Improving Wizard of Oz creation by applying game concepts

Master's Thesis Information Sciences
By
Dave Nabuurs

Radboud University Nijmegen
Faculty of Science
Institute of Computing and Information Sciences

June 28, 2011

Supervisor: dr. S.J.B.A. Hoppenbrouwers
Reviewer: dr. P. van Bommel
Thesis Number: 148 IK

Acknowledgment

In 2005 I started with my study at the Fontys Hogeschool Eindhoven where I studied Computer Science. After this study was completed, I did not feel I was ready to work and I wanted to take a different direction in the ICT. I chose for this study Information Science and I am very happy about this. Now I think that I have a good basis for the ICT field and that a good moment has come to finish my study career. The closure of this career is done by this Master thesis. Here I have been working hard and with a lot of fun and I think the results are good.

I want to thank two teachers who have helped me so much for these years of education and to bring here a successful end. Especially Mr. Hoppenbrouwers, who not only gave good input for this research, but has given me the trust and has the patience. He is also very compassionate and helpful during his course research methods. I would also like to thank Mr. van der Weide. He has ensured that my study could be rounded off smoothly. Finally, I want to thank my family who supported in difficult times and dragged me through difficult situations. Especially my girlfriend whom I often discussed my ideas with and explained my problems too. And my parents who have helped me in the entire process of my education.

Dave Nabuurs
June 28, 2011

Contents

Chapter 1

Introduction

The field of Human Computer Interaction (HCI) concerns the study of interaction between people and systems. One method commonly used to investigate this is the Wizard of Oz (WOz method). The WOz method can help detect in an early stage of development whether the functionality and GUI are what the customer is looking for. When the requirements are set, a prototype without the (complex) functionalities can be developed. The WOz method is a technique in which the user (the "participant") tests the system with the illusion that he test the whole system (including functionality). In reality, he sits behind an interface and send all actions to another person (the "wizard"). The wizard responds to this action and it seems for the participant he interacts with the system.

This provides the advantage that in an early stage, without too much development having taken place, can be discovered, whether the requirement engineers have gather the right info of the customer and whether the customer is satisfied with the result. But this method also has its disadvantages, for example, that the wizard is cognitively heavily loaded in order to simulate the system.

This research attempts to improve the WOz method by using Game Design Theory (GDT). The reason is that I see strong similarities between this method and a game. Therefore I would like to view the WOz method as a game. To see this method as a game, insights may arise that can greatly improve this method.

1.1 Research questions

This has lead me to the research question:

Can the WOz method be improved by applying GDT?

Because this research question is too big to resolve at once, I divided it into multiple sub-questions.

1. *Can GDT ensure that the WOz method is more generic, so that this method becomes easier to apply on an arbitrary IT project?*
2. *Can GDT contribute to a more structured approach of the WOz method?*
3. *Can GDT eliminate or reduce the disadvantages associated with the method?*

1.2 Project plan

In this section I describe how I will get to a reshaped WOz method. The figure below gives a schematic overview of the steps I will take to customize and improve the current WOz method.

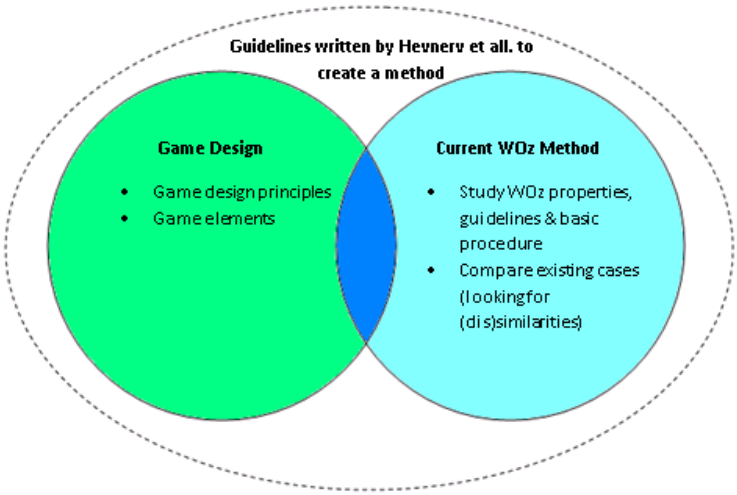


Figure 1.1: *Schematic overview of the three disciplines that help create the new WOz method.*

1.2.1 Design science guidelines

Hevner [13] wrote an article with guidelines. These seven guidelines describe how to create an artifact (e.g. an information system or method) via a scientific way. These guidelines increase the chance to get a valid and correct artifact. Hevner combines two distinct paradigms. The first one is behavioral science, which uses scientific research methods. It is used to develop and explain theories (like laws and principles). The other paradigm is the design science. In this paradigm science and artifact engineering are central. It can be seen as a problem solving paradigm and it helps by creating innovations and using creativity. In contrast to behavioral science, it does not use any laws or behavioral theorems, but it uses especially kernel theories, which would be tested, modified and above all extended by creativity of the researcher [20, 28].

By combining these two distinctive paradigms, they can help with designing and evaluating the artifacts with a good result. The goal of behavioral science is to falsify and validate. The goal of design science is to make something useful. In other words, design science will help us to create an artifact with innovation and creativity, while behavioral science helps us with a systematic approach by designing and evaluating.

1.2.2 Game design theory

The WOz method and other system activities cannot be captured in so-called "user manuals", or in other words an imperative language. This must be done with a declarative language, for example a game. This is the initial reason why people have started using the game metaphor to "catch" a system or method. For example, Järvinnen [16] wrote in one of his articles that an information system can be seen as a game and vice versa. When using the game metaphor for an (information) system, certain similarities and dissimilarities can be found. This is helpful by testing this system with the WOz method

In this chapter, two aspects are discussed. First, game elements will be discussed. These can be used to structure the WOz method. Also, existing WOz cases would be decomposed to understand how these cases are constructed. The other part will cover the game principles. These principles are the base for a good game. There could be a better result when these principles are used for design of the new WOz method driven by GDT. GDT and the game principles will be supported by the design science. In this chapter, no approaches to develop a game are discussed. Firstly, because there are very many different approaches to design a game and secondly, every designer is different and every company has their own procedures, so generalization is impossible at this point [7].

1.2.3 Wizard of Oz method

I started with studying the current WOz method. In this study I found the procedure that gives an impression of how to apply the WOz method. This procedure shows the basics of the method. Of course, there are a lot of variations of these method, but every project that uses the WOz method, has a couple of things in common. This procedure will be explained in chapter 4.

Not only will the procedure be examined. Also, properties and guidelines of the WOz method will be examined. These characteristics should give a clear picture of the WOz. In a later stadium, these properties can be used for defining characteristics of game design to apply. The next section will also clarify which properties these are. After these properties are clear, there two existing WOz cases will be compared. So the similarities and dissimilarities will be analyzed. Also a game will be decomposed. This can be found in chapter 4. A process model and a data model can help here. By determine the similarities and dissimilarities, it will give me a better view on generalizations and subsets that can be identified and might be useful for customizing the current WOz method.

Chapter 2

Design guidelines

There are many articles available that describe how to design a new method or modifying a existing method. I have chosen for an article written by Hevner [13]. The goal of the article is to help design or evaluate an artifact (particularly information systems) via a couple of guidelines. This fits well in my research, in which the current WOz method will be modified by GDT. In the article two different paradigms come together. Namely, the behavioral science paradigm and the design science paradigm.

As described in the article of Hevner, March and Smith [19] distinguish between two design processes and four design artifacts. The processes can be divided into the design and evaluation of an artifact. When designing, a number of activities must be walked trough to come to an artifact. In evaluating this, the problem will become clearer and the quality of the artifact can be improved. These two steps will be iterated until the desired result is obtained. Below you can see a schematic representation of this [26] .

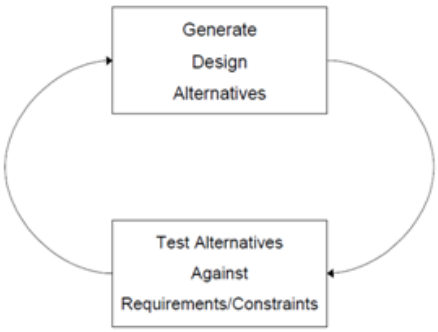


Figure 2.1: *Design process described by March and Smith* [19].

The artifacts from design science research that can occur are: constructs, models, methods and instantiations. Constructs can be seen as the language of the problem and the solution and is defined to communicate. Models are the graphic representation of this and provide a better adhesion. The methods ensure that structural problems can be solved. Instantiations ensure that the constructs, models and methods can be implemented in working system. In this way the feasibility can be reviewed. In addition, it shows whether the artifact has a contribution to the solution of the problem.

To achieve a good artifact, Hevner outlined seven guidelines in his article. If people work conform these guidelines, it's likely to reach a good artifact. In the sections below, these guidelines are summarized.

2.1 Guideline 1: Design as an artifact

Guideline 1 says that it is important that an artifact is created that is purposeful. It should contribute to solving the problem. This seems at first sight an unnecessary guideline, because it sounds logical. Sometimes this can overlooked in complex situations and/or environments. And people designs haphazardly made artifacts that already exist or simple solutions where available. It's important that the artifact is effective and can be applied easily to the domain.

2.2 Guideline 2: Problem relevance

Guideline 2 implies that there should be a real problem. If there is no problem, it makes no sense to design an artifact. Again, this will complement perfectly the design science and behavioral science together. Thus the behavior science tries to achieve this goal by theories explaining or predicting phenomena. Design science attempts to achieve this goal by building an artifact that are meant to solve this problem. As Simon argued and Hevner beautifully expressed: "Formally, a problem can be defined as the differences between a goal state and the current state of a system." [13, 26]. This definition can help to see if there really is a problem. And if there is a problem, is the solution to build a new artifact?

2.3 Guideline 3: Design evaluation

An important part of the process of designing an artifact, is to evaluate it. This review ensures that the artifact can be improved in a next iteration. The artifact is only proper if the requirements achieved and constraints are reduced or eliminated. The behavioral science is properly a major contributor in this. Thus the artifact must be in a proper way be evaluated, with the right tools, proper information gathering and correct analysis. The artifact can be evaluated in many forms, such Hevner lists "functionality, completeness, consistency,

accuracy, performance, reliability, usability, fit with the organization, and other relevant quality attributes." [13]. There are also many methods to evaluate an artifact. The table below shows the types of evaluation methods with a brief summary. For each artifact should be explored for the best evaluation method. These evaluation methods are listed below.

1. Observational	Case study: study artifact in depth in business environment
	Field study: monitor use of artifact in multiple projects
2. Analytical	Static analysis: examine structure of artifact for static qualities
	Architecture analysis: study fit of artifact into technical IS architecture
	Optimization: demonstrate inherent optimal properties of artifact or provide optimality bounds on artifact behavior
	Dynamic analysis: study artifact in use for dynamic qualities (e.g. performance)
3. Experimental	Controlled experiment: study artifact in controlled environment for qualities (e.g. usability)
	Simulation: execute artifact with artificial data
4. Testing	Functional (black box) testing: execute artifact interfaces to discover failures and identity defects
	Structural (white box) testing: Perform coverage testing of some metic (e.g. execution paths) in the artifact implementation
5. Descriptive	Informed argument: use information from the knowledge base (e.g. relevant
	research) to research to build a convincing argument for the artifact's utility
	Scenarios: construct detailed scenarios around the artifact to demonstrate its utility

Table 2.1: *Design evaluation methods by Hevner [13].*

2.4 Guideline 4: Research contributions

This guideline describes the importance of designing an artifact with a number of conditions. It should contribute in the field of the designed artifact, the way it's built, or the way how the artifact is evaluated. At least one of these three points must be in the form of knowledge to contribute to a good way to get an artifact from the design science. Mostly, the knowledge is acquired from the artifact and gives more insight in the domain where it takes place. It's not necessarily to get new knowledge. It can also sharpen or broaden existing knowledge.

2.5 Guideline 5: Research rigor

According this guideline it's important to design and evaluate the artifact in a rigorous way. This is partly derived from behavior science in which collecting data and analyzing them happens by rigorous methods. The other part comes from the design science and is often based on mathematical formalisms. The artifact that will be designed is often a HCI problem. Here behavior theories and empirical research play an important role to design and evaluate the artifact. But the ultimate goal is to determine whether the artifact is working properly. Theorizing or evidence on why the system works is something that behavioral experts or scientists can research after the artifact works.

2.6 Guideline 6: Design as a search process

Making an artifact, should be done as an iterative process. The best way to do this, is as a heuristic search process to find a solution. This process is often untraceable. This has mainly to do with creativity because you come to a solution and this is not a method or road map. Effective design requires knowledge of the application domain (e.g. requirements and constraints) and solution domain (e.g. technical and organizational). Often the problem that must be solved can be divided into sub problems. Mainly at the initial stage can this decomposition to obtain guidance. Later, this approach will not satisfy, because the problems and the environment are too complex for this.

Since creativity plays a major role, abstractions and rules can play a central role. By abstractions, the problem can be seen from a different perspective. New solutions can arise via this perspective. Rules are uncontrollable for the designer, but he can't ignore them. A common way to deal with this is to apply it in mathematical research methods. Different solutions can be arrived at these methods. Obviously there are many situations where this is impossible. Consider a design with many variables. This increases the complexity whereby these solution methods are not feasible to achieve a suitable solution. In these situations, one should fall back on heuristic search methods and not all possible solutions can be explicitly mentioned. Sometimes it's not clear why this heuristic search method works well. While this is still an important part of the investigation, a proper working artifact has still the highest priority which in Guideline 5 already stated. But looking to characteristics of the environment, a satisfied answer might be found.

2.7 Guideline 7: Communication of research

Important here is that there is a good communication between technical and organizational people. Technically, because these people should be able to deploy the artifact and should be able to understand. Another benefit of good communication is that it is in practice easier deployable and it can be easier improved, extended or further investigate by scientists.

Organizational, because the artifact must fit in the context of the organization. Besides, Zmud [30] suggests that the information on the artifact is not important, but what it can do. In this way it is easier to look for the management if the artifact fits in context of the organization.

2.8 My approach to design science

The following figure is the theory above graphically summarized. Hereby I try to form a clearer picture. I also made some adjustments. Below the scheme I will explain why. The figure consists of blocks with a bold number. I will discuss these below the figure.

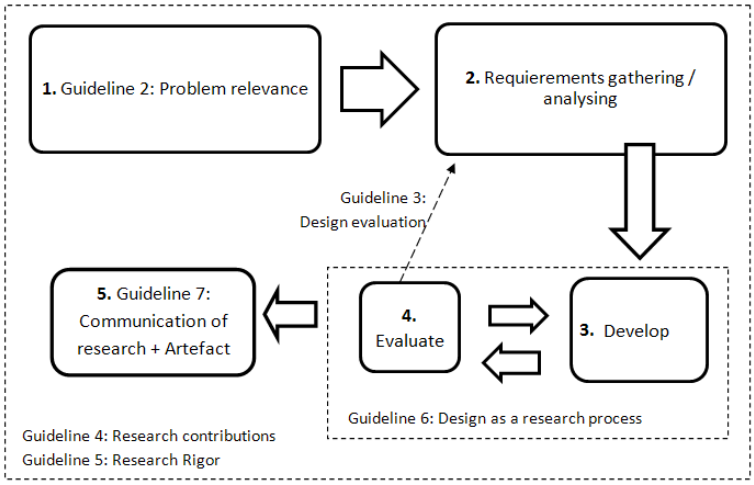


Figure 2.2: *Graphical summary of my approach to design science.*

1. Hevner writes everything as guidelines. I think he's doing this because every problem, environment or a combination of this is unique. And he would not exclude situations to set it up as rules or give it an ordering. In my opinion, guideline two is a rule. There is no reason to think of an artifact if there is no problem. Furthermore, he indicates that these guidelines don't need be sequential. I believe that guideline two will be first to be completed, because it must first be established whether there is a problem.

2. Once there is a problem, information will be won on the requirements and constraints of the problem and environment. This is of course already a little bit done in step one to validate whether there really is a problem , but will be written down formally in step two, so that it can be used in the next step and these requirements and restrictions again be consulted in the evaluation.

3. Now we enter the cycle that Simon [26] has already been described. The first time stepped inside this cycle, will start with thinking about the artifact, design the artifact and then develop the artifact, taking into account the requirements and constraints of the artifact and environment. Here begins guideline 6 which classified that you should be designed as a search process. In the next iterations will focused on improving the artifact. And perhaps adapting to the requirements and restrictions that were updated.

4. After developing the artifact, the review will take place. These can be evaluated according to guideline three. Importantly, the feedback into the requirements and constraints of the artifact and its environment. It will also be considered whether the artifact haves the desired result. Steps three and four will repeat until the desired effect is attained.

5. Now is the artifact ready for business and it's time to apply guideline seven, which indicates that the communication about the artifact is important. It is pure communication described in guideline seven, and no communication to the customer in the development process. Guideline seven would always be possible. For example if the artifact is not feasible and step five will never be achieved. But in general, this communication is always the last to be done.

Guideline four and five concern the importance of a contribution and rigor to examine is the entire process. Guideline one is an implication of guideline two.

Chapter 3

Game design theory

I would like to use GDT to improve the WOz method. GDT is not the same as game theory. However, people are still very divided about this and discuss what GDT and game theory exactly is. I am using Hoppenbrouwers's description based on Järvinnen, Salen and Zimmerman's work [14, 16, 25]. They said game theory helps to analyze strategies for playing games. GDT helps to analyze the structure of the game, like the rules, etc. excluding the behavior of human beings.

Many characteristics that a game must have, which help the development of the WOz method, are caught by the definition of Wilmont derived from Salen and Zimmerman's definition [29, 25].

Wilmont: *"A game is a system in which players voluntarily engage in a goal-oriented, artificial conflict, that results in a quantifiable out-come. The activity takes the form of a process which is defined by rules, yet offers freedom of action."* [29].

I think Wilmont's definition is too wide for my purpose. So the definition I use is:

A game is a system in which players voluntarily engage in a goal-oriented, artificial conflict, that results in a quantifiable out-come.

I don not want to dissect the whole definition, but I will discuss the important points which I will apply within the WOz method. For an excellent explanation of the whole definition, I refer to Wilmont [29].

System: Like Järvinnen said, a game can been seen as a (information) system and vice versa [16]. Both can have the same elements and consists always of the same basic elements, namely: components, information, mechanism and a rule in the form of a goal which will be explore in more detail in the next section.

Goal oriented: Games have a clear goal. Mostly more than one. Goals are a kind of rule. In the WOz method there are also rules which are the boundaries of the system and providing the procedure.

Players: In a game there will always be players. Without these, there is no game. In the WOz method, the players could be seen as the participant and the wizard. The only distinction between a real game and this method is that the participant does not know he "plays" against a human being instead of a computer.

Voluntarily: People always play games voluntary. Often people have to pay for a game. In the WOz method, people participate voluntarily, because the participants are the customers and want a system, and the company that build the system wants a satisfied customer.

Artificial conflict: Wilmont means with artificial that the game designers created their own boundaries in time and space. That is the reason why it is artificial. The conflict is the thing that must be eliminated to achieve the goals. In other words you can say that this represents the challenges in the game. In the context of the WOz method, you can say that the artificial conflicts are the difficult tasks for a computer to solve and these are the wizard's tasks, particularly in a early stage of development. In the next chapter these artificial conflicts will be explored more.

3.1 Game elements

Every game that exists, can be divided according the elements below. Not only these elements can be applied to games. Hoppenbrouwers has already shown that there is a clear link with modeling [14]. Also, I think using these game elements in the WOz method can provide a clear guidance. Below, the 9 elements are described. In the next section, some of these elements are included as important principles.

1. **Components:** These are objects that the player can manipulate or possess in the course of the game. For example, a deck of cards, pawn on a board game or a football.
2. **Rule set:** Rules provide possibilities and limitations in a game. Roughly, designing a game is similar to making rules or using of existing rules for a new game. In general, rules makes the relationship between game elements. Rules can be divided into two main functions. Goals and procedures.
   (a) *Goals:* Rules for the purposes of targets are used to end conditions to give the game to win. An example is get more points than your opponent, or play your cards on the deck rather than the opponents.

   (b) *Procedures:* The second type of rules dealing with procedures. It defines how the game should be played. For example, how the cards should be dealing to the players of the game or the behavior of the components in a virtual game. Mainly, these rules can be used for two purposes. (A) Assigning values to different game situations, such penalties or rewards. (B) The relationship between game elements and their attributes.

3. **Environment:** The environment is the stage of the game, like a board, a field or a virtual environment.

4. **Game mechanics:** This means the interaction between the player and game elements. With this mechanics a player can try to influence a game state. For example: throwing a basketball in a bucket or maneuver with a car on a digital race circuit. Usually provide game mechanics the quality of the game. The unpredictability, such as whether you get the ball in the bucket, provides the excitement and challenge for players who play the game.

5. **Theme:** The theme is the subject of the game. This gives the game a game context and the components gets a meaning. For example a personal shooter game set in the second world war or even in the future. Actually, the theme has a metaphorical meaning. It helps the player showing the rules in the context of game rather than the rules themselves.

6. **Information:** This is what the system and players need to know about the game state. This can be achieve for example by a score board. There a couple of different kinds of information in a game and this can be divided in four categories:

   (a) *Information about the events:* this can be seen as result information, like the consequences of game mechanics.

   (b) *Information about the agents:* this means the information about the player roles, location of the players etc. This information can be often seen in virtual games with artificial intelligence.

   (c) *Information about the objects:* information about the attributes and components.

   (d) *Information about the game play:* Information in the form of procedures from the rules and given information about the state of the game.

7. **Interface:** This can be seen as the instruments which the game mechanics can be controlled, like a race steer or a mouse and keyboard.

8. **Player(s):** Players are the ones who play the game. The human factors which affect the game, for example their behavior, emotion, skills and the relationship they have with games or ever have played and preferences as they kind of game.

9. **Context:** The physical location of the game. The time when the game is played. The background of the players who play the game and other external factors that may affect the perception of the game.

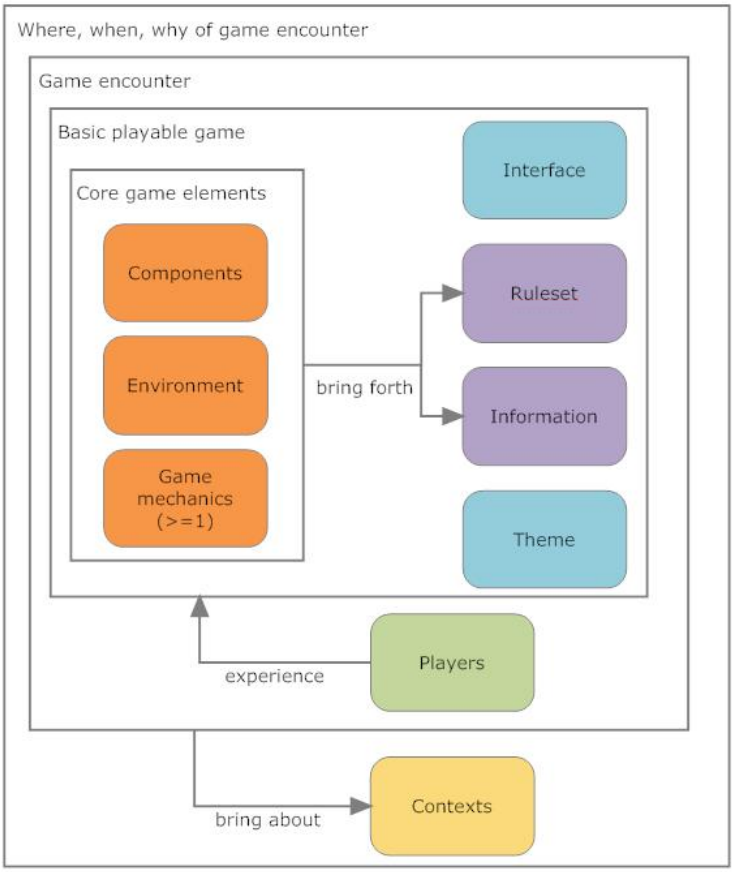


Figure 3.1: *Relation between game elements created by Wilmont [29]*

As Hoppenbrouwers [14] indicates, not all elements are required in a game. The minimum which is certainly needed it as a game to consider are:

- Components with associated rules or the conduct of the components.
- An information structure that the game status, component attributes and maintains relationships.
- At least one mechanism for game players to do something.
- A goal in a final status or win status covers. Without these four minimum requirements, the player has no purpose or intention of playing the game.

