OTTO-VON-GUERICKE UNIVERSITY MAGDEBURG

FACULTY OF COMPUTER SCIENCE
DEPARTMENT OF SIMULATION AND GRAPHICS

CHILDREN, COMPUTER AND CREATIVITY: USABILITY GUIDELINES FOR DESIGNING A GAME AUTHORING TOOL FOR CHILDREN

MAIZATUL HAYATI BINTI MOHAMAD YATIM

CHILDREN, COMPUTER AND CREATIVITY:

USABILITY GUIDELINES FOR DESIGNING A GAME AUTHORING TOOL FOR CHILDREN

DISSERTATION

for the acquisition of the academic degree

Doktoringenieur (Dr. –Ing.)

issue by the Faculty of Computer Science
Otto-von-Guericke University Magdeburg
to

M.Sc. (IT) Maizatul Hayati Binti Mohamad Yatim
born on 1 January 1974 in Perak, Malaysia

Supervision Examiners:
Jun. Prof. Dr. Raimund Dachselt
Prof. Dr. Maic Masuch
Prof. Dr.Wolfgang Broll

Place and date of graduation colloquium:
Magdeburg, 12 October 2009

CHILDREN, COMPUTER AND CREATIVITY:
USABILITY GUIDELINES FOR DESIGNING A GAME AUTHORING TOOL FOR CHILDREN

DISSERTATION

zur Erlangung des akademischen Grades

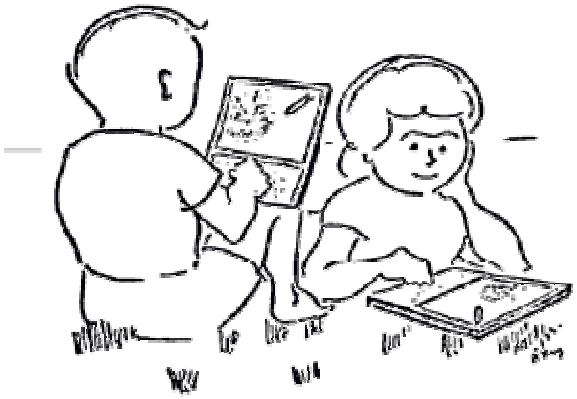
Doktoringenieur (Dr. –Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg
von

M.Sc. (IT) Maizatul Hayati Binti Mohamad Yatim
Geboren am 1. Januar 1974 in Perak, Malaysia

Gutachter:
Jun. Prof. Dr. Raimund Dachselt
Prof. Dr. Maic Masuch
Prof. Dr. Wolfgang Broll

Ort und Datum des Promotionskolloquiums:
Magdeburg, 12 October 2009



The best way to predict the future is to invent it…Alan Kay
Illustration of children using the Dynabook originated from Kay's paper [Kay72].

Special thanks to my parents to whom I dedicate this work.
Mohamad Bin Mat Diah & Hamidah Binti Mohd Nor
Mohamad Yatim Bin Tajid & Rogayah @ Kamariah Binti Ahmad
Haji Fisal Bin Haji Othman & Hajah Che Bebi Binti Haji Sulong

Abstract

The thesis is based on implementing a child-centered design approach in the creation of a game authoring tool for children to be used in the classroom as one of their learning tools. Indeed, research in designing such educational software is primarily focused on the learning outcomes, but not on the usability. However, the usability of such software for children is a prerequisite for any learning to be useful and significant. The thesis purposes with a belief that substantial improvements can be made by paying careful attention to usability during the design of the game authoring tool. Unlike the related approaches, the scope of research focuses on the game authoring tool which emphasizes on the design of the game authoring tool and concentrates on the flexibility of the interaction and construction activity in making games. This research methods started by investigating works from other researchers including comparing the usability guidelines used in other fields and exploring the existing educational programming tools for children. The findings exposed the possibility of planning and designing guidelines for usability and limitations or problems faced in these tools. The outcomes were also used to guide the first draft of usability guidelines. To demonstrate and validate these guidelines, a preliminary study was conducted. The study consists of two user studies conducted in two different locations with the goal, to get as much information as possible in childrens' preferences and needs based on the usage of two given software. Informed by these results, the prototype of a game authoring tool named Gatelock was designed. There, design features were used – game programming, game designing and game playing. In game programming, the visual programming method is being carried out. The method is being implemented in a variety of application software for children, especially in the field of interaction design for children. In game design, children use the basic elements of game design in their game making, including planning the games, designing, building, gameplay testing, and re-design. In game playing, the children can participate in the design processes with learning material and they can incorporate elements of various lessons such as physics, mathematics, science and drawing. Throughout the design and development of Gatelock, formative and summative evaluations were conducted to examine the usability and fun aspects of Gatelock. This thesis is a contribution to the scientific domains of child-centered design. It draws on ideas and techniques from the fields of children and interaction design, usability study, game design, as well as evaluating software for children. With respect to game design and children, the main contribution of the thesis can be specified as product-based and process-based. Product-based contributions include a prototype (Gatelock) which has a unique set of features due to its child-centered design, and a list of usability guidelines for designing a game authoring tool for children. Process-based contributions include a preliminary study of creating a new game authoring tool with usability as the first objective, as well as empirical evidences characterizing the preliminary study with results made by children, formative and summative studies which demonstrate the usability of the game authoring tool, empirical evidences on the usefulness of methods, tools and techniques used for the evaluating process with children, and a list of usability guidelines for designing a game authoring tool for children. This thesis also presents the results of the research with the presence of usability guidelines. At the end, both, the game authoring tool and the usability guidelines are iteratively involved in the design refinement.

Zusammenfassung

Diese Doktorarbeit setzt einen auf Kinder ausgerichteten Design-Ansatz bei der Entwicklung eines Spiele-Autorenwerkzeuges für Kinder um, wobei das Autorenwerkzeug als ein mögliches Lerninstrument in Schulklassen genutzt werden soll. Bisher ist die Forschung bei der Entwicklung solcher Lernsoftware im Wesentlichen auf die Lernergebnisse ausgerichtet, jedoch nicht auf die Benutzerfreundlichkeit. Die Benutzerfreundlichkeit von Software für Kinder ist jedoch die Voraussetzung für sinnvolles und nachhaltiges Lernen. In der Doktorarbeit wird die Überzeugung vertreten, dass durch die sorgfältige Beachtung der Benutzerfreundlichkeit während der Entwicklung eines für Spiele vorgesehenen Autorenwerkzeuges wesentliche Verbesserungen erzielt werden können. Im Gegensatz zu ähnlichen Ansätzen konzentriert sich der Schwerpunkt dieser Forschung auf das Design des Autorenwerkzeuges und auf die Flexibilität der Interaktions- und Erstellungsabläufe bei der Entwicklung von Spielen. Die vorliegende Arbeit widmet sich zunächst der Betrachtung von existierenden Forschungsarbeiten, der Untersuchung von vergleichbaren, in anderen Bereichen verwendeten Usability Guidelines sowie der Untersuchung existierender pädagogischer Programmierwerkzeuge für Kinder. Diese Analyse erlaubte die Identifikation von Möglichkeiten, Richtlinien für benutzerfreundliche Software zu entwerfen und Einschränkungen bzw. Probleme, die bei diesen Werkzeugen festgestellt wurden, zu beheben. Die Analyseergebnisse konnten ebenfalls dazu genutzt werden, eine erste Fassung der eigenen Usability Guidelines zu erstellen. Zur Veranschaulichung und Validierung dieser Richtlinien wurde eine Vorstudie durchgeführt. Diese setzt sich aus zwei, in verschiedenen Ländern durchgeführten, Benutzerstudien zusammen, die unter Nutzung zweier existierender Software-Werkzeuge darauf abzielten, so viel Informationen wie möglich über die Vorlieben und Bedürfnisse von Kindern zu erhalten. Anhand dieser Ergebnisse wurde der Prototyp des eigenen Spiele-Autorenwerkzeuges *Gatelock* entwickelt. Dabei wurden Design-Aspekte wie Spieleprogrammierung, Spielentwurf und das Spielen bzw. Testen der Spiele berücksichtigt. Für die Spieleprogrammierung kommt eine Variante des *Visual Programming* zum Einsatz. Diese Technik wurde bereits in zahlreichen anderen Programmen für Kinder mit Erfolg verwendet, insbesondere im Bereich Interaction Design. Für den Entwurf von Spielen können Kinder typische Grundelemente des Game Designs zur Erstellung eines Spiels, inklusive der Planung, des Designs, des Aufbaus, des Testens des Spielflusses sowie der Neugestaltung nutzen. Während des Spielens bzw. Testens können sich die Kinder mit Lernmaterialien an den Entwicklungsprozessen beteiligen und Elemente verschiedener Unterrichtsfächer, wie z.B. Physik, Mathematik, Naturwissenschaften und Kunst einfließen lassen. Während des gesamten Entwurfs und der Entwicklung von Gatelock wurden formative und summative Evaluationen durchgeführt, um die Benutzerfreundlichkeit und die Spaßfaktoren von Gatelock zu untersuchen.

Acknowledgement

My deepest thanks are due to my supervisor Prof. Dr. -Ing. Raimund Dachselt who encourages me to pursue my own research interest and guides me into the field of software engineering with his careful supervision and timeless support. I owe my sincere thanks to Prof. Dr. -Ing Maic Masuch for his guidances in the field of game design. My special thanks go to my third examiner, Prof. Dr. Wolfgang Broll for taking the time to review the thesis.

I would like to thank Niklas Röber for his help, advice and continuous support through this undertaking. I would like also to thank Lennart Nacke who pointed out interesting discussion in his master thesis and for the tireless support in this research. My special thanks to Marco Schmidt and Sergej Roth who developed the game authoring tool used in my major study and also for their interesting discussion and ideas. I really appreciate these wonderful people for helping me in preparing and also conducting the evaluation study – Henry Sonnet, Franciska Krueger, Jana Sieber, Michael Freisleben, Amily Shafila Shariff, Nur Saadah Fathil, Azniah Ismail, Syamimah Abdul Hamid, Masyarah Zulhaida Masmuzidin, Maryati Saad, Nurul Amalni Abdul Wahab, Rozana Zainon, Faridza Fisal and Fauzie Fisal. I also would like to express my gratitude towards the supports and encouragement from the former members of Games Group and also current members of the User Interface and Software Engineering Group.

I acknowledge the careful work of Stefanie Quade, who has skilfully revised the language and reviewing the present manuscript. All staff in the Department of Simulation and Graphics, Otto-von-Guericke University Magdeburg especially Petra Schumann and Petra Specht, deserves a special mention for their excellent secretarial help, continuous support and friendship during these years. Not to forget, all the technicians, ISG friends and colleagues for their warm-welcome and full support towards my stay.

I would also like to thank numerous children and teachers who involved in usability test. I really appreciate for the support and excellent, friendly, and valuable criticism and patience in guiding me in scientific research and thinking which improved this thesis, as our interesting conversations have been of great importance during these years.

Lastly, I would like to thank my friends and family for their unconditional love and support. Not forgetting, I owe my loving thanks to my dearest husband Fauzie Bin Fisal, who has shown his interest in my study and share both good and bad time with me. Your unlimited love, understanding and excellent humour has helped me overcome all hardships during these years.

This study has been supported financially by the Ministry of Higher Education, Malaysia and technically by Ministry of Education, Malaysia. All these are gratefully acknowledged.

Table of Contents

Appendixes

References

Acronyms

List of Figures

List of Tables

1
Introduction

As a multi-disciplinary field, *human-computer interaction* (HCI) is concerned with how humans interact with computer technology. It draws its theories and models from ergonomics, psychology, social science and computer science and has traditionally been concerned with the use of systems for work. For the purpose of children as ultimate users, another field has widely become known as *child-computer interaction* (CCI) which is meant to be the subset of the HCI field but also has the need to rely on theories from child psychology, education and game research [Read05a] [Read05b]. Since it is a new discipline, many unresolved issues are still in work, particularly in respect of the methods and techniques used and implemented in HCI to be adopted in CCI, involving children in the design and evaluation of their own software, and the acceptance of opinions and judgements from children concerning the software they interact with [Hanna97][Donker01]. These unresolved issues and questions related to them were the main motivations to establish this research in the field of children and interaction design.

This research combines theories and techniques from the fields of usability study, interaction design and children, and game design. It aims to find a usable and extensible usability guideline for the development of a game authoring tool. It is also derived from a specific interest in how the tool might be deployed by children for learning and gaining computer literacy. The work is expected to deliver both a product-based (a game authoring tool and evaluation instruments) and a process-based approach (a set of usability guidelines) by developing a similar game authoring tool for children that focuses on fun in the learning environment.

1.1 Motivation

First, other than the aspect of children being involved in software design and development, children's software applications are simply developed without looking at the direction of usability engineering which is aimed at the effectiveness and efficiency of the software and the user satisfaction. However, research on this matter is relatively scarce and studies with children in particular areas are rare [Read06]. Prominent guidelines such as Web usability guidelines [Nielsen94], interface design guidelines [Shneiderman97], interface design guidelines by Microsoft [Microsoft95] and Macintosh [Apple92], and a list of guidelines on how to conduct usability tests with children [Hanna97] have been carried out and tested, but none of them specifically aims for a game authoring tool with children as users [Yatim07b]. Nevertheless, related works in this research field are presented in several studies. One work refers to a study that propagates 70 guidelines for a Web-based user interface design for children [Gilutz02], while other studies refer to the development of programming environments such as Icicle and HANDS. In the Icicle programming environment, the designer established a list of guidelines for developing a programming environment for children [Sheehan03], while the designer of HANDS concentrated on the aspects of HCI and usability throughout the development [Pane02].

Overall, the usability guidelines existing today are not sufficiently catered to children's software [Barendgret06][Read06][MacFarlane05b]. They should always be included among the criteria that are considered during the design of any system. Depending on the constraints of a particular project and target audience, usability may be given more or less weight. However, it is always worth considering for at least those decisions that are not already determined by other design criteria. Addressing these issues, the thesis demands to go from design concepts in building necessary technologies, to putting together the research approach and conducting studies to learn from the process. In the end, a list of usability guidelines for designing a game authoring tool for children will be presented. The usability guidelines constructed throughout this research could be helpful for the design of other game authoring tools, or could be used in other software for children. The usability guidelines can be used in two ways, either as a benchmark to evaluate the usage of children's software [Nielsen07] or as a checklist to design children's software [Resnick05]. Together with the HCI theories, principles and standards, the designed guidelines can help designers and researchers to produce good or better software products [Yatim07d], and at the end they will support and consolidate the usability guidelines designed in this research.

Second, when the thesis work began, there had been a huge popular interest in integrating children with software used for designing or constructing activities. Previous research was done by researchers in the field of children and interaction design [Harel90][Kafai96b][Robertson01]. These scholars agreed upon one thing: Learning can take place by doing and practicing with the design activities. The idea came from the constructivist theory approach which originated from Papert's ideas and works [Papert80]. The works focus on the aspects of children using educational software such as programming or simulation software, except for the work of Kafai who focused on computer game design with Logo programming for children's personal and creative expression [Kafai96b]. Nevertheless, there are still a number of unanswered questions in terms of game design software technologies for children.

Figure 1.1 Edutainment software products. (*Left*) The first title of the *Math Blaster* series was released in 1994 by Davidson and Associates. (*Middle*) *Typing of the Dead* was released in 1999. (*Right*) The first Carmen Sandiego software game, *Where in the World is Carmen Sandiego?* was released in 1985.

Developers of educational software often used games as a reward for children upon completion of the given learning objectives [Kaiser02]. The early educational software title once known as *edutainment* became synonymous with the *chocolate-covered-broccoli* or *eye-candy* approach to gaming [Bruckman97][Jegers01]. This approach defined the edutainment era where more edutainment titles such as *Math Blaster*, *Typing of the Dead* and *Where in the World is Carmen Sandiego*, as shown in Figure 1.1**,** dominated the games market. On top of that, in the educational field a debate came up about the use of computer games

In other words, edutainment titles received a bad reputation and led to a great crash [Egenfeldt-Nielsen05]. It formulates the boring learning lessons and materials (mathematical worksheets, in this case) within a fun gaming environment [Edwards01]. With small development budgets, less innovation, and a lower quality of edutainment titles, the edutainment market faces a decreasing consumer interest [Dyck03][Muda05].

The edutainment approach to integrate games and learning has reported as not working and has been critizised for combining the worst aspects of both the educational and the entertainment field [Habgood07][Kerawalla05] [Trushell01]. The main reason for this is the capability of the software itself in teaching the users. Although this type of software is very common and can be easily found on the Web or on any CD-ROM format, it is usually inappropriate for long-term learning especially for children, since they have no control over its creation [Prensky06]. Most educational software provides pre-programmed scenarios and tells children what to do. The children are only being informed instead of directly and actively getting involved with the software within their own learning pace. In the end, the educational software does not provide children with creative outlets [Peppler05]. Due to this, some people are reluctant to use the word *edutainment* even though the content delivery is just the same, but yet change it to other terms such as *playful learning* [Resnick04] and *digital game-based learning* [Prensky01]. The terms refer the edutainment field to a new perspective where games are not totally designed to present the learning aspects but also primarily used to reinforce the understanding of presented material and to add a variety in learning.

.

> *"I prefer to focus on play and learning (things that you do) rather than entertainment and education (things that others provide for you). It might seem like a small change, but the words we use can make a big difference in how we think and what we do*".
>
> *Mitchel Resnick (2004), page 1.*

> *"The premise behind Digital Game-Based Learning is that it is possible to combine computer and video games with a wide variety of educational content, achieving as good or better results as through traditional learning methods in the process*".
>
> *Marc Prensky (2001), page 145.*

In a new era of game development, game designers have started to examine the future course of games and started to integrate pedagogy aspects directly in the game play. This time, the term used is *serious games* [Michael05]. The serious games initiative aims at the future of game design and development in the area of education and training, government, healthcare, military simulation, business and corporate. These games are designed and developed for the purpose to educate, inform and train people who want to do things more seriously than simply having fun. Serious games are considered as the next wave of technology-mediated learning which offer a powerful and effective approach to learning and skill development in today's workforce [Bergman05][Derryberry07]. Why does the game design movement aim to be serious about games? Do serious games really promote learning? What do learners, including children, think about serious games? Several reviews of literature have been conducted regarding the educational gaming [Wolfe98][Randel92][Cavallari92], but still lack of research on the effectiveness of serious games as learning tools [Egenfeldt-Nielsen06]. Some findings of the learning outcomes from playing games are positively promising, but others are sceptical with the aspects of methodology flaws and contradictory results [Kirriemuir03]. Therefore, the thesis aims to seek the answers

by applying games outside the realm of a 'normal' games-related initiative. In other words, instead of searching the development and social impact of children playing games, the thesis offers findings on the impact of children designing games using a game authoring tool. It is assumed that game design activities can offer a new teaching and learning tool for children, but the processes or impacts have not been investigated yet. Ideally, such software should provide a medium for children to express themselves in ways that would not be possible without a computer. As a result, an investigation with a proper research method is needed to identify the underlying factors that contribute to the problem and to find the ways and means to address it.

Third, another aspect of interest that could derive from this thesis are the promising aspects of *learning by doing* and *by thinking about what they do*, which can improve the questions of *what* and *how* children learn [Papert80]. Papert suggests that technology is an ideal way to make practical use of the theories on learning by doing as suggested by Dewey and Piaget [Ackermann09][Papert93]. For this research, the most significant question is whether the game design activity would make the computer access easier for children to facilitate learning, or if it fails to provide any of it. The *constructionism theory* is grounded in the idea that people learn by actively constructing new knowledge, rather than having information poured into their heads [Clements99]. It is believed that children can use and capitalize on the benefits of technology by using integrated computer activities to increase their achievements. The computer seems to have the potential to offer children a wealth of opportunities to express themselves and to learn. As a creative medium, computers can help children express their imagination in ways that are not possible with other materials [Peppler05]. Another issue is that sophisticated software is not learned by osmosis or even by repeated use [Kafai96b]. Children acquiring computer skills still need formal instruction to master sophisticated applications and keep their skills updated.

