Scenarios in a serious game for training communication skills

Raja Lala

Scenarios in a serious game for training communication skills

ISBN/EAN: 978-90-831713-2-6

This work has been partially funded by the EU H2020 project RAGE (Realising an Applied Gaming Eco-system), http://www.rageproject.eu/, Grant agreement no 644187. This work reflects only the author's view, the European Commission is not responsible for any use that may be made of the information it contains.

This work has partially received funding from the European Institute of Innovation and Technology (EIT). This body of the European Union receives support from the European Union's Horizon 2020 research and innovation programme.

Scenarios in a serious game for training communication skills

Scenario's in een serious game voor het trainen van communicatievaardigheden

(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. H.R.B.M. Kummeling,
ingevolge het besluit van het college voor promoties
in het openbaar te verdedigen op
woensdag 22 september 2021 des middags te 4.15 uur

door

Raja Lala

geboren op 1 maart 1973
te Pune, India

Promotor: Prof. dr. J.T. Jeuring

Contents

Chapter 1

Introduction

Is the term 'serious game' an oxymoron [22]? Or are serious games meant to change the world[1]? The reality is perhaps somewhere in between. Djaouti et al. [22] define a serious game as a piece of software that merges a non-entertaining purpose (serious) with a video game structure (game). The first record of a game using a serious context is Chaturanga (the precursor to the game of chess), from 7th century India, that uses a military context in a board game [82]. The first digital games using a serious context were coincidently also developed for the military [108]. Over the years, serious games have been used in various domains: education, advertising, healthcare, culture, ecology, etcetera.

Mankind is perhaps defined by the ability to communicate with each other. Communication skills as a scientific discipline emerged in the 20th century, largely from three other study disciplines: psychology, sociology, and political science [86]. The term communication skills is often used interchangeably with the terms 'social skills' and 'interpersonal skills' [36]. Hargie [36] defines the skill as: "A process in which the individual implements a set of goal-directed, inter-related, situationally appropriate social behaviours, which are learned and controlled". Hargie divides the study of communication skills in the context of developmental (in children), remedial (redressing inadequacy) and specialised (professional encounters) domains [36]. We are particularly interested in the last context, as most professionals need to interact with a client or with another professional.

Communication skills are designated as one the keys skills for the 21st century [58]. Practice by training on the job or in role play with a simulated patient are effective ways to learn communication skills [8]. Serious games offer the possibility to teach and learn a skill effectively [16]. Serious games for communication skills have been developed in the past two decades e.g. [43, 11, 28]. In these games, a player performs a dialogue with a virtual character to learn and practise a (professional) communication skill in a particular situation, for example 'Aggression De-escalation' [11] when facing intimidating client behaviour. A player chooses a statement from multiple choice options to start/continue a dialogue. A virtual character responds depending on the player choice. At the end of

[1] `https://www.growthengineering.co.uk/10-serious-games-that-changed-the-world/`

a dialogue, a player gets a score and feedback on the particular communication skill(s). In terms of affective computing [84], in these games a virtual character (computer) can express affect (gestures etc.) though cannot perceive affect of a player (human). Attempts to detect affect of a player while playing a serious game to practice a communication skill(s) is a more recent topic of research [102].

The entertainment video games industry is a mature industry, dynamic and growing, in 2013 already worth over $56bn a year worldwide and over $15bn a year in Europe alone [101]. The serious games industry is different from the entertainment games industry and this difference has an effect on the nature of serious games. The difference in these industries can be illustrated by the five forces model of competitive strategy from Porter [87]. The five forces are: the rivalry within firms in the industry, the threat of new entrants, the threat of substitute products or services, the bargaining power of buyers, and the bargaining power of suppliers.

The entertainment games industry is dominated by Nintendo, Sony and Microsoft (referred to as the 'big three' for the rest of this section), with strong competition amongst each other for market share and buyers. The serious games industry is fragmented [101], with small companies focusing on a niche and with low competition amongst each other. The suppliers to the big three are game studios and hardware suppliers for the specific game console. Given the market capital size of the big three companies and the large number of game studios, the bargaining power of suppliers is not high. For each of the big three, a proprietary platform and related ecosystem is desirable as it leads to a customer lock-in for their respective proprietary products. A low bargaining power for suppliers leads to proprietary technology to deliver to the respective big three console platforms. In contrast, serious games mostly run on generic devices. Serious games companies could benefit from reusable and interoperable components on generic platforms to develop a quality serious game with a faster time to market [100]. However, serious games companies often follow a niche strategy and produce customised products, with a low degree of reusability and interoperability [37]. These factors coupled with operating in a niche market, could perhaps be a reason that generic components are not widely available in the serious games industry.

The threat of potential entrants in entertainment games is low owing to the size and economies of scale of Nintendo, Sony and Microsoft leading to a high barrier to entry. In serious games, there is an upfront financial investment risk of entry in a niche market [41]. This risk can perhaps be reduced by lowering the cost of development of a serious game, for instance by using off the shelf components.

The buyers are of a different nature in the entertainment and serious games industries. While entertainment games buyers are mostly consumers, serious games buyers are usually large institutions, for example universities, enterprises or public sector organizations, with the goal to support learning and teaching. Entertainment games is typically a business to consumer (B2C) industry while serious games a business to business (B2B) industry. The bargaining power of buyers determines the profit that can be derived from a product while addressing the price and quality demands [87]. An entertainment game targets a wide audience, while a serious game often targets a narrow audience with specific learning characteristics [41]. In the entertainment games industry, given the economy of scale for the big three, one can argue that an individual buyer does not have much bargaining power. An entertainment game, while addressing quality and price demands of the vast number of retail buyers, is unlikely to be modified to address a particular buyer. In the serious games industry, an institutional buyer can exert bargaining power on a serious games company, for instance to modify/tailor a serious game for a specific audience or need. A teacher/lecturer at a University using a serious game in blended teaching is an example of an (indirect) institutional buyer. Gaming and learning have different mechanics and a teacher using a serious game often needs to balance learning aspects and game mechanics [5]. This need of balancing the 'serious' and 'game' aspects is a major difference to an entertainment game. A teacher can exert influence on a serious games company to modify a serious game to address this balancing aspect.

Finally from Porter's five forces, the threat from substitute products for entertainment games are Indiegames (Independent games), which run on generic devices instead of the big three console proprietary platforms. For serious games, there are a lot of substitute products like MOOC's and Intelligent tutoring systems (ITS); these 'substitutes' are more established and mature compared to serious games. One can argue that a serious game is a substitute itself. An effective manner to gain (or retain) market share for (substitute) products is to demonstrate differentiation and value [87]. It is hence important to evaluate the pedagogical effect of a serious game to establish value [7].

I distil three aspects from these characteristic differences in the entertainment games and serious games industries to consider for a serious game:

- A1: Develop generic components for 'assembling' a serious game, and tools for creating/modifying serious game content. Since serious games run on generic platforms and the barrier to entry is low, off the shelf generic components of good quality enable a faster time to market both for current serious games companies as well as new entrants [41]. A tool enables a teacher to create, modify or analyse the content of a serious game to tailor it to a specific audience.

- A2: Analyse gameplay in a serious game, in particular address the challenge of balancing learning aspects and game mechanics [5].

- A3: Evaluate and improve pedagogical aspects of a serious game [7] similar to evidence based educational intervention efficacy [41].

This dissertation focuses on these aspects (A1 - A3) in the context of Communicate [43], a serious game for practising communication skills. Communicate was initially created for the faculties of Psychology, Medicine, Veterinary medicine, and Pharmacy at Utrecht University, who used it to train their respective students in communication skills. A player, for example a medical student, practices a dialogue scenario with a virtual character, for example a patient. A dialogue scenario is often based on a communication protocol, for example delivering bad news [6]. Communicate was developed iteratively by three successive student project teams at the computer science department at Utrecht University[2]. As Communicate became more widely adopted, a startup company DialogueTrainer[3] was formed, which uses Communicate to offer clients online training in professional conversations. Communicate has since been rebranded as DialogueTrainer. I will use the name Communicate in this dissertation as the initial work and academic papers refer to the serious game as Communicate.

I contributed to two EU funded projects, RAGE (Realising an Applied Gaming Ecosystem[4]), and PrepDoc. The RAGE project addressed some of the challenges in the serious games industry, and one of RAGE's key objectives was to develop a repository of game components to help reduce the cost and risk in developing a high quality serious game [107]. These game components were developed to provide readymade solutions for typical components of a serious game, for example a component to easily control a gesture of a virtual character[5]. PrepDoc aimed to use a serious game environment (Communicate) to prepare an older adult for a consultation with a healthcare professional. In the following sections, I expand on the aspects to consider for a serious game (A1 - A3), in the context of Communicate [43] and the RAGE & PrepDoc projects.

1.1 Generic components and tools

Engström et al. [23] investigate challenges a game writer faces in writing a dialogue scenario for a game. Engström et al. note the absence of a generic tool or format to create dialogue scenarios. A game writer sometimes uses a tool to create a prototype of a dialogue scenario, and play tests it. Engström et al. recommend creating a diverse set of tools to support a game writer in writing a scenario. In intelligent tutoring systems (ITS, a more mature substitute for a serious game), authoring content requires significant effort from an expert [1]. Usability of an authoring tool in an ITS often comes at the expense of expressiveness [73]. Engström et al. find that game scenario development tools (e.g. Twine[6]) often have a scripting programming interface for game mechanics. A game writer focussing on dialogue scenarios may not be proficient in programming [23]. In summary, a game scenario authoring tool should balance usability and expressiveness.

Communicate [43] provides a communication skills expert (hereafter referred to as a scenario author), with an editor for authoring a dialogue. A scenario author, often a non-programmer, uses the expressive constructs in the editor to develop a scripted scenario. A student can thereafter play a scenario and navigate a dialogue with a virtual character to practise communication skills. As part of the RAGE project, we 'carved' out a component for dialogue integration based on the Communicate editor. The research questions in Chapter 2 are: Can we distil common characteristics of serious games with scripted dialogues, use these characteristics to define a format for scenarios and to create

[2] Final year students of the computer science bachelor course at Utrecht University work together in a software project team and develop a software product.

[3] `https://www.dialoguetrainer.com/en/`

[4] `http://www.rageproject.eu/`

[5] `https://www.gamecomponents.eu/content/231/bml-realizer`

[6] `https://twinery.org/`

an authoring tool? We call this authoring tool a scenario editor[7]. A scenario author can develop (numerous) scenarios in the scenario editor, each scenario addressing a particular situation. A game developer can easily integrate a scenario in a serious game, thereby reducing costs and development time.

In a learning environment, the perceived fairness of a result may affect learning [65, 68]. We expect the same holds true in playing a scenario in Communicate, i.e. the perceived fairness of a score (result) on a communication skill could affect learning. A scenario author would like to know if there are potential design issues before a student plays a scenario and gets a score. In the entertainment games industry and also serious games, play testing is often used to detect potential errors before deployment. While useful, play testing requires significant time investment. In the entertainment games industry, a lot of play testers can be recruited, often for free, to evaluate a 'beta' version. Investing significant effort in play testing upfront could represent a barrier for a serious games developer.

As scenarios grow more complex, it becomes more difficult to develop them, and the chance of a design issue increases. There are some ways to support a scenario author, for example Kasperink [48] applies an argument based approach [46] to validate the 'bad news' scenario in Communicate and Gupta [33] uses psychometric techniques to determine the quality of assessment in a scenario. These methods require player data to perform an analysis.

In Chapter 3, we use a concept similar to code smells [24] to describe potential design problems in a scenario. We define a scenario smell as an indicator of a potential scoring problem in a scenario. We develop a tool to signal scenario smells in a scenario to a scenario author. This scenario smells tool can be used already during scenario development, and does not require player data to perform an analysis. The research questions investigated in Chapter 3 are:

- What are the challenges for a scenario author when developing a scenario?
- How can we develop a tool that helps signalling potential problems related to scoring aspects in a scenario?
- What are the results of using the tool on actual scenarios?

We interview key Communicate scenario authors to distil challenges while creating a scenario, develop a scenario smells tool and evaluate this tool using actual scenarios. To take an analogy from software development, the scenario editor is comparable to a Compiler while the scenario smells tool is comparable to a Lint tool.

1.2 Balancing pedagogical aspects and game mechanics

So far, we looked at developing tools to develop and improve a scenario in a serious game. What about gameplay, how can we improve student interaction with a virtual

[7] `https://github.com/UURAGE/ScenarioEditor`

character? In the initial version of Communicate, a player engages in a dialogue with a virtual character by choosing a scripted statement from multiple statement options at a step of a scenario. Research shows that writing open text responses requires higher order cognitive skills compared to selecting from predefined options [65]. Advances in Natural Language Processing (NLP) offer possibilities for open text interactive dialogues with virtual characters [111]. However, the usage of NLP techniques is still scarce in games, as it often requires additional NLP expertise for a game content developer and substantial processing [107].

Lessard [59] describes natural language game conversation based on experience from three digital games. Each of these games uses ChatScript for handling a player's open text input with a virtual character. Lessard finds these game conversations non-linear compared to a scripted dialogue, though there is a player expectation that the virtual character understands and responds like an actual person. For some serious game environments [102, 77], researchers collected a large amount of open text utterances for a specific scenario, thereafter developed a corpus and used this corpus during game play to process an open text input. Allison et al. [2] performed a wizard of Oz study to gather player open text inputs to a virtual character in the entertainment game Minecraft. Allison et al. found commonality in concepts that players used in open text input, classified as imperative, interrogative, declarative, exclamatory and fragment (partial sentence). However, there were significant variations in the specific wording of an open text input. This variation in open text input is a complicating factor for natural language processing and could be an additional reason why the usage of NLP techniques is scarce in games.

A scenario author is an expert/teacher of communication skills and uses her expert knowledge in developing a scenario in Communicate. A scenario is 'rich' in information and the hypothesis is that we can use information within a scenario to create a scenario specific corpus and use this corpus to match open text input from a player during gameplay. We call this the scenario specific corpus method (SSCM) and in Chapter 4, we investigate the following research questions:

- How can we use the information in a scenario to develop a scenario specific corpus?
- How can we match an open text to scripted statements in SSCM?
- How can we use Communicate to create a golden dataset to compare matching an open text to one of multiple scripted statements?
- How does matching using SSCM compare to matching with an NLP method using generic corpora?

In a European Institute of Innovation and Technology (EIT) funded project called PrepDoc, we worked together with NLP researchers from the University Politehnica of Bucharest and the University of Edinburgh. These researchers applied some generic NLP methods to match an open text student input. We compared results of open text input matching using SSCM versus using these generic NLP methods.

A scenario author can use the scenario specific corpus method (SSCM) to offer open text input to a student during scenario play. SSCM does not require NLP expertise from a scenario author nor substantial processing. The next step is to introduce open text input in Communicate and to handle this open text. In introducing a new open text mechanism in Communicate, we considered scaffolding [109], a term used in education, to design support for a student while learning. Some form of scaffolding is also used in entertainment games[8], where a player often gets a prompt at a new challenge, and sometimes a hint if (s)he fails to complete some aspect of game-play. We used SSCM to match an open text input to the possible scripted statement options at a step of a scenario in Communicate. If an open text input from a student could be matched to a scripted statement option, Communicate highlighted the best matched statement and asked a student to confirm the match or choose another statement among the predefined scripted statement options closer to her input. If an open text input could not be matched, Communicate gave a hint. Our research question in Chapter 5 is: Can we handle matched input by highlighting a statement and unmatched input by providing a hint?

1.3 Evaluating and improving pedagogical aspects

Despite the promise of game based learning, there is a shortage of methodical assessment studies for both entertainment and serious games [7]. I am also a teacher in addition to being a researcher. I coordinate and give lectures in the software project course at the Information and Computing Sciences department of Utrecht University. One of the key learning goals of this course for a student is to develop communication skills to work together in a team and a good candidate to use Communicate.

Prior to 2018, we organised a single lecture on teamwork in a classroom session with all students enrolled in the software project course. Since spring-summer 2018, we started with a blended learning session per project team where we gave a workshop on collaboration skills using Communicate. I developed a 'Collaborate' scenario in consultation with a communication skills lecturer and an experienced scenario author from DialogueTrainer. I developed a session evaluation questionnaire together with an expert from the educational sciences faculty. Our goal was to improve the blended learning sessions. We applied the action research method [62] over three semesters, applying an intervention in a session. We applied the same intervention in all the sessions in a semester. After each session, students filled in the same questionnaire. After each semester, we analysed the student feedback and designed the intervention for the next semester. We compared the student feedback from the three semesters, spring-summer 2018, fall-winter 2018 and spring-summer 2019. Our research question in Chapter 6 is: How does student evaluation vary with different interventions in blended learning sessions using Communicate?

1.4 Origin of the chapters in this dissertation

The chapters in this dissertation are copies of published papers, except Chapter 4, which is under submission.

[8] https://www.gamerslearn.com/home/2018/6/11/scaffolding-and-adaptation-2-6m35t-bc6ct

1.4.1 Scenarios in virtual learning environments for one-to-one communication skills training

The paper 'Scenarios in virtual learning environments for one-to-one communication skills training' [55], Raja Lala, Johan Jeuring, Jordy van Dortmont, and Marcell van Geest, has been published in the International Journal of Educational Technology in Higher Education [9]. The paper is included as Chapter 2 of this dissertation.

My contributions included an analysis of the characteristics of scenarios, and providing a classification to represent such scenarios. I performed the analysis through a literature review and by comparing virtual learning environments for scenario based training.

1.4.2 Scenario smells: signalling potential problems in dialogue scenarios in a serious game

The paper 'Scenario smells: signalling potential problems in dialogue scenarios in a serious game' [80], Timo Overbeek, Raja Lala, and Johan Jeuring, has been published in the International Journal of Serious Games, Volume 7, Issue 4, December 2020[10]. The paper is included as Chapter 3 of this dissertation.

My contribution included developing the research idea, defining smells in a published conference paper[11], helping with the development of the scenario smells tool and testing the tool on actual scenarios.

1.4.3 Introducing a scenario specific corpus method to match open text input to scripted statements in a communication skills serious game

Chapter 4 is based on published conference extended abstract/posters in the proceedings of the:

- Semantics and pragmatics of dialogue (SemDial) conference [12]
- Artificial Intelligence in Education (AIED) conference [13]
- Computer supported collaborative learning (CSCL) conference [14]

My contribution included defining the research idea, conducting sessions to gather open text, integrating the different NLP matching methods into Communicate, and comparing the results of the open text matching using the NLP methods.

[9] https://link.springer.com/article/10.1186/s41239-017-0054-1

[10] https://journal.seriousgamessociety.org/index.php/IJSG/article/view/364

[11] https://openlib.tugraz.at/download.php?id=5e6b52f0539f4&location=medra

[12] https://f.hypotheses.org/wp-content/blogs.dir/4280/files/2018/11/paper_1.pdf

[13] https://link.springer.com/chapter/10.1007/978-3-030-23207-8_45

[14] https://ris.utwente.nl/ws/portalfiles/portal/129957345/CSCL_2019_Volume_2.pdf#page=363

1.4.4 Scaffolding open text input in a scripted communication skills learning environment

The paper 'Scaffolding open text input in a scripted communication skills learning environment' [56], Raja Lala, Johan Jeuring, and Marcell van Geest, has been published in the proceedings of the International Conference on Games and Learning Alliance 2019 [15]. The paper is included as Chapter 5 of this dissertation.

My contributions included defining the research idea, conducting sessions for the experiment, and performing the analysis of the results.

1.4.5 Evaluation of interventions in blended learning using a communication skills serious game

A paper, titled 'Evaluation of interventions in blended learning using a communication skills serious game' [52], Raja Lala, Gemma Corbalan, and Johan Jeuring, has been published in the proceedings of the International Conference on Games and Learning Alliance 2019 [16]. The paper is included as Chapter 6 of this dissertation.

My contributions included defining the research idea, and conducting and improving blended learning sessions over three semesters based on the analysis of student feedback.

[15] `https://link.springer.com/chapter/10.1007/978-3-030-34350-7_17`

[16] `https://link.springer.com/chapter/10.1007/978-3-030-34350-7_31`

Chapter 2

Scenarios in virtual learning environments for one-to-one communication skills training

A scenario is a description of a series of interactions between a player and a virtual character for one-to-one communication skills training, where at each step the player is faced with a choice between statements. In this paper, we analyse the characteristics of scenarios and provide a classification to represent such scenarios. The analysis is performed through a literature review and by comparing virtual learning environments for scenario based training. Using this analysis we specify requirements for describing communication scenarios related to their: structure (linear, branching, interleaving), properties (static information stored per scenario like situation, background, which virtual character to show), and parameters (characteristics of a scenario that can be modified per statement like a score on a learning goal and an emotional effect in a virtual character). We define a schema for representing such communication scenarios and present an authoring tool to create a scenario.

Published in International Journal of Educational Technology in Higher Education [1]

[1] https://link.springer.com/article/10.1186/s41239-017-0054-1

2.1 Introduction

One-to-one communication skills are essential in almost all professions. A doctor [98], veterinarian [31], and most other professionals need to appropriately communicate with clients. Many university and vocational programs require communication skills, and train their students in communication skills. A doctor, a psychologist and a veterinarian need to practice a consult with a patient or an owner. A consult varies from gathering the history of a patient to understanding the symptoms of a complaint. An engineer and an IT expert discuss system requirements with a client while a pharmacist advises a customer who comes to a pharmacy to collect medicines for the first time. More generic communication skills such as conflict management, negotiation, and behaving assertively are useful for many professionals. Communication skills are also often trained on the job, at any level: from management (how to deliver bad news), to call center operators (how to convince people to buy a particular product).

Communication skills are best learned through practice, in role-play or with a simulated patient [8]. Feedback has a positive effect in such training situations. Training one-to-one communication skills requires at least three people: two people performing a conversation, and a third assessing the conversation and skills. Such trainings require a substantial amount of time and organisation. The last decade has seen the development of a number of digital environments that support training communication skills. A student performs a conversation with a virtual character, or with a character that appears in videos, and the learning environment assesses the performance of the student. Usually these environments do not replace the traditional real-life trainings, rather these are used to better prepare students for such training sessions, or to offer training possibilities in situations where it is hard to arrange real-life training sessions, or as a complement to such trainings. As far as we are aware there does not exist an effect analysis study on the use of virtual characters in communication skills training. In general, using virtual humans in health care educational programs leads to results comparable to other training methods (and is significantly better than using no intervention) [17].

Learning environments for training one-to-one communication skills offer scenarios that typically occur in real-world situations. A scenario consists of a setting, and a sequence of usually alternating statements between a student and a virtual character. A bad news conversation for a doctor typically involves a diagnosis of a terrible illness, for a manager firing an employee. Although these bad news conversations in different situations follow a similar pattern, the subtleties of a conversation and the responses are very different.

At an abstract level, we can compare scenarios along several axes:

- Does the scenario follow a predefined structure or not?
- What aspects of a virtual character can be controlled by a scenario: utterances, emotions, gestures, ...?
- Are both the choices for a student and the responses from a virtual character completely scripted, or are some of these aspects generated by means of some AI

technique?

- Is it possible to define or use learning goals, and score interactions from a student on these goals?

Scenarios for interactive learning environments are a relatively recent phenomenon. This paper investigates the various variants of scenarios that have appeared in the literature, compares them, and proposes a structure for scenarios to have a common language to describe a scenario. A common structure for scenarios is also useful to provide a means to validate a scripted scenario between a player and a virtual character; for developing a (visual) editor for a scenario that can be used by a domain expert to develop a scenario; and to simulate a dialogue based on a scenario.

This paper is organised as follows. Section 2.2 discusses virtual learning environments that offer scenario based training or assessment. Section 2.3 distils the characteristics from virtual learning environments, proposes a definition for scenarios and presents an authoring tool for creating a scenario. Section 2.4 describes various use-cases. Section 2.5 concludes.

2.2 Learning environments for communication skills

In this section we list the virtual learning environments for communication skills we found in the literature, and describe their main characteristics. Although not our main focus, we also include environments primarily used for assessment.

Vogel et al. [104] define a game as software that has a goal, is interactive, and is rewarding (gives feedback). Interactive simulation activities must interact with a user by offering options to choose or define parameters of a simulation. A user then plays a newly created simulation rather than simply selecting a pre-recorded simulation.

The top-level characteristic for learning environments for communication skills is whether or not, or to what extent, scenarios are scripted. According to Realdon et al. [91], scripting different ad hoc perspectives is a fundamental prerequisite for a narrative structure in order to reproduce both the flexibility and regularity of communicative exchanges. In 'Face-to-Face Interaction with Pedagogical Agents, Twenty Years Later,' Johnson et al. [44] evaluate the evolution of features of pedagogical agents over two decades. They present the possibility of interactive natural language dialogue by combining advances in natural language understanding and dialogue management. They argue that complex technologies can be difficult to understand, control, and author; and that this could stand in the way of acceptance and adoption of these technologies by teachers and other stakeholders.

Although agent based approaches for generating a scenario have made considerable progress in the last decade [9], there are still some problems to solve to achieve realistic simulations in which the degrees of freedom are suitable [72]. In the latter paper, Muller et al mention that an advantage of an agent approach is that the non-deterministic behaviour in an agent approach allows for variability in a game-experience. As a disadvantage, they mention that developing an agent based model requires a greater up-front development effort when compared to scripts. It also requires a stricter collaboration between game designers, domain experts and programmers than with scripting approaches, as agent behaviour is not solely defined by fixed scripts. They mention the need for authoring tools to support game designers in developing content.

Communication skills teachers from our university view non-deterministic behaviour in a communication scenario as a disadvantage rather than an advantage. In teaching a communication protocol like bad news, a consultation, a negotiation, a conflict and the like, communication experts want to precisely control every utterance of a virtual character.

In this paper we restrict ourselves to learning environments in which scenarios are scripted, at least to a large extent. We do not include environments in which scenarios are generated by virtual agents approaches.

2.2.1 Selecting relevant literature

Learning environments for communication skills are used in very different contexts, and collecting the available (descriptions of) environments is non-trivial. We searched for papers on learning environments for communication skills on Scopus by looking at the title, abstract, and keywords using a query and refined the query stepwise.

The first step is a search for use of a scenario in communication skills. This leads to 2000+ results in 27 subject areas ranging from Chemistry to Humanities. The first articles were published in the 1970's; there are 100+ results each year since 2006, the number of results rising steadily till 2013 to 200+ articles each year and a slight decline in 2016 (176 articles).

In a meta-analysis of the cognitive and motivational effects of serious games, Wouters et al. [110] find that serious games were found to be more effective in terms of learning and retention, but not more motivating than conventional instruction methods. We refine our search to the area of serious/applied games. This leads to 668 results in 23 subject areas. The first articles date from the 1980's, and there are already ten articles in 2017.

For this paper we refine the search query to virtual environments and simulations. This leads to 19 results; the first article is from 2008 and there are fewer than 5 articles per year since. The papers are in 9 subject areas: Psychology, Earth sciences, Chemistry, Humanities, Medicine, Mathematics, Engineering, Social sciences and Computer science. Finally, we refine the search further to exclude environments that focus on agent approaches. This search leads to fourteen papers and the trend is the same as in the previous step.

The final search query is:

```
(dialog* OR scenario OR script) AND communication skills
AND (( serious OR applied ) AND gam*)
AND (virtual OR simulat*)
AND NOT agent
```

The use of scripted scenarios in virtual learning environments for communication skills training is a relatively new area used in selected domains.

Besides this query, we asked communication skills teachers from the faculties of Pharmacy, Medicine, Veterinary medicine, and Psychology at our university to provide references to research papers on learning environments for communication skills in their respective fields. After scanning the abstracts returned by our query and the suggestions from the communication skills teachers, we selected nine papers that describe a learning environment with scripted communication skills scenarios [12, 11, 28, 15, 105, 93, 61, 43, 32].

2.2.2 Analysing the literature

The selected papers use varying terms to describe similar concepts, for example: scenario or dialogue, section or subject, statement or fragment or choice. In this section, we consistently use a single term for such a concept. We describe these terms on an abstract level and go into more detail in the following paragraphs per paper.

- A *scenario* is a sequence of interleaved subjects in a context.
- A *subject* is a directed graph of statements within a scenario, usually dealing with a particular theme.
- A *statement* is a piece of text.
- An *authoring tool* is software that allows an author (usually a non-programmer) to create a scenario.

Bracegirdle et al. [12] present a ‘programmable patient’, with a ‘brain’ that is essentially a scenario in our description. Clinical educators can add subjects to a scenario. A student asks or selects questions that correspond to nodes in a tree. If a question is not recognized, a student can choose to store the question together with an answer in this tree. Scenarios are not completely scripted: questions corresponding to nodes can be asked in various ways. A scenario does not affect the emotion of a programmable patient (virtual character) in this game. A programmable patient is standardised in comparison to an actor and this leads to equity in assessment.

Bosse et al. [11] use a scenario in which a student chooses between several statements. In addition to statement choices for a student, a statement of a virtual character may also have choices. The latter choices depend on values of emotional parameters, which in turn may vary for different characters. A statement choice of a student influences the value of these parameters. The main technology in the implementation is a scenario with conditional branches, in which emotional parameters are used. Students find this game an effective learning tool.

Gebhard et al. [28] present Visual SceneMaker, an authoring tool for creating interactive applications with multiple virtual characters. SceneMaker separates content, such as statements, from logic, such as conditional branching. Content is specified in a scenario in the form of statements and stage directions for controlling gestures, postures, and facial expressions. The logic of an interactive performance and interaction with virtual characters is controlled by a sceneflow. A sceneflow is a hierarchical scenario with nodes that structure content of an interactive presentation and edges that specify how this content is linked together. Visual SceneMaker is evaluated in experiments; the visual programming approach and modelling structures are comprehensible and let non-experts create virtual character applications in a rapid prototype fashion.

Claudio et al. [15] demonstrate Virtual Humans (VH) in an interactive application, the goal of which is to train and assess self-medication consultation skills. A domain expert develops a scenario using an authoring tool called Dialogue Creator. A scenario is a graph implementation and consists of a sequence of statements between a VH patient and a player, with associated scores per answer. Each node in a graph corresponds to a statement that is articulated by a VH or a statement option for a player and its associated score. Seven pharmacy experts performed an evaluation of the application. These experts consider this application a valuable tool for training and assessing the over-the-counter counselling skills of a player, and recommend to follow such training with real interactions and live evaluation.

Leary's Rose [105] focusses on natural language recognition of player sentences. Input sentences are classified on two axes: the vertical axis indicates whether a speaker is dominant or submissive towards a listener, and the horizontal axis indicates a speaker's willingness to cooperate with a listener. A scenario engine selects an appropriate response to a player's input on the basis of identified keywords and relative positions of both player and Virtual Agent on the axes. A player's input is fed to a finite state machine (FSM) and if an input matches a state, the FSM selects a reply from the available follow-up states.

