", *Child Development*, 1978, 788–799.
[129] Hattie, J. & Timperley, H., "The power of feedback", *Review of Educational Research*, vol. 77, no. 1, 2007, pp. 81–112.
[130] Hays, R. T., *The effectiveness of instructional games: A literature review and discussion*, tech. rep., DTIC Document, 2005.
[131] Hecker, C., "Games", *Communications of the ACM*, vol. 43, no. 7, 2000, pp. 35–39.
[132] Helsdingen, A. S., Van den Bosch, K., Van Gog, T. & Van Merriënboer, J. J., "The effects of critical thinking instruction on training complex decision making", *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 52, no. 4, 2010, pp. 537–545.
[133] Herrington, J., Oliver, R. & Reeves, T. C., "Patterns of engagement in authentic online learning environments", *Australian Journal of Educational Technology*, vol. 19, no. 1, 2003, pp. 59–71.

[134] Heuvelink, A., Van den Bosch, K., Van Doesburg, W. A. & Harbers, M., *Intelligent agent supported training in virtual simulations*, tech. rep., DTIC Document, 2009.
[135] Hevner, A. R., "The three cycle view of design science research", *Scandinavian Journal of Information Systems*, vol. 19, no. 2, 2007, pp. 87–92.
[136] Hindriks, K. V., Van Riemsdijk, B., Behrens, T., Korstanje, R., Kraayenbrink, N., Pasman, W. & De Rijk, L., "Unreal GOAL bots", in: *Agents for Games and Simulations II*, Springer, 2011, pp. 1–18.
[137] Hoekstra, R., *Ontology representation: design patterns and ontologies that make sense*, IOS Press, 2009.
[138] Hollnagel, E. & Woods, D. D., "Cognitive systems engineering: new wine in new bottles", *International Journal of Man-Machine Studies*, vol. 18, no. 6, 1983, pp. 583–600.
[139] Houtkamp, J. M., "Affective appraisal of virtual environments", PhD thesis, Utrecht University, 2012.
[140] Hovers, P., "Scenario Authoring of Serious Games - An Interface for a First Aid Game", MA thesis, University of Amsterdam, 2013.
[141] Hunicke, R., "The case for dynamic difficulty adjustment in games", in: *Conference on Advances in Computer Entertainment Technology*, 2005, pp. 429–433.
[142] Hunicke, R. & Chapman, V., "AI for dynamic difficulty adjustment in games", in: *AAAI Workshop on Challenges in Game Artificial Intelligence*, 2004, pp. 91–96.
[143] Igarashi, T. & Hughes, J. F., "A suggestive interface for 3D drawing", in: *Symposium on User Interface Software and Technology*, 2001, pp. 173–181.
[144] Jackson, S. A., Thomas, P. R., Marsh, H. W. & Smethurst, C. J., "Relationships between flow, self-concept, psychological skills, and performance", *Journal of Applied Sport Psychology*, vol. 13, no. 2, 2001, pp. 129–153.
[145] Jennett, C., Cox, A. L., Cairns, P., Dhoparee, S., Epps, A., Tijs, T. & Walton, A., "Measuring and defining the experience of immersion in games", *International Journal of Human-Computer Studies*, vol. 66, no. 9, 2008, pp. 641–661.
[146] Jennings, N. R., "Agent-oriented software engineering", in: *Multiple Approaches to Intelligent Systems*, Springer, 1999, pp. 4–10.
[147] Jennings, N. R., "An agent-based approach for building complex software systems", *Communications of the ACM*, vol. 44, no. 4, 2001, pp. 35–41.
[148] Johnson, W. L., Rickel, J. W. & Lester, J. C., "Animated pedagogical agents: face-to-face interaction in interactive learning environments", *International Journal of Artificial intelligence in education*, vol. 11, no. 1, 2000, pp. 47–78.
[149] Jonassen, D. H., "Designing constructivist learning environments", *Instructional-Design Theories and Models: A New Paradigm of Instructional Theory*, vol. 2, 1999, pp. 215–239.
[150] Jonassen, D. H., Howland, J., Moore, J. & Marra, R. M., *Learning to solve problems with technology: A constructivist perspective*, Prentice Hall, 2002.

[151] Kallmann, M. & Thalmann, D., "Modeling objects for interaction tasks", in: *Eurographics Workshop on Animation and Simulation*, vol. 9, 1998, pp. 73–86.

[152] Kalyuga, S. & Sweller, J., "Rapid dynamic assessment of expertise to improve the efficiency of adaptive e-learning", *Educational Technology Research & Development*, vol. 53, no. 3, 2005, pp. 83–93.

[153] Kapoor, A., Burleson, W. & Picard, R. W., "Automatic prediction of frustration", *International Journal of Human-Computer Studies*, vol. 65, no. 8, 2007, pp. 724–736.

[154] Kasap, Z. & Magnenat-Thalmann, N., "Intelligent virtual humans with autonomy and personality: state-of-the-art", *Intelligent Decision Technologies*, vol. 1, no. 1, 2007, pp. 3–15.

[155] Kebritchi, M. & Hirumi, A., "Examining the pedagogical foundations of modern educational computer games", *Computers & Education*, vol. 51, no. 4, 2008, pp. 1729–1743.

[156] Keller, J. M., "Development and use of the arcs model of instructional design", *Journal of Instructional Development*, vol. 10, no. 3, 1987, pp. 2–10.

[157] Kickmeier-Rust, M. D. & Albert, D., "Micro-adaptivity: protecting immersion in didactically adaptive digital educational games", *Journal of Computer Assisted Learning*, vol. 26, no. 2, 2010, pp. 95–105.

[158] Kirschner, P. A., Sweller, J. & Clark, R. E., "Why minimal guidance during instruction does not work: an analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching", *Educational Psychologist*, vol. 41, no. 2, 2006, pp. 75–86.

[159] Kirschner, P. A., Beers, P. J., Boshuizen, H. & Gijselaers, W. H., "Coercing shared knowledge in collaborative learning environments", *Computers in human behavior*, vol. 24, no. 2, 2008, pp. 403–420.

[160] Klein, G., "Naturalistic decision making", *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 50, no. 3, 2008, pp. 456–460.

[161] Koedinger, K. R. & Corbett, A. T., "Cognitive tutors: technology bringing learning science to the classroom", *The Cambridge Handbook of the Learning Sciences*, 2006, pp. 61–78.

[162] Koedinger, K. R., Corbett, A. T. & Perfetti, C., "The knowledge-learning-instruction framework: bridging the science-practice chasm to enhance robust student learning", *Cognitive Science*, vol. 36, no. 5, 2012, pp. 757–798.

[163] Kopp, V., Stark, R. & Fischer, M. R., "Fostering diagnostic knowledge through computer-supported, case-based worked examples: effects of erroneous examples and feedback", *Medical Education*, vol. 42, no. 8, 2008, pp. 823–829.

[164] Kotovsky, K., Hayes, J. R. & Simon, H. A., "Why are some problems hard? Evidence from Tower of Hanoi", *Cognitive Psychology*, vol. 17, no. 2, 1985, pp. 248–294.

[165] Landis, J. R. & Koch, G. G., "The measurement of observer agreement for categorical data", *Biometrics*, vol. 33, no. 1, 1977, pp. 159–174.

[166] Laurillard, D., *Rethinking University Teaching - a framework for the effective use of educational technology*, 1993.

[167] Lester, J. C., Converse, S. A., Kahler, S. E., Barlow, S. T., Stone, B. A. & Bhogal, R. S., "The persona effect: affective impact of animated pedagogical agents", in: *The ACM SIGCHI Conference on Human factors in Computing Systems*, 1997, pp. 359–366.

[168] Lester, J. C., Voerman, J. L., Towns, S. G. & Callaway, C. B., "Deictic believability: coordinated gesture, locomotion, and speech in lifelike pedagogical agents", *Applied Artificial Intelligence*, vol. 13, no. 4-5, 1999, pp. 383–414.

[169] Levesque, H. J., Cohen, P. R. & Nunes, J. H. T., "On acting together", in: *National Conference on Artificial Intelligence*, vol. 90, AAAI Press, 1990, pp. 94–99.

[170] Linnenbrink, E. A. & Pintrich, P. R., "Motivation as an enabler for academic success", *School Psychology Review*, vol. 31, no. 3, 2002, pp. 313–327.

[171] Lipshitz, R. & Strauss, O., "Coping with uncertainty: a naturalistic decision-making analysis", *Organizational Behavior and Human Decision Processes*, vol. 69, no. 2, 1997, pp. 149–163.

[172] Lipshitz, R., Klein, G., Orasanu, J. & Salas, E., "Taking stock of naturalistic decision making", *Journal of Behavioral Decision Making*, vol. 14, no. 5, 2001, pp. 331–352.