Learning to program is also a valuable part of a general education for children [Papert99]. In addition to being a nice introduction to structured problem solving, programming also enables children to gain experience with complex systems and provides them with computational thinking skills that can be applied to a broad range of disciplines from sciences to mathematical contents [Harel90][Kafai96b][Guzdial01a]. Programming provides children with some hands-on experience dealing with complex systems that they create themselves [Seth96][Baldwin00]. When their programs do not behave as expected, children have to learn to isolate the problems and to solve them. They learn to narrow the scope of a problem and that a single malfunctioning program component can cause other program components to malfunction as well. In addition to the critical thinking skills that children develop through programming, an understanding of computer programming may prove to be a valuable job skill for many people nowadays. The thesis shows a promising area of the game design and development field in engaging children towards learning. This approach is referred to as *intrinsic integration* [Malone87] which combines a playful approach with a constructive activity [Dix03]. Intrinsic integration deals with delivering learning content with the flow of experience of a game [Kafai01][Habgood07].

This section reviews three motivational issues around the research. Regarding the usability guidelines, the research will propose a list of guidelines not only to be used for designing a game authoring tool, but also to be used for other children's software. In the games field, the research will investigate the integration of computer games technology within the educational field, not only for playing a game, but also for designing it. In the field of

learning by designing, the research will incorporate the constructive designing game activity into a meaningful educational lesson that increases the learning motivation. These issues are aimed to provide a contribution to extend the capabilities of finding and practicing the usability guidelines in designing a game authoring tool for children. This research is subject to a more serious study within the disciplines of child psychology, education, computer science and sociology, all in the name of learning. However, these issues are too broad and go beyond the scope of this thesis. Therefore, the research only complies with the basic concepts and integrates them into the goal of the research, which is explained in the next section. The ability to grasp these broad and multidisciplinary areas is critical, as it represents the fundamentals in seeking usability guidelines that deal with the construction of a game authoring tool for children to be used for designing games. In addition, even though this thesis is based on education and learning, it also incorporates related perspectives of the game development community itself.

1.2 Research Goal, Objectives and Approach

The goal of this research was *to develop constructed and refined usability guidelines of a game authoring tool.* To achieve this goal, the idea was *to design a game authoring tool for children iteratively while constructing and refining the usability guidelines*. Three main objectives were specified:

- To explore the possibility of a game authoring tool and its technology for children, specifically to facilitate learning in the classroom;
- To identify the usability problems and fun aspects that might arise from the game authoring tool design and development cycle; and
- To suggests usability guidelines as a benchmark for designing and developing a game authoring tool for children.

According to the aforementioned, the research goes into three major directions, which are (1) usability for designing a game authoring tool, (2) children and game authoring tool, and (3) interaction design and children. In this research, a design process for developing a game authoring tool for children is created, whereas usability is treated as a main objective. It is assumed that the use of a list of usability guidelines would *improve usability and fun aspects* of a game authoring tool for children. Finally, the use of a game authoring tool would convey *computer literacy skills* for children including thinking, learning and programming skills. The research is not only about testing these assumptions, but also to provide some insight into each of them. The main question guiding this research was *how the process of developing and evaluating a game authoring tool and a construction of usability guidelines can facilitate children's computer literacy and learning skills*. To answer this main question, more questions were formulated and are listed below:

- Can the usability guidelines really show the usability problems that occur in the game authoring tool?
- Do the elements in the game authoring tool need to be refined to better accommodate learning on how to design games for children?
- Do satisfaction and fun factors have a close relationship with usability in the game authoring tool?

With such questions in mind, a game authoring tool for children was designed with a focus on children as main users. By all means, its features should match the requirements of children including the user interface, the storage system, the framework system, the way the program is integrated into the system, and even the language that the children can use to produce their own programs. The comparable existing systems are going to be explained in Chapter 2. But it is definitely too much work needed to develop a system of that extent. Most existing systems were the product of many years of development by an extensive team of designers and programmers. After investigating some alternatives, only the *programming capability* and *interaction components* are focused as the basis for this research. This is due to the fact that game programming is the main design activity which cannot be realized without a computer [Pleva04]. Other tasks, such as drawing game characters or avatars, music and audios etc. can be accomplished by integrating with any application tools.

The target audience for this research are children between seven and twelve years. The age range was chosen, because these children (also known as *middle childhood*) comply with the Theory of Cognitive Development [Piaget83]. Children of this age use *logical operations* when solving problems. The cognitive development stages are taken into consideration in this research to understand how children learn and think. Although a child's cognitive development is not a major part of the research, this point will be briefly explored in the next chapter. Secondly, children in the selected age range often have a *high interest* in learning how to program [Mano04][Robertson04]. They have creative and ambitious goals for their program making that are similar to the applications they use especially in games and simulations which are well-known for their graphically rich and highly interactive activity. It is also important to know how children use and work with software tools including the degree to which the learning matches the childrens' capabilities and their ability to involve in evaluating the tools [Hanna04]. The final reason is that they comply with the *age classification standard* set by the game industry [Bijvank09][Pan09][Unterhaltungs09]. The age rating is primarily concerned with the game rating system, game content descriptors, and as guidance for parents when buying games. All these factors act in accordance with the direction of this research.

Next, the approach of this research will be explained by referring to Figure 1.2. There are four main components of this research which are the requirements analysis, initial study, prototype development, and evaluation study. First of all, the research started with *surveys and investigating works* done by other researchers, including comparing the usability guidelines used in other fields and exploring the existing educational programming tools for children. The aim was to find a possibility of planning and designing guidelines for usability and also to find the limitations or problems faced in these tools. The outcomes were used to guide the first draft of usability guidelines. The second component was the realization of the *preliminary study,* which comprises of two user studies known as User Study 1 and User Study 2. At the end of both user studies, the children received questionnaires. These questionnaires can help to find problems in the existing tools and validate the earlier draft of the usability guidelines. The third component of the research approach is the design of *a game authoring tool* named Gatelock. The tool was designed and developed according to the guidelines established based on theoretical studies and user studies. The last component of the research approach is the *evaluation* phase. The game authoring tool was used and tested by selected children in two separate evaluation studies: the formative evaluation and the summative evaluation. As shown in the same figure, the usability guidelines have been revised constantly throughout the design and the development of the intended game

authoring tool. It is understood that the guidelines do not specify exactly what a game authoring environment for children should look like and how it should work. But they do offer help when choices need to be made on how something should work or appear.

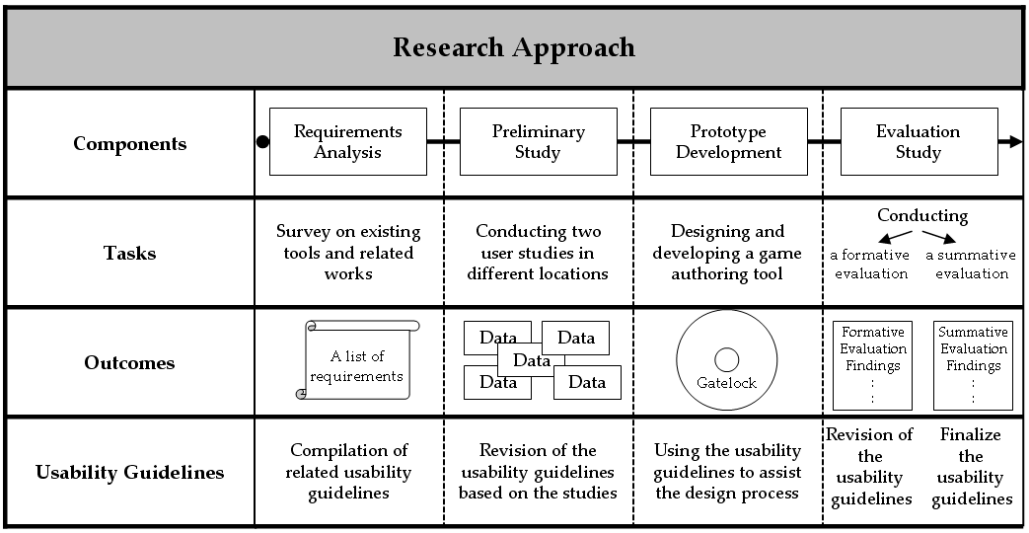


Figure 1.2 The research approach includes four main components. For each component, several tasks are identified, and their major outcomes or deliverables are defined. They indicate the completion of each task. The usability guidelines have been created and revised respectively based on the findings from each task.

The research approach is constructed in a way that is believed to provide solutions for three important issues. First, by testing several existing or related tools and having a list of usability guidelines before designing the prototype, common problems during the software development such as poor requirements, an unrealistic schedule and inadequate testing can be avoided or minimized. Second, by conducting a preliminary study before designing the prototype, the children are understood by the designers and researchers in terms of their cognitive behavior and attitude characteristics. Third, by having an iterative design of prototype, common problems found in the usability studies can be fixed. This means, there must be a cycle of designing, testing, measuring and redesigning that is repeated as often as necessary. Other than that, the research approach also follows the field of child-centered design (CCD) which was derived from the user-centered design (UCD) philosophy. There is a strong direction in children's technology design as applied research including the evaluation of the developed software product [Jensen05]. CCD provides a number of opportunities in the integration of children, designers and researchers for different perspectives on children's use of technology. This includes understanding the framework used in CCD, focusing on the development of new systems or parts of systems, redeveloping existing systems or parts of systems, and assessing or validating the systems for evaluation purposes. Due to the complexity of the field, this research will focus on the related theoretical knowledge in CCD including the evaluation process and procedures.

Several challenges have been identified throughout this research while using the approach. These challenges can be grouped into four aspects, which are engagement and learning, the design and development of the prototype, the procedure and methods of conducting evaluation studies, and the process of outlining the usability guidelines iteratively. Most of

the challenges and problems have been solved throughout the research, while some have not been solved yet and are treated as future work. The challenges and problems will mainly be discussed in the last two chapters of this thesis; nevertheless, they are also going to be discussed in the respective chapters.

1.3 Research Contributions

This thesis presents the results of the research on designing a game authoring tool for children with the presence of usability guidelines. Therefore, the whole research contributes to the usability for designing a game authoring tool, children and game authoring tools, and interaction design and children. More importantly, through the development process of this research the game authoring tool can conform to children's needs and desires by using a set of usability guidelines. Finally, the outcomes can be summarized as technologies, applications and usability guidelines aimed at supporting children's creativity and learning. By integrating these activities, the specific research contributions are:

- a preliminary study of creating a new game authoring tool where usability is the first objective;
- empirical evidences from the preliminary study with children, which can be used to help designers generate the initial design requirements and select the appropriate authoring features;
- a game authoring tool prototype which has a unique set of features due to its child-centered design;
- formative and summative studies which demonstrate the usability of the game authoring tool;
- empirical evidences on the usefulness of methods, tools and techniques used for the evaluation process with children; and
- a list of usability guidelines for designing a game authoring tool for children.

1.4 Structure of the Thesis

The remaining chapters of this thesis are organized as follows:

Chapter 2 introduces the reader to the state of the art in the field of usability and the field of children and interaction design. The discussion starts with an overview of the existing usability guidelines, including an explanation on why current guidelines need to be more focused on children`'s software. The next part of the chapter justifies the study of usability especially focusing on the children. Then, the connection between usability and game design is explained with the premise of fun, flow theory and the field of child-centered design. The chapter continues with empirical work on a survey of educational programming environments for children including some remarks on the difference between simulation tools and game authoring tools. Three game authoring tools for children have been selected as example and have been described regarding their programming concepts, features and capabilities. The findings will be interpreted and discussed using this knowledge for creating the first version of usability guidelines.

Chapter 3 describes the preliminary study that comprises of two user studies examining different software used by children. The chapter discusses the studies thoroughly by looking at the possibility of using a simulation tool (Squeak Etoys) and an authoring tool (Game Maker) to facilitate children's learning. The findings of these studies are helpful in refining the usability guidelines drafted earlier. The chapter ends with discussions and an outlook on the study as the basis for designing the earliest usability guidelines which will also be used for designing a game authoring tool prototype.

Chapter 4 details the design and development of the game authoring tool called Gatelock. This chapter will not only focus on the technical aspect of its design and development, but also concentrate on the implementation of Gatelock. This includes software requirements, system models, the visual representation used for the user interface and object-oriented design.

Chapter 5 describes more studies to evaluate the features and functionalities of Gatelock, as well as empirically derived evidences with respect to the children's abilities in using Gatelock (formative and summative evaluation). The findings do not only include usability and fun, but also engagement and learning as well as the field of children and interaction design.

Chapter 6 discusses the achievements and limitations of Gatelock. This includes the capabilities that Gatelock does not have, and suggestions on its improvement. It also summarizes the reflections on the usability guidelines that have been involved in several iterative processes. This chapter ends with a discussion on the lessons learned in conducting the usability studies with children.

Chapter 7 draws and summarizes the contributions of this thesis from discussions about the major findings of the research and thoughts on the use of the game authoring tool for children. This chapter concludes by outlining additional future directions in designing the game authoring tool, and by implications on its usage in the classroom.

2
Related Work

Usability has been a buzzword among HCI practitioners and academias in recent years. It is at the center of the whole HCI phenomenon where many of the HCI processes and methods have been developed around this concept. However, in consideration of children's software, none of the usability aspects touch on construction software for children, such as a game authoring tool. In this chapter, the basic and related concept concerning usability and children's software technology is explained and discussed. The first part of the chapter will explain the prominent standard of usability for software including the reasons why the existing usability guidelines are still insufficient and why these guidelines need to be considered when developing or evaluating software. Here, related theories and techniques from the perspective of the usability study involving examples from the selected existing usability guidelines will be clarified. The second part involves the usability guidelines for children. It is crucial to have different usability guidelines for different types of users, especially with regard to children [Barendgret04]. Here, questions on why children should also be involved in usability research will be rationalized including its implications on this research. The third part contains the aspects of fun and flow theory. All these parts are based on the premise that the elements of enjoyment are universal [Monk02], providing a general model that summarizes the common concepts when experiencing enjoyment. The fourth part of this chapter refers to the surge of popular interest in children designing games. Since this research contributes to the field of game design, the discussion will be towards works carried out by respective researchers including child psychologists and constructivists especially concerning the idea of learning by doing and learning by programming [Roussou04][Rowland05]. The reasons for choosing game play and game design as directions of this research and for future design work of a game authoring tool will be explained. A survey on educational programming environments (EPEs) for children will be conducted to understand the current situation and problems faced. Therefore, three well-known game authoring tools are selected as examples and explained regarding the aspects of their programming concepts, features and capabilities. The chapter ends with a depth clarification of the research approach which focuses on the child-centered design (CCD) in designing, developing, using and evaluating children's software.

2.1 Existing Usability Guidelines

Usability is an important factor in establishing if educational software will facilitate the acquisition of knowledge. In 1982, the first conference on computer usability took place in Maryland (USA) and three years later, a committee of the Human Factors and Ergonomics Society set up to publish the first standards for human-computer interface design. The project took over 15 years to complete, and in 1998 the International Organization for Standardization (ISO) published ISO 9421, a standard guideline for visual displays for office work. In addition, ISO 9241-11 was introduced, which defines usability as the extent to which a product can be used by specific users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use [ISO98]. If users perceive that a system is very difficult to use, the perception may influence their ability to absorb material provided by the system.

Effectiveness is often measured by reduction of mistakes or errors that users make. A presumption is that such errors indicate problems in the design of a software product. Effectiveness indicates the accuracy and willingness of users to achieve a set of goals.

Efficiency equates with expanding the least amount of resources to complete an end goal. While iterative rounds of usability tests help identify problems with a software design and contribute to its improvement during the development process, such results do not imply that the software is effective in helping users to accomplish their goals with the software.

User satisfaction involves multiple dimensional concepts including an immersive environment and compelling experiences. But measuring user satisfaction is hard to accomplish, since the element of satisfaction involves a progression towards the achievement of goals which may vary considerably between different users.

With the definition given by ISO9241-11, finally the HCI field had substantial guidelines for the definition of usability and its relation with the interface design methodology. One of the most famous usability methods is heuristic evaluation [Nielsen94]. A list of ten originally developed heuristic evaluations was aimed to ensure that usability is taken into account in general interfaces [Nielsen90]. However, Nielsen's ten revised heuristics, as popular and general as they are "*do not stretch far enough to fully cover educational softw*are" [Nielsen00]. In order for educational software to be successful, all usability problems must be addressed before students have access to them. This must be ensured so that students can fully concentrate on the learning tasks, without their attention being diverted to usability problems of the system. Shneiderman's Eight Golden Rules of Interface Design are also a guide to good interaction design [Shneiderman97]. The guidelines derived heuristically from experiences and were believed to be applicable to most interactive systems. In his recent work, Shneiderman states that software designers must "*offer usability plus reliability to prevent frustration from undermining fun and engage users with fun-features"* [Shneiderman04]. This means, the golden rules need to be refined and extended according to the type of software developed by designers. Another prominent work about usability guidelines was established according to the principles of learnability, flexibility and robustness, which were further classified into sub-categories [Dix98]. However, these guidelines "*do not employ any category referring to user satisfaction or related notions"* [Wiberg03]. Prominent standards for usability guidelines, as discussed before, have shown that the existing guidelines are still insufficient and more work needs to be exercised to interpret the most recent usability guidelines. In another work, a set of specific guidelines for the evaluation of educational software was produced [Costabile07]. It was believed that educational software must not overwhelm the students. In order for students to be fully engaged in the learning content, the interfaces must be well designed, concentrate on the learner' needs and goals, provide a clear idea of content organization and system functionalities, and have a simple navigation, as well as advanced personalization of paths and processes [McFarlane02]. However, the issues of students' *engagement and enjoyment* in either playing or designing games are still lacking [Hull03][Oblinger04][Beavis05][Ang08].

Why do usability guidelines need to be considered in developing or evaluating software? It is important to acknowledge the main and real focus of usability throughout the design process. A poor system design can result in a system failure due to *no use, under use or*

incorrect use [Shneiderman02]. Software with a poorly designed interface can have inhibiting consequences. This means that the guidelines *serve as recommendations from good* practice [Stewart03]. The suggestions are mainly to improve the *user experience* with the software used and to generate predictions towards *user performance*. If these guidelines are treated as serious business, they can become a standard in HCI theories and principles, such as Fitt's Law. The standard in HCI can be formality (legal, e.g. ISO 9241), proprietary (company or manufacturer, e.g. Microsoft [Microsoft95] or Macintosh [Apple92]), and in-house (style guidelines developed by organizations for their own internal use, e.g. Google [Google09]). Despite the different aims and functions of educational software, the evaluation of software *differs according to actual users* [Baauw05]. For adults, understandability, familiarity, ease of performing small tasks, and user interface are among the usability heuristics [Stewart03]. These heuristics are more important features in any software than technical objectives such as mathematical elegance, efficiency, verifiability, or uniformity. These heuristics are commonly known and used in the field of HCI. But using the existing heuristic evaluation for children's products is more difficult, at least for two reasons. Firstly, the experts are usually adults who are very different in obvious ways from the intended users, and therefore find it harder to say whether a particular issue is going to be a problem for the users [Druin94][MacFarlane05a]. Secondly, constructing such heuristic usability may be interpreted differently children and adults [Druin99a][Prince04]. In other words, heuristics designed for adults' products may not be appropriate for children's products.