Rety et al. [93] present a framework for interactive narrative authoring in the domain of virtual storytelling where a user is provided with navigation choices in an interactive narrative. The framework is based on a generalised concept of a section, which is close to a subject in our description. A subject consists of hierarchically structured fragments (a statement in our description). A statement is defined as an atomic content element containing simple text and possibly other media. A statement may also be composed recursively and consist of a set of statements, a behaviour and a termination property. Behaviour can be set by an author as either deterministic to indicate that a user reads in the order of subjects or as non-deterministic to indicate that a user reads with no predefined order. A termination property is a condition that an author sets to determine when a subject terminates. A user then moves on to a following subject. A user reads an interactive narrative created in this framework by successively accessing statements in a subject.

Leuski and Traum [61] describe a Non Player Character (NPC) editor, which includes natural language understanding and generation, and statement management. This authoring tool is designed primarily to develop question answering characters. The focus is to deliver an appropriate answer for a user's question. An author creates a response handling strategy by populating a language database with specific answers to sample questions and assigning these answers to an individual character. Different characters may have different knowledge about an event and respond differently to the same question. At run-time, a statement manager matches a user's question to an answer of an associated non player character. NPCEditor has been evaluated and is an effective and versatile system in applications for virtual question answering characters. As future work, the autors mention dialogue management as a specialized component.

Jeuring et al. [43] describe Communicate, a serious game for practicing communication skills. An important aspect of Communicate is the de-coupling of scenario development by communication skills experts from the implementation. A communication expert iteratively develops a scenario as a directed acyclic graph of statements in an authoring tool. A statement has an incremental score, emotional effect, and feedback. Statements can be structured under subjects. A scenario parser uses this graph to generate a scenario specific reasoner. During gameplay, a player receives information about statement choices at each step in a series of interactions with a virtual character. At the end of a simulation a player receives a final score on communication skills.

Guo et al. [32] present a game prototype for practicing communication skills in a medical consultation process. The communication skills module accesses knowledge content and a scenario through a restful web service. Knowledge is modelled as rules, which determine which player actions have a positive or a negative effect during a medical consultation, as well as the influence of each type of action on a patient's stress level. A scenario models a virtual patient with some key characteristics, which are chosen from the knowledge content. A scenario also defines possible actions for a player with a virtual patient in each phase of a medical consultation process.

2.2.3 Dialogue authoring assets

In addition to the literature review, we looked at available dialogue authoring tools by searching in the Unity asset store (`www.assetstore.unity3d.com`) and asking game developers from the RAGE (Realising an Applied Gaming Eco-system: `www.rageproject.eu`) project for recommendations. From the selection, we looked at:

- Dialoguer (`www.dialoguer.info`)
- Dialogue System (`www.pixelcrushers.com`)
- Dialogue (`www.ninjapokestudios.co.uk`)
- Articy (`www.nevigo.com`)
- Chatmapper (`www.chatmapper.com`)
- Twine (`twinery.org`)

These authoring tools concentrate mostly on the dialogue aspect of a scenario used in entertainment games. They range in functionality from simple tools offering basic dialogue sequences to advanced tools. We identify three features relevant to a scenario in these tools.

- The tools support the construction of a sequence of statements representing a dialogue flow from a player to a computer and vice-versa.

- They provide the possibility for a player to have multiple choices as a response to a computer question or a situation.

- Many tools use preconditions; a particular dialogue fragment is only reached if a certain condition is fulfilled. For example, a player must have completed a certain task prior to reaching a particular dialogue fragment.

Articy is a visual environment for the creation and organization of game content offering virtual characters, locations and a game development environment. Chatmapper has a basic free version available. Twine is an open-source tool for creating interactive, non-linear stories and resembles Interactive narratives [93].

2.3 Characteristics of scenarios

In this section, we extend the description of a scenario given in the previous section to more precisely define what we mean with a scenario. We formalise this definition in an XML schema, which we can use to validate a scenario. We present a scenario editor using which a scenario author can create a scenario.

2.3.1 Scenario definition

The majority of the papers in the literature [11, 12, 15, 28, 43, 93] describe a communication exchange between a player and a virtual character as a tree, graph or chart. Communication flows from a virtual character to a player and vice-versa. Player and virtual character statements are represented as nodes and the flow of conversation is represented as an edge with a direction. A dialogue proceeds from one statement to another between characters (player and virtual) and flows forward. Such a communication exchange can be represented by a directed acyclic graph (DAG). A DAG consists of a finite set of nodes and edges between nodes, with no cycles. We define a communication exchange, often dealing with a particular *subject*, as a DAG. Characteristics of a subject DAG, as found in commercially available dialogue authoring assets/tools and in the literature [11, 15, 43, 93], are:

- multiple player nodes to represent player statement choices.

- a precondition in a node.

Bosse et al. [11] present a game for aggression de-escalation training with a scenario simulation in a public transport tram. Bracegirdle et al. [12] and Guo et al. [32] present simulations in clinical communication skills which play in a hospital and a clinic respectively. Claudio et al. [15] present a game which plays in a pharmacy and allows student engagement in a true-to-life situation in a controllable environment. Jeuring et al. [43] present a simulation where a teacher can select from virtual characters and backgrounds in a context of a scenario. According to Realdon et al. [91], an adequate scaffolding structure for the training of communicative competence in an e-learning environment entails fixing both contextual boundaries and degrees of freedom to let the learner have the opportunity to give sense to the perspective selected. We infer that a communication scenario has a context in which a student/user plays. Three possible components of a context are for example a location, a background image and a hair color for an interactive virtual character. We define these components as *properties* for a scenario that provide context during gameplay. A *property* is a static piece of information for a scenario, a statement or a virtual character. An example of a *property* for an entire scenario is a location, of a *property* for a statement is an intent of a player, and of a *property* for a virtual character is a name.

Since we search for literature on serious or applied games, most papers [11, 12, 15, 28, 32, 43, 105] mention learning goals. A player's choices determine his/her score on these learning goals. A player can get feedback either during gameplay following a player choice or at the end of a scenario. In some of the games [11, 43, 105] an emotional state of a virtual character can change depending on a player statement choice. In papers that describe an authoring tool for scenarios [11, 28, 93, 61, 43], a scenario writer may change such an emotional state or progress towards a learning goal by defining an effect on a parameter at a statement. We define a *parameter* as a dynamic piece of information of a scenario that can be modified per statement. Examples of *parameters* are a character's emotion and a score for a scenario.

In interactive narratives [93] a user may read subjects in a hierarchy without any predefined order. To overcome predictable behaviour in a conversation with an agent, agents in conversational trees [11] are endowed with an internal state to introduce variability in statements. In Communicate [43], interleaving enhances expressiveness in a communication scenario by allowing students to navigate a conversation with a virtual character through subjects. If the order in which statements within two subjects (or more) are navigated is not important, then these subjects are interleaved. We infer that variability in navigating subjects in a scenario is important to add expressiveness to a scenario and partly overcomes predictable behaviour. We characterise this variability for two (or more) subjects as interleaved in a scenario where a player gets statement choices from within these interleaving subjects without a predetermined order. We define a scenario as a sequence of interleaved subjects.

Summarising, in our definition, a scenario is a sequence of interleaved subjects, where each subject is a DAG. In a scenario static information is represented by properties, and dynamic information by parameters. An overview of which concepts appear where in the selected papers and tools is depicted in Table 2.1.

2.3.2 XML schema for validating a scenario

We define the Utrecht University Dialogue Scenario Language (UUDSL) as an XML language for describing the structure of a scenario including a dialogue structure, virtual character(s), properties (e.g. location, character name) and parameters (e.g. emotion, score).

It is schematically represented in Fig. 2.1

Papers	Scenario structure	Parameter	Property	Dialogue variability
Bosse et al	✓	✓	✓	✓
Bracegirdle et al	✓	✓	✓	
Claudio et al	✓	✓	✓	
Gebhardt et al	✓	✓		✓
Guo et al		✓	✓	
Jeuring et al	✓	✓	✓	✓
Leuski et al	✓	✓		
Realdon et al	✓		✓	✓
Rety et al	✓			✓
Vaasen et al		✓		✓
Tools				
Dialoguer	✓	✓		
Dialogue system	✓	✓		
Dialogue	✓			
Articy	✓	✓		✓
Chatmapper	✓	✓		
Twine	✓			✓

Table 2.1: Which concepts are used where?

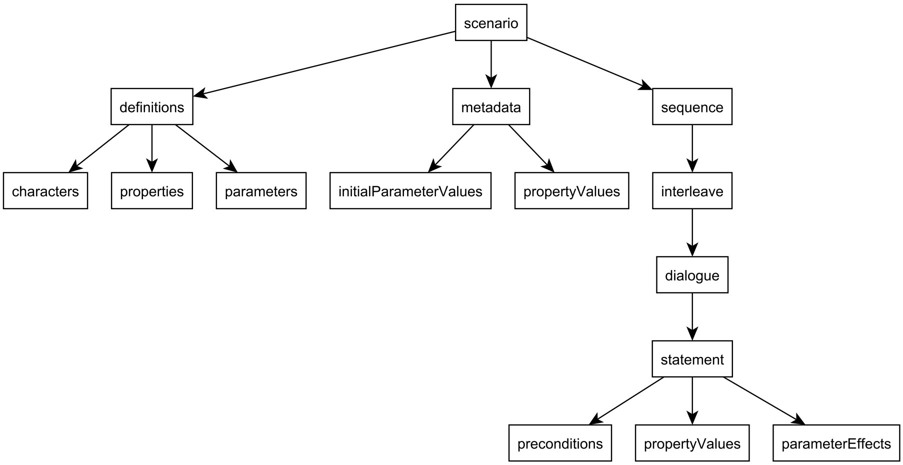


Figure 2.1: Schematic representation of the UUDSL

The top-level element contains definitions, metadata and an element for sequencing interleaved subjects. Characters, properties and parameters are defined inside the definitions element for reference in the rest of the dialogue scenario. A property is a static piece of information with a scope. A property scope can be per-statement or independent of statements, which evaluates to a piece of information about a scenario. A parameter is, in contrast to a property, a dynamic piece of information and is by definition per-statement. A statement can change the value of a parameter. A character has an ID and a set of properties and parameters.

The metadata element contains all data that describes a scenario, but is not part of it (e.g. name, description). The metadata element also contains property values and initial parameter values. These initial parameter values are used to initialise a scenario's parameters during game-play. Property values stored in metadata define properties of a scenario.

The sequence element contains a sequence of interleaved subjects. The interleave element delimits the interleaving dialogue elements, which represent a piece of dialogue about a subject. Inside a dialogue element there are statements representing the contents and effects of the dialogue. A statement element contains text, preconditions, property values, parameter effects, links to possible response statements and whether the statement belongs to a player or computer.

The UUDSL can be accessed at: `https://uudsl.github.io/scenario/`

2.3.3 An authoring tool for creating a scenario

In this section we present a scenario editor, an authoring tool in which a communication expert (an author) iteratively develops a communication scenario as a sequence of interleaved subjects, consisting of statements with parameters per step. This editor has been developed in collaboration with communication teachers from our university and external users, and runs in a web-browser.

The editor has the following basic features:

- Dialogues are structured in subjects.
- An author develops a subject in a scenario as a DAG of statements between a virtual character and a player. The subject represented by a DAG starts at the top and ends at the bottom.
- Within a subject, an author develops a dialogue between a virtual character (computer node) and a player (player node).
- An edge from a computer node to a player node (and vice-versa) depicts a flow of conversation.
- An author connects a computer node to multiple player nodes to create statement choices for a player.

- Each player statement choice leads to an incremental score, and affects for example an emotion in a virtual character.
- A statement can have a precondition.

Subjects on the same horizontal level are interleaved. Fig. 2.2 shows an example scenario, which consists of 5 subjects. 'EatingOut', 'Cooking', and 'Take away' are interleaved subjects. This indicates that a player will get statement choices from within these subjects in no particular order during a simulation of this scenario. She can navigate these subjects in any order in a simulation.

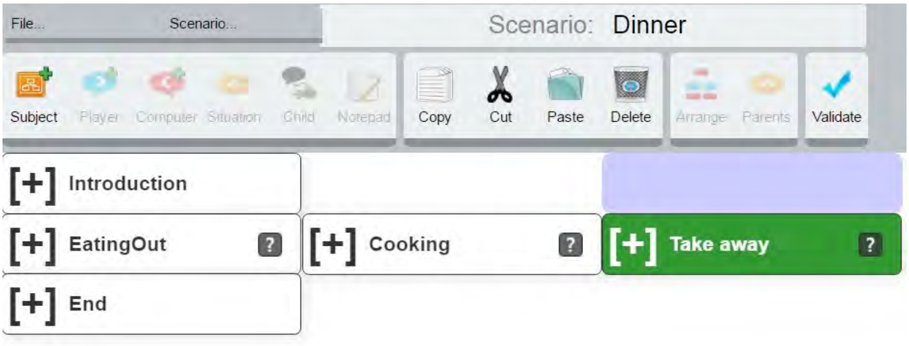


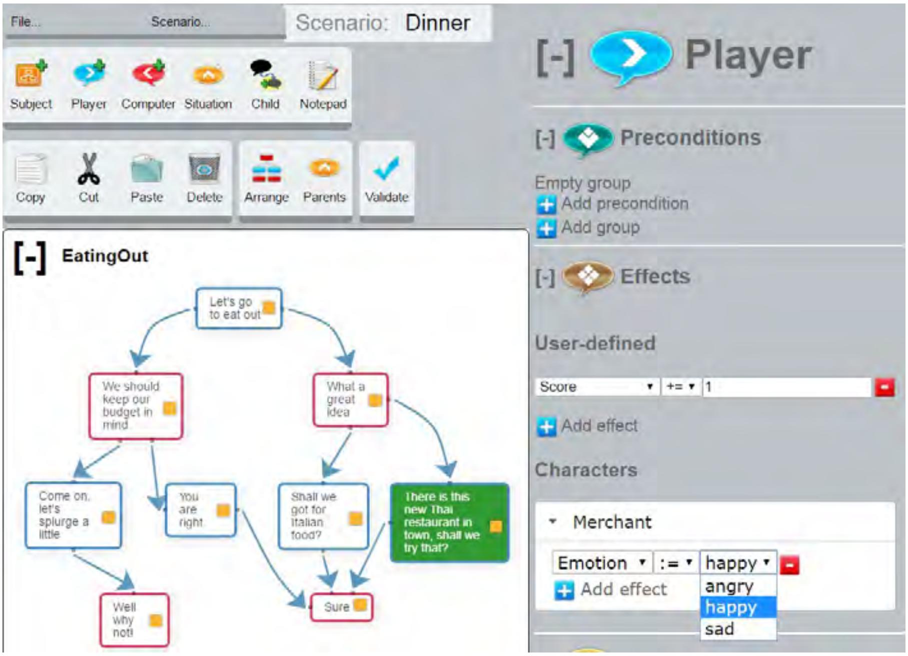
Figure 2.2: Example scenario subject level

A subject may be marked as optional, depicted in Fig. 2.2 with a question-mark, which means that the subject can be skipped altogether in a simulation.

The scenario editor is configurable: a game developer can specify a virtual character(s), properties (e.g. name of a character) and parameters (e.g. emotion and gesture) specific to a game. We provide a configuration XML schema to create a game-specific instance of the authoring tool. A scenario author, usually a non-programming expert, can use this instance of the scenario editor to create a scenario(s) specific to a game. The resulting scenarios and their associated properties and parameters can be used by the game developer for the game.

In an example configuration, the type of a property such as location can be specified by a game developer as an enumeration with the elements town, village and city. This enables a scenario author to select a location from the enumeration as a context for a scenario. The name property of a virtual character can be specified by a game developer as a value of the type string, which allows a scenario author to define the name of the character. The emotion for a virtual character can be specified by a game developer as an enumeration parameter consisting of elements angry, happy and sad. This parameter enables a scenario author to create an effect on the emotion of a virtual character at a statement. Additionally, a scenario author can also define a user parameter for a specific scenario, for instance a score as an integer, which can be set at a player statement. Fig. 2.3 also displays the parameters of this example configuration of the authoring tool.

Fig. 2.3 below shows an example of a DAG for a subject (example EatingOut subject from Fig. 2.2).



Scenario: Dinner

[-] EatingOut

Let's go to eat out

We should keep our budget in mind

What a great idea

Come on, let's splurge a little

You are right

Shall we got for Italian food?

There is this new Thai restaurant in town, shall we try that?

Well why not!

Sure

Empty group

Score += 1

Merchant

Emotion := happy

angry

happy

sad

Figure 2.3: Example subject DAG

Summarising, our authoring tool:

- is developed from a communication teacher's (non-programmer) perspective. Existing tools (reviewed in Section 2.2.3) are primarily developed for game-developers (programmers). Our tool leads to de-coupling of scenario content from programming and a domain expert can develop and modify a scenario independent of a game developer.

- runs in a web browser as opposed to existing client-based tools. This makes it more easily accessible to scenario authors.

- is (easily) configurable to the needs of a scenario author or a game-developer.

- creates a scenario that can be validated by using a schema.
- develops a scenario as a sequence of interleaved subjects. This abstraction of a scenario (often following a communication protocol) in subjects is unique.

Our authoring tool is open-source and can be accessed at: `https://github.com/UURAGE/ScenarioEditor`

An author can export a scenario from the authoring tool. The exported scenario is an XML document based on the UUDSL and can be validated using the UUDSL Schema. An exported scenario XML is integrated in a game.

2.4 Use cases

This section describes how our UUDSL and authoring tool can be used in various applications.

Our authoring tool is used with a serious game as part of a learning environment for practising communication-skills (`https://communicategame.nl/`) at our university. This environment is used by over 20 communication teachers and teaching-assistants from the faculties of Psychology, Pharmacy, Medicine and Veterinary science as part of their curriculum to teach communication skills to their respective students. This environment is also used by a training company, who trains clients in communication skills. An example client is a foundation that trains healthcare professionals working in poor neighbourhoods.

We work in a broader context in the RAGE (Realising an Applied Gaming Ecosystem) project (`www.rageproject.eu`). RAGE brings together three kinds of stakeholders: game component developers such as universities and research institutes; game companies who integrate game components to form an applied game, and users of applied games, universities, colleges or employment and training agencies who wish to deploy applied games in education and training of students or personnel. Our editor is one of the RAGE components.

Our editor asset is used by one of the RAGE pilot games. In this game, a player works in a virtual game studio as an office assistant. Her goal is to increase the success of the virtual company by improving the social skills of a virtual team. During game-play, a player can visit a water cooler in the virtual company to converse with different virtual characters. A dialogue starts with a statement from a virtual character for group working/conflict management behaviour and a player can choose to reply from multiple-choice statements. A teacher/communication expert from the college scripts these dialogues using a configured instance of our editor asset. The college uses a pedagogical model based on the Thomas-Kilmann Conflict Mode Instrument [49]. A dialogue requires a player statement choice to have an intent. This intent is configured as a property of a statement with values: Competing, Accommodating, Avoiding, Collaborating or Compromising. A game development company integrates these dialogues in a game. The game will be piloted and evaluated by students.

Another RAGE asset is a virtual human controller that uses BML (Behavior markup language) input to control aspects of a virtual character, for instance face emotion and arm gesture. We configured an instance of our editor to specify an emotion and a gesture for a virtual character. Both are defined as a parameter of type enumeration. We integrated the assets in a simple Unity game that plays a scenario from our asset with a virtual character from the other asset. Initially, the other asset supported three emotions: happy, angry and sad. Later 'surprised' was added and it was easy to add an element to the enumeration parameter in our editor. We can modify and play a scenario without changing the software of the game.

We analyse how our authoring tool can be used for some of the games described in the reviewed papers. We sketch a configuration of the scenario editor to enable a scenario author to create a scenario for a game in the reviewed papers. For Bosse et al. [11], the editor configuration consists of a single virtual character; parameters: a score (integer) and feedback to a player (string); properties: a context (a string) with value aggression de-escalation, and a background-image with as value the image of a tram. For the pharmacy communication simulation of Claudio et al. [15], the editor instance has a single virtual character, a scoring (integer) parameter, and a background-image property with as value a picture of a clinic. Finally in the simulation from Guo et al. [32], the editor instance offers a single virtual character, a background-image property with as value a picture of a medical consultation room, and a scoring parameter.

2.5 Conclusion and future work

Communication skills are needed in the majority of professions and are best learned through practice. These skills may be developed in learning environments that simulate real-world communication scenarios, for instance a bad news conversation. We analyse the literature and contemporary simulations and tools for common characteristics of communication scenarios. Characteristics of communication scenarios are their structure, properties, and parameters. We propose a schema that models the various aspects of communication scenarios. We present a configurable authoring tool to create a scenario. We can use this tool to construct scenarios for various applied games.

RAGE aims to deliver five pilots of applied games. Our asset is also planned to be used in two other games; one intended for a job-seeker to practice interview skills at an employment agency and another for a student to practice a business negotiation. In the context of Rage, we plan to investigate hybrid approaches to dialogue management in applied games together with INESC [21]. In our methodology, we handle dialogue management in a scripted, centralised manner. In a distributed agent approach, a dialogue emerges from individual decisions that are made by a virtual character according to their internal Artificial Intelligence. We can experiment if a hybrid approach leads to more variability during game-play while preserving the consistency and other advantages of a scripted approach as described in Section 2.2.

We also want to determine schema extensions that enable us to translate a high-level communication protocol to communication scenarios. An example of such a protocol is the protocol to hand out medicine to a customer who visits a pharmacy for the first time. A scenario based on this protocol varies depending on aspects such as the side-effects of the medicine, and the name of a character. Another example of a protocol is a bad news conversation, which for a doctor involves telling a patient about a diagnosis of a terrible illness, for a veterinarian telling an owner of a pet about a diagnosis, and for a manager firing an employee. It is a research question how we best can pass arguments to communication protocols.

2.6 Acknowledgements

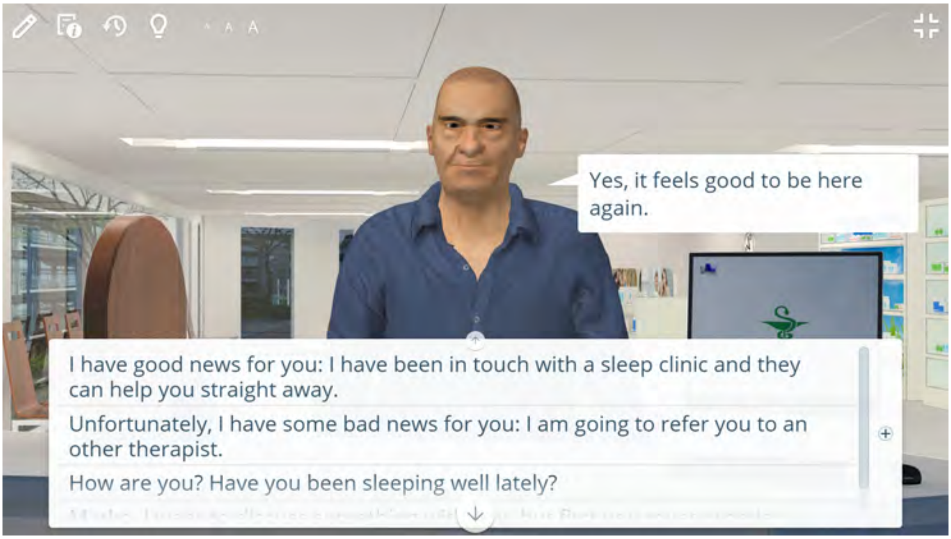
This work has been partially funded by the EU H2020 project RAGE (Realising an Applied Gaming Eco-system, `http://www.rageproject.eu/`), grant agreement no 644187. This paper reflects only the author's view, the European Commission is not responsible for any use that may be made of the information it contains.

Chapter 3

Scenario smells: signalling potential problems in dialogue scenarios in a serious game

Many serious games employ a scripted dialogue for player interaction with a virtual character. In our serious game for one-to-one communication skills training Communicate, a scenario author develops a structured, scripted scenario as a sequence of interactions between a player and a virtual character. A player is often a student learning communication skills and a virtual character represents a person that a player talks to, e.g. a patient. A player gets a score on her performance after playing a scenario. Scoring is an important aspect of a scenario. As scenarios get more complex, assigning scores in a scenario also gets more complex, and the risk of an incorrect design increases. To determine what kind of challenges scenario authors experience when assigning scores in a scenario, we conduct a structured interview study with a focus group of scenario authors who use Communicate. Based on these challenges, this paper introduces the concept of scenario smells, and investigates how we can support scenario authors in detecting and addressing such scenario smells. A scenario smell is a symptom of a scenario that could be an indicator of an error or incorrect design in the scenario. We develop a tool that supports a scenario author by identifying scenario smells in a scenario in Communicate. Scenario authors evaluate the tool and find most of the scenario smells useful.

Published in:
International Journal of Serious Games
Volume 7, Issue 4, December 2020
http://dx.doi.org/10.17083/ijsg.v7i4.364



Yes, it feels good to be here again.

I have good news for you: I have been in touch with a sleep clinic and they can help you straight away.

Unfortunately, I have some bad news for you: I am going to refer you to an other therapist.

How are you? Have you been sleeping well lately?

Figure 3.1: Screenshot playing bad news scenario in Communicate

3.1 Introduction

Many serious games offer a player the possibility to perform a dialogue with a virtual character [11, 12, 15, 28, 32, 43, 63, 102]. Some of these serious games, such as Communicate [43] and Visual SceneMaker [28], provide a learning environment, where a scenario is created in an authoring tool and thereafter played in a game simulation. A scenario author can create several scenarios in these learning environments. Most of the other serious games offer a single scenario to a player. In all these serious games, a player performs a dialogue with a virtual character (VC) by choosing a statement from multiple statement options, see for example Fig. 3.1. Choosing from multiple statement choices resembles a multiple choice test in some ways. After playing a scenario, a player gets a score depending on the choices she made while playing the scenario. A player might play a scenario game simulation several times, trying out different statement options at a step of a scenario, to find out how a VC responds to a particular statement, or to obtain a better score. How do we know that a score given by a game simulation is of good quality? Can we help a scenario author in verifying that the various scores awarded by the game simulation to playing a scenario reflect the intentions of the scenario author?

A scenario in these games is a graph, and a dialogue is a path in this graph of alternating player statement choices and corresponding VC responses. As scenarios become more complex, it becomes more difficult to develop them. One challenging aspect when developing a scenario is to ensure that the scores awarded in a scenario properly reflect the quality of the choices of a player in a scenario. For example, a scenario to practice collaboration skills could have as a goal to balance result- and relation-orientation in raising a difficult issue with a team mate. After playing this scenario a player gets a score percentage on result- and relation-orientation. Ideally, all score combinations are possible (low-low, low-high, high-low, high-high). However, if a player can only obtain low-low or high-high scores in this scenario, the scorings on these aspects are probably correlated. Possibly one of the scoring aspects is superfluous, which is probably not the intention of a scenario author who wants to balance result- and relation-orientation. Correlation of the scoring aspects result- and relation-orientation could be a potential quality problem in this scenario. Another problem might be that it is impossible to obtain high scores on both aspects, in which case a student might consider the assessment in the scenario unfair. Perceived fairness of a multiple choice test affects learning in a student [65, 68]. Correlation of scoring aspects, or the impossibility to achieve high scores on all scoring aspects are indicators of potential quality problems of a scenario. We define a scenario smell as a symptom of a potential scoring problem in a scenario. How can we help a scenario author with detecting and addressing such scenario smells?

This paper addresses the following research questions:

- RQ1: What are the challenges for a scenario author with respect to scoring aspects when developing a scenario?
- RQ2: How can we develop a tool that helps signalling potential problems related to scoring aspects in a scenario?
- RQ3: What are the results of using the tool on actual scenarios?

To investigate scenario smells, and to develop tooling to support signalling them, we need a learning environment that is used by multiple scenario authors to develop their own scenarios, including the scoring of dialogues in the scenario. We choose to investigate scenario smells in the context of the learning environment Communicate. Communicate is widely adopted: more than twenty communication skills teachers (scenario authors) from various universities and faculties use this learning environment in teaching communication skills at several universities. A startup company DialogueTrainer[1] uses it to train their clients. We expect that Communicate is one of most used learning environments for performing dialogues with VCs: the scenario database of Communicate contains more than two thousand (variants of) scenarios, and the amount of users runs in the tens of thousands. Although we answer our research questions in the context of a single learning environment, the results are to some extent generalisable. A scenario in the Communicate learning environment is created in a separate authoring tool, and thereafter played in a game simulation. There is an open source version of the authoring tool[2],and a published scenario language definition. This allows for an independent development of a tool for analysing scenario smells, separate from the Communicate development environment.

[1] https://www.dialoguetrainer.com/en/

[2] https://github.com/UURAGE/ScenarioEditor

The open source authoring tool has not only been used to develop scenarios for Communicate, but also for the serious games JobQuest[3] and WaterCooler[4]. WaterCooler is a game where a student learns to develop team building skills. Jobquest is a game where a jobseeker can practice interview skills with a virtual character. For both of these games, game writers developed scenarios using the authoring tool.

To answer RQ1, we conduct structured interviews with six scenario authors using Communicate. These six interviewees include three scenario authors from three different faculties (medicine, pharmacy and veterinary science respectively), one teaching assistant and two scenario authors from the startup company. These interviews lead to formulating several scenario smells. We answer RQ2 by developing a smells tool signalling the scenario smells that the scenario authors deem most important. The smells tool takes a scenario as input and generates a scenario smells report. To answer RQ3, we evaluate the smells tool with four scenario authors using actual scenarios used in practice.

Most of the results of this paper come from an MSc thesis [78]. Some results of the thesis have been published in a conference paper [54], which mainly describes the research problem, and a technical report [79], which describes the initial interviews.

Section 3.2 describes related work. Section 3.3 describes the research context, and describes previous work on Communicate. Section 3.4, 3.5, 3.6 answer the research questions RQ1, RQ2, and RQ3, respectively. Section 3.7 summarises and discusses the answers to the RQ's. Section 3.8 presents the conclusions and future work.

3.2 Related work

Belotti et al. [7] investigate assessment in serious games. Game play based assessment can provide detailed assessment and be tailored to a player as opposed to a standardised test. Belotti et al. advocate in-game assessment as it does not break a player's game experience. Belotti et al. also find that the proper design of assessment in a game is a challenging and time consuming activity and advocate for supplementary assessment tools.

The concept of providing smells as an indicator of a potential problem to an author (developer) is used in several other disciplines, the most notable being code smells [24] in software development. A code smell is not necessarily an error, but a symptom of a potential quality problem or perhaps a design issue. A code smell is often an indication to investigate if code needs refactoring. An example of a code smell is duplicate code: identical or very similar code that appears in two or more locations in the code base of a piece of software. Another kind of smell are usability smells: Almeida et al. [3] present a catalogue of six usability smells as indicators of a user interface (UI) design and maintenance problem. They interview five programmers, and find that most interviewees agree that the proposed usability smells point to potential problems, and that the proposed refactorings lead to improved usability. Almeida et al. also present a tool extension to automate usability smell detection.