3.2 Game principles

What makes a game successful or unsuccessful? This is all based on the basics of motivation in my opinion. Just the wish to continue to play or want to play again, means that a game is successful. This is clearly reflected in the definition of game that I use, namely the word "voluntary". To motivate the player, there are a couple of principles to use. Not all principles will be discussed below, but the one which can also apply to the WOz method. It is important that all the principles are sufficient present, otherwise there will be no motivation for the game. It is important that all fundamentals are presented in a sufficient way, otherwise there will be no motivation to play the game. I see this as an analogy of a house of cards. The strength of the house of cards symbolizes the size of motivation. Each card is a solid foundation and shows the degree of firmness. The firmness shows how good each card is developed. So Every card can stand still strong, if there is one card not strong enough, the whole house of cards is tumbling down.

3.2.1 Meaningful play and goals

First it is important that the game has a goal, because if there is no goal that can be achieved there cannot be a winner. In 1955 Johan Huizinga came with a concept[15]. From this concept emerges the sense of game play, but this concept can be interpreted in many ways. Salen and Zimmerman[25] have converted this interpretative form in meaningful play. It means that playing a game has a goal and is not an unordered collection of actions. If you play a game, you make your choices. For example, where do I put my pawn down or I will skip a turn or pay a fine. All choices and actions based on the context of the game. Actually, a game itself is nothing, but rather the way people interact with each other within the rules or with the game mechanics makes a game. A good example of Zimmerman is chess. The choice one makes is based on the position and the opponent's pieces and the relationship between them. Once the action has been completed by a piece on the board to oppose and the relationship between the pieces changing, there will be a new situation. By trying to understand the interaction, it helps us to see what happened inside the game.

Meaningful play means, you must have a goal to achieve. As I mentioned in the previous paragraph, a goal is a subset of rules. Another property which can be helpful for improving the WOz Method is goals. There are three kinds of goals, namely short term goals, middle term goals and long term goals. A long term goal can often be achieved at the end of the game, like defeat the boss. The middle term goals are goals that can not be directly solved, but in a short period can be achieved. For example, get all the points before the door opens to the next level. Finally there are short term goals, which are simple goals that can be achieved quite easily. These goals, ensuring that there is a certain motivation and interest in order to continue playing.

3.2.2 Autonomy

Autonomy or freedom of actions is also an important principle. The player must have the feeling to have all the freedom and the ability to make choices in the game. If a player is limited in his actions will impair his autonomy and the game will be less interesting, because he feels that he is controlled by every action. Autonomy may be obtained by applying the game mechanics in a flexible manner, "movements and strategies have to be free choices, and any rewards should provide feedback rather than controlling the player's behavior" [29]. There must be taking care of that autonomy does not violate meaningful play. Thus with soccer it makes no sense to make a slide tackle if there is no ball around. As Zimmerman points out, guidance and freedom of action are mutually exclusive. Outside the independence, the game should be able to give the player a sense of ownership, so that in every action the player gets a response from the game [23, 29].

3.2.3 Challenge and reward

Another important principle is challenge and reward. It is all about a player being challenged and depending on his performance he should be punished or rewarded. Important is that the degree of challenge, matches the skill level of the player [27]. This can be seen as an upward spiral which increasingly challenge the skills and increased the skill to more difficult challenges demands. If the challenges are to difficult for the player, this creates anxiety and if the challenge is too easy, there will be potential apathy [17]. A proper level of challenge will be prevent a game from being boring or discouraging [27]. The experience will partly be obtaining new skills, positive and negative feedback and to be challenged, will motivate the player to continue playing. Not only the challenges are important, but also the rewards or punishments for the challenges. The reward of punishment must satisfy the expectations of the player. Thus, for a difficult challenge a higher reward is needed than a less difficult challenge. When this is not in proportion, the system will not work [29].

3.2.4 Accessibility

With good controls and interfaces you can obtain good accessibility. This is particularly used in the virtual gaming industry. The game should not be difficult because the controls are hard to learn, but because the game is challenging in itself. This can be achieved by using only a limited number of game mechanics [29, 22]. A few years ago, the controls could only have adverse effects. If the controls were bad, the motivation would decrease but if the controls were good, the motivation would not increased. Since the Wii or XBOX Kinect is on the market, the controls can also increase the motivation of a player. The interface has always been either a positive or negative motivational factor. Games with nice graphics, are always in favor. Not only the appearance but also the structure of the interface (where is your score or mini map displayed) is important.

Not only the controls and interface are import for the accessibility, but also the game mechanics. Players should be able to start immediately without reading the manual first. Often, games start with tutorials to explain the rules and controls [12]. "In-game tutorial feedback can be used to allow quick progression in learning the basic mechanics for playing" [21, 27]. The game mechanics are structured so the player feels constantly challenged. Game mechanics are actually meant to achieve the goal and therefore they are closely related. Again, we can make a distinction between main mechanics and sub mechanics. Main mechanics are the basic mechanics and sub mechanics belongs to a main mechanic. Wilmont gave an example about hitting an opponent in a virtual boxing game. Hitting the opponent in order to weaken him is a main mechanic. Picking him up from the ground to hit him again is a sub mechanic[29]. It is important that game mechanics are not to difficult, because this decreases the accessibility. According to Rose[22], adding complexity to the player input, will not change the output substantially. "Therefore the complexity of the mechanics has to be well thought over, as it may never interfere with the gamer's ability to comprehend the game world and active experiences." [29]. It is therefore important to remove unnecessary mechanics and to focus on a small number of mechanics.

Chapter 4

Wizard of Oz method

The WOz method is a widely used testing method in the HCI. The main core of this method is to check the requirements and expectations of the customer before the whole system is implemented. The way this happens is by telling the customer the system is ready for testing but has some performance limitations. From now on, we call the customer a participant, because he will participate in the test. Actually the system contains no or a few functionalities. The participant testing the system with this limitations. The wizard sits in an another room and observes the participant's action. The participant starts with testing the system, but actually the wizard simulates the system's response. The participant is observed and there can be seen what the effect of the "system response" is. After the test, the participant will informed that he interacted with the wizard instead of the system. It's important that he does not know this before the test, because this influenced the test [8].

This method makes it easier to design a system. Via this way systems it can be tested before investing money and time in complex I/O or algorithm and checked if the participant gets what he wants, without building a whole prototype [9]. The figure below shows a scheme of how the WOz method works.

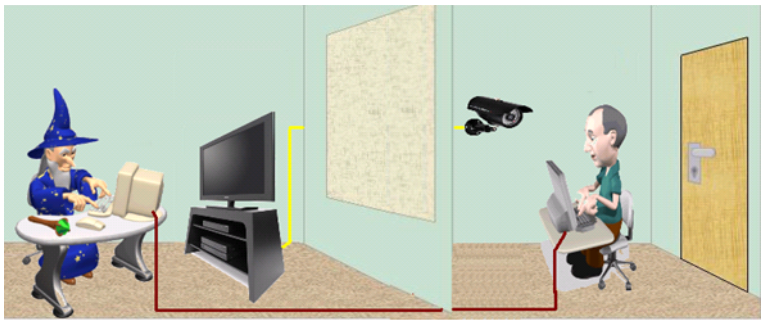


Figure 4.1: *On the left side sits the wizard and monitoring the participant which sits on the right side.*

4.1 Tasks that the wizard simulates

According to Dow [10], the function of the wizard that simulating the system can be divided into three groups, namely: responding on sensor input, sending contextual data and simulating system intelligence. Below I will briefly describe these three with some examples. Obviously, many more applications and examples can be found.

Sensor input is difficult to interpret for a computer. Computers are very good at math and even the best chess players. But for a computer it's difficult to understand the outside world through image or sound (or any other input). For a computer a picture or sound is a series of ones and zeros. It is very difficult to indicate what they mean in the outside world. A simple example is dictation software (which the user is talking and the computer "types"). Sometimes, this generates a strange word, because the computer does not understand the word. Computer vision is also a good example. For example, where the computer is able to recognize objects or emotions. Through various learning models a computer seeks to understand all this input, but this is something which is not trivial. Perhaps in the future sensor input will be more advanced, but now this is a good task for the wizard before building a whole prototype.

Contextual data is also difficult for a computer. Take for instance why-questions. Recently Verberne is performed in a research where she tried to resolve why-questions by a computer. In her research she could answer about 60 percent of these questions. The main reason that she could not resolve the other 40 percent was the lack of knowledge. Computers do not have associative knowledge. The example she gave was as follows: Consider the question: why do Americans speak English? If you read the answer: "British colonization brought the English to the United States" than you unconsciously put the relationship between 'Americans' and 'USA'. This knowledge is very difficult for a computer because the two words are not exactly synonyms to each other. Yet the association be-

tween these words is for human readers very strong [3].

System intelligence means that the computer "thinks" as a human being, i.e. solves problems or makes decisions. An example can be found in a virtual soccer game, such as Fifa 2010. Playing an advantage is something which often goes wrong. If a player of team A tackles a player of team B, but the player of team B can play on (with ball possession), then team B plays an advantage. At the moment the player of team B has not the ball anymore, there is no playing an advantage and the referee (computer) might have to blow the whistle. The problem is that the distinction in playing an advantage often goes wrong, because for example the player of team B just touches the ball and then loses the ball to a player of team A. The computer thinks that player B played in advantage and then lost possession of the ball.

These are the three main groups in which the wizard can play a role, but obviously the wizard can take all random tasks. Only some tasks are easier to program and test by the computer than by the wizard.

4.2 Advantages and disadvantages

The WOz method have some advantages and disadvantages [4]. They will be described below.

4.2.1 Advantages

- Future technologies can be tested before building an expensive and time consuming prototype, because the wizard provides this functionality.
- Iterations can be rapid, especially by minor changes, because the code does not need to be changed to be tested.
- It provides a good understanding of how the participant interacts with the system, because the wizard is also interactive and observes the participant.
- The system can be tested in every stage of the design process. The WOz method can be deployed at different times in the project, as is illustrated in the figure below [13]. In the beginning of the project, the wizard serves as controller, where almost the entire functionality the wizard arrives. Later in the process serves the wizard as a supervisor if the system has almost all of the functionalities, but the wizard still has total control and can intervene if necessary. Example due to errors in the system. The compromise between controller and supervisor also exists, but is less common [13]. This is mainly used when a system component has been implemented, but not quite to be trusted on its operation. If an error occur, the wizard comes in action here.

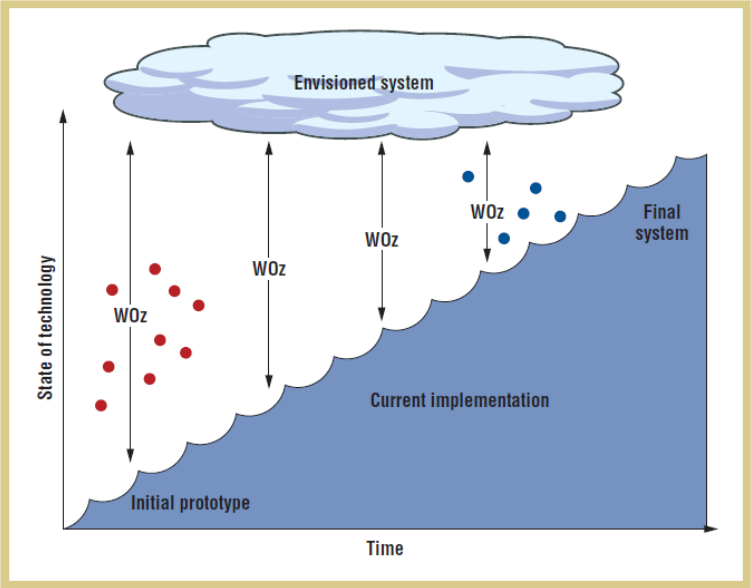


Figure 4.2: *A general scheme which over time can be applied to the WOz method. The left dots are located at the moment where the complex functions are not in the prototype, and the wizard takes over this. The right dots are located at the moment where most of the functionality already built and deployed, and the wizard acts as supervisor.*

4.2.2 Disadvantages

Unfortunately this method also has some disadvantages. I will try to eliminate these using GDT so that a more powerful testing method will arise.

- The wizard requires a training. This means extra costs for the project.
- It's hard for the wizard to respond adequate and consistently, because the role of wizard requires a lot of cognitive ability, especially if the test takes a long time or the wizard must respond to a lot of sensors.
- It's difficult to test a large GUI, because there are a lot of variables which require attention.

It should also be kept in mind that with this method specific attention will be paid to the HCI. Errors created by the system (particularly on performance) will not be detected. This can be regarded as a disadvantage, but when placing emphasis on the interaction between user and system, this disadvantage can be ignored.

4.3 Requirements for the WOz method

As Salber described [24], this approach has a number of requirements to a functional WOz method to comply. These are viewed from three perspectives.

4.3.1 Requirements from the wizard perspective

This perspective concerns complexity of the tasks for the wizard. In other words, it means that the wizard functions properly. The difficulty here is that the wizard should observe very closely the participant. Video recordings alone are not sufficient[24]. So the wizard uses a computer that helps observe the participant by collecting system data, like which buttons has been clicked. Additionally, batches are hereby established, so not everything needs to be performed by the wizard, but the computer can help here.

An other thing that often occurs is a multi wizard setup, where not one but two wizards are used. The first wizard (I/O wizard) focuses on the interaction of the system and on the user, while the second wizard (Task Wizard) tries to interpret the actions of the user that he receives from the I/O wizard and tries to give a correct response to the I/O wizard who sends it to the user. In an experiment, this configuration proved to be a successful configuration [11]. Also it's not unusual to have multiple wizards, which each user input (mouse, keyboard, voice, camera) is observed separately by a wizard.

4.3.2 Requirements from the system perspective

From this perspective, it is important that the system has a flexible and an acceptable response time. This response time is mainly determined by the wizard. How quickly he gets input from the participant processed and how quickly he may give a response. The flexibility of the system according Salber [24] can be divided in two parts. First the configuration flexibility, which means the number of wizards and input and output devices. It does not have to be the case that the wizard always stays at his input and output, but if another wizard is heavily loaded, the less heavily loaded wizard can take over part of the job.The other part of the flexibility is the communication protocol between the various wizards. Certainly for the performance, it's important that wizards communicate well if there is used a multi wizard setup.

4.3.3 Requirements from the evaluation perspective

Since there is a lot of test data generated (camera, voice, mouse, keyboard, etc.), it's important that it will be stored in a well-structured way. It's also an advantage if this data is synchronized and the best thing would be if some automated analysis can be performed, something which on longer term is not unthinkable. I will not go into the automated analysis, as this is beyond the scope of the project, but for the sake of completeness. It is, however, important

for my project that any test data is structured and synchronized, for easier evaluation.

4.4 Procedure

There are many variants of the WOz method. Nevertheless, there is always a basic procedure on which these variants are based. The basic procedure presented below is a citation of the usability body of knowledge. "The basic wizard of Oz procedure involves the following steps:

1. Develop a simulated user interface for the target technology.
2. Develop a detailed test plan with the instructions for the facilitator, wizard, participants and other staff. Determine if you need to set any expectations about the simulation's "performance" so participants are prepared for sub-par performance.
3. Recruit users who meet the appropriate user profile, try to cover the range of users within the target population.
4. Prepare realistic task scenarios for the evaluation.
5. Develop a procedure where the wizard can respond to input from a participant.
6. Train the wizard.
7. Design the instructions for the study so that the participant knows that they are working with an early prototype and that performance is not "optimized" yet.
8. Conduct pilot tests to refine the procedure and give the wizard some practice. Make any changes to the procedures and test plan.
9. Ensure recording facilities are available and functioning.
10. Conduct each session. The facilitator instructs the user to work through the allocated tasks interacting and responding to the system as appropriate.
11. Conduct a debriefing of the participants. Obtain feedback on the "performance of the wizard system". Tell the users about the wizard and explain why you couldn't tell them earlier.
12. Collate, analyze, and summarize the data from the study. Consider the themes and severity of the problems identified.
13. Summarise design implications and recommendations for improvements and feed back to design team. Video recordings can support this.
14. Where necessary refine the prototype and repeat the above process." [4].

Chapter 5

Decomposing WOz method

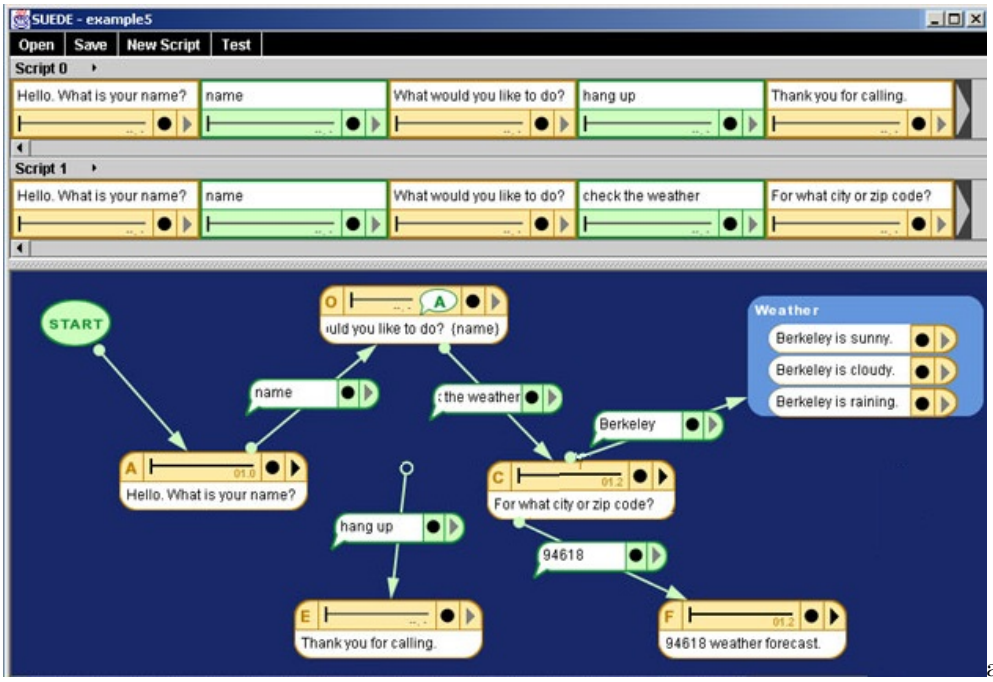
In order to make a proper framework for the new WOz method, it seems prudent to first analyze existing WOz cases. This way it will become clear what for each WOz case is generic and what for each WOz case is specific. I have chosen for two totally different cases to get a clear view on the generic and specific parts. To get a clear picture of the cases, I created a model of it, like a data model using Object Role Modeling (ORM). After I created this model, I tried to decompose the cases into the game elements described in section 4.1.

In scientific studies, a number of prominent cases are frequently used. Two of them I would like to use for my research and totally discuss in detail. What is striking about all these existing cases is that they are quite wide-ranging and not meant for a functionality test, but for testing a whole system on a certain topic. For example the first case that I will discuss is called SUEDE. This case is about speech user interfaces, which can be seen in many automated answering machines. The other case is called Topiary and this is a tool for location-enhanced applications prototypes. Again this case shows that it serves a wide-ranging functionality. For example, "searching available room tour guide" and a "friend finder" or "nearest thing finder". In daily business life it will be more likely that one feature about a particular product will be tested with the WOz method, than a whole system about a certain subject. In the following sections the cases will be further decomposed.

This decomposition is described like a process. Since this is a process in which it is becoming increasingly clear what the WOz method exactly is, the meaning of the usual game elements also change slightly. This began with the definitions described in Section 4.1 and then slowly transformed into definitions that perfectly fit into the WOz method. The intention is to do this as transparent as possible. In this chapter, no game principles will be applied to the cases or the game, because I don not want to judge how well the game works, but just search for the structure of the cases and the game.

5.1 Case: SUEDE

As I already mentioned in the section above, SUEDE is a speech user interface. In this case the wizard is able to begin drawing a case in design mode. This can be tested in test mode. Below is a picture of an example scenario.



as much as possible pieces.

Figure 5.1: *GUI example of design interface.*

Figure 5.2: *GUI example of test interface*

By so-called prompt and response cards placing in a sequence, a use case can be created. Prompt cards which are meant as "answering machine" and response cards as the users answers. If the answer is not available in testing mode, the wizard can indicate this through a click on the button "this answer is not possible". With a minimum number of components and with a fairly wide range of interfaces regarding the speech user interface, this program can test a lot of cases. I do not want to discuss the precise working of this program. In appendix A, you can find the user manual of this program[1]. Below, you can find a ORM model which is extracted from the documentation. This ORM scheme shows the system SUEDE. The users of the system are not modeled. After this scheme, the game elements are extracted from the documentation and the model.

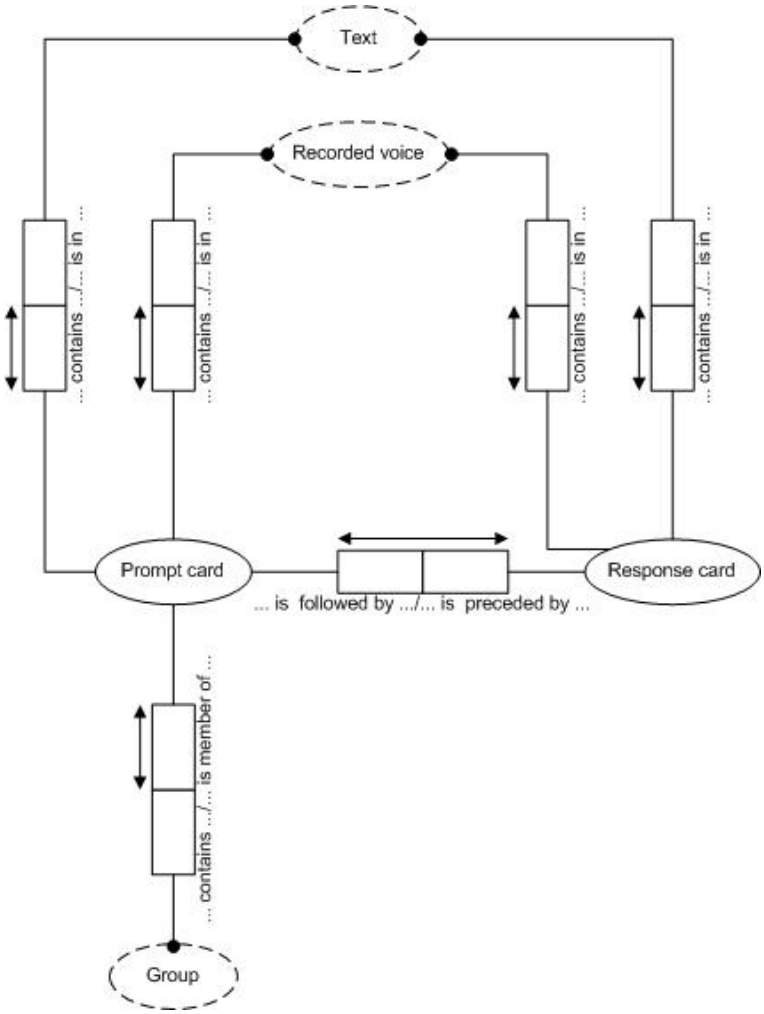


Figure 5.3: *ORM model of SUEDE at system level.*

Game elements

Players: two players namely, human and human (and one player in design mode)

Components: Design mode: buttons to handle prompt cards, response cards aas much as possible pieces.nd group labels. Test mode: buttons to run test.

Environment: Virtual environment

Theme: Design mode: space to put prompt and response cards in and connect these with each other. Test mode: screen with buttons

Interface: Design mode: mouse. Test mode: telephone (participant) and mouse (wizard)

Context: N / A

Game-mechanics:

Design mode:

- Put prompt cards, response cards or group labels on space
- Give prompt card, response card or group label a name
- Record voice to prompt card or response card
- Connect prompt card to response card and buttons
- Put prompt cards in group labels

Test mode:

- Click on button

Information

1. Events: Arise from rules
2. Agents: N / A
3. Objects: How are the cards and labels linked with each other
4. Game play: These are the rules

Rules

Goals
Design mode: To complete a user speech interface
Test mode: To complete the scenario that is created in the design mode

Procedures
Design mode:

- A group label consists minimally one prompt card
- There cannot be two different prompt cards in sequence
- There cannot be two different response cards in sequence
- A prompt card or a group label will always have a predecessor, like a start or response card

Test mode

- Click on a button that represents a prompt question/answer

- "Barging-in is when the user begins giving his or her response before the prompt is done playing. If the user does this, you can click the "barge-in" button and SUEDE will stop the prompt from playing and will immediately start recording"[1]
- "If a user does not response to a prompt after a certain time it is considered a "timeout" error. In SUEDE, if this occurs, the wizard has to option to click the "time out" button which will stop the recording and repeat the prompt"[1]
- "If you don not hear the user's command pressing this button will play a pre-recorded message asking the user to repeat themselves"[1]
- "If the user's response does not match any of the response links pressing this button will play a message telling the user that they are not able to say that at the moment"[1]

5.2 Case: Topiary

The second case is Topiary. As I previously mentioned, this is a tool for location-enhanced applications prototypes. It was created in 2004 with the aim of supporting interaction designers in early stage of the process of developing a system. The tool can create a map containing places, persons and objects and these can interact with each other. An example is elaborated in [18] which is about a school campus. People get a PDA which is showing a menu. If they are near an object or place, the menu will automatically adapt the context. Also the PDA can give a signal when people get close to each other. The development of the system includes a number of phases. First a map with the places, persons and objects must be defined. Thereafter, menus should be created and last the conditions when they should be active on the PDA must be defined. These steps are the design phase. When the design phase is done, testing follows. The designer can manually move persons and objects all over the map to see if the menus are working properly and at the right time. In the second phase of testing, a person gets a PDA, the designer is now in the role of the wizard, and moves the person over the map manually by a tablet pc. In the final test phase the person will be equipped with GPS and the design will be tested fully automatically. These three phases are a perfect practical example of figure 4.2 were Dow [10] shows that the wizard can be used in several stages of the development. Below is an illustration of this program to get an impression of it. The precise working of it, can be found in appendix B.