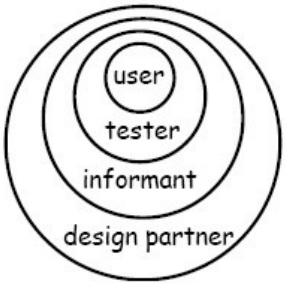


Figure 2.1 Four roles of children in the design of software [Druin99a].

Regardless of the enormous guidelines for software designed, products for children are also not exceptional. What is so special about children and why do products for children need specific usability guidelines? With an abundance of commercial software designed for children, a wide concern among those who care about the usefulness of software for children started to emerge. In fact, people consistently expressed their concerns regarding inappropriate software for children. Furthermore, most educators expressed their major concern with the problem of *selecting good and appropriate software* for children to be used in the classroom to facilitate learning [Haugland97][Gorp01][Zaman05a]. Since the computer has become an influential tool to enhance the potential of children and to facilitate their learning process, people who are directly or indirectly involved with technology for children should carefully evaluate these technologies before they reach the children [Halverson05]. The definition of usability itself is mainly focused on applications for adults or mature users. Children are different from adults. Their motivations are different and they have different desires and expectations [Habgood05]. Many products for children are weak in addressing *specific developmental needs and skills* of their intended users [Gelderblom04][Wyeth03]. For testing the usability of children's software, a classification of the involvement of children was created. Children can play four roles in the design of software: They are users, testers,

informants or design partners [Druin99a], see Figure 2.1. These roles involve different levels of engagement and enforce different opportunities and limitations. The *segregation of children's roles* also means different or similar usability test methods that can be used to measure products for children.

In terms of the age, as mentioned in Chapter 1, the target audience for this research are children between seven and twelve years. Finding the right *age range is crucial* when doing research with children referring to their cognitive abilities and social skills [Druin02a][Egloff04]. Children as young as seven or eight can *reliably differentiate* between constructs such as ease of use and fun aspects of an interface [Jegers03]. Children in this age range are also relatively easy to include in software usability testing. According to the Theory of Cognitive Development [Piaget83] they are *able to follow* a task by specific directions with a higher attention span. These children use *logical operations* when solving problems and appreciate new levels of abstraction [Scheider96]. Their interest in learning how to use software also matches with their capabilities [Hanna97][KIM2008]. Older children seem to be less patient with new software and less willing to explore the software on their own [Kharrazi05]. Furthermore, children in this age range are elementary school-aged children. Despite being *exposed with computer technology* today, either in school or at home, the children have creative and ambitious goals in making programs that are similar to the applications they use, especially in games and simulations which are well-known for their graphically rich and highly interactive activity [Becker05a]. In other words, they often have a *high interest* in learning how to program [Mano04][Robertson04][Peppler05]. It is also important to know how children use and work with the software tools, including the degree to which the learning matches the children's capabilities and their ability to involve in evaluating the tools [Fernaeus03]. Their experiences in school allow them to get ready to fulfill designated tasks, to follow directions, and above all, they are not self-conscious about being observed as they use the software or are involved in the evaluation process [Hanna97][Pan09]. The final reason is to align the age range with the *age classification standard* set by the games industry [Bijvank09][Unterhaltungs09]. The age rating is primarily concerned with the game rating system, game content descriptors and as guidance for parents when buying games. Therefore, in principle, older children are not being covered in this research.

2.2 Usability Research of Software for Children

When choosing software, those reflecting children's understanding and inviting them to contribute with their own experiences, are most favorable [Spencer86] including highly structured software (such as games, drill-and-practice and tutorials) and less structured software (such as simulation type of software). Well-designed graphics, colors and reinforcement features which are intrinsically related to the program's content also contribute to the quality of a program including exciting sounds for the children to have fun using and experimenting them [Röber08]. However, the issues of reliable usability guidelines for children's software are still vague. Nevertheless, current works are dealing with this matter for pursuing better quality of children's software. The effectiveness and good quality of software can be recognized in many ways. Children are fond of software that is easy to use, easy to understand, and fun to work with in various ways, and built up their ideas of software with the help of educators [Nakagawa99]. It is necessary to find software that enables children to shape their ideas more easily. In addition, there are three important aspects in choosing children's software: (1) the children will enjoy using it, (2) the children are able to quickly transfer knowledge from one playset to the next, and (3) age

appropriateness [Egloff04]. Another three key elements to consider when reviewing children's software are gender preferences, level of development (ability to use and understand the information presented) and ease of use [Buckleitner99][Hayes05]. Additional characteristics of educational multimedia systems that are important for this matter include (1) learning environments that have different goals and approaches depending on the workplace environments [Inkpen97], (2) what works for adults will not necessarily work for children, and (3) girls and boys interact differently with technology [Inkpen95][Ray04].

Why do the usability guidelines need to be concerned with designing software for children? A high-quality software for children can often be recognized by the presence of certain characteristics or program features such as comprehension of the world around them, program content appropriate and interesting for children, high degree of interaction from children, clear directions, advertised as a program that children can run themselves, and a program designed to be used repeatedly [Pane98][Yusof02] [Markopoulus08]. Ironically, these software evaluation systems were established by looking at the software usage by children, and claim to come out with the best looking interface or children-computer (or software) interaction [Ibrahim04]. There are two major limitations to utilizing any software evaluation system for software selection [Haugland97]. Firstly, it is virtually impossible for any system to evaluate every software program on the market, given the current pace with which software is produced. It is likely that some programs which would be effective teaching resources have not yet been evaluated [Ericson05]. Secondly, there is always a time lag between the evaluation of a system and the date when these evaluations become accessible to public. *Why do the usability guidelines need to be considered when evaluating children's software*? There are several reasons for asking children about their opinion on interactive products [Read06]. One is that adults and children live in different worlds and adults may not understand what children want. Secondly, there is a move towards including children in decisions about their own environments. This arose from a greater awareness that children are actors and participants rather than users in the society. Software for children can be looked at from two perspectives: the quality of children's software and the involvement of children themselves in evaluating the quality of the software.

Several individuals, educators, researchers, parents and organizations devised evaluation systems that they are using to evaluate software. Table 2.1 summarizes five software evaluation systems used to form a basis for the software selection for children. Each system has a different philosophical approach and specific priorities to be considered when evaluating software. From the table, most of the evaluation systems were established by the respective research institutions and software companies involved in developing children's software or committed in the future of learning technologies for children. Most guidelines serve as universal criteria that can be applied to any software selection strategies. Fortunately, these guidelines can also be found in the field of interaction design and children. Referring to the table, the pedagogical aspect of the software is treated as the most important value of the listed evaluation systems. The ability of the software to educate the children and the capability of the children themselves to learn the software are crucial elements for developing children's software. In fact, the balance between the educational value and the entertainment value needs to be integrated into the children's software [Fromme03]. Overall, a basic understanding of the design of children's products is problematic for most children. This is due to a poor plan and implementation without thinking about the aspects of its usability.

Table 2.1 shows selected software evaluation systems used by companies or research institutes for selecting children's software.

Children's Technology Review [CTR09]	**Instructional Software Evaluation Rubric** [Prince04]	**Haugland / Shade Developmental Scale** [Haugland97]	**Educational Software Evaluation Tool** [Lamb06]	**Eye Square GmbH** [Duda98]
• Ease of use • Child proof • Ability to educate • Ability to entertain • Design features • Value	• Curriculum content • Teaching and learning approach • Critical thinking • Perspective • Pedagogy • Effectiveness • Content customizing features • Assessment components • Support materials • Program accessibility features	• Age appropriateness • Child control • Clear instructions • Expanding complexity • Independent • Process orientation • Real-world model • Technical features • Trial and error • Transformations • Anti-bias deduction	• Overview • Content aspects • Technical aspects • Support aspects • Information presentation • Instructional aspects • Motivation aspects • Productivity aspects	• Cover and booklet • Entertainment value • Suitability for children • Ease of use • Load • Educational value
© 1998 Children's Technology Review	© 2004 Prince George's County Public Schools	© 1996 Revised by Susan W. Haugland	© 2006 Annette Lamb & Larry Johnson	© 2008 Eye Square GmbH

One question that needs to be answered is how to build new software technologies that offer children control and the opportunity to challenge themselves. These technologies must meet the terms referring to the usability aspects as discussed before. Children need time to develop their knowledge and skills for a variety of technology applications before they can use them independently [Druin99b]. They should have better opportunities to explore open-ended, developmentally appropriate software programs in a playful, supportive environment. This will help children to become more confident in using the computer and it will provide the foundational skills needed to use more advanced applications as they grow older. However, this thesis will look at the different perspective of software engineering in general, and the usability study in detail. This includes concentrating on the aspect of usability research in designing and developing a game authoring tool software for children with careful attention to optimizing the advantages of computers and the learning of children [Strohecker00][Goolnik06].

Why was the game field chosen for the path of this research? First, it has to do with the children's *experience in games* (playing games or designing games). Game designers face the challenge of creating games that can be easily learned, effectively played, and emotionally enjoyed by gamers [Ye04][Denner05][Latif07]. Second, *no guidelines* or principles have been established to help game designers to do their work [Yatim09a]. The vast majority of literature on heuristic evaluation is concerned with evaluations of usability. However, guidelines exist for other aspects of the user experience of interactive products, such as fun, instructional design, and game playability [Skalski06]. A study specified three aspects of usability, rather than player enjoyment of games. The three aspects are interface (controls and display), mechanics (interacting with the game world) and game play (problems and challenges). There are also game design researchers who are using HCI methods to enable a game design and evaluation that could be a benefit for the game industry. A method called *Heuristic Evaluation for Playability* (HEP) was developed as a comprehensive set of heuristics for playability based on current literature and was reviewed by several playability experts and game designers [Desurvire04]. Attempts to generate heuristics and usability guidelines for the creation and evaluation of fun in games have also been done [Federoff02]. However, prominent guidelines for designing a game authoring tool are still not sufficient.

When talking about games (the term includes video games, computer games, hand-held electronic games and games played on game consoles), it is obvious that playing games is a popular recreational activity for children [Klawe95][Prensky06][Freisleben08]. Some believe that children will gain benefits by playing games because games are designed to motivate children to successfully learn and embrace active critical learning principles [Klopfer05][Shaffer06]. Leading scholars have long argued that games provide a convincing context for children's learning such as communication, collaboration, design, creativity and technology, ICT, and thinking [Papert93][Crawford03][Jenkins06][Gee05]. Second, games can improve important 21st century skills such as problem solving, decision making, collaboration and communication, visual memory and perception, eye-motor coordination and ICT literacy [Inkpen99b][Diezmann00][Prensky01][Gee03][Fernaeus06]. Third, designing games is an interdisciplinary learning process. Besides designing games, children can also adopt other aspects of learning such as arts, languages or mathematics [Sedighian96c][Hussain06]. But there are also potentially negative effects of games including reduced physical fitness, risks for social behavior, undesirable effects of violent games, and lower academic performance [Anderson01][Gentile04][Anderson08]. But a study shows that using games as educational tools is more positive than negative [Rosas03]. Nevertheless, the

future of the game industry became well-accepted as a part of the computer science field. This research goes beyond and looks at the potential benefits gained by children actively creating and designing games instead of just playing them. The next section will touch the aspects of fun and flow that are important for the game industry.

2.3 Fun and Flow

Satisfaction in the usability definition refers to user satisfaction regarding the efficiency and effectiveness of the software. Additionally, fun should also play a role in the evaluation of computer games for children because enjoyment is one of the major motivations for children to interact with technology. The ease of use of a game's control and interface is closely related to fun ratings for the game. But the usability concept should be extended to include the element of fun, since fun is not the same as satisfaction [Carroll04]. Others state that satisfaction involves progress towards goals and that fun is not goal-oriented [MacFarlane05b]. More and more people consider the importance of fun as one of the critical success factors in determining the usability of application software for children. Since the objective of software for children is to provide an engaging learning environment, it urges to keep their attention by providing fun and exciting environments. The earliest study on fun goes back to the work of Malone [Malone80] on educational games. He summarized the design heuristics for an enjoyable interface with three intrinsic motivator elements. These elements are *challenge* (the level of difficulty), *fantasy* (the scenario in which the activity is embedded) and *curiosity* (the introduction of new information and non-deterministic outcomes). Later he added another element, *control,* which makes the players feel in-charge [Malone87]. These motivator elements focus on motivational support and have become key aspects of the design of entertainment software products, especially in game design that fosters engagement. Children expect to have fun when using technology and link the idea of fun to challenges, social interaction, and control over their world [Inkpen97]. In fact, learning is not to be facilitated by devices, but by engaging with the devices [Larssen04]. Here, the term *hard fun* was introduced to show that children are engaged in software environments, which is a hard job in a fun way [Papert80][Sedighian96b][Rose04].

As mentioned above, fun is associated with playing for pleasure, and activities should be carried out for their own sake through freedom of choice [Shneiderman04]. In other words, *fun is not goal-related*. It should be noted that fun is not the same as satisfaction in the definition of usability above. Satisfaction involves the progress towards goals, while fun does not [Wiberg03][Yatim08]. But the concept of usability for any entertainment software including games should be extended to include fun. However, in most cases, the heuristics of the different sets of software seem to contradict each other. For example, while the purpose of usability heuristics is to remove challenge, fun heuristics say that a good game needs challenge [Malone83]. Problems, both related to fun and usability, are rated with a higher severity level than problems that are only caused by a lack of usability. This gives an idea of the importance of evaluating fun in usability tests. Focussing both on usability and fun during the usability test provides extra information about the quality of the game [Malone81][Robertson94]. Therefore, traditional usability guidelines need to be revamped for educational and entertainment software, especially in game play and game design.

Researchers found that the heuristics in the games literature, especially in measuring fun, overlap closely with the elements of flow [Malone87] [Sweetser05]. Flow happens when people are creatively facing challenges to achieve personal goals. Most games have a difficulty

option so that, if the game is too easy or too hard, the users can select a different level of difficulty that better suits their skills [Parker92][Pausch94]. Educational games can use this to try to maintain a level of difficulty that matches the players' competency level. This ramping of difficulty can lead to what Csíkszentmihályi has called a *flow* state [Csikszentmihalyi94]. A flow state is achieved when the users' abilities match the challenge of what they are doing and they get an immediate feedback on their performance. The theory of flow is described as the feeling of optimal experience which has been widely referenced across a variety of fields. According to Csíkszentmihályi, *"instead of being buffeted by anonymous forces, we do feel in control of our own fate ... we feel a sense of exhilaration, a deep sense of enjoyment*" [Csikszentmihalyi96]. In brief, Figure 2.2 shows the players' skills on one axis and the degree of challenge on the other axis. The flow channel occurs around the space where the players' skills and the degree of challenge are in step (A1, A4). If the players' skills are better than the degree of challenge, they will become bored (A2). If the challenges utterly exceed their abilities, the players will suffer anxiety or frustration, and at the end, they will most probably stop playing (A3). For this case, flow derives from the activities that provide *enjoyment* and the players will then *being engaged* with the activities carried out by them.

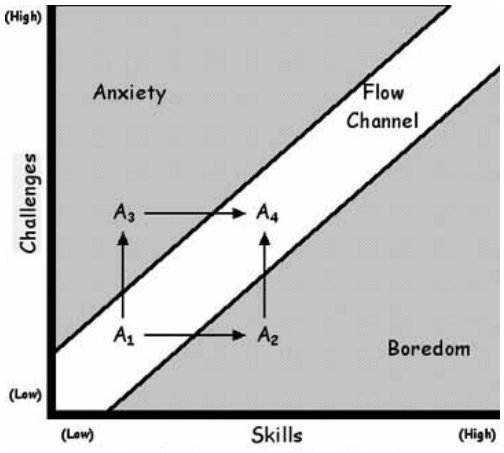


Figure 2.2 The original diagram of flow and the psychology of optimal experience [Csikszentmihalyi94].

Through motivation, flow, learning environment and game design, there are clear connections that show how learning and gaming are fundamentally built from the same base [Paras05][Sweetser05]. The challenge for educational and software designers is to build environments where the dynamics of learning are fully integrated into the dynamics of game play or game design activities [Blythe03][Fisch05]. From the software perspective, the capability of the software is crucial to keep the players or users in their flow channel. Hypothetically, an Eyetoy or similar devices can be used to monitor the player's facial expressions and detect frustration or boredom, especially by monitoring the flow or even measuring fun. Unfortunately, this research does not consider the psychophysiological field which is concerned with the various techniques in performing observations on psychological and physiological factors such as eye blinks, gaze tracking or brain activities [Horie03]. This could be a way to determine the flow and fun aspect of using software such as in the work of Stellmach [Stellmach08], but this research only used common research instruments and measurements such as questionnaires, interviews and self-reports.

2.3.1 Relationship of Usability Guidelines and Fun

Until today, there are no specific usability guidelines to measure the effectiveness, efficiency and satisfaction or fun in a game authoring tool or similar. For this research, the ISO definition of usability is used and the fun element is treated as another independent measurement. But it should be noted that there are several approaches to measure the usability within the frame of a user study. The first way is to observe what happens. Secondly, it is to note any evidence that occurs during the interaction, and thirdly the users should be enabled to assess the usability during the interaction. Figure 2.3 shows three general heuristics used in usability guidelines for designing software for different user experiences regardless the age. The three general usability guidelines will be used throughout this study. In addition to the field of educational gaming which comprises the aspect of fun, the usability heuristics effectiveness, efficiency and satisfaction will also be covered in this research.

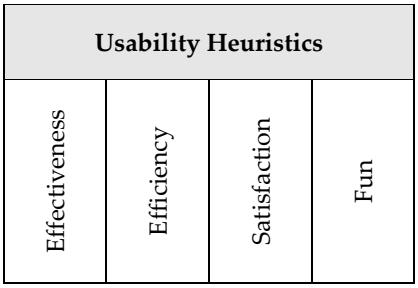


Figure 2.3 The three general usability heuristics including fun, specifically designed for this research.

In addition to the principles to follow, the process of designing the tool strives to achieve certain measurable goals. These usability goals help to evaluate the design and to assess the usability. The specific goals of interface design are: learning time, speed of performance, rate of user errors, retention over time, and subjective satisfaction. Each of these goals is measurable. Learning time, speed of performance, retention over time, and rate of child errors are quantifiable goals which can be measured by user tests. Subjective satisfaction including the fun factor is still vague, since it depends entirely upon the opinions of the children. As mentioned in Chapter 1, several studies will be conducted to seek and generate standard usability problems in the game authoring tool that is designed as research prototype. These studies will use existing usability goals as measurement and benchmark to determine the usability guidelines that will be designed later. The measurements are concurrent with the usability heuristics used in this research, as depicted in Figure 2.3.

Learning time = time taken to perform a certain task.

To measure *learning time*, two measurements will be pursued: the decision to determine when the software has been learned, and the ability to perform a certain task. One common approach is to find children who never used the software before and measure the time they need to learn it. The data on the learning time will provide the ease-of-learning measures for the software – *Usability Heuristic (Efficiency)*.

Speed of performance = time taken to perform a certain task in a specified period of time.

To measure *speed of performance*, a specific benchmark task or tasks will be designed for the children to perform those tasks using the software. Again, the time will be used as measurement unit to see how long each child takes to complete the given tasks. The tasks are consistent across all users and the children will be trained on the software before they actually have to perform the given tasks. The findings from measuring the speed of performance will show the ease-of-use factor of the software – *Usability Heuristic (Efficiency)*.