[173] Little, R. J. A. & Rubin, D. B., "The analysis of social science data with missing values", *Sociological Methods & Research*, vol. 18, no. 2-3, 1989, pp. 292–326.

[174] Liu, C., Agrawal, P., Sarkar, N. & Chen, S., "Dynamic difficulty adjustment in computer games through real-time anxiety-based affective feedback", *International Journal of Human-Computer Interaction*, vol. 25, no. 6, 2009, pp. 506–529.

[175] Locke, E. A., Shaw, K. N., Saari, L. M. & Latham, G. P., "Goal setting and task performance: 1969–1980", *Psychological Bulletin*, vol. 90, no. 1, 1981, pp. 125–152.

[176] Löckelt, M., Pecourt, E. & Pfleger, N., "Balancing narrative control and autonomy for virtual characters in a game scenario", in: *Intelligent Technologies for Interactive Entertainment*, Springer, 2005, pp. 251–255.

[177] Lopes, R., "Scenario adaptivity in serious games", in: *Conference on the Foundations of Digital Games*, 2010, pp. 268–270.

[178] Lopes, R. & Bidarra, R., "A semantic generation framework for enabling adaptive game worlds", in: *Conference on Advances in Computer Entertainment*, vol. 8, 2011.

[179] Maehr, M. L. & Meyer, H. A., "Understanding motivation and schooling: where we've been, where we are, and where we need to go", *Educational Psychology Review*, vol. 9, no. 4, 1997, pp. 371–409.

[180] Magerko, B., "Story representation and interactive drama", in: *Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2005, pp. 87–92.

[181] Magerko, B., Laird, J. E., Assanie, M., Kerfoot, A. & Stokes, D., "AI characters and directors for interactive computer games", in: *Conference on Innovative Applications of Artificial Intelligence*, vol. 1001, IAAI'04, AAAI Press, 2004, pp. 877–883.

[182] Malone, T. W., "Toward a theory of intrinsically motivating instruction", *Cognitive Science*, vol. 5, no. 4, 1981, pp. 333–369.
[183] Marsella, S. C. & Johnson, W. L., "An instructor's assistant for team-training in dynamic multi-agent virtual worlds", in: *Conference on Intelligent Tutoring Systems*, Springer, 1998, pp. 464–473.
[184] Mateas, M. & Stern, A., "A behavior language for story-based believable agents", *Intelligent Systems*, vol. 17, no. 4, 2002, pp. 39–47.
[185] Mateas, M. & Stern, A., "Façade: an experiment in building a fully-realized interactive drama", in: *Game Developers Conference, Game Design track*, vol. 2, 2003.
[186] Mayer, R. E. & Moreno, R., "Nine ways to reduce cognitive load in multimedia learning", *Educational Psychologist*, vol. 38, no. 1, 2003, pp. 43–52.
[187] Mayer, R. E., "Applying the science of learning: evidence-based principles for the design of multimedia instruction", *American Psychologist*, vol. 63, no. 8, 2008, pp. 760–769.
[188] Mayes, J. T. & Fowler, C. J., "Learning technology and usability: a framework for understanding courseware", *Interacting with Computers*, vol. 11, no. 5, 1999, pp. 485–497.
[189] McCarthy, K., Reilly, J., McGinty, L. & Smyth, B., "Experiments in dynamic critiquing", in: *Conference on Intelligent User Interfaces*, 2005, pp. 175–182.
[190] McGrenere, J. & Ho, W., "Affordances: clarifying and evolving a concept", in: *Graphics Interface Conference*, vol. 2000, 2000, pp. 179–186.
[191] McLaren, B. M., Adams, D., Durkin, K., Goguadze, G., Mayer, R. E., Rittle-Johnson, B., Sosnovsky, S., Isotani, S. & Van Velsen, M., "To err is human, to explain and correct is divine: a study of interactive erroneous examples with middle school math students", in: *European Conference on Technology-Enhanced Learning*, Springer, 2012, pp. 222–235.
[192] McNamara, D. S., Jackson, G. T. & Graesser, A. C., *Intelligent tutoring and games (ITaG)*, ed. by Baek, Y., 2010.
[193] McQuiggan, S., Rowe, J., Lee, S. & Lester, J., "Story-based learning: the impact of narrative on learning experiences and outcomes", in: *Conference on Intelligent Tutoring Systems*, Springer, 2008, pp. 530–539.
[194] Mehm, F., Konert, J., Göbel, S. & Steinmetz, R., "An authoring tool for adaptive digital educational games", in: *European Conference on Technology Enhanced Learning*, Springer, 2012, pp. 236–249.
[195] Melis, E., "Design of erroneous examples for activemath", in: *Conference on Artificial Intelligence in Education*, 2005, pp. 451–458.
[196] Merrill, M. D., "First principles of instruction", *Educational Technology Research & Development*, vol. 50, no. 3, 2002, pp. 43–59.
[197] Merrill, M. D., "A pebble-in-the-pond model for instructional design", *Performance Improvement*, vol. 41, no. 7, 2002, pp. 41–46.
[198] Merrill, P. F., "Hierarchical & information processing task analysis: a comparison", *Journal of Instructional Development*, vol. 1, no. 2, 1978, pp. 35–40.

[199] Mihalca, L., Salden, R. J. C. M., Corbalan, G., Paas, F. & Miclea, M., "Effectiveness of cognitive-load based adaptive instruction in genetics education", *Computers in Human Behavior*, vol. 27, no. 1, 2011, pp. 82–88.

[200] Mikropoulos, T. A. & Natsis, A., "Educational virtual environments: a ten-year review of empirical research (1999–2009)", *Computers & Education*, vol. 56, no. 3, 2011, pp. 769–780.

[201] Militello, L. G. & Hutton, R. J. B., "Applied cognitive task analysis (ACta): a practitioner's toolkit for understanding cognitive task demands", *Ergonomics*, vol. 41, no. 11, 1998, pp. 1618–1641.

[202] Millán, E., Loboda, T. & Cruz, J. L. Pérez-de-la, "Bayesian networks for student model engineering", *Computers & Education*, vol. 55, no. 4, 2010, pp. 1663–1683.

[203] Miller, G. A., "The magical number seven, plus or minus two: some limits on our capacity for processing information", *Psychological Review*, vol. 63, 1956, pp. 81–97.

[204] Minsky, M., "A framework for representing knowledge", *The Psychology of Computer Vision*, ed. by Winston, P., 1975.

[205] Mioch, T., Smets, N. J. J. M. & Neerincx, M. A., "Predicting performance and situation awareness of robot operators in complex situations by unit task tests", in: *Conference on Advances in Computer-Human Interaction*, 2012, pp. 241–246.

[206] Mitchell, M., "Situational interest: its multifaceted structure in the secondary school mathematics classroom", *Journal of Educational Psychology*, vol. 85, 1993, pp. 424–424.

[207] Mitrovic, A., "Modeling domains and students with constraint-based modeling", in: *Advances in Intelligent Tutoring Systems*, ed. by Nkambou, R., Bourdeau, J. & Mizoguchi, R., vol. 308, Studies in Computational Intelligence, Springer Berlin / Heidelberg, 2010, pp. 63–80.

[208] Mitrovic, A. & Martin, B., "Evaluating the effect of open student models on self-assessment", *International Journal of Artificial Intelligence in Education*, vol. 17, no. 2, 2007, pp. 121–144.

[209] Mitrovic, A. & Ohlsson, S., "Evaluation of a constraint-based tutor for a database", *International Journal of Artificial Intelligence in Education*, vol. 10, 1999, pp. 238–256.

[210] Mitrovic, A., Mayo, M., Suraweera, P. & Martin, B., "Constraint-based tutors: a success story", in: *Engineering of Intelligent Systems*, Springer, 2001, pp. 931–940.

[211] Moray, N., Sanderson, P. M. & Vicente, K. J., "Cognitive task analysis of a complex work domain: a case study", *Reliability Engineering & System Safety*, vol. 36, no. 3, 1992, pp. 207–216.

[212] Moreno-Ger, P., Sierra, J. L., Martínez-Ortiz, I. & Fernández-Manjón, B., "A documental approach to adventure game development", *Science of Computer Programming*, vol. 67, no. 1, 2007, pp. 3–31.

[213] Mota, S. & Picard, R. W., "Automated posture analysis for detecting learner's interest level", in: *IEEE Workshop on Computer Vision and Pattern Recognition*, IEEE Computer Society, 2003.