Figure 5.4: GUI example of design interface.

Striking in this case is that today it is fairly easy to test a GPS system. But in the past a high level of technical knowledge to relatively low-level sensing technologies such as GPS was necessary. It was a heavy job to produce a prototype, evaluate and improve this. This case shows that although the WOz method can always be performed, there is a shift at the moment at which it is deployed.

Like for the first case, a data model has been created. This can be found below. In this model, the user is also not modeled, but only the system.

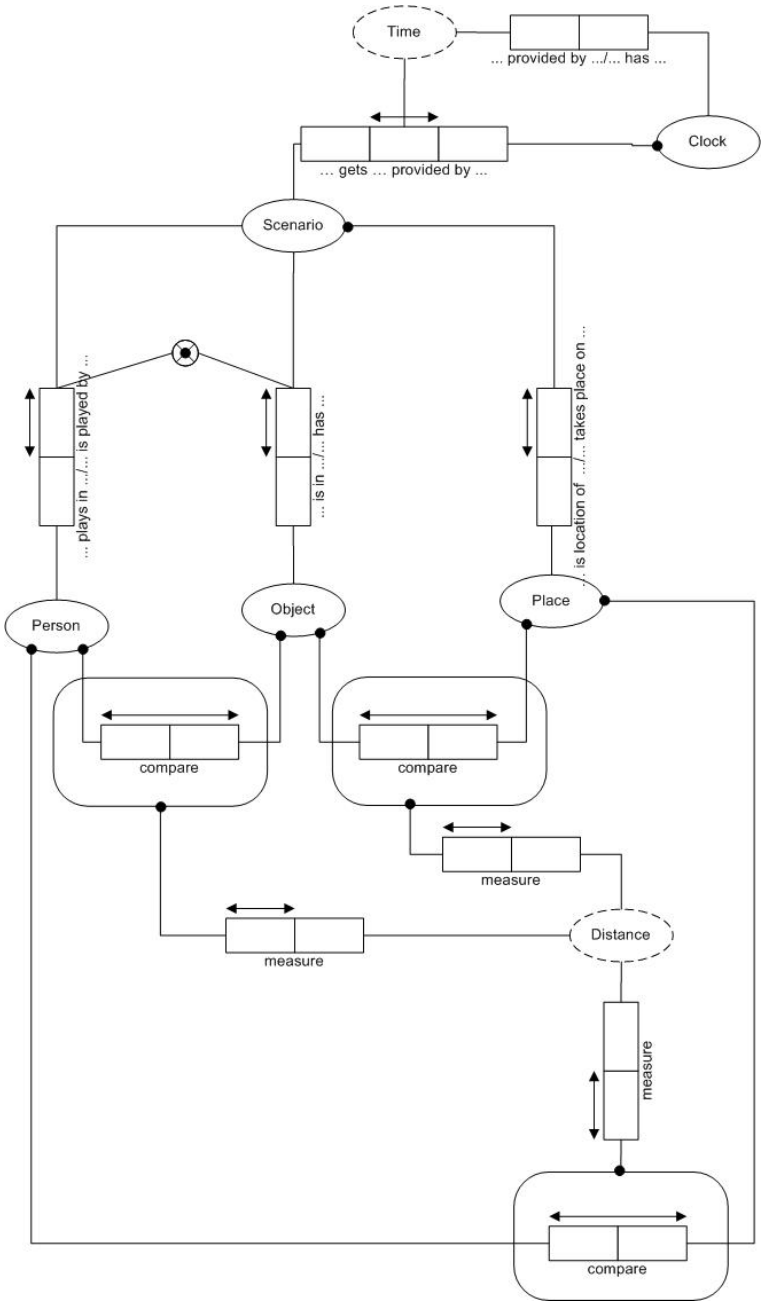


Figure 5.5: *ORM model of Topiary at system level*

Analyzing the cases SUEDE and Topiary was not only done by ORM, but also

with Business Process Modeling Notation (BPMN). In the case of SUEDE, I got a slightly representable model. In Topiary I did not, because Topiary is a larger case. The abstraction level can be adapted so that a case of this magnitude does not give a mess of arrows and can be quite normally modeled. A disadvantage of this is that at this abstraction level, the model has no meaning anymore and can be interpreted in many ways. The reason why I do not show BPMN in my thesis is that it gives no clarity in the decomposition of the WOz method, except if the idea is implemented in the extreme. This would render the model below. This model is not about the design phase, but purely reflects the test phase whereby the wizard interact with the participant.

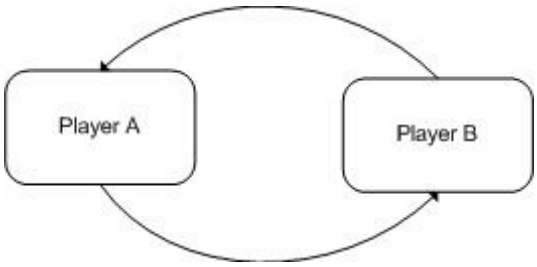


Figure 5.6: *This model shows an interaction between two players. Player A makes a turn and then the other does. This continues until an end stage is reached. This is end stage is not processed in this model.*

The above model is rather worthless. It gives no information about the case. On the other hand, it can be seen as a generalization of any (information) system or game. This makes it clear that a process notation has no value for elaborating on a case. However, it gives insight in the development and improvement of the WOz method. It shows that the cases can be better described in a declarative language rather than an imperative (process-oriented) language.

Game elements

Players: two players namely, human and human (and 1 player in design mode)

Components:

Design mode: buttons for: place entities, person entities and object entities, pages.

Test mode: buttons for: person and object entities that can be manipulated by the wizard. The participant can press on buttons and in a later stadium he can manipulate his own location by GPS.

Environment: Virtual environment and in later stadium also outdoors.

Theme: Design mode: map where place can be indicated and person and object can be placed. Test mode: map where person and object can be moved.

Interface: Design mode: mouse. Test mode: PDA (in later stadium with GPS) (participant) and mouse (wizard)

Context: N / A

Game-mechanics:

Design mode:

- Places, persons and objects can be put on the map
- Menus can be created this
- Constrains when the menus appears can be set

Test mode:

- Person and objects can be moved

Information

1. Events: Arise from rules
2. Agents: N / A
3. Objects: location of the person and objects
4. Game play: These are the rules

Rules

Since this case is too large to show all procedure rules, only certain rules will be listed.

Goals

- Design mode: Create a map which is useful
- Test mode: Menus appears on the right place and on the right time.

Procedures

Design mode:

- Each entity must have a unique name
- There can be created places within other places.
- Everything can be draw or written in pages.
- There can be create a explicit link when a line is drawn from a page to another page.

.

Test mode

- Entities like people and things can be moved. Places not
- Menu will be shown when two persons come close together
- Menu will be shown when a person come close to a place with context.

5.3 Game: Draughts

Besides the two cases, it seemed useful to decompose a game in game elements and modeling an ORM scheme of this game to get a clear picture on how this works in a real game instead of an (information) system. The reason why I chose for draughts is because it is a relatively simple game and this makes it all concrete and clear. There are many variations on the rules. The rules that I use are described under rules of the game elements below [6]. But first, I will show the ORM diagram.

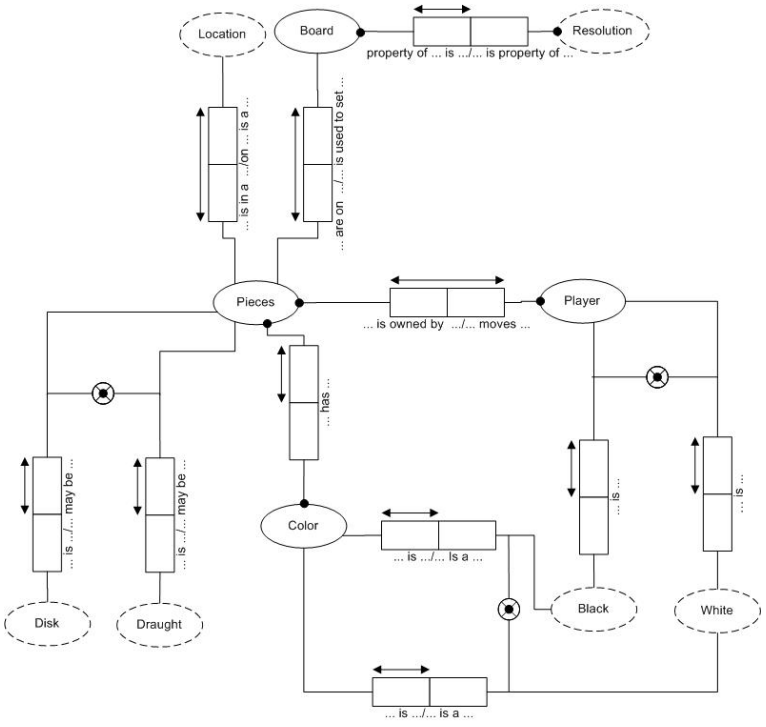


Figure 5.7: *ORM model of the game draughts*

The ORM model above, shows that many game elements can be seen in this model. For example "pieces" are the components of the game and "player" is the player of the game. I chose for modeling the players in the model, because this gives insight in what players interact with. In the previous models I deliberately did not do this, because they were modeled on another abstraction level.The game elements, which I have tried to extract from draughts, will be shown below. Some elements are not applicable, such as context. I also think that theme and environment can be seen under the same denominator, because in a system there is no real theme and a system is always a virtual environment. With the game element theme in this sense, the appearance of the environment is meant. This can be seen as the GUI but will not be included under interface because it does not relate to the controlling of the system. Compared to the two cases above, the game element "game mechanics" is slightly changed. In the previous cases the game mechanics describe what could be done with it. This analysis will take a deeper look at how this is done. Also, the rules in the first cases are handled differently from the rules of this game. This difference is in the format. From now on, rules will be written as atomic sentences, with one rule per point described. This is done purely for clarity.

Game elements

Players: two players namely, human vs. human, human vs. pc, pc vs. pc

Components: Disks

Environment: Virtual environment

Theme: Board with 10 by 10 squares (black, white sequenced)

Interface: Mouse

Context: N / A

Game-mechanics: Opposing disks

Information

1. Events: Arise from rules
2. Agents: N / A
3. Objects: Location of disks and draughts
4. Game play: These are the rules

Rules

Goals

Ensure that the opponent cannot do any legal move anymore. This can be:

- The opponent have no pieces anymore

- There will be no possibility to win, for example if both have one king, it will be a draw game

Procedures

- 1 player is white, the other black
- The white player begins
- The players set all disks on the black squares closest on his side.
- Pieces are only allowed to shift diagonal 1 square farther.
- Pieces are not allowed to shift over other pieces.
- White disks cannot be placed on the location where are black disk is possessed.
- Capturing is jumping over the opponent disk to an unoccupied square.
- Capturing is mandatory.
- The player is allowed in case of capturing to shift backwards.
- It is mandatory to capture as much as possible pieces.
- By capturing multiple pieces it is allowed to make an angle of 90 degrees.
- If the opponent forgot to capture, the player may decide if he let his piece on the board or still let it be captured.
- The captured pieces may only be removed when his turn is over.
- In a capturing over several pieces, the same piece may not beat two times.
- When a player reached the furthest row of his side, his disk becomes a king. This is done by put an extra disk on the disk.
- A king is allowed to move backwards.
- A king is allowed to move diagonal multiple squares at the same time.
- A king may capture a separate piece that is possessed in the same diagonal line. The player is free to decide where he put his king in the same line.
- The player is allowed to place the king in a line were he can directly capture another opponents piece.
- The rule which said multiple captures is mandatory, disks have the same value as kings.
- A special situation arises when the disc is arrives at the furthest line , but still has to capture (in a backward move).In this situation, the disk does not become a king, because at the end of the turn the piece is not possessed on the furthest line.

5.4 Findings

One thing can be quickly noticed, after decomposing the cases and the game. Rules make the game! A game can be seen as a free world, where everything is possible, until a rule says otherwise. As Hoppenbrouwers beautifully said during a presentation "Games have rules. But games also leave lots of space to decide your own moves and therefore to make mistakes, or be brilliant."[2]. Hoppenbrouwers [14] already makes the distinction between goal rules, what the state defined when you win, and procedure rules, which have again two variants. (a) assign values to different situations and (b) relationships between the attributes and game elements. As part of the real method I like the distinction in the rules otherwise explained and give my own swing to it, but there will be an overlap between my and Hoppenbrouwers his definition.

I would like distinguish five main types of rules. Starting with goal rules. These are almost the same as in the definition of Hoppenbrouwers. Only, I still make the distinction that I mentioned in chapter 4. Namely, long term, middle term and short term goals. Applied to draughts it looks like: the long term goal is to win. This is done by forcing the opponent in a situation where he is no longer able to set a regulatory step. Middle term goal is to capture disks or draughts of the opponent. A short term goal is to get into position to beat the opponent in the next turns.

Besides these goal rules, I want to introduce initialization rules. These are rules that (a) prepare the game and (b) the first step to start the game. Back to the draughts as an example is (a) "the players set all disks on the black squares closest to his side". An initialization rule to do the first step is "The white player begins". After this rule is executed, we come in a loop where player A follows player B, etc. as shown in figure 6.10.

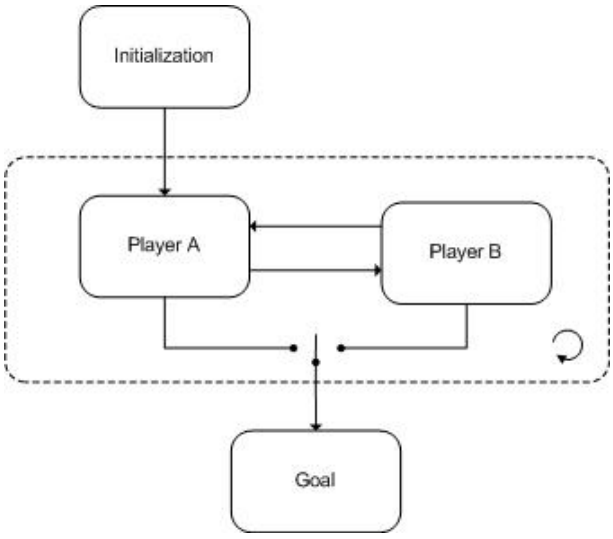


Figure 5.8: *The system or game starts with the initialization rules. When these are executed, the system comes in a loop were player A and player B interact with each other. When the long goal term is accomplished, the system steps out of the loop. Al these blocks are procedure rules at a very high level.*

This figure is a very high level of a system or game that is obtained from the BPMN models. If we zoom on this figure on another way and not with a BPMN view, we can, with less grip on the real world that defines the model, create a good picture of the game or system. For this, I introduce two types of rules. Namely boundary rules and procedure rules. Procedure rules are the rules that can be seen as a decision tree. The boundary rules can be seen as background rules that say something about the procedure rule. For example the procedure rule, "Capture opponents pieces" and a boundary rule "By capturing multiple pieces it is allowed to make an angle of 90 degrees". Here an example given of draughts to clarify procedure and boundary rules.

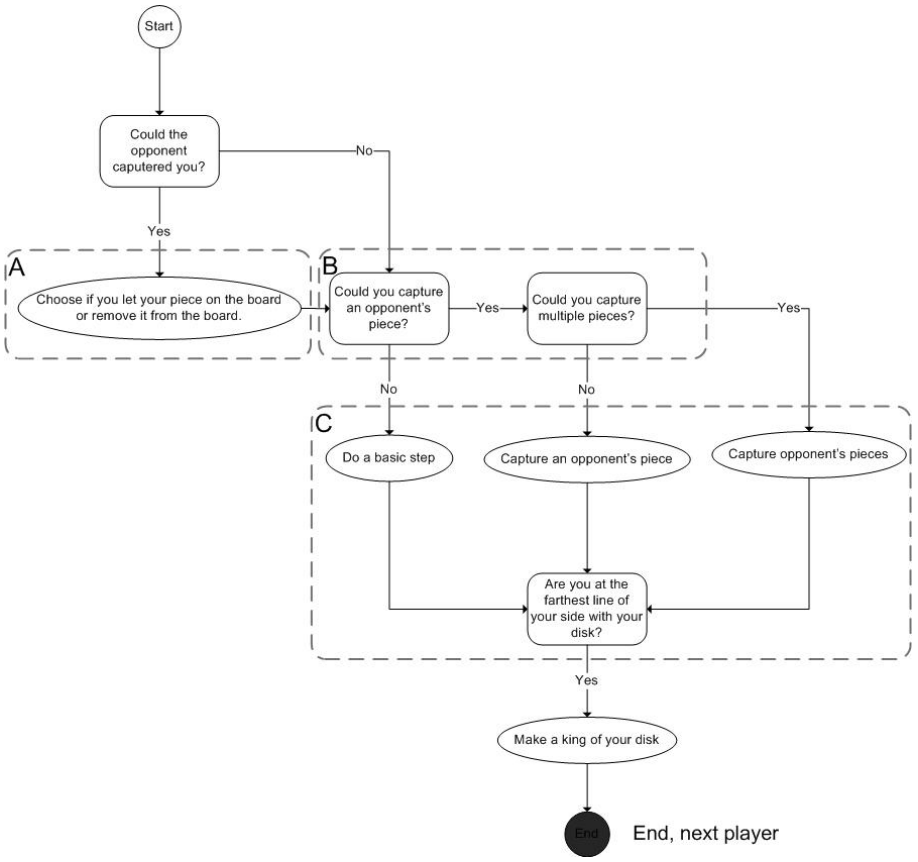


Figure 5.9: *Procedure rules for draughts.*

What is striking is that no boundary rule is visible in this model. This is because otherwise the whole overview is gone. One way to show boundary rules is to number each procedure rule and listing all boundary rules with corresponding number. Noteworthy is that boundary rules and procedure rules can vary. Even in the same model. This depends on the modeler, like at which level of abstraction he modeled. Take the extreme figure 6.10. In that model almost all rules are boundary rules. I do not want to set a syntax or define a new model language. My goal is to provide a better understanding of rules, particularly to improve the WOz method. The way of modeling differs per modeler, for example the dotted block A in figure 6.11, "Choose if you want let the disk / draught on the board or take it away" is an action, while you also can say, "Do you want let the disk / draught on the board?" followed by an action. Also two questions can be in a block or the questions can be put in a separate block linked with an

arrow, like the dotted block B. This has the same effect. Alternatively, the right question of dotted block B can be seen as a boundary rule for the left question in dotted block B. Also forks and joins are can be used in a model, like the dotted block C. Again, in the context of this study is not important how this is exactly will be modeled, but the result and the clarity of this system that is obtained are here important.

Last, there are implicit rules. These are the unwritten rules. The implicit rules are self-evident and therefore not to be written, because this is infinite. For example, if you play a game with you friends, it is self-evident that you do not stay a long time away during that game. In addition, the question is where exactly the separation between explicit and implicit rules is. The wizard should always be trained to learn to to act like the system. Particularly the boundary rules are important here and it is also important to handle the implicit rules consistent. Procedure rules can act as guidance here, like going through a step by step menu. This step by step menu influences the cognitivity and the training process in a positive way. These step by step menus shall be used carefully and should not be too extensively seen from the perspective of the performance of the system. In addition, the implicit rules that are used by the wizard can be noted in the analysis and are taken into account in developing the functionalities of the system.

Finally, I would like something say about writing the rules. This can be done with guidelines called "Rule Speak" [5]. This is a useful set of guidelines that can be used to formulate business rules in a business-friendly and precisely manner. But I think it can also be used in formulating rules for a system. It is not a language or syntax, but rather a set of guidelines. And these guidelines seem useful to me in a team setting, so everyone interprets the rules in the same way. A final note is that in analysis of the game draughts, I wrote the rules in an atomic and point wise way. This helped me to make the rules more clear then in either of the other two cases, were I wrote multiple rules under one point.

To extract the rules of a system, it is important to know what your components are. The components can be used as a trick to find the rules. Game mechanics will be represented as the formulated rules which prescribe how components can be manipulated. For example in draughts, every arbitrary disk (of yourself) can be moved, if you follow the rules. So components can be useful to find the rules and game mechanics are the result of the rules applied to the components. I think that every game mechanic generates information. Sometimes this is information is useless, but mostly this information is valuable. Like the information about the participant in the topiary case or the location of a piece with draughts. This location (and the location of the other pieces) lets the other player decide what he will do in his turn.

But not all parts of the element information that Hoppenbrouwers gives is represented in cases. Information about agents is not present, Because the wizard

is not an agent, but in my setup it is a player. Also the participant is a player. Information about the game play has been a strong presence. These are in fact all the rules described above and need no further explanation here anymore. Information about the objects can have any value. For example in the case Topiary, where the location of objects and people the main core of the case are. The information that is needed for objects can be found in the ORM model. Information about the event I would like to discuss in the next section.

Besides decomposing games and cases, the WOz method itself can also be decomposed in terms of game elements. While I was analyzing this WOz method, I noticed an important thing. Although this method is generically written, it is important to analyze the specific part (case/game). Take a cook who controls all the operations for cooking (generic). He still has to think hard and pay attention to cook a particular recipe. In other words, the generic section is just generic. The focus is needed on the specific part (case / game).

Besides this finding, there is another finding to notice. In designing the test plan, the game element context plays a role. To be more precisely the physical location where it takes place (noise), the time, the background of the people are important. And also the guideline from Hevner (number three) plays an important role. By combining context with an appropriate test method, the system can be validated in a proper way. Another tool, which can be important is the game element information about the events. A value assigned to the operations. Here I explicitly use the word value and not the word score, because this can be associated with a win / lose ratio. I mean by a value a measure to check how consistent the wizard adheres to the prescribed rules. Also in this information, the value of the participant will be included and the person might give a rating to what extent the system did what they had expected. By combining these information flows, a clear picture can be created about the system. In chapter 8 this will be explored more.

Chapter 6

My case and improved WOz method

In this chapter, the theory discussed in this previous chapters comes together with a case which I invented. In this way, a test can be quickly established to validate if my findings are correct. And there may be a clear procedure, as I think it should be, to perform a correct WOz method. I am aware that there is no right order to do this. When I start with my case, I can adjust my procedure and when I start with my procedure, I can adjust my case. The only way to eliminate this problem is to treat multiple cases. Most likely in every following case that is dealt with my procedure will change slightly. Given the size of my project I am sticking with one case and I will try as good as possible taking this risk into account.

6.1 My case

The case is meant to test and where necessary change the obtained findings and is about developing a system that photographically shows, views and asks what the participant thinks about these views. In other words, it is a system that elicits what the user wants. This system can be applied to any domain. I applied it on characteristics of holiday destinations. By showing a number of these photographs in different sets like environments, climates, cultures and impressions, the system pretends to get an idea of what participants favorite destinations for holiday are. To tell the participant that the system is in development, I can explain that the performance are not optimal. In appendix C, a letter is shown which participants first read, before they do the test. Below is shown how the program looks for the participant.