Retention over time = time taken to perform a certain task by the same group of children in different periods of time.

Retention over time requires the same group of children to participate in the tests more than once. For this purpose, the children have to perform the benchmark tasks and are invited again after a certain time to find out how much they remember when working through the same tasks again. The times of the first use will be compared with the times of the second use to provide data on the retention-over-time measures – *Usability Heuristic (Efficiency)*.

Rate of errors = number of errors made by children when working through the benchmark tasks.

To measure the *rate of errors*, count the number of errors made by children when working through the benchmark tasks. Clearly specify which errors to be counted prior to running the tests. The real problems are problems identified during the software testing with children. In other words, the relation between the number of problems uncovered with testing (predicted by evaluators) and the total standard problems (problems identified by testing with children) can be identified – *Usability Heuristic (Effectiveness)*.

Satisfaction = pleasure or contentment derived from gratification.

Satisfaction can be measured by asking the children to rate the software. Questionnaires can be used to get opinions and comments from them after using the software. The children might respond slightly more positive if they know that the person administering the survey has a vested interest in the outcome. Therefore, a questionnaire on interface design and user satisfaction will be used, which was adapted from the original QUIS (Questionnaire of User Interface and Satisfaction) [QUIS09] – *Usability Heuristic (Satisfaction)*.

Fun = pleasant and enjoyable feeling gained through an amusement activity.

Fun is best measured by observation and by asking the children to rate the software, since only they know whether they like it or not. Observation is not only to be looking, listening or writing down events occurred, but it also demands an adequate method. Different from other quantitative research which deals with numbers as one of the methods in qualitative research, the observation seeks the *why* and not the *how* of the topic. Nevertheless, by conducting a fun survey which relies on self-reported responses, the children can give sincere and honest answers [Read05c]. Literature reveals that self-report scales can be made more child-friendly, for example through the use of smiley emoticons (emotion icons) such as the Smileyometer and the Fun-Sorter [Read06], see Chapter 6 for further explanations – *Usability Heuristic (Fun)*.

This study also holds that a study of flow theory can enable a better understanding of what goes on within both patterns of interaction in software, and within certain types of learning environments. The next section describes the types of learning environments for children.

2.4 Educational Programming Environments (EPEs)

Programming languages are just like natural human languages, but the languages used are either slightly or totally different. Programming languages are also favored in certain aspects, such as the graphical user interface (GUI), the metaphor used, the ways of programming and, most important the ways of thinking [Myers92][Gibson03]. The programming language for children is another story to talk about. It has been discussed since 30 to 40 years and still remains the favorite topic for those who are interested in the field of children and computer [Noss96][Guzdial04][Kelleher05][Gómez-Albarrán05]. Unfortunately, regarding the programming languages and environments available, a deep understanding of these applications is needed before creating these programs. Further in this section, this genre of children software will be described contributing to the field of children and software. The current focus is on motivating and engaging children in learning and on providing an easier and more natural way of programming.

To better understand the issues of children and their way of programming, the term educational programming environments (EPEs) will be introduced and discussed next. The term refers to "*any programming environments that are designed as learning instruments instead of writing the real application*" [Yatim09c]. Other notations used to refer to the same philosophy are educational programming language [Yoo06], mini-languages [Brusilovsky97] and microworld [Papert80]. But here, the word *environment* differentiates the philosophy behind the programming tool itself. The programming *language* is a machine-readable language designed to create programs such as C++ and Python, whereas the programming *environment* can be an integrated development environment or an IDE (a graphical user interface program that integrates programming aspects such as debugger, visual interface builder and project management), or a command-line environment (a collection of commands that can be typed in to edit files, compile source codes and run programs). Since the research is more focused on the surface-level, user interface constructs, learning goals and the pedagogical reasoning for some design choices, the word *environment* will be used.

Researchers have different approaches in classifying the programming environment tools. The classification of EPEs in this research is stepping beyond the existing surveys which are more focused on the surface-level [Kelleher05] and user interface constructs [Gómez-Albarrán05] and only deal with a small number of languages [Guzdial04]. The main reason is

to consider learning goals and pedagogical reasoning issues as well as user interface constructs for some design choices in these tools [Wook07][Pouyioutas08]. The strength of this survey is the wide range of included tools and the emphasis on education topics. It is somewhat more up to date than other surveys and the classification is done in a unique way considering a large number of programming languages (about 30 tools) and describing the progression of this research area since the last 30 years.

The classifying survey of EPEs has been described in detail in this paper [Yatim09c]. These tools have been categorized into five groups using two steps. The first step was to review the tools according to the *timeline* (the year the tool was designed and introduced) and the second step was to get hands-on experiences with the tool's *features and capabilities*. The survey was carried out with more than 30 solutions described in the literature with different approaches and different aims. In addition, nearly half of the EPE tools described in this paper were used to get a deeper understanding of their features and capabilities. At the end, the survey revealed five distinctive groups of EPEs which are *textual programming languages, graphical programming, programming by demonstration, tangible programming* as well as *game development and creation tools*.

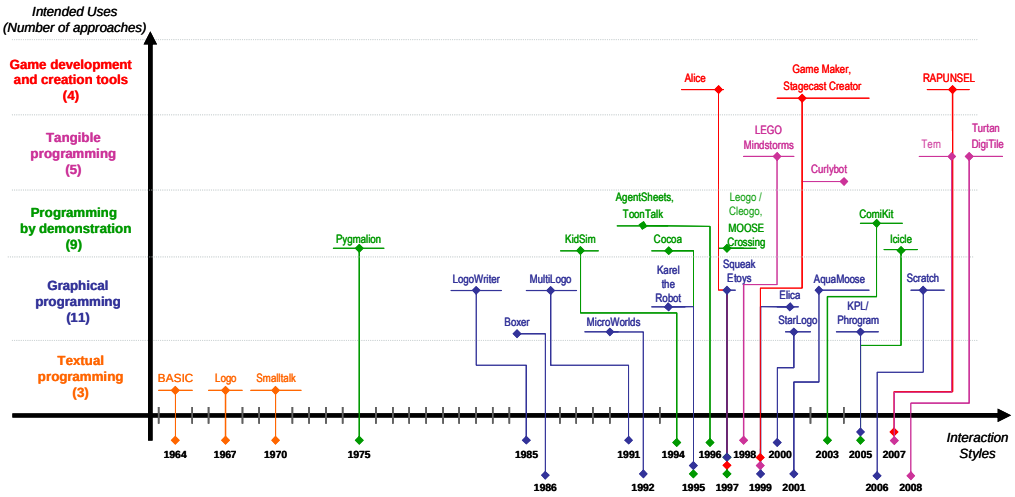


Figure 2.4 Selected EPEs used in the survey arranged according to time, intended usage and interaction styles [Yatim09c].

Figure 2.4 shows the taxonomy of the selected EPEs according to the two steps used in the survey, the timeline and the features and capabilities of the tools. The first observation was the increasing interest in designing a programming language for educational purposes in the 60s and 70s by introducing BASIC, Logo and Smalltalk known as the basis of *textual programming languages*. It is interesting to observe that there was a considerable gap in the EPE development between 1975 and 1985. Those were the years where developers strengthened the software they built and implemented it in various fields and applications. For example, BASIC was introduced to schools in the early 80s [Papert80], as described in the next section. This gap in the timeline may also be attributed to the fact that only the outstanding tools were covered in this survey. The second observation took place in the 80s where the next large amount of approaches can be attributed to the *graphical programming* group as a natural successor. The simplified-syntax programming group mainly adapted the design of visual programming systems. Apart from the drag and drop activity of script

blocks, this approach reduces and eliminates problems and complexity of the programming activity [Myers86][Gaelli06]. For example, Squeak Etoys and Scratch came from the same root and both rely on an approach based on building visual script blocks [Malone04]. Both research teams are convinced that this approach can make programming easier for children. In fact, learning visual programming can lead to a better textual programming later [Rader97][Dann05][Mattila06].

The third observation was that a new way of programming for end-users emerged. It introduced *programming by demonstration* with Pygmalion. Pygmalion became the first iconic programming language and the basic idea only became widely accepted when more software was introduced following the same trend in the 90s. The forth observation was the rise of new applications adopting several approaches in educational programming at the same time. As the years went by, more advanced software was designed and each of it brought either a new concept of programming (such as *tangible programming*) or merged ideas from previous programming paradigms, such as *game development and creation tools*. In the late 90s, researchers began to create game development tools to capture the attention and interest of children in designing games [Strohecker99][Slaughter01][Valente05]. At the same time, children were expected to practice and learn how to program through their game projects. Finally, other observations showed the acceptance of *tangible programming* as a new paradigm for educational programming environments which combines a set of tangible tools with user interface metaphors to make programming more fun and exciting for children. It is expected that the tangible programming group will grow and become more important in the future with the increasing availability of appropriate hardware [Raffle06][Gottschlag07]. Another area is the game design group. More *game development and creation tools* are developed to assist the game creation process with programming as core activity. It is also believed that by applying programming to the game design activity, this will keep the children actively engaged in mastering the programming techniques and gaining computer literacy.

But the time dimension alone is not sufficient. For the second step, these tools were studied according to their features and capabilities in order to finally classify them into five categories of different programming styles. Therefore, the main attributes of the tools were listed that were considered to be crucial in any educational software, as shown in Table 2.2. This includes features such as typing programming scripts, the source code editor, drag and drop interaction, visualization-based displays, action-based recording, 3D object manipulation, hardware with haptic capability, object and media manipulation, granularity learning, explorative learning, integrating tutorial or learning agents, social collaboration, ability in creating games, free open-source software and multi-platform features. These attributes were chosen because they involve both technical and educational aspects. For example, the importance of open-source and multi-platform capabilities of the EPEs is suggested in order to be able to use them in the classroom or to reduce the total costs of ownership. Furthermore, these attributes help to classify the EPEs. As mentioned before, all EPEs were studied with regard to their features and capabilities in order to finally classify them into five groups.

Table 2.2 shows the relationship between selected attributes within the selected EPEs.

EPE Software \ Attributes		Involves typing text, scripting	Source code editor	Drag and drop interaction	Visualization-based display	Action-based recording	3D object manipulation	Involves hardware with haptics	Object and media manipulation	Granularity learning	Explorative learning	Tutorial or learning agent	Social collaboration	Ability in creating games	Free, open source	Multi-platform
Textual Programming	BASIC	×												×		
	Logo	×			×	×			×				×	×		
	Smalltalk	×												×		
Graphical Programming	LogoWriter	×			×											×
	Boxer			×						×			×	×	×	
	Karel the Robot	×	×		×				×		×	×				×
	Squeak Etoys		×	×	×				×	×	×		×		×	×
	Elica	×		×	×		×		×		×				×	×
	StarLogo			×			×		×		×	×	×		×	×
	AquaMOOSE				×		×		×		×			×	×	
	KPL-Phrogram		×				×		×	×				×		
	Scratch			×				×	×		×	×	×	×	×	×
Programming by Demonstration	Pygmalion		×		×	×										
	AgentSheets		×	×	×	×			×	×	×		×	×		×
	ToonTalk			×	×	×	×				×	×	×	×		×
	Leogo/Cleogo	×			×	×							×			
	MOOSE Crossing	×											×	×	×	×
	ComiKit				×	×			×		×			×	×	
	Icicle				×	×			×		×			×		
Tangible Programming	LEGO Mindstorms	×	×		×			×		×	×		×			
	Curlybot					×		×			×		×			
	Tern					×		×			×	×	×	×		
	DigiTile			×	×			×	×		×		×		×	×
	Turtan			×	×			×	×		×		×			
Game Development & Creation Tools	Alice	×		×	×		×		×	×	×		×	×	×	×
	Stagecast Creator			×		×			×		×		×	×		×
	Game Maker	×	×	×	×				×	×	×		×	×	×	
	RAPUNSEL	×			×		×			×	×		×	×		×

* Unmarked area indicates that the software might either not have covered the listed attributes or it was not mentioned in any papers listed in the references. It might also have had these attributes during or after conducting the survey [Yatim09c].

The features and capabilities were considered according to two aspects in the understanding of programming concepts and structures that are appropriate for children. The first aspect is the features that focus on tasks compared to concepts. It is believed that the results of learning can be better seen in the design process. The second aspect is the style of thinking in the tools, since different tools assume different mental models of their users. Scaffolding in EPEs is important, because the tool should be able to demonstrate and explain the process and encourage children in what they are doing. The next section will concentrate on one particular programming style, the *game development and creation tools*, which are closely related this research.

2.4.1 Simulation Tools and Authoring Tools

With multiple programming approaches in varying tools, researchers still attempt to ensure that ordinary people can program. So, what is the most suitable definition referring to these kinds of tools; is it programming or authoring? It is strongly believed that authoring tools reduce the amount of programming expertise required in order to be productive [Abdullah08]. Furthermore, these tools use visual symbols and icons in flowcharts to make programming easier [Yokokawa06]. *Programming* is defined through the use of syntactic and semantic rules to determine structure and meaning, respectively. Some authors restrict the term programming language to those languages that can express all possible algorithms [Knuth84][Seth96]. On the other hand, *authoring* usually enables the user to create a final application merely by linking together objects and sequencing them in an appropriate order [Good04]. The word authoring always coined and used in the development and design of hypertexts or multimedia applications. Most authoring systems also support a scripting language for more sophisticated applications, such as Lingo in Macromedia Director. However, the distinction between programming tools and authoring tools is not clear enough [Yatim07b]. Therefore, by definition, *authoring* is the easier process of arranging and structuring content in an interactive environment, while *programming* is the more advanced technique of writing the source code of a computer program. Typically, authoring tools require less technical knowledge to master and are used exclusively for applications that present a mixture of textual, graphical, and audio data [Huber07]. The idea of programming tools for children should be simple enough to help the children think and solve given problems in a systematic way [Kahn01][Kindborg03][Al-Imamy06].

Another issue is the confusion between simulation tools and authoring tools. A *simulation tool* is defined as any application software that informs children about its content by using images as well as visual and animated features. These software tools force children to think about the cause and effect upon performing certain tasks or behaviors that are normally implemented in either drill-and-practice applications or multimedia courseware tools. *Authoring tools* seem a bit different and can be referred to application software that allows children to construct or design other applications which follow the principles of the constructivist theory [Papert80]. These tools will permit an active self-guided learning and can also enable the social interaction among children. An ideal authoring tool would scale in programming granularity in order to raise and nurture children's computer literacy along with their capability to program [Yatim07b]. Furthermore, constructivists believe that children can be taught to use their iconic and symbolic mentalities when solving problems [Kafai96b][Yamaguchi08]. These properties can be valuable in nurturing creative thinking. At one tool Web site [Stagecast09] it claimed that the Stagecast Creator is suitable for children because it involves less programming activities. It is understandable that children who are

no trained programmers will have some difficulties in understanding the system. But the ability to manipulate programs and their interactivity is considered essential in software for children [Crawford84].

To understand how children learn to use authoring tools, their understanding of programming should be explored first. Programming skills can be seen as the target domain of children's learning. Four contexts for analysing children's programming activities have been discussed; those are the programming context, game play context, game design context and school assignment context [Tholander02]. These contexts are concerned with the children's understanding of rules that they constructed by programming and designing their own games. More interestingly, what kind of programming style is most suitable for children, and why? A number of EPEs provide programming styles that are apparently far more attractive to children by using visual representation compared to other conventional programming [Myers90][Sword00]. Visual languages are also said to be one of the benefits to make programming easier and more accessible to children, although not all aspects lead to visual representations [Ioannidou03][Becker05b]. It is believed that the styles of programming used by children are depending on the children themselves such as their experience, age and learning motivation [Wright05]. The important factor is that the tools provide a gradually transition from a simple visual programming representation to a more high-level abstract programming language [Smith94][Yatim07b]. For the purpose of this research, in order to gather information that could assist in designing a game authoring tool for children, an attempt was made to find out what children are able to understand about game programming and will further guide the development of a game authoring tool.

2.4.2 Why a Game Authoring Tool?

Serious games movement is trying to move away from the old traditional entertainment games. The game designers and developers are trying their best to cope with the game market industry [IGDA08]. For example, in most cases, the latest and greatest games hardware and consoles are probably not available in certain environments such as in schools [Pelletier05b]. Most schools prefer to use what they already have rather than buying and investing in new equipment. So, serious game developers need to go beyond any typical assumptions about gamers and hardware accessibility. Furthermore, the interest in games and learning has grown significantly [Gee03][Shaffer04]. Introducing game making as a way of learning has brought a great deal of thoughts in spending time and effort in designing educational games on content matters, graphical representations and instructional design. The greatest learning benefit remains reserved for those who are engaged in the design process: the children as game designers. Hence, this research has the opportunity to engage children in game-related activities, whether it is about playing games or making games for learning.

Firstly, the idea of learning by doing is parallel to the theory of constructivism. The serious games initiative shifts its attention to the same area. Despite playing games, the implementation of appropriate tools for designing games can also embrace the movement of the serious games initiative. In fact, this is the main reason why this type of tool is chosen to be developed during this research. The earliest computer-based learning (CBL) application was known as computer-assisted instruction (CAI). The transition from just having a pre-programmable application software (such as CAI) to a construction type is a must in the current situation [Haugland97]. Secondly, the tendency of today's children to spend most of

their time with games and computers has made this research believable [Prensky01]. For instance, children started to spend their ample time in cybercafés just to play games with their friends, even though they had to spend a lot of money there. Since the children grow up in the computer era, a game authoring tool can allow them to be creators and designers in many ways [Steiner06], and finally they can share their creations with their friends and families [Kafai06]. Thirdly, besides just playing games, the development of a game itself can bring beneficial advantages [Begel97]. It is a new research field and people still argue on the beneficial side of it in terms of educational value. Most people believe that this type of activity can bring entertainment value, but there is also a need to look at the potential of this type of software to be used as a teaching and learning tool in the classroom [Papert96][Kirriemuir03]. Seeking information on the minimum requirement for a potential usage in the classroom is a must. Furthermore, the support towards the research has to be considered. For example, the introduction of ICT and related changes in the school curriculum shows a greater focus on the children's activities and responsibilities [Ahmad98][MOE09]. At the same time, the role of teachers is expected to change [Egenfeldt-Nielsen04][Annetta08]. The shifting of the focus in education from content to applying certain software can make it even more difficult for teachers to control the learning process [Robertson05]. This has created a situation where *to do* something with the computer seems to be more important than to understand the content of different subjects.

In order to gather information that could assist this research, previous efforts in the same direction and field are taken into consideration. A reknown work in the field of children and programming is from Papert who he introduced Logo to children aged seven to ten [Papert93]. The principal idea is to introduce programming by drawing using turtle graphics within an explorative and constructive environment where the target audience are children and teachers. One of his students has extended Papert's interest in believing that today's children are empowered to create their own software applications [Harel90]. Both held that by consuming, choosing and creating software such as in this instructional design project, children can control their own experiences in designing and creating their own applications, which creates an entirely new way of thinking and decision-making. They created a learning environment where children learned factions with Logo programming and this was the beginning of what they considered as learning through design activities. The next generation in this field is the work by Kafai who believes that children can rather be game producers than game consumers [Kafai95Kafai98b][Kafai98]. She focused her efforts in a different direction: the designing of games. Despite embedding the learning lessons directly in the games, her goal was to provide children with greater opportunities to construct their own games, which has far more potential to engage the children's enthusiasm in playing and designing games. Her work concentrated on a class of 10-year-old children who created their own games using Logo programming. Afterwards, she used the games to teach fractions of them to a group of younger children in the same school.