[214] Mott, B. W. & Lester, J. C., "U-director: a decision-theoretic narrative planning architecture for storytelling environments", in: *Conference on Autonomous Agents and Multiagent Systems*, 2006, pp. 977–984.
[215] Mott, B. W., Callaway, C. B., Zettlemoyer, L. S., Lee, S. Y. & Lester, J. C., "Towards narrative-centered learning environments", in: *Symposium on Narrative Intelligence*, 1999, pp. 78–82.
[216] Muller, T. J., Heuvelink, A., Van den Bosch, K. & Swartjes, I., "Glengarry Glen Ross: using BDI for sales game dialogues", in: *Conference on Artificial Intelligence and Interactive Digital Entertainment*, AAAI, 2012, pp. 167–172.
[217] Murray, T. & Arroyo, I., "Toward measuring and maintaining the zone of proximal development in adaptive instructional systems", in: *Conference on Intelligent Tutoring Systems*, 2002, pp. 749–758.
[218] Murray, T. & Arroyo, I., "Toward an operational definition of the zone of proximal development for adaptive instructional software", in: *Conference of the Cognitive Science Society*, Boston, MA, 2003.
[219] Nacke, L. & Lindley, C. A., "Flow and immersion in first-person shooters: measuring the player's gameplay experience", in: *Conference on Future Play: Research, Play, Share*, 2008, pp. 81–88.
[220] Nau, D. S., "Current trends in automated planning", *AI Magazine*, vol. 28, no. 4, 2007, pp. 43–58.
[221] Nau, D. S., Au, T.-C., Ilghami, O., Kuter, U., Murdock, J. W., Wu, D. & Yaman, F., "SHOP2: an HTN planning system", *Journal of Artificial Intelligence Research*, vol. 20, 2003, pp. 379–404.
[222] Neerincx, M. A. & Lindenberg, J., "Situated cognitive engineering for complex task environments", in: *Naturalistic Decision Making and Macrocognition*, Ashgate, 2008, pp. 373–390.
[223] Neerincx, M. A., Lindenberg, J, Smets, N, Grant, T, Bos, A, Olmedo-Soler, A, Brauer, U & Wolff, M, "Cognitive engineering for long duration missions: human-machine collaboration on the Moon and Mars", in: *Conference on Space Mission Challenges for Information Technology*, IEEE, 2006, pp. 7–14.
[224] Neerincx, M. A., Bos, A, Olmedo-Soler, A, Brauer, U, Breebaart, L, Smets, N, Lindenberg, J, Grant, T & Wolff, M, "The mission execution crew assistant: improving human-machine team resilience for long duration missions", in: *International Astronautical Congress*, 2008, pp. 1–12.
[225] Neerincx, M. A., Kennedie, S., Grootjen, F. & Grootjen, M., "Modelling cognitive task load and emotion for adaptive maritime interfaces", in: *International Conference of the Augmented Cognition: Neuroergonomics and Operational Neuroscience*, ed. by Schmorrow, D., Estabrooke, I. & Grootjen, M., Berlin/Heidelberg: Springer, 2009, pp. 260–269.
[226] Neerincx, M. A., "Situated cognitive engineering for crew support in space", *Personal and Ubiquitous Computing*, vol. 15, no. 5, 2011, pp. 445–456.
[227] Niehaus, J. & Riedl, M. O., "Scenario adaptation: an approach to customizing computer-based training games and simulations", in: *Workshop on Intelligent Educational Games (AIED Conference)*, vol. 3, 2009, pp. 89–98.

[228] Norling, E., "Folk psychology for human modelling: extending the BDI paradigm", in: *Conference on Autonomous Agents and Multiagent Systems*, IEEE Computer Society, 2004, pp. 202–209.

[229] Norling, E. & Sonenberg, L., "Creating interactive characters with bdi agents", in: *Australian Workshop on Interactive Entertainment*, 2004, pp. 69–76.

[230] Norman, D. A., "Cognitive engineering", in: *User-Centered System Design*, Law-rence Erlbaum Associates, 1986, pp. 31–61.

[231] Norman, D. A., *The design of everyday things*, Basic books, 2002.

[232] Noy, N. & McGuinness, D., *Ontology Development 101: A guide to creating your first ontology*, tech. rep., Knowledge Systems Laboratory, Stanford University, 2001.

[233] Noy, N. F., Fergerson, R. W. & Musen, M. A., "The knowledge model of Protégé-2000: combining interoperability and flexibility", in: *Knowledge Engineering and Knowledge Management Methods, Models, and Tools*, Springer, 2000, pp. 17–32.

[234] O'Brien, P. D. & Nicol, R. C., "FIPA - towards a standard for software agents", *BT Technology Journal*, vol. 16, no. 3, 1998, pp. 51–59.

[235] Ohlsson, S., "Some principles of intelligent tutoring", *Instructional Science*, vol. 14, no. 3, 1986, pp. 293–326.

[236] Ohlsson, S., *Deep learning: How the mind overrides experience*, Cambridge University Press, 2011.

[237] Oser, R. L., Cannon-Bowers, J. A., Salas, E. & Dwyer, D. J., "Enhancing human performance in technology-rich environments: guidelines for scenario-based training", *Human Technology Interaction in Complex Systems*, vol. 9, 1999, pp. 175–202.

[238] Paas, F., Tuovinen, J. E., Van Merrienboer, J. J. G. & Darabi, A. A., "A motivational perspective on the relation between mental effort and performance: optimizing learner involvement in instruction", *Educational Technology Research and Development*, vol. 53, no. 3, 2005, pp. 25–34.

[239] Paas, F. G. W. C., Van Merriënboer, J. J. G. & Adam, J. J., "Measurement of cognitive load in instructional research", *Perceptual and motor skills*, vol. 79, no. 1, 1994, pp. 419–430.

[240] Paas, F. G. & Van Merriënboer, J. J., "Variability of worked examples and transfer of geometrical problem-solving skills: a cognitive-load approach", *Journal of Educational Psychology*, vol. 86, no. 1, 1994, pp. 122–133.

[241] Paetsch, F., Eberlein, A. & Maurer, F., "Requirements engineering and agile software development", in: *IEEE Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, IEEE, 2003, pp. 308–313.

[242] Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "Scenario-based training: director's cut", in: *Conference on Artificial Intelligence in Education*, 2011, pp. 264–272.

[243] Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "Situated cognitive engineering: requirements and design of directed scenario-based training", Unpublished peer-reviewed proceedings of the *International Workshop on Authoring Simulation and Game-Based Intelligent Tutoring* (AIED Conference), 2011.

[244] Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "An ontology for integrating didactics into a serious training game", in: *Workshop on Pedagogically-driven Serious Games (ECTEL Conference)*, ed. by Bocconi, S., Klamma, R. & S., B. Y., vol. 898, CEUR Workshop Proceedings, 2012, pp. 1–10.

[245] Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "Situated cognitive engineering: the requirements and design of directed scenario-based training", in: *Conference on Advances in Computer-Human Interaction*, ed. by Miller, L. & Roncagliolo, S., Xpert Publishing Services, 2012, pp. 266–272.

[246] Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "The design and effect of automated directions during scenario-based training", *Computers & Education*, vol. 70, 2014, pp. 173–183.

[247] Peeters, M. M. M., Van den Bosch, K., Meyer, J.-J. C. & Neerincx, M. A., "An ontology for automated scenario-based training", *International Journal for Technology Enhanced Learning*, ed. by Bocconi, S., Klamma, R. & Bachvarova, Y. S., (conditionally accepted).

[248] Piaget, J., "Intellectual evolution from adolescence to adulthood", *Human Development*, vol. 15, no. 1, 1972, pp. 1–12.

[249] Plowman, L., Luckin, R., Laurillard, D., Stratfold, M. & Taylor, J., "Designing multimedia for learning: narrative guidance and narrative construction", in: *Conference on Human Factors in Computing Systems*, 1999, pp. 310–317.

[250] Popescu, O., Aleven, V. & Koedinger, K., "A knowledge-based approach to understanding students' explanations", in: *Workshop on Tutorial Dialogue Systems (AIED Conference)*, 2003.

[251] Prince, M., "Does active learning work? A review of the research", *Journal of Engineering Education*, vol. 93, no. 3, 2004, pp. 223–231.

[252] *Protégé: a free, open source ontology editor and knowledge-base framework*, Dec. 2013, URL: `http://protege.stanford.edu`.

[253] Randel, J. M., Pugh, H. L. & Reed, S. K., "Differences in expert and novice situation awareness in naturalistic decision making", *International Journal of Human-Computer Studies*, vol. 45, no. 5, 1996, pp. 579–597.

[254] Rao, A. S. & Georgeff, M. P., "Modeling rational agents within a BDI-architecture", in: *Conference on Principles of Knowledge Representation and Reasoning*, ed. by Allen, J., Fikes, R. & Sandewall, E, Morgan Kaufmann, San Mateo, CA, 1991, pp. 473–484.

[255] Rao, A. S. & Georgeff, M. P., "BDI agents: from theory to practice", in: *Conference on Multi-Agent Systems*, vol. 95, 1995, pp. 312–319.