[3] `https://www.gamecomponents.eu/content/598/job-quest`

[4] `youtu.be/d_irGrEpBdc`

Engström et al. [23] describe the challenges faced by a game writer in developing a narrative scenario for a game. Engström et al. mention the limited interest in dialogue/narrative/story writing for games in academia. Game-related academic research often focusses on theoretical reasoning and pedagogical content [45, 50] rather than on more pragmatic game writing. Engström et al. note the absence of a generic tool or format to create narrative scenarios, and mention that a game writer often uses Excel to communicate a dialogue to a game programmer. A game writer sometimes uses a tool to create a prototype of a narrative scenario, and play test it. Engström et al. compare these tools, and find that most tools have a scripting programming interface. They recommend creating a diverse set of tools to support a game writer and present a prototyping tool to support game writers in the domain of point-and-click adventure games.

Some learning environments, such as Enact [63] and deLearyous [105], offer game play tailored to a specific psychological model. Enact [63] is based on five styles of handling interpersonal conflict proposed by Rahim [90]. A scenario author for such a learning environment uses the psychological model for the scenario validation. A scenario in these environments is difficult to adapt to a new psychological model or protocol as opposed to the Communicate learning environment which provides an authoring tool with expressive constructs for easily creating scenarios.

Commercial tools such as for example Chatmapper[5] and Articy[6] offer a visual authoring environment for creating a dialogue. Twine[7] is an open source authoring tool often used for narrative or dialogue writing. Chatmapper uses Lua scripts[8] to set or modify parameter values in a dialogue scenario. Scripting in Lua requires programming skills. Articy is a visual environment for the creation and organization of game content offering virtual characters, locations and a game development environment. Modelling game mechanics, e.g. set or modify parameter(s), requires programming effort. Twine is predominantly aimed at developing a branching narrative and offers scripted programming support, but lacks extensive features to support game mechanics [26]. In the Communicate authoring tool an author has a visual interface to assign conditional logic, set or modify parameter values without the need for programming proficiency. Chatmapper, Articy and Twine offer some form of checking a dialogue. For example, a graph can be checked for completeness and the absence of dead links. This is similar to the Communicate authoring tool, in which a scenario author can check whether all nodes are reachable, nodes do not connect back to a parent, and if every path has an endpoint.

Yessad et al. [113] present an approach to validate game play in an interactive serious game. The approach consists of three steps: specifying a scenario using a generic pattern, model the pattern using a coloured Petri Net, and model checking the coloured Petri Net. Yessad et al. apply this approach to aid the development of a scenario in the design stage of a serious game.

Some game authoring environments generate a ‘correct by design’ dialogue for a game. Samyn et al. [96] develop a story engine that uses linear scenario templates, metadata,

[5] www.chatmapper.com

[6] www.nevigo.com

[7] twinery.org

[8] www.lua.org

and current game state to generate a scenario. Lessaerd et al. [60] present a tool based on context free grammars (CFG) combined with markup to design a virtual character (VC) dialogue. Lessaerd et al. argue that this tool enables a dialogue author to create new content with predictable runtime VC behaviour, and to identify errors during dialogue creation. However, Lessaerd et al. state that the current version of this tool is not very author friendly, and the proposed benefits have not been tested in practice.

3.3 Research context

In this section, we explain some of the required background knowledge about, and important concepts used in, the description of a scenario in Communicate, since we investigate scenario smells in the context of this learning environment. Almost all of this work has appeared in earlier publications about Communicate.

3.3.1 Authoring a scenario

Usability of an authoring tool often comes at the expense of expressiveness [73, 23]. The open source authoring tool used in Communicate provides expressive constructs for variability in a scenario [55]. A communication skills expert (called scenario author for the rest of the paper), usually a non-programmer, authors a scenario dialogue in the authoring tool. A scenario developed in the authoring tool is validated against an XML schema[9]. A scenario is parsed and produces a scenario 'reasoner' that interacts with the game simulation at run-time to provide information about the possibilities at each step of a scenario. This approach allows a scenario author to develop various scenarios without knowledge of the implementation of the game simulation. The Communicate serious game learning environment already addresses some limitations mentioned by Engström et al. [23], namely offering a graphical interface for authoring a scenario, expressive constructs, a format to validate scenarios and separation of content creation from game simulation.

In this paper, we explore how we can further support a scenario author in developing a scenario, by giving feedback on potential problematic aspects around scoring a scenario. To use an analogy from the Computer Science discipline: we can view the scenario authoring tool as a compiler and the smells tool developed in this paper as a Lint tool.

3.3.2 The structure of a scenario

A scenario is a sequence of interleaved subjects (see Fig. 3.2), where each subject is a directed acyclic graph (see Fig. 3.3) consisting of a sequence of statements alternating between a virtual character statement and multiple player statement choices.

A scenario author models a communication protocol from top to bottom. A subject usually is a part of a dialogue about a particular subject between a player and a virtual

[9] `https://uudsl.github.io/scenario/`

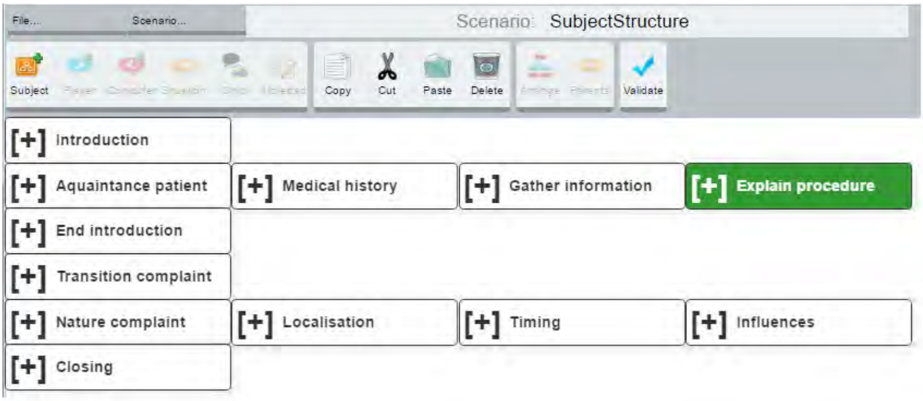


Figure 3.2: Example interleaving subjects in a Communicate scenario

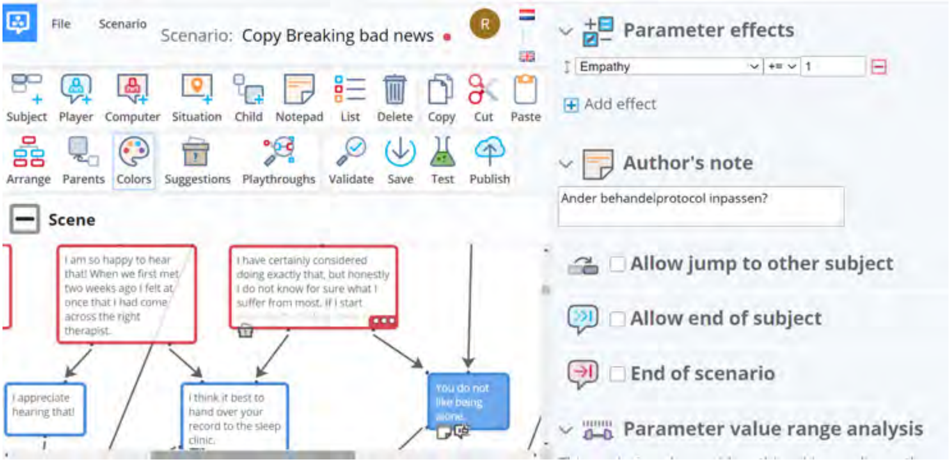
character. If the order in which a player navigates statements across two subjects (or more) is not important, then these subjects may be interleaved, which means that a player can choose between responses that come from both subjects. In the Communicate authoring tool, subjects on the same horizontal level are interleaved. A player gets statement choices from interleaved subjects with no predetermined order in a simulation.

A scenario author defines a learning goal as a parameter in a scenario. An example of such a parameter is 'Empathy'. A parameter is usually encoded as an integer, see Fig. 3.3.

A player statement (in blue) and a virtual character statement (in red) are nodes and the flow of conversation is an edge. A player statement usually has an incremental score (e.g. Empathy +1), emotional effect on a virtual character (e.g. 'Angry'), and feedback text for a player. Multiple player nodes per computer node indicate multiple statement choices for a player. A scenario author assigns a value to a parameter (or multiple parameters) on a player statement choice node.

A scenario author can further increase variability in a dialogue by marking a node with the following:

- Conditional: show only if a particular condition(s) is met.
- Jump: allow to jump from this node to another node in a subject at the same horizontal interleave level.
- Early end of subject: allow to jump from this node to another node in one of the subjects at the same horizontal interleave level. If there are no other subjects at the same level, allow to jump to a node in a subject in the next vertical interleave level.
- End of scenario: terminate a scenario after this statement.



Scenario: Copy Breaking bad news

I am so happy to hear that! When we first met two weeks ago I felt at once that I had come across the right therapist.

I have certainly considered doing exactly that, but honestly I do not know for sure what I suffer from most. If I start

I appreciate hearing that!

I think it best to hand over your record to the sleep clinic.

You do not like being alone.

Empathy += 1

Ander behandelprotocol inpassen?

□ Allow jump to other subject

□ Allow end of subject

□ End of scenario

Figure 3.3: Scenario authoring tool

3.3.3 Scoring in a scenario

Assigning a parameter value across nodes in a graph is challenging and can possibly lead to undesired effects. To illustrate, Fig. 3.4 shows part of a dialogue with a single parameter in the left graph, while the right graph shows the same dialogue with two parameters. r_1 to r_6 depict the player statement choice nodes. Note that the statement texts in these nodes is not important.

When playing the left dialogue, a player can choose between r_1 and r_2 in the first step of a simulation, where r_1 gives a better score. If a player chooses r_1, she gets the choices r_3 and r_4 in the following step of the simulation. It might not be unreasonable to assume that choosing a statement with the highest score at each step results in a total maximum score for a scenario. In this example, the sequence of choices r_1r_3 yields a total parameter score of +2. However, the sequence r_2r_5 gives a total score of +3, the best score for this parameter. It is possible that the scenario author made an explicit decision for this choice, or might have made an error, but the situation 'smells'.

The use of more parameters increases scoring complexity, as depicted in the right graph in Fig. 3.4. The best parameter scores are the sequences r_1r_3 (assertive: -1, helpful: +2), and r_2r_5 (assertive: +4, helpful: +1). Are these parameters (negatively) correlated in this scenario and is that the intention of a scenario author? Scoring complexity further increases with the use of interleaving and other features.

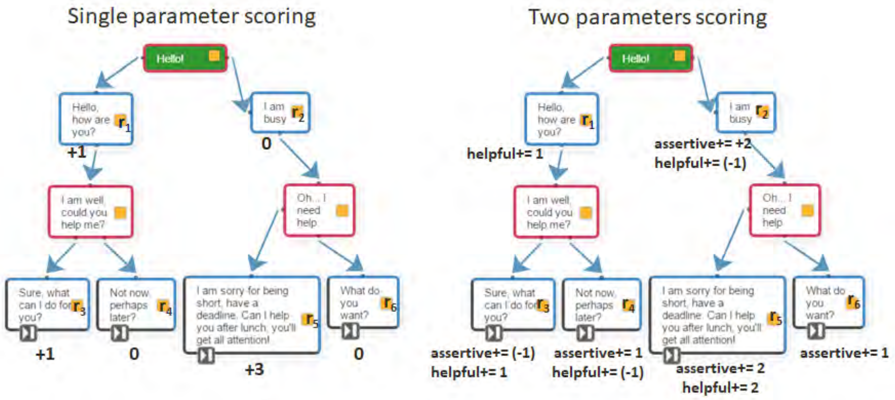

Single parameter scoring
Hello!
Hello, how are you? r1
+1
I am busy r2
0
I am well, could you help me?
Oh... I need help
Sure, what can I do for you? r3
+1
Not now, perhaps later? r4
0
I am sorry for being short, have a deadline. Can I help you after lunch, you'll get all attention! r5
+3
What do you want? r6
0
Two parameters scoring
Hello!
Hello, how are you? r1
helpful+= 1
I am busy r2
assertive+= +2
helpful+= (-1)
I am well, could you help me?
Oh... I need help
Sure, what can I do for you? r3
assertive+= (-1)
helpful+= 1
Not now, perhaps later? r4
assertive+= 1
helpful+= (-1)
I am sorry for being short, have a deadline. Can I help you after lunch, you'll get all attention! r5
assertive+= 2
helpful+= 2
What do you want? r6
assertive+= 1

Figure 3.4: Scoring challenges example

3.3.4 The quality of a scenario

A scenario can be viewed as a multiple-choice test, where the questions posed to the player depend on the answers to the previous questions. When viewed as an assessment, a scenario should be valid and reliable. Besides validity and reliability, there are various other quality aspects of scenarios we can study.

Kasperink [48] applies an argument based approach [46] to validate Communicate and a scenario in Communicate. She validates the ‘breaking bad news’ scenario for Psychology students, where a student gives bad news to a client. She concludes that the argument based approach is suitable for validating a serious game scenario.

Pel et al. [83] co-create a scenario for shared decision making with co-creation sessions with older adults. Pel et al. evaluate these sessions and improve the scenario by further explaining the goal and the context to positively support the usability and play experience for older people.

Gupta [33] uses psychometric techniques to determine the quality of assessment in a scenario. She analyses the dialogue data from multiple scenarios and determines item difficulty, item discrimination and Cronbach’s alpha.

Lala et al. [52] collects student feedback after conducting a communication skills teaching session using Communicate. Lala et al. apply the action research method [62] over three semesters to iteratively improve the quality of a scenario and a communication skills teaching session based on the feedback from students.

In the following section, we introduce scenario smells as another method that can be used by a scenario author to analyse aspects of the quality of a scenario. Scenario smells can be used during scenario development, and do not require player data.

3.4 RQ1: Scenario author challenges and related smells

In this section, we discuss the interviews about challenges in scenario development with six scenario authors, and define various scenario smells based on the results from these interviews.

3.4.1 Challenges for scenario authors in developing scenarios

We interviewed six scenario authors to find out the challenges they face in developing a scenario in Communicate. Our goal was to find information that a scenario author finds important for improving the quality of a scenario. We structured an interview in parts, the entire questionnaire is included in Appendix 3.A. First, we collected information about a scenario author and how she uses Communicate. Second, we posed questions about how scenario authors develop scenarios. Third, we asked questions related to scenario smells. Fourth, we asked a scenario author what s(he) considers an optimal path in a scenario. Fifth, we proposed enhancements in the authoring tool to aid a scenario author, and asked for open comments. We describe the results of these interviews in the rest of this subsection.

Of the six scenario authors, three were communication skills lecturers at Utrecht University from the faculties of medicine, pharmacy and veterinary science; one was an honours student who recently started using Communicate and two were employees of the startup company DialogueTrainer, which specialises in creating scenarios for training clients in communication skills. The expertise in using the Communicate authoring tool varied considerably, ranging from an author who used only one subject and no conditionality in nodes to an author who used multiple interleaved subjects with entire dialogue paths dependent on a condition.

Most scenario authors created a dialogue first and then assigned parameter and emotional values to a statement. Some scenario authors also created an 'optimal' dialogue path first, and thereafter added 'erroneous' paths. The challenges in creating a scenario varied with the experience of the user, for example when asked about the usage of 'view parents' and 'view parameters' functions, multiple scenario authors indicated that they were not aware of these functions. Scenario authors who did use these functions, used them frequently. Testing a dialogue also varied, some scenario authors played a scenario themselves numerous times, while others let teaching assistants play test a dialogue.

We prepared closed statements to help define scenario smells and asked a scenario author to score the statement on a scale of 1 to 5, where 1 meant complete disagreement with the statement, and 5 meant complete agreement with the statement. We asked scenario authors to rate fifteen statements. Of these fifteen statements, all interviewed scenario authors rated the following with a score of 4 or 5:

- If a player chooses the best option (the option with the highest score increase) at every node, she should have the highest possible score at the end of the scenario.

- It should be possible to get the maximal total score.

- For every parameter there should be a path that generates the maximal score.

- Every node should be reachable.

Scenario authors rated the following statements with a score of 1 or 2 (meaning that they consistently disagree):

- The longest path should yield the highest score.

- The longest path should yield the lowest score.

- Parameters are always positive.

The scenario authors neither agreed nor disagreed with or did not consistently rate the other eight statements (score between 2 and 4):

- The sequence in which subjects have been navigated should not matter for the end score.

- Two parameters should have no correlation.

- It should be possible to get the minimal score.

- It should be possible to get the minimal score on every parameter, but not necessarily the minimal end score.

- The player should sometimes sacrifice a scoring opportunity to receive a better opportunity later.

- A jump point from a subject is not a good idea.

- All parameters should use the same scale (e.g. 1 to 10, 0 to 100).

- It should be possible to score above 50% on every parameter.

We asked the scenario authors seven closed questions on what they considered to be an optimal path in a scenario. One of our goals was to determine if it is possible to unambiguously define the concept of an optimal path in a scenario. The definition most scenario authors prefer is: ‘The optimal path is the path with the highest weighted average of all parameters.’ We refer to the optimal path as the **best score** for the rest of the paper. The **best score** is not a smell as such, but useful for scenario development.

We proposed some potential enhancements to the Communicate authoring environment, and asked for feedback. Three possible enhancements addressed displaying scores in a scenario graph in the Communicate scenario editor. Scenario authors had two primary concerns about these statements: would the proposed enhancements lead to information overload, and how useful is displaying a score in a scenario graph in actual practice? Three possible enhancements related to showing the optimal path while editing. Scenario authors were positive about highlighting an optimal path, even if limited to within a subject. We also asked about the trade-off between calculation time and accuracy. All authors indicated accuracy as more important than calculation time. Three possible enhancements related to marking a particular node as (un)desirable and getting a warning by (not) being on an optimal path; scenario authors were not enthusiastic about these. Finally, two enhancements related to viewing statistics for a scenario. All scenario authors indicated that the maximum score was important; while some authors were interested in the minimum score and the correlation between parameters.

3.4.2 Selected scenario smells

Based on the interviews, we selected the following scenario smells. We give the name of a smell in brackets; we will use these names in the rest of the paper:

- The minimum (***minimum***) or maximum (***maximum***) score of a parameter is not attainable.
- Always choosing the best option does not result in the maximum score (***always best option***).
- A longest path yields the maximum score (***longest***).
- Two parameters correlate (***parameter correlation***).
- A scenario is not subject monotone (***subject monotonicity***).

Scenario authors assume that a minimum and maximum score of a parameter is attainable while playing a scenario. For example, a scenario author could specify in a scenario definition that a parameter is an integer value between 0 and 10. A player might assume that a score of 6 represents a sufficient result; 8 a good result and 10 a perfect result. If it is impossible to achieve a score higher than an 8 through any path, a ‘perfect’ performance (of 8) corresponds to a score of 80%. If the minimum and maximum score of a parameter is not attainable in a scenario, it constitutes a smell.

Scenario authors indicate that if the longest path yields the maximum score, it is a case of bad scenario design. The reasoning is that a player taking a longer path is perhaps inefficient in a dialogue. If the longest path yields a maximum score, it constitutes a smell.

If two parameters have a positive correlation, perhaps a scenario author should combine these parameters, since having the two different parameters is apparently superfluous. In some cases parameters for conflicting goals, for example as in the right graph of Fig. 3.4, should have a negative correlation. The sign (positive/negative) and degree of correlation for parameters in a scenario should match the expectations of a scenario author.

On a subject level with interleaving, Communicate allows a player to traverse interleaved subjects in any order. Scenario authors indicate that modifying a scenario can present challenges, especially with interleaved subjects. A dialogue path (good or bad) is difficult to remember. We define a scenario where subjects are independent of each other as a subject monotone scenario. If a scenario is not subject monotone, it represents a smell and we report this to the scenario author.

In addition to the scenario smells indicated by the scenario authors, we develop a ***shortest*** smell. Our assumption is that the shortest path should not yield a maximum score. Most dialogues often have obstacles, such as a bad statement choice that ends a subject or a scenario prematurely. Such a node is often in the shortest path of a scenario. It follows that a shortest path should not result in a maximum score. We define this smell as ***shortest***: A shortest path yields the maximum score for a parameter.

3.5 RQ2: A tool for reporting scenario smells

This section describes the rationale of the implementation of our software tool to report scenario smells. We build a tool that takes as input a scenario based on the Communicate scenario language definition[10]. Our tool generates a file that reports the **best score** and scenario smells for the given scenario.

A parameter value in Communicate may decrease in a child node, e.g. for a player statement choice that is a 'bad' response. We implement an all paths traversal algorithm, where we compare all paths to determine an optimal path for the **best score**. We use a depth first tree traversal algorithm to find the paths with a maximal score, the paths with a minimal score, the shortest paths, and the longest paths. Further details of the implementation can be found in T.Overbeeks's Master thesis [78].

We use the Pearson correlation coefficient to calculate correlation between parameters. We calculate the possible end values of each parameter and then apply the Pearson correlation to all pairs of parameters.

Subject monotonicity is based on the mathematical concept of monotonic functions. A monotonically increasing function f is a function for which for all x and y, if $x \geq y$, then $f(x) \geq f(y)$. In the case of scoring scenarios, the input consists of the net parameter changes within a subject, and the output is the set of parameter scores at the end of a scenario. A scoring is monotonically increasing if an increase in the net parameter change in a subject leads to an end score set of values equal to or greater than before. A decrease in the net change in one subject should lead to an end value equal to or lower than before for a monotonic decreasing scoring. A scenario that satisfies these conditions is subject monotone. In a subject monotone scenario we consider the subjects to be independent. For instance, if we increase one of the parameter changes in a subject, the end value of that parameter in the scenario will also increase or stay the same.

The output of our tool consists of plain text files. A main file contains an overview of the presence of the different scenario smells.

For example, a portion of the main file looks like:

```
BEST SCORE: 7
FINAL PARAMETER VALUES:
```

[10] https://uudsl.github.io/scenario/

```
ResultOriented: 8
GoalOriented: 6

Scenario Smells
The longest path yields the maximum score
Correlation between ResultOriented and GoalOriented:
-0.384302256778923

GoalOriented
Always choosing the best option does not result in the best score.
```

For detailed information, a scenario author can inspect a text file for each parameter and one for the total score. These files contain path information of the scenario smells for a scenario. Path information consists of a list of the statement texts of the nodes in a path.

3.6 RQ3: Evaluating the scenario smells tool

In this section, we evaluate the scenario smells tool with scenario authors using scenarios used in practice.

We developed the tool in collaboration with a scenario author from Utrecht University, who tested development versions of the software on a scenario he used in blended teaching sessions. This scenario author also used the software tool while making changes to the scenario. We also tested with test scenarios of varying complexity.

The final software was evaluated with four other scenario authors, two from Utrecht University and two from the DialogueTrainer company. One of the DialogueTrainer scenario authors was a new employee and not part of the initial six interviewed scenario authors. The other three scenario authors were part of the initial interviewees. Prior to an interview, a scenario author provided one or two scenarios: an 'average' and a complex scenario. The evaluation interview questionnaire is given in appendix 3.B. We summarise the answers in the following subsection.

3.6.1 Evaluation interviews with four scenario authors

In the initial interviews (Section 3.4.1), scenario authors indicated the optimal path (**best score**) to be the most important piece of information. In the evaluation round of interviews, scenario authors indicate that the **best score** as a function they will definitely use in practice. In addition to the **best score**, our tool presents the scenario smells listed in Section 3.4.2. In the evaluation interviews, the scenario authors rate each smell on a scale of 1 to 5, where 1 denotes that they are unlikely to use a smell, and 5 that they will definitely use a smell in practice to improve a scenario. In addition, the scenario authors also provide a rationale for (not) using a smell to improve a scenario.

The top rated smells (score of 4 and above) include: The ***minimum*** and ***maximum*** score of all parameters is not attainable; always choosing the best option (***always best***

option) does not result in the maximum score; and a ***longest*** path yields the maximum score. Scenario authors reiterate that a player should experience a scenario as fair and an important aspect is that a highest score is attainable, as also mentioned during the initial interviews.

The smell least likely to be used is if the ***shortest*** path yields the maximum score. The scenario authors indicate a common practice that they include a short erroneous path that often leads to a premature end of a scenario simulation. Since the scenario authors are aware of this erroneous path, they consider this smell unnecessary.

Other lower ranked smells (3.5 and lower) include ***subject monotonicity*** and ***parameter correlation***. Some scenario authors find ***subject monotonicity*** somewhat difficult to understand.

The scenario authors further differentiate between using a scenario smell while developing a scenario and after finishing a scenario. During development, they are more likely to use one or two functions, e.g. ***maximum***, and after finishing a scenario, they might use an entire scenario smells report. While developing a scenario, authors wish to iteratively make changes to a scenario, and run the tool. Thus the tool needs to report scenario smells within a few minutes, or sometimes at most an hour. For correctness after finishing a scenario, scenario authors are willing to wait overnight for a full report.

3.6.2 Tool performance on actual scenarios

We evaluate our tool performance on a computer with an Intel Core i7 7700 (Quad core, 3.6 GHz). Scenario authors indicate that they use their computer while the tool is running and nowadays most computers have a working memory of 8 to 16 GB. We limit memory usage for the tool to 4 GB for evaluation and the maximum time to execute to a day (24 hours).

Our tool computes a complete scenario smell report for all ‘normal’ (including test) scenarios well under a minute (max. time 0.5758 mins using 2.2 GB memory). However, four complex scenarios expose the performance limits of the tool. We examine the **best score** and the complete smell report for these four complex scenarios. The scenario characteristics and tool limitations related to these four scenarios are:

- The first complex scenario has four sequential subjects and a highest node count of 72 within a subject. In this scenario, multiple player statement nodes have often no (delta)scores. In our tool, an ***always best option*** tree records all the child nodes with an optimal parameter increase. If no child node increases a parameter value, the search algorithm treats all child nodes equal. A complete report run leads to out of memory within the hardware constraints, the likely cause is the ***always best option*** tree, which is relatively memory intensive. The **best score** calculation runs in 7.74 minutes.

- The second complex scenario has six sequential subjects, node count within a subject ranges from 16 to 58. This scenario is structured so that if a player makes a major mistake during a dialogue, she receives feedback and goes to another extra part of the dialogue. After the initial run fails, we dissect this scenario to test the limits of our tool. We start with only the top level subject, execute our tool and add the next subject and run the tool iteratively. The **best score** run works within our predetermined limits until the 3rd subject (total nodes 154). Adding the fourth subject (total nodes 170) exceeds the 24 hours limit. The complete smell report works only until two subjects (total node count 96) and goes out of memory with the 3rd subject (total nodes 154). The likely cause is the ***always best option*** tree which is memory intensive.

- The third complex scenario has 4 interleaved subjects with six jump points. We analysed that interleaving should have an effect of increasing calculation time by a factorial. We notice an increase in calculation time when adding an interleaved subject in our test scenarios, however this scenario executes within our predetermined limits. The third complex scenario goes out of memory rather than out of calculation time. Analysis of this scenario reveals that numerous child nodes do not have a parameter score and the problem is similar to complex scenario 1. The likely cause is the ***always best option*** tree which treats all child nodes equal in the absence of a 'best option' and is memory intensive.
- Finally, the fourth complex scenario has numerous player statement nodes as a response to a virtual character node. This scenario has four sequential subjects and the node count of the subjects is 92, 65, 78 and 31 respectively. In this scenario, quite a few virtual character nodes have up to eight player choice nodes. In comparison, in most scenarios virtual character nodes have at most three to four subsequent player statement nodes. We predicted that a high path count severely impacts calculation time, but not necessarily memory usage. After the initial run fails, we also dissect this scenario to test the limits. Even with one subject, the calculation time for **best score** is high (759.57 minutes) while the memory usage is relatively low (26 MB). Adding another subject results in an out of calculation time for our tool report.

Summarising, the smells tool processes most scenarios within user-acceptable processing time. These scenarios contain two to eight parameters, one to seven subjects, one to four interleaved (parallel) subjects, and subjects contain 20 to 132 nodes. For a limited number of exceptional scenarios, a combination of these scenario characteristics, e.g. a scenario with more than three interleaved subjects each with more than 60 nodes, leads to a longer than user-acceptable processing time. The ***always best option*** tree is the likely cause in this case. The scenario authors indicated that most scenarios have fewer than three interleaved subjects at any level and that the trend is towards more compact dialogue graphs within a subject. To check these statements, we examined the last ten published scenarios in Communicate from both Utrecht University and the DialogueTrainer company respectively. These twenty scenarios had at most two interleaved subjects at any level and only two scenarios had one subject with more than 60 nodes. We conclude that the smells tool therefore works within user-acceptable processing time for almost all scenarios.

3.7 Discussion

We hypothesized that offering a tool to signal potential scoring issues can support a scenario author. To answer RQ1 (Section 3.4), we interviewed scenario authors and found that scoring is indeed a difficult aspect of scenario development. Scenario authors consider **best score** as one of the most important aspects of a scenario in Communicate. In addition, we introduced the concept of scenario smells. Signalling scenario smells may support a scenario author by pointing to potentially problematic parts of a scenario.

To answer RQ2 (Section 3.5), we used tree traversal algorithms in developing a tool to report scenario smells and tested the tool on a number of test scenarios. The smells tool can processes most scenarios in less than a minute.

To answer RQ3 (Section 3.6), we found that scenario authors were likely to use our tool, using mostly the **best score** during scenario development, and the complete smell report after scenario completion. The smells tool processed most scenarios within user-acceptable processing time. Scenario authors find that a player should experience a scenario as fair and are likely to use smells that help signalling that scoring might be perceived as unfair. Some smells, in particular ***subject monotonicity*** and ***parameter correlation*** were ranked lower than we expected. The interviewees found ***subject monotonicity*** initially difficult to understand, however on explanation were slightly more favourable. ***Parameter correlation*** ratings had a high variance: two authors rated it high, two low. From the initial interviews, we observed that scenario authors had a different approach to scenario development. Different approaches could relate to the likelihood of using (or not using) ***parameter correlation***.

3.8 Conclusion and future work

Communication skills are essential for the majority of professions and are best learned through practice. These skills can be developed in virtual learning environments that simulate real world communication scenarios, for instance a bad news conversation. In Communicate, a scenario author develops a scenario in an authoring tool. A scenario author wants to develop a scenario of good quality, in which the scoring reflects the intentions of the scenario author, and is perceived as fair.