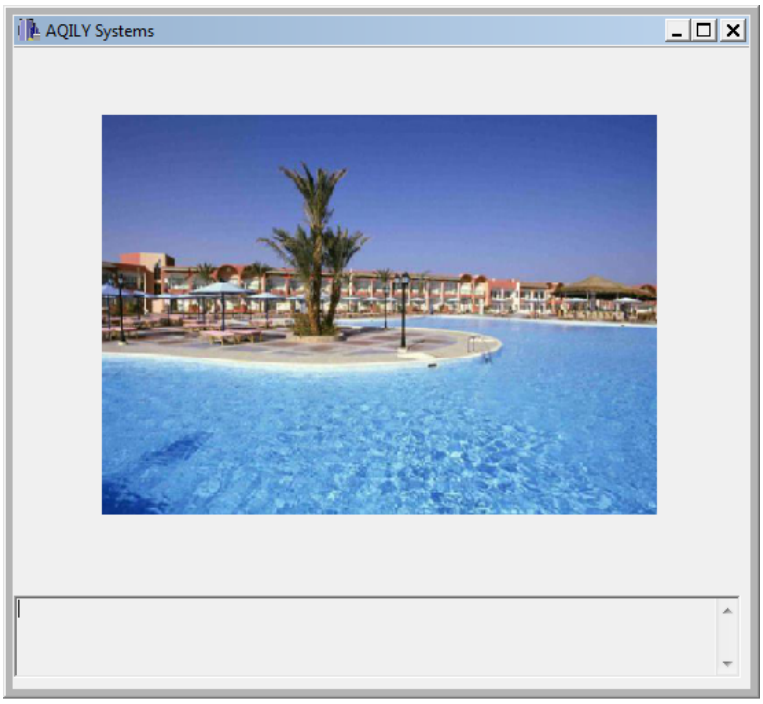


Figure 6.1: The participants GUI, where photographs will be shown and below some space for possible information for the participant.

As you can see, there are no buttons or other input components visible. This is because the interface of the participant has speech input. By looking at the photographs and thinking aloud about it, the system "gets an idea" what the participant prefers. In the next section, this case is explored with a suitable theory.

6.2 Applying improved WOz method on case

The theory has come together here to develop the case through the WOz method. This will be done by setting the theory as a procedure and filling it in with my case. In this way, I will attempt to described a theoretical yet comprehensible method. This procedure will purely focus to come from an idea (a case) to a tangible piece of software for the wizard. Obviously, this session will be recorded, but this will be explored in the next chapter. The order of the procedure below is fairly fixed, but some steps can be done in a different order.

1 Extract goal rule (aka long term goal) from case
Before starting with designing or modeling the case, the long term goal or goal

rule must be clear. Without this goal rule, there is no case. The goal rule for my case is: extract characteristics of the participant's favorite holiday destinations.

2. Design GUI of participant
Since the WOz method uses the requirements of the participant to validate or verify, the design of the GUI is not taken in the design of this method. Especially the GUI for the participant includes an instigator for testing by the WOz method and that method is not as designing a GUI for the participant. The GUI for the participant is shown in figure 7.1.

3. Create ORM scheme
Also there will be clarity about how the case looks like. This can be achieved by an ORM scheme. It defines the data structures of the program. I my opinion, players (participant and wizard) must be modeled. In a multi-setup wizard could even be modeled multiple wizards. The reason to model this, is that game mechanics arise indirectly of it. You could choose not to model the participant, because these game mechanics are not important in the design of the program for the wizard, but it seems to me a better idea if this is included in the model.

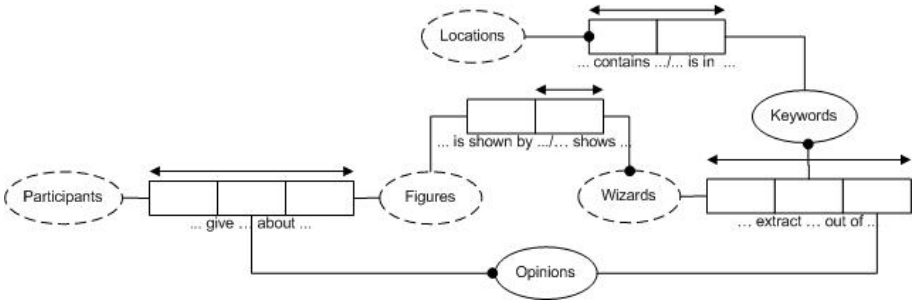


Figure 6.2: *ORM scheme of my case.*

4. Extract short and middle term goals
As indicated in section 4.1 the game mechanics are interactions between the players and components. The game mechanics has a goal. A goal to reach the final goal. Either the game mechanics are made for short and middle term goals to achieve the long term goal. The ORM diagram above, shows what these short and middle term goals are. Namely (1) "Participants give Opinions about Figures", (2) "Wizards extract Keywords out of Opinions" and (3) "Wizards show Figures". These goals are important when designing the program, especially the wizard's GUI. These goals can be seen as "What" exactly needs to happen. The way to extract these goals is looking in ORM scheme at any relationship in with the players. Often you find the short and middle term goals this way, but there is a possibility that you do not find them all. It will be wisely to look carefully. Looking reversibly is also possible, because there is a change that not all short

and middle term goals will emerge out of the ORM scheme. But I think with proper modeling almost all of these goals come forward.

5. Game mechanics
Now that the short and middle term goals are clear, we know the "What", but this is not enough. There is an another step to take to make it useful, namely the short and middle term goals must be transformed into game mechanics, the so-called "How". This "How" can be seen as an abstraction lower as figure 5.8 were interaction between two players is shown. Where player A makes a turn and then the other does. This continues until an end stage is reached. The game mechanics represent the short and middle term goals in an action itself. So a player that interact with components. In my case, the game mechanics are the three short and middle term goals above controlled by a player via components.

6. Components
The components that are needed to carry out the game mechanics, will be chosen by insight. For my case this would be a button for showing the (next) image for the goal "Wizards show Figures". For the other game mechanic "Wizards extract Keywords out of Opinions", a way to select keywords must be invented. I chose to use a button for every single keyword. The wizard can click on the button with the right keyword if he heard the wizard speak about this. He can click on the positive or negative button per keyword. As you can see, the designer must use some creativity and wisdom to get the right components.

7. Information
Not only components are necessary for the game mechanics. Every game mechanic gives a result. This can be anything, like establish something, but it also gives information. I believe that every game mechanic has at least information as a result. This information will always be as outlined in chapter 6 deal with events, agents, objects or game play. I do not distinguish this type of output, but for whom this output is relevant. This can be information for the system, information for the wizard or this can be information for the participant. Despite these information is always to the players or the system relevant, I can imagine that this information is not always useful for those. Therefore, there is a clear distinction between information that is useful and relevant. The information for the participant will not be taken into account in the method, because this has to do with the GUI of the participant. The information for the system is also not visible in the method. Only information for the wizard will be treated in this method. In this case there are two information streams for the wizard. First, if the wizard clicks on the next button, he will see the following picture. If the wizard clicks on the button with keywords, a counter will also indicate the state for that keyword. Just as with components, the designer must also use some creativity and wisdom.

8. Rules
The components and information are guided by rules. This will provide the wiz-

ard with clarity on how exactly the program should work. I use my approach as described in chapter 6. These rules are constructed in an iterative way. The following table shows the procedural rules. Below are the boundary rules. The implicit rules described in the theory will not be reported here because it will be done by the wizard. As indicated in the theory of my method there is no syntax or other support to establish to drawn rules. The initial rule is not displayed in my schedule, but this will be explained. Below are the rules applied to my case.

Procedure rules

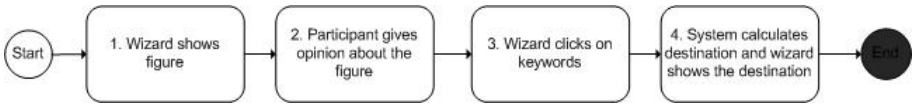


Figure 6.3: *Procedure rules.*

Boundary rules

#	Applied on procedure rule	Boundary rule
1	1 to 3	Procedure rule 1 till 3 is a cycle.
2	1	Wizard waits until participant has finished speaking.
3	2	Participant should think aloud.
4	3	Wizard should select keywords in a logical way.

Table 6.1: Boundary rules.

Initial rules
Wizard shows first figure.

9. Design GUI of wizard
Now everything what should be used for the GUI is clear, this can be designed. This is namely the components and the element information that are obtained in step 6 and 7 of this plan. Also these steps indicated that creativity and wisdom in placing the elements in the GUI comes to play. However, there one can work according to the game principles that are described in section 4.2. In addition, the interface (in the definition of the game elements) also helps determine the GUI. Still, I think this is usually the mouse and keyboard wizard that will be used. Below is shown how the program looks for the wizard.

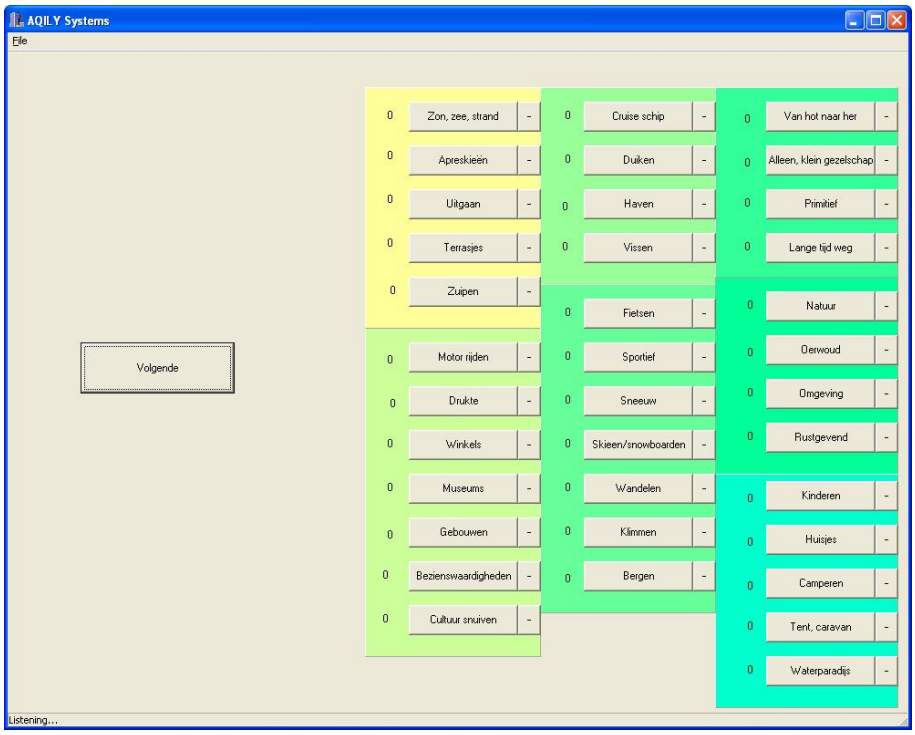


Figure 6.4: *GUI for the wizard. Here he can control the system. Clicking on a keyword means that the participant rated the keyword positive. Clicking on the minus means that the participant rated the keyword negative.*

10. Scenarios

Also scenarios can be used. These scenarios are especially useful for testing a large GUI. I do not want to treat how to get to these scenarios. This will usually be emerge in the process of gathering the requirements. If these scenarios are clear, a better structure for the wizard can be created. The wizard will know about what the participant will do. This will be less cognitively exhausting. Also the unnecessary components and information for that scenario can be removed from the GUI. I think it is useful to always let the components and information at the same place, and only make them visible or invisible, otherwise the wizard can be confused where to look. Given the size of my case, this step will not be applied.

11. Train the wizard

The system is now completely finished and ready for testing. First, the wizard trains with the system and also tests if it works well. Maybe, the system could have some small adjustments to fit everything. It depends on the case and wizard how many times there will be trained. In the case that I use, the wizard

has trained once.

12. Ready to use
After these iterative steps are completed, the system is ready to test. Since this is an important aspect to improve the system and method, a new chapter will be dedicated to this. The next chapter will also explore results of the test.

Chapter 7

Method

The new WOz method can be examined from two perspectives. First, we can look at the process of designing. With this it can be tested if designers understand and properly implement the new WOz method. Another test can be examined from the wizard's expectations.

Because of the size of my project, I will unfortunately not be able to study the process of designing, because it is simply too big for my project. In short this test will go as follows. People get some information about the new WOz method and afterward they are asked some questions about it. Then these people will be assigned to groups and will discuss a case. Before they discuss this, they are instructed to think aloud and they must tell why they come to that result. Through this video transcript and application of a context analysis to these records, insights can be obtained on how this method would be used. This new information can be used as new input to improve the WOz method in a second iteration.

The other test I would like to elaborate on in my research. Again, the size of this test is small, because of the size of my project. Thus, multiple wizards, participants and cases give a clearer picture of it and the WOz method can be improved on a more valid way. In my test, 1 wizard and 1 case will be tested. Also, 1 participant participates in the test.

To test this, first it will be necessary to address what exactly needs to be tested. The WOz method will be tested by a specific case. The case is considered like the perspective illustrated below.

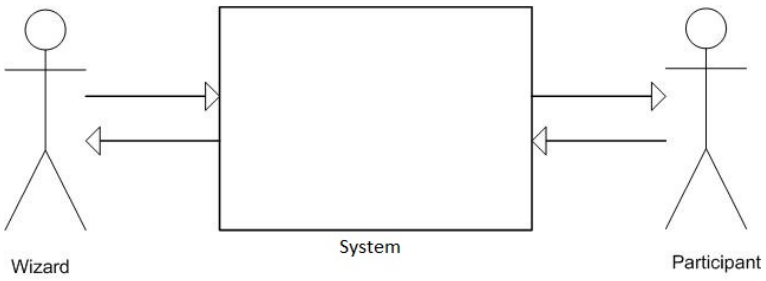


Figure 7.1: *Communication between wizard and participant in a formal way.*

Figure 7.1 can be seen as communication between the wizard and participant in a formal way, although the participant does not know that he communicates with the wizard. This formal communication is through the designed program and the intention is that this way the wizard and the participant must understand each other via the in- and output of the program.

First, a conceptual model will be shown below, to outline what the indicators are.

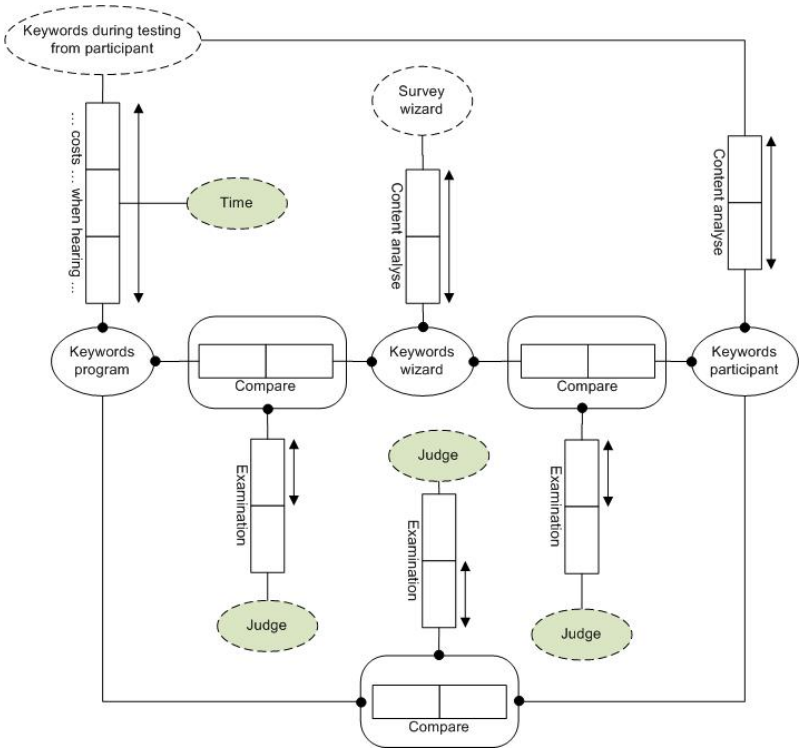


Figure 7.2: *Conceptual model, with the green value circles as indicators.*

This conceptual model above, will obtain the indicators in the following way:

1. First, the participant and wizard are instructed separate of each other. The wizard will only be told to click on keywords when the participant calls out these words . And the keywords the wizard chooses, must make sense. For the rest he can do what he wants and has total freedom. Before the test starts, the wizard has already been able to practice in the training to get a grip on how it works. The participant only gets a letter with an introduction before the test begins. This can be found in Appendix C.

2. The test will be executed like described in chapter 6. The wizard shows the first image and the participant gives his opinion about the image. Meanwhile, the wizard clicks the keywords that are applicable. The participant says: "next" if a new image must appear and the wizard will show the new image.

3. After the test, the wizard may answer an open survey question, where he can answer what the participant likes or what he thinks is important for a holiday destination for the participant.

4. Then the responses of the participant and wizard will be compared. This will be done by content analysis and counting the keywords. The keywords here are the nouns and verbs that are rating the shown figure. This comparison allows to see if the wizard has understood the participant. Also the data of that is entered in the system and will be compared with the data of the wizard. This shows if the wizard has chosen the right keywords. Finally, the participant data will be compared with the data that is entered in the system that characterized the requirements for a holiday destination. In addition, the time between the participant saying a keyword and the wizard clicking on that keyword will be measured. With this information, training effects can be traced and it can be seen if the wizard gets fatigued during the test.

In addition to this plan, the wizard will also be monitored. Through this it can be determined if the wizard thinks that the system is working properly. There will be a focus on the information that goes from the wizard to the system and vice versa. This can be done by using a camera and a think aloud protocol. I think in this case it is not practical to think aloud during the test, because the wizard must listen to the participant and it is difficult for him to think aloud at the same moment. I want resolve this to watching the video directly after the test with the wizard and do a retro perspective think aloud protocol. This is done immediately after wards, so everything is still fresh in his memory. I will act as facilitator and try to trigger the wizard to say what he thought at that time.

Chapter 8

Results

In this section the results of the test are shown. In appendix D the analyzed data is displayed. There you also can find the raw data and it is therefore not necessary to make a separate appendix for it. The comparisons between participant, wizard and system data are made. And the response times of the wizard have been reviewed. This process will be shown as transparent as possible to one hand by saying why something is being done and on the other hand by showing the entire data that can be found in Appendix D.

8.1 Comparing participant data with wizard data

First, it will be checked whether the wizard can tell what the participant would like. If this is not, the system will never have the good answers. This first comparison is used to verify whether the wizard properly understood the participant. In the survey question asking what the participant likes and what he does not like, the wizard answered:

*"Geen **kinderen**! Hij houdt van **uitgaan**. **Primitieve vakanties/backpacken**, in een **mooie omgeving** voornamelijk **oerwoud** vindt hij leuk. En hij zou ook graag naar **steden** willen, maar daar heeft hij een mindere voorkeur voor. Hij zou ooit wel eens een **cruise** willen doen. En hij vindt **camperen** heel leuk! En dan het liefst zo **primitief** mogelijk. Houdt niet van **sportieve vakanties**."*

Below is a table with the nouns and verbs that are positive or negative rated.

words	**emotion**
kinderen	-
uitgaan	+
primitieve vakanties / backpacken	+
mooie omgeving (oerwoud)	+
steden	+
cruiseschip	+
camperen (liefst primitief)	+
sportieve vakanties	-

Table 8.1: *Positives and negatives nouns and verbs of the answer from the wizard about the participant.*

First thing to notice is that the words in this table are almost the same as the words used in the system. This so-called priming effect is causes the wizard has been working with these words all the time. Below is the graph of keywords that the participant responded with. These are decomposed in the same way with the response of the wizard in keywords. The graph shows the sum of a keyword. In other words, there has been looked at how many times each keyword is rated positive and negative and the number of negatives is deducted of the number of positives.

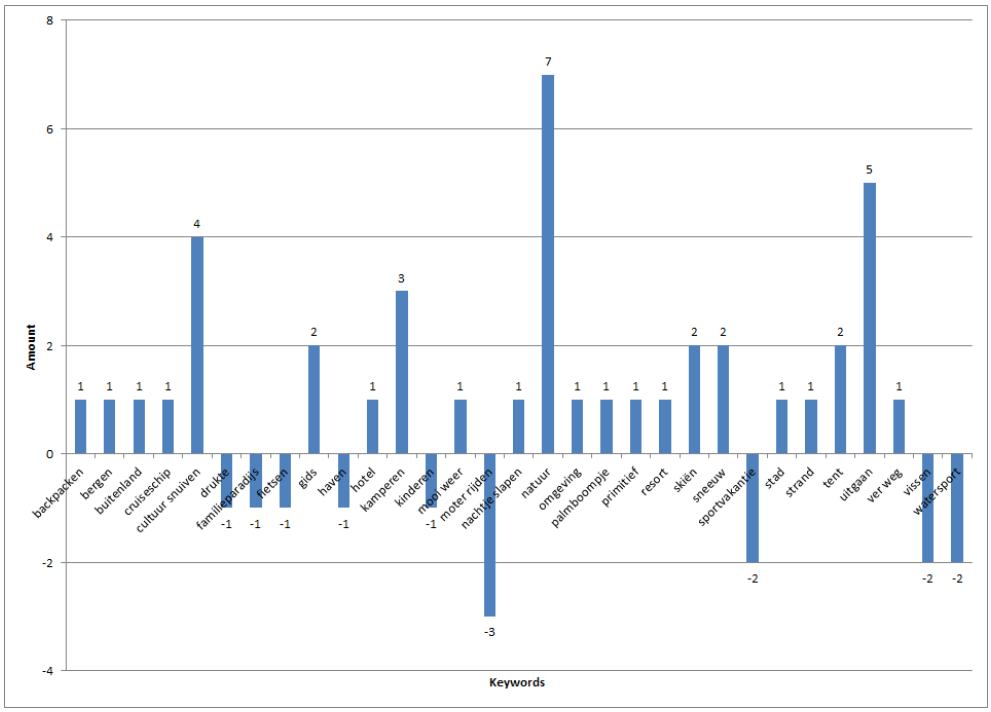


Figure 8.1: *Data of what participant said.*

Because the participant said quite a lot of different keywords, I tried to categorize these in groups. As an example I categorized mountain biking and cycling. This is done with multiple keywords. Because many interpretations can be used, I tried to do this as transparent as possible. In appendix E, there is an overview of the words that are grouped. The groups I made, are based on the context of the participant. I tried to ignore the wizards comment, because I do not want the view of the wizard being involved in this section, but purely look as an analyst and researcher to this data. Below the extreme values of the graph are compared with the table 8.1 extracted from the response of the wizard.

Kids (kinderen)
The wizard begins her answer with, "no children", while the chart does not clearly suggests that the participant does not want that. The reason is that he does not actually use the word children (or child) as if he saw a picture of it. The focus of the pictures were children so that the wizard got the idea that the participant rated children negative.

Clubbing (uitgaan)
The wizard also indicated that the participant would like a destination with

clubbing options. This is clearly shown in the graph 8.3.

Primitive holidays / backpacking (primitieve vakanties / backpacken)
Primitive holidays / backpacking, according to the chart is not often positively appointed by the participant, or the wizard has often clicked on it. The wizard said in the conversation that she often based on the image that these were primitive and frequently got the feeling that the participant thought it most be done in a primitive way. For example, the wizard clicked on primitive when the participant used the word backpacking.

Beautiful environment / jungle (mooie omgeving / oerwoud)
Like clubbing, the answer of the wizard matches nicely with the chart of 8.1. As you can seen in Appendix E, I grouped jungle with nature.

Cities (steden)
Some pictures of towns have been showed, but the keyword is in certain contexts positive and negative rated. This makes the result a value of 1 for cities.

Cruise ship (cruiseschip)
Noteworthy, only 1 picture of a cruise ship is be displayed. It is not a extreme value but the wizard gave it in his answer. Perhaps because it appears once and the participant is just enthusiastic here.

Camping (kamperen)
Also camping is correct and is called more than the most other keywords.

Sport holidays (sportieve vakanties)
The wizard also indicated that the participant would prefer not to do sport vacations. And this is generally true, but for example skiing is a sport and scores positive. But for example with the keywords water sports, cycling and sports holiday you come to a fairly negative response from the participant for sporting holidays.

If we look at the chart, and we look at the points on the top and bottom-stabbing and which have not yet been dealt with (because they are not in the response of the wizard), then we see that culture sniffing has not been noticed by the wizard, which the participant four times positively assessed. This is because it has been rated positive 3 times at 1 image. In addition, motorcycling is negatively rated but was rated negative twice at 1 image by the participant.

8.2 Comparing wizard data with system data

We have seen that the wizard is reasonably able to find the right keywords compared with the responses of the participant. Some words (such as motorcycling or sniffing culture) were not entirely correct, but there was a valid reason for this. This section examines whether the wizard has passed the correct keywords to the system. For this, we will look to table 8.1 and the chart below.