The game design activity using a game authoring tool is valuable for a number of reasons. First, children can develop transferable software skills together with their friends and it creates an active learning environment [Rieber01][Hoyles04]. Here, children become active designers and gamers in planning, making decisions and solving problems. In fact, they can also teach each other what they have learned and participate in collaborative learning [Cockburn98][Williamson05]. Second, designing a game is a mentally demanding task which results in cognitive and affective (attitude) gains for children to learn through design [Pelletier05][Robertson05]. Children learn aspects about game programming, thinking about

game interface design, designing graphical elements, conceiving stories and creating instructional representations. Third, making games can support other types of learning such as languages, art, and mathematics lessons (including fractions, geometry or tangrams) and sciences [Harel90][Sedighian96][Sim04][Kafai05]. Children will be put into situations where they can construct, revise and facilitate an intrinsic integration of the game context and the content of the lessons.

2.5 Observation of Existing Game Authoring Tools

The following section covers three selected and well-known game authoring tools designed for children of all ages, including the programming concepts, features and capabilities of each tool. The main aims of the observation are (1) to choose an appropriate computational metaphor which supports recognition rather than recall [Pane02], (2) to provide a high-level instruction and a collection of resources as part of the programming environment [Sheehan03], and (3) to seek possible ways to minimize the programming by means of appropriate computational media [diSessa87]. In programming concepts, a brief idea about the concept of programming shown by each tool is described. The idea is to identify each tool speciality in teaching programming concepts, algorithm thinking and implementation in programming education for children. The visual modeling capabilities of each tool are exploited in order to relate them to how children learn to program using these tools. The main distinguishing features of each tool are described based on the relationship between its features and cognitive burdens among children, as stated in Table 2.3. The purpose of selecting three game authoring tools as examples is that these tools represent different distinguishing features in designing games for children aged seven to twelve. On top of that, design and implementation of the tools are highly motivational and extremely accepted by non-professional programmers. Certainly, the choice of the game authoring tools is important to scaffold the design process of a game authoring tool for this research.

2.5.1 Alice [Alice09]

Alice is an introductory object-oriented language, which is suited to training purposes. The environment was developed by a research group of the Carnegie Mellon University in the USA. In this environment, beginners can start with the development of interesting and very simple 3D environments and they are allowed to explore the new medium of interactive 3D graphics. Alice is primarily a scripting and prototyping environment [Pierce03]. The user has the option of using simple drag-and-drop interface animations from 3D models that can be created, as shown in Figure 2.5. Alice focuses on telling stories in animated videos or games [Conway99].

User Study with the Involvement of Children: Alice has also been involved in certain evaluation tests including undergraduate students who mainly attended programming courses. In one of the studies, researchers revealed that high risk students, who were both weak in mathematics and computer science, showed high retention rates and a great increase in attitude after participating in the Alice course [Moskal00].

Table 2.3 shows the main four aspects of each tool which consists of programming style, features and capabilities.

Tool / Aspects	**Alice** [Alice09]	**Stagecast Creator** [Stagecast09]	**Game Maker** [GameMaker09]
Programming Style:	Introduces an advanced 3D programming with numerous conditionals, count loops, while loops, variables, parameters and additional procedures of objects. Each object has its own properties, methods and functions. The main idea of Alice in expressing programs is to simplify the entering code and finding alternatives to typing programs. Alice introduces pre-defined command blocks or instruction blocks within its drag- and-drop interface.	Uses a rule-based system caused by matching rules with the current state of the program and executing the instructions of the matched rules. The rules are known as *before and after* (similar to *if-and-then* rules) and are graphically defined. The world consists of one or more stages, which is a 2D coordinate system, as shown in Figure 2.6. The coordinate system represents objects involved in the program based on the movement from one coordinate to another. The basic object is called agent, defined by the programmers in the grid-based 2D world. Each agent occupies one cell in the grid and when the agents move, they move one cell per step.	Game Maker has its own scripting language called GML which extends its functionality, especially for advanced graphics features that are hidden from less experienced users. Thus, GML provides a level of scalability for different users of Game Maker.This feature is not child-oriented, but advanced users can have full control of the game-making interface, and the scripts in GML can be assigned to game elements, to finely define behaviors and events.
Features:	With five regions of interface (screen windows, object tree, object details area, methods editor area and events area, as shown in Figure 2.5, the programming environment of Alice is perfectly designed to teach programming through modeling 3D virtual worlds.	Since the Stagecast Creator uses graphical *before and after* rules, the representation has strong symbolic signs. The user interface that is based on windows, menus and icons has a similar interaction style as graphical user interfaces in windowing systems. The main emphasis of the tool is on the simplification of expressing the program by abolishing the syntax at least in terms of user perception.	Game Maker defines the world by dividing it into one or more rooms that can contain objects. The objects represent the entities. The rooms have background graphics attached. Each object can have a graphical sprite representation that changes when an event occurs, as shown in Figure 2.7. Its interesting interface feature enables users to define behaviors quite easily and uses interesting syntax in an object-oriented environment.
Capabilities:	More suitable for undergraduate students who have some or little knowledge in programming 3D environments, since it allows students to gain experience in all standard programming commands taught in introductory classes without making major syntax errors.	This method works well in the intended domain of a programming environment for children. Regarding the children's orientation of the programming environment, one important feature of the tool is its strong connection between the expressing program and the running program. The basic actions of programming can be performed in the same fashion either in the sate or in the rule-definition window. Thus, there is a smooth transition between the execution area and the rule-definition area which is a major advantage when solving programming bugs.	Due to its ease of use and the fact that it is a freeware, it has developed into a huge, non-professional game development community. On top of that, Game Maker has an interesting track record in terms of educational value and usage, not only for teaching the technology in summer camps and elementary schools (children aged eleven to thirteen), but also in high schools and universities.

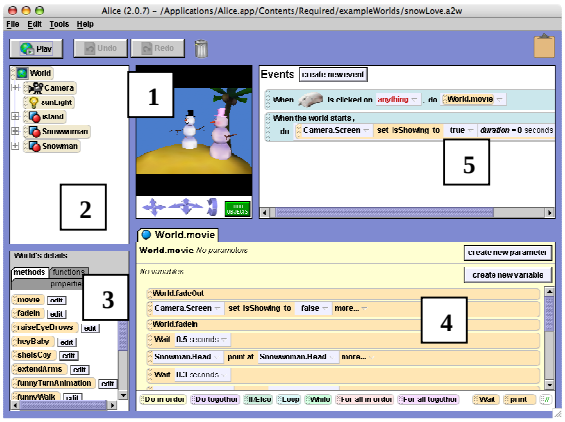


Figure 2.5 A screenshot of the Alice interface. (1) The screen window provides a view of the virtual world that will be controlled by a student's program. (2) The object tree contains a list of the 3D objects in the virtual world. (3) The object details area shows the properties, methods and functions of the object selected from the object tree. (4) The methods editor area shows the code that defines a method the user is working with. (5) The events area allows users to call methods based on events in the world, such as mouse clicks or value changes of a variable.

2.5.2 Stagecast Creator [Stagecast09]

Using Stagecast Creator, users can write a program to build games and simulations. It is easy to use but powerful enough to create 2D games. The preparation is done visually, without programming, as shown in Figure 2.6. It can be used both in the classroom and at home. The program also offers the opportunity to publish created projects on the Web and those using the Java engine directly online. Stagecast Creator (originally from KidSim and Cocoa) was produced by Apple and released as a commercial product.

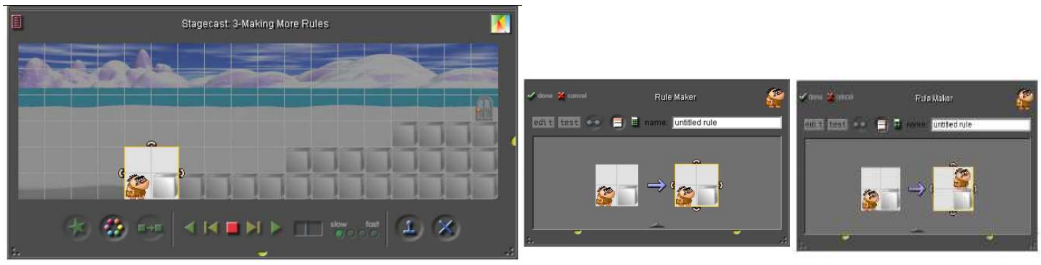


Figure 2.6 Defining the rule in Stagecast Creator by selecting the intended area by dragging its handles. (*Left*) The rule definition becomes visible on new windows, as shown on the right. (*Right*) The rule is based on the *before* and *after rule*.

User Study with the Involvement of Children: The tool has been introduced to the classroom for children to create interactive simulations not only for games but also to be integrated into formal lessons such as mathematics, languages, sciences and more. The designers of the tool claimed that it can be used by children at any grade level in a wide variety of learning styles. There are also some findings from several evaluations made upon the usage of the tool with children mainly aged ten to

twelve [Lin05]. On top of that, Stagecast Creator has received numerous industry awards and positive reviews from educators as well as from parents.

2.5.3 Game Maker [GameMaker09]

Game Maker is a game development environment which was developed by Mark Overmars and was released in 1999. With the game environment users can intuitively and without knowledge of a complex programming language, such as C++, use the drag and drop visual design, as shown in Figure 2.7. Game Maker is composed of elements from which the children can use a library and a built-in scripting language, the Game Maker Language (GML) to expand their game project. Game Maker can be used for the preparation of simple games and also for the development of larger projects. The tool gained establishment and is accepted to be used in schools to teach children computer programming, computer literacy and game design. Furthermore, it is also used in universities.

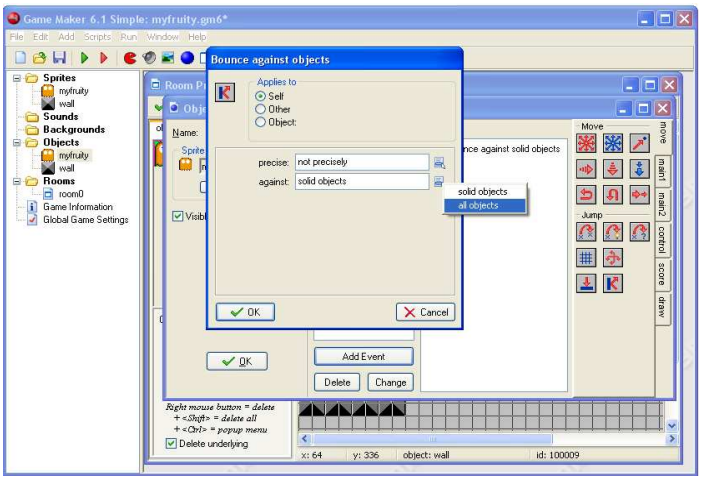


Figure 2.7 Game Maker and its interface consisting of multiple windows with own features.

User Study with the Involvement of Children: Game Maker has become widely known and used among undergraduate students also in computer science or game design and development classes. The tool is not only used to create simple arcade game, but it can also be used to design first-person (FPS) or third-person shooters (TPS) and massively multiplayer online games (MMOGs). Over the years, more educators used Game Maker for educational purposes within a variety of activities such as summer camps and for students from elementary schools and high-schools. The designers of this tool also provide teaching materials for children aged above eleven years [Robertson05]. In fact, Game Maker is also used as a tool to teach undergraduate students in applying software engineering principles in the context of game design [Overmars04][Rankin08].

2.6 Child-Centered Design (CCD)

A prominent standard, ISO 13407 (Human-Centered Design Process) defines UCD as *an approach to design that grounds the process in information about the people who will use the product* [UPA09]. The UCD approach pays special attention to users at all design process stages,

which requires the software designers to analyze user acceptance towards their software and to verify earlier assumptions regarding the actual user perception of the software. There are various methods of implementing UCD in the design process which normally consists of four software development phases: analysis, design, implementation and deployment. Among the famous methods are cooperative design [Greenbaum91], participatory design [Schuler93] and contextual design [Beyer98]. Other conventional UCD methods have been well-known to position users as testers or evaluators in order to obtain a range of feedback about the software design. In contrast, participatory design focuses on users acting as design partners [Jörgenson98][McGee05], which explicitly gives them equal responsibility to design and develop the software [Magnussen03].

The concept of UCD has been expanded to CCD [Druin99a][Henry00][Read05b]. CCD follows the same ideology of UCD including methods used and developed to support UCD, such as usability testing [Abras04], heuristic evaluation, Wizard of Oz [Höysniemi04], contextual design and participatory design [Druin99b], but with a different focus. For example, in participatory design, users are given equal responsibilities as the designers. But in the context of children as users, the same manner is not appropriate to be implemented. One reason is the fact that children themselves are not mentally and cognitively ready to discuss certain learning goals [Scaife99]. Another reason why UCD for adults should not be adapted specifically for children is the condition of the children themselves. Children have different skills than adults and therefore the methods used for designing or evaluating software should be adjusted accordingly [Bekker04]. Researchers in the field of children and interaction design have started adopting and implementing the same or similar research methods in their research within the field of CCD, including case studies, field studies, action research, lab experiments, survey research, applied research, basic research and normative writing [Jensen05]. The reasons for the differences basically have to do with the nature of the users themselves, and children definitely have different needs, characteristics and behaviors [Read06]. By definition, CCD techniques focus on children whose work is to be supported by software applications including their characteristics, tasks, and environments. On the other hand, there is also a need to look at the implications and practices of UCD in games as argued by others [Pagulayan03][Sotamaa05].

From the idea of implementing CCD together with its philosophies, this research follows some of the methods used in evaluating children's products? [Yatim09e]. Children were involved in the very early phase of the design process to give them a voice in the design and development of the game authoring tool, thus enhancing the quality of the resulting system. Hence, the obvious idea is to involve them as early as possible in the design process, but this is not necessarily followed in practice because of time constraints and other disturbing factors such as communication barriers. Consequently, the early design sessions with the children have been carried out by using high-tech prototypes (hi-fi) instead of using low-tech prototypes (lo-fi) or an intergenerational design team as central elements. For this purpose, the existing game authoring tools were used in order to get feedback and opinions from the children in form of a preliminary study. The study will be described in detail in the next chapter. Although this research does not fully use the CCD approach, the important concerns on this approach are taken into account. Nevertheless, this research follows the traditional qualitative research within the design of a game authoring tool, especially in adapting the combination of techniques in gathering data, initiating ideas, developing a prototype and testing the designed prototype iteratively.

2.7 Usability Guidelines for Designing a Game Authoring Tool

In order to come out with a list of usability guidelines, the research involves multi-disciplinary areas specifically looking at the three major activities called game design, game programming and game playing. Both, formative and summative evaluation will be conducted using this approach in designing and developing the game authoring tool. By definition, formative evaluation is the process of evaluating the effectiveness and efficiency of the game authoring tool during the development process. Summative evaluation, in contrast, is the evaluation of the completed software product. For this research, the definition of complete software is being referred to the game authoring tool that is treated as a prototype. The evaluations were conducted to measure the usability of the software including the fun aspects and to compare the findings with the earlier expectation of the research [Yatim09a]. It is argued that from the formative and summative evaluation a set of usability guidelines for developing a game authoring tool can be derived and analyzed at the end of the research.

As mentioned before, since there are no specific guidelines for designing a game authoring tool, an effort in designing one is urgently needed. To start, the existing usability guidelines were gathered from several previous works and are listed in Table 2.4 [Nielsen90][Begel97][Shneiderman97][Duda98][Gilutz02][Sheehan03][Chiasson05a]. The guidelines were picked because they offer general and specific design rules with direct details of the design. They can also be found in a variety of locations such as in professional journals, reknown HCI books or HCI handbooks, and in works based on the respective research. In the table, these guidelines are categorized into seven parts to simplify the existing guidelines by categorizing them into consistency, child-friendliness, degrees of functionality, staying in the flow, reflection, familiar conceptual model and familiar way to program, as shown in Figure 2.8. From the table, it is not assume that each existing usability heuristic will cover all seven categories, but in most cases the heuristics will be put in the most suitable category although there will be cases of overlapping, for example in consistency and visibility of the system status. Later, these guidelines will be used in a preliminary user study as described in Chapter 3. In the future, the usability guidelines will face some refinements for the purpose of this research. The aim is to achieve complete usability guidelines for designing a game authoring tool for children at the end of this research.

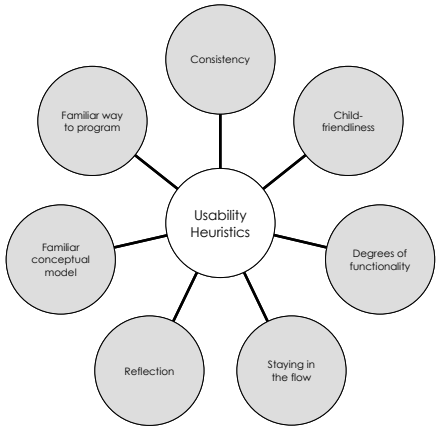


Figure 2.8 Usability guidelines for designing products for children gathered from several available resources (The figure shows the connection of these guidelines in a simpler form).

2.8 Summary

This chapter described the usability aspects of children's software. Furthermore, the relationship between usability and games was described including the aspects of fun and flow theory. In order to understand the premise that the elements of enjoyment are universal, especially in nurturing children's skills on how to program, the chapter touched the important features of educational programming language for children for further designing a plan of an ideal game authoring tool whose usability will be measured parallel to the CCD approach. The chapter ends with three well-known game authoring tools, whereas each tool was discussed regarding its programming concepts, features and capabilities. In the next chapter, a preliminary user study is conducted to clearly understand these issues. In this study, two software tools called Squeak Etoys and Game Maker are used.

Table 2.4 shows usability heuristics gathered in previous work done by other researchers.

Usability Heuristics	Nielsen90	Begel97	Shneiderman97	Duda98	Gilutz02	Sheehan03	Chiasson05a
Consistency: Ease of use, visual consistency, content-specific metaphors and as minimum text as possible.	Consistency and standard Visibility of system status	Uniformity and elegance	Strive for consistency		General interaction	Make its models explicit	
Child-friendliness: Putting together the concept of game design and interface design for children, including children's familiarities and age-appropriateness.	Aesthetic and minimalist design Help and documentation	Understandability	Cater for universal usability	Suitability for children	Graphical user interface	Interactive Usable for children of all ages	Literacy
Degrees of functionality: High-level of instructions corresponding to the things children want to represent in their games, including producing games that have been supported by the collection of resources such as pictures, sounds and interactivity.	User control and freedom Flexibility and efficincy of use	Functionality Powerful ideas		Ease of use	Text and multimedia	Collections of resources High-level of instructions	Social interaction Collaboration
Staying in the flow: Providing leverage and exciting environment for children to explore different domains of knowledge in order to capture their interests.			Fun in doing	Entertainment value		Gentle learning curve Fun	Motivation and engagement
Reflection: Indication for children to refine their ideas about the games they are making.	Help and recovery from errors Error prevention	Reflection and audience	Informative feedback Prevent errors	Educational value	System errors and help	Wide variety of programs See and manipulate	Feedback and guidance
Familiar conceptual model: Emphasis on the content-area skills including game design, control design, elements design and game mechanics.	Match with the real world Recognize rather than recall	Transparency	Design dialogs to yield closure Permit easy reversal of actions	Reduced children memory load	Navigation and search	Self-construct Reversible steps manipulations	Mental development
Familiar way to program: Understandable style of scripting and authoring that is appropriate for the children's tasks.		Interactivity	Internal focus of control Reduce short-term memory load		Content	Explicit sequences of instruction Easy to follow	Motor skills and tangibility

3
Preliminary Study of the Existing Tools with Children

In this chapter, a preliminary user study will be discussed to clearly understand certain issues. The similarities and differences between the simulation tool and the authoring tool will be discussed. To meet this purpose, all discussions made in the previous chapters are going to be implemented by conducting a real study of children using both tools. For this purpose, Squeak Etoys will be used to represent the simulation tool while Game Maker will be used to represent the authoring tool. There are two reasons why these tools have been chosen for this study. First, they can be freely downloaded. Second, both tools are designed for children to embrace creativity and to facilitate learning.