This paper introduced the concept of a scenario smell as a symptom of a potential scoring problem in a scenario. To find out which scenario smells often occur, we determined the scoring challenges that scenario authors face by conducting interviews with scenario authors who use the Communicate authoring tool. To automatically detect the identified scenario smells in a Communicate scenario, we developed a tool that takes a scenario as input and produces a report on the detected smells. We conducted evaluation interviews with scenario authors and evaluated the tool on actual scenarios in Communicate. Scenario authors generally appreciated the feedback they got from the smells tool, and were likely to use it during scenario development. Although we developed and evaluated our work in the context of Communicate, we hypothesize that our results can be applied in any environment that uses scripted dialogues with scoring at each choice a user makes.

In the future we would like to investigate if we can improve the search algorithm, for example by pruning paths with the same parameter scores and investigating the effect on calculation time versus accuracy for a scenario. We could also add a preprocessing step in our scenario smells tool. If an input scenario contains more than three interleaved subjects and/or subject(s) with more than 60 nodes, we can provide a warning message that the processing of this scenario could likely be long. Furthermore, we would like to visualise some scenario smells in the Communicate authoring tool. For example, we could highlight the path to obtain the **best score** in a scenario.

3.9 Acknowledgements

We thank the scenario authors, and J. van Dortmont and M. van Geest from the development team of DialogueTrainer.

This work has been partially funded by the EU H2020 project RAGE (Realising an Applied Gaming Eco-system)[11], Grant agreement no 644187. This paper reflects only the author's view, the European Commission is not responsible for any use that may be made of the information it contains.

[11] http://www.rageproject.eu/

3.A Initial interview questionnaire

General Information

1. What is your profession and field of expertise?

2. How many scenarios have you created with Communicate?

3. Have you used Communicate in a class room situation?

4. Do you use the validate option?

5. Do you use the view parents option?

6. Do you use the parameter value range analysis?

7. Do you usually work node by node (including an utterance, preconditions, emotions, parameters) or do you first write an entire dialogue and thereafter add parameters and preconditions?

Their Ideas

8. What kind of problems do you experience in creating Communicate scenario's?

9. Are there problems that are specific to scoring?

10. Do you have any ideas about how we could solve those problems?

11. What kind of steps do you take to avoid problems in scoring?

Validating statements related to possible smells

I now would like to ask some questions about scenarios. The Communicate authoring tool gives a lot of freedom to develop scenarios. This can be a problem for creating validation tools. We try to make some assumptions about what constitutes a 'good' scenario. Could you please describe how true you think the following assumptions are? You have the following options: always true, mostly true, cannot say, mostly false, always false.

12. The longest path yields the maximum score.

13. If a player chooses the best option (the option with the highest score increase) at every node, she should have the highest possible score at the end of the scenario.

14. The sequence in which subjects have been navigated should not matter for the end score.

15. It should be possible to get the maximum total score.

16. It should be possible to get the maximal score on every parameter, but not necessarily the maximum total score.
17. Every node (even those with prerequisites) should be reachable in some way.
18. Two parameters should have no correlation.
19. The longest path should yield the lowest score.
20. It should be possible to get the minimal score.
21. It should be possible to get the minimal score on every parameter, but not necessarily the minimal end score.
22. The player should sometimes sacrifice a scoring opportunity to receive a better opportunity later.
23. A jump point from a subject is not a good idea.
24. All parameters should use the same scale (e.g. 1 to 10, 0 to 100).
25. It should be possible to score above 50% on every parameter.
26. Parameters are always positive.

Could you please select three assumptions that you currently find the hardest to validate in your own scenarios? Are there any assumptions we have missed?

Optimal Path

An optimal path is a term that usually describes a sequence of nodes that rewards a player with the highest score or the most wins. We could conclude that the path with the highest overall score is the optimal path in the case of Communicate, but this is not necessarily satisfactory. I state some ways to determine the optimal path. Could you please tell me for each definition if you agree or disagree that this would be a workable definition?

27. The path with the highest unweighted average of all the parameters.
28. The path that includes the highest score for an individual parameter.
29. The path with the highest weighted average of scoring parameters. (Note to readers: a scenario author often indicates an total score as a weighted score of parameters)
30. The shortest path.
31. The path with the highest score for a parameter.
32. The longest path.

33. Same as one of the above, but with the extra requirement that all the scores need to be above a certain threshold.

Which of these definitions do you thinks works best? Are there any other possible definitions that we did not think of?

Solutions / UI

I know discuss possible improvements to Communicate. These are hypothetical improvements, so we do not consider feasibility. For each of these improvements, can you tell me what you think about the following aspects:

- Does using it improves your scenarios?
- Do you think you would make use of this improvement?
- How much margin for errors is there? Should a tool give an exact answer or is it sufficient to give a reasonable answer?
- Would it be okay if we ignore some parts of a scenario in the tool?

34. Visually displaying all the parameter changes for every node.
35. Color coding all the nodes based on the parameter changes. Green for a positive effect on the overall score, orange for no effect and red for a negative effect.
36. Visually displaying the change in the overall score for each node.
37. A button to show the optimal path through a subject.
38. A button to show the optimal path through the entire tree.
39. A button to show the optimal path from a specified node.
40. An option to repeatedly show optimal paths from different start nodes, whereby the start points are randomly chosen by the computer.
41. An option to mark nodes as desirable and to get a warning if a desirable node is not included in the optimal path.
42. An option to mark nodes as undesirable and to get a warning if an undesirable node is included in the optimal path.
43. Statistics about the scores like correlation, minimum value, maximum value etc.
44. A profile of every subject that shows how often a parameter is used in that subject, and what the maximum changes (both positive and negative) are.

Finishing Thoughts

That was my last question. Now that we have finished the entire interview, do you have any last thoughts. Did you think of any new possible improvements that could be made to Communicate with respect to scoring? Did I miss something important?

3.B Evaluation interview questionnaire

General Information

1. How many scenarios have you created in Communicate?

2. Are you currently using Communicate in a classroom situation?

3. What is the context of your example scenario?

Demonstration

We will now demonstrate our tool. Afterwards, we want to discuss the different scenario smells. We want to focus on the following aspects:

- Was the result similar to what you expected?

- If not, what was different?

- Would you make changes to this scenario after seeing the results?

We will discuss the scenario smells in this order:

4. Always choosing the best option does not result in the maximum score.

5. A longest path yields the maximum score.

6. A shortest path yields the maximum score.

7. Is it possible to obtain a parameter value lower than the specified minimum value?

8. Is it possible to obtain a parameter value higher than the specified maximum value?

9. Is there a correlation between parameters?

10. Is the scenario not subject monotone?

Our tool

The tool we are about to demonstrate has multiple features. We want to know how useful you think these features are. Can you describe how likely it is you would use these functions by giving a number between 1 and 5, where a 1 means “would not use at all”, and a 5 means “would definitely use”?

11. Calculating the best path (overall score).

12. Calculating the worst path (overall score).

13. Calculating the longest path.

14. Calculating the shortest path.

15. Checking if always choosing the best option will lead to the maximum score.

16. Checking if the longest path will also be the best path.

17. Checking if the shortest path will also be the best path.

18. Checking if the minimal and maximum value of a parameter is attainable.

19. Calculating the correlation between different parameters.

20. Checking if a scenario is subject monotone.

Conclusion

21. Do you think that the demonstrated program might help you improve your scenarios?

22. Are you likely to use the program if it remains separate from the Communicate authoring tool?

23. Are you more likely to use the program if it is integrated in the Communicate authoring tool?

24. Were the calculation times a problem for you?

25. For your personal use, what would be the most useful feature of the current program?

Chapter 4

Introducing a scenario specific corpus method to match open text input to scripted statements in a communication skills serious game

In our serious game Communicate, a student plays a scenario with a virtual character to practice a communication skill(s) such as giving feedback to a team mate. In the original version of Communicate, a student 'converses' with a virtual character by choosing a statement option from one of the scripted statements, similar to selecting from a multiple choice test item answers. We want to introduce open text input for a student, as research shows that open text requires more complex levels of thinking. A scenario in Communicate consists of statements following a specific protocol such as giving bad news in a specific situation. A statement has additional information like a score and virtual character emotion. In this paper, we introduce a scenario specific corpus method (SSCM) that uses scenario information to create a scenario specific corpus, which is used in matching an open text against scripted statement options. We gather open text utterances from students in communication workshops using Communicate and create a golden dataset. We evaluate SSCM by comparing it against generic natural language processing (NLP) methods. We find that SSCM slightly outperforms generic NLP methods for matched text but underperforms for unmatched text.

This chapter is based on published conference extended abstract/posters in the proceeding of:

- Semantics and pragmatics of dialogue (SemDial) conference [1]
- Artificial Intelligence in Education (AIED) conference [2]
- Computer supported collaborative learning (CSCL) conference [3]

[1] https://f.hypotheses.org/wp-content/blogs.dir/4280/files/2018/11/paper_1.pdf

[2] https://link.springer.com/chapter/10.1007/978-3-030-23207-8_45

[3] https://ris.utwente.nl/ws/portalfiles/portal/129957345/CSCL_2019_Volume_2.pdf#page=363

4.1 Introduction

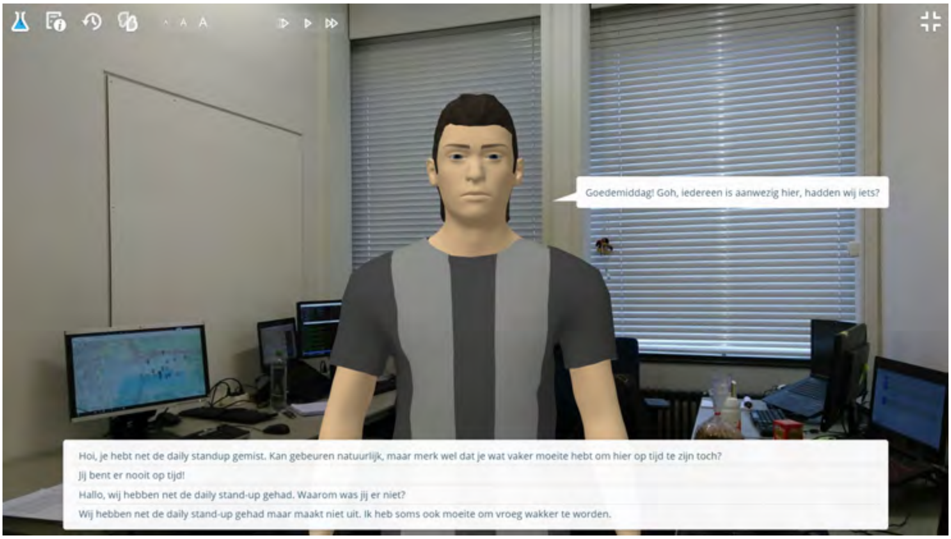
Serious games offer the potential for learning and teaching a skill effectively. In a systematic literature review, Connely et al. [16] find evidence of learning, skill enhancement and engagement in a player of a (serious) game. A well-designed serious game can enhance a player's motivation through engagement and immersion [106].

In serious games for communication skills such as Aggression de-escalation [11], Communicate [43], Glengarry Glen Ross for Sales Game Dialogues [72], SimProphet [59], a Tough Sell [59], and Visual SceneMaker [28], a student converses with a virtual character to practise a communication skill. The mechanics in these serious games varies: Aggression de-escalation, Communicate and Visual SceneMaker use a scripted dialogue; SimProphet and a Tough Sell use a chatbot engine; and Glengarry Glen Ross uses an agent architecture.

Arnab et al. [5] describe the challenge of balancing pedagogical aspects with game mechanics in a serious game. A chatbot [59] and agent approach [72] allow for non deterministic virtual character behaviour and lead to variability in the game experience for a player. However, these approaches are more difficult to develop, especially for a non programmer. A scripted approach has the advantage that it allows deterministic scoring, and full control over all utterances, an aspect communication skills teachers find important [54]. Scripted communication skills learning environments such as Communicate [43] and VisualSceneMaker [28] provide an authoring tool for communication skills teachers. In Communicate [43], a player chooses from multiple statements options at a step of a scenario to 'converse' with a virtual character, see Fig. 4.1.

Martinez [65] describes how a student's cognitive load varies in different test item formats. Multiple choice items often elicit low level cognitive processing, whereas constructed-response items more often require complex thinking. According to Krathwohls' cognitive process taxonomy [51], multiple choice questions test cognitive process dimension levels 1 to 5: remember, understand, apply, analyse and evaluate respectively. Cognitive process dimension level 6: create, is not tested. To test the cognitive process of creation, a student needs to construct text input herself.

Communicate is widely adopted: at our University alone, more than 20 teachers and teaching assistants use it in their teaching. The scenario database contains more than two thousand (variants of) scenarios. The ease of creating and modifying a scenario in our authoring tool helps in the pedagogical aspects of our serious game [83]. A communication skills teacher, hereafter referred to as a scenario author, scripts both the player and virtual character statements in a scenario, see Fig. 4.2. See [55] for more details about the structure of a scenario. Multiple player statements represent multiple choices for a player to choose from during gameplay. A scenario author typically encodes a learning goal for a scenario, such as assertiveness, as a parameter. A scenario author scripts a player statement and adds information to that statement, e.g. an (incremental) effect on a parameter (learning goal) and an emotional effect, such as 'Happy', for a virtual character, see right hand side of Fig. 4.2. During gameplay, if a player chooses this statement, the parameter (learning goal) is incremented with this value and the virtual



Goedemiddag! Goh, iedereen is aanwezig hier, hadden wij iets?

Hoi, je hebt net de daily standup gemist. Kan gebeuren natuurlijk, maar merk wel dat je wat vaker moeite hebt om hier op tijd te zijn toch?

Jij bent er nooit op tijd!

Hallo, wij hebben net de daily stand-up gehad. Waarom was jij er niet?

Wij hebben net de daily stand-up gehad maar maakt niet uit. Ik heb soms ook moeite om vroeg wakker te worden.

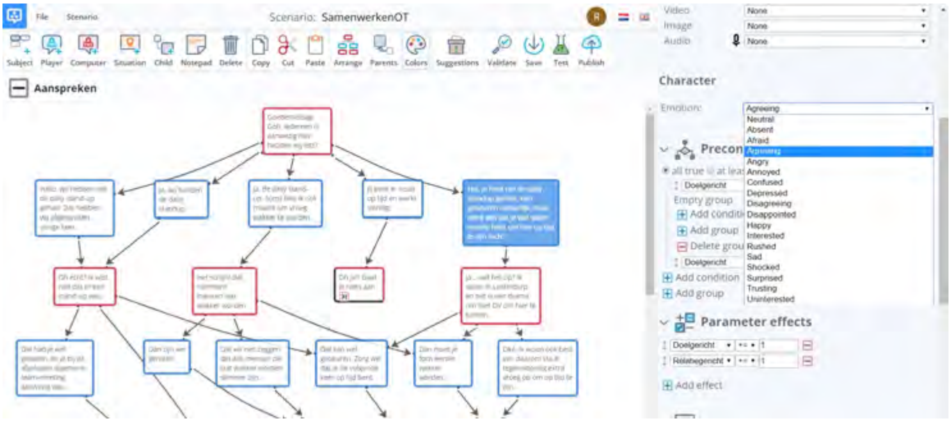
Figure 4.1: Communication skills scenario original multiple choice

character displays the given emotion.

To support a student's creation cognitive process, and positively effect immersion, we want to add the possibility to input open text to Communicate. Our approach to 'understand' open text input is by matching an open text input with one of the multiple scripted player statement options at a scenario step, an approach similar to the one used in Façade [66]. In Façade, an open text input is matched to a predefined set of discourse acts, for example criticism, praise etc. Based on a classification of the input text, the virtual character responds by selecting from a predefined set in Façade. In our approach, a virtual character responds with the scripted statement following the matched player statement.

To match an open text input, we use the concept of a corpus. Riloff et al. [94] present a corpus approach to build a semantic lexicon for a specific domain. According to Riloff et al., a semantic lexicon is useful in many contexts of natural language processing, for example for word sense disambiguation, selection, discourse processing etc. They present a system where an input consists of an initial set of words and a representative text for a specific category. The system outputs a ranked list of words associated with the category. More contemporary related work is given in Section 4.2.

A scenario represents the expert knowledge of a communication skills teacher for a particular protocol in a domain. Our hypothesis is that we can use information in a scenario to create a scenario specific corpus. Subsequently we use this corpus to match open text input to a scripted statement. We refer to our method as scenario specific



Scenario: SamenwerkenOT

Aanspreken

Video: None
Image: None
Audio: None

Character

Emotion: Agreeing

Parameter effects

Doelgericht += 1
Relatiegericht += 1

Figure 4.2: Authoring tool

corpus method (SSCM). Developing and applying a scenario (domain) specific corpus in the context of a serious game is a novel contribution.

In our approach, we match an open text input to one of multiple scripted statements. To evaluate SSCM, we compare with NLP methods that match using generic corpora. To compare, we need a dataset consisting of records of open text input matched to one of multiple scripted statements.

Our research questions are:

- RQ1: How can we use the information in a scenario to develop a scenario specific corpus?

- RQ2: How can we match an open text to scripted statements in SSCM?

- RQ3: How do we use Communicate to create a golden dataset to compare matching an open text to one of multiple scripted statements?

- RQ4: How does matching using SSCM compare to matching with an NLP method using generic corpora?

This paper builds upon an extended abstract describing the research problem in SemDial2019 [53], and a conference poster in AIED2019 [95] on using generic NLP methods to match an open text input to scripted statement options. This paper describes SSCM in detail and compares SSCM to generic NLP methods.

Section 4.2 describes related work, Sections 4.3, 4.4, 4.5, answer the research questions RQ1, RQ2, and RQ3, respectively. Section 4.6 presents the results (RQ4) and Section 4.7 presents the conclusions and future work.

4.2 Related Work

Hennigan [39] examines using natural language processing for player interaction with a virtual character in serious games to enhance player immersion. Player immersion in turn leads to positive learning outcomes. Hennigan distinguishes between natural language understanding (NLU) to understand player input and natural language generation (NLG) for virtual character response. Picca et al. [85] review NLP usage in selected serious games and propose an NLP framework for serious games. They find that NLP techniques in these games are somewhat old and primarily used for text classification or speech tasks in these games.

Ochs et al. [77] develop a virtual reality based simulation to train a doctor to break bad news. An embodied conversational agent (ECA) 'acts' as a virtual patient and follows a scenario. A human operator interprets the speech input from a player (playing a doctor) in real time, and selects a semantic match with one of the available 136 prototypical sentences. These prototypical sentences are based on a previously transcribed corpus of doctor interactions with trained actors playing a patient. The matched sentence is sent to a dialogue system that generates a verbal and non-verbal response of the ECA.

Van der Lubbe et al. [102] present a serious game for training verbal resilience to a doorstep scam. In the training, a player learns to deny an unusual request. As in Communicate, a player navigates a conversation with a virtual character in a scenario. An initial version of the game uses a multiple choice format, where a player can choose an option from four statements at each step of a scenario. In [102], van der Lubbe et al. introduce a speech analysis module. In this version of the game, a player, after choosing a response, needs to utter and record the chosen response. The game then classifies the player utterance as dominant or submissive. The virtual character response depends on the player choice and utterance classification. The speech module of the game is built on 681 sentences recorded by four actors and uses a support vector machine (SVM), which classifies a player utterance at a step of a scenario as dominant or submissive.

Façade [66] is an interactive narrative game where a player 'visits' a virtual married couple and becomes involved in a marital conflict of the couple. Façade uses a method to map a player open text to a discourse act and thereafter map a discourse act into a character response. Façade game writers wrote thousands of rules to determine a discourse act from an open text.

All these serious games or simulations require a considerable effort in developing a corpus for a scenario. We will develop an approach that requires neither additional effort to develop a scenario specific corpus, nor human operator intervention during a simulation.

4.3 Developing a scenario specific corpus

In this section, we present the rationale for our scenario-specific corpus method and its implementation. We first examine some domain specific corpus methods. These methods typically use a root or seed word set characterising a specific domain, and annotate words in a generic corpus or lexicon to create a domain specific corpus.

Cui et al. [18] present a method to create a domain specific corpus for ontology construction using Wikipedia pages. The method takes a root word and creates a directed graph using linked category information in a Wikipedia page. Cui et al. use the classification from the Library of American Congress as a reference and perform a manual check of a produced directed graph versus relevant Wikipedia articles. Results of a search using this method to find relevant pages vary considerably, depending on the scoring algorithm and choice of root word.

Hamilton et al. [34] and Hammer et al. [35] introduce their respective methods to develop a domain specific sentiment lexicon. Hammer et al. combines existing generic lexicons with a corpus of a domain. Hamilton et al. uses a set of domain specific seed words as input and labels words in a lexicon as positive, negative or neutral. For example, the word soft is labelled positive when describing a stuffed toy, and negative in sports. Hamilton et al. determine the sentiment of text from some sampled Reddit forums and find that using the domain specific sentiment lexicon outperforms using Wordnet.

Remus et al. [92] present a method to develop web corpora by using a domain specific sample text and focused web crawling. Remus et al. collect relevant web pages in a specific domain and language. Domain specific web crawling outperforms non focused web crawling in collecting relevant web pages.

Summarising, for a particular goal such as finding relevant pages, or determining the sentiment of a text, a method using a domain specific corpus tends to outperform a method using a generic corpus.

In Communicate, a scenario author scripts a scenario with communication skills in mind, for example assertiveness, empathy, etc. A scenario author develops a scenario as a directed acyclic graph of statements, specifies (delta) values for a parameter per statement and assigns an emotional effect, such as ‘Surprised’, for a virtual character, see Fig. 4.2. Garside et al. [27] define corpus annotation as the practice of adding interpretative, linguistic information to an electronic corpus of spoken and/or written language data. We can view a scenario in Communicate as a directed acyclic graph of ‘annotated’ statements, where the annotations are an emotional value and a parameter score. A scenario represents the communication skills knowledge of a scenario author.

NLP often uses Latent semantic analysis [19] to analyse a document and relationships between documents. LSA constructs a word occurrence matrix, and uses singular value decomposition to create weighted vectors. These vectors represent the semantic value of a document. LSA compares two document vector representations and returns a cosine similarity. A high cosine similarity indicates related documents.

We use an approach similar to LSA to use the information in a scenario to develop a scenario specific corpus (RQ1). We create a specific corpus per scenario where we assign an *emotion* change value to each word in a statement. For example, if a statement, ‘Unfortunately, I’ve got bad news for you: I have to refer you to another therapist.’ triggers the *emotion* ‘Surprised’ in a virtual character, the value of ‘Surprised’ for all words in this statement is incremented with 1. We create a scenario specific corpus with either emotion or parameter values. If selecting a statement leads to incrementing a parameter such as for example ‘ResultOriented’ with 1, the value of ‘ResultOriented’ is incremented with 1 for all words in this statement. We want to investigate how well we can match using a corpus based on emotion or parameter values. We will investigate how well combining these values works in future work.

We cannot directly use LSA for our purpose, because LSA requires large amounts of text to create a set of vectors, and the total amount of text in a scenario is usually limited. We create an emotion vector or a parameter vector T for each unique word in a scenario. The dimension of the emotion vector is equal to the number of possible emotions, and the dimension of the parameter vector is equal to the number of different parameters. In the example sentence in the preceding paragraph, ‘unfortunately’, ‘news’, etc. are adapted with +1 in $T(Surprised)$ for an emotion vector. We add all occurrences of a unique word vector in a scenario, which results in a vector for the word in a scenario, for example unfortunately: {...,(‘Sad’, 4), (‘Surprised’, 2), ...}. We then divide each number in the vector by the number of occurrences of that specific word, resulting in a normalised word vector V.

$$V_{word} = \frac{T}{\#occurrences} \tag{4.1}$$

If ‘unfortunately’ occurs 10 times in a scenario, the vector value of ‘unfortunately’, for this scenario specific corpus is {..., (‘Sad’, 0.4), (‘Surprised’, 0.2),...}. E.g. the vector for ‘unfortunately’ in a scenario specific corpus looks like:

$$V_{unfortunately} = \{0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4, 0.2, 0.0, 0.0, 0.0\} \tag{4.2}$$

Creating a word *parameter* vector is similar. We calculate these vectors for all words in a scenario, which results in a corpus of words with their respective *emotion* or *parameter* vector for a scenario. We do not include irrelevant words, such as ‘the’, ‘a(n)’ and ‘to’. The resulting corpus is a scenario specific corpus.

4.4 Matching an open text to a scripted statement

The first step to match an open text input is to calculate a vector representation of the input. We create an open text input vector, for emotions, whose dimension is equal to the number of possible emotions, or for parameters, whose dimension is equal to the number of parameters in the scenario. We look up the words in an open text input in the scenario specific corpus. If a word appears in the scenario specific corpus, we add the value of the word in the corpus to the open input text vector. If it does not appear, we add a value of 0. We perform this for all words in an open text input. We then divide the summed open text input vector value by the numbers of words in the open text input to create a normalised open text input vector, see (4.3).

$$V_{input} = \frac{\sum V_{word}}{\#words} \tag{4.3}$$

Similarly, we create a vector for each statement option for the corresponding step in a scenario, V_{option}.

For matching an open text input vector to a statement option vector, we use either cosine similarity or Euclidean distance. Cosine similarity is also used in LSA. Both Euclidean distance and cosine similarity calculations are commonly used in determining vector similarity [112, 14].

The Euclidean distance between the vectors V_{input} and V_{option} with 'i' dimensions is:

$$EuclideanDistance = \sqrt{\sum (V_{input\ i} - V_{option\ i})^2} \tag{4.4}$$

The higher the Euclidean distance, the lower the similarity between two vectors. We determine a threshold value empirically (described later in this paper). If the Euclidean distance between an open text input vector and a statement option vector is lower than this threshold, we consider an open text input matched. The statement option vector with the lowest Euclidean distance to an open text input vector is the match. If all Euclidean distances between an open text input vector and statement options vectors are higher than the threshold value, we consider an open text input unmatched.

Cosine similarity is the cosine of the angle between two vectors (4.5) and equals the dot product of the two vectors divided by the product of the magnitudes of the two vectors.

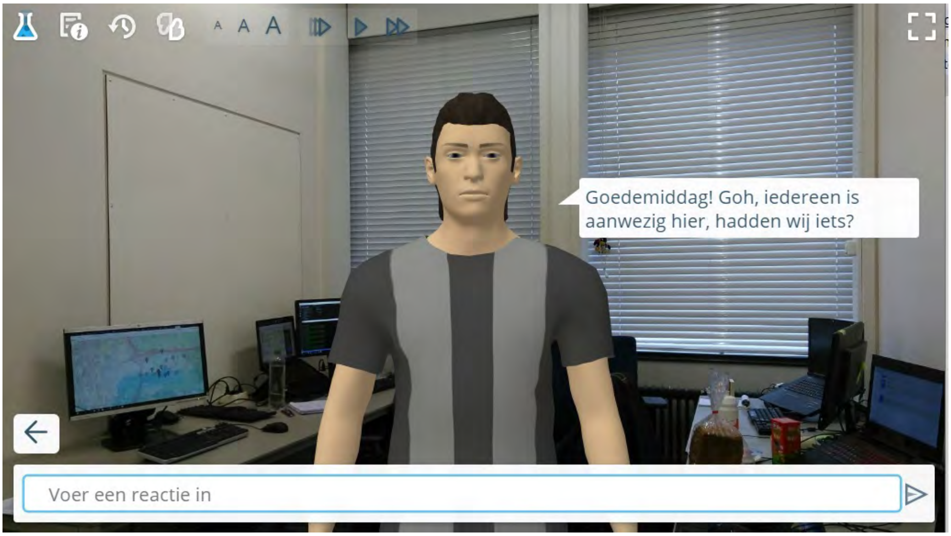
$$cosineSimilarity = \frac{V_{option} \cdot V_{input}}{\|V_{option}\| \times \|V_{input}\|} \tag{4.5}$$

A cosine similarity value is between 0 (no match) and 1 (perfect match) and a higher similarity value indicates a higher match. We calculate the cosine similarity between an open text input and each statement option at a step of a scenario. If all the cosine similarities between an open text input and every option at a step are below a threshold value, then the open text input is unmatched. We determine this threshold value empirically, described later in this paper. If at least one cosine similarity between an open text input and an option is above the threshold value, then the open text is matched. The statement option with the highest cosine similarity with a given open text input is the match.

Summarising, SSCM takes an open text input, creates a vector representation and calculates similarity to the statement options vectors at a step of a scenario. If at least one similarity is better than a threshold, then an open text input is matched, else unmatched. The statement option with the best similarity score is the match for a given open text input.

4.5 Creating a golden dataset

The limited availability of Dutch annotated corpora makes comparing different matching methods challenging. We perform sessions with students to gather open text and match



Goedemiddag! Goh, iedereen is aanwezig hier, hadden wij iets?

Figure 4.3: Open text box

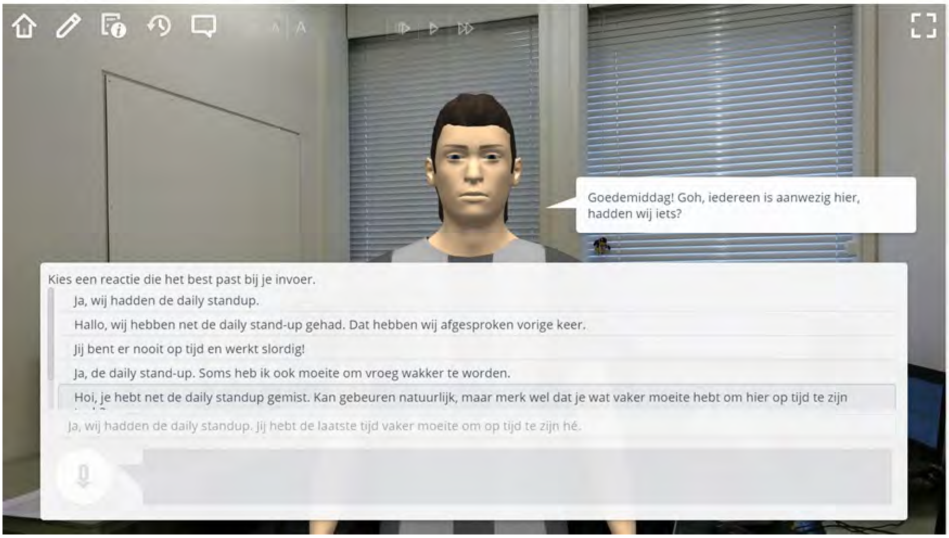
information, and use this data to create a 'gold-standard'. At our University, final year bachelor computer science students work in a team project and develop a software product for a real customer. In the spring-summer 2018 semester, we had a total of 82 students assigned to 8 software project teams. We developed a scenario called *Samenwerken* (Collaborate) in Communicate for practising collaboration skills. The parameters for this scenario are relation-oriented and goal-oriented. We conducted guided sessions to gather open-text utterances from students. 78 students gave consent to use their data for research. The age of the students ranged between 20 and 28 years.

We adapted Communicate to gather open text: a student gets an open-text input box in which she writes her response instead of choosing from the multiple choice options at each step, see Fig. 4.3. In this example step of the scenario *Collaborate*, a student needs to respond to the virtual character who says (translated), "Good afternoon! Wow, everyone is present, did we arrange something?"