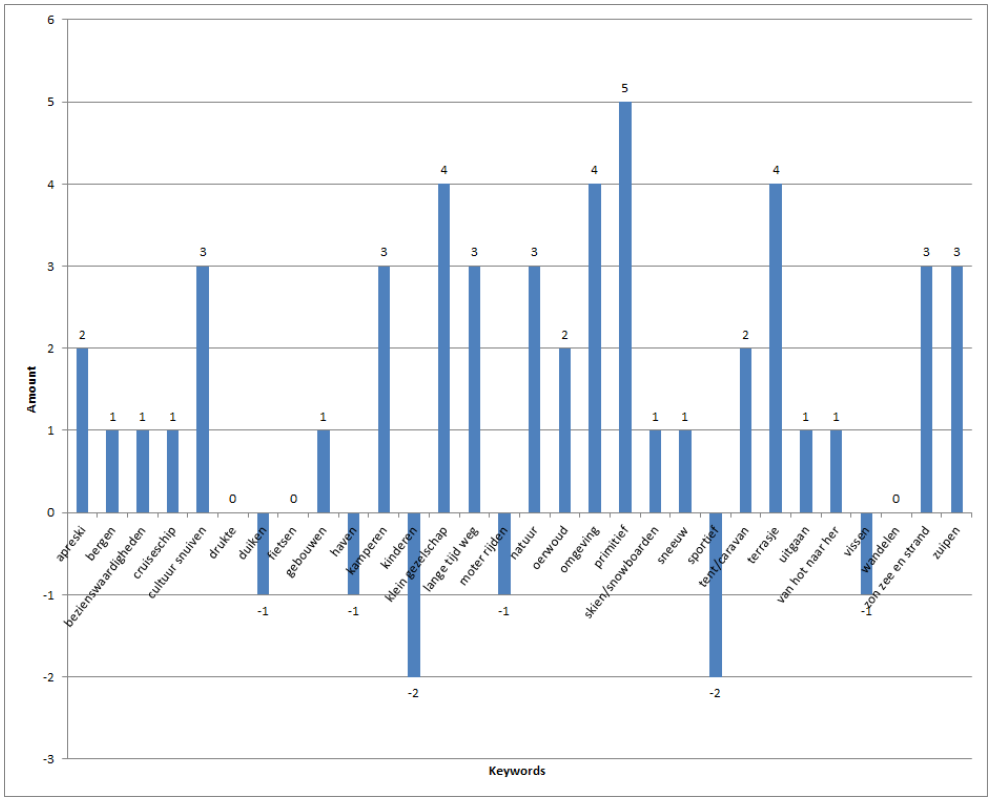


Figure 8.2: *Data of system.*

The chart above shows that many comments as given in the previous subsection can be given here. Below the table or extreme values of the chart will be treated.

Kids (kinderen)
In this graph you can see that children are rated negative. This is actually the same as indicated in the previous section, namely, the wizard saw the images which the participant was (negative) talking about, without making mention exactly where he talked about.

Clubbing (uitgaan)
Nightlife is not very highly rated in the graph of 8.4, but as terrace and booze will be counted, then it may be correct.

Primitive holidays / backpacking (primitieve vakanties / backpacken)
Primitive is the highest ranked in the chart, so true. Especially if we add backpacking to it

Beautiful environment / jungle (mooie omgeving / oerwoud)
Same as primitive holidays / backpacking is a beautiful environment rated very well. Jungle is also fairly positive rated.

Cities (steden)
Cities is not shown in the graph. This is because there was no way in the system to choose for this keyword. Still, the wizard does know that the participant find this important and this can also be traced back to the analysis (Appendix D). Also, the wizard said in the interview that she tried to approximate city by clicking on "buildings" or "busy".

Cruise ship (cruiseschip)
For cruise ship is the same as in the previous subsection.

Camping (kamperen)
Camping, preferably primitive, is also correct. Primitive is the most positive rated in chart 8.4 and "tent / caravan" also highly appreciated.

Sport holidays (sportieve vakanties)
Also here is the same as in the previous subsection. If you add all keywords in sports together, the participant does indeed not like a sporting holiday. This is also highlighted by the keyword "sportief" with the value -2.

The wizard did not give in her answer that the participant prefers to go on a holiday with a small group. Yet in Chart 8.4, you can see that this keyword is highly valued, namely 4. This is because the wizard has clicked on small group if she heard something about "primitive / backpacking". This is also reflected in her answer (Appendix D).

8.3 Comparing participant data with system data

Actually, this comparison determines whether the system works or fails and the two previous sections have shown that the system works reasonably well, but there are some minor points to improve. First, the figures of 8.3 and 8.4 will be compared. That is what the participant said and what is clicked by the wizard. I preferred that the reader gets the opportunity to compare the data. Therefore it is decided to not merge these two graphs together, because

a number of keywords should have been transformed in order to achieve this. I have made a number of groups, because otherwise there were too many different keywords. These groups can be found in Appendix E.

If you compare the two graphs (figure 8.3 and figure 8.4), you can see that some keywords match well. Below you can find certain keywords that match well.

- **Culture sniffing** (cultuur snuiven)
- **Camping** (camperen)

There are also some keywords that fit well, but with some explanation.

Nature / jungle (natuur / oerwoud)
If you look at nature in figure 8.3 and look at nature, jungle and environment in figure 8.4, you can see that this occurs often

Motorcycling (moter rijden)
Motorcycling occurs in figure 8.3 three times negatively and in figure 8.4 just 1 time. The reason for this is that at one image, the word motorcycling occurred three times in one phrase and the wizard clicked once for this phrase.

Clubbing (uitgaan)
In figure 8.3 clubbing occurs often. In figure 8.4 is the same as what is stated in the previous section, namely that with terrace and booze counted also, it may be correct.

8.4 Time data

It has also been examined whether there is anything to say about the time that the wizard clicked on a keyword button. This is done by measuring the time between when the participant says a keyword and when the wizard finished entering the keywords accordingly, rounded off to seconds. Keywords that the participant said, but were not processed by the wizard, are disregarded. The chart below shows how the test in terms of time has elapsed.

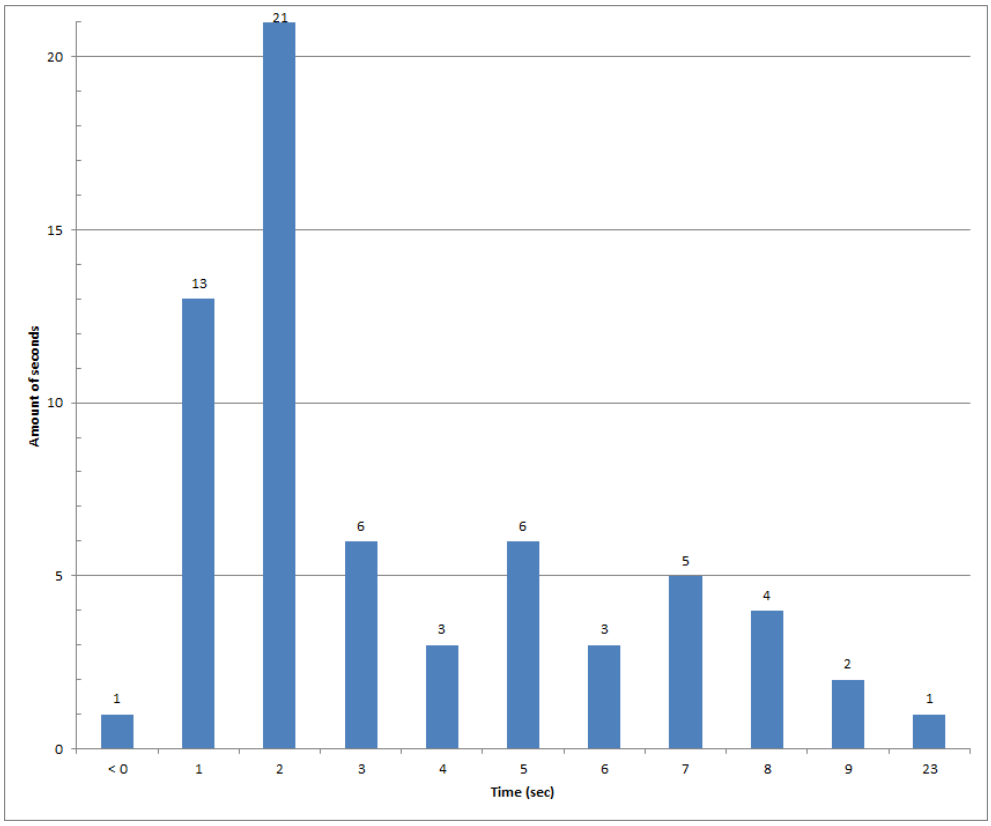


Figure 8.3: *Frequency of times that wizard clicked.*

Above, you can see the frequency that it took to click. Notice that one time that the participant already had clicked before he heard the word. In the retro perspective think aloud the wizard said, that she had an impression of what the participant would say and clicked before the participant spoke. Furthermore, you can see that most words are processed by the wizard within three seconds, namely 56 percent. And under five seconds is even 70 percent. There is no association between words that come more frequently and rapidly clicked. This is totally on random basis. There is also a time of 23 seconds, because the wizard saw and clicked on a keyword from an image before.

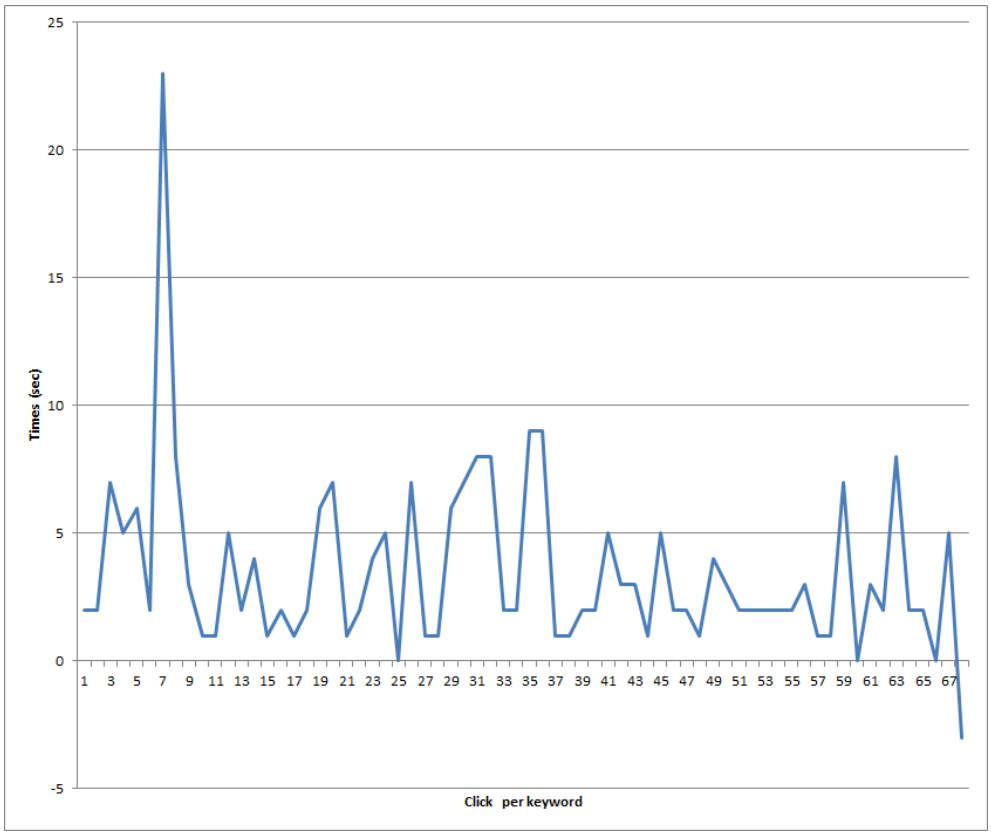


Figure 8.4: *Time over clicks.*

Above is the length of time per click. The peak is the click whereby the keyword was found 23 seconds later. In addition, there is no training effect to see or that the wizard was fatigued. Finally I would like to say that this time information is not as interesting with one measurement, but if several measurements (processed with improvements) are made. These points can be compared and one can examine if there are positive effects on the time.

8.5 Discussion

You can ask yourself, is it fair to show some images about a topic more than an another topic, for example camping versus cruise ship. The answer is that in a realistic setting, this is not a fair way of testing, because more will be told about the subject that is more represented which leads to more extreme values of that topic. This could be solved by applying a normalization. But that is not the scope of this project.

In the section above, we could have focused on false negatives and false positives. To do nothing with these results was a deliberate choice. These false positives and negatives can both have a further distinction. False positives may be considered by the impression of an image or the wizard clicked when the participant said something about the image without saying the word literally. Or perhaps there is no reason that the wizards clicked on a keyword.

A distinction can also be made with false negatives. Some keywords that are not mentioned literally by the participant, are not clicked on by the wizard. And other keywords are used repeatedly in 1 phrase and clicked once, but is this a false negative? Or the participant use derivations of keywords and these were not noticed by the wizard. Also, it happens too often that the participant is saying words that the wizard cannot click on the in GUI, for example city where the wizard is forced to do this with other words. All these variations and uncertainties have led to the advice not to use these results, because they are unrepresentative and biased to this test session. Precisely these variations and uncertainties are good input to the case and the methodology.

Above are a couple of improvements already mentioned which I will explain here. To begin with the chosen keywords. A good example was the keyword cities. These are some frequently mentioned by the participant, but the wizard could not click on it and choose keywords that best fitted. The keyword "zon zee en strand" is related on too many keywords. For these cases, the input of the participant could improve the keywords that will be used for the system. This way keywords in the system can be removed or can be added. For this improvement more participants are needed and by using content analysis (word count), a good decision can be made. Another important improvement is rules. For the training and the test began, the wizard is told they only press for keywords on the GUI if the participant say something about it, as long it makes sense. This keywords could literally be called or look like, for example car and vehicle. At the training no recording took place and no data was available about it. This training was only for the wizard so that it was clear for him where all buttons were. The analysis of the actual test shows that the wizard sometimes used his own impressions, if she saw the image. A good instruction for the case is to forbid clicking on own impressions. Sometimes the participant also talked about the picture, without saying where he talked about. This was probably understood by the wizard, but at the content analysis it was not clear and gave some strange results. This could be a point of improvement, and would again be explicitly stated that the intention is that this should be done. Another uncertainty that could be captured is what to do with a keyword that occurs two times in 1 phrase or what to do with derivations. These uncertainties shows why the WOz method is suitable for this type of cases so that these little changes can be changed quickly and the system does exactly what the customer wants.

Another conclusion from the interview with the wizard is that she thinks that the participant said "next" too fast and she had no time to find and click on all keywords. On the one hand she did not want to let the participant wait. Finally,the wizard preferred more training before the test session.

Beyond these improvements, the wizard also said good things about the test program. First, she thinks that the program is very easy to use and she found it very nice that she saw the pictures that the participant also saw. It was also nice that it was clear what the value of a keyword was. Even when it was clicked, the GUI showed the keyword that was clicked, so she did not start to doubt whether she had pressed on the keyword button. Besides this positive feedback from the wizard, there are also positive quantitative results. For example, more than a half of the keywords was clicked within three seconds and almost three quarters within five seconds. For the first time in my opinion, these are already well-conceived keywords. The wizard had all emotions well chosen. The fact that these things are positive in the case, also means that the methodology has a good basis. In the next chapter, this will be even better when the improvement from this section have emerged in the processing method.

Chapter 9

Second iteration: improving WOz method

As is shown in the previous section, there are a number of improvements already mentioned. All these improvements are specific to the case, for example the adaptation of certain rules and add or delete some words in the wizard's GUI. In this section I will discuss further improvements of the methodology itself. Starting with a remark of the previous section, namely that the wizard must too quickly press on next. As indicated in subsection 4.3.2, a characteristics of the WOz method is that the "system" may respond slowly. You could instruct the participant to mentioned this, but not in general terms such as "the system can react slowly, but more specifically in the use case: "When you say next, the next picture can appear with a delay. The system is processing the answers and this takes a few seconds.". Another solution could be that the wizard presses on a key when he heard "next" and the participant get the message: "one moment, please". The wizard has the time to click all keywords he heard and then shows the next image. On one hand you show the participant that the system responds, but needs processing time. On the other hand this costs the wizard more cognitive power.

The main thing that has emerged out of this test is the realization that training is more then introducing the wizard with the GUI. The results show that not only a feeling with the GUI is important for the wizard, but also practicing with the rules. Of course not all rules, because many rules provided are for the characteristics of the program and the customer decides how the program must behave. An example of rules that can be trained is that the wizard is not allowed to click on a keyword by his own understanding or impression of the image he sees. Also, from the results it becomes clear that the participant sometimes did not literally say he gave an assessment. This would be in the instructions of the participant to come. Because this requires a reasonably comprehensive analysis, I want to add a step in the plan after the training step. The training

will not change, but a pilot test will be placed to come after the training. Data be obtained in two ways. The first is to observe the wizard and his actions. The second is interviewing the wizard and asking him what he wants to improve and where he has doubts. On this way, the wizard can practice a couple of extra times and with his input the system can be improved. Through this method, no comprehensive analysis has to be done to get useful information.

Finally, the wizard said in the interview that some keywords did not have a logical location in the GUI. In step 9 of the improved WOz method it is stated that creativity and wisdom must be used. I think it is important to make a GUI with a GUI designer. Because the GUI in itself is already an issue. Since GUI design is a topic itself, I do not want to discuss this in my plan. A less extreme solution, is that the wizard contributes to the GUI design.

There are three factors in this design which can improve the WOz method. I think there is more to improve, but it is not clear what else can be improved. This could be reached by testing multiple times with multiple cases, participants and wizards.

Chapter 10

Conclusion

This thesis is about improving the WOz method. The study investigated if GDT could improve the WOz method. After the theoretical study, existing cases were decomposed into game elements. This analysis showed that the two cases had a fairly generic structure. Then I made a plan for the WOz method. There already was a plan, but it was limited. By combining the previous findings with the existing plan, an attempt has been made to draw a fairly generic improved plan. After the plan was created, an invented case was tested. A case where the wizard should find out what the participant's favorite characteristics of a holiday destination are. This test gave some very interesting results. On one hand this confirmed my plan worked and on the other hand it gave new input for further research and improvement.

Before we can answer the main question, the the sub-questions should be answered. Below are the answers to these sub-questions:

Can GDT ensure that the WOz method is more generic, so that this method becomes easier to apply on an arbitrary IT project?
I started to decompose the two WOz cases with GDT. At many points these cases let themselves be decomposed by game elements. It even appeared that there were many similarities between them. If these cases can be decomposed, it would plausible if the reverse is also true, namely a case built up by game elements. This was true and many game elements let themselves be captured in the process of creating the WOz plan. Because it was possible by multiple cases, it could be cast into a method and this method is actually a generalization of these cases. Since there is only one case tested once with the new WOz method, it is not fully validated that the method is generic, but this is was test with an arbitrary case, from this test it can be stated that for now, the method is generic and I dare to say that the method is generic enough for it to be used in a wide variety of IT projects.

Can GDT contribute to a more structured approach of the WOz method?
The new WOz method is developed from a plan that is widely used in this field, but more as guidelines then as rules. Because this plan is combined with game elements extracted from decomposed WOz cases, I think there is solid basis to execute a WOz case. If the steps are executed in the right order, a good result can be achieved. The results of the test showed that the new WOz method is quite good. I think that this plan offers a good supported and more structured approach to execute the WOz method.

Can GDT eliminate or reduce the disadvantages which belong to the method?
To answer this question, first the disadvantages will be mentioned.

- The wizard requires a training. This means there are extra costs to the project.
- It's hard for the wizard to respond adequately and consistently, because the role of wizard requires a lot of the cognitive ability. Especially if the test takes a long time or the wizard must react on a lot of sensors.
- It's difficult to test a large GUI, because there are a lot of variables which require attention.

First, these disadvantages can never be completely eliminated by GDT or other resources, but these disadvantages can be reduced by the new WOz method. Game mechanics extracting the right input and output. A plethora of parameters might have been avoided what training time will decrease. I think that in the future a way should be explored when it is financially attractive to use the WOz method or test it on another way. The cost for the training is one of these variables. But more about this in the next section.

The last two disadvantages can be reduced by using scenarios. In this way, a timeline will be through the program and the wizard knows what will come and this decreases the cognitive load. Scenarios can also make parts of the GUI invisible at times when they are not needed. The exact effect of this is untested, but I think it should be a good way to hide unnecessary variables when they are not needed. Another solution to these problems is using a multi wizard setup. Due to the size of my project, I have not examined this. But it seems to me that it requires less cognitive power and the wizards can be more consistent and adequate in response, but this should be conducted in future research.

Now all sub-questions are answered the main question can be answered:

Can the WOz method be improved by applying GDT?

The WOz method can can be provided a more generic structure, by applying GDT on this. Also disadvantages of this method can be reduced by applying GDT. I think with this research a proper WOz method has been developed, but I also think that with more research a number of points can be improved. In the next section will examine this in more depth.

10.1 Future work

In my opinion, the WOz method is highly improved. Still, I think in the future a number of things can be explored to further enhance this method. Firstly, to testing my new WOz method with more cases, more participants and more wizards, could improve and refine the WOz method. The design as described in chapter 7 could also be tested.

Secondly, there could also be experimented with a multi wizard setup. This can be done in several ways, for example two wizard who each take care for a part of the GUI. Or multiple wizards with each a different role, like one wizard reads the input, a second wizard decides and the third wizard sends the output to the participant. There can also be examined what kind of protocol is needed for multiple wizards. An another question you can asked yourself is, is it useful to using a multi wizard setup or using just one wizard and when is it useful to use one wizard or a multiple wizard setup.

Finally there could be investigated if it is cost effective to use the WOz method or use another test method. Or maybe to program a part of the system that needs to be tested and use partly the WOz method. When is a function too expensive for the WOz method? I think if it is easy to program and there is no inconsistency in the implementation, it seems wiser to program, but if not, I think the WOz method is a good way to test it. The ratio when to program or use the WOz method would be interesting to examine.

10.2 Future computer program

When more research is done about this method, maybe a program can be created that will guide you through the whole process. How this will looks like exactly, is unknown, but I can give a number of features, which I think are important to in this program.

I want the part of design an ORM scheme ignore, because there are already a lot of programs with this functionality. The only thing that conventional ORM tools do not have, is the feature to mark or even recognize the game mechanics, but it could also be printed and can be shaded. The game mechanics shows the in- and output of the system, which are important in making the GUI for the wizard. One tool for the GUI to make for the wizard, will certainly belong in this program. By using a specific timeline in the system, scenarios could be used, which makes GUI elements of the wizard temporary invisible or visible. In addition, a good protocol is needed to enable the creation of network communication with the system. Also, the communication protocols between various wizards could be elaborated in this program.

Recording capabilities should also be a feature of the program. When recording the data, a good way of data management is important. Another important aspect is that some programs maybe have functionalities that already are programmed and other functionalities that needs to be tested by using the WOz method. One way to address this is to write a library and include this in your program that need to be tested. This library handles the communication with the future program. The disadvantage of this approach is that a library for each language needs to be written, but I think if there is a library for the five most common languages, a big part of the systems is covered.

Bibliography

[1] Suede documentation. http://dub.washington.edu:2007/projects/suede/docs/tutorial/, 2001.

[2] Business rules slides - games for modelling, 2009.

[3] In search of the why: developing a system for answering why-questions. http://www.narcis.info/research/RecordID/OND1331415/Language/en, 2010.

[4] Wizard of oz. www.usabilitybok.org/methods/wizard-of-oz, 2010.

[5] Rule speak. http://www.rulespeak.com, 2011.

[6] Wikipedia, nl, dammen. http://nl.wikipedia.org/wiki/Dammen, 2011.

[7] B. Bates and R.A. Bates. *Game design*. Course Technology Press Boston, MA, United States, 2004.

[8] N. Dahlback, A. Jonsson, and L. Ahrenberg. Wizard of oz studies–why and how. *Knowledge-Based Systems*, 6(4):258–266, 1993.

[9] S. Dow, J. Lee, C. Oezbek, B. MacIntyre, J.D. Bolter, and M. Gandy. Wizard of oz interfaces for mixed reality applications. In *CHI'05 extended abstracts on Human factors in computing systems*, pages 1339–1342. ACM, 2005.

[10] S. Dow, B. MacIntyre, J. Lee, C. Oezbek, JD Bolter, and M. Gandy. Wizard of oz support throughout an iterative design process. *IEEE Pervasive Computing*, 4(4):18–26, 2005.

[11] J.M. Francony, E. Kuijpers, and Y. Polity. Towards a methodology for wizard of oz experiments. In *Third Conference on Applied Natural Language Processing*, 1992.

[12] D. Ghozland. Designing for motivation. *Gamasutra, June 7th. DOI= http://www. gamasutra. com/view/feature/1419/designing_ for_motivation. php*, 2007.

[13] A.R. Hevner, S.T. March, J. Park, and S. Ram. Design science in information systems research. *Mis Quarterly*, pages 75–105, 2004.