3.1 Introduction

Research on children as programmers suggests that language constructs do not pose major stumbling blocks for novices learning to program [Smith00][Sheehan03]. Furthermore, the important aspects of programming activities have become a medium for creative expression for children and offer a rich environment for them to become engaged and to learn more about programming strategies. The first stage in understanding the relationship between children and programming is to explore the earlier intention of programming for children. The purpose of programming is not only to memorize the knowledge of the programming language, but also to let the learners acquire useful, creative and logical thinking, problem solving, and modeling [Schwartz06]. As argued in Chapter 2, the skills gained in the programming activities have a close relation to mathematics, and earlier research indicates the expression of mathematical ideas in form of computer programs suggesting a promising line of enquiry [Feurzeig69]. Logo and BASIC are the most obvious examples. Yet, the idea of children programming is no longer restricted to textual interfaces [Morgado06]. The notion of programming has fundamentally shifted to include visual aspects and manipulates objects by point-and-click [Kahn01][Kindborg03]. The shift aims to provide the children with much more expressions and shall be more intuitive for them. Recently, a great interest in tools that use graphics to aid the programming activities has been applied. Visual programming, programming by example and programming by demonstration are exciting areas of computer programming research especially for children, and promise to improve the user interface of programming environments.

In order to gather information that would assist in designing the intended prototype of a game authoring tool for children, an attempt was made to find out what children already understood about programming. This chapter will describe two user studies acting as preliminary studies for the further understanding of this matter. The preliminary studies have three goals: (1) to get to know the children's abilities and skills in computer usage, (2) to get to know their problems with existing software, and (3) to suggest additional factors in designing usability guidelines that need to be taken into consideration when designing a game authoring tool for children. The aim of this preliminary study is to develop the requirements for designing a game authoring tool that simplifies the programming complexity for children to enhance their problem-solving skills by gaining information and

feedback from the usage of these two different tools by selected children. These tools are mainly designed for children to enhance their creative thinking and problem-solving skills. On top of that, these tools may give a feedback to the usability guidelines previously discussed in Chapter 2. Subsequently, a new strategy for developing a game authoring tool for children will be purposed in Chapter 4. This strategy is focused on simplifying the basis of how to program without eliminating the purpose of the programming itself. The two basic motivations driving the study were to polish the children's problem-solving skills and to empower them to learn and create games using the tool. A two-month preliminary study was conducted with a total of 66 students who attended a game making workshop. Overall, the results suggested that in order for children to get engaged with the programming language elements of the designed environment, intermediary level objects have to be included in the games. These will allow children to design games in a way that also connects them with important elements in the programming language they are working with. Therefore, this chapter reports the preliminary user study that proposes to investigate the experiences among children while using two chosen software tools that provide insights to understand the children's backgrounds. This chapter also proposes to investigate the characteristics of the software (design, usefulness, and understanding visual information on the screen) within these tools preferred by the participating children.

3.2 The Preliminary User Study

Two user studies were conducted to examine factors that may influence the perceived abilities of children towards the selected software. Several factors have been identified as critical in shaping children's perceived ability to learn and use the selected software. Four hypotheses were formulated to address the issues: (1) children with higher computer experience will have a higher ability to learn game programming, (2) children prefer a game authoring tool compared to a simulation tool, (3) boys are more fond of game design activities than girls; and (4) differences between the usability guidelines for game authoring software and other children's software will be revealed. The aim of these studies was to give children an opportunity to design and create games as an approach to develop literacy skills, and to discover the strengths and weaknesses of the selected tools. As mentioned, these studies were focused on the programming styles that children were engaged in the selected software. The studies were also conducted in order to test the usability guidelines designed in the previous chapter, including the software environment requirements found in the survey carried out earlier. Furthermore, since the software requirements are gathered through the survey, there is a need to gather the children's requirements as well. Therefore, a preliminary user study was done which comprises of two user studies with children who attended a game making workshop. Even though the studies were conducted in two different locations, the same data gathering methods and procedures were used, as shown in Table 3.1. Due to getting different perspectives on children's requirements towards the same software, the studies were conducted in different places. It is believed that when different groups of children are used as respondents, more information will be received, such as demographic profiles, preferences, gender, culture and learning styles. The aim was to gather as much information as possible regarding the children's requirements. Compared to conducting long-range studies in the same venue with the same children, this type of research is far more suitable to measure the effects of children's performance when using certain software [Kafai96b]. Nevertheless, the same procedures, methods and research instruments were used in both venues.

Table 3.1 shows description of children participating in the studies conducted at different venues.

User Study 1	User Study 2
Magdeburg, Germany 8 June until 13 July 2006 2.30 pm - 5.30 pm Zone: One Stone Center, Universitätplatz	Tg. Malim, Malaysia 22 August until 25 August 2006 9.30 am - 12.00 pm and 2.00 pm - 4.30 pm Multimedia Lab, Sultan Idris University of Education

3.2.1 Participants

Table 3.2 shows the classification of the participating children according to their background variables that were thought to relate to game exposure including age, gender, places used to play games, and game experiences. A total of thirty one (31) children, including 27 boys and 4 girls, aged eight to twelve years participated in the first study that was treated as a game design workshop in Magdeburg, Germany. In the second study, a total of thirty five (35) children participated, including 17 boys and 18 girls, aged nine to eleven years. In both studies, the subjects participated voluntarily. Since the children were asked to participate in these studies, an equal gender distribution was not achieved. 66.7% of the subjects were boys, while the rest were girls. The majority of the participants in the preliminary study were 10 years old (36.4%). The children were considered as experienced game players if they spent more than 4 hours per month playing games or if they reported to having played more than 20 types of games up to that time. One survey reported that children with computer access spent am average of 4 hours per day playing games [Subrahmanyam00]. Two factors—the hours spent in playing games and the number of games played at one time—were used, since practice and skills are two main components related to identifying game experiences. The two factors will be discussed in detail in Section 3.2.4. In summary, only 15.2% of the participants are considered as experienced games players. The participating children were not given any reward after the studies being conducted, but they only had the chance to take part in the respective game making workshop.

Table 3.2 shows personal data of children participating in two user studies.

Criteria	User Study 1	User Study 2	Percentage (%)
Age			
9	2	0	3.0
10	2	11	19.7
11	11	13	36.4
12	10	11	31.8
	6	0	9.0
Gender			
Girls	4	18	33.3
Boys	27	17	66.7
Game playing			
Experience	3	7	15.2
No experience	28	28	84.8
Total	31	35	66 children

3.2.2 Procedures

Since programming should be a vehicle to introduce formal ways of thinking to the children, they will learn how to program during the user studies. At the same time, instructions in form of a game design tutorial were given so that the children could learn how these games are working and apply their knowledge to the study. For this preliminary study, two types of software were chosen for the workshop, as mentioned before. The main reason for selecting the software is to investigate whether the software could be used as materials for learning, which software is more preferred by the children and why. The first software, Squeak Etoys, is a graphical environment that is not only used to teach programming concepts to children [Kay05], but also mathematics and science simulations [Karuno03][Matsuzawa04]. Squeak Etoys was installed at both a Macintosh Apple machine and a Windows-based PC (Personal Computer). Squeak Etoys was selected as the software representing a group of simulation authoring tools. The second software, Game Maker, was selected to represent a group of game authoring tools. Game Maker was installed at Windows-based PCs. All children were given an introductory lecture with specific tasks designed for them to follow. They also received a game design tutorial for both Squeak Etoys and Game Maker, for self-learning and group discussions. Both software tools were installed by the researchers on selected computers one day before the workshop session actually started. Figure 3.1 shows children participating in both user studies that were conducted in different countries.

Figure 3.1 Two studies conducted at two separate locations. (*Left*) User Study 1 conducted in Magdeburg, Germany with 31 participants. (*Right*) User Study 2 conducted in Tg. Malim, Malaysia with 35 participants.

In both workshops with Squeak Etoys, the children were asked to develop a game using the tool according to the tasks given by the instructors. They were asked to design a two-person racing car game. By using color coding, the 'red' car was controlled by player 'A' using the mouse, while the 'blue' car was controlled by player 'B' using the keyboard (A, W and D keys). In Game Maker, the children were instructed and guided during the design process of a simple arcade game, similar to Pac-Man. During the workshops, the usage and engagement of the children in developing the games were observed and recorded. This includes their activity levels (active, energetic or mood), social skills in sustaining communication or collaboration, and behavioral patterns (hand gestures, eye rolling or face expressions). At the end of each workshop, the children were interviewed regarding their experiences and knowledge in learning how to create games by using the tool. Furthermore, they answered given questionnaires. The children worked in a team of two to three persons,

who were involved with fun challenges which were mainly relevant to the science and mathematics curriculum. The sessions started with a 15 to 30 minutes game design tutorial lesson on selected topics followed by an hour of exploration and challenges. Tutorials and game design samples on how to navigate the tool for exploration were also provided to the children before the session started. Four instructors were involved in the study as teachers and researchers, one instructor concentrating on the Squeak Etoys workshop and the other one on the Game Maker workshop. The other two instructors acted as study observers. The observers had to fill in the screen number, the problems encountered by the children, expected causes and outcomes of the problems, how the children dealt with the problems (if any), and the software features that attracted the children.

3.2.3 Instruments and Data Collection

Before the workshop session started, all instructions and guidances were given to the children in form of a step-by-step game design tutorial intended for collaborative and cooperative learning. Please refer to the CD attached to this thesis for the Game Design Tutorial for both Squeak Etoys and Game Maker. Another instrument used to evaluate the tools was a questionnaire. The questionnaire designated as study questionnaire was designed by the researchers to elicit information regarding the children's personal data and background, including game playing information. Another section of the questionnaire contained questions regarding the usability and acceptance of the tools by the children. These questions were adapted from the original satisfaction questionnaire [QUIS09] which consists of a list of twenty-four items dealing with fairly general questions about usability and aspects of the interface. This includes the screen factors, terminology and system feedback, learning factors, system capabilities, and technical manuals. For the purpose of these studies, the number of items in the original questionnaire was condensed into a smaller number without eliminating the main usability questions. Another aspect is the use of smiley emoticons to replace a traditional point rating scale. The main reasons are to avoid using *number* as the point measurement or using weird and unknown *words* to label the point scale due to the differences in the children's cognitive level. The administration of these questionnaires was carried out after the workshops ended in the presence of the respective researchers to enable the collection of first-hand (primary) data. Although two user studies were held in separate locations at different times, the same questionnaire was used. Figure 3.2 shows the snapshot of the questionnaire mentioned and the details can be seen in Appendix A.

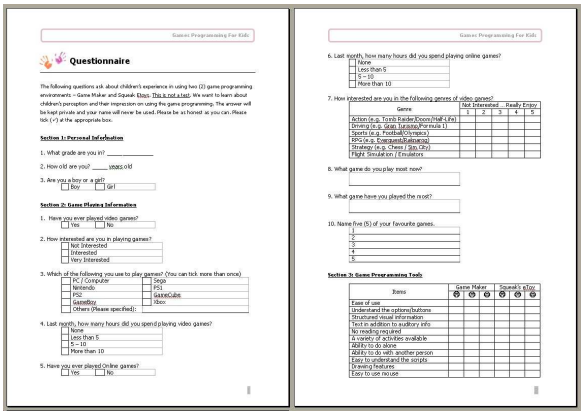


Figure 3.2 Snapshot of the questionnaire used, refer to Appendix A for details.

3.2.4 Results

On the last day of the workshop, there was a group activity where each child reported on his or her game project. In addition, data from series of interviews and questionnaires were later transcribed. Thess data give an insight into the workshop from the children's point of view, and for evaluating the features of the game authoring tools used in the workshop.

Table 3.3 shows data of children in two user studies.

Criteria	User Study 1	User Study 2	Percentage (%)
Hours per day playing games			
30 minutes	0	0	0
1 hour	12	13	37.9
2 hours	14	11	37.9
between 2 to 3 hours	3	6	13.6
more than 3 hours	2	5	10.6
PC at home			
None	0	0	0
1	18	24	63.6
more than 1	13	11	36.4
Places for playing			
Home	11	14	37.9
School	6	6	18.2
Others	14	15	43.9
Number of games played			
None	0	0	0
1 to 5	7	4	16.7
6 to 10	15	13	42.4
10 to 15	6	12	27.3
15 to 20	2	4	9.1
more than 20	1	2	4.5
Like to design games			
Yes	31	35	100
No	0	0	0

After transcription of the data, the results were grouped into four general categories, namely (1) prior experiences, (2) use of the tools, (3) game design, and (4) game programming. The main types of responses to each category will be presented in the following sections.

a) Prior Experiences.

Table 3.3 shows statistical data of children who participated in the study, including the time that children spent playing games on a daily basis. It also shows the distribution of children who have a personal computer at home, the number of games played, which involves games played on any platform, and their interest in designing games. From Table 3.3, the majority of children (75.8%) spend less than 2 hours per day playing games, while the rest spends more than 2 hours. Furthermore, all children indicated that they have at least one personal computer at home. 37.9% of the children stated that they are experienced with playing

games at home, whereas 18.2% played games at school. The majority of children (43.9%) reported that they like to play games at cybercafé, with relatives or friends or in the office of their parents. In addition, the children reported to play more than five games, and 4.5% of the participating children have already played more than 20 games. The children were asked to identify their favorite game genre. The answers varied from Football Manager to Civilization, Anno 1503, Sim Series, Diablo, the Age of Empire III, the Legend of Zelda, Need for Speed, Ragnarok, Diablo, Nintendogs, Poseidon, Sims, Bud Redhead and Solitaire. None of them was experienced in using game authoring tools or similar software to design games. But all children reported that they would like to design their own games if they had the chance to do so.

Table 3.4 shows description of children in two user studies conducted at different places (Question: Platform used for playing games).

Criteria	User Study 1		User Study 2	
	Female (%)	**Male (%)**	**Female (%)**	**Male (%)**
Platform used for playing games				
Personal computer	68.7	84.1	88.3	91.1
Mobile phone	37.2	69.4	35.4	48.6
Sony PlayStation	15.4	16.1	16.4	21.4
Microsoft XBox	9.6	9.5	1.7	2.6
Nintendo	19.1	23.1	5.4	6.7

Although the number of children involved is different (Magdeburg = 31 and Tg. Malim = 35), the situation does not affect the purpose of the study to gather the children's preferences in the tools used. In fact, more results were gained, including the preferences for different kinds of game consoles. During the studies, the children were asked about their platform used to play games. The platform may be a personal computer, a mobile phone, or a game console, as for example a Sony PlayStation (PS1, PS2, PS3 or PSP), Microsoft XBox (or XBox 360), or Nintendo (GameBoy, DS or Wii). According to Table 3.4, the children seem to be more experienced in using the personal computer as main platform for playing games (User study 1: Girls = 68.7%, Boys = 84.1%; User study 2: Girls = 88.3%, Boys = 91.1%). Other preferred platforms are mobile phones, followed by Nintendo, Sony PlayStation, and Microsoft XBox. Here, the results show that the children play more computer games compared to games that can be played using other platforms or game consoles.

One difference between these studies is the usage of the game consoles Nintendo and Microsoft XBox. In the first user study conducted in Magdeburg, Germany, the children had their own Nintendo and it seemed as if this type of game console is more famous among children compared to the Microsoft XBox. This may be due to the mobility of the game consoles produced by Nintendo, such as GameBoy and Nintendo DS. In the second user study conducted in Tg. Malim, Malaysia, the patterns are similar, since the children were less interested in playing games using the Microsoft XBox. According to the study, most of the children do not own a Microsoft XBox, because the console and the appropriate games are too expensive. Another aspect is that the children spent less time playing games using the Microsoft XBox or Nintendo, since they do not have a console at home.

Table 3.5 shows description of children in two user studies conducted at different places (Questions: Reasons for playing games and preferred playing method).

Criteria	User Study 1		User Study 2	
	Female (%)	**Male (%)**	**Female (%)**	**Male (%)**
Reasons for playing games				
Entertainment	84.7	89.1	89.9	92.8
Challenge	46.5	87.4	51.5	88.3
Graphics	23.4	45.2	32.7	57.2
Fun	45.1	64.2	48.4	72.4
Fantasy	12.6	34.6	19.8	24.4
Storytelling	76.9	51.8	86.1	49.1
Game characters	19.5	27.1	26.4	31.1
Preferred playing method				
Alone	67.1	71.9	77.5	81.9
Partner	84.4	94.2	81.9	89.7
Group	41.0	71.4	27.8	65.4

Another field that this study wanted to investigate is the reason why children play games. Which motivational factors exist? The children claimed that the major reason for them to play games is the entertainment value of the games, as shown in Table 3.5, where entertainment achieved the highest score. But there is another factor that highlights the difference between children who love to play games: the gender. At both venues, the boys picked challenge as their second favorite reason [User study 1 = 87.4%, User study 2 = 88.3%), while the girls picked storytelling (User study 1 = 76.9%, User study 2 = 86.1%). All children agreed that the least favorite reason is the fantasy derived by the game they played. The children seemed to understand and know how to rank their favorite reasons based on their experiences in playing games before. Many children were open-minded about any style of playing method, this means, they can easily adapt to the game playing patterns according to the situation. According to Table 3.5, 47% of the children preferred playing with partners or in groups, while the rest likes to play alone. This is a positive outcome, as these results show that children are willing to play with others, are well adjusted to such environments, and feel more confident if working together with others. This shows that playing games can be entertaining, it challenges the gamers to achieve certain levels, it soothes their visual sensory, and it can be a fun activity for self-pleasure or group entertainment.

b) Use of the Tools.

An effective software design for children is based on the usability of the interface. The usability of any software depends on whether or not the designers consider the mental model of children during the design process. The mental model is the first impression of users about the function of a system as well as their responses. After each workshop, the children were observed and were asked questions related to the use of each individual tool. The observation was mainly focused on the direct interaction of children with the software. It was observed that, within 15 minutes, the novices were able to create running simulations with moving interacting objects in both tools. However, they were more attracted by the

Game Maker, since it offers a real game play environment, whereas the game play environment in Squeak Etoys is not separated from the development environment. The children used Game Maker longer than the other tool and were more actively engaged in the games they designed. Nevertheless, in both tools, the children spent too much time for drawing the game features, instead of programming the game. It could be observed that the girls spent more time for drawing rather than programming compared to the boys. The boys started drawing as soon as the drawing artifacts were ready to use and they were eager to continue with the rest of the game design tasks. To relate to the findings from the observations, all participating children received a questionnaire, as mentioned before. All questions in the questionnaire had three different answer choices and the children needed to fill in their most suitable answers. Each choice is represented by a symbol: (1) ☹ means *I don't think so at all*, (2) 😐 means *neutral*, and (3) ☺ means *I strongly think so*. Table 3.6 shows the feedback (in percentage) of the children regarding the use of the selected tools.

Table 3.6 shows comparison of the results for Squeak Etoys and Game Maker in percentage (%).