[256] Rasmussen, J., *Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*, Elsevier Science Inc., 1986.

[257] Renkl, A., "Learning mathematics from worked-out examples: analyzing and fostering self-explanations", *European Journal of Psychology of Education*, vol. 14, no. 4, 1999, pp. 477–488.

[258] Renkl, A., "Worked-out examples: instructional explanations support learning by self-explanations", *Learning and instruction*, vol. 12, no. 5, 2002, pp. 529–556.

[259] Renkl, A., Stark, R., Gruber, H. & Mandl, H., "Learning from worked-out examples: the effects of example variability and elicited self-explanations", *Contemporary Educational Psychology*, vol. 23, no. 1, 1998, pp. 90–108.

[260] Resnick, L. B., Wang, M. C. & Kaplan, J., "Task analysis in curriculum design: a hierarchically sequenced introductory mathematics curriculum", *Journal of Applied Behavior Analysis*, vol. 6, no. 4, 1973, pp. 679–709.

[261] Rieber, L. P., "Seriously considering play: designing interactive learning environments based on the blending of microworlds, simulations, and games", *Educational Technology Research & Development*, vol. 44, no. 2, 1996, pp. 43–58.

[262] Rieber, L. P., Smith, L. & Noah, D., "The value of serious play", *Educational Technology*, vol. 38, 1998, pp. 29–36.

[263] Riedl, M. O., Stern, A., Dini, D. & Alderman, J., "Dynamic experience management in virtual worlds for entertainment, education, and training", *International Transactions on Systems Science and Applications, Special Issue on Agent Based Systems for Human Learning*, vol. 4, no. 2, 2008, pp. 23–42.

[264] Riedl, M. O. & Stern, A., "Believable agents and intelligent story adaptation for interactive storytelling", in: *Technologies for Interactive Digital Storytelling and Entertainment*, Springer, 2006, pp. 1–12.

[265] Ross, D. T. & Schoman, K. E., "Structured analysis for requirements definition", *Software Engineering*, no. 1, 1977, pp. 6–15.

[266] Salas, E. & Cannon-Bowers, J. A., "The science of training: a decade of progress", *Annual Review of Psychology*, vol. 52, no. 1, 2001, pp. 471–499.

[267] Salas, E., Priest, H. A., Wilson, K. A. & Adler, A. B., "Scenario-based training: improving military mission performance and adaptability", in: *Military Life*, Praeger Publishers, 2006, pp. 32–53.

[268] Salden, R. J. C. M., Paas, F. & Van Merriënboer, J. J. G., "A comparison of approaches to learning task selection in the training of complex cognitive skills", *Computers in Human Behavior*, vol. 22, no. 3, 2006, pp. 321–333.

[269] Salden, R. J. C. M., Paas, F. & Van Merriënboer, J. J. G., "Personalised adaptive task selection in air traffic control: effects on training efficiency and transfer", *Learning and Instruction*, vol. 16, no. 4, 2006, pp. 350–362.

[270] Salmon, P., Jenkins, D., Stanton, N. & Walker, G., "Hierarchical task analysis vs. cognitive work analysis: comparison of theory, methodology and contribution to system design", *Theoretical Issues in Ergonomics Science*, vol. 11, no. 6, 2010, pp. 504–531.

[271] Sampayo-Vargas, S., Cope, C. J., He, Z. & Byrne, G. J., "The effectiveness of adaptive difficulty adjustments on students' motivation and learning in an educational computer game", *Computers & Education*, vol. 69, 2013, pp. 452–462.

[272] Savery, J. R. & Duffy, T. M., "Problem-based learning: an instructional model and its constructivist framework", *Educational Technology*, vol. 35, no. 5, 1995, pp. 31–38.

[273] *sCET: the situated Cognitive Engineering Tool*, Jan. 2014, URL: `http://www.scetool.nl/`.

[274] Schneider, W. & Shiffrin, R. M., "Controlled and automatic human information processing: detection, search, and attention", *Psychological Review*, vol. 84, no. 1, 1977, pp. 127–190.

[275] Schouten, D., Smets, N., Driessen, M., Hanekamp, M., Cremers, A. H. & Neerincx, M. A., "User requirement analysis of social conventions learning applications for non-natives and low-literates", in: *Engineering Psychology and Cognitive Ergonomics. Understanding Human Cognition*, Springer, 2013, pp. 354–363.

[276] Schraagen, J. M., Chipman, S. F. & Shalin, V. L., *Cognitive task analysis*, Psychology Press, 2000.

[277] Schunk, D. H., Pintrich, P. R. & Meece, J. L., *Motivation in education*, Pearson/Merrill Prentice Hall, 2009.

[278] Schunk, D. H. & Swartz, C. W., "Goals and progress feedback: effects on self-efficacy and writing achievement", *Contemporary Educational Psychology*, vol. 18, no. 3, 1993, pp. 337–354.

[279] Shaker, N., Asteriadis, S., Yannakakis, G. & Karpouzis, K., "Fusing visual and behavioral cues for modeling user experience in games", *Transactions on System Man and Cybernetics, Special Issue on Modern Control for Computer Games*, vol. 43, no. 6, 2013, pp. 1519–1531.

[280] Shanks, G., Tansley, E. & Weber, R., "Using ontology to validate conceptual models", *Communications of the ACM*, vol. 46, no. 10, 2003, pp. 85–89.

[281] Shapira, Z., "Task choice and assigned goals as determinants of task motivation and performance", *Organizational Behavior and Human Decision Processes*, vol. 44, no. 2, 1989, pp. 141–165.

[282] Shaw, M. L. G. & Woodward, J. B., "Modeling expert knowledge", *Knowledge Acquisition*, vol. 2, no. 3, 1990, pp. 179–206.

[283] Shute, V., Ventura, M., Bauer, M. & Zapata-Rivera, D., "Melding the power of serious games and embedded assessment to monitor and foster learning", *Serious Games: Mechanisms and Effects*, 2009, pp. 295–321.

[284] Si, M., Marsella, S. & Pynadath, D., "Directorial control in a decision-theoretic framework for interactive narrative", *Interactive Storytelling*, 2009, 221–233.

[285] Si, M., Marsella, S. C. & Pynadath, D. V., "Thespian: using multi-agent fitting to craft interactive drama", in: *Conference on Autonomous Agents and Multiagent Systems*, 2005, pp. 21–28.

[286] Skorupski, J. & Mateas, M., "Interactive story generation for writers: lessons learned from the Wide Ruled authoring tool", in: *Conference on Digital Arts and Culture*, 2009.

[287] Skorupski, J. & Mateas, M., "Novice-friendly authoring of plan-based interactive storyboards", in: *Conference on Artificial Intelligence and Interactive Digital Entertainment*, AAAI, 2010, pp. 174–179.

[288] Skorupski, J., Jayapalan, L., Marquez, S. & Mateas, M., "Wide Ruled: a friendly interface to author-goal based story generation", in: *Conference on Virtual Storytelling*, Springer, 2007, pp. 26–37.

[289] Smallman, H. S. & John, M. S., "Naive realism: misplaced faith in realistic displays", *Ergonomics in Design: The Quarterly of Human Factors Applications*, vol. 13, no. 3, 2005, pp. 6–13.

[290] Sowa, J. F., *Knowledge representation: logical, philosophical, and computational foundations*, vol. 13, MIT Press, 2000.

[291] Spiekman, M. E., Haazebroek, P. & Neerincx, M. A., "Requirements and platforms for social agents that alarm and support elderly living alone", in: *Social Robotics*, Springer, 2011, pp. 226–235.

[292] Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I. & Postma, E., "Adaptive game AI with dynamic scripting", *Machine Learning*, vol. 63, no. 3, 2006, pp. 217–248.

[293] Stanton, N. A., "Hierarchical task analysis: developments, applications, and extensions", *Applied Ergonomics*, vol. 37, no. 1, 2006, pp. 55–79.

[294] Sweetser, P. & Wyeth, P., "GameFlow: a model for evaluating player enjoyment in games", *Computers in Entertainment*, vol. 3, no. 3, 2005, pp. 1–24.

[295] Sweller, J., Ayres, P. & Kalyuga, S., *Cognitive load theory*, vol. 1, Springer, 2011.

[296] Tambe, M., "Towards flexible teamwork", *Journal of Artificial Intelligence Research*, vol. 7, 1997, pp. 83–124.

[297] Tavakol, M. & Dennick, R., "Making sense of Cronbach's alpha", *International Journal of Medical Education*, vol. 2, 2011, pp. 53–55.

[298] Theune, M., Faas, S., Heylen, D. K. J. & Nijholt, A., "The virtual storyteller: story creation by intelligent agents", in: *Conference on Technologies for Interactive Digital Storytelling and Entertainment*, Fraunhofer IRB Verlag, 2003, pp. 204–215.