[14] S. Hoppenbrouwers, P. van Bommel, and A. Jarvinen. Method engineering as game design-an emerging hco perspective on methods and case tools. In *workshop proceedings of EMMSAD08: Exploring Modeling Methods for Systems Analysis and Design affiliated to CAiSE08, Montpellier, France*. Citeseer, 2008.

[15] J. Huizinga. Homo ludens: A study of the play-element in culture. 1950. *Reprint. Bos-ton: Beacon*, 1955.

[16] A. Jarvinen. Games without frontiers: Theories and methods for game studies and design. *Acta electronica Universitatis Tamperensis. Tampere University Press, Tampere*, 2008.

[17] D. Johnson and J. Wiles. Effective affective user interface design in games. *Ergonomics*, 46(13):1332–1345, 2003.

[18] Y. Li, J.I. Hong, and J.A. Landay. Topiary: a tool for prototyping location-enhanced applications. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 217–226. ACM, 2004.

[19] S.T. March and G.F. Smith. Design and natural science research on information technology. *Decision Support Systems*, 15(4):251–266, 1995.

[20] M.L. Markus and A. Majchrzak. A design theory for systems that support emergent knowledge processes. *Mis Quarterly*, 26(3):179–212, 2002.

[21] R.J. Pagulayan, K. Keeker, D. Wixon, R.L. Romero, and T. Fuller. User-centered design in games. *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications, Lawrence Erlbaum Associates, Inc., Mahwah, NJ*, 2002.

[22] J. Rose. Fewer Mechanics, Better Game, 2008.

[23] R.M. Ryan, C.S. Rigby, and A. Przybylski. The motivational pull of video games: A self-determination theory approach. *Motivation and Emotion*, 30(4):344–360, 2006.

[24] D. Salber and J. Coutaz. Applying the wizard of oz technique to the study of multimodal systems. *Human-Computer Interaction*, 753, 1993.

[25] K. Salen and E. Zimmerman. *Rules of play: Game design fundamentals*. MIT Press, 2004.

[26] H.A. Simon. *The sciences of the artificial*. The MIT Press, 1996.

[27] P. Sweetser and P. Wyeth. GameFlow: a model for evaluating player enjoyment in games. *Computers in Entertainment (CIE)*, 3(3):3.

[28] J.G. Walls, G.R. Widmeyer, and O.A. El Sawy. Building an information system design theory for vigilant eis. *Information Systems Research*, 3(1):36–59, 1992.

[29] I Wilmont. A gaming approach to collaborative modelling. Master's thesis, Radboud University Nijmegen, 2009.

[30] B. Zmud. Editor's comments volume 21 iss. 3. *Management Information Systems Quarterly*, 21(3):1, 1997.

Appendix A

SUEDE documentation

SUEDE is a system that helps speech user interface designers quickly create, test, and analyze speech user interface prototypes. SUEDE couples a simple prompt/response card model with the Wizard of Oz technique.

This user's guide leads you through the basics of designing, testing, and analyzing a speech user interface design in SUEDE. There are also exercises to guide you through the creation of an example speech user interface.

Terminology

The user is the person who actually uses the interface. This is the person who hears the prompts and responds accordingly. The Wizard is the person working in the background. He or she is the one who runs SUEDE when testing an interface. When designing is SUEDE, you manipulate "cards". There are four different types:

Figure A.1: *The start card is included automatically in every SUEDE document. This card is used to tell SUEDE where the speech design begins and also serves as a linking point for the start of the interface. You can move the start card by right-clicking on it and dragging it around. Release the mouse button to put it down.*

Figure A.2: *An orange prompt card is used to indicate a system prompt, something the computer says to the user. You can add a prompt card to your design by either clicking your right mouse button in the canvas area or by dragging a card from the script bar. (The script bar is covered in detail in section 2.) Prompt cards are moved in the same way as the start card: by right-clicking over it and dragging it around.*

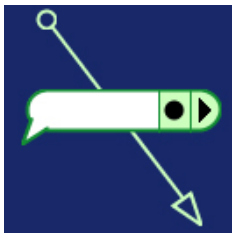


Figure A.3: *A green response card represents user input, what the user can say to the system. To set up a response card, left-click and drag your work area. A left mouse drag gesture between two prompt cards creates a link while dragging from the background canvas creates a global (user commands that can be spoken at any time, regardless of the prompt). As with prompt cards, response cards can be dragged from the script bar. Note: Response cards don't need to link. A response card with no finish will have a circle around its arrow, as in the diagram. To link such a card, left-drag the arrow over the prompt you'd like to follow the response. The same method is used to reassign a response card's link.*

Figure A.4: *A blue group card is a set of prompt cards that can all be alternative replies after a specific user response. A flick gesture on the canvas area creates a group and a right mouse drag will allow you to move it.*

Elements of the SUEDE Environment

We will now illustrate how to design a speech user interface with a specific example. Suppose you have been asked to design a phone service that reads the

weather to you. When you call, the service asks you for your name and greets you. You can then look up the weather by city or zip code.

When you first start SUEDE, both your canvas and script bar will be empty. While you can immediately make a the user interface design on the canvas, let's suppose you want to start by creating predefined scripts and designing your interface from those.

In order to make a script, click the arrow on the script bar. As you can see below, by the clicking the arrow the script extends by adding either a prompt or response, depending on what preceded it. Create the script below.

For each card you add, type the appropriate description in the each card and record what the prompt or response should be. You can do this by pressing the record button and speaking into your microphone. When you are done speaking press the stop button. To listen to your recording, press the play button.

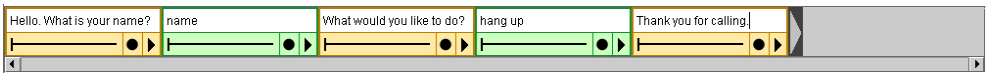


Figure A.5: *You can create additional scripts by clicking the New Script button above the script bar.*

Beginning the Interface Graph

Creating a Graph From the Scripts

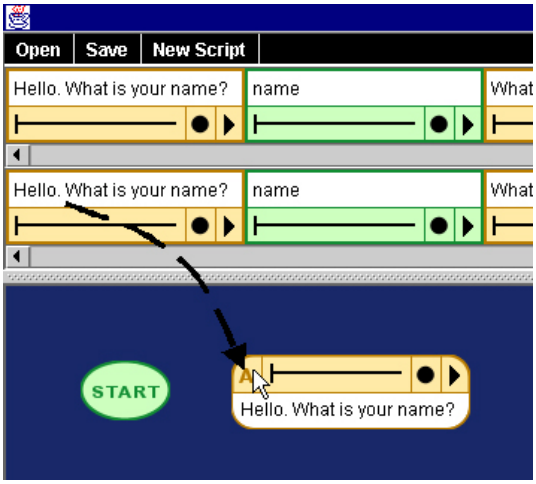


Figure A.6: *Dragging your prompt cards from the script bar onto the canvas will duplicate them, both their text and sound.*

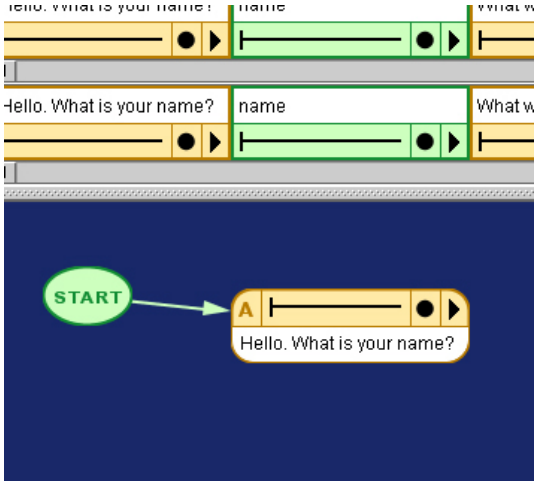


Figure A.7: *For your design to run properly you must designate a prompt card that will be the starting point of your user interface. To do this, click and drag from the start card to the prompt that should be first. An arrow will extend from "start" and will connect to the prompt.*

Adding Response Cards From the Scripts

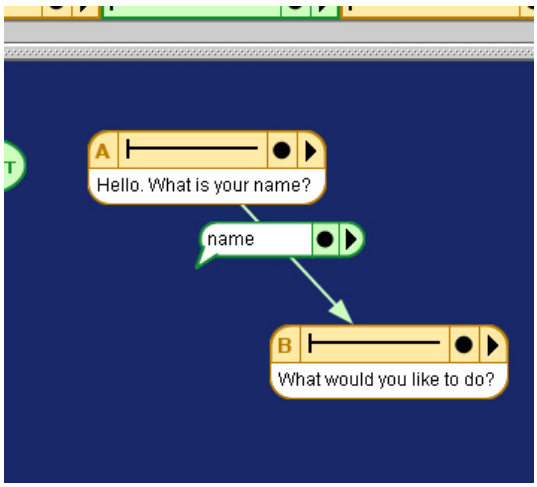


Figure A.8: *Response cards are placed by dragging them from the script bar but unlike prompt cards, dragging a response link over an existing design graph prompt card creates an outbound response link from that card.*

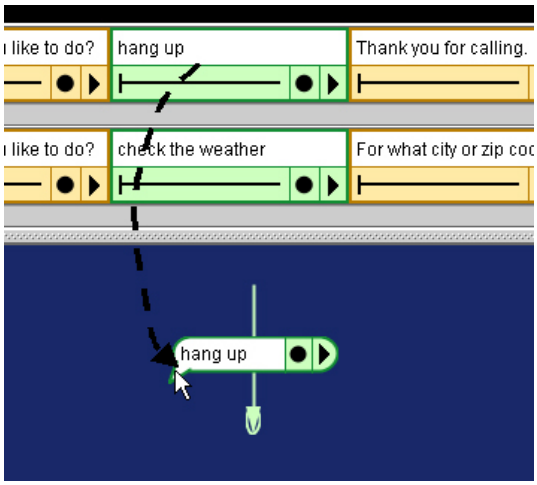


Figure A.9: *Response cards that aren't associated with prompt will become globals. These responses can be said by the user at any time. You can drag and drop the "hang up" response over the canvas, this will allow the user to choose to hang up no matter where they are in the system.*

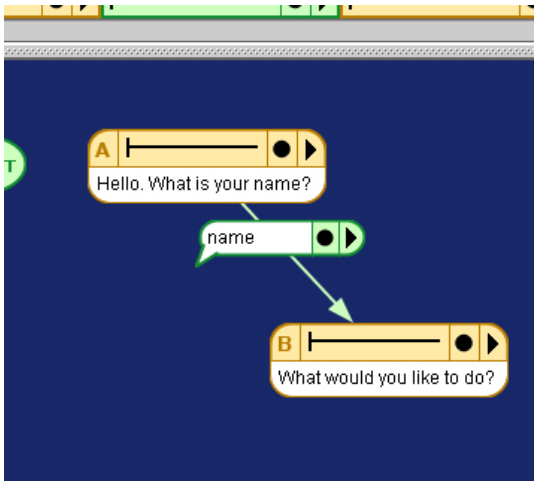


Figure A.10: *To add (or change) a destination of a response card, click on its arrow. The arrow will change colors and you can drag it over the prompt card you'd like to follow that particular response. When you let go of the arrow, the response card will link itself to the prompt card.*

Expanding the Interface

Adding Additional Prompt Cards

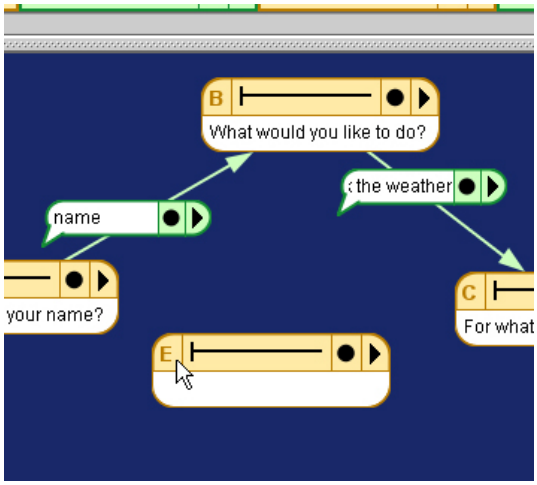


Figure A.11: *You are not limited to the prompt cards in the script bar. You can add as many prompt cards as necessary by simply right-clicking on the canvas. Right-dragging will allow you to move them. Clicking with the middle mouse button (the wheel on a three button mouse) deletes cards (be careful, there is no undo!).*

Adding Additional Response Cards

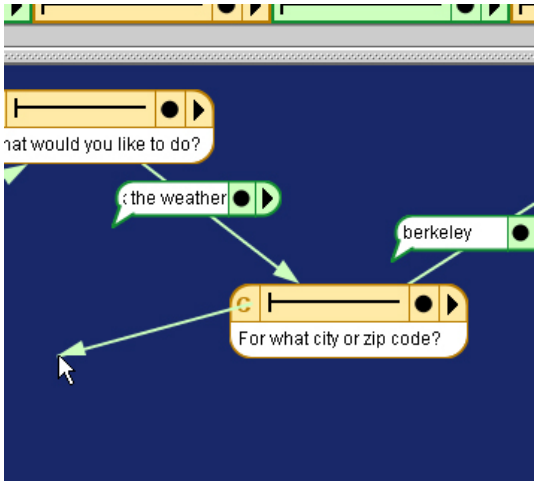


Figure A.12: *Nor are you limited in response cards. Simply click on a prompt card (or canvas for globals) and drag. A response link will appear that can be linked to other prompts. You can delete a response link by clicking on it with the middle mouse button (the wheel on a three button mouse).*

Additional Features

Parameterized Audio Response

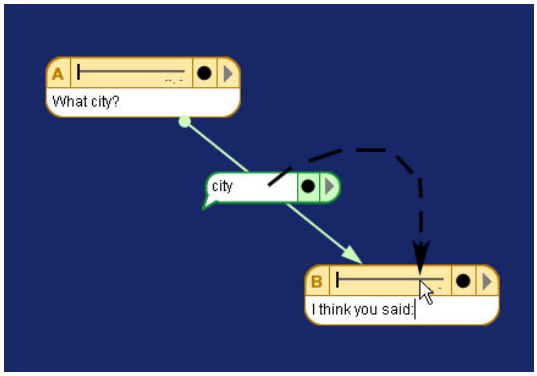


Figure A.13: *If you drag a response over a prompt card, a voice balloon is added. The voice balloon "fills in the blank" in the prompt with the associated response. The description of the response card is added to the prompt card text within {curly brackets}. The letter in the balloon represents the prompt card for which the response is received. You can also add an external wav file to a card. Place the mouse cursor over the card you wish to add the file. Hold down the shift button while clicking the middle mouse button. Then select the wav file you wish to add.*

Prompt Groups

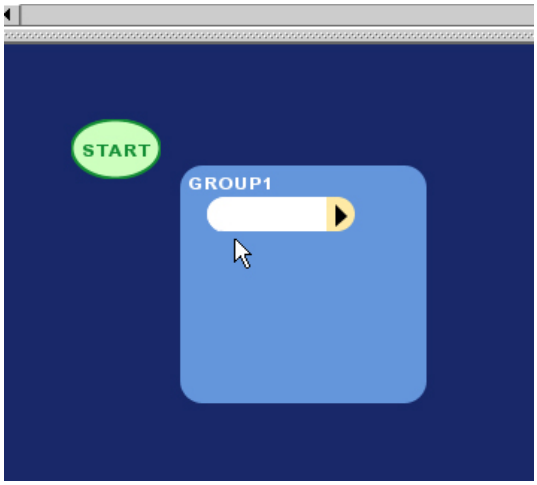


Figure A.14: *A group is a set of prompt cards that can all be alternative replies after a specific participant response. To create a group right-drag on the canvas. To add prompts to the group simply drag them (using the right mouse button) over the group. The same method is used to remove prompt cards from a group. Groups are linked with response cards just like prompts. All the prompts in the group will follow the response link from it. Prompt card can be linked individually also. If you want a certain prompt in a group to have its own response options simply link the prompt before adding it to the group. The same applies to a prompt being linked to.*

Testing the Interface

Figure A.15: *To enter test mode, click on the Test button. A new window will open up that contains the test mode interface. To enter test mode, click on the Test button. A new window will open up that contains the test mode interface.*

Error Modeling

Figure A.16: *In the session menu bar, there is an option to set the error percentage. This value sets the simulated speech recognition error. As the Wizard is running the system, SUEDE can insert random misrecognition errors as a real speech recognizer might do. The frequency of these errors is based on this value.*

Barge-In

Figure A.17: *In the Wizard options bar you have the option to select barge-in. Barging-in is when the user begins giving his or her response before the prompt is done playing. If the user does this, you can click the "barge-in" button and SUEDE will stop the prompt from playing and will immediately start recording.*

Timeouts

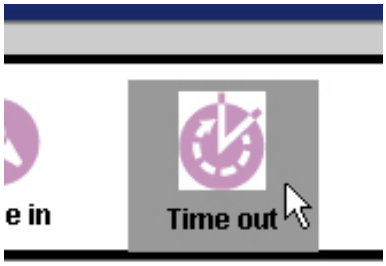


Figure A.18: *Along side the barge-in button in the Wizard options bar is the timeout button. If a user does not respons to a prompt after a certain time it is considered a "timeout" error. In SUEDE, if this occurs, the Wizard has to option to click the "time out" button which will stop the recording and repeat the prompt.*

Not Heard

Figure A.19: *Last is the Not Legal button. If the user's response doesn't match any of the response links pressing this button will play a message telling the user that they aren't able to say that at the moment.*

Thinking About the Interface

Analysis mode is automatically started when the Test window is opened. You can also open saved analyses with the open command. In SUEDE, data collected in Test mode can be reviewed in analysis mode. The analysis mode interface is similar to the Design mode interface. The canvas area cannot be modified from analysis mode. In the script bar, saved transcripts from user session are displayed. In addition, SUEDE automatically calculates the number of times each response was said throughout all the user sessions and displays this number in a red circle in the response card. Each response card's arrow grows proportionally bigger based on the number of times it was followed.

Appendix B

Topiary documentation

Topiary is a system for rapidly prototyping location-enhanced applications . Topiary allows you to create a map to specify spatial relationships of people, places and things, to use this map to demonstrate the scenarios describing the location contexts, for example, "Alice is in Room 525" or "Alice is near Bob". You can then use these scenarios to create storyboard mockups, for example, "Show this screen of Welcome when Alice enters Room 525".

After a series of storyboards have been created, you can "run" the design in Topiary's Test workspace to try out the interaction flow. Here, you act as a Wizard , simulating the current location of people and things. You can also let end-users try out early designs, by letting them try out the interface mockup on one device, while the Wizard follows them around and updates their location on another device. As an optional feature, you can also try out your design with real sensor input if your device has a Wi-Fi.

The Topiary Environment

Topiary has three workspaces: Active Map, Storyboard , and Test workspaces.

In the Active Map workspace, you can create a model of the location of people, places, and things and demonstrate scenarios describing location contexts, such as "Alice is in the Gym" or "Bob is entering Room 525".

In the Storyboard workspace, you can sketch pages and links to create interface mockups, using scenarios as conditions or triggers on a link. For example, you can specify that clicking a button goes to one page if "Alice is in the Gym", or automatically go to another if "Bob is entering Room 525".

In Test workspace, you can test your design or let real users try out the design. You can "run" the sketches on a mobile device like a PDA. A user can interact with these sketches, while a wizard follows the user and updates the

location of people and things on a separate device. Optionally, a sensor infrastructure can be used to update location information, if available.

Basic Elements of the Active Map and the Storyboard Workspace

- The Radar View is an overview of the Workspace area
- The Scenario Repository contains all of the scenarios you have created
- The Toolbox contains various tools that you can use while designing
- The Canvas is the area where you specify scenarios or create storyboards
- The Workspace Tabs lets you switch between the Active Map (where you specify scenarios describing spatial relationships) and the Storyboard (where you can sketch out user interfaces)

Note I: The Active Map workspace and the Storyboard workspace have different tools in their toolboxes.
Note II: The Test workspace has a different layout and it can be brought out in the Storyboard workspace (rather than using tabs).

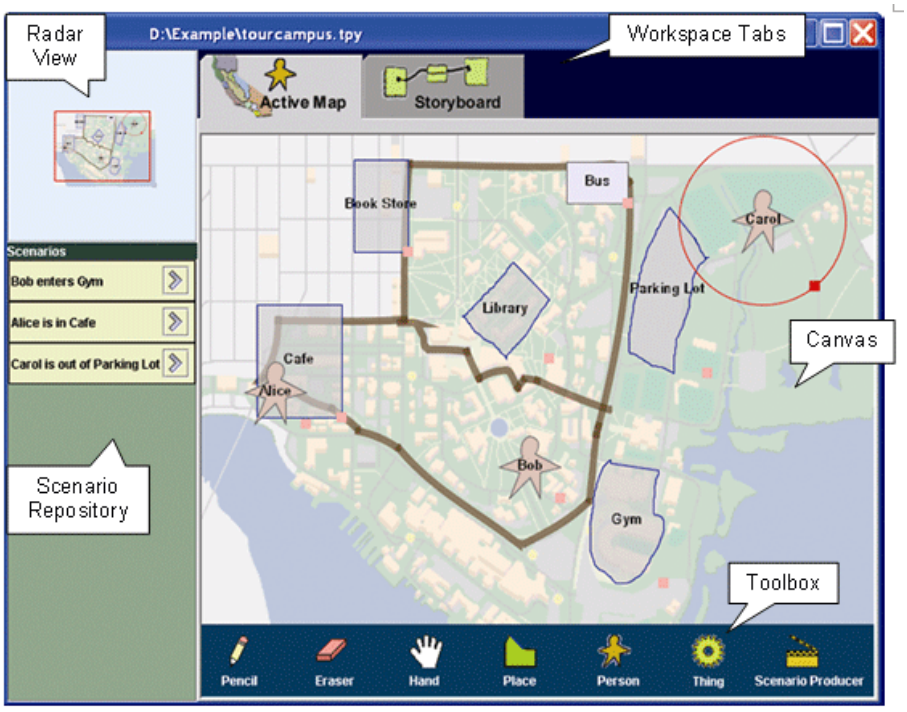


Figure B.1: *Overview of Topiary.*

Active Map Workspace

You can model the spatial relationships between people, places, and things in this workspace. You can load a GIF or JPEG image of a map into the workspace as a background image to help with positioning of these entities. You can use the Pencil tool to draw paths, which are used for the wayfinding feature. Topiary parses these informal sketches into a road network that can be dynamically searched for the shortest path.

Creating Entities

You can create the models of entities in the Active Map workspace. Each of these entities (place, person, or thing) is given a unique default name such as "Place5". This name can be replaced with typed text. Places can also be created within other places, creating hierarchies.

Figure B.2: *Toolbox of the Active Map workspace.*

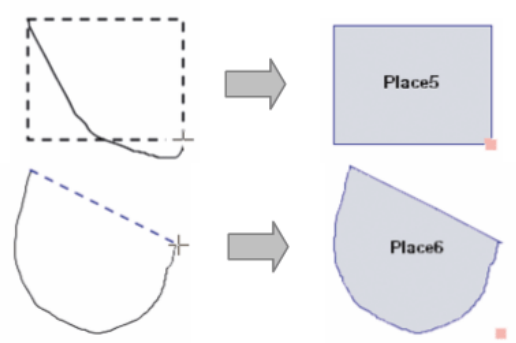


Figure B.3: The Place tool is used to create places. To use it, you can outline the boundaries of a place. You can easily create rectangular places as well as arbitrarily-shaped ones.

The Person tool is used to create a person. You can select the Person tool and then click on where you wants the person to be. The Thing tool works in the same manner, but is used for creating things such as cars and printers. The models of people and things can be moved around by dragging and dropping.