Criteria	Squeak Etoys			Game Maker		
	☹	😐	☺	☹	😐	☺
Ease of use	31.4	37.2	31.4	28.6	31.4	40.0
Easy to understand the functions	25.7	34.3	40.0	17.1	37.2	45.7
Structured visual information	17.1	37.2	45.7	20.0	25.7	54.3
Variety of activities available	11.4	31.4	57.2	22.9	17.1	60.0
Ability to do alone	28.6	31.4	40.0	31.4	25.7	42.9
Ability to do with others	34.3	22.9	42.9	34.3	22.9	42.9
Easy to understand the scripts	34.3	40.0	25.7	34.3	28.6	37.2

Regarding both tools, seven criteria were asked in terms of the children's satisfaction when using the tools, as listed in Table 3.6. Each criterion represents the characteristics of the children, such as prior experience with computers, knowledge of the software, expectations of the software, preferences regarding software features, psycho-motor skills, as well as learning and understanding abilities. The functions of the tools appeared to be easy to understand, but Game Maker achieved a higher acceptance with 45.7% compared to 40% for Squeak Etoys. This pattern was similar when the children were asked about the structure of visual information and the variety of available activities. In both cases, Game Maker achieved a higher acceptance (54.3% and 60%) compared to Squeak Etoys (45.7% and 57.2%). The children also agreed that Game Maker is easier to use, either alone (42.9%) or with others (42.9%) compared to Squeak Etoys (40% and 42.9% respectively). Using the scripts of Game Maker is easier (37.2%) compared to the scripts of Squeak Etoys (25.7%). Approximately 34.3% of the children reported that they had problems in understanding the scripts of Squeak Etoys. This is a very unusual result, since Squeak Etoys is designed for younger children (aged between seven and twelve), while Game Maker is designed for older children (aged ten and above). In this study it was observed that the interface of each tool did matter to the participating children.

40% of the children agreed that Game Maker is easier to use compared to Squeak Etoys (31.4%). According to the children, Game Maker has a similar interface like common

Windows applications such as MS Word and Paint Brush that they are usually using. Squeak Etoys uses a different kind of interface that they are unfamiliar with. A simple drag and drop interface releases the children from the programming syntax and allows them to concentrate on the game design tasks. Although Squeak Etoys is a powerful and intuitive tool that aims for simplicity, the children seemed to be confused about the list of embedded tile scripts, finding the correct tile script for their intended actions, and the interaction of dragging and dropping the correct tile scripts into the script editor. These steps and tasks were far too difficult, especially for the novices. The concept of sequencing and stacking blocks of scripts in Squeak Etoys was confusing to them. The participating children found it difficult to use Squeak Etoys especially to design games, since this involves finding and selecting the appropriate tile scripting. Compared to Game Maker, it has an easy built-in animator where the children do not need to switch back and forward between the applications or even to execute the designed game. It was observed to be well suited to the children in terms of the graphical representation of objects. However, the children enjoyed the drawing tool in Squeak Etoys due to its flexibility and the fact that it resembled other drawing tools they used before. It was observed that older children participating in the studies were comfortable with using Squeak Etoys despite its troubles.

c) Game Design.

When the children were asked about the aspects of game design that were most fun, *creating game character* was selected by 51.4%, as referred to in Table 3.7. Creating game characters involved drawing and making the game environment look nicer. This also has to do with the nature of children who like to draw. According to the observations, the children spent almost half an hour just drawing the objects related to the game they created. The second favorite game design activity was the *element design* (17.1%), which involves the creation of other game characters such as game background, game utilities and so on. The tool should offer drawing features to enable the children to draw their own ideas and use their creativity freehand instead of taking them from the predefined library. The third most favorite design activity for the children was *adding the game mechanics* (14.3%). For example, the children loved to set the score points in their games. Surprisingly, they liked to set a bigger amount of scores (for example 10,000 points for each fruit). One child said that he enjoyed setting a high score, because the players who are going to play his game will think that it is "interesting and cool".

Table 3.7 shows feedback on the game design aspect.

Item	Percentage (%)
Creating game character	51.4
Element design	17.1
Game mechanics	14.3
Adding sound effects	11.4
Interface design	5.8

The children voted for *adding sound effects* (11.4%) as their next favorite. Most participants stated that it was fun to have the opportunity to choose their own sound and the right sound effects for their games. Most of them suggested that they would like to have more time and

be trained on how to create own sound effects. The last criterion chosen by the children was the *interface design* (5.8%). Upon asking why they ranked this criterion as the least favorite, they stated that they do not care much about the interface aspect as long as they can enjoy their self-designed project.

d) Game Programming.

Of all things the children enjoyed least, game programming was mentioned most frequently. It was observed that, despite the short amount of time for them to complete the games, they were always able to create working games. In fact, the children only took less than half an hour to get familiarized with the environment of the tools. It was noticed that the participants also took different roles during the game design and game programming activities. Some children were observed to be more interested in designing the games, while others were more interested in playing the games. The children ended up being more imaginative by featuring new ideas in the game design including inserting unexpected game characters such as ghosts, aeroplanes and other creative things. Another obvious observation was that the children also created games actively collaborating with other children. The cooperative aspect in designing games can act as a means towards a healthy interaction in situations where they collaborate and work together to achieve particular tasks [Kristensen03]. The children shared their strengths in designing games, developed their weaker skills and dealt with the game design problems.

As mentioned before, in Squeak Etoys, the programming style is more based on selecting and dragging predefined scripting tiles into the script windows, as shown in Figure 3.3 (Left). This procedure is helpful if the predefined scripting tile is simple and not too tedious. The truth is that there are many predefined scripting tiles in each folder and children seem to have difficulties in searching the right tile for the right scripting needed. However, the children—especially the younger ones—took more time to drag and drop the tile scripts and to put them in the appropriate spot or space in the script editors.

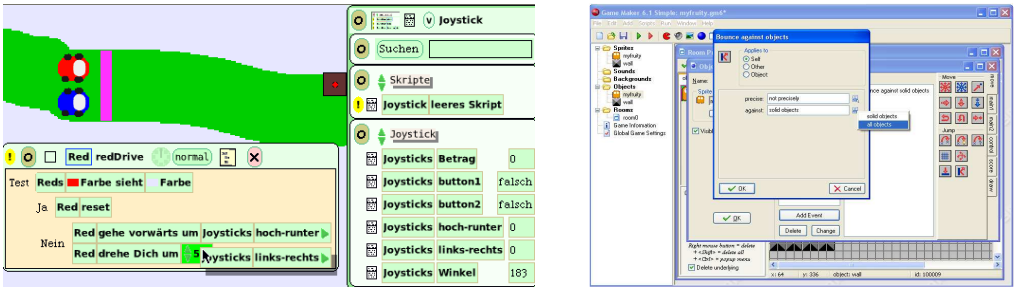


Figure 3.3 The differences of programming styles used in each tool. (*Left*) Tile scripts in Squeak Etoys. (*Right*) Drag-and-drop scripts in Game Maker.

It was observed that some of the children kept on missing the spot and had to redo the tasks. If problems occurred frequently, the children gave up and started to hand over the task to their partner. The programming style in Game Maker is completely based on the visual aspect in representing game events and actions, as shown in Figure 3.3 (Right). This shows that it is possible to create a game without using a single line of code. Game Maker also has additional features that help the children to design simple arcade game. The tool has a built-in programming language for further additional features and functions and it can be used by

older children to create more sophisticated games. The children seemed to expect the adults to instruct them on everything, especially on the game programming activities. It is believed that children need to work freely in their given environment, since the behavior of adults affects the way children will act, learn and express themselves. They seemed to copycat the steps that have been shown to them in order to get some ideas on how to complete the given tasks.

3.2.5 Discussion

The purpose of this study was to gather basic data regarding the children's usage of two software tools, and to discover the strengths and weaknesses of these authoring tools. Overall, the children found the use of both tools enjoyable. Nevertheless, both tools have advantages and limitations. In general, the children tended to prefer a free sketching feature in the tool, especially in Squeak Etoys. They can spend hours to boost their creativity by just drawing one game character. They also agreed that Squeak Etoys is more suitable for simulation activities compared to Game Maker which is obviously designed for game making purposes. This is why Game Maker was chosen as the best tool to design a game. Altogether, both tools have great programming features, especially to help beginners to make a quick start. Squeak Etoys is more superior in drag and drop features and the altered variables are visible while the program is running simultaneously. Game Maker has a conventional Windows UI, while Squeak Etoys uses the innovative Morphic UI that is different from what is currently running on most computers worldwide.

The differences in the programming styles of Squeak Etoys and Game Maker are mainly focused on the arrangement of scripts [Yatim07c][Ismail07]. Squeak Etoys uses tile scripting where the children need to find the needed blocks of scripts and drag them into the script containers. The activity of dragging and dropping the scripts seemed to be too difficult for the children, since they needed to 'drop' the scripts into the precise location in the script container. Children, who used Squeak Etoys for the first time, needed more time to learn how to do this, and some of them felt quite intense. Although both tools contain a built-in scripting programming language, allowing the children to customize their game and expand its features, the children seemed lost with all the drag-and-drop scripting functions. It is clear that not only writing the scripting but also arranging the scripts can act as a barrier for some children and prevent them from developing and enjoying their own ideas and creativity.

The major drawback of Game Maker is that it can only be used on a Windows platform, whereas Squeak Etoys runs on all platforms. Another disadvantage of Game Maker is that the games designed with it cannot run or be executed on the Web, compared to the Squeak Etoys projects. The main problem is that Game Maker is too advanced for the age of the children. The children seemed to struggle in understanding the object-oriented method of Game Maker. They still needed some guidance from the adults in order to proceed with their game project and they agreed that Game Maker is more specified for making game than Squeak Etoys. According to the children's prior experiences, they are well-verse in playing games and they have a desire to design their own games. Both tools, Squeak Etoys and Game Maker use a visual development environment, where children click and drag out events, actions or even scripts that they want to add to their created objects. Since the participating children had varying typing skills, not having to type in order to design a

game is a great feature for them. To recap the earlier hypothesis of this study, the main findings from the two studies are stated below:

- *Children with higher computer experience have a higher ability to learn game programming.* The data derived from the questionnaires show the relationship between experience and ability to learn game programming. The statistical data of children participating in the study, including the time that children spent playing games on a daily basis, shows that those children who frequently use a computer had a higher ability to learn game programming. The results were confirmed through the observations where the children seemed to have more knowledge and skills in computing and in using the tools, since they had been introduced to similar tools before.

- *Children like to use and prefer a game authoring tool compared to a simulation tool.* The results show a higher percentage of children selecting Game Maker as their favorite tool for creating games, since handling and usage are easier. The results from the questionnaires suggest that if the children are given the appropriate tools, they can engage in game creation activities that enable them to work with some powerful elements in these tools. It was observed that the children tend to be more comfortable with the computer and the software tools and are more adept in using it. They were able to grasp complex concepts in game design and game programming with little effort, regardless of how the tools were represented.

- *Boys were fonder in game design activities than girls.* There is no difference in gender preferences in designing games; both, boys and girls were eager to learn and to be involved in game design activities. However, boys and girls have their own ideas about the kind of game genre they would like to design. The boys seemed to be more motivated and fond of participating in the game design activities. This was assessed by their energy and commitment in solving the given tasks. The gender differences in designing games are not the main priority of this preliminary study, but the findings are reported in this paper [Yatim06b]. Although the total of girls involved in the studies was less than boys, the girls showed their enthusiasm during the sessions by staying longer, even if the session had already ended.

- *Differences exist between usability guidelines for game authoring software and other software for children.* The heuristics for usability both in game authoring software and other software for children are mainly the same. But the game authoring software requires some enhancements. This due to the nature of the game authoring software itself, which is a constructive tool, and the applications constructed with it are games. Therefore, some value-added heuristics will be added to the earlier usability guidelines, including the playability of the game, which will be explained in Chapter 6.

In summary, the results of this preliminary study revealed a number of interesting aspects regarding the appropriate environment of a game authoring tool for children [Yatim06b]. The following aspects will be the main factors for designing the prototype which will be described in detail in Chapter 4:

- Understandable metaphors, languages, tools, design environments and instructions embedded in these tools;

- Interactivity between the tools and children should be directly manipulable, possible course of action, instant feedback and help assistance when needed;
- Powerful ideas in making a fun toolkit so that children can explore a large design space and utilize it to learn while designing;
- Provision of functionality tools that have a variety of outputs supporting graphics and animations, and can be controlled by children; and
- A need to combine interface design with programming metaphors from the children's point of view.

3.3 Children and Game Design Activities

During the game making process, the children have to solve problems, whereas each step involves to find solutions and to get along with the process of refinement. In the games, no one tells the children the rules of the play in advance. They must figure them out themselves by observation, trial and error, and a process of hypotheses testing. The rules go beyond the decoding of the meaning of individual icons on the screen [Inkpen99a][Uden00]. Besides figuring out what the symbols mean, players must discover how they act. It is believed that the possibilities to integrate any subject lesson in the game making activities are endless, especially when it comes to the idea of using a program such as Squeak Etoys in the classroom. Creativity is not just a question of creating new solutions, but creating 'better' solutions that require a critical judgment. By understanding this, learning while making games can really connect the motivations from making games to motivations for learning sciences and mathematics.

The preliminary study confirms that creating games using game authoring tools such as Game Maker is an activity that children (aged from eight to twelve) find highly enjoyable, engaging and rewarding. These advantages indicate how game authoring tools can be used to nurture both computer literacy and children's enjoyment of game making activities. On top of that, the simplicity of visual programming and the rich features such as the drawing tool in Squeak Etoys can attract children to put themselves in a fun and creative environment towards learning how to design games. As revealed in the study, the Game Maker is far more popular and a favorite for the participating children. First and foremost, the name "Game Maker" makes a direct appeal to the children who want to make games compared to Squeak Etoys. With Game Maker, the concept of making games is simple, clean and appealing. Although games can also be developed by using Squeak Etoys, it lacks the look-and-feel of designing real games.

The most critical success factor for the children to be able to use these tools depends on the guidance of adults such as teachers, parents, or older siblings. Sometimes, these tools and their underlying concepts are too difficult to understand and most of the children had to seek advice and guidance from adults. There is a greater emphasis on teaching children as young as possible to prepare them for their future. The best way to engage children in real learning today is through computer games, not only by playing games but also by making them [Becker01]. Children are not only able to be users, but also designers and producers who gain, organize and use valuable information. Designing a game, or in other words, learning by doing, can help children to plan and validate their game plan requirements and to explore possible ways for game solutions. This process will allow children to relate the game making activities in a way that also connects them to problem-solving skills in their real learning.

To demonstrate the implications and possibilities of learning by making games, the study has a positive impact for teaching children how to make games. The goal is to provide children with a tool that enables them to create their own games. Furthermore, the children shall not learn how to program, but the tool should rather be a vehicle to introduce children to a way of thinking. In certain instances, children are able to explore the underlying programming model of the tool they use to construct games within the selected subject areas such as mathematics and sciences.

3.4 An Attempt to Refine the Earlier Usability Guidelines

As mentioned before, since there are no specific guidelines for designing a game authoring tool, an effort in designing one is urgently needed. In Chapter 2, a set of guidelines have been formulated for designing a game authoring tool for children. These guidelines are gathered from previous works which have also been used in the preliminary study. The results from the study have been analyzed and were compared with the exisitng usabilitiy guidelines. In the next step, a refinement is done within the frame of this research. The aim is to get complete usability guidelines for designing a game authoring tool for children. From the study, a number of interesting aspects regarding the appropriate environment of game authoring tools for children can be found. Most of the aspects have already been mentioned in the usability guidelines (directly or indirectly). Hence, the guidelines need to be refined based on a number of points such as (1) a need to look closer at the suitable interface and interaction for children which will reduce the burden in programming games, (2) the features of the tools can be expanded in future in order for children to gradually transform their way of thinking to the programming of games, and (3) the features of the tools are acceptable and can be controlled by children or can be suitable as a learning aid for the usage at home or in school.

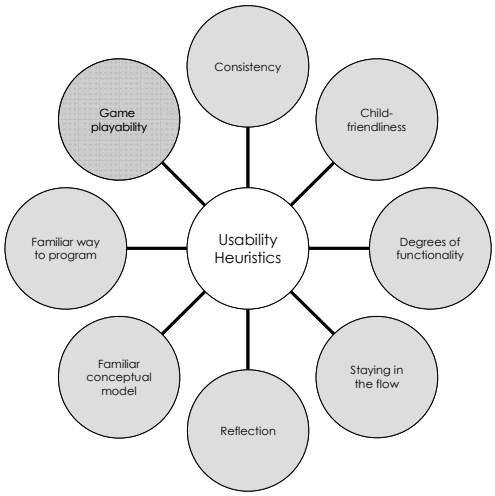


Figure 3.4 Refinement of the usability guidelines based on the findings from the preliminary study. One heuristic, namely game playability has been added.

After considering the outcomes from the studies and understanding the relationship with the earlier usability guidelines in the previous chapter, the existing usability guidelines are involved with refinement by extending each heuristic's definition and description. One

additional heuristic is *playability*. Earlier in the research, the usability guidelines concentrated on the usability aspects of a game authoring tool software without thinking about the importance of the usability of the applications designed with the tool. From the studies carried out, it was found that the children were not only enthusiastic about designing games but also playing the games they designed. The playability factor seems to be important in this type of software, i.e. software to design application software. Again, the list of usability guidelines involves minor changes, as shown in Figure 3.4. Table 3.8 shows a complete definition and description of each heuristic in the usability guidelines produced after several refinements. The list will be used again as guideline in designing a game authoring tool for children, as described in the next chapter.

Table 3.8 shows playability as additional usability heuristic and its description.

Heuristic	A game authoring tool should…	Descriptions
Game Playability	Represent supportively designed game environments such as player-created goals, game progressions, game rewards and others.	Allows children to freely create games that have the look-and-feel of real games.

3.5 Summary

This chapter illustrated the importance of knowing the real users. For this research it meant children aged between seven and twelve. The selected children were involved in a preliminary study of two tools which were Squeak Etoys and Game Maker. Squeak Etoys represents a simulation authoring tool and Game Maker represents a game authoring tool. The findings from the study are used as benchmark for designing a prototype of a game authoring tool for the purpose of this research along with the refinement of the earlier usability guidelines. Finally, the key findings were summarized in design requirements that were then adopted as goals for the game design. The next chapter will explain the modeling design and development of the game authoring tool named Gatelock, including its user interface and interaction from both the system level and the user level.

4
Designing Gatelock

The design of the prototype of a game authoring tool is related to the results of two studies witth children using the existing tools Squeak Etoys and Game Maker for creating games and for simulation, as described in detail in the previous chapter. The studies focused on the programming styles that children were engaged in. It became obvious that an object-oriented programming style was used instead of a rule-oriented one. Both styles are important to learn to build meaningful games. The results suggested that in order for children to get engaged in the programming language elements of the environment, intermediate level objects have to be included in the games. These allow children to design games in a way that also connects them to important elements in the programming language they are working with. In this chapter, the implementation of a game authoring tool for children between seven to twelve years is going to be described.

4.1 Modeling Game Programming in the Game Authoring Tool

The goal of this software development is to design a game authoring tool that simplifies the programming complexity for children to enhance their problem-solving skills. The primary objective of this study is to enhance creative thinking and problem-solving skills using the tool while another objective is to teach programming design skills in the context of game making. Furthermore, children will have the opportunity to express their own ideas creatively and to practice their problem-solving skills using the tool in the classroom [Squire05]. Further tests are planned with the tool, and additional unanticipated positive effects from the tests are expected. Two basic motivations are to enhance the children's problem-solving skills and to empower them to learn and create games using the tool. The game authoring is called Gatelock (Game Authoring Tool for Educational Learning for OLPC Kids) [Schmidt08]. Gatelock was developed as a game authoring tool which allows children to design simple 2D arcade games for a single user on a PC platform.