[299] Thorndyke, P. W., "Cognitive structures in comprehension and memory of narrative discourse", *Cognitive Psychology*, vol. 9, no. 1, 1977, pp. 77–110.

[300] Togelius, J., de Nardi, R. & Lucas, S. M., "Towards automatic personalised content creation for racing games", in: *Symposium on Computational Intelligence and Games*, IEEE, 2007, pp. 252–259.

[301] Tripp, S. D. & Bichelmeyer, B., "Rapid prototyping: an alternative instructional design strategy", *Educational Technology Research & Development*, vol. 38, no. 1, 1990, pp. 31–44.

[302] Tsang, S., Balakrishnan, R., Singh, K. & Ranjan, A., "A suggestive interface for image guided 3D sketching", in: *conference on Human factors in computing systems*, 2004, pp. 591–598.

[303] Tsovaltzi, D., Melis, E., McLaren, B. M., Meyer, A.-K., Dietrich, M. & Goguadze, G., "Learning from erroneous examples: when and how do students benefit from them?", in: *Sustaining TEL: From Innovation to Learning and Practice*, Springer, 2010, pp. 357–373.

[304] Tudorache, T., Vendetti, J. & Noy, N. F., "Web-Protégé: a lightweight OWL ontology editor for the web", in: *Workshop on OWL: Experiences and Directions*, vol. 432, 2008.

[305] Tutenel, T., Bidarra, R., Smelik, R. M. & de Kraker, K. J., "The role of semantics in games and simulations", *Computers in Entertainment*, vol. 6, no. 4, 2008, pp. 1–35.

[306] Van den Bosch, K. & Riemersma, J. B. J., "Reflections on scenario-based training in tactical command", in: *Scaled worlds: Development, validation, and applications*, ed. by Schiflett, S., Ashgate, 2004, chap. 1, pp. 1–21.

[307] Van den Bosch, K., Harbers, M., Heuvelink, A. & Van Doesburg, W., "Intelligent agents for training on-board fire fighting", in: *Conference on Digital Human Modeling*, Springer, 2009, pp. 463–472.

[308] Van der Spek, E. D., Wouters, P. & Van Oostendorp, H., "Code red: triage or cognition-based design rules enhancing decisionmaking training in a game environment", *British Journal of Educational Technology*, vol. 42, no. 3, 2011, pp. 441–455.

[309] Van der Vecht, B., Meyer, A. P., Neef, M., Dignum, F. & Meyer, J.-J. C., "Influence-based autonomy levels in agent decision-making", in: *Coordination, Organizations, Institutions, and Norms in Agent Systems II*, Springer, 2007, pp. 322–337.

[310] Van der Vecht, B., Dignum, F., Meyer, J.-J. C. & Neef, M., "A dynamic coordination mechanism using adjustable autonomy", in: *Coordination, Organizations, Institutions, and Norms in Agent Systems III*, Springer, 2008, pp. 83–96.

[311] Van der Vecht, B., Dignum, F., Meyer, J.-J. C. & Dignum, V., "Organizations and autonomous agents: bottom-up dynamics of coordination mechanisms", in: *Coordination, Organizations, Institutions and Norms in Agent Systems IV*, Springer, 2009, pp. 17–32.

[312] Van Diggelen, J., Muller, T. & Van den Bosch, K., "Using artificial team members for team training in virtual environments", in: *Conference on Intelligent Virtual Agents*, Springer, 2010, pp. 28–34.

[313] Van Diggelen, J., Janssen, J., Mioch, T. & Neerincx, M., "Flexible design and implementation of cognitive models for predicting pilot errors in cockpit design", in: *Human Modelling in Assisted Transportation*, Springer, 2011, pp. 147–154.

[314] Van Doesburg, W. A., Heuvelink, A. & Van den Broek, E. L., "Tacop: a cognitive agent for a naval training simulation environment", in: *Conference on Autonomous Agents and Multi-Agent Systems*, 2005, pp. 34–41.

[315] Van Est, C., Poelman, R. & Bidarra, R., "High-level scenario editing for serious games", in: *Conference on Computer Graphics Theory and Applications*, 2011, pp. 339–346.

[316] Van Joolingen, W. R. & De Jong, T., "Modelling domain knowledge for intelligent simulation learning environments", *Computers & Education*, vol. 18, no. 1, 1992, pp. 29–37.

[317] Van Merriënboer, J. J. G., *Training Complex Cognitive Skills: A Four-Component Instructional Design Model for Technical Training*, Educational Technology Publications, 1997.

[318] Van Merriënboer, J. J. G., Clark, R. E. & de Croock, M. B. M., "Blueprints for complex learning: the 4C/ID-model", *Educational Technology Research & Development*, vol. 50, no. 2, 2002, pp. 39–61.

[319] Van Oijen, J., Vanhée, L. & Dignum, F., "CIGA: a middleware for intelligent agents in virtual environments", in: *Agents for Educational Games and Simulations*, Springer, 2012, pp. 22–37.
[320] Van Rooij, C. A., "User modeling for complex personalization problems: A general methodology and BDI-agent-based framework", MA thesis, Utrecht University, 2014.
[321] Van Welie, M., Van der Veer, G. C. & Eliëns, A., "An ontology for task world models", in: *Eurographics Workshop on Design Specification and Verification of Interactive Systems*, vol. 98, 1998, pp. 3–5.
[322] VanLehn, K., "The behavior of tutoring systems", *International Journal of Artificial Intelligence in Education*, vol. 16, no. 3, 2006, pp. 227–265.
[323] VanLehn, K., Jordan, P. W., Rosé, C. P., Bhembe, D., Böttner, M., Gaydos, A., Makatchev, M., Pappuswamy, U., Ringenberg, M. & Roque, A., "The architecture of Why2-atlas: a coach for qualitative physics essay writing", in: *Intelligent tutoring systems*, Springer, 2002, pp. 158–167.
[324] Vergunst, N., "BDI-based generation of robust task-oriented dialogues", PhD thesis, Utrecht University, 2011.
[325] Viappiani, P., Pu, P. & Faltings, B., "Conversational recommenders with adaptive suggestions", in: *Conference on Recommender Systems*, 2007, pp. 89 –96.
[326] Vicente, K. J., "Task analysis, cognitive task analysis, cognitive work analysis: what's the difference?", in: *Meeting of the Human Factors and Ergonomics Society*, vol. 39, 9, SAGE, 1995, pp. 534–537.
[327] Vygotsky, L. S., *Mind in Society: the Development of Higher Psychological Processes*, Cambridge, MA: Harvard University Press, 1978.
[328] Westera, M., Boschloo, J., Van Diggelen, J., Koelewijn, L. S., Neerincx, M. A. & Smets, N. J., "Employing use-cases for piecewise evaluation of requirements and claims", in: *European Conference on Cognitive Ergonomics*, 2010, pp. 279–286.
[329] Westra, J., Dignum, F. & Dignum, V., "Organizing scalable adaptation in serious games", in: *Agents for Educational Games and Simulations*, Springer, 2012, pp. 106–122.
[330] Weyhrauch, P. & Bates, J., *Guiding interactive drama*, Carnegie Mellon University Pittsburgh, PA, 1997.
[331] Wiegers, K. E., "Writing quality requirements", *Software Development*, vol. 7, no. 5, 1999, pp. 44–48.
[332] Wielinga, B. J., Schreiber, A. T. & Breuker, J. A., "KADS: a modelling approach to knowledge engineering", *Knowledge Acquisition*, vol. 4, no. 1, 1992, pp. 5–53.
[333] Wilson, K. A., Bedwell, W. L., Lazzara, E. H., Salas, E., Burke, C. S., Estock, J. L., Orvis, K. L. & Conkey, C., "Relationships between game attributes and learning outcomes review and research proposals", *Simulation & Gaming*, vol. 40, no. 2, 2009, pp. 217–266.
[334] Winn, W., "Some implications of cognitive theory for instructional design", *Instructional Science*, vol. 19, no. 1, 1990, pp. 53–69.
[335] Wood, H & Wood, D, "Help seeking, learning and contingent tutoring", *Computers & Education*, vol. 33, no. 2, 1999, pp. 153–169.