Capturing Scenarios

Figure B.4: *Scenarios are captured with the Scenario Producer tool.*

Figure B.5: *Like a screen capture tool, selecting the Scenario Producer tool brings up a recording window that can be positioned over entities of interest. This picture shows that the green recording window is positioned over three entities, Bob, Alice and the Gym.*

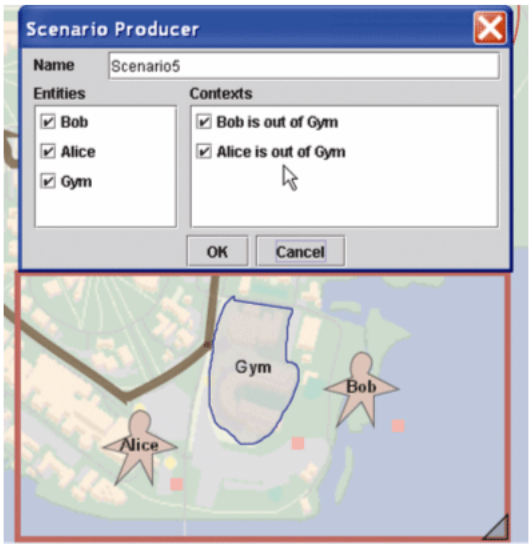


Figure B.6: *Once the recording window is dropped, Topiary will distill location contexts from thespatial relations of the included entities. This window can be resized to include or exclude entities.*

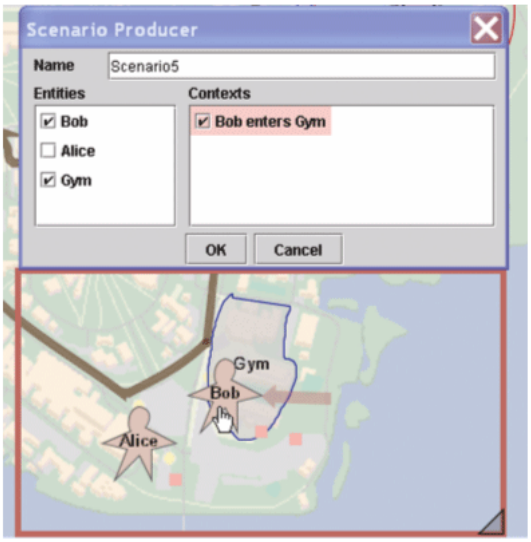


Figure B.7: *A dialog box is then brought up that lets designers select contexts of interest. The left side of this dialog box contains a list of entities that can be used for filtering contexts. Unchecking an entity removes all contexts associated with that entity.*

You can also demonstrate transitions by moving entities within the recording window. For example, dragging Bob into the Gym changes the event "Bob is out of Gym" into "Bob enters Gym". The below picture also shows an example of filtering. Entity Alice is unchecked, and all related contexts are filtered out.

Clicking the OK button, a new scenario will be added to the Scenario Repository. A scenario can be customized or generalized in a Scenario Detailed View.

Storyboard Workspace

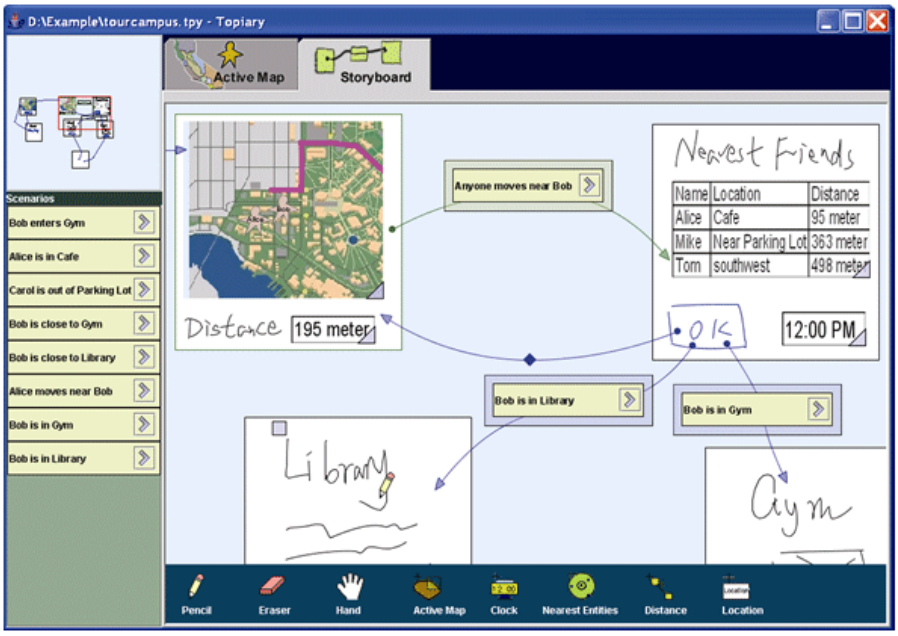


Figure B.8: *You can create interface mockups in the Storyboard workspace, creating pages that represent screens and links that represent transitions between pages. S cenarios created in the Active Map workspace can be used as conditions or triggers on links, and context components can be embedded into pages.*

Basic Interaction

- Creating pages and links using the Pencil tool
- Manipulating objects
- Deleting objects using the Eraser tool
- Navigation
- Adding context components using the five Context Component tools

You can select an object by a right-click while using a mouse or by Ctrl-tap or barrel tap while using a pen. You can also select a group of objects, e.g., pages or the ink on a page, by drawing a circle to include the target objects. Then you can manipulate the objects by moving, copying, pasting them.

Pages

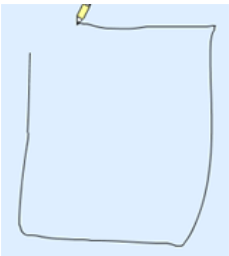


Figure B.9: *A page represent a screen of visual output of an application. To create a page, use the Pencil tool and draws a rectangle, which is recognized as a page.*

Figure B.10: *Pages allow freeform ink, which are processed only for smoothing and grouping. Tglobalshere are no other forms of recognition. In addition, context components can be embeded into a page.*

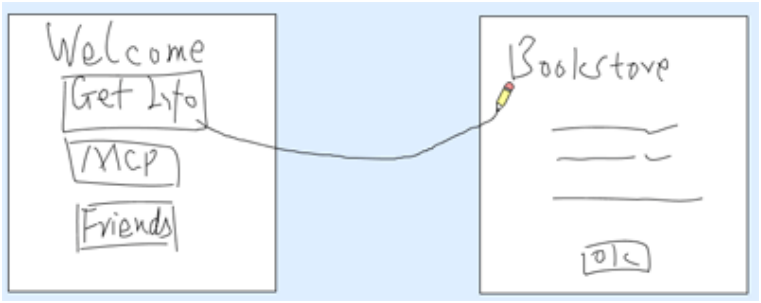
Links

Topiary has two kinds of links: explicit links and implicit links. Explicit links, denoted in blue, start on ink with a page. Explicit links represent GUI elements that users have to click on, e.g., buttons or hyperlinks. Implicit links, denoted in green, start on an empty area in a page. Implicit links represent transitions that automatically execute when scenarios associated with that link occur. Explicit links model actions taken by end-users, whereas implicit links model sensed data.

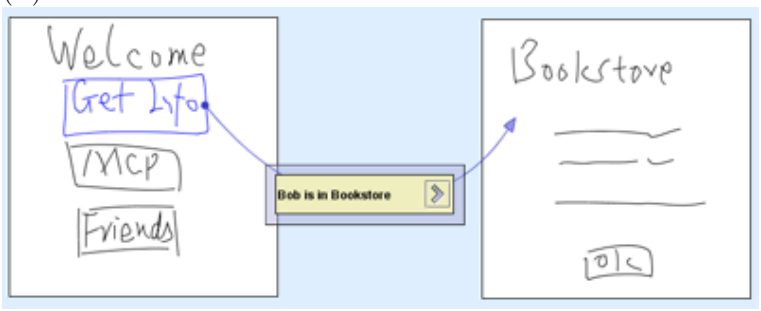
- Creating Explicit Links
- Creating Implicit Links

- Link Manipulators

(A)



(B)



(C)

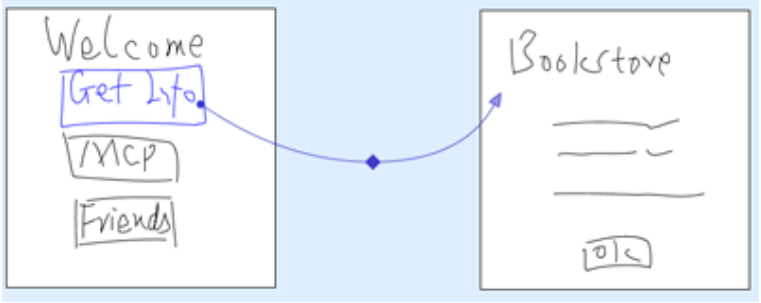


Figure B.11: *Creating Explicit Links.*

(A) To create an explicit linik, draw a line from the ink on a page to another page.
(B) An explicit link is created.
(C) A scenario "Bob is in Bookstore" is dragged into this link. Now the transition can take place when a user clicks on "Get Info" and Bob is in the bookstore.

Figure B.12: Creating Implicit Links

(A) To create an explicit linik, draw a line from an empty area in a page to another page.
(B) An implicit link is created.
(C) A scenario is dragged into this link. Now the transition can automatically take place once "anyone moves near Bob".

The solide tag in the middle of a link can be dragged to adjust the layout of the link. It can also used to insert built-in scenarios. The tag is called Link Manipulator in Topiary. The link manipulator becomes a translucent rectangle container when scenarios are put into this link, by which you can still adjust the link. In addition to scenarios produced via the Scenario Producer in the Active

Map workspace, two kinds of built-in scenarios can be directly inserted into a link. They are movement speed and temporal conditions (times, time intervals and elapsed times). To insert a built-in scenario to a link, bring up the pie menu on a link manipulator , and then use the pie menu item "Insert Conditions->..."

Using Context Components

Topiary provides five built-in context components for displaying spatial and temporal information. These components make it easy to prototype features common in many location-enhanced applications that are hard to specify using storyboard alone. The values for these components are automatically updated based on the simulated (or sensed) locations of people, places, and things in the Test workspace.

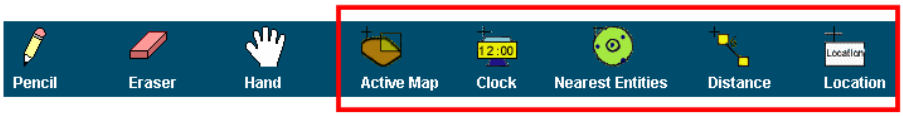


Figure B.13: *Toolbox of the Storyboard workspace.*

To add a context component, pick the according tool in the toolbox first, and then use the tool to specify a location and size on a target page by drawing a rubber band. A component-specific dialog box will appear by which you can customize a component.

Active Map Component

The Active Map component allows you to embed a view of the Active Map workspace into a page, letting end-users see either part of or the entire map, the current location of people and things, as well as the shortest path to a destination (if one is specified).

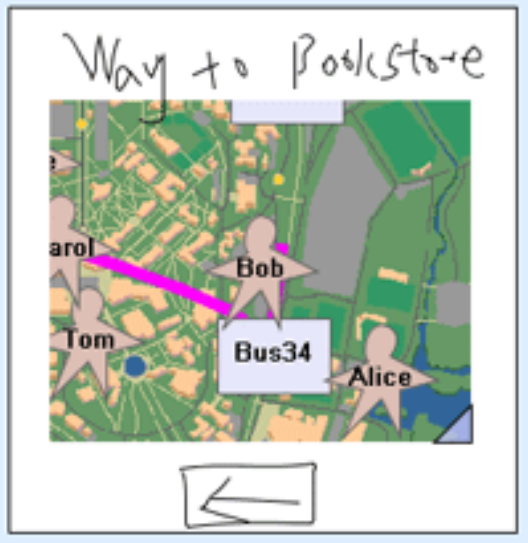


Figure B.14: *An Active Map component is embeded into a page.*

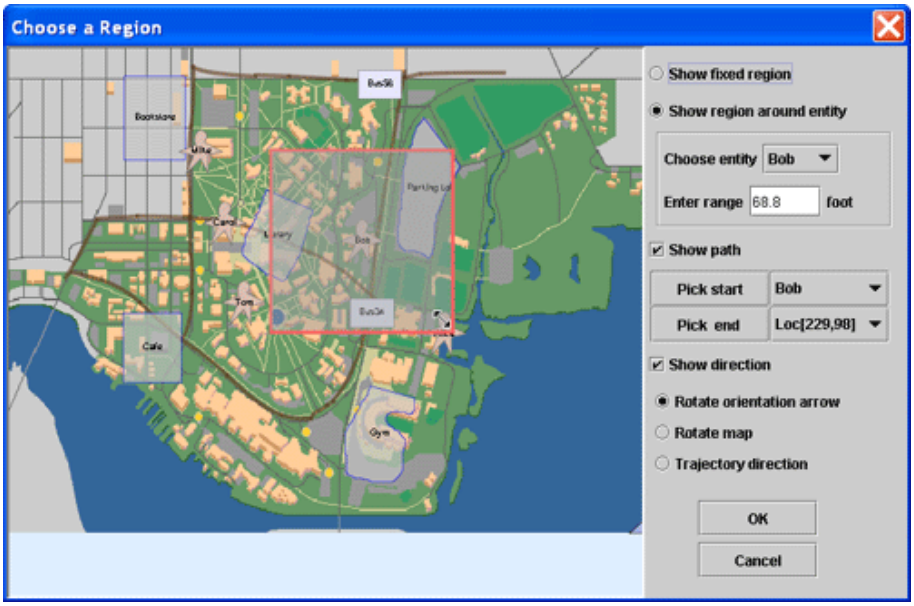


Figure B.15: *A dialog box lets the designer show a fixed region of the Active Map workspace or a region around an entity, e.g., a 68 foot square around Bob. You canset up a mapping from pixels to physical measurements .*

You can choose to show a path by specifying a starting point and an end point, both of which can be fixed points or an entity. This path is dynamically generated based on the road network drawn in the Active Map workspace. You can also choose to show directional information. Three options are available for showing directional information:

- Rotate orientation arrow: An arrow on the map indicates the direction that an end user is facing.
- Rotate map: The whole map automatically rotates according to the simulated (or sensed) direction information so that the orientation of the map always keeps consistence with the real geographical directions.
- Trajectory direction: An arrow on the map shows the trajectory direction of an end user.

Time Component

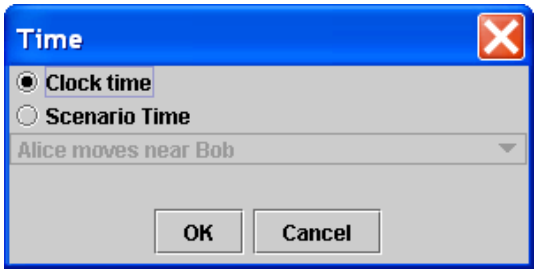


Figure B.16: The Time component is used to display either the current time or the time when a scenario with context transitions happened, e.g., the time when Alice movesLinks near Bob.

Nearest Entities Component

Figure B.17: *The Nearest Entities component displays a table of the nearest N entities from a set of entities.*

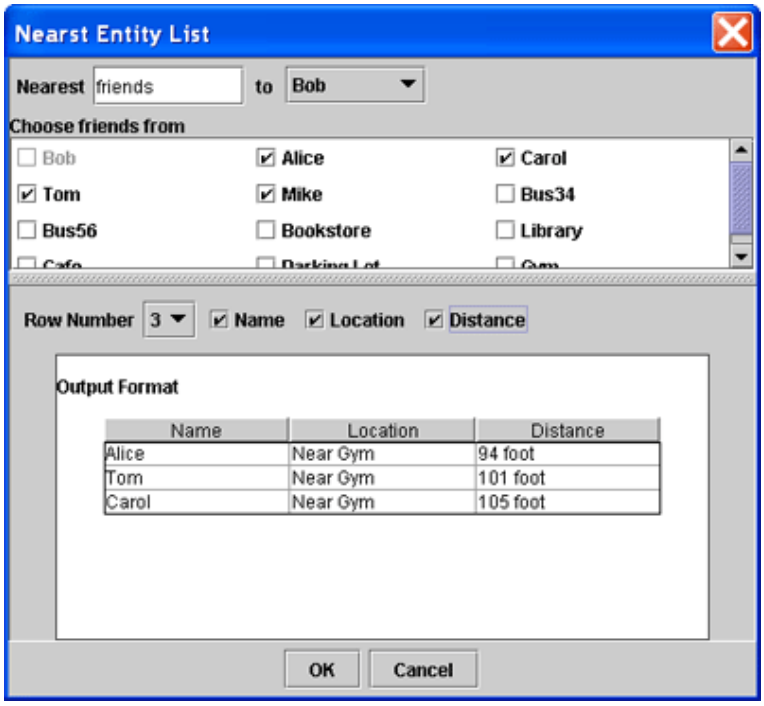


Figure B.18: *When this component is used, a dialog is brought up that lets you select the set of entities to choose from. You can also choose to show the name, location, and distance of these entities.*

Distance Component

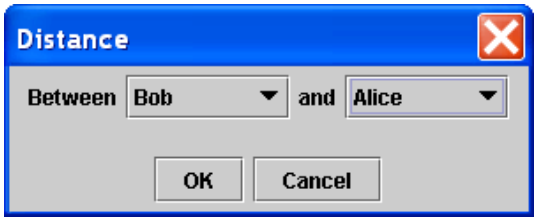


Figure B.19: *The Distance component shows the distance between two entities.*

Location Component

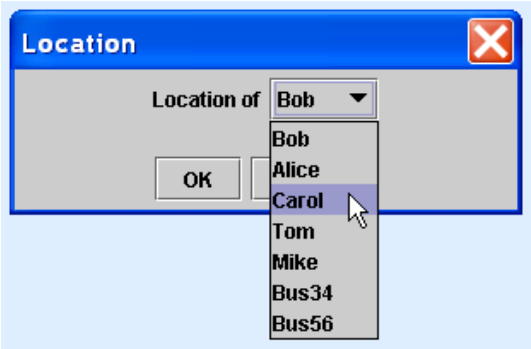


Figure B.20: *The Location component displays an entity's location by name.*

Test Workspace

After a few mockups have been created, you can try out your designs in the Test workspace. To enter the Test workspace, open a pie menu on the desired start page and then selecting the Test option.

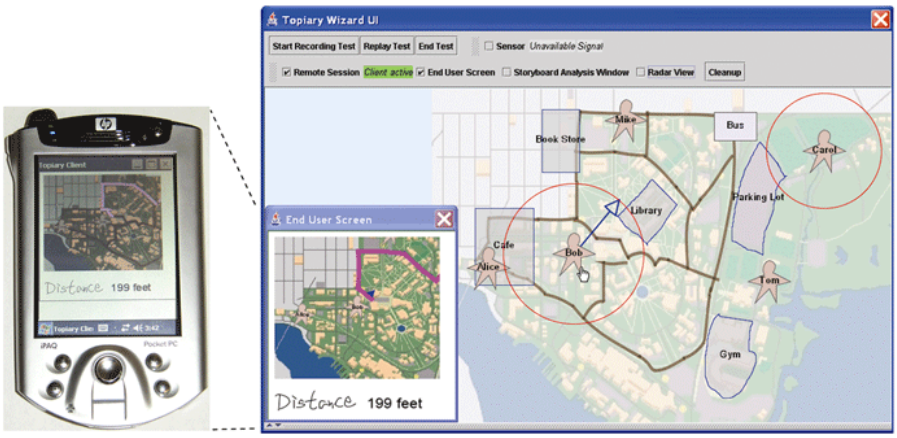


Figure B.21: *The Test workspace has two major parts: the End-User UI (left) and the Wizard UI (right). The End-User UI is what end users will see and interact with. The Wizard UI is where the designer can simulate location contexts, while observing and analyzing a test. These UIs can be run on the same device (to let a designer try out a design) or on separate devices (one for the Wizard, the other for the user).*

Wizard UI

The Wizard UI allows you to test your design using the technique of Wizard of Oz . The Wizard UI has four parts. The Wizard Map is a copy of the Active Map workspace , with the key difference being that it represents the current location of people and things. You can simulate location context by moving people and things around to dynamically update their location. If moving a person or a thing causes an implicit link to activate, then the End-User UI will automatically transition to that page.

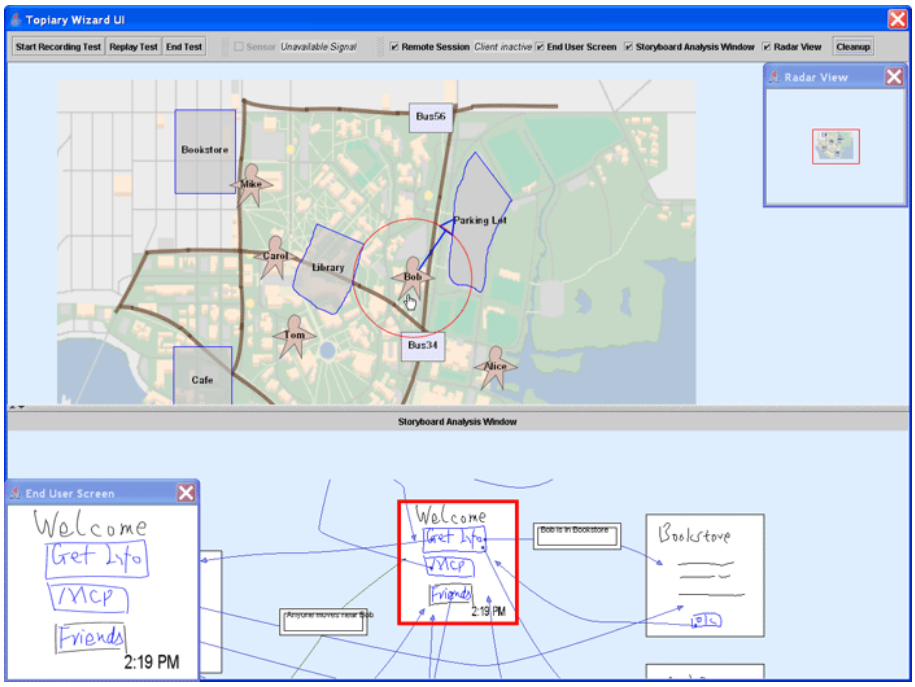


Figure B.22: *On the bottom-left is the End-User Screen, a copy of the End-User UI , which also updates in response to end-user input on a PDA. You can click on the same links that a user could as well for test purpose. A Radar View (see the top-right corner) of the map area is provided for navigation. The Storyboard Analysis window (see the bottom panel) shows a simplified view of the storyboard workspace with the current page and the last transition highlighted, which can help to figure out interaction flows.*

Analyzing a Design

You can analyze a design by watching the transitions in the Storyboard Analysis Window or Recording&Replaying a test.

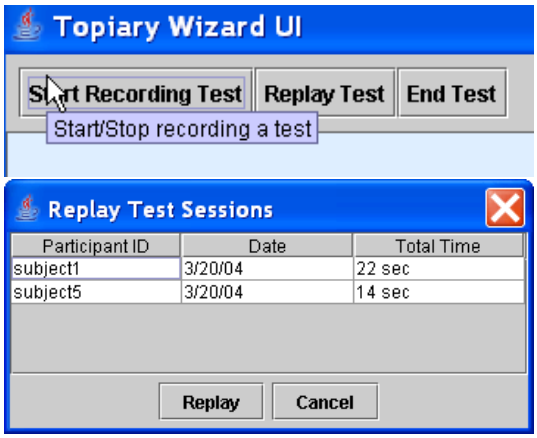


Figure B.23: *You can replay a recorded session. Topiary can capture users' actions, like mouse movements and clicks, as well as physical paths traveled.*

Testing with Real Sensor Input

Wizard of Oz techniques allow you to test a design in any situations without deploying a supporting infrastructure. However, to enable larger scale and more realistic testing, Topiary also allows testing with real sensor input (checking the Sensor checkbox in the Test workspace enables Topiary to accept the location input from a sensor infrastructure). Topiary is independent of sensor infrastructures. Place Lab has been added into Topiary as a source for sensor input.

Appendix C

Introduction to the test

Beste meneer/mevrouw,

Allereerst bedankt voor het meewerken aan dit onderzoek. In dit onderzoek is het de bedoeling een systeem te testen. Dit is de eerste week waarin het systeem wordt getest en bevat hierdoor ook nog enkele belemmeringen. Zo kan het systeem af en toe trager dan je gewend bent van een systeem reageren en kan de uitkomst nogal verrassend zijn.

Zo dadelijk zal het systeem een aantal foto's omtrent vakanties laten zien. Het is de bedoeling om je mening te geven over de foto's, zodat het systeem duidelijk krijgt van wat voor soort vakantiebestemmingen u houdt. Door op deze manier over een aantal foto's je mening te geven, zal het systeem een kaart van de aarde tonen met de aangewezen lokaties voor u. Let hierbij op dat dit systeem nog volop in ontwikkeling is. Het systeem werkt op het principe van een learn based model. Dit houdt in dat naarmate er meer mensen hun mening geven, het systeem steeds meer informatie hieruit kan extraheren. Aangezien dit de eerst week is, zal het systeem hierdoor nogal beperkt kunnen overkomen, doordat deze niet genoeg "leerdata" heeft verkregen. Juist door deze fotos waar veel over te vertellen valt, kan het systeem zien waar de testprsoon op let en dus waarde aan hecht. Op deze manier informatie extraheren is de toekomst. Afhankelijk van wat u zegt, zal het systeem plaatjes weergeven waardoor hij probeert meer informatie te extraheren. Als u naar een volgende afbeelding wilt, kan u gewoon "volgende" zeggen. Deze methode wordt nu bij vakanties toegepast, maar kan bij een door ontwikkelde methode ook op andere toepassingen worden gebruikt, bijvoorbeeld bij het maken van keuze welke auto u graag wilt kopen.