A student inputs an open text, after which Communicate displays the available scripted statement options at this step and prompts a student to select the statement that is closes to her open text input, see Fig. 4.4. In this example, a student input the text (translated), "Yes, we had the daily standup. You have been struggling to be on time lately, right?".

At each step the system also offers an option *'Er is geen vergelijkbare reactie'* (Translated 'There is no comparable response'), which a student can select in case no scripted statement matches her input, see Fig. 4.5. In this example, the input text is, "Well, you

Open text input: 'Ja, wij hadden de daily standup. Jij hebt de laatste tijd vaker moeite om op tijd te zijn hé.'



Goedemiddag! Goh, iedereen is aanwezig hier, hadden wij iets?

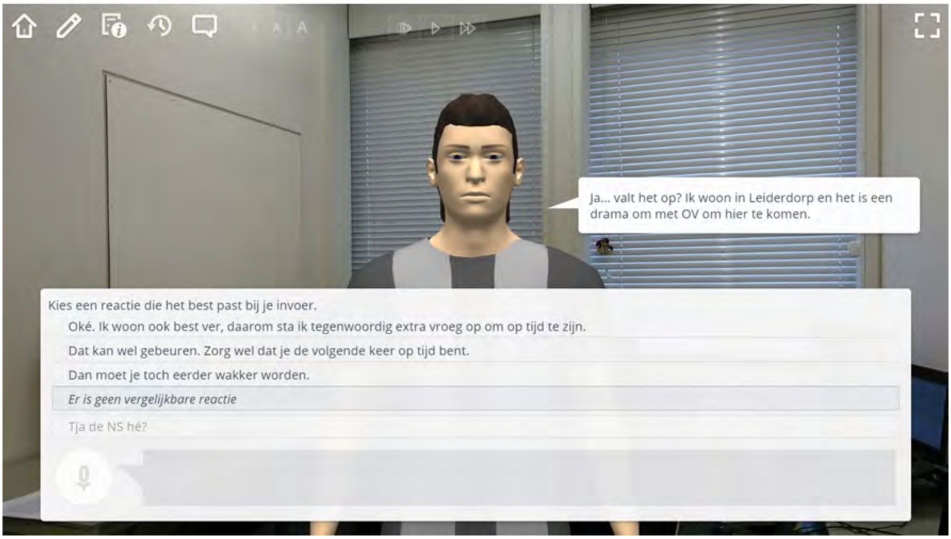
Kies een reactie die het best past bij je invoer.

- Ja, wij hadden de daily standup.
- Hallo, wij hebben net de daily stand-up gehad. Dat hebben wij afgesproken vorige keer.
- Jij bent er nooit op tijd en werkt slordig!
- Ja, de daily stand-up. Soms heb ik ook moeite om vroeg wakker te worden.
- Hoi, je hebt net de daily standup gemist. Kan gebeuren natuurlijk, maar merk wel dat je wat vaker moeite hebt om hier op tijd te zijn

Ja, wij hadden de daily standup. Jij hebt de laatste tijd vaker moeite om op tijd te zijn hé.

Figure 4.4: Case of a matched open text

Open text input: 'Tja de NS hé?', choose '*Er is geen vergelijkbare reactie*'



Ja... valt het op? Ik woon in Leiderdorp en het is een drama om met OV om hier te komen.

Kies een reactie die het best past bij je invoer.

- Oké. Ik woon ook best ver, daarom sta ik tegenwoordig extra vroeg op om op tijd te zijn.
- Dat kan wel gebeuren. Zorg wel dat je de volgende keer op tijd bent.
- Dan moet je toch eerder wakker worden.
- *Er is geen vergelijkbare reactie*

Tja de NS hé?

Figure 4.5: Case of a unmatched open text

know the NS" and there is no comparable response among the statement options available at this step of the scenario. A student chooses either a statement that matches her input or 'There is no comparable response' and moves to the next step in the scenario.

We received a total of 1143 open text input statements from students. The open text input statements had on average 9.84 words / 52.34 characters. The scripted statements in the scenario were similar in size: 10.44 words / 59.33 characters on average.

Two distinct annotators independently examined each student input and chose one of the available scripted statements at that step or *No response matches.* We compared the results from a student with the two annotators. We used a majority voting rule (2 of the 3 annotations) and were left with 974 statements in our dataset (i.e. the other 169 statements had three distinct votes). We ran a two-way random effects model of ICC and Cronbach's alpha which denoted acceptable agreement (Cronbach's alpha of .777) and a high average ICC measure of .742. An example of an item of the gold-standard dataset is shown in appendix 4.A.

We describe how we create a training, validation, and testing dataset from the 974 open text input statements and a corresponding match (or no match). A challenge is to randomly assign items to these sets while having a good distribution (more than half the items in training and approximately similar number of items in validation and testing datasets). In the scenario *Collaborate* that we used in collecting open text input from students, there are 24 computer statements. An open text input statement is a response to one of these 24 computer statements. To create training, validation and testing datasets, we use the following splitting procedure:

- We order the 24 computer statements by the number of corresponding open text input responses.

- We assign the open text input response items to the two most often and two least often responded computer statements to the training dataset. (Item responses to 4 computer statements.)

- From the remaining items, we assign the open text input responses to 4 computer statements each randomly to training, validation, and testing datasets respectively. (Item responses to 12 computer statements.)

- We assign the remaining items to the training dataset. (Item responses to 8 computer statements.)

- That leads to open text input response items to 16 computer statements in the training set, 4 in the validation set and 4 in the testing set.

- The number of items in the resulting training, validation and testing datasets are 533, 217 and 224 respectively (total 974 items).

- This results in a distribution of more than half the items in the training dataset and an almost equal number of items in the validation and testing datasets.

4.6 Results

To evaluate SSCM, we take the following steps:

- Run different configurations of SSCM on the training dataset. Calculate the accuracy for each configuration.

- Run the best performing SSCM configurations on the validation dataset. Shortlist SSCM configurations for the testing dataset.

- Apply NLP methods with which we want to compare on the training and validation datasets.

- Run shortlisted SSCM configurations and NLP methods on the testing dataset.

In addition to the base configuration described in section 4.3, we add the possibility to use additional NLP techniques in SSCM:

- Stemming: use stems of words. We differentiate between database and algorithmic stemming. For database, we use a table composed using Wiktionary (`https://www.wiktionary.org/`). For algorithmic, we use Snowball, a string processing language for stemming algorithms in information retrieval. Snowball extracts word stems using pattern matching. We use the Accord.NET framework (`http://accord-framework.net/`) in our implementation.

- Synonym: substitute words by synonyms. We create a synonym database using Wiktionary and Open Dutch Wordnet. Open Dutch Wordnet[88] is a dataset containing 28727 rows of a lemma and its synonym(s). The resulting database contains 314858 lemma-synonym sets.

A configuration of SSCM is a specific combination (see also Section 4.3) of: emotion or parameter vector, Euclidean distance or cosine similarity, stemming or no stemming, synonyms or no synonyms, and threshold value for a match.

Using an SSCM configuration, we take as input an open text input item, and calculate similarity scores between the input and all statement options. The output is an SSCM match item, which combines the various scores. An example of an SSCM match item is shown in appendix 4.A.

If at least one sscmScore is higher than a threshold value, then an open text input is matched (best statement score is the sscmMatch value), else unmatched (indicated as null for the sscmMatch value). For each specific configuration of SSCM, we calculate results using different thresholds, starting with a threshold value of 0.0 and incrementing by 0.025 for each calculation.

We compare the sscmMatch value against the goldenMatch value for an item. We perform this calculation and comparison for all items in the dataset. For each configuration, we calculate the accuracy percentage of correctly matched and unMatched items. We perform this calculation exhaustively on the training dataset and shortlist 22 configurations with high accuracy scores. We run these 22 configurations on the validation dataset to shortlist three configurations with: highest matched accuracy, highest unmatched accuracy and highest total accuracy.

The configuration with the highest matched accuracy in the validation dataset uses cosine similarity, a threshold value of 0.700, the emotion vector, stem all (input and options) using Snowball stemmer, and no synonyms. We abbreviate this configuration as 'Best Matched' to run on the testing dataset.

The configuration with the highest unMatched accuracy uses Euclidean distance, a threshold value of 0.800, emotion vector, no stemming, and synonyms. This latter configuration also has the highest total accuracy. We abbreviate this configuration as 'Best unMatched' to run on the testing dataset.

The complete results can be found in [38].

4.6.1 NLP matching methods for comparison

In an earlier independent investigation [95], we applied generic NLP methods on the datasets. The selected NLP methods were open source and incorporated a Dutch corpus:

- SpaCy (https://spacy.io) is an NLP framework and uses pre trained semantic models in Dutch for part-of-speech tagging and dependency parsing. SpaCy computes the cosine similarity of an open text input to a scripted statement based on average word vector scores of the two texts.

- WordNet has a lexical ontology in Dutch, and contains more than 115000 *syncsets* (similar words and corresponding relationships) [88]. For Dutch, the NLP methods include path length [69], Leacock-Chodorow and Wu-Palmer [13].

- String kernels compute the similarity between an open text and a statement option by counting common character n-grams. The most common string kernels to compute n-grams overlap are presence, intersection and spectrum [42]. We compute the average of these three types of string kernels for n-grams ranging in size between 3 to 7 characters.

- We use a neural network (NN) with a hidden layer to compute the best combination using the above methods.

Similar to SSCM, these comparison NLP methods are trained and validated on the training and validation datasets respectively to determine a threshold for a match [95].

4.6.2 Comparison

The testing dataset contains 147 matched and 77 unmatched items respectively. We show the accuracy results of the above NLP methods versus the top two SSCM configurations (Best Matched, Best unMatched) in the table below:

		Matched		**unMatched**		**Total**	
Totals		147		77		224	
Method		#	%	#	%	#	%
comparison NLP	SpaCy	38	26	77	**100**	115	51
	Path length	25	17	74	96	99	44
	Leacock-Chodorow	19	13	74	96	93	42
	Wu-Palmer	19	13	76	99	95	42
	String kernels	72	49	64	77	136	61
	Neural network	72	49	65	77	137	**61**
SSCM	Best Matched	74	**50**	36	47	110	49
	Best unMatched	35	24	67	87	102	46

Table 4.1: Comparison SSCM with other NLP methods

4.6.3 Discussion

SSCM performs slightly better than the NLP methods for matched open text input and performs worse for unmatched input. The presence of a scenario specific word in an open text input provides a higher similarity score to statements at a step of a scenario compared to generic NLP methods. It however also decreases the chance of no match, as the presence of a scenario specific word (e.g. 'Unfortunately') in an otherwise unMatched open text input could wrongly lead to a match.

SSCM configurations using parameter vectors performs worse than configurations using emotion vectors. The Collaborate scenario contains two parameters and we think that the limited number of parameters negatively affects matching. The difference between the vector values of two statement options using parameter vectors is perhaps too low in determining a correct match.

SSCM finds many more correct Matched items by using a lower threshold. An example SSCM configuration using Euclidean distance, stem all and a threshold value of 0.0, yields a percentage of 60.5% correctly Matched items on the training dataset. However, that configuration also leads to zero correct unMatched items.

Looking at the NLP methods with which we compare, string kernels performs the best. A probable reason for string kernel performing the best is that the method detects common keywords in two statements, while not being influenced by other words.

The neural network (NN) combination provides the best overall score. In our serious game Communicate a scenario changes over time and one needs to re-train a NN. In a game environment, the response time to a player input is important, SSCM is fast, and can compute a match in milliseconds.

4.6.4 Threats to validity

The datasets we constructed are not entirely uniform. The validation set contains relatively more unMatched items than the testing dataset which contains relatively more matched items. We may have selected a suboptimal SSCM configuration for the testing dataset. The same holds true for the threshold determination of the NLP methods with which we compare.

4.7 Conclusion and future work

We developed a method to create a scenario specific corpus using information in a scenario, and an algorithm to match an open text input using this scenario specific corpus. We collected open text input and created a training, validation and test dataset with a set of annotated Dutch statements. These datasets can be used to evaluate an NLP matching method. We evaluated SSCM on these datasets and compared it with selected NLP methods. SSCM performs slightly better for matched open text input than the other NLP methods. SSCM's main drawback currently is in detecting an unMatched open text input.

For future work, possibilities to improve SSCM include integrating synonyms and word lemmatization to handle variations of input categories. Another possibility is to use multiple methods and generate a confidence-score for a match. Response time is an important consideration for a game and calculating using multiple methods could be a bottleneck. We plan to use SSCM to handle open text input from a player during game play in Communicate.

4.8 Acknowledgements

This activity has partially received funding from the European Institute of Innovation and Technology (EIT). This body of the European Union receives support from the European Union's Horizon 2020 research and innovation programme.

The authors acknowledge Jordy van Dortmont and Michiel Hulsbergen for their help in annotating the open text input and Marcell van Geest for creating the datasets. We also acknowledge Stefan Ruseti, Gabriel Gutu-Robu and Mihai Dascalu from University Politehnica of Bucharest (UPB) for developing and applying the NLP methods with which we compare.

4.A Examples dataset item

Example golden dataset annotated item

```
{
"playthroughRID": 1140310840,
"predecessorID": "edit.30",
"input": { "text": "Dit klopt niet, je code werkt niet." },
"options": [
{ "text": "Goed om te horen dat het goed gaat. Het is even wennen
denk ik." },
{ "text": "Check je wel dat de software werkt als geheel voordat
je het pusht naar de main branch?"},
{ "text": "Probeer integratietesten uit te voeren om te controleren
of alles samen goed werkt."},
{ "text": "Merkte dat de code die jij hebt gepusht niet werkte met
andere componenten. Zo hadden we de opdrachtgever onze resultaten
niet kunnen laten zien."}
],
"playerMatch": 4,
"expertMatches": [4, 4],
"goldenMatch": 4
},
```

English translation of statements in the example item

input: This is incorrect, your code does not work.

- option1: Good to hear that things are going well. It takes some getting used to I think.
- option2: Do you check that the software works as a whole before you push it to the main branch?
- option3: Try to run integration tests to verify whether everything works well together.
- option4: Noticed that the code you pushed was not working with other components. This way we could not have shown our results to the client.

Example SSCM match item

```
{
"playthroughRID": 1140310840,
"input": { "text": "Dit klopt niet, je code werkt niet." },
"options": [
{
"text": "Goed om te horen dat het goed gaat. Het is even wennen
denk ik.",
"sscmScore": 0.6242484707939902
},
{
"text": "Check je wel dat de software werkt als geheel voordat
je het pusht naar de main branch?",
"sscmScore": 0.48416726387153114
},
{
"text": "Probeer integratietesten uit te voeren om te controleren
of alles samen goed werkt.",
"sscmScore": 0.42009607191028603
},
{
"text": "Merkte dat de code die jij hebt gepusht niet werkte met
andere componenten. Zo hadden we de opdrachtgever onze resultaten
niet kunnen laten zien.",
"sscmScore": 0.7547437669371
}
],
"playerMatch": 4,
"expertMatches": [ 4, 4 ],
"goldenMatch": 4,
"sscmMatch": 4
},
```

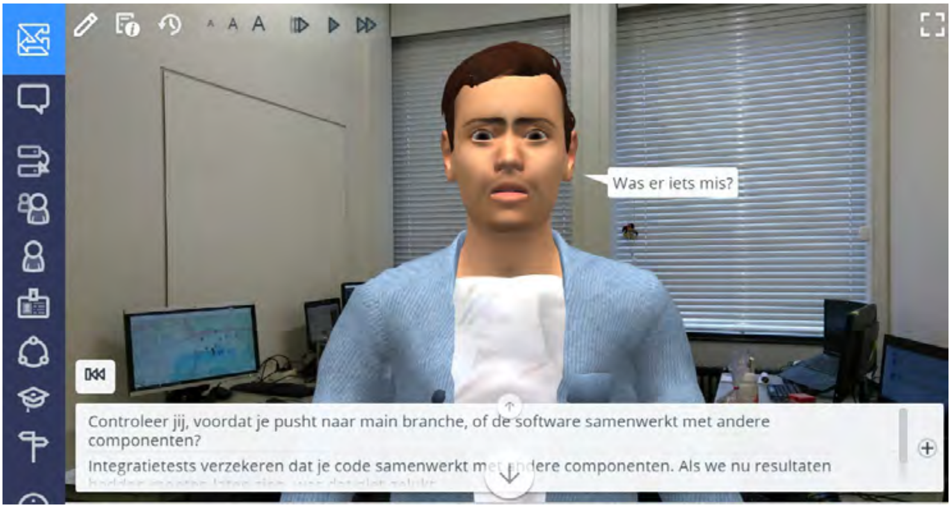
Chapter 5

Scaffolding open text input in a scripted communication skills learning environment

Serious games, as well as entertainment games, often employ a scripted dialogue for player interaction with a virtual character. In our serious game Communicate, a domain expert develops a structured, scripted scenario as a sequence of potential interactions in an authoring tool. Communicate is widely used and several domain experts have already developed over a thousand scenarios. In the original version of Communicate, a student 'navigates' a dialogue with a virtual character by clicking one of the multiple statement options at a step of a scenario. Open text response often requires more complex thinking from a student. In this paper we explore ways to handle open text input from a student at a step of a scenario. Our goal is to match open text to scripted statements using a Natural Language Processing (NLP) method and explore mechanisms to handle matched and unmatched input.

Published in International Conference on Games and Learning Alliance [1]

[1] https://link.springer.com/chapter/10.1007/978-3-030-34350-7_17


Was er iets mis?

Controleer jij, voordat je pusht naar main branche, of de software samenwerkt met andere componenten?

Integratietests verzekeren dat je code samenwerkt met andere componenten. Als we nu resultaten

Figure 5.1: Communicate game

5.1 Introduction

Communication skills are best learned through practice, in role play or with a simulated patient [8]. In Communicate [43], a serious game for training communication skills, a student practices a communication skills dialogue with a virtual character, see Fig. 5.1. Communicate is used in multiple domains to practice diverse communication skills and protocols, including assertiveness training, breaking bad news, visit to a pharmacy and collaboration.

Authoring content in an Intelligent tutoring system often requires significant effort [1], and authoring tool usability is often at the expense of expressiveness [73]. Communicate provides an authoring tool that combines expressive dialogue constructs with ease of use [55] and runs in a web browser. We release a dialogue scenario editor authoring tool as open-source as part of an EU project RAGE (Realising an Applied Gaming Ecosystem), see `gamecomponents.eu`. A communication skills teacher, usually a non-programmer, authors a scenario. A scenario is a sequence of interleaved subjects, where each subject is a directed acyclic graph consisting of a sequence of statements alternating between a virtual character and a player. A learning goal is typically encoded as a parameter in a scenario. An author assigns values to this parameter, typically an integer, per player statement option. An author also assigns an emotion to a player statement.

Communicate [43] presents statement options to a player at a step of a communication scenario, see Fig. 5.1. Choosing a player statement elicits the emotion assigned to the statement from a virtual character. At the end of a simulation, a player gets a score depending on her statement choices during the simulation. Martinez et al. [65] review cumulative research on the cognitive demand of multiple choice versus constructed response test item formats. Test item formats pose trade offs in cognitive demand, psychometric characteristics, and costs of administration/scoring. Multiple choice items often elicit low level cognitive processing from a student but are deterministic and easy to score. Open text response often requires complex thinking, but is more difficult to score. Students consider multiple choice fair, but they pay more attention to content when preparing for an open response test.

Our goal is to enhance Communicate by offering a student the possibility to enter open text at a step of a scenario. Adding an input box for a player to enter open text to our simulation is trivial, the challenge is to process this open text. To process open text, we use Natural Language Processing (NLP) techniques. In a previous experiment we gathered student open text input for a scenario and created a golden dataset on which we ran a range of open source NLP methods [57]. NLP methods that use local information (e.g. string kernels) give better matching results than NLP methods that use a generic corpora (e.g. semantic matching using latent semantic analysis, or paraphrasing). Even with a sizeable dataset, the results of NLP methods are not entirely accurate: NLP methods often require very large datasets to train a model [95]. It is unlikely we will obtain large datasets for all scenarios in Communicate: since authors can easily create and/or modify a scenario, we have over a thousand (variants of) scenarios about different communication protocols in different contexts. Our main contribution in this paper is to introduce open text input without significant additional effort from an author and/or extensive data gathering for a scenario, which is infeasible given the number of scenarios, and to explore mechanisms to handle matched and unmatched input. There is a wider implication for entertainment and learning games using scripted dialogues.

Realdon et al. [91] suggests a scaffolding structure in a learning environment to give a student an opportunity to learn. Given the limitations mentioned in the previous paragraph, our approach is to use an NLP method to match a student open text input and use scaffolding to handle matched and unmatched input. This NLP method takes a scripted scenario as input to build a scenario specific corpus [53]. For matching an open input text, this method uses the scenario specific corpus and returns a match score per scripted statement at a step of the simulation. We established a threshold score for the NLP method [57] and if all match-scores are below this threshold, we consider an open text input as unmatched. If at least one match-score is above the threshold, we consider an open text input as matched. Fowler and Barker [25] find that highlighting improves retention of material and to handle matched input, we highlight the best matching statement. To handle unmatched input, we look at Intelligent Tutoring Systems (ITS). Van Lehn [103] studies common behaviour of ITSs and recommends giving a hint for a next step when a student needs a hint. He recommends sequencing hints, starting with encouraging a student to think herself, and after that giving more detail about a next step. Our research questions (RQs) are: Can we handle matched input by highlighting a statement and unmatched input by providing a sequence of hints? We introduce variations in blended teaching sessions to answer our research questions.

This paper is organised as follows. Section 5.2 discusses related work. Section 5.3

describes our method to answer the research questions. Section 5.4 discusses results and Section 5.5 presents conclusions and future work.

5.2 Related work

This section gives related work on introducing natural language input in a learning environment or serious game. We look at different approaches to introduce open text input.

Autotutor [30] is a well known tutoring environment using natural language technologies. In AutoTutor, a student answers a question by means of a paragraph (approximately seven sentences) of text. Autotutor provides feedback on this text, and engages in a turn taking dialogue until a student arrives at a number of correct sentences. To handle open text, AutoTutor uses NLP methods like latent semantic analysis and speech act classifiers; techniques that focus on the general meaning of the input and functional purpose of an expression but also require a large dataset. It is unclear how AutoTutor processes unmatched input that does not fit any classifier in the script. The time and cost of authoring a script is considerable and requires extensive collaboration between computer scientists, cognitive psychologists and content experts.

Lessard [59] investigates the design of a natural language game conversation (built in ChatScript) based on experience from three digital games primarily involving a dialogue between a player and a virtual character. An NLP interaction provides creative conversational play, role playing, content contribution and non-linear conversations in these games. The drawbacks mentioned are: a player expectation that the system understands and responds like a person, leaky fictional coherence, unrestricted input, i.e. a player can say anything and a virtual character cannot have a response for everything and ‘amnesia’ specific to a chatbot that cannot ‘remember’ previous utterances.

Higashinaka et al [40] conduct an experiment to collect question answer pairs from users to create a chatbot character with consistent personality. The authors use a text engine to index the collected question answer pairs. They develop a chatbot that takes an open text input as a query to retrieve the most relevant question and responds with the pair answer. They use different retrieval methods and perform a ‘subjective’ evaluation to rate an answer in terms of naturalness and consistency. Higashinaka et al. find that the chatbot character has a consistent personality but the text retrieval methods are not entirely accurate. (In)accuracy of an NLP method is related to our work; we use scaffolding to handle an open text input.

Min et al [70] present a multimodal framework that predicts breakdowns in a student conversation with a pedagogical agent. The authors characterise a dialogue breakdown as a situation in which an agent misinterprets a student utterance and responds incorrectly. The framework incorporates natural language, eye gaze, student gender, and task state. Min et al. investigate this framework in a study of 92 middle school students in a game based learning environment. They find that incorporating eye gaze achieves high predictive accuracy for dialogue breakdown. We give a sequence of hints to a player when an open text input cannot be matched to a scripted statement.

In summary, there is a diversity of methods to introduce open text input in serious games and learning environments. Methods range from collecting and classifying input, semantic matching, chatbots, and agent technology. In our approach we use a scripted scenario as the basis, which ensures consistency, fictional coherence and control in utterances. We use an NLP method that takes a scripted scenario as input to build a scenario specific corpus [53]. We match an open text input using this corpus and use scaffolding to handle matched and unmatched input.

5.3 Method

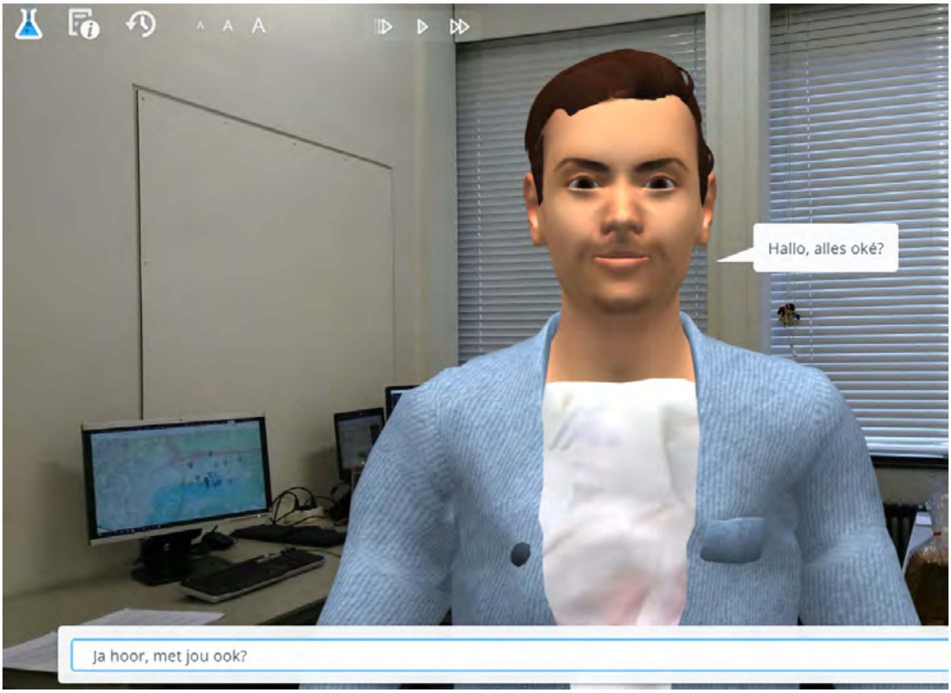
To answer our research question, we introduce variations in our teaching intervention. At our University, final year computer science students learn to work together in a software project team. During the project we provide a communication skills blended learning session per team consisting of 10 to 12 students each. In this session, students play a scenario about collaboration. A student needs to converse with a team mate (virtual character) who has not followed quality procedures (integration tests). In a session, an instructor introduces Communicate, students play the scenario, the instructor explains the communication protocol that forms the basis of the scenario, and there is a plenary discussion. To enable open text, we provide a text box at each step of a scenario, see Fig. 5.2.

We added hints to the Collaboration scenario that we use in the sessions. For example, we added the hint *'Try to give feedback about working together: his code does not work with other components, so you know that he has not performed integration tests'* to the subject *'Express'*, and we added the hint *'verbalise behaviour'* to the player statement, *'You pushed code that does not work with other components'*.

Our research method is to vary the aspect under consideration. For highlighting our treatment is to let a student match her open text input to a statement option with and without highlighting in separate rounds of sessions respectively. For hints, the treatment is to divide the students randomly into an experimental and control group, where within the same session, a student from the experimental group gets a sequence of hints and the control group get no hints. The design of the sessions in the semester of fall-winter 2018 is shown schematically in Fig. 5.3.

A student fills in an open input text (see Fig. 5.2, for example 'Ja hoor, met jou ook?') and the NLP match method takes this open input text 'a' and returns a match score per scripted statement option at a step of the scenario. In a previous experiment [57], we empirically determined a baseline threshold value for a match. If at least one statement option has a match score above the threshold value of the match method, we call that an mmSOM (match method some option matches). If a match method detects at least one mmSOM (∃ arrow upwards), Communicate displays all scripted statement options and highlights the best mmSOM match, see Fig. 5.4, in this example 'Jazeker, met jou ook?'.

Communicate asks a student to match the closest option to her open input text 'a' from the displayed statement options. If a student selects one of the statement options, we log for statistical comparison (see Section 5.4) as 'pSOM1' (player some option matches).



Hallo, alles oké?

Ja hoor, met jou ook?

Figure 5.2: Open text input box

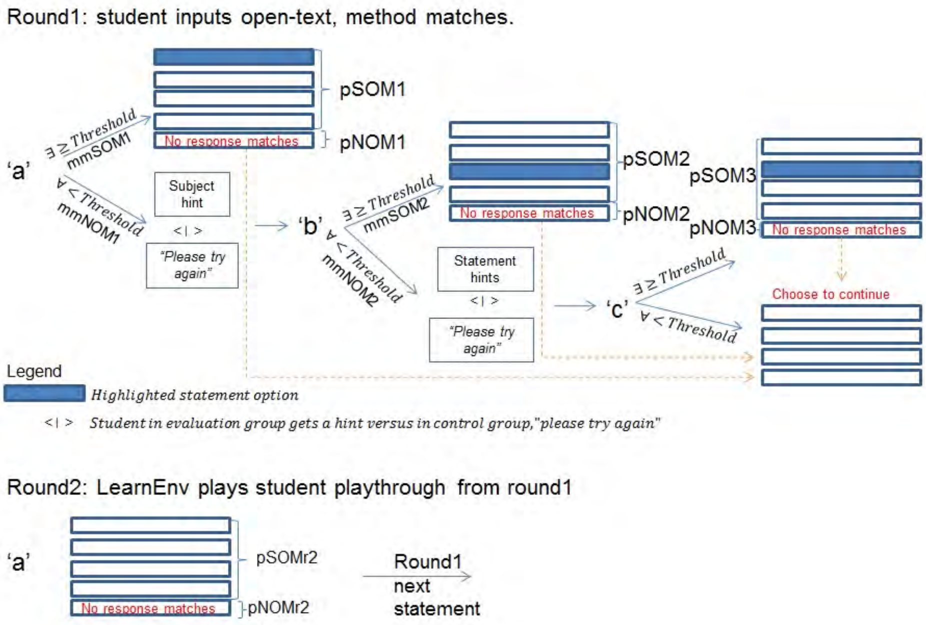
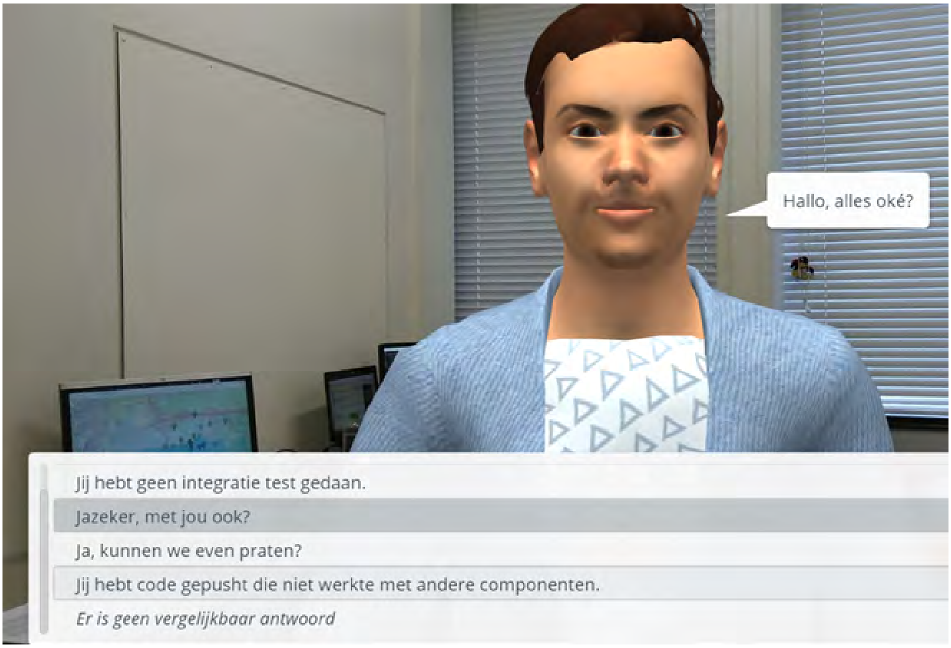


Figure 5.3: Setup within sessions of fall-winter 2018



Jij hebt geen integratie test gedaan.