[336] Wood, R. E., "Task complexity: definition of the construct", *Organizational Behavior and Human Decision Processes*, vol. 37, no. 1, 1986, pp. 60–82.
[337] Woods, D. D. & Roth, E. M., "Cognitive engineering: human problem solving with tools", *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 30, no. 4, 1988, pp. 415–430.
[338] Wooldridge, M., *An introduction to multiagent systems*, Wiley, 2008.
[339] Wooldridge, M. & Jennings, N. R., "Intelligent agents: theory and practice", *Knowledge Engineering Review*, vol. 10, no. 2, 1995, pp. 115–152.
[340] Wu, W.-H., Chiou, W.-B., Kao, H.-Y., Alex Hu, C.-H. & Huang, S.-H., "Re-exploring game-assisted learning research: the perspective of learning theoretical bases", *Computers & Education*, vol. 59, no. 4, 2012, pp. 1153–1161.
[341] Wulf, G. & Shea, C. H., "Principles derived from the study of simple skills do not generalize to complex skill learning", *Psychonomic Bulletin & Review*, vol. 9, no. 2, 2002, pp. 185–211.
[342] Yamnill, S. & McLean, G. N., "Theories supporting transfer of training", *Human Resource Development Quarterly*, vol. 12, no. 2, 2001, pp. 195–208.
[343] Yannakakis, G. N. & Hallam, J., "Real-time game adaptation for optimizing player satisfaction", *Computational Intelligence and AI in Games*, vol. 1, no. 2, 2009, pp. 121–133.
[344] Yen, J., Yin, J., Ioerger, T. R., Miller, M. S., Xu, D. & Volz, R. A., "Cast: collaborative agents for simulating teamwork", in: *International Joint Conference on Artificial Intelligence*, vol. 17, 1, Lawrence Erlbaum Associates Ltd, 2001, pp. 1135–1144.
[345] Yen, J., Fan, X., Sun, S., Hanratty, T. & Dumer, J., "Agents with shared mental models for enhancing team decision makings", *Decision Support Systems*, vol. 41, no. 3, 2006, pp. 634–653.
[346] Young, M. F., "Instructional design for situated learning", *Educational Technology Research & Development*, vol. 41, no. 1, 1993, pp. 43–58.
[347] Zapata-Rivera, J.-D. & Greer, J. E., "Inspecting and visualizing distributed bayesian student models", in: *Intelligent Tutoring Systems*, Springer, 2000, pp. 544–553.
[348] Zook, A., Urban, S. L., Riedl, M. O., Holden, H. K., Sottilare, R. A. & Brawner, K. W., "Automated scenario generation: toward tailored and optimized military training in virtual environments", in: *Conference on Foundations of Digital Games*, vol. 7, 2012, pp. 164–171.

Summary

This thesis investigates a personalized educational game (PEG) to train people in First Aid. By using artificial intelligence techniques to automate the training processes, learners are able to practice at their own level and in their own pace, even if the instructor is absent. Yet in case the instructor is present, he or she can influence the training process and game environment. The current research tested the design for PEGs by developing prototypes for First Aid training. However, the proposed design facilitates the development of games for other training domains as well.

Lots of professions require people to make good decisions under risky and stressful circumstances. Examples of such professions can be found in the police force, fire department, and military, but also in, for instance, aviation and medicine.

In order to become proficient in these domains, learners need ample practice and experience with situations that are typical and critical for their work. Yet people working in these so-called high-risk professions cannot acquire their experience through learning on-the-job, because in real-life situations erroneous decisions may well result in grievous consequences. To provide learners the experience they need, scenario-based training (SBT) is sometimes used. During SBT learners engage in interactive role-playing exercises, called 'scenarios'. Scenarios are often staged within a simulated environment. They address typical and critical situations and allow learners to experience the consequences of correct and incorrect decisions in a relatively safe and controlled environment. SBT is considered to be a suitable and effective training form for providing learners the experience they need.

Despite its potential, SBT also has its limitations. First of all, SBT requires considerable logistic and organizational efforts. For example, SBT demands finding and preparing a location; assembling and instructing staff personnel and actors to play the roles of the characters in the scenario; and allocating staff to monitor the learner's performance. In addition, it is hard, or even impossible, to alter the course of events in the scenario once it starts playing. This makes it difficult to personalize training. Furthermore, it is often problematic to monitor and interpret events in the scenario in a structured, systematic, and non-ambiguous manner.

Development of new training technology may alleviate the obstacles that prevent ample and effective use of SBT. One major improvement would be to reduce the number of staff personnel currently required for the preparation and delivery of SBT. This may be achieved by automating the activities currently performed by mem-

bers of the staff, for instance by adding artificial intelligence to a virtual (computer-based) environment. An automated system for SBT would allow learners to engage in training more often. Additionally, a virtual environment is easily configurable, thereby enabling adjustments in the course of events from 'behind the scenes', thus increasing possibilities for personalization. Moreover, an automated system requires a formalization of performance standards, which is beneficial for the transparency and consistency of performance assessments.

The envisioned solution for automated SBT, as proposed in this thesis, combines two existing approaches to technology-enhanced learning: 'educational games' and 'intelligent tutoring systems'. The result of this combination is called a *'Personalized Educational Game (PEG)'*. PEGs enable learners to engage in interactive scenarios in a virtual environment. The artificial intelligence of the PEG makes sure that scenarios are automatically tailored to learners' individual needs. As a result, learners can develop their competencies at their own level and in their own pace. The behavior of the characters in the scenario and the events taking place in the virtual environment are also controlled by artificial intelligence, such as planners and intelligent agents. To make learning deliberate and goal-directed, the behavior of the PEG's artificial intelligence is based on well-known educational principles.

The endeavor to make SBT more autonomous and personalized through the development of PEGs, demands a new role for the instructor. On the one hand, the instructor should no longer be needed for a learner to engage in training. On the other hand, however, a PEG must also support and exploit the expertise of instructors by enabling them to, for instance, determine the learning goal, author the scenario, or define the behaviors of the characters. The human instructor is therefore certainly not excluded from or replaced by the PEG, but rather the PEG allows for variable levels of automation, depending on instructors' preferences and availability.

The research presented here investigates 'What is a suitable design for (semi-) automated personalized training with the use of a Personalized Educational Game?'. To investigate this design problem, the situated Cognitive Engineering (sCE) method was used (see Chapter 3). The sCE method closely involves users in the design process from an early stage onward through focus groups, interviews, and workplace analysis. Designs are evaluated through rapid prototyping and human-in-the-loop experiments. Such evaluations not only investigate the human-computer interaction on the interface level, but also aim to include investigations of possible effects on the users' work flows as a result of introducing the new technology in the work environment.

PEGs can be viewed as SBT enhanced with autonomous and personalized support. In order to design the right support, a thorough understanding is needed of the knowledge and processes involved in SBT, like: 'What functionalities are needed to develop an effective and efficient PEG?'; and 'What knowledge is needed to support those functionalities, for instance, about the learner, the training domain, and training & instruction?'; and 'How can these functionalities be integrated in a comprehensive design for a PEG?'.

The design research presented here was carried out in two parts. The first part entailed the design of an overall architecture for PEGs consisting of multiple functional components. The second part entailed several studies into the design of each

of the individual components.

With regard to the design of an overall PEG architecture, an investigation was conducted into the concept and use of SBT and its foundations in cognitive science (see Chapter 4). This resulted in the identification of the following functions necessary to deliver effective scenario-based training: 1) the virtual environment (including the characters), 2) the learner model, 3) the scenario creator, 4) the authoring tool, 5) the scaffolding component, and 6) the reflection component.

Subsequently, a multi-agent organization was developed that fully specifies the coordination and collaboration *between* the components, yet leaves the *internal* structure of the components unspecified. The multi-agent organization also enables an instructor to step in and perform some of the functions (i.e. control one of the components) manually.

To support unambiguous communication between the PEG's components and to enable the artificial intelligence to reason about its tasks and the state of events, an ontology was developed (see Chapter 6). The ontology models all relevant information about the training domain, events taking place in the simulated environment, the behavior of the participating characters, and teaching strategies for effective learning. It is reusable across training domains and applications. Furthermore, it warrants consistency in terminology throughout the system specification. Together, the multi-agent organization and the ontology provide the infrastructure for the components to collaboratively deliver SBT.

The second part of the research consisted of multiple studies into designs for each of the components, being the learner model, the scenario creator, the authoring tool, the scaffolding component, and the reflection component.

Personalization is a crucial aspect in the concept of PEGs. To enable automated personalization, a *learner model* is essential. The learner model keeps track of relevant information about the learner (e.g. competency levels and motivation) and uses that information (1) to automatically select a suitable learning goal and (2) to determine a suitable difficulty level (see Chapter 10).

To offer learners training exercises that fit their needs, the *scenario creator* automatically generates a scenario that targets the selected learning goal at the selected difficulty level (see Chapter 8). The created scenario is (a) representative for the task domain, (b) suitable for the learner's individual characteristics, and (c) includes instructions for assessment of the learner's performance. For this, the scenario creator starts by planning the ideal action sequence for the learner to perform. Then, it selects an appropriate context for these actions and places the necessary objects in the simulated environment to create the conditions for the learner to perform the desired action sequence.

The scenario creator enables the learner to train autonomously in the absence of the instructor. However, although the scenario creator produces scenarios of sufficient quality, a study (see Chapter 8) showed that experienced instructors produce better scenarios than the scenario creator. This finding made us decide to include an option for instructors to monitor, control, and overrule the automated process of the scenario creator by using an authoring tool.