The design and development of the game authoring tool is guided and influenced by the needs and the ability of the tool itself to address the problems indicated in the previous section. Three contexts have been identified as most relevant to initiate the game authoring tool design: the programming context, the computer game context and the game design context. Firstly, in the programming context, the process of programming is shaped to be more like thinking for children. Children are given programming tasks designed to investigate their understanding of mechanisms in form of a game making where they have to program and design. The purpose is that the children are developing skills that are related to programming, including learning about mechanisms and behaviors. These programming skills can be seen as the target domain of the children's way of learning. Secondly, in a computer game, children will learn the basic aspects and rules of the game itself very fast. By creating games, children do not only gain authoring or programming skills whilst participating in the development process, but they will also learn many aspects about the subjects' domains, such as physics, mathematics, arts and sciences. In other words, learning programming while creating a game is not only helpful to increase their problem-solving skills, but also good because it requires knowledge about the game topic and thus motivates

learning. Lastly, for the game making the children will use basic elements of project management. These elements include planning the game, designing and building it, testing the gameplay, and redesigning it. Designing a game can help children to validate their game requirements and to explore possible ways for particular solutions. Therefore, the design concepts of the tool were divided into two main aspects: activity design and screen design. The following list shows the two main aspects that have also been referred to the outcomes and experiences gained in previous studies.

Activity design – The design activities will be based on powerful ideas in making a fun game authoring tool so that children can explore the design space and utilize it as a way to learn while designing. On top of that, the design activities will be able to attract children to enhance in an interesting and challenging learning activity. The design activities will be supported by functionality features that are easy to remember and understand, including graphics, animation and control of designed objects.

Screen design – The screen design has visually meaningful metaphors, languages, tools, design environments and instructions. The interactivity between the tools and the children should have a direct manipulation, a possible course of action, instant feedback and assistance when needed. The cursor is larger to enable a better visual communication. The usage of rollover will be incorporated with the highlighting to indicate the right functionality. Finally, the design of the tool stresses the need to combine the interface design with programming metaphors from the children's point of view.

Next, a new strategy for a game authoring tool for children that will simplify the programming without eliminating the benefits of learning of the programming itself, was developed. The strategy was based on the following three elements of (1) an educationally appropriate interface representation, (2) an interaction protocol that naturally shifts the children's attention from an intuitive interaction to focusing on the structure and operation of the representation, and (3) a gradual elimination of representation components so that children are required to take over cognitive responsibilities. An *educational appropriate interface representation* means that the game authoring tool design will have a child-appropriate interface metaphor [Ko04]. Instead of using the desktop metaphor commonly used in most software, a different metaphor will be used rather than having office peripherals such as file folders. This is due to the fact that children are not experienced enough with office environments [Inkpen97a]. However, some of the regular operations such as delete and undo are necessary in software for children. Another point that needs to be stressed is the intellectual programming power brought by the game authoring tool especially for learning. Furthermore, the *interaction protocol* used in the game authoring tool will feature an easy way for children to begin programming while maintaining the required flexibility. Therefore, the mouse will be considered as main interaction device instead of the keyboard. On top of that, a game pad can also be used for testing and playing the designed games. The devices were chosen to facilitate the children's interaction which concerns the usage of the tool and how the tool actually works. In other words, to make a connection with the tool, children should be able to deliberate results by using means of suitable interaction devices efficiently and effectively. In addition, visual and graphical programming will be introduced instead of drag and drop programming. The reason is to make the interaction more accessible to the children with a clear programming representation. The *gradual elimination of representation components* means that the design of the game authoring tool is

inherently interesting and challenging for the children. The game design activity in the tool will allow them to learn while moving from one level to the next. Thus, the tool will allow the children to maintain a strong focus on the game construction and the process of deliberation.

4.2 Gatelock and the Prototype Methodology

Since the research focuses mainly on usability, the well-established design principles from usability engineering were acknowledged. A general approach towards the usability engineering involves three main steps which are focusing on children as early as possible, using effective empirical measurements, and designing iteratively [Mayhew99]. The approach was used by usability engineering experts and is equally applicable to any software project. One of the famous methodologies which also incorporates and enhances the three main steps in usability engineering is the Usability Engineering Lifecycle, where the usability activities are presented before, during and after a certain software is designed and implemented. Meanwhile, the fastest and cheapest methodology in the software design and development lifecycle is prototyping. Here, by definition, a prototype is a basic working model of software usually built for demonstration purposes as part of the development process. Therefore, both methodologies have been incorporated and integrated in this research due to the significance and complexity of the game authoring tool. Since the research had limited time, budget and resources, the prototyping methodologies were implemented with the knowledge of the usability engineering lifecycle approach. In other words, some recommendation steps, activities or methods were treated and used throughout the development process.

Another aspect of the tool design are the evaluation methods. In usability engineering, there are various usability evaluation methods, whereas each of them is served differently according to their purpose of use. For software or prototype testing, to see the effectiveness of the user interface in supporting children to do their tasks, researchers can use methods such as coaching, think-aloud, remote testing or a question-asking protocol [Chiasson05b][Als05]. For inspection or examination of usability-related aspects of the software user interface, researchers can apply cognitive walkthrough, feature inspection or heuristic evaluation. For getting information from children regarding their understanding of the software, researchers can use field observations, questionnaires, interviews or logging data. For the research, this domain of interest will shift towards the evaluation methods used when children are involved. There has been a growing discussion focused on this matter that has lead to the development of usability evaluation methods for children within the domain of usability engineering [Druin99a][Markopoulus08]. Given the large number of methods available and the different ways of classifying them, this matter will be described in the chapters that are related to the evaluation of the designed tool. Overall, the choice of evaluation methods depended on the costs of the evaluation itself, the appropriateness to the project, and time constraints.

As mentioned, the game authoring tool was design and development following the prototype methodology. Therefore, the design of the tool was separated into four phases, whereas each phase has it own outcomes and contributes to the other phases iteratively. This life cycle is also concurrent with the evaluation studies carried out during the design process, as explained in Chapter 5. The tool was designed and developed using the prototype methodology due to several reasons. Firstly, to stimulate certain aspects of the

actual program features that may enable a faster feedback from the children as early as possible. Secondly, by using the prototype, researchers can match and compare the prototype with the software specification, in this case, the usability guidelines. The prototype can shorten the software development lifecycle and improve the information sharing that helps the designer to speak more constructively. Thirdly, to avoid a longer time required to design the tool within a limited budget and short expertise. Lastly, the children's involvement in designing a game authoring tool which is satisfying to them in terms of its look and feel shall be increased. Figure 4.1 shows the relationship between the phases and activities carried out during the design and development of the game authoring tool.

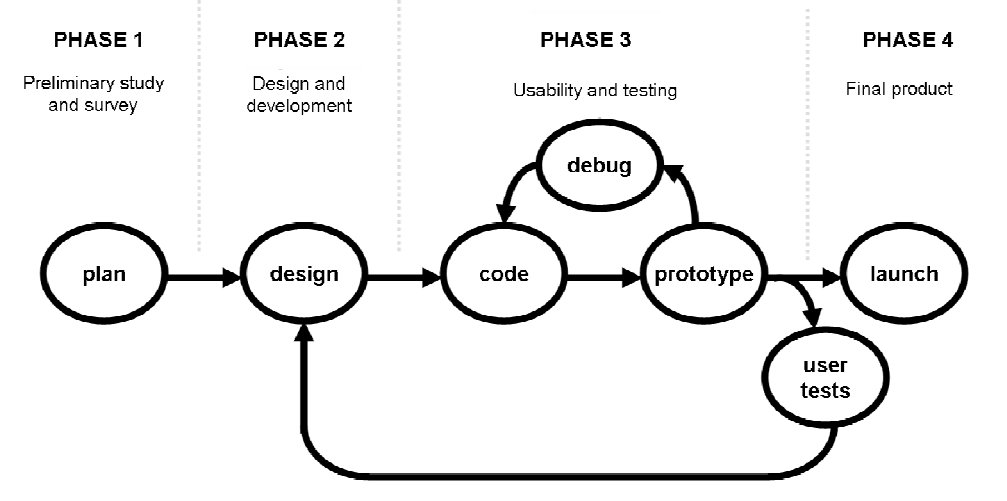


Figure 4.1 The design and development of Gatelock was carried out using a prototype methodology.

Phase 1 – Requirements Analysis (8 months): In this phase, the existing related tools such as Squeak Etoys and Game Maker were analyzed. Other educational programming tools such as Agentsheets, ComiKit and others were involved in a survey in terms of their status, improvement and innovation. Both activities were carried out before the game authoring tool was designed, and were explained in Chapter 2 and Chapter 3. Findings from both the surveys and the preliminary study were used and treated as main input for the next phase.

Phase 2 – Design and Development (12 months): The design and development phase was carried out iteratively with intended evaluations and usability tests. This indicates that the design was utterly changed, due to multiple evaluations and findings from the usability tests conducted with the participating children. The design also faced minor problems, especially in preparing and implementing the game engine framework, as will be further described in this chapter.

Phase 3 – Usability and Testing (12 months): Several evaluation and usability tests were concurrently conducted in this phase, while the prototype was still under design and development. The tests were treated as formative and summative evaluation in

order to get feedback from the children, both during the design of the prototype and after the design of the final prototype.

Phase 4 – Final Product (including the usability guidelines): In this research, the prototype is defined as the incomplete version of the real software, which signifies that the prototype and its specifications can be improved as evolutionary prototype, or can be discarded to show the real requirements and how the tool may look like when they are implemented in the finished tool. Here, in phase 4, it will be decided if the prototype can be declared as final product or if it will be refused and if the findings and outcomes will be treated as input for designing a new game authoring tool. Therefore, the vision of the final game authoring tool will be covered in the last chapter.

4.3 Gatelock: Requirements Analysis

The requirements analysis is the first stage in the development of the game authoring tool and it requires a strong relationship between children and tool designers, especially to identify and gather their needs and document them in a proper order. Several techniques can be used such as conducting interviews or focus group workshops, setting a list of potential requirements, designing a low- or high-fidelity prototype or asking the children, since they know best what they need and want. For this research, a combination of these methods was employed to establish the exact information needed so that the designed tool meets the children's requirements. For example, a *survey* of features and capabilities of the existing software tool, as described in Chapter 2, has provided the tool designers with *a list of requirements* for children's products. Besides the survey, the tool designers also had the chance to use (*hands-on*) a number of existing tools such as Squeak Etoys, Game Maker, ComiKit and others, to get a feeling for the use of these tools. Another method was applied in the preliminary user study described in Chapter 3. A *questionnaire* was used to reveal requirements that were not detected in the previous survey before conducting the questioning session. In other words, the findings from questioning the children can confirm and also enhance the list of requirements established earlier. An informal *interview* was also conducted in combination with the questionnaire to provoke the children with discussions about their feedback. Therefore, the requirements for the Gatelock project were based on the findings provided by conducting multiple requirement gathering methods. On top of that, some problems were identified during the requirements analysis, such as the incorrect believe to get a perfect agreement between the designers. Consequently, the analysis was carried out by the designers themselves instead by an expert in knowing people and with a proper knowledge about children's needs. Besides this, due to little time, the designers developed the requirements analysis specifications without assistance from a technical writer. It is wise to include a technical writer throughout the entire process of the Gatelock development, since this offers several benefits [Mayhew99]. Fortunately, all requirements for this research were based on the children's and the designers' perspectives. It was believed that the requirements analysis specifications can govern the design and development process of Gatelock and serve as validation check for testing and evaluating its usability aspects that will be described in the next chapter. Table 4.1 shows a list of the Gatelock requirement specifications that were divided into three main components: functionality requirements, design requirements and architectural requirements.

Table 4.1 shows description of Gatelock's requirements.

Functionality Requirements	Design Requirements	Architectural Requirements
☐ Creating a new game project, editing and deleting game projects. ☐ Browsing options. ☐ Drawing functions. ☐ Automatically saving of the current game project. ☐ Storing game design project artifacts for later retrieval. ☐ Allowing integration with other applications. ☐ Providing a game design environment for creating and playing the designed games. ☐ Allowing the operations *undo* and *redo*. ☐ Switching between development area and play area.	☐ Making a fun tool. ☐ Allowing children to explore and utilize the design space. ☐ Functionality features are easy to remember and understand. ☐ Using a simple and robust type of game programming for children. ☐ Visually meaningful metaphors and design environments. ☐ Direct manipulation. ☐ Instant feedback and help assistance when needed. ☐ Larger cursor. ☐ Rollover effects for highlighting functionality. ☐ Mouse interaction.	☐ Making a localized game authoring tool to be used with the PC. ☐ The tool incorporates a game engine. ☐ The programming is done using Squeak programming language. ☐ The images are in *.png format. ☐ Support open source architecture. ☐ Not a web-based or online game authoring tool. ☐ Teaching and learning aids for educators and children. ☐ Minimum requirements on hardware and software for school's lab usage.

4.4 Gatelock: Design and Development

The design and development phase was done iteratively with intended evaluation and usability tests. This indicates that the design was utterly changed, due to multiple evaluations and findings from the usability tests conducted with the participating children. The design also faced minor problems, especially in preparing and implementing the game engine framework, as further described in this chapter. In this section, both activity design and screen design will be described in accordance with the Gatelock design intention. But first, before explaining the implementation of the two main design concepts of activity design and screen design, details regarding the programming environments and tools used to design Gatelock will be described. Gatelock was developed by using three tools: Squeak, Tweak, and a game engine named iEngine. All tools were derived from the pioneer and oldest programming language Smalltalk.

a) Smalltalk

In the 1970s, researchers at the Xerox PARC Research Center developed the first consistent object-oriented programming (OOP) language called Smalltalk [Ingalls97]. Initially, the language was part of the Dynabook project, which was derived from Alan Kay's idea in the 1960s. Smalltalk was as a simple programming tool for children of all ages to help them to simply design their own programs. The idea of Dynabook has established a vision of a dynamic and interactive media device (such as today's notebooks). Its development environment enabled children to experience an easy-to-use platform. At that time, the use of Smalltalk in local schools also helped to create a system that is more responsive and fun for children, especially regarding the operation mode. The first draft of the language was called

Smalltalk-72. Later it became a professional programming language and was commercially deployed. With Smalltalk-80, the language was standardized and became publicly available. The interface of Smalltalk-80 is represented in Figure 4.2 (*Left*).

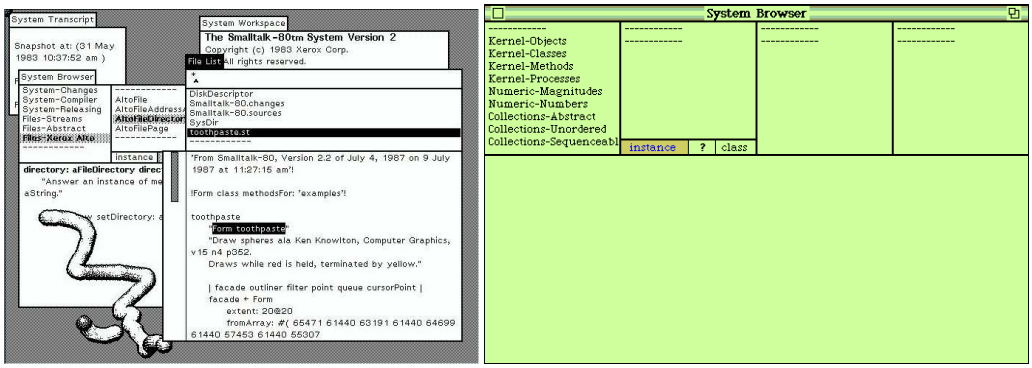


Figure 4.2 Smalltalk. (*Left*) Smalltalk-80 version taken from the original screenshot [Ingalls97]. (*Right*) Squeak interface based on the integration of the Smalltalk language and the Squeak system browser.

In the middle of the 1990s, Smalltalk faced an advancement for the development of a complex and multi-facet system that was adapted in a changing world especially from finance and insurance companies. The language became highly demanded for its productivity and famous for its short development cycles. Smalltalk was claimed to be a good tool for any rapid development of interactive software prototypes. All this happened, when Smalltalk changed into a more user-friendly environment. A Smalltalk system typically consists of an integrated development environment, in addition to a graphical user interface with a window display and extensive class library addition, a compiler, debugger and a browser system. Many ideas, such as a graphics screen with sliding windows and writings of various sizes were considered in the new Smalltalk development environment.

b) Squeak

After a long exploration of programming environments for children, Kay and his team continued their research to make the concept of Dynabook realizable [Ingalls97]. As a result, Squeak was born as a versatile development environment with the creation of multimedia elements. Besides the programming possibility, Squeak provided various graphics, sound and animation tools, and supported many popular media formats to create an application. Through numerous enhancements and adjustments over time, Squeak became a modern platform of the Smalltalk development environment, although the language syntaxes are still very close to the Smalltalk-80 standard. All syntaxes are situated in a single file called image that is the current state of all objects stored. When Squeak was used on a different platform, an image was implemented on its virtual machine that used the *Model-View-Controller* (MVC) architecture as presented in Figure 4.2 (*Right*). The MVC was used as Smalltalk-80 user interface to give visual feedback such as overlapping windows for the user to handle objects. Another user interface environment for Squeak is Morphic, as shown in Figure 4.3. Morphic is a direct-manipulation user interface replacing MVC for Smalltalk-80. It includes a framework for building graphical objects called morphs which can be interactively animated with a set of tools to support the Smalltalk environment.

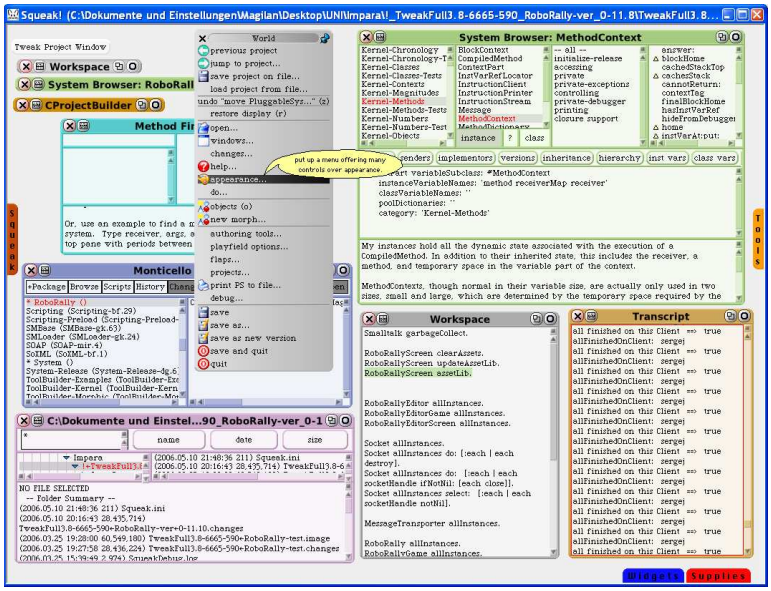


Figure 4.3 Squeak Morphic interface including its system browser.

Squeak is an open-source project that is under liberal licensing on the Internet for free. Squeak offers the potential to strike a bridge to computer science field and the role of the computer as a medium for learning is mentioned simultaneously during the 80s and 90s. Systems like Squeak offer rich-media applications particularly for children to nurture their thinking in a rich and playable way. One famous application of Squeak is Squeak Etoys that offers rich and playable ideas for learning [Guzdial