Jazeker, met jou ook?

Ja, kunnen we even praten?

Jij hebt code gepusht die niet werkte met andere componenten.

Er is geen vergelijkbaar antwoord

Figure 5.4: Highlighted best match

We also log whether a student selects the highlighted option or another option. A student also gets an option to choose ‘No response matches’ (*Er is geen vergelijkbaar antwoord*). If a student chooses this option, we log this as ‘pNOM1’ (player no option matches). If a student finds that no option matches, Communicate asks the student to select one of the scripted statement options to continue the scenario, shown in Fig. 5.4 as the dotted line from the upper left part in round1 to the lower right part of round1.

If all NLP match method scores for an input statement ‘a’ are below the threshold value (∀ arrow downwards) of the match method, we say that the open text input is unmatched at that step in the scenario. Note that an open text input is either matched or unmatched using the NLP method. We log an unmatched input as mmNOM (match method no option matches). If the student is in the experimental group, Communicate gives a first hint and prompts to try again. This hint is a subject hint. If the student is in the control group, Communicate gives no hint and prompts a student to try again. A student enters a new response ‘b’, which is matched to obtain an mmSOM or mmNOM. In case the match method finds an mmSOM, we process it in the same way as described in the paragraph above. In case a match method detects an mmNOM again, Communicate displays the hints for all the statement options at that step, e.g: *Try to: verbalise behaviour; refer to agreement; ask a question* for the experimental group. For the control group Communicate again gives no hint and prompts a student to try again. A student enters a new response ‘c’, which again is matched to obtain an mmSOM or mmNOM. In case of an mmSOM, we again process as described above. In case of an mmNOM, Communicate asks a student to select one of the scripted statement options to continue the scenario.

For the treatment with no highlighting, we conduct a 2nd round of sessions with the students, three weeks after the 1st round. For all students who agreed in their user profile to store their open text input, Communicate presents a student her first playthrough from the 1st round. At each step of the playthrough, the first open text a student entered (the string ‘a’) in the 1st round is displayed, and Communicate displays all the statement options available at that step of the scenario and the special option *No response matches.* No statements are highlighted unlike in round1. There are a total of 210 open text input statements that students match. Communicate asks a student to match her input (string ‘a’) to an option, and we log if a student matches to a scripted option or if she chooses *No response matches.* Communicate displays the match the student made in the first round and continues with the next entered input from her playthrough from the 1st round until the end of the playthrough. Other students, who did not agree to store their open text input, played the scenario in multiple choice format.

5.4 Results and discussion

In the fall-winter semester 2018, there were a total of 52 students in five project teams, who played the same scenario in a modified version of Communicate where they typed in open text responses. Our research question is whether we can handle matched and unmatched open text input using highlighting and hints respectively. In the majority of open

	%
Highlighting effect cases	09.05%
True Positive (TP)	17.14%
True Negative (TN)	07.14%
False Negative (FN)	16.19%
False Positive (FP)	01.90%
NLP match differs from student match	20.48%
Student chooses inconsistently	28.10%

Table 5.1: Highlighting comparison table

text input in round1 (389 statements out of 503 statements, 77.34%), the match method matched with at least one of the scripted statements. For the remaining statements (114 of 503 statements, 22.66%) for which no match was found: Communicate showed a hint to students in the experimental group for approximately half the unmatched cases (56 statements, 11.13%) versus no hint in the control group (58 statements, 11.53%).

RQ: Can we handle matched input by highlighting a statement?

We compare the match choice from a student in the first round (best matched statement highlighted) versus the second round (no statement highlighted). In the first round, for an mmSOM, a player gets to match her open text input to one of the scripted statements while the best match is highlighted, see Fig. 5.4. She can choose either the highlighted statement, another statement, or *No response matches*. We analyse the different combination cases that occur. We discuss how a simulation without a highlighting scaffolding (referred as automatic match) would look like. We also gather insight into NLP matching versus student matching. All percentages in this subsection are from a total of 210 open text input statements that students match in round 2, see Section 5.3. The comparison between round1 (highlighting) versus round2 (no highlighting) is summarised in Table 5.1.

Does highlighting increase the chance of matching? We argue that highlighting had an effect when a student matches the highlighted statement in round1 and to a different statement; or to *No response matches* in round2. This occurs in 16 (7.62%) and 3 (1.43%) statements respectively, total 9.05% of the statements (210) matched in the 2nd round, shown in the first row of Table 5.1.

When a player matches her open input with the match method highlighted statement choice in round1 and chooses the same statement (unhighlighted) in round2, we call this a true positive (TP), 36 statements (17.14%) are TPs, shown in the second row of Table 5.1. When a player chooses *No response matches* in round2 and the match method also detects no match (mmNOM) in round1, we call this true negative (TN), 15 statements (7.14%) are TN, shown in the third row of Table 5.1. A TN open text input might be a signal of a missing scripted player statement option at a step of a scenario. In an automatic match, Communicate response would be accurate in these (TP and TN) cases. When a student matches her input to a statement in round2 but the match method detects no match in round1, we call this a false negative (FN, 34 statements, 16.19%). In an automatic match, Communicate would incorrectly provide a hint for an FN. When a student chooses *No response matches* in both rounds but the match method finds at least one match value above the threshold, we call this a false positive (FP, only in 4 statements, 1.90%). For an FP in an automatic match, a virtual character would provide a response, but Communicate should have given a hint.

When the NLP match method matches differently than a student statement match in both rounds (43 statements out of 210, 20.48%), in an automatic match a virtual character would provide a different response than possibly intended by the scenario author. E.g. a student entered, 'De code die je gisteren hebt gepushed conflicteert' for which the NLP method has the best match score to 'Ik wil het even met jou hebben over je werk van gisteren' whereas the student matches to 'Je code werkte niet samen met het geheel'. In this example, the NLP choice is not incorrect, however these statements are opportunities to examine if we can improve the NLP match method further.

There are cases when a student chooses inconsistently: when a student matches to an unhighlighted statement in round1 but chooses a different statement in round2 (43 statements out of 210, 20.48%), coincidentally the same amount but other statements than the statements that the NLP method matches differently than a student. We examined these statements: sometimes an input was a mix of two scripted statements or perhaps the statement options seemed similar to a student. Another inconsistency is when a student chooses *No response matches* in one of the rounds and an unhighlighted option in the other round (16 statements, 7.62%). In automatic processing these cases (total 28.10%, shown in the last row of Table 5.1), either a virtual character's response or a hint would be somewhat correct.

To answer our research question: using highlighting for matched open text input is not effective. The total of absolute errors in matching: false negative and false positive (18.09%) is limited and we argue on the basis of our results that for a matched open input, Communicate should not use highlighting, and automatically continue with the simulation.

RQ: Can we handle unmatched input by providing a sequence of hints? We evaluate if giving a hint in the experimental group leads more often to a matched input than in the control group.

We perform a chi-distribution test, which compares the observed cases versus expected values, see Table 5.2. The experimental (hint) group matches slightly better than expected after a subject hint and slightly worse than expected after a statement hint. The control group matches the other way around, slightly worse than expected after the first prompt to try again and slightly better than expected after the second prompt to try again. The differences are not significant (p-value 0.2521), and the reasons could be multifold. We paid attention while scripting the hints that a hint would not result in a match by copy-pasting. Perhaps a student tried the same words as in a hint and it could be that a student is perhaps frustrated by having to type something again. The match method is also not entirely accurate, perhaps a hint is similar to a student input

	Control group	Hint group
Initial unmatched statements (mmNOM)	58	56
Observed matches after subject hint	24	28
Expected matches after subject hint	26.46	25.54
Unmatched statements after subject hint	34	28
Observed matches after statement hints	16	9
Expected matches after statement hints	13.71	11.29

Table 5.2: Hint evaluation table

which was incorrectly unmatched. To answer our research question, giving a hint for unmatched open text input has no significant impact in our experiment. We recommend to not give hints, but instead to display the available statement options immediately to allow a player to continue a simulation.

5.5 Conclusions and future work

In this paper we take steps to enhance our learning environment from a multiple choice player input to open text player input. Enabling player open text input in our learning environment leads to more student interaction. Scaffolding, highlighting matched open text input and giving hints for unmatched input, has only a limited effect in Communicate. This result can perhaps be generalised for serious games that use a dialogue graph, want to incorporate open text input and have no extensive dataset. Our experiment results in a dataset of open text matched to a statement annotated by a student herself. The total of absolute matching errors by our NLP method on this dataset is small. The dataset provides a good distribution of student open text with corresponding matching and can be used by other NLP methods to improve match accuracy.

For future work, we recommend and plan to have guided sessions with minimal scaffolding, where in case of matched input a simulation continues as if a virtual character has understood the input (i.e. no extra highlight step to confirm a match) and in case of unmatched input, we present the available statement options to a player to select and continue a dialogue. This setup will involve no extra effort for a scenario author.

5.6 Acknowledgements

This activity has partially received funding from the European Institute of Innovation and Technology (EIT). This body of the European Union receives support from the European Union's Horizon 2020 research and innovation programme. The authors acknowledge Gemma Corbalan and Matthieu Brinkhuis for their help in the statistical analysis.

Chapter 6

Evaluation of interventions in blended learning using a communication skills serious game

Serious games often employ a scripted dialogue for player interaction with a virtual character. In our serious game Communicate, a domain expert develops a structured, scripted scenario as a sequence of potential interactions in an authoring tool. A player is often a student learning communication skills and a virtual character represents a person that a student talks to. In the original version of Communicate, a player ‘converses’ with a virtual character by clicking on one of the multiple statement options. Since 2018, we perform blended learning sessions for final year computer science students using Communicate. Our goal is to improve these sessions and in this paper, we apply the action research method over three semesters to iteratively improve these blended learning sessions. In the first semester, our baseline, we conduct sessions where students play a scenario in multiple choice format. In the second semester, we enhance Communicate by enabling a student to enter open text input in an improved scenario. In the third semester, we enhance a session by incorporating peer teaching. Students fill in an evaluation survey after a session and we compare the evaluation of students from the three semesters. Results show that student ratings are significantly higher in sessions incorporating peer teaching compared to the baseline.

Published in International Conference on Games and Learning Alliance [1]

[1] https://link.springer.com/chapter/10.1007/978-3-030-34350-7_31

6.1 Introduction

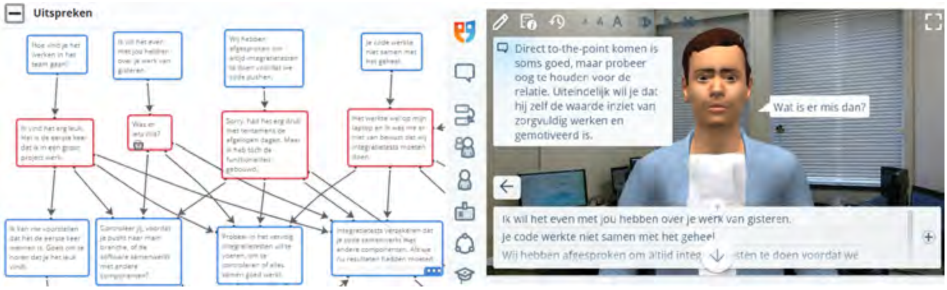
Most professions require communication skills, for example a doctor needs to communicate with a patient [98], while an IT expert discusses system requirements with a client. Generic communication skills such as conflict management, or being assertive are useful for many professionals. Communication skills are best learned through practice, in role play or with a simulated patient [8].

Communicate [43] is a serious game for practising communication skills. A communications expert/teacher scripts a scenario in an authoring tool that provides expressive constructs for variability in a dialogue [55]; see the screenshot at the left hand side in Fig. 6.1. In the original version of the game, a player navigates through a dialogue by selecting a statement option from one of the scripted player statements, see the screenshot at the right hand side in Fig. 6.1. This is similar to a multiple choice question answer format.

In this paper, we apply the action research method [62] to teach collaboration skills in a blended learning session to final year bachelor computer science students at Utrecht University. We aim to improve these sessions. Lewin [62] describes action research as: 'a spiral of steps, each of which is composed of a circle of planning, action and fact-finding about the result of the action'. In our case, a step is a set of blended learning sessions that we teach per semester. We collect student evaluation in these sessions. At the end of the semester, we critically reflect on the sessions and also analyse the student evaluations to identify potential improvements. We then introduce improvement(s) to the blended learning sessions in the next semester, again collect student evaluations, and compare these to the previous semester.

Final year computer science students at Utrecht University need to work in a project team (of 10 to 12 students each) to develop a software product for a real client. Prior to 2018, we gave a teamwork lecture in a single classroom session to all students. Since the semester spring-summer 2018, we provide a blended learning session: a face-to-face workshop for collaboration skills using Communicate per student project team. We teach a student to handle a collaboration situation where another student in a team (represented as a virtual character in Communicate) does not follow agreed team quality measures (integration tests). In consultation with communication skills experts, we developed a scenario called Collaborate for our students, which uses a protocol consisting of the subject-phases: Approach, Express, Discuss and Agree next steps. We present our study from blended learning sessions with successive batches of final year bachelor computer science students from three semesters: spring-summer 2018, fall-winter 2018 and spring-summer 2019.

We start with sessions where a student plays the Collaborate scenario in a multiple choice manner (in spring-summer 2018). These sessions form our baseline case. In a session an instructor introduces Communicate, students play a scenario, the instructor explains the communication protocol that forms the basis of the scenario, and students play the scenario again. After playing, students and the instructor have a plenary discussion. Students fill in an evaluation form after each session. This form gathers student



Direct to-the-point komen is soms goed, maar probeer oog te houden voor de relatie. Uiteindelijk wil je dat hij zelf de waarde inziet van zorgvuldig werken en gemotiveerd is.

Wat is er mis dan?

Ik wil het even met jou hebben over je werk van gisteren.

Je code werkte niet samen met het geheel

Wij hebben afgesproken om altijd integ[illegible]sten te doen voordat we

Figure 6.1: Communication skills scenario simulation

perceptions about some of the didactic aspects of Communicate. Dhaqane et al [20] find a correlation of student satisfaction with student performance in higher education. The evaluation form includes five statements on student satisfaction and self-efficacy, for example: I know better how to give relevant feedback, measured on a 5-point Likert scale, rating from 1 (completely disagree) to 5 (completely agree). Fig. 6.2 shows all the questions. Additionally, the form includes two open questions: 'What do you think about the game?' and 'Suggestions to improve the scenario?'

In addition to the evaluation of students, we also look at the application of activity theory to human computer interaction design [47] for potential improvements to our blended learning sessions. Multiple choice versus open text (test)questions are two common forms encountered in learning and education. Choosing from multiple options is similar to a multiple choice test, which often evokes low level cognitive processing, whereas open text response often requires complex thinking [65]. Ozuru et al [81] conduct a study to compare text comprehension from multiple choice and open text questions. Ozuru et al. find a positive correlation of the performance on open text questions with the quality of an explanation, and a positive correlation of the performance on multiple choice questions with the level of prior knowledge related to the text. In Communicate, a student reads and responds to a text from a virtual character, and gets a score on her performance in a scenario. This is a similar activity as in the study by Ozuru et al [81], where a student reads a short text while explaining preselected sentences. An open text response should lead a better quality of a response and possibly higher cognitive skills. In fall-winter 2018, we change the activity mechanism for a student to interact with the virtual character by typing open text input.

After fall-winter 2018, we evaluate the blended learning sessions again. A common feedback from students is the wish to play more scenarios. For the activity mechanism we change the interaction between students within a blended learning session to encourage insight in peer behaviour. Goldschmid et al [29] review peer teaching (when a student teaches another student) in higher education. The authors find that peer teaching among students can enhance active participation, develop skills in cooperation and interaction when used in conjunction with other teaching and learning methods. In spring-summer

2019 we incorporate peer teaching in our blended learning sessions by asking students to play Communicate in pairs and explain their motivation of a statement choice at a step of a scenario to each other. The interaction with a virtual character is reverted to the baseline case of spring-summer 2018, i.e. multiple-choice.

In this paper, we compare student evaluations from the three semesters. Our research question is: 'How does student evaluation vary with different interventions (multiple choice, open text and peer teaching) in blended learning sessions using Communicate?'

This paper is organised as follows. Section 6.2 discusses related work. Section 6.3 presents the interventions in the blended learning sessions in more detail and compares the student evaluations from the three semesters, Section 6.4 discusses results and Section 6.5 presents conclusions and future work.

6.2 Related Work

Provoost et al [89] perform a scoping review of embodied conversational agents (ECAs) in clinical psychology. The authors find that an ECA has a positive effect on user engagement and effectiveness of the interventions. However, Provoost et al. find only a limited number of evaluation and implementation studies of ECAs, in particular with larger sample sizes and control groups. The authors advocate a 'low tech' (simple) implementation to improve psychological interventions in the field.

Mazza et al [67] present a learning environment aimed at a student to learn how to interview a patient. A simulation in this environment includes a set of patient videos of interconnected doctor visits. A professional actor plays a virtual patient and simulates moods, attitudes and responses. The authors present a component that matches an open text speech input to a set of available choices. Eight students tested the free speech simulation and were in general positive about this component. However, the students noted problems with input for which no, or an erroneous, match was found. In the former case the software asks a student to rephrase her input, and in the latter case the software matches an input to an incorrect event. To address the problems, the authors propose to extend the set of available choices for a student. This seems to involve hiring the same actor to act responses to the extended set of choices, which might be cumbersome.

Van der Lubbe et al [102] develop a virtual training environment to teach a player (especially older adults) about situations where a potential swindler tries to gain trust. The author present a prototype consisting of six scenarios. In a scenario, a player interacts with a virtual character by either clicking on an option from multiple choice options or by speaking aloud an option. A speech module detects the level of assertiveness in case of a spoken response. The authors conduct an evaluation with a focus group of five security advisors. The focus group was in general positive about the prototype and expressed desire for more scenarios, being able to go back within a scenario, alter an answer and navigate to the tips/feedback menu.

Ochs et al [76] develop a virtual reality (VR) based simulation to train a doctor to break bad news. An ECA 'acts' as a virtual patient and follows a scenario. A player's (doctor's) speech input is interpreted real time by a human operator who selects a semantic match to one of the available 136 prototypical sentences. These prototypical sentences are based on a previously transcribed corpus of doctor interactions with trained actors playing a patient. The matched sentence is sent to a dialogue system that generates a verbal and non-verbal response of the ECA. Evaluation of the VR experience shows an positive impact of the environment display on the sense of presence, sense of co-presence, and believability of the virtual patient [77].

The work presented in this paper differs from the research described above: we incorporate multiple interventions (based on feedback from students) using the same learning environment over three semesters with successive student groups. We also compare the evaluation from the students over the three semesters.

6.3 Method and results

Action research [62] consists of cycles of taking action, and investigating the effects of the action. A step is a cycle of planning, action and fact finding. Our final year computer science students work in a software project as part of their curriculum towards the end of their bachelor program. Within this course, we provide a workshop session per team to address collaboration skills. Our goal is to improve these sessions and an action step-cycle is described in the Introduction (Section 6.1). Students from each semester fill in the same survey after a session, covering the following questions: five communication learning questions (see Fig. 6.2) with options on a 5-point Likert scale, and two open evaluation questions *(1. What did you think of the game? 2. Suggestions to improve the scenario?).* We analyse this evaluation and introduce improvements to the blended learning sessions. In this section, we describe the blended learning sessions in each semester in more detail in cycles of action and results.

6.3.1 Baseline multiple choice sessions of spring-summer 2018

In our first version of Communicate **(first intervention)**, a player navigates through a simulation and converses with a virtual character by clicking a statement option from one of the prescripted player statements. This is similar to a multiple choice question answer format; see the right hand side of the screenshot of our learning environment in Fig. 6.1.

In spring-summer 2018, we organised blended learning sessions where students played the Collaborate scenario. A student interacted with a virtual character by choosing an option from the multiple choice statements at a step of the Collaborate scenario. There were a total of 82 students assigned in eight project teams of 10 to 12 students each. After a project team session, students filled in the evaluation form. A total of 75 students filled in the form. Fig. 6.2 and Table 6.1 show the student Likert scale ratings on the five learning questions (label SS18 for spring-summer 2018 intervention).

In the feedback to the two open evaluation questions, students expressed the wish to input open text and had reservations about limited statement choices. To address this feedback, we conducted sessions in the same semester to gather open text input from

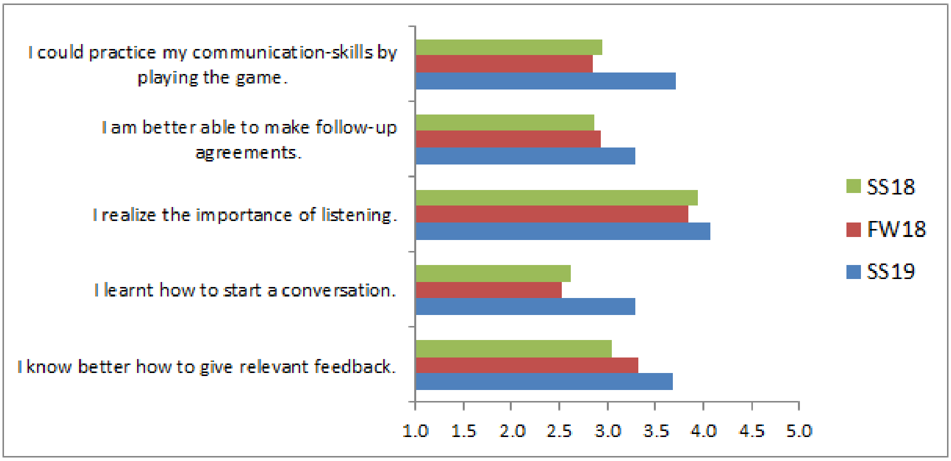


Figure 6.2: Chart representation of student rating on the five learning questions

students [95]. Gathering open text input also enabled us to create a dataset to develop and test several NLP methods [95], and to improve the Collaborate scenario. An improved scenario in the game is a teaching material improvement [74]. We applied a clustering algorithm to the list containing students' open text and the predefined statement options for each node in our scenario [57]. Two experts analysed the clustered input and improved the Collaborate scenario in multiple ways. First, if two predefined options at a step of a scenario were contained in the same cluster, then one of them was removed, since they were too similar when compared to the variety of student open text responses. Second, if a number of student open text responses formed a cluster that did not include any predefined scripted statement options, we added a new statement option (similar to the open text responses in the cluster), at that step of the scenario. Third, we modified a response from the virtual character to better frame a context and provide step feedback at some nodes of the scenario. We tested various NLP methods on the dataset created from open text input in spring-summer 2018, but the results were not entirely satisfying [95]. We choose the best matching NLP match method for use in the session of fall-winter 2018.

Another common feedback from the two open evaluation questions was that the VC was seemingly dumb and some students playing the scenario did not want to help / collaborate with the VC. Students also commented about the texture and gestures of the virtual character. We developed a new virtual character with better texture, gestures and animation. We simplified the scenario dialogue to give the VC a 'happy go lucky' character: an engaged and social team member, who works somewhat irregularly.

6.3.2 Intervention open text input, fall-winter 2018

In the blended learning session of fall-winter 2018 (**second intervention** abbreviated to FW18), we incorporate open text in Communicate. A student enters an open text at a step of the scenario Collaborate and the NLP method attempts to match this input to one of the scripted statement choices at that step of Collaborate. Since the NLP match method is not entirely accurate, we scaffold a match result. Communicate highlights the best match when an open text matches to at least one scripted statement and gives a sequence of hints when an open text does not match with any scripted statements at a step. In total 52 students were assigned to five project teams. After playing in open text sessions, 40 students filled in the evaluation form. The evaluation of the students in the five process questions is shown in Fig. 6.2 and Table 6.1 (label FW18 for fall-winter 2018 intervention).

The baseline is the evaluation from spring-summer 2018 (abbreviated to SS18) when the students played the Collaborate scenario (Collaborate) in multiple choice format. We compare the (Likert) student ratings on the five learning questions in Table 6.1 to the student ratings baseline spring-summer 2018. The first column in Table 6.1 displays the feedback Question that students rated on a 5-point Likert Scale rating from 1 (completely disagree) to 5 (completely agree). The second column denotes the semester and the third column shows the number of students who filled in the evaluation form. The fourth column shows the average rating of the students of the particular question and the fifth column shows the standard deviation to the mean score. In the fifth column, the p-value indicates the likelihood that an increase (or decrease) in rating is due to chance. A p-value <0.05 is significant, and a smaller p-value indicates a high likelihood that an increase (or decrease) in rating is due to an intervention.

Despite the extra scaffolding steps there was no significant deterioration (nor improvement) in the five learning questions ratings compared to spring-summer 2018. In the feedback to the two open questions, students were positive about the sessions but had reservations about the matching and expressed a wish to play more scenarios.

6.3.3 Intervention students peer teach and play multiple scenarios, spring-summer 2019

In spring-summer 2019 in our **third intervention** (abbreviated to SS19), we incorporate peer teaching and multiple scenarios in the blended learning sessions. To increase the number of scenarios, we requested communication skills teachers from other faculties to share scenarios and received scenarios on breaking bad news, self reflection, giving feedback etc. In these SS19 sessions, an instructor introduces Communicate, and students play multiple scenarios; at least *Collaborate* and one other scenario (e.g. breaking bad news, giving feedback etc). Students play the scenarios in multiple choice mode. The instructor explains the communication protocol of *Collaborate*: Approach, Express, Discuss and Agree. Thereafter, students play a scenario in pairs of two, where a student explains her statement choice to respond to a virtual character to the other student. We incorporate this action with the goal to provide insight into peer-behaviour, to teach each other and improve interaction in a session [29].

Question	Intervention	N	Mean	Std. Dev.	*p*-value
I could practice my communication skills by playing the game.	SS18	75	2.95	1.089	
	FW18	40	2.85	1.027	0.639
	SS19	78	3.72	0.952	<0.000
I am better able to make follow up agreements.	SS18	75	2.87	1.082	
	FW18	40	2.93	0.971	0.769
	SS19	78	3.29	0.899	0.009
I realize the importance of listening.	SS18	75	3.95	1.138	
	FW18	40	3.85	1.075	0.654
	SS19	78	4.08	0.879	0.431
I learnt how to start a conversation.	SS18	75	2.61	1.051	
	FW18	40	2.53	0.960	0.651
	SS19	78	3.29	0.955	<0.000
I know better how to give relevant feedback.	SS18	75	3.05	0.985	
	FW18	40	3.33	0.917	0.144
	SS19	78	3.68	0.830	<0.000

Table 6.1: Student ratings on the five communication process propositions

In total 81 students were assigned to eight project teams of which 78 students filled in the evaluation form after the sessions. In SS19 there is a significant improvement in four of the five learning questions ratings compared to the baseline, see Table 6.1. The fifth question: *'I realize the importance of listening.'* receives high ratings in all three semesters. The scenario seems to elicit the importance of listening in general. In the feedback to the two open questions, some students were 'pleasantly surprised' by the quality of the game and virtual character, some found that the game needed no further improvements, some wished for more feedback in *Collaborate*. We argue that the improved student evaluation is directly related to the third intervention.

6.4 Discussion

In the student evaluation, student ratings on the Likert scale questions show no significant difference in fall-winter 2018 sessions incorporating open text input versus the baseline of multiple choice input of SS18. In the response to the evaluation form open questions, students made remarks about the quality of an NLP match and that could be a reason that student ratings are not significantly higher. Students also complained about typing multiple times due to the extra scaffolding steps and that could be another reason for no significant rating increase.

Results show that the ratings of the spring-summer 2019 sessions incorporating peer teaching and multiple scenarios are significantly higher for four of the five questions compared to the baseline of spring-summer 2018, see Section 6.3.3. Application of activity theory to human computer interaction design [47] often focusses on interaction. Our results suggest that while interaction (e.g. multiple choice versus open text input) is important, it is crucial to investigate blended learning sessions with respect to pedagogical aspects such as, in our case, peer teaching.

6.5 Conclusions and future work

In this paper we plan, implement and evaluate interventions in our blended learning sessions using Communicate. We start with sessions using multiple choices in Communicate, enhance Communicate by enabling a student to enter open text input in an improved scenario with an improved virtual character and finally incorporate peer teaching and playing multiple scenarios in a session. Results show student ratings of open text input sessions do not significantly differ from multiple choice and that ratings of sessions incorporating peer teaching and multiple scenarios are significantly higher compared to multiple choice sessions.

For future work, we plan to have blended learning sessions again with multiple scenarios and peer teaching. As a difference to spring-summer 2019, we plan to have scenarios in both open text (in Collaborate scenario) and multiple choice (breaking bad news, giving feedback etc.) input modes. In these sessions, for open text input we plan to match with minimal scaffolding, where with matched input a simulation continues as if a virtual character has understood the input (i.e. no extra highlight step) and with unmatched input, we present the available statement options (i.e. no hint step). After playing, a debrief step will be introduced to ask the students in a session to reflect on a recent experience and have a plenary discussion on what went well and what could be improved. After guided sessions with a new batch of students, we can collect student feedback from the sessions.

6.6 Acknowledgements

This activity has partially received funding from the European Institute of Innovation and Technology (EIT). This body of the European Union receives support from the European Union's Horizon 2020 research and innovation programme. This research was also partially supported by the 644187 EC H2020 RAGE project.