The developed *authoring tool* enables instructors to specify the following properties of a scenario: (1) the learning goal, (2) the setting, (3) the non-player characters

(NPCs) in the scenario, and (4) the narrative goals and attitudes assigned to those NPCs (see Chapter 9). The properties specified by the instructor are then used by the scenario creator to generate a scenario. The authoring tool helps both the PEG and the instructor to benefit from each other's expertise. On the one hand, the scenario creator benefits from the instructor's inputs to generate appropriate scenarios. On the other hand, the authoring tool supports the instructor by providing suggestions for the desired behavior of the NPCs and feeds this back to the instructor. In this way, the instructor and the PEG collaboratively define instructive, consistent, and coherent scenarios.

The generated scenario is then presented to the learner as an interactive, playable storyline in the game environment. In the game environment, the learner can interact with virtual characters which are controlled by artificial intelligence. To ensure that the scenario keeps targeted at learners' individual competency levels, the *scaffolding component* can execute adaptations while the scenario is playing (see Chapter 7). Such adaptations attune the level of challenge or support to a learner's competencies. In order to do this, the scaffolding component is able to intervene in the behavior of the characters and the occurrence of events in the scenario.

The *reflection component* helps learners in acquiring a deeper understanding of the events taking place in the simulated environment. It does so by encouraging learners to reflect on their performance after the scenario has come to an end. In addition, the reflection component prompts learners to provide explanations for their decisions (see Chapter 10). If learners are unable to provide adequate explanations themselves, the reflection component provides hints and additional instructional explanations.

The scaffolding component, scenario creator, and authoring tool were all *evaluated in pilot studies*. Instructors and learners worked with a prototype, answered questions, and were interviewed. In this way, the design was tested for the intended effects. For the scaffolding component, it was found that the adjustments in the behaviors of the virtual characters resulted in scenarios that were better attuned to the learner's individual needs. For the scenario creator, it was found that the produced scenarios had at least the same quality as the scenarios produced by laymen, yet not as good as the ones produced by domain experts. For the authoring tool, it was found that the support function increased the quality and efficiency of the authoring process and the resulting scenarios. Scenario quality was rated in terms of representativeness and suitability for the learning goal and competency level. The designs of the two remaining components (the learner model and the reflection component) still require testing and are therefore presented as 'components in development'.

This research has resulted in a theoretically and empirically founded design for a new form of automated and semi-automated scenario-based training: a personalized educational game. This design aims to optimally utilize the instructor's expertise to create and shape qualitative training. In addition, the PEG design accommodates personalized training when no instructor is available. On such occasions, the artificial intelligence takes control of the training processes.

Samenvatting

Dit proefschrift onderzoekt een gepersonaliseerd educatief spel (PEG) om EHBO-vaardigheden te trainen. Door gebruik te maken van technieken uit de kunstmatige intelligentie kunnen leerlingen in een PEG op hun eigen niveau en in hun eigen tempo oefenen in afwezigheid van de instructeur. Indien aanwezig, kan de instructeur echter ook invloed uitoefenen op het verloop van het spel. Het voorgestelde ontwerp voor PEGs is onderzocht in het domein van EHBO, maar is ook geschikt voor het ontwikkelen van spellen voor andere trainingsdomeinen.

Er bestaan veel beroepen waarin mensen beslissingen nemen onder risicovolle of stressvolle omstandigheden. Voorbeelden hiervan zijn te vinden bij de politie, de brandweer, het leger, de luchtvaart, en de gezondheidszorg.

Om vaardig te worden in zulke beroepen, is veel oefening of praktijkervaring nodig. Echter is praktijktraining in de opleiding tot zulke beroepen vaak niet mogelijk omdat fouten ernstige gevolgen kunnen hebben. Om leerlingen toch in de gelegenheid te stellen hun taken te oefenen, wordt regelmatig gebruik gemaakt van *scenario-gebaseerde training (SBT)*. Tijdens SBT krijgen leerlingen een representatief scenario uit de praktijk voorgelegd in de vorm van een rollenspel. Soms wordt daarbij gebruik gemaakt van een gesimuleerde taakomgeving, bijvoorbeeld een simulatie of een computerspelomgeving. Het is bekend dat SBT een geschikte en effectieve trainingsvorm is om leerlingen praktijkervaring op te laten doen in een nagebootste omgeving.

Hoewel SBT een effectieve trainingsvorm is, heeft het ook een aantal *tekortkomingen*. SBT vereist namelijk de nodige logistieke en organisatorische inspanningen, bijvoorbeeld bij het vinden en gereed maken van een locatie; het verzamelen en instrueren van de rollenspelers, vaak gespeeld door leden van de staf; en het toewijzen van observatoren om de prestaties van de leerlingen te volgen en te beoordelen. Een andere beperking van SBT is dat het in de praktijk lastig is om het scenario tijdens het spel aan te passen aan de momentane leerbehoefte, waardoor de training niet of nauwelijks gepersonaliseerd kan worden.

Er is behoefte aan nieuwe educatieve technologie om de effectiviteit, efficiëntie en flexibiliteit van SBT substantieel te verbeteren. Met methodes en technieken uit de kunstmatige intelligentie zou bijvoorbeeld een deel van de taken geautomatiseerd kunnen worden die nu nog door stafleden worden uitgevoerd. In combinatie met een virtuele omgeving zouden leerlingen bovendien vaker en op een

gepersonaliseerde manier kunnen trainen. Door de vereiste formalisatie van didactische methodes en prestatiebeoordelingen, kan automatisering ook helpen om de kennisbasis van de training te verstevigen (denk daarbij aan het realiseren van transparantie en consistentie in de ondersteuning en toetsing).

In dit proefschrift wordt een ontwerp aangedragen voor geautomatiseerde SBT, namelijk een *gepersonaliseerd educatief spel (PEG)*. PEGs bieden een veilige, gecontroleerde spelomgeving om te oefenen en ervaring op te doen met vaak voorkomende en kritieke scenario's. De scenario's zijn zo ingericht dat leerlingen de gevraagde competenties op hun eigen niveau en in hun eigen tempo kunnen ontwikkelen. De gebeurtenissen in de spelomgeving en de gedragingen van de karakters in het scenario worden aangestuurd door kunstmatige intelligentie, zoals slimme planningsalgoritmen en intelligente software agenten. Bij die aansturing wordt gebruik gemaakt van beproefde educatieve principes. Op deze manier combineert een PEG twee concepten uit de trainingstechnologie, te weten: 'educatieve spellen' en 'intelligente tutor systemen'.

Een autonome en gepersonaliseerde vorm van scenario-gebaseerde training, zoals in een PEG, vergt een nieuwe rol van de instructeur. Enerzijds zou de instructeur niet langer nodig moeten zijn wanneer een leerling wil trainen. Maar anderzijds moet een PEG ook de expertise van instructeurs kunnen benutten, bijvoorbeeld door hen in staat te stellen om een leerdoel te selecteren, een scenario op te stellen, of de gedragingen van de karakters in het scenario te bepalen. Een PEG vervangt de instructeur dus niet, maar ondersteunt verschillende niveaus van automatisering, afhankelijk van de voorkeur en beschikbaarheid van instructeurs.

Dit proefschrift rapporteert over een onderzoek naar de vraag 'Wat is een geschikt ontwerp voor (semi-)automatisch gepersonaliseerde training met behulp van een PEG?'. Voor dit onderzoek is gebruik gemaakt van de *situated Cognitive Engineering* (sCE) methode (Hoofdstuk 3). In de sCE methode worden toekomstige gebruikers van het systeem vanaf een vroeg stadium nauw betrokken bij het ontwerpproces. Dit gebeurt door middel van focusgroepen, interviews, en werkplek-analyses. Van concept-ontwerpen worden prototypes ontwikkeld die beproefd worden met "gebruikers-in-de-loop" experimenten. Daarbij wordt niet alleen gekeken naar de interactie tussen mens en machine op interface-niveau, maar ook naar mogelijke effecten van het ontwerp op de workflow, de werkwijze en de aard van het werk.

Een PEG is te beschouwen als SBT met geautomatiseerde, gepersonaliseerde ondersteuning. Het ontwerpen van de juiste ondersteuning vereist kennis van de processen die nodig zijn voor het geven van SBT, zoals: 'Welke functionaliteiten zijn er nodig voor een effectieve en efficiënte PEG?'; 'Welke kennis is er nodig om die functionaliteiten te ondersteunen, bijvoorbeeld over de leerling, het trainingsdomein, en training en instructie?'; en 'Hoe kunnen deze functies bijeen worden gebracht in een overkoepelend ontwerp van een PEG?'.