Ik wens u veel plezier bij het testen van dit systeem.

Met vriendelijke groet,

Dave Nabuurs

Appendix D

Analysed data

Below are the test results. This data is also analyzed. First the image were the participant gives his opinion about is displayed. Then the opinion of the participant is displayed. Here are the keywords bold. This can be nouns or verbs that say something about the picture. After this a table what the keywords of the participant are, is displayed. Then whether it is positive (+), neutral (o) or negative (-). Followed by the words that the wizard clicked and if he clicked the word positive or negative and then the time between them, rounded to seconds. After the table, the conversation between me and the wizard is shown. Finally, there is an extended horizontal line and starts the next image.

Uhm.... Ja wat moet ik hier van vinden. Moter, ***moter rijden*** *ben ik niet zo van,* ***kamperen*** *vind ik wel cool, maar dan zou ik dat liever met een* ***tent*** *doen of zo.* ***Mooi weer*** *vind ik dan wel cool en het is wel een vette wagen. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
moter rijden	**-**	moter rijden	**-**	2
kamperen	+	kamperen	+	2
tent	+			
mooi weer	+	zon, zee en strand	+	7

Dave: En dacht je hier al iets bij?
Alice: Uhm... Hier miste ik iets. Hij zei van "mooi weer is wel cool", maar er staat alleen maar één ding waarbij zon, zee en strand in staat. Terwijl hij niks over zee en strand zegt.

Ja, is wel cool, mooi ***sneeuw****, mooi* ***uitzicht****,* ***stad****, lekker gezellig. Nou gewoon perfect, ik zou hier wel in een* ***hotelletje*** *kunnen zitten voor een paar dagen. Volgende.*

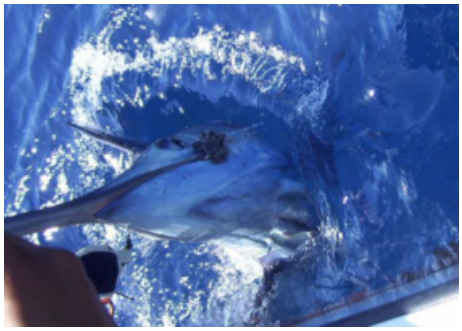
words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
sneeuw	+	x		
uitzicht	+	x		
stad	+	drukte	+	5
hotelletje	+	terrasje	+	6

Alice: Achteraf zie ik nu dat er sneeuw bijstaat, maar tijdens de test heb ik deze niet gezien.
Dave: Hoe komt het dat je sneeuw dan niet hebt gezien, wist je niet dat die er was?
Alice: Ik meende eigenlijk wel dat die er was, maar doordat hij eigenlijk veel zegt in een korte tijd, en ik een hoop dingen moet verwerken, en tegelijkertijd aan het zoeken ben, heb ik hem niet gevonden.

Ja beetje hetzelfde verhaal als net he, kan er eigenlijk niet veel aan toevoegen. Het is gewoon weer mooi. ***Sneeuw****, ik ben wel van de* ***sneeuw****, krijg er wel en beetje een kerstgevoel van. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
sneeuw	o	x		
sneeuw	+	apreski, sneeuw	+	2, 23

Dave: Hier heb je sneeuw ook niet gevonden?
Alice: Nee inderdaad.



Wat is dat? Oh een ***vis****.* ***Vissen****, nee doe mij maar niet op de* ***zee****, ik ben niet zo van het* ***water****. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
vis	o	x		
vissen	-	vissen	-	8
zee	-	zon, zee, strand	-	3
water	-	zon, zee, strand	-	1

Alice: Ik heb hier zon zee strand weer op min gezet, omdat hij zegt dat hij niet van het water is. Ook vissen heb ik op min gezet, want hij zei letterlijk dat hij niet zo van vissen hield. En dan vind ik ineens de sneeuw, dus heb deze nog

aangeklikt van het vorige plaatje.
Dave: Dus niet van het eerste plaatje?
Alice: Nee, daar heb ik niet meer aan gedacht.

***Bergen**, mmm. **Sportief**, nee ik zou niet op vakantie te gaan om **sportief** bezig te zijn. In ieder geval zoals die kerel hier staat. Misschien wat minder, een keertje **mountainbiken** lijkt wel leuk, maar om er nou zo'n **sportvakantie** van te maken, lijkt me niks. Maar om er een keer **chill doorheen te lopen**, lijkt me dan wel mooi. Een keer van het **uitzicht** genieten ofzo. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
bergen	o	x		
sportief	o	x		
sportief	-	sportief	-	1
mountainbiken	+	fietsen	+	5
sportvakantie	-	x		
chill doorheen te lopen	+	wandelen	+	2
uitzicht	+	omgeving	+	4

Alice: En hier klik ik fietsen aan, omdat hem mountainbiken wel leuk lijkt, en aangezien mountainbiken ook fietsen is, klik ik die aan. Hij zegt hier uitzicht genieten en dan zie ik dat als omgeving.

Uhh nee, volgens mij ben ik hier te jong voor, haha... Volgende

No keywords have emerged out of the content analysis, but the wizard clicked on the keywords "kinderen".

Dave: En waarom klik je hier kinderen aan?
Alice: Hij ziet kinderen en hij zegt ik ben hier te jong voor. Ik ga er vanuit dat dat over de kinderen gaat in plaats van over het huisje.
Dave: En wat had je zonder die kinderen dan gedaan.
Alice: Dan had ik het huisje op min gedaan.

*Geweldig! Een beetje kijken hoe andere **stammen** of wat het ook zijn beetje leven. Mooie **natuur**, mooi **weer**, **ver weg**. Ja briljant echt briljant. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
stammen	+	omgeving, cultuur snuiven	+, +	1, 2
natuur	+	natuur	+	1
weer	+	zon, zee en strand	+	2
ver weg	+	lange tijd weg, primitief	+, +	6, 7

Alice: Dingen die mij hierbij opvallen is mooi weer, dus zon zee strand, terwijl er geen zee en strand te zien is. Daarbij heb ik de natuur en omgeving aangeklikt, cultuur snuiven, aangezien hij stammen wilde bekijken en primitief en lange tijd weg, omdat hij zo enthousiast was over het plaatje en het plaatje was vrij primitief.
Dave: Dus je hebt puur op het plaatje primitief aangeklikt?
Alice: Ja

*Is een beetje hetzelfde, beetje **cultuur** bekijken, bekijken waarom het hier zo'n feest is. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
cultuur	+	cultuur snuiven	+	1

Alice Hij zei cultuur bij dit plaatje, dus heb ik hier cultuur aangeklikt.

*Dit zullen wel wandelschoenen zijn, of **bergschoenen**. Zo fanatiek als hij of zij vind ik niet nodig, maar een keer door de **bergen** lopen, lijkt me dan wel weer leuk. Volgende.*

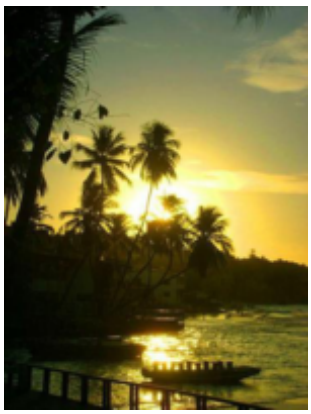
words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
bergschoenen	-			
bergen	+	bergen	+	2

Alice: Er wordt wel de hele tijd over bergen gepraat, maar niet echt over berg-beklimmen, maar uiteindelijk zeg hij op het einde dat hij bergen wel leuk vindt, en daarom heb ik "bergen" aangeklikt en niets met "klimmen" gedaan.

*Ja, het is een **stad**, maar om te zeggen dat dit een mooie **stad** is. Het **uitzicht** doet me ook niet zo gek veel hier. Nee, vind dit eigenlijk niet zo leuk hier, ik houd niet van zo'n soort **steden**. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
stad	o	x		
stad	-	x		
uitzicht	-	x		
steden	-	x		

Alice: Hier heb ik niks aangeklikt, omdat hij veel zegt over het inhoudelijke van het plaatje, maar niet over "ik vind in de stad op vakantie gaan leuk". Hij zegt meer dat hij die stad niet leuk vindt. En aangezien hij het puur over deze stad heeft, heb ik aangenomen dat het puur aan dit plaatje ligt.



*Relaxed, **palmboompje**, **biertje** erbij wat wil je nog meer. Eigenlijk wel het ideale vakantieplaatje, lekker ver weg in het **buitenland** zitten. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
palmboompje	+	zon, zee en strand	+	4
biertje	+	terrasje	+	5
buitenland	+	cultuur snuiven		

Alice: Hier zei hij wel echt zo palmboompje biertje erbij, dus heb ik weer zon zee strand aangeklikt en terras.
Dave: Want?
Alice: Palmbomen staan op het strand, en bier kan je op het teras drinken.
Dave: En waarom heb je op "cultuur snuiven" gedrukt*?*
Alice: Omdat hij het lekker ver weg in het buitenland wel leuk vond en ik moest hierbij meteen aan "cultuur snuiven" denken.

Ja, ***cruiseschip*** *lijkt me dan wel een keer leuk, omdat dat zo'n groot* ***schip*** *is, en lekker luxe en niet echt het idee heb dat je echt vaart. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
cruiseschip	+	cruiseschip	+	0
schip	o	x		

Alice: Hier alleen cruiseschip aangeklikt, omdat hij exact zegt dat hij het leuk vindt om dat nog een keer te doen.

***Skiën**, zou ik altijd al een keertje willen doen, ben wel eens benieuwd hoe het is, heb het eigenlijk nog nooit gedaan en lijkt me wel leuk, maar ook wel moeilijk, maar dan wel met vrienden en niet met me vriendin. 'S middags lekker **skiën** en dan 's avonds mooi de **kroeg** in. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
skiën	+	skien/snowboarden	+	7
skiën	+	x		
kroeg	+	apreski, zuipen	+, +	1, 1

Alice: Hij zegt skiën en zuipen en apreskiën, dus die heb ik aangeklikt. Dit zijn precies de woorden die ook als knopjes staan, dus dat is wel lekker makkelijk.
Dave: Hij zei eigenlijk niet apreskieën, maar de kroeg in.
Alice: Ja met skievakantie vind ik dat wel apreskiën.

*Tja, als je dan toch met je **vrienden** op **vakantie** bent, kun je er gelijk bij gaan **zwemmen** haha, Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
vakantie	o	x		
zwemmen	+	x		

Alice: Hier heb ik niks aangeklikt, want zwemmen staat er niet bij en dat is het enige wat hij eigenlijk zei.

Uhm... ik weet niet of die aan het wandelen is of aan het backpacken. ***Wandelen*** *heb ik al mijn verhaaltje over gedaan. En* ***backpakken****, dat lijkt me dan wel cool, maar dan zou ik het niet in deze* ***omgeving*** *zoeken, maar meer in het* ***oerwoud*** *of zo. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
wandelen	0	x		
backpacken	+	van hot naar her, klein gezelschap, lange tijd weg, primitief	+, +, +, +	6, 7, 8, 8
omgeving	-	x		
oerwoud	+	oerwoud	+	2

Alice: Hij zegt "heb ik al genoeg over gezegd", maar omdat ik niet meer wist wat er over het wandelen werd gezegd, heb ik niets meer aangeklikt. Daarbij zeg hij dat packpakken wel leuk lijkt. Maar het woord backpacken staat er niet bij, maar heb de woorden die mij hier aan doen denken, namelijk van "hot naar her", "alleen klein", "gezelschap", "primitief" en "lange tijd" weg aangeklikt.
Dave: En ging het volgende drukken altijd goed?
Alice: Ja, maar soms ging het volgende zeggen te snel, dat ik nog bezig was met na te denken welke dingen ik zou moeten aanklikken, dus dan voelde je de druk om snel op volgende drukken.

*Tja, in een **tentje** te gaan slapen. Lijkt me dan wel mooi, maar om dan op zo'n **familieparadijs** te zitten, daar ben ik eigenlijk nog te jong voor. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
tentje	+	kamperen, kamperen, tent/caravan	+, -, +	2, 9, 9
familieparadijs	-	kinderen	-	1

Alice: Hij zegt kamperen lijkt me wel leuk, maar niet op zo'n familie camping, dus kinderen weer min en kamperen weer plus. Een aangezien hij ook tent zei, heb ik die ook aangeklikt.

*Kijk dat bedoel ik, lekker **primitief** aan een dingentje hangen, **zee** of wat is het? **Meertje**. Lekker **kamperen**, maar dan wil ik wel met een **gids** zijn en niet zelf op de bonnefooi bij een meer gaan zitten. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
primitief	+	primitief	+	1
zee	0	x		
meertje	0	x		
kamperen	+	tent/caravan,	+,	2,
		kamperen	+	2
gids	+	klein gezelschap	+	5

Alice: Ik heb kamperen aangeklikt en ook primitief, omdat hij hier weer enthousiast over wordt en ook letterlijk zegt, maar hij wil het dan wel met een gids doen, dus heb ook klein gezelschap aangeklikt, omdat hij met een gids ook meestal met een klein gezelschap bent.

Moter rijden *vind ik helemaal niks,* ***bergen*** *vind ik leuk, maar om daar nu met de* ***moter*** *doorheen te rijden, geef ik helemaal niks om. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
moter rijden	-	x		
bergen	+	x		
moter	-	x		

Alice: Hier heb ik niks aangeklikt
Dave: Ook geen "moter rijden"?
Alice: Nee die kon ik niet vinden
Dave: Maar straks is die al aangeklikt?
Alice: Ik denk doordat hij al weer snel volgende zei, ik niet meer verder ben gaan zoeken.

Tja, ***kamperen*** *lijkt me dan wel mooi, maar dan zou ik het niet in dit gebied doen. En volgens mij zijn ze daar op* ***fietsvakantie*** *en dat lijkt me dan wel weer wat minder.* ***Fietsvakantie*** *daar houd ik niet zo van. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
kamperen	+	kamperen	+	3
fietsvakantie	-	fietsen	-	3
fietsvakantie	-	sportief	-	1

Alice: Kamperen weer aangeklikt, omdat hij het weer leuk vond. Aangezien hem fietsvakantie niet leuk leek, heb ik fietsen op min gezet. En ook sportief heb ik op min gezet, omdat fietsen sportief is.

Nou, leuk. Denk wel over een paar jaar, ben er nu nog te jong voor. Heb wel het idee dat ik in een duits toeristisch plaatsje ben. Waar ze een ***haven*** *hebben, maar uh. . . ja. . . ja, leuk voor over een paar jaar. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
haven	-	haven	-	5

Alice: Hier wordt veel over het plaatje gezegd, over wat hij ziet, maar eigenlijk kon ik alleen haven op min zetten, omdat hij zegt eigenlijk ben ik nog te jong, maar hij zegt niet waar hij te jong voor is, dus had ik niks om op te klikken.

*Ja, gezellig, lekker **wandelen**. **Natuur** is wel mooi. Beetje van de **omgeving** genieten. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
wandelen	+	x		
natuur	+	natuur	+	2
omgeving	+	omgeving	+	2

Alice: Hij zei "gezellig lekker wandelen", maar wel op een sarcastische manier. Ik heb er eigenlijk niks bij aangeklikt, omdat ik het eigenlijk niet zeker wist. Mooie natuur en omgeving genieten heb ik weer aangeklikt, om hij het hier wel positief over had.

*Oh gek.. nee! Veel te **druk**, hier ga ik echt niet aan zitten, hoewel het **weer** is wel mooi. **Strand** vind ik ook wel cool, maar dan zit ik liever aan een chill **strand** in verweggistan. Lekker met een **cocktailtje**. Maar dit gaat hem niet worden. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
druk	-	drukte	-	1
weer	+	zon, zee en strand		
strand	+	zon zee en strand	-	4
strand	0		+	3
cocktailtje	+	terrasje	+	2

Alice: Ik zie hier een strand op het plaatje, en hij zegt meteen "gek, wat druk!". Dus strand minnetje, want hij lijkt het niet leuk te vinden. Ook op drukte heb ik min gedaan. De zin later wordt er gezegd: "strand lijkt me wel leuk", dus heb ik deze weer op een plusje gezet

Dave: Één plusje? Om neutraal te maken of twee om hem positief te beoordelen?

Alice: Een plusje, omdat hij er eerst iets negatiefs over zegt .

Oeeh.. krijg het idee dat ik in New York of een andere grote ***stad*** *ben. Lijkt me wel leuk om hier door heen te lopen, kijken hoe het daar is.* ***Nachtleven*** *lijkt me ook wel cool. Ik denk dat je in zo'n* ***stad*** *ook wel lang kunt vermaken. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
stad	+	gebouwen	+	2
nachtleven	+	zuipen	+	2
stad	+	x		

Alice: Hij zegt "stad vind ik wel leuk", maar het woord stad staat er niet bij, maar gebouwen staan in een stad, dus heb ik die aangeklikt. Het nachtleven lijkt hem wel leuk om te bekijken, maar omdat nachtleven er niet bij staat, heb ik zuipen aangeklikt, maar nu zie ik dat uitgaan er ook bij staat en ik die er eigenlijk beter bij vindt passen.

Haha, ***wandelen****. Ik geeft er niet zo veel om om dat op die manier te doen. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
wandelen	-	wandelen	-	2

Alice: Hier zegt hij alleen dat hij wandelen niet leuk vindt, dus heb ik wandelen op min gezet.

Wel ***druk****, maar volgens mij zie je hier een beetje India of Afrika of zo. Lijkt me wel interessant om te* ***kijken wat daar te bedoeling*** *is. Lekker daar eens* ***rond te lopen*** *,* ***marktjes kijken****. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
druk	0	x		
kijken wat daar te bedoeling	+	cultuur snuiven, bezienswaardigheden	+, +	2, 3
rond te lopen	+	omgeving	+	1
marktjes kijken	+	x		

Alice: Hij zegt wel druk, maar geeft er niet echt een oordeel over, dus heb ik niks aangeklikt over dit. Ook zegt hij dat hij wil weten hoe de cultuur is en wat ze op dat plaatje aan het doen zijn. Dus heb ik bezienswaardigheden, cultuur

snuiven en omgeving aangeklikt.
Dave: Waarom omgeving?
Alice: Weet ik eigenlijk niet, is meer een impuls.
Dave: Ok....
Alice: Ja, je moet ook heel snel klikken, dus doe je waarschijnlijk ook wat gevoel erbij betrekken.

Oerwoud*,... Op zich lijkt me wel cool, om samen door het* ***oerwoud*** *te gaan. Ennuh met een* ***gids*** *en* ***wilde dieren*** *kijken misschien met een* ***nachtje slapen****. Dat hoor je ook wel eens, lijkt me wel vet. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
oerwoud	o			
oerwoud	+	oerwoud	+	1
gids	+	klein gezelschap	+	7
wilde dieren	+			
nachtje slapen	+	primitief	+	0

Alice: Hij zegt hier oerwoud, en dan is het lange tijd stil, toen had ik al op oerwoud geklikt. Ik was me hier wel van bewust, dus ben er lang op blijven staan, totdat hij er een oordeel aangaf.
Dave: En waarom plus en geen min in eerste instantie?
Alice: Hoe hij het woord oerwoud zei, kwam positief op mij over en ik had oerwoud al een keer positief aangeklikt. Hij zegt ook gids en nachtje slapen, dus heb ik klein gezelschap en primitef aangeklikt, omdat in een oerwoud mij vrij primitief lijkt en met de gids is vaak met klein gezelschap.

Wat staat er allemaal, ***skie, center****? Tja een keer een* ***drankje*** *hier doen kan geen kwaad. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
skie, center	o	x		
drankje	+	terrasje	+	3

Alice: Hier zeg hij eigenlijk niet veel over, behalve een keer een drankje doen kan geen kwaad, dus heb ik terrasje op plus gezet.

Ja, net zoals die andere foto, mooie ***natuur****, lijkt me het wel mooi om hier over een paar jaar eens rond te lopen. Ik weet ook niet waar we zijn, lijkt wel een beetje Noorwegen of zo. Hebben ze wel mooie* ***natuur****, dus ja... Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
natuur	+	x		
natuur	+	natuur	+	2

Alice: Hier zegt hij opnieuw "misschien over een paar jaar, ik vind mezelf hier te jong voor". Maar weer niet waar hij zich te jong voor voelt, dus heb niks kunnen aanklikken. Natuur vond hij wel mooi, dat zei hij twee keer, dus heb ik die wel op plus gezet .
Dave: Een of twee keer op plus?
Alice: Een keer.

Dave: Waarom een?
Alice: De eerste keer zei hij "mooie natuur", maar kreeg daarbij niet het idee dat hij het leuk vond om daar naar die natuur op vakantie gaan.

Disco! *Ja cool, maar dan niet met je vriendin, maar met mijn vrienden en dan gewoon lekker* ***zuipen*** *en dan er een* ***zomervakantie*** *van maken. Ja lijkt me wel leuk. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
disco	+	uitgaan	+	8
zuipen	+	zuipen	+	2
zomervakantie	+	zon, zee, strand	+	2

Alice: Ik heb "zon zee strand", "zuipen" en "uitgaan" aangeklikt, omdat hij het eigenlijk ook allemaal wel letterlijk zei.

Tja, Nee ***vissen****, ik zie me eigen als dit,* ***duiken****, ook niet echt doen. Lijkt me wel mooi maar ook raar om zo* ***onder water*** *te zitten. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
vissen	-	x		
duiken	-	duiken	-	0
onder water	-	x		

Alice: Ik heb duiken op min gezet, omdat hij daar niet positief over is.

*Lijkt wel op een **resort**, tja op zich, maar wel als je eruit kan. Uit het **resort**. Niet dat je buiten het **resort** neergeknalt wordt, omdat het daar zo gevaarlijk is, dus ja mits ik er ook uit kan. Volgende.*

words by participant	emotion	words by wizard	emotion	time (sec)
resort	+	zon, zee en strand	+	5
resort	o	x		
resort	o	x		

Alice: Ik heb hier alleen zon zee strand aangeklikt, omdat deze vaak bij een resort zijn, en het woord resort staat er niet bij.

*Uhh ja, ik denk dat ik hier nog 20 jaar voor moet wachten. Het is een beetje, tja al die **kindjes**.. nee, hier ben ik nog te jong voor. Volgende.*

words by participant	**emotion**	**words by wizard**	**emotion**	**time (sec)**
kindjes	-	kinderen	-	-3

Alice: Kinderen heb ik weer op min gezet, omdat ik weer kinderen zag op het plaatje en hij weer zei dat hij hier nog te jong voor was.
Dave: En ik zag dat je af en toe ook nog dingen van andere plaatjes leek aan te klikken?
Alice: Klopt, dat waren woorden die ik eerder niet kon vinden en toen ineens zag staan.
Dave: En als je zou moeten aangeven met jouw ervaringen wat er positief en negatief aan dit systeem was?
Alice: Het programma werkt heel makkelijk en was makkelijk te begrijpen. Ook fijn dat ik zelf de plaatjes zelf zag.
Dave: Waarom?
Alice: Omdat je zelf al naar punten kon zoeken die bij het plaatje kunnen horen. Waardoor je ze sneller kon aanklikken als er iets over gezegd werd. Maar misschien heeft het wel mijn beoordelingen beïnvloed, omdat ik zelf mijn mening erin verwerkte.
Dave: Nog meer positieve dingen of negatieve dingen?
Alice: Het is handig te zien wat op plus en min staat en ook als ik er op klik, dat ik zeker weet dat ik hem heb aangeklikt.
Dave: Negatieve dingen dan?
Alice: Ik had zelf de volgorde anders gedaan, omdat ik ze nu niet ordelijk vond staan en achteraf kom ik er achter dat ik een aantal knopjes heb gemist. Ja. En ik miste ook een aantal knopjes, en bijvoorbeeld zon zee strand moeten eigenlijk gesplitst worden.
Dave: We hebben dit 1 keer eerder getraind, had je vaker willen trainen?
Alice: Ja, ik denk dat ik dan de knopjes beter had kunnen vinden en daardoor secuurder had kunnen aanklikken.

Appendix E

Group table

The following table shows the groups. The first column shows the name of the group, the other collumns shows the keywords called by the participant.

Group name	**Keywords called by participant**		
cultuur snuiven	kijken wat daar de bedoeling is	marktjes kijken	rond te lopen
	stammen		
fietsen	fietsvakantie	mounainbiken	
moter rijden	moter	moter rijden	
natuur	natuur	oerwoud	
omgeving	omgeving	uitzicht	
	wilde dieren		
sportvakantie	sportvakantie	sportief	
uitgaan	biertje	cocktailtje	
	nachtleven	drankje	disco
watersport	duiken	onder water	