This work has been developed in collaboration with the Software and Gameproject of the bachelor Computer Science in the faculty of Science, Utrecht University. The authors also acknowledge Majanne Wolters, Marjan van den Akker and Michiel Hulsbergen for their help in teaching and scenario improvement; and to Jordy van Dortmont and Marcell van Geest for implementing the desired changes to Communicate.

Chapter 7

Epilogue

In this chapter, I reflect on the work in this dissertation, and discuss some possible future work.

7.1 Conclusions

In the introduction, I distilled three aspects to consider for a serious game. These were to develop generic components and tools; to analyse learning aspects and game mechanics; and to evaluate and improve pedagogical aspects. These topics were the subject of the core chapters in this dissertation. In this section I touch upon some conclusions per topic.

7.1.1 Generic components and tools

We successfully created a generic dialogue integration component that can be used to develop and integrate dialogue scenarios seamlessly in a serious game. The architecture of the dialogue integration component enables a game narrative writer to develop a scenario independent of a developer. The dialogue integration component consists of an authoring tool[1] and a scenario parser/reasoner[2]. A scenario created using the authoring tool can be integrated into a game by a developer using the scenario parser/reasoner. The parser parses a scenario produced using the authoring tool, and returns an ID of the scenario. At run-time a game interacts with the scenario reasoner using the ID of the parsed scenario. The scenario reasoner then provides information about the possible following steps at each step in the series of interactions.

In the Rage project, scenario writers from Hull college[3] and the Randstad company[4] developed dialogue scenarios in specific instances of the authoring tool configured to their

[1] `https://github.com/UURAGE/ScenarioEditor`

[2] `https://github.com/UURAGE/ScenarioReasoner`

[3] `https://www.hull-college.ac.uk/`

[4] `https://www.randstad.fr/`

specific needs. The respective scenarios were integrated into the serious games WaterCooler[5] and JobQuest[6] by the game development companies Nurogames[7] and BiPMedia[8] respectively.

We also successfully created a scenario smells tool that a scenario author can use to detect potential scoring problems in a scenario in Communicate, already during scenario development. Scenario authors find assigning scores a difficult aspect of scenario development in Communicate. During evaluation, the smells tool processed most scenarios within user-acceptable processing time. Scenario authors were likely to use our smells tool. We also used the tool extensively while developing the Collaborate scenario to assign and to fine-tune the scores for the player statement options in the scenario. Though we developed the smells tool in the context of Communicate, we hypothesize that our results can be applied in any learning environment that uses scripted scenario dialogues with scoring for player statement choices.

7.1.2 Balancing pedagogical aspects and game mechanics

In Krathwohls' cognitive process taxonomy [51], multiple choice question-answers test cognitive process dimension levels 1 to 5 (remember, understand, apply, analyse and evaluate). Cognitive process dimension level 6 (creation) requires a student to construct text herself. To support a student's creation cognitive process, we want to add the possibility to input open text to respond to a virtual character at a step of playing a scenario in Communicate. Our hypothesis was that we could use information within a scenario to create a scenario specific corpus and use this corpus to match open text input from a player during gameplay. We call our method scenario specific corpus method (SSCM). We collected open text input from students using a modified version of Communicate. Two independent experts annotated each input match and we created training, validation, and testing datasets. These datasets can be used to evaluate an NLP matching method. We evaluated SSCM against generic NLP methods and found that SSCM slightly outperformed generic NLP methods for matched text but underperformed for unmatched text.

To handle open text input during gameplay, we introduced scaffolding: highlighting a statement when an input could be matched or providing a hint when an input could not be matched. In our experiment this scaffolding to handle open text in Communicate had only a limited effect. Our experiment did result in a dataset of open text matched to a statement annotated by a student herself. The dataset also provides a good distribution of student open text with corresponding matching. This dataset can be used to further improve an (other) NLP method(s) to improve match accuracy.

[5] https://www.youtube.com/watch?v=d_irGrEpBdc

[6] https://www.gamecomponents.eu/content/598/job-quest

[7] https://www.nurogames.com/

[8] https://bip-media.com/

7.1.3 Evaluating and improving pedagogical aspects

Demonstrating educational intervention efficacy [41] can establish the value of a serious game. We wanted to evaluate and improve blended learning sessions in which we used Communicate to teach collaboration skills. We applied the action research method [62] over three semesters, applying different interventions in the sessions. In the first semester (spring-summer 2018), we used Communicate with the original multiple choice approach, where a student responded to a virtual character by choosing a statement from multiple options. In the second semester (fall-winter 2018), we introduced open text input (with scaffolding) to respond to the virtual character. In the third semester (spring-summer 2019), we reverted to multiple choice; and introduced peer teaching in a session.

We compared the student feedback from the three semesters. There was no significant difference in the student ratings for the second semester sessions compared to the first semester sessions while there was a significant higher rating for the third semester sessions. These results suggest that in designing and employing serious games, it is crucial to investigate pedagogical aspects such as peer teaching.

7.2 Reflection

In this section, I reflect on some game components of Communicate relevant in the context of a serious game.

The first core component of our approach that I reflect upon is a (scripted) scenario. The strength of a scripted scenario is that it provides a teacher (scenario author) with total control on an utterance and emotion of a virtual character as a response to a player choice. A weakness of a scripted scenario is perhaps the predictability of a virtual character response. While we have expressive constructs in the scenario editor to increase variability, e.g. interleaving, conditional statements etcetera, playing a scenario multiple times can lead to increasingly predictable responses to a player statement choice. One can argue that exploring the various statement options is still conducive to learning, but increasing predictability could perhaps be less engaging for a student. For comparison, a pedagogical agent [72, 21, 64, 10] with an 'internal' emotional state, could provide more player engagement but less response predictability. Our scenario authoring tool, balancing usability and expressiveness, is a strength, as it enables a game writer to create complex scenarios without being proficient in programming. In general, I recommend serious game environments to provide an authoring tool to a teacher to create and modify content to tailor to a specific audience.

Open text input processing in a serious game is still some way off (Chapter 5, [59, 85]). Inputting open text triggers a higher order cognitive skill [51, 65] in a player (a strength) versus selecting from options. The unpredictability of matching(/understanding) open text input and providing an 'incorrect' response is a weakness. In addition to this strength and weakness, Bosman et al. [10] also argue that selecting an option from multiple statements has a risk that a player feels forced to select a statement that she would never choose in real life. For open text input, Bosman et al. note the risk that a system generates 'backup' responses like 'I do not understand, please rephrase' too often. This can also lead to player disengagement.

In our experiments with open text input, we only looked at typing open text and not speech open text. The practical reason was that students playing the scenario in the same room would be distracted by each other's speech but not by each other's typing. There is an option to input free speech in Communicate that makes use of the Google speech to text API. While I did not investigate further in speech open text input in Communicate, my reflection while playing a scenario in open speech text mode is that intonation provides subtle differences in meaning. Trying to perceive such subtle differences would involve an entirely new investigation.

Another aspect not part of my research (yet) is with respect to a virtual character. One can see the evolution of the virtual character in the screenshots from Chapter 4 to Chapter 5. Students playing the scenario with the virtual character in Chapter 4 often made comments about the virtual character (e.g. 3D model looked pale, better animation, improved graphics etc.) in the evaluation open questions. There were few such comments with the improved virtual character in Chapter 5, though one student mentioned 'uncanny valley'. Mori [71] hypothesized that a human's feeling towards a robot becomes more positive as the robot appearance becomes more human. However, after a certain point of resemblance, a human's feeling becomes negative. The 'uncanny' refers to a human feeling of unsettling strangeness and the 'valley' refers to the dip in feeling from positive to negative. My reflection is that investigating the effect of virtual character appearance, gestures and utterances in a communication skills serious game is another entirely new line of research.

7.3 Future work

In this section, we look at some possible future work related to the three aspects to consider for a serious game, namely tools, game mechanics and pedagogical value.

Looking at tooling to further help a scenario author in improving a scenario in Communicate, perhaps a tool to visualise a dialogue path a student takes in playing a scenario is useful. An aggregation and analysis of these dialogue paths can provide insight into statement choices and paths that are (not) often chosen in playing a scenario. Combining this analysis with psychometric techniques to determine item difficulty and item discrimination [33] can provide a scenario author with insight on improving statements choices and dialogue paths in a scenario.

For future work on gameplay, we could further investigate different player input mechanisms in playing a scenario in Communicate. A possibility is to offer open text input with minimal scaffolding, where in case of matched input a simulation continues as if a virtual character has understood the input (i.e. no extra highlight step to confirm a match) and in case of unmatched input, we present the available statement options to a player to select and continue a dialogue. This setup does not require extra effort for a scenario author.

Another interesting input mechanism to consider is nonverbal input, for example the affective state of a student while playing a scenario. In the Rage marketplace[9], there are two components for affective state detection: Facial emotion detection[10] and Arousal detection using galvanic skin response[11]. Both components can work real time, though require extra hardware: a webcam and biofeedback electrodes respectively. The Facial emotion detection (FED) component analyses human player facial expressions to determine an emotion. This emotion is one of seven emotions: happy, sad, surprise, fear, disgust, anger and neutral. Galvanic skin response (GSR) requires two electrodes to be placed on two adjacent fingers, and can be used to determine autonomic arousal of a human. GSR can be integrated and used in a serious game to detect player arousal during gameplay. Integrating these components into Communicate is future work. In this integration, we determine the affective state of a player while playing a (step of a) scenario. This information can be used for research, for example: What affective state does a virtual character statement evoke in a player?

If we can accurately perceive the affect of a human player, we want to investigate what we would need to do to use the perceived affect of a human player to modify the response of a virtual character in Communicate. This would require a change in the structure of a scenario.

In terms of affective computing [84], perceiving the affect of a player, responding accordingly and expressively would shift Communicate to category 4 of the affective computing matrix [84] from currently category 2 (computer expresses affect but cannot perceive affect).

Finally for future work on pedagogical aspects, we could investigate ‘skill prediction’ in Communicate by developing and using student models. A serious game provides an opportunity for stealth assessment [97]. Alonso et al. [4] use game learning analytics data to predict student knowledge after playing a serious game. Intelligent tutoring systems [75] often use a student model to represent the emerging skills and capabilities of a student. An area of future work could be to develop a student model in Communicate. A set of scenarios in Communicate, each validated using psychometric techniques [33], could form a learning path for a student. Learning analytics data could also provide a topic based personalization approach [99] for student learning, where each topic could be a scenario.

[9] `www.gamecomponents.eu`

[10] `www.gamecomponents.eu/content/339/real-time-facial-emotion-detection-software-component`

[11] `www.gamecomponents.eu/content/233/real-time-arousal-detection-using-galvanic-skin-response`

Chapter 8

Samenvatting

Is de term ‘serious game’ een oxymoron [22]? Of zijn serious games bedoeld om de wereld te veranderen[1]? De realiteit zit er misschien ergens tussenin. Djaouti et al. [22] definiëren een serious game als een stuk software dat een serieus doel combineert met een videogamestructuur (game). De eerste vermelding van een spel met een serieuze context is Chaturanga, de voorloper van het schaakspel, uit het 7e-eeuwse India, dat een militaire context gebruikt in een bordspel [82]. De eerste digitale games met een serieuze context werden toevallig ook ontwikkeld voor de militaire sector [108]. In de loop der jaren zijn serious games in verschillende sectoren ingezet zoals: onderwijs, reclame, gezondheidszorg, cultuur, ecologie.

De mensheid wordt wellicht bepaald door het vermogen om met elkaar te communiceren. Communicatieve vaardigheden als een wetenschappelijke discipline ontstond in de 20e eeuw, grotendeels uit drie andere onderzoeksgebieden: psychologie, sociologie en politieke wetenschappen [86]. Communicatieve vaardigheden worden aangemerkt als één van de belangrijkste vaardigheden voor de 21e eeuw [58]. Oefenen door training op het werk of in rollenspellen met een gesimuleerde patiënt zijn effectieve manieren om communicatieve vaardigheden te leren [8]. Serious games bieden de mogelijkheid om een vaardigheid effectief te onderwijzen en leren [16]. In de afgelopen twee decennia zijn er verschillende serious games voor communicatieve vaardigheden ontwikkeld [43, 11, 28]. Deze serious games bieden een leeromgeving waarin een speler communicatieve vaardigheden kan oefenen en leren.

De entertainment games industrie is een volwassen, dynamische en groeiende industrie. In 2013 was deze industrie wereldwijd al meer dan $56 miljard per jaar waard, waarvan meer dan $15 miljard per jaar alleen al in Europa [101]. De serious games industrie is anders dan de entertainment games industrie en dit verschil heeft zijn weerslag op de aard van serious games. Het verschil tussen deze bedrijfstakken kan worden geïllustreerd aan de hand van het vijfkrachtenmodel van concurrentiestrategieëen van Porter [87]. De vijf krachten zijn: de concurrentie tussen bedrijven in de industrie, de

[1] https://www.growthengineering.co.uk/10-serious-games-that-changed-the-world/

dreiging van nieuwkomers, de dreiging van vervangende producten of diensten, de onderhandelingspositie van kopers en de onderhandelingspositie van leveranciers.

De entertainment games industrie wordt gedomineerd door Nintendo, Sony en Microsoft (in de rest van deze samenvatting de 'grote drie' genoemd). Er is een sterke onderlinge concurrentie tussen deze grote drie. Ze strijden om marktaandeel en kopers. De serious games industrie is daarentegen gefragmenteerd [101], met kleine bedrijven die zich richten op een niche en met weinig onderlinge concurrentie. De leveranciers van de grote drie zijn gamestudio's en hardware leveranciers. Gezien de omvang van het marktaandeel van de grote drie en het grote aanbod van gamestudio's is de onderhandelingspositie van leveranciers niet groot. Voor de grote drie is het wenselijk dat een koper zich bindt aan hun respectievelijke producten en niet makkelijk kan overstappen naar een ander. De goede onderhandelingspositie van de grote drie zorgt ervoor dat leveranciers games maken die specifiek draaien op de gameconsole van één van de grote drie. Serious games draaien daarentegen meestal op generieke devices zoals een PC. Vaak worden serious games ontwikkeld en vermarkt door hetzelfde bedrijf. Serious games bedrijven zouden kunnen profiteren van herbruikbare en interoperabele componenten op generieke devices om een serious game te ontwikkelen met een snellere time-to-market [100]. Serious games bedrijven volgen echter vaak een nichestrategie en maken serious games met een lage mate van herbruikbaarheid en interoperabiliteit [37]. Dat zou kunnen verklaren waarom er niet veel generieke componenten algemeen verkrijgbaar zijn in de serious games industrie.

De dreiging van potentiële nieuwkomers in entertainment games is laag vanwege de omvang en schaalvoordelen van Nintendo, Sony en Microsoft, wat leidt tot een hoge toetredingsdrempel voor nieuwkomers. Bij het ontwikkelen van serious games [41] is er een financieel investeringsrisico voor toetreding tot een nichemarkt. Dit risico kan wellicht worden verkleind door de ontwikkelingskosten van een serious game te verlagen, bijvoorbeeld door kant-en-klare componenten te gebruiken.

De kopers van serious games verschillen van kopers van entertainment games. Terwijl kopers van entertainment games meestal consumenten zijn, zijn kopers van serious games meestal grote instellingen, bijvoorbeeld universiteiten, ondernemingen of organisaties in de publieke sector. Entertainment games worden doorgaans geleverd door business-to-consumer (B2C) industrie, terwijl serious games doorgaans een rol spelen in een business-to-business (B2B) industrie. De onderhandelingspositie van kopers bepaalt de winst die met een product behaald kan worden [87]. Een entertainment game is gericht op een breed publiek, terwijl een serious game vaak gericht is op een klein publiek met specifieke leerdoelen [41]. In de entertainment games industrie kan, gezien de schaalvoordelen voor de grote drie, worden gesteld dat een individuele koper niet veel onderhandelingsmacht heeft. Hoewel een entertainment game tegemoet komt aan de kwaliteits- en prijseisen van het grote aantal kopers in de detailhandel, is het onwaarschijnlijk dat het wordt aangepast aan een individuele koper. In de serious games industrie kan een koper van een grote instelling onderhandelingsmacht uitoefenen op een serious games bedrijf. Een docent die een serious game toepast in blended teaching is een voorbeeld van een (indirecte) institutionele koper. Het game aspect en het leren aspect hebben verschillende mechanismen en een docent die een serious game gebruikt, moet vaak een balans vinden tussen leeraspecten en gamemechanica [5]. Deze behoefte om de 'serious' en 'game' aspecten in evenwicht te brengen, is een groot verschil met een entertainment game. Een docent kan invloed uitoefenen op een serious games bedrijf om dit evenwichtsaspect voor een serious game aan te pakken.

De laatste van Porters vijf krachten is de dreiging van substituten. In de entertainment games industrie zijn substituten zogenoemde 'Indie games' (Independent games), die vaak draaien op generieke devices zoals een PC. Voor serious games zijn er veel substituten zoals MOOC's en intelligente tutoring systemen (ITS); deze 'substituten' zijn vaak volwassener dan serious games. Men kan stellen dat een serious game op zich een substituut is. Een effectieve manier om marktaandeel te winnen (of te behouden) voor producten (substituten) is door differentiatie en waarde aan te tonen [87]. Het is daarom belangrijk om het pedagogische effect van een serious game te evalueren om de waarde vast te stellen [7].

Uit de bovengenoemde kenmerkende verschillen tussen de entertainment en de serious games industrie haal ik drie aspecten:

- A1: Ontwikkel generieke componenten voor het 'samenstellen' van een serious game, en tools voor het maken/wijzigen van de inhoud van een serious game. Aangezien serious games vaak op generieke platforms worden gespeeld, maken standaard verkrijgbare generieke componenten van goede kwaliteit een snellere time-to-market mogelijk en de toetredingsdrempel laag voor zowel huidige serious games bedrijven als nieuwkomers [41]. Met een tool kan een docent de inhoud van een serious game creëren, aanpassen of analyseren om deze af te stemmen op een specifiek publiek.

- A2: Analyseer de gameplay in een serious game om een goede balans tussen leren en spelen te creëren [5].

- A3: Evalueer en verbeter pedagogische aspecten van een serious game [7] om de waarde van de serious game aan te tonen [41].

Dit proefschrift richt zich op deze aspecten (A1-A3) in de context van Communicate [43], een serious game voor het oefenen met communicatieve vaardigheden. De Communicate game is in eerste instantie ontwikkeld voor de faculteiten Diergeneeskunde, Farmacie, Geneeskunde, en Psychologie van de Universiteit Utrecht, die het gebruiken om hun respectievelijke studenten te trainen in communicatieve vaardigheden. Een speler, bijvoorbeeld een arts in opleiding, oefent een dialoogscenario met een virtueel karakter, bijvoorbeeld een patiënt. Een dialoogscenario is vaak gebaseerd op een communicatie protocol, bijvoorbeeld het slechtnieuwsgesprek [6]. Communicate is iteratief ontwikkeld door drie opeenvolgende studentprojectteams van de opleiding Informatica van de Universiteit Utrecht[2]. Omdat Communicate op grotere schaal werd gebruikt werd een startup

[2] Studenten in het laatste jaar van de bacheloropleiding Informatica aan de Universiteit Utrecht werken samen in een softwareprojectteam en ontwikkelen een softwareproduct

bedrijf DialogueTrainer[3] opgericht. DialogueTrainer gebruikt Communicate om klanten online training in professionele gesprekken aan te bieden. De serious game Communicate is sindsdien omgedoopt tot DialogueTrainer. Ik gebruik de naam Communicate in dit proefschrift, aangezien het eerste werk en de academische papers verwijzen naar de serious game als Communicate.

Ik heb bijgedragen aan twee door de EU gefinancierde projecten, RAGE (Realizing an Applied Gaming Eco-system[4] en PrepDoc. Het RAGE-project heeft een aantal uitdagingen in de serious games industrie aangepakt. Eén van de belangrijkste doelstellingen van RAGE was het ontwikkelen van een marktplaats van generieke gamecomponenten. In PrepDoc hebben wij een serious game omgeving (Communicate) toegepast om een oudere volwassene voor te bereiden op een consult met een arts. In de volgende secties ga ik in op de bovengenoemde drie aspecten voor een serious game (A1 - A3), in de context van Communicate en de RAGE & PrepDoc projecten.

8.1 Algemene componenten en tools

Engström et al. [23] beschrijven de uitdagingen voor een dialoogauteur bij het schrijven van een dialoogscenario voor een game. Engström et al. merken op dat er geen generiek hulpmiddel of formaat is om dialoogscenario's te creëren. Een dialoogauteur gebruikt soms een hulpmiddel om een prototype van een dialoogscenario te schrijven en testen. Engström et al. bevelen aan om een gevarieerde set tools te creëren om een dialoogauteur te ondersteunen bij het schrijven van een scenario. In intelligente tutoring systemen vereist het schrijven van inhoud ook aanzienlijke inspanning van een expert [1]. Bruikbaarheid van een authoringtool in een ITS gaat vaak ten koste van expressiviteit [73]. Engström et al. constateren dat tools voor het ontwikkelen van spelscenario's (bijv. Twine[5]) vaak een scripting programmeerinterface hebben. Een dialoogauteur heeft vaak weinig ervaring met programmeren [23]. Samengevat, een tool voor het schrijven van een scenario moet een balans vinden tussen bruikbaarheid en expressiviteit.

Communicate biedt een dialoogauteur, vaak een docent communicatievaardigheden (hierna scenarioschrijver genoemd), een editor voor het schrijven van een scenario. Deze editor is ontwikkeld met het oog op bruikbaarheid voor een communicatie docent [43]. Een student kan daarna een scenario spelen, en een dialoog met een virtueel karakter navigeren om communicatieve vaardigheden te oefenen. De onderzoeksvragen in hoofdstuk 2 zijn: Kunnen we gemeenschappelijke kenmerken van serious games met gescripte dialogen distilleren? Kunnen we vervolgens deze kenmerken gebruiken om een formaat voor scenario's te definiëren en om een authoringtool te creëren?

Als onderdeel van het RAGE-project hebben we een component voor dialoogintegratie ontwikkeld. De architectuur van de component stelt een scenarioschrijver in staat om een scenario te ontwikkelen onafhankelijk van een ontwikkelaar. De dialoogintegratie

[3] `https://www.dialoguetrainer.com/`

[4] `http://www.rageproject.eu/`

[5] `https://twinery.org/`

component bestaat uit een scenario editor[6] en een 'scenario reasoner'[7]. Een scenario dat met de scenario editor is ontwikkeld kan door een ontwikkelaar in een game worden geïntegreerd met behulp van de scenario reasoner. Een scenarioschrijver kan (talrijke) scenario's ontwikkelen in de scenario editor, waarbij elk scenario een bepaalde situatie aan de orde stelt. Een gameontwikkelaar kan een scenario eenvoudig integreren in een serious game, waardoor kosten en ontwikkelingstijd kan worden bespaard.

In het Rage-project hebben scenarioschrijvers van Hull college[8] en Randstad[9] scenario's ontwikkeld in de scenario editor. De respectievelijke scenario's zijn geïntegreerd in de serious games WaterCooler[10] en JobQuest[11] door de gamebedrijven Nurogames[12] en BiPMedia[13] respectievelijk.

Als een toets resultaat niet als eerlijk wordt ervaren heeft dat een negatieve invloed op het leren [65, 68]. We verwachten dat hetzelfde geldt voor het spelen van een scenario in Communicate, d.w.z. als de score (het resultaat) na het spelen van een scenario niet als eerlijk wordt ervaren, kan dat het leren van een student negatief beïnvloeden. Een docent (scenarioschrijver) wil graag weten of er mogelijke ontwerpproblemen in een scenario zijn voordat een student een scenario speelt en een score krijgt. In de entertainment games industrie en ook in serious games industrie, wordt vaak 'playtesten' gebruikt om potentiële fouten op te sporen voordat een game wordt uitgebracht. Playtesten is nuttig, maar vereist een aanzienlijke tijdsinvestering. In de entertainment games industrie kunnen veel speltesters worden gerekruteerd, vaak gratis, om een 'bètaversie' te evalueren/testen. De aanzienlijke playtest inspanningen vooraf kunnen een belemmering vormen voor een ontwikkelaar van serious games.

Naarmate scenario's complexer worden, wordt het moeilijker om ze te ontwikkelen en neemt de kans op een ontwerpprobleem toe. Er zijn een aantal manieren/mogelijkheden om een scenarioschrijver te ondersteunen. Kasperink [48] past bijvoorbeeld een argumentgebaseerde benadering toe [46] om het scenario 'slecht nieuws' te valideren in Communicate, en Gupta [33] gebruikt psychometrische technieken om de kwaliteit van toetsing in een scenario te bepalen. Deze methoden vereisen spelersgegevens om een analyse uit te kunnen voeren.

In hoofdstuk 3 gebruiken we een concept dat lijkt op code smells [24] om potentiële ontwerpproblemen in een scenario te signaleren. We definiëren een 'scenario smell' als een signaal van een mogelijk scoreprobleem in een scenario. We ontwikkelen een tool om scenario smells in een scenario te rapporteren aan een scenarioschrijver. Deze scenario smells tool kan al tijdens het ontwikkelen van scenario's worden gebruikt en vereist geen spelersgegevens om een analyse uit te kunnen voeren. De onderzoeksvragen in hoofdstuk 3 zijn:

[6] https://github.com/UURAGE/ScenarioEditor

[7] https://github.com/UURAGE/ScenarioReasoner

[8] https://www.hull-college.ac.uk/

[9] https://www.randstad.fr/

[10] https://www.youtube.com/watch?v=d_irGrEpBdc

[11] https://www.gamecomponents.eu/content/598/job-quest

[12] https://www.nurogames.com/

[13] https://bip-media.com/

- Wat zijn de uitdagingen voor een scenarioschrijver bij het ontwikkelen van een scenario?
- Hoe kunnen we een tool ontwikkelen die helpt bij het signaleren van mogelijke scoringsproblemen in een scenario?
- Wat zijn de resultaten van het gebruik van de tool op werkelijke scenario's?

We hebben scenarioschrijvers geïnterviewd om veel voorkomende uitdagingen in het schrijven van een scenario in Communicate te vinden. Scenarioschrijvers vonden het toekennen van scores een moeilijk aspect van scenario ontwikkeling in Communicate. We ontwikkelden een tool om scenario smells in een scenario te rapporteren, en evalueerden deze tool op werkelijke scenario's. Om een analogie te maken met softwareontwikkeling: de scenario editor is vergelijkbaar met een compiler, terwijl de scenario smells tool vergelijkbaar is met een Lint tool. Hoewel we de smells tool hebben ontwikkeld in de context van Communicate, denken we dat onze resultaten kunnen worden toegepast in elke leeromgeving die gebruik maakt van scenariodialogen met scores.

8.2 Pedagogische aspecten en spelmechanica

Tot nu toe hebben we gekeken naar het ontwikkelen van tools om een scenario voor een serious game te ontwikkelen en te verbeteren. Maar hoe zit het met gameplay; hoe kunnen we de interactie tussen leerlingen en een virtueel karakter verbeteren? In de eerste versie van Communicate gaat een speler een dialoog aan met een virtueel karakter door een statement te kiezen uit meerdere statement opties in een stap van een scenario. Onderzoek toont aan dat het schrijven van open tekst reacties hogere orde cognitieve vaardigheden vereist in vergelijking met het selecteren uit vooraf gedefinieerde opties [65]. De vooruitgang in Natural Language Processing (NLP) biedt mogelijkheden voor interactieve dialogen met open tekst met virtuele karakters [111]. NLP technieken worden echter nog steeds weinig gebruikt in games, omdat het vaak NLP expertise vereist van een dialoogauteur en substantiële verwerkingstijd nodig heeft [107].

Voor sommige serious game omgevingen [102, 77], verzamelden onderzoekers een grote hoeveelheid open tekst uitingen voor een specifiek scenario. Daarna ontwikkelden ze een corpus gebaseerd op deze open tekstuitingen. Vervolgens gebruikten ze het corpus om een open tekst af te handelen tijdens het spelen van het scenario. Allison et al. [2] voerden een Wizard of Oz onderzoek uit om open tekst van spelers te verzamelen voor een virtueel karakter in de entertainment game Minecraft. Allison et al. constateerden overeenkomsten in concepten die spelers gebruikten bij open tekstinvoer. Allison et al. constateerden echter ook aanzienlijke variatie in de specifieke bewoordingen van open tekst. Deze variatie in open tekst is een complicerende factor voor natuurlijke taalverwerking en zou een extra verklaring kunnen zijn voor het schaarse gebruik van NLP technieken in games.

Een scenarioschrijver is vaak een expert/docent communicatieve vaardigheden die haar expertise gebruikt bij het ontwikkelen van een scenario in Communicate. Een scenario is 'rijk' aan informatie en we verwachten dat we die informatie kunnen gebruiken om een scenario specifiek corpus te creëren, dat we kunnen gebruiken om open tekst tijdens het spelen te matchen met een gescript statement. We noemen dit de scenario specifieke corpus methode (SSCM) en in hoofdstuk 4 zijn onze onderzoeksvragen:

- Hoe kunnen we de informatie in een scenario gebruiken om een scenario specifiek corpus te ontwikkelen?

- Hoe kunnen we een open tekst matchen met gescripte statements met behulp van SSCM?

- Hoe kunnen we een golden dataset creëren om het matchen van een open tekst met een gescript statement te kunnen vergelijken?

- Hoe verhoudt het matchen met behulp van SSCM zich tot het matchen met behulp van een NLP methode met generieke corpora?

We verzamelden open tekstinvoer van studenten met behulp van een aangepaste versie van Communicate. Twee onafhankelijke experts hebben elke invoer geannoteerd en we hebben trainings-, validatie- en testdatasets gecreëerd. Deze datasets kunnen worden gebruikt om een NLP matchingsmethode te evalueren. We vergeleken de resultaten van het matchen van open tekstinvoer met behulp van SSCM versus generieke NLP methoden. We constateerden dat SSCM iets beter presteerde dan generieke NLP methoden voor gematchte tekst, maar slechter presteerde voor niet-gematchte tekst.

Een scenarioschrijver kan de scenario specifieke corpus methode (SSCM) gebruiken om de mogelijkheid om open tekst in te voeren aan te bieden tijdens het spelen van een scenario. SSCM vereist geen NLP expertise van een scenarioschrijver noch substantiële verwerkingstijd. De volgende stap is om open tekst in Communicate te introduceren en deze open tekst af te handelen. Bij de introductie van een nieuw open tekstmechanisme in Communicate hebben we gebruikt gemaakt van scaffolding [109], een term gebruikt voor ondersteuning voor een student tijdens het leren. Een vorm van scaffolding wordt ook gebruikt in entertainment games[14], waarbij een speler vaak een prompt krijgt bij een nieuwe uitdaging, en soms een hint als zij een bepaald aspect van het spel niet voltooit. We hebben SSCM gebruikt om een open tekst te matchen met de mogelijke gescripte statement opties in een stap van een scenario in Communicate. Onze onderzoeksvraag in hoofdstuk 5 is: Kunnen we met open tekst omgaan door een statement te markeren voor gematchte invoer of een hint te geven voor ongematchte invoer?