Dit ontwerpgerichte onderzoek bestond uit twee delen. Het eerste deel onderzocht het ontwerp van de algemene architectuur voor PEGs, bestaande uit verscheidene functionele componenten. Het tweede deel bestond uit een aantal studies naar de ontwerpen van de individuele componenten.

Ten behoeve van het ontwerp van de algemene architectuur voor PEGs is onderzoek verricht naar het gebruik van SBT in de praktijk en naar wat erover bekend is

in de literatuur (Hoofdstuk 4). Op basis van dit onderzoek zijn de volgende functies geïdentificeerd die nodig zijn om geautomatiseerde SBT op effectieve wijze te verzorgen: 1) de virtuele omgeving, 2) het leerlingmodel, 3) de 'scenario creator', 4) de 'authoring tool', 5) de 'scaffolding component', en 6) de reflectie component.

Vervolgens is een multi-agent organisatie opgesteld die de coördinatie en samenwerking *tussen* de componenten volledig beschrijft, maar die de *interne* structuur van de componenten open laat (Hoofdstuk 5). De multi-agent organisatie maakt het mogelijk dat een instructeur een deel van de processen handmatig uit voert.

Om ervoor te zorgen dat de verschillende componenten met elkaar kunnen communiceren is het nodig dat alle *concepten* van SBT op eenduidige wijze beschreven zijn. Daarnaast is het nodig om de *relaties* tussen deze concepten duidelijk te beschrijven, zodat de kunstmatige agenten kunnen redeneren over hun taken en de gebeurtenissen tijdens SBT. Om de concepten en de relaties ertussen eenduidig te specificeren, is een ontologie opgesteld (Hoofdstuk 6). De ontologie bevat alle relevante informatie over: het trainingsdomein, de gebeurtenissen die plaatsvinden in de virtuele omgeving, het gedrag van de karakters in het scenario, en de lesmethoden voor effectieve instructie. De ontologie zorgt ook voor een consistente terminologie in de systeemspecificatie. De multi-agent organisatie en de ontologie vormen samen de infrastructuur voor de componenten.

Het tweede deel van het onderzoek bestond uit verschillende studies naar de ontwerpen voor de individuele componenten, zijnde het leerlingmodel, de scenario creator, de authoring tool, de scaffolding component, en de reflectiecomponent.

Personalisatie is een essentiële eigenschap van een PEG. Om dit mogelijk te maken is een *leerlingmodel* van cruciaal belang. Het leerlingmodel houdt relevante informatie bij over de leerling (bijvoorbeeld competentieniveaus en motivatie) en gebruikt die informatie om automatisch (1) een geschikt leerdoel voor de leerling te kiezen en (2) een geschikt moeilijkheidsniveau te bepalen (Hoofdstuk 10).

Om leerlingen de oefeningen aan te bieden die ze nodig hebben, genereert de *scenario creator* automatisch een scenario op basis van het geselecteerde leerdoel en moeilijkheidsniveau (zie Hoofdstuk 8). De gegenereerde scenario's zijn representatief voor het taakdomein. Daarnaast bevatten de scenario's aanwijzingen voor het toetsen van de prestatie van de leerling. Het ontwerp voorziet in dit soort scenario's door (1) een actieplan te genereren dat de leerling zou moeten uitvoeren om het leerdoel te behalen; (2) een geschikte context te kiezen voor deze acties; en (3) de vereiste objecten in de gesimuleerde omgeving te plaatsen.

De scenario creator maakt het mogelijk om autonoom te trainen in de afwezigheid van de instructeur. Hoewel een prototype van de scenario creator scenario's van voldoende kwaliteit produceert, liet een studie (Hoofdstuk 8) zien dat ervaren instructeurs nóg betere scenario's produceren. Deze bevinding heeft ons doen besluiten om instructeurs de mogelijkheid te geven het automatische proces van scenario generatie te volgen en te beïnvloeden met behulp van een authoring tool.

De ontwikkelde *authoring tool* stelt instructeurs in staat om de volgende eigenschappen van een scenario te specificeren: het leerdoel, de setting, de karakters in het scenario, en het gedrag van de virtuele karakters in het scenario (Hoofdstuk 9). De scenario creator gebruikt deze input om een passend scenario te genereren. Op deze manier bevordert de authoring tool de samenwerking tussen de PEG en de

instructeur: Aan de ene kant maakt de scenario creator gebruik van de informatie die de instructeur aanlevert via de authoring tool. Aan de andere kant ondersteunt de authoring tool de instructeur door suggesties aan te leveren voor het gewenste gedrag van de virtuele karakters op basis van de gekozen leerdoelen. Zo ontwikkelen de PEG en de instructeur gezamenlijk een leerzaam, consistent en coherent scenario.

Het gegenereerde scenario wordt vervolgens aan de leerling gepresenteerd als een speelbare verhaallijn in de spelomgeving. Daar kan de leerling interacteren met virtuele karakters die worden aangestuurd door kunstmatige intelligentie. Om ervoor te zorgen dat het scenario ook *tijdens het spel* aansluit bij de competentieniveaus van de individuele leerling, kan de *scaffolding component* het scenario aanpassen terwijl het gespeeld wordt (Hoofdstuk 7). Die aanpassingen kunnen bijvoorbeeld als doel hebben om het scenario makkelijker of moeilijker te maken. Om dit te bereiken kan de scaffolding component gebeurtenissen wel of niet laten plaatsvinden of de virtuele karakters in het scenario zich op een bepaalde manier laten gedragen (bijv. wel of geen hulp aanbieden).

De *reflectie component* helpt leerlingen achteraf bij het verkrijgen van inzicht in het scenario. Hiervoor stimuleert de reflectie component leerlingen om te reflecteren op hun prestatie nadat het scenario is afgelopen. Ook moedigt de reflectie component de leerlingen aan om verklaringen te geven voor hun beslissingen (Hoofdstuk 10). Als leerlingen zelf niet in staat zijn om een adequate verklaring te geven, dan geeft de reflectiecomponent aanvullende hints en toelichting.

De scaffolding component, scenario creator, en authoring tool zijn allen *geëvalueerd in pilot studies*. Instructeurs en leerlingen werkten met een prototype, beantwoordden vragen, en werden geïnterviewd. Op deze manier werden ontwerpen getoetst op de beoogde effecten. Een studie naar het effect van de scaffolding component wees uit dat de aanpassingen in het gedrag van de karakters ervoor zorgden dat scenario's beter aansloten bij de individuele leerbehoeften van de leerlingen. Verder bleken de scenario's die afkomstig waren van de scenario creator van een minstens zo goede kwaliteit te zijn als scenario's die door leken geschreven waren, maar niet zo goed als de scenario's die door instructeurs geschreven waren. Een studie naar de effecten van de ondersteuningsfunctie in de authoring tool liet zien dat deze functie leidde tot scenario's met een hogere leerwaarde. Bovendien waren instructeurs sneller klaar met het specificeren van het scenario wanneer zij gebruik konden maken van deze functie. De twee resterende ontwerpen voor het leerlingmodel en de reflectiecomponent moeten nog getest worden en worden daarom in het proefschrift gepresenteerd als 'componenten in ontwikkeling'.

Dit promotieonderzoek heeft een theoretisch en empirisch onderbouwd ontwerp opgeleverd voor een nieuwe vorm van (semi-)automatische scenario-gebaseerde training (SBT): een gepersonaliseerd educatief spel (PEG). Dit ontwerp maakt zo goed mogelijk gebruik van de expertise van de instructeur om goede trainingsscenario's te leveren. Daarnaast stelt het ontwerp leerlingen ook in staat om gepersonaliseerde training te volgen zonder dat daarbij een instructeur aanwezig hoeft te zijn. Op zulke momenten neemt de kunstmatige intelligentie het trainingsproces over.

Curriculum Vitae

Maria (Marieke) Margaretha Magdalena Peeters

08-12-1981	Born in Heerlen, the Netherlands
	Education
1994-1999	High School Studies at Bernardinus College Heerlen, the Netherlands
2000-2006	Cognitive and Educational Psychology (MSc) Faculty of Psychology and Neuroscience Maastricht University, the Netherlands
2006-2010	Cognitive Artificial Intelligence (MSc) Department of Philosophy Utrecht University, the Netherlands
	Work Experience
2014-current	Researcher (Postdoc) Dep. of Intelligent Systems Delft University of Technology, the Netherlands
2010-2014	Researcher (PhD student) Dep. of Information and Computing Sciences Utrecht University, the Netherlands
2003-2009	Occasional Student Assistant / Tutor / Lecturer Subject: Research Methods & Statistics Faculty of Social Sciences Utrecht University, the Netherlands Faculty of Economics and Management University of Applied Sciences, the Netherlands Faculty of Psychology and Neuroscience Maastricht University, the Netherlands