In ons experiment had deze scaffolding (markeren of hint geven) om open tekst af te handelen maar een beperkt effect. Ons experiment resulteerde in een dataset van open tekst die was gekoppeld aan een statement match (of geen match) die door een student zelf was geannoteerd. Deze dataset kan ook worden gebruikt om (een) andere NLP-methode(n) verder te verbeteren.

[14] `https://www.gamerslearn.com/home/2018/6/11/scaffolding-and-adaptation-2-6m35t-bc6ct`

8.3 Evalueren en verbeteren van pedagogische aspecten

Er zijn maar weinig methodische evaluaties van zowel zowel entertainment- als serious-games [7]. Naast onderzoeker ben ik ook docent. Ik coördineer en geef colleges in de cursus 'Software Project' in de bacheloropleiding Informatica van de Universiteit Utrecht. Een van de belangrijkste leerdoelen van deze cursus is het ontwikkelen van vaardigheden om samen te werken in een team. Communicate kan helpen bij het leren van dit soort communicatieve vaardigheden.

Vóór 2018 organiseerden we een hoorcollege over samenwerken in een klassikale sessie met alle studenten die deelnamen aan de cursus software project. In de projecten van lente-zomer 2018 zijn we begonnen met een blended learning sessie per team waarin we een workshop hebben gegeven over samenwerkingsvaardigheden met behulp van Communicate. In overleg met een docent communicatievaardigheden en een ervaren scenarioschrijver van DialogueTrainer heb ik een scenario 'samenwerken' ontwikkeld. Samen met een expert van de faculteit pedagogische wetenschappen heb ik een evaluatieformulier ontwikkeld voor een sessie. We hebben de action research method [62] toegepast gedurende drie semesters, waarbij we een interventie in een sessie hebben toegepast. Binnen de sessies in een semester hebben wij dezelfde interventie toegepast. Na elke sessie hebben studenten het evaluatieformulier ingevuld. Na elk semester hebben wij de evaluatie van de studenten geanalyseerd. Vervolgens hebben wij de interventie voor het volgende semester ontworpen. We vergeleken de feedback van studenten van de drie semesters: lente-zomer 2018, herfst-winter 2018 en lente-zomer 2019. In hoofdstuk 6 is onze onderzoeksvraag: hoe varieert de evaluatie van studenten met verschillende interventies in blended learning sessies met Communicate?

Het aantonen van de effectiviteit van educatieve interventies [41] kan de waarde van een serious game laten zien. In het eerste semester (lente-zomer 2018) gebruikten we Communicate op de originele multiple choice manier, waarbij een student reageerde op een virtueel karakter door een statement uit meerdere opties te kiezen. In het tweede semester (herfst-winter 2018) hebben we open tekst (met scaffolding) geïntroduceerd om te kunnen reageren op het virtuele karakter. In het derde semester (lente-zomer 2019) zijn we teruggegaan naar multiple choice, en introduceerden peer teaching tussen studenten.

We vergeleken de feedback van studenten van de drie semesters. Er was geen significant verschil in de beoordelingen van studenten voor de sessies van het tweede semester in vergelijking met de sessies van het eerste semester, terwijl er een significant hogere beoordeling was van de sessies van het derde semester met peer teaching. Deze resultaten suggereren dat het bij het ontwerpen en gebruiken van serious games cruciaal is om niet alleen technische verbeteringen te verzamelen, maar ook pedagogische aspecten zoals peer teaching te overwegen.

Acknowledgements

It was a mix of events that led me to start a PhD. When in 2006 I embarked on an executive MBA, after more than a decade of work experience, it was tough and challenging. A few years later I took a Sabbatical leave and backpacked through South East Asia. I liked the academic inspiration so much that a decade later after my executive MBA I wanted to do a PhD, part time next to my job just like I did during the executive MBA. This time I wanted to go back to my computer science and engineering roots. I applied and met Prof. Johan Jeuring, who had four PhD positions to fill in computer science at Utrecht University. There was an immediate personal click, however the idea of a part time PhD candidate did not land in the department. A couple of months later, my employer then was offering a voluntary redundancy scheme. I thought of taking another travelling Sabbatical and applied for the scheme. A week later I wrote to Johan, saying that I could do a PhD full time. I was then thinking of travelling for a year and thereafter embarking on a PhD. As luck would have it, one of the four PhD positions was still open and after an interview, Johan inquired if I would be interested in starting from 1st September 2015. Since I needed to hand over work at my employer and the vision of backpacking for a year was just fresh in my mind, we decided on a few months later. This time I went backpacking in Guatemala and thereafter started 1st of Nov 2015 as a PhD candidate at Utrecht University.

I am thankful to you Johan for giving me the chance to pursue a PhD. Your humility in discussions and offering suggestions is admirable, there was never really much of a power distance. During the course of these past years, you were promoted to the head of department role. While you were rather busy, you always managed to find time to discuss and review my work.

Thank you Jordy van Dortmont and Marcell van Geest, for showing me the intricacies of modern day programming. Getting back to technical stuff, after so many years of working in non technical roles, was extremely difficult for me. It was also rather ‘gezellig’, especially during the trips to Edinburgh and Bucharest.

I thank my colleagues and friends for the support during my doctoral adventure. In no particular order: Alejandro, Aniek, Aabhas, Anna-Lena, Arjen, Bastiaan, Casper, Claudia, Erwan, Gabi, Gerard, Hieke, Ignaz, Isaac, Ivica, Kimberley, Jeroen, Jet, Josje, Jurriaan, João, Lennart, Lily, Marie-José, Marjan, Marcel, Matthieu, Michiel, Nico, Nicole, Pavel, Qixiang, Rashmi, Rick, Robert, Sarah, Sergey, Timo, Victor, Vedran, Wouter, Wishnu, and all others who found time to engage in stimulating discussions. A special word of thanks to Susan, for regularly making time to keep me company this past pandemic year.

I end this acknowledgements by thanking my family who have always been there for me. To my parents, you have raised me well, instilling curiosity, discipline, respect and integrity. ‘Debating’ with my sister in the teenage years has perhaps taught me to better formulate probing questions. My family’s support in any path I choose in life is a source of comfort to me.

Curriculum Vitae

Raja Lala, geboren te Pune op 1 maart 1973

1990 – 1994
Computer science and engineering, Delhi institute of technology

1994 – 1996
IT engineer, CMC ltd. India

1996 – 2000
Senior technisch ontwerper, ICT Automatisering

2000 – 2005
Managing consultant, Capgemini Ernst & Young

2005 – 2007
Manager Systems and Process Assurance, PriceWaterhouseCoopers

Jan 2006
Passed CISA (Certified Information Systems Auditor) exam

2006 – 2008
Master of business administration (MBA), Rotterdam school of management

2007 – 2015
Business development executive, IBM

2015 – 2020
Promovendus, Utrecht Universiteit

2020 – date
Docent Informatica, Universiteit Utrecht

Bibliography

[1] Vincent Aleven, Jonathan Sewall, Bruce M McLaren, and Kenneth R Koedinger. Rapid authoring of intelligent tutors for real-world and experimental use. In *Sixth IEEE International Conference on Advanced Learning Technologies (ICALT'06)*, pages 847–851. IEEE, 2006.

[2] Fraser Allison, Ewa Luger, and Katja Hofmann. How players speak to an intelligent game character using natural language messages. *Transactions of the Digital Games Research Association*, 4(2), 2018.

[3] Diogo Almeida, José Creissac Campos, João Saraiva, and João Carlos Silva. Towards a catalog of usability smells. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 175–181. ACM, 2015.

[4] Cristina Alonso-Fernández, Iván Martínez-Ortiz, Rafael Caballero, Manuel Freire, and Baltasar Fernández-Manjón. Predicting students' knowledge after playing a serious game based on learning analytics data: A case study. *Journal of Computer Assisted Learning*, 36(3):350–358, 2020.

[5] Sylvester Arnab, Theodore Lim, Maira B Carvalho, Francesco Bellotti, Sara De Freitas, Sandy Louchart, Neil Suttie, Riccardo Berta, and Alessandro De Gloria. Mapping learning and game mechanics for serious games analysis. *British Journal of Educational Technology*, 46(2):391–411, 2015.

[6] Walter F Baile, Robert Buckman, Renato Lenzi, Gary Glober, Estela A Beale, and Andrzej P Kudelka. Spikes—a six-step protocol for delivering bad news: application to the patient with cancer. *The oncologist*, 5(4):302–311, 2000.

[7] Francesco Bellotti, Bill Kapralos, Kiju Lee, Pablo Moreno-Ger, and Riccardo Berta. Assessment in and of serious games: an overview. *Advances in human-computer interaction*, 2013, 2013.

[8] Marianne Berkhof, H. Jolanda van Rijssen, Antonius J.M. Schellart, Johannes R. Anema, and Allard J. van der Beek. Effective training strategies for teaching communication skills to physicians: An overview of systematic reviews. *Patient Education and Counseling*, 84(2):152–162, 2011.

[9] Karel Bosch, Arjen Brandenburgh, Tijmen Joppe Muller, and Annerieke Heuvelink. Characters with personality! In *Proceedings IVA 2012: the 12th International Conference on Intelligent Virtual Agents*, volume 7502 of *LNAI*, pages 426–439. Springer, 2012.

[10] Kim Bosman, Tibor Bosse, and Daniel Formolo. Virtual agents for professional social skills training: An overview of the state-of-the-art. In *10th International Conference on Intelligent Technologies for Interactive Entertainment, INTETAIN 2018*, pages 75–84. Springer Verlag, 2019.

[11] Tibor Bosse and Simon Provoost. Integrating conversation trees and cognitive models within an eca for aggression de-escalation training. In *Proceedings PRIMA 2015: the 18th International Conference on Principles and Practice of Multi-Agent Systems*, volume 9387 of *LNCS*, pages 650–659. Springer, 2015.

[12] Luke Bracegirdle and Stephen Chapman. Programmable patients: Simulation of consultation skills in a virtual environment. *Bio-algorithms & Med-Systems*, 6:111–115, 2010.

[13] Alexander Budanitsky and Graeme Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational linguistics*, 32:13–47, 2006.

[14] Osmar A Carvalho Júnior, Renato F Guimarães, Alan R Gillespie, Nilton C Silva, and Roberto AT Gomes. A new approach to change vector analysis using distance and similarity measures. *Remote Sensing*, 3(11):2473–2493, 2011.

[15] Ana Paula Cláudio, Maria Beatriz Carmo, Vítor Pinto, and Afonso Cavaco. Virtual humans for training and assessment of self-medication consultation skills in pharmacy students. In *Proceedings ICCSE 2015: the 10th International Conference on Computer Science & Education*, pages 175–180. IEEE, 2015.

[16] Thomas M Connolly, Elizabeth A Boyle, Ewan MacArthur, Thomas Hainey, and James M Boyle. A systematic literature review of empirical evidence on computer games and serious games. *Computers & education*, 59(2):661–686, 2012.

[17] David A Cook, Patricia J Erwin, and Marc M Triola. Computerized virtual patients in health professions education: a systematic review and meta-analysis. *Academic Medicine*, 85(10):1589–1602, 2010.

[18] Gaoying Cui, Qin Lu, Wenjie Li, and Yi-Rong Chen. Corpus exploitation from wikipedia for ontology construction. In *LREC*, 2008.

[19] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[20] Mahad Khalif Dhaqane and Nor Abdulle Afrah. Satisfaction of students and academic performance in benadir university. *Journal of Education and Practice*, 7(24):59–63, 2016.

[21] Joao Dias, Samuel Mascarenhas, and Ana Paiva. Fatima modular: Towards an agent architecture with a generic appraisal framework. In *Emotion Modeling: Towards Pragmatic Computational Models of Affective Processes*, volume 8750 of *LNCS*, pages 44–56. Springer, 2014.

[22] Damien Djaouti, Julian Alvarez, Jean-Pierre Jessel, and Olivier Rampnoux. Origins of serious games. In *Serious games and edutainment applications*, pages 25–43. Springer, 2011.

[23] Henrik Engström, Jenny Brusk, and Patrik Erlandsson. Prototyping tools for game writers. *The Computer Games Journal*, 7(3):153–172, 2018.

[24] Martin Fowler and Kent Beck. *Refactoring: improving the design of existing code.* Addison-Wesley Professional, 1999.

[25] Robert L Fowler and Anne S Barker. Effectiveness of highlighting for retention of text material. *Journal of Applied Psychology*, 59(3):358, 1974.

[26] Jane Friedhoff. Untangling twine: A platform study. In *DiGRA conference*, 2013.

[27] Roger Garside, Geoffrey N Leech, and Tony McEnery. *Corpus annotation: linguistic information from computer text corpora.* Taylor & Francis, 1997.

[28] Patrick Gebhard, Gregor Mehlmann, and Michael Kipp. Visual scenemaker—a tool for authoring interactive virtual characters. *Journal on Multimodal User Interfaces*, 6(1):3–11, 2011.

[29] Barbara Goldschmid and Marcel L Goldschmid. Peer teaching in higher education: A review. *Higher Education*, 5(1):9–33, 1976.

[30] Arthur C Graesser, Shulan Lu, George Tanner Jackson, Heather Hite Mitchell, Mathew Ventura, Andrew Olney, and Max M Louwerse. Autotutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36(2):180–192, 2004.

[31] Carol Gray and Jenny Moffett. *Handbook of Veterinary Communication skills.* Wiley-Blackwell, 2010.

[32] Jing Guo, Nicolas Singer, and Rémi Bastide. Design of a serious game in training non-clinical skills for professionals in health care area. In *Proceeding SEGAH 2014: IEEE International Conference on Serious Games and Applications for Health*, pages 1–6. IEEE, 2014.

[33] Anushree Gupta. Quality of assessment of sequences of choices in the serious game communicate. Master's thesis, Utrecht University, 2020.

[34] William L Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. Inducing domain-specific sentiment lexicons from unlabeled corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2016, page 595. NIH Public Access, 2016.

[35] Hugo Hammer, Anis Yazidi, Aleksander Bai, and Paal Engelstad. Building domain specific sentiment lexicons combining information from many sentiment lexicons and a domain specific corpus. In *IFIP International Conference on Computer Science and its Applications*, pages 205–216. Springer, 2015.

[36] Owen Hargie. *The handbook of communication skills.* Psychology Press, 1997.

[37] Jannicke Baalsrud Hauge, Stefan Wiesner, Rosa Garcia Sanchez, Poul Kyvsgaard Hansen, Giusy Fiucci, Michel Rudnianski, and Jon Arambarri Basanez. Business models for serious games developers-transition from a product centric to a service centric approach. *International Journal of Serious Games*, 1(1), 2014.

[38] F.P.M. Heemskerk. Processing open text input in a scripted communication scenario. Master's thesis, Utrecht University, 2019.

[39] Brad Hennigan. Making the case for nlp in dialogue systems for serious games. In *8th international conference on natural language processing (JapTAL), 1st workshop on games and NLP*, 2012.

[40] Ryuichiro Higashinaka, Masahiro Mizukami, Hidetoshi Kawabata, Emi Yamaguchi, Noritake Adachi, and Junji Tomita. Role play-based question-answering by real users for building chatbots with consistent personalities. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 264–272, 2018.

[41] Paul Hollins, Wim Westera, and Borja Manero Iglesias. Amplifying applied game development and uptake. In *European Conference on Games Based Learning*, page 234. Academic Conferences International Limited, 2015.

[42] Radu Tudor Ionescu, Marius Popescu, and Aoife Cahill. Can characters reveal your native language? a language-independent approach to native language identification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1363–1373, 2014.

[43] Johan Jeuring, Frans Grosfeld, Bastiaan Heeren, Michiel Hulsbergen, Richta IJntema, Vincent Jonker, Nicole Mastenbroek, Maarten van der Smagt, Frank Wijmans, Majanne Wolters, and Henk van Zeijts. Communicate! — a serious game for communication skills —. In *Proceedings EC-TEL 2015: Design for Teaching*

and Learning in a Networked World: 10th European Conference on Technology Enhanced Learning, volume 9307 of *LNCS*, pages 513–517. Springer, 2015.

[44] W. Lewis Johnson and James C. Lester. Face-to-face interaction with pedagogical agents, twenty years later. *International Journal of Artificial Intelligence in Education*, 26(1):25–36, 2016.

[45] Lindsey Joyce. Creating collaborative criteria for agency in interactive narrative game analysis. *The Computer Games Journal*, 4(1-2):47–58, 2015.

[46] Michael T Kane. An argument-based approach to validity. *Psychological bulletin*, 112(3):527, 1992.

[47] Victor Kaptelinin and Bonnie A Nardi. *Acting with technology: Activity theory and interaction design.* MIT press, 2006.

[48] Sylvia Kasperink. Assessing validity of the serious game communicate: An argument-based approach to validation. Master's thesis, Utrecht University, 2017.

[49] Ralph H. Kilmann and Kenneth W. Thomas. Interpersonal conflict-handling behavior. *Psychological Reports*, 37(3):971–980, 1975.

[50] Hartmut Koenitz. Interactive storytelling paradigms and representations: a humanities-based perspective. *Handbook of Digital Games and Entertainment Technologies*, pages 1–15, 2016.

[51] David R Krathwohl and Lorin W Anderson. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Longman, 2009.

[52] Raja Lala, Gemma Corbalan, and Johan Jeuring. Evaluation of interventions in blended learning using a communication skills serious game. In Antonios Liapis, Georgios N. Yannakakis, Manuel Gentile, and Manuel Ninaus, editors, *Games and Learning Alliance*, pages 322–331, Cham, 2019. Springer International Publishing.

[53] Raja Lala, Johan Jeuring, FPM Heemskerk, Marcell Van Geest, Jordy Van Dortmont, Gabriel Gutu, Stefan Ruseti, Mihai Dascalu, Beatrice Alex, and Richard Tobin. Processing open text input in a scripted communication scenario. In *SEMDIAL 2018: The 22nd workshop on the Semantics and Pragmatics of Dialogue*, pages 211–214, 2018.

[54] Raja Lala, Johan Jeuring, and Timo Overbeek. Analysing and adapting communication scenarios in virtual learning environments for one-to-one communication skills training. In *iLRN Immersive learning research network conference*, 2017.

[55] Raja Lala, Johan Jeuring, Jordy van Dortmont, and Marcell van Geest. Scenarios in virtual learning environments for one-to-one communication skills training. *International Journal of Educational Technology in Higher Education*, 14(1):17, May 2017.

[56] Raja Lala, Johan Jeuring, and Marcell van Geest. Scaffolding open text input in a scripted communication skills learning environment. In *International Conference on Games and Learning Alliance*, pages 169–179. Springer, 2019.

[57] Raja Lala, Marcell van Geest, Stefan Ruseti, Johan Jeuring, Mihai Dascalu, Jordy van Dortmont, Gabriel Gutu-Robu, and Michiel Hulsbergen. Enhancing free-text interactions in a communication skills learning environment. In *Proceedings of the 13th international conference on Computer supported collaborative learning*, June, 2019.

[58] Lotta C Larson and Teresa Northern Miller. 21st century skills: Prepare students for the future. *Kappa Delta Pi Record*, 47(3):121–123, 2011.

[59] Jonathan Lessard. Designing natural-language game conversations. *Proc. DiGRA-FDG*, 2016.

[60] Jonathan Lessard, Etienne Brunelle-Leclerc, Timothy Gottschalk, Marc-Antoine Jetté-Léger, Odile Prouveur, and Christopher Tan. Striving for author-friendly procedural dialogue generation. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, page 67. ACM, 2017.

[61] Anton Leuski and David Traum. NPCEditor: Creating Virtual Human Dialogue Using Information Retrieval Techniques. *AI Magazine*, 32(2):42–56, July 2011.

[62] Kurt Lewin. Action research and minority problems. *Journal of social issues*, 2(4):34–46, 1946.

[63] Davide Marocco, Daniela Pacella, Elena Dell'Aquila, and Andrea Di Ferdinando. Grounding serious game design on scientific findings: The case of enact on soft skills training and assessment. In *Proceedings EC-TEL 2015 Design for Teaching and Learning in a Networked World: 10th European Conference on Technology Enhanced Learning*, volume 9307 of *LNCS*, pages 441–446. Springer International Publishing, 2015.

[64] Ati Suci Dian Martha and Harry B Santoso. The design and impact of the pedagogical agent: A systematic literature review. *Journal of educators Online*, 16(1):n1, 2019.

[65] Michael E Martinez. Cognition and the question of test item format. *Educational Psychologist*, 34(4):207–218, 1999.

[66] Michael Mateas and Andrew Stern. Natural language understanding in façade: Surface-text processing. In *International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, pages 3–13. Springer, 2004.

[67] R. Mazza, L. Ambrosini, N. Catenazzi, S. Vanini, D. Tuggener, and G. Tavarnesi. Behavioural simulator for professional training based on natural language interaction. In *EDULEARN18 Proceedings*, 10th International Conference on Education and New Learning Technologies, pages 3204–3214. IATED, 2-4 July, 2018 2018.

[68] Paul McCoubrie. Improving the fairness of multiple-choice questions: a literature review. *Medical teacher*, 26(8):709–712, 2004.

[69] Rada Mihalcea, Courtney Corley, Carlo Strapparava, et al. Corpus-based and knowledge-based measures of text semantic similarity. In *Aaai*, volume 6, pages 775–780, 2006.

[70] Wookhee Min, Kyungjin Park, Joseph Wiggins, Bradford Mott, Eric Wiebe, Kristy Elizabeth Boyer, and James Lester. Predicting dialogue breakdown in conversational pedagogical agents with multimodal lstms. In *Proceedings of the Dialog System Technology Challenges Workshop (DSTC6)*, 2017.

[71] Masahiro Mori, Karl F MacDorman, and Norri Kageki. The uncanny valley [from the field]. *IEEE Robotics & Automation Magazine*, 19(2):98–100, 2012.

[72] Tijmen Joppe Muller, Annerieke Heuvelink, Karel van den Bosch, and Ivo Swartjes. Glengarry glen ross: Using bdi for sales game dialogues. In *Proceedings AIIDE 2012: 8th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 167–172. AAAI Press, 2012.

[73] Tom Murray. An overview of intelligent tutoring system authoring tools: Updated analysis of the state of the art. In *Authoring tools for advanced technology learning environments*, pages 491–544. Springer, 2003.

[74] Nienke Nieveen and Elvira Folmer. Formative evaluation in educational design research. *Design Research*, 153:152–169, 2013.

[75] Hyacinth S Nwana. Intelligent tutoring systems: an overview. *Artificial Intelligence Review*, 4(4):251–277, 1990.

[76] Magalie Ochs, Philippe Blache, and Grégoire De Montcheuil. What common ground between a human and a virtual agent? the case of task-oriented dialogues for breaking bad news. In *22nd Workshop on the Semantics and Pragmatics of Dialogue (SemDial 2018)*, 2018.

[77] Magalie Ochs, Daniel Mestre, Grégoire De Montcheuil, Jean-Marie Pergandi, Jorane Saubesty, Evelyne Lombardo, Daniel Francon, and Philippe Blache. Training

doctors' social skills to break bad news: evaluation of the impact of virtual environment displays on the sense of presence. *Journal on Multimodal User Interfaces*, 13(1):41–51, 2019.

[78] Timo Overbeek. Using scenario smells to analyze scripted communication scenarios in virtual learning environments. Master's thesis, Utrecht University, 2019.

[79] Timo Overbeek, Raja Lala, and Johan Jeuring. Using scenario smells to analyse scripted communication scenarios in virtual learning environments. Technical report, Department of Information and Computing Sciences, Utrecht University, 2017.

[80] Timo Overbeek, Raja Lala, and Johan Jeuring. Scenario smells: signalling potential problems in dialogue scenarios in a serious game. *International Journal of Serious Games*, 7(4):51–73, 2020.

[81] Yasuhiro Ozuru, Stephen Briner, Christopher A Kurby, and Danielle S McNamara. Comparing comprehension measured by multiple-choice and open-ended questions. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 67(3):215, 2013.

[82] David Parlett. *The Oxford history of board games.* Oxford University Press, USA, 1999.

[83] Ruth Pel-Littel, Henk van Zeijts, Nadine Schram, Henk Herman Nap, and Johan Jeuring. A training simulation for practicing shared decision making for older patients. *Procedia Computer Science*, 141:287–293, 2018.

[84] Rosalind W Picard and Roalind Picard. *Affective computing*, volume 252. MIT press Cambridge, 1997.

[85] Davide Picca, Dominique Jaccard, and Gérald Eberlé. Natural language processing in serious games: a state of the art. *International Journal of Serious Games*, 2(3):77–97, 2015.

[86] Jefferson Pooley. The new history of mass communication research. In *The History of Media and Communication Research.* Peter Lang, 2008.

[87] Michael E Porter. Competitive strategy. *Measuring business excellence*, 1997.

[88] Marten Postma, Emiel van Miltenburg, Roxane Segers, Anneleen Schoen, and Piek Vosse. Open dutch wordnet. In *Proceedings of the Eight Global Wordnet Conference*, Bucharest, Romania, 2016.

[89] Simon Provoost, Ho Ming Lau, Jeroen Ruwaard, and Heleen Riper. Embodied conversational agents in clinical psychology: a scoping review. *Journal of medical Internet research*, 19(5):e151, 2017.

[90] M Afzalur Rahim. A measure of styles of handling interpersonal conflict. *Academy of Management journal*, 26(2):368–376, 1983.

[91] Olivia Realdon, Valentino Zurloni, Linda Confalonieri, Marcello Mortillaro, and Fabrizia Mantovani. Learning communication skills through computer-based interactive simulations. In *From communication to presence: Cognition, emotions and culture towards the ultimate communicative experience*, pages 281–303. IOS Press, Amsterdam, 2006.

[92] Steffen Remus and Chris Biemann. Domain-specific corpus expansion with focused webcrawling. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3607–3611, 2016.

[93] Jean-Hugues Réty, Nicolas Szilas, Jean Clément, and Serge Bouchardon. Authoring interactive narratives with hypersections. In *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, pages 393–400. ACM, 2008.

[94] Ellen Riloff and Jessica Shepherd. A corpus-based approach for building semantic lexicons. *ArXiv*, cmp-lg/9706013, 1997.

[95] Stefan Ruseti, Raja Lala, Gabriel Gutu-Robu, Mihai Dascalu, Johan Jeuring, and Marcell van Geest. Semantic matching of open texts to pre-scripted answers in dialogue-based learning. In *Proceedings of the 20th International Conference on Artificial Intelligence in Education*, June, 2019.

[96] Koen Samyn, Gaetan Deglorie, Peter Lambert, Rik Van de Walle, and Sofie Van Hoecke. Generating non-linear narrative for serious games with scenario templates. In *Proceedings of the 10th international Conference on computer graphics theory and applications*, pages 523–530. SCITEPRESS-Science and Technology Publications, Lda, 2015.

[97] Valerie J Shute, Matthew Ventura, Malcolm Bauer, and Diego Zapata-Rivera. Melding the power of serious games and embedded assessment to monitor and foster learning. *Serious games: Mechanisms and effects*, 2:295–321, 2009.

[98] Jonathan Silverman, Suzanne Kurtz, and Juliet Draper. *Skills For Communicating With Patients, Third Edition.* Radcliffe Publishing Limited, 2013.

[99] Sergey Sosnovsky and Peter Brusilovsky. Evaluation of topic-based adaptation and student modeling in quizguide. *User Modeling and User-Adapted Interaction*, 25(4):371–424, 2015.

[100] Ioana Andreea Stănescu, Harald Jan Gemt Warmelink, Julia Lo, Sylvester Arnab, Francesca Dagnino, and Jane Mooney. Accessibility, reusability and interoperability in the European serious game community. *eLearning & Software for Education*, 2(2), 2013.

[101] James Stewart, Lizzy Bleumers, Jan Van Looy, Ilse Mariën, Anissa All, Dana Schurmans, Koen Willaert, Frederik De Grove, An Jacobs, and Gianluca Misuraca. The potential of digital games for empowerment and social inclusion of groups at risk of social and economic exclusion: evidence and opportunity for policy. *Joint Research Centre, European Commission*, 2013.

[102] Laura M van der Lubbe, Charlotte Gerritsen, Daniel Formolo, Marco Otte, and Tibor Bosse. A serious game for training verbal resilience to doorstep scams. In *International Conference on Games and Learning Alliance*, pages 110–120. Springer, 2018.

[103] Kurt Vanlehn. The behavior of tutoring systems. *International journal of artificial intelligence in education*, 16(3):227–265, 2006.

[104] Jennifer J. Vogel, David S. Vogel, Jan Cannon-Bowers, Clint A. Bowers, Kathryn Muse, and Michelle Wright. Computer gaming and interactive simulations for learning: A meta-analysis. *Journal of Educational Computing Research*, 34(3):229–243, 2006.

[105] Jeroen Wauters, Frederik Broeckhoven, Maarten Overveldt, Koen Eneman, Frederik Vaassen, and Walter Daelemans. delearyous: An interactive application for interpersonal communication training. In *Serious Games: The Challenge: Joint Conference of the Interdisciplinary Research Group on Technology, Education, and Communication, and the Scientific Network on Critical and Flexible Thinking Ghent*, volume 280 of *Communications in Computer and Information Science*, pages 87–90. Springer, 2012.

[106] Wim Westera. Games are motivating, aren't they? Disputing the arguments for digital game-based learning. *International Journal of Serious Games*, 2(2):3–17, 2015.

[107] Wim Westera, Rui Prada, Samuel Mascarenhas, Pedro A Santos, João Dias, Manuel Guimarães, Konstantinos Georgiadis, Enkhbold Nyamsuren, Kiavash Bahreini, Zerrin Yumak, et al. Artificial intelligence moving serious gaming: Presenting reusable game AI components. *Education and Information Technologies*, 25(1):351–380, 2020.

[108] Phil Wilkinson. A brief history of serious games. In *Entertainment computing and serious games*, pages 17–41. Springer, 2016.

[109] David Wood, Jerome S Bruner, and Gail Ross. The role of tutoring in problem solving. *Journal of child psychology and psychiatry*, 17(2):89–100, 1976.

[110] Pieter Wouters, Christof Van Nimwegen, Herre Van Oostendorp, and Erik D Van Der Spek. A meta-analysis of the cognitive and motivational effects of serious games. *Journal of Educational Psychology*, 105(2):249–265, 2013.

[111] Georgios N Yannakakis and Julian Togelius. *Artificial intelligence and games*, volume 2. Springer, 2018.

[112] Jun Ye. Cosine similarity measures for intuitionistic fuzzy sets and their applications. *Mathematical and Computer Modelling*, 53(1-2):91–97, 2011.

[113] Amel Yessad, Thibault Carron, and Jean-Marc Labat. An approach to model and validate scenarios of serious games in the design stage. In *International Conference on Web-Based Learning*, pages 264–273. Springer, 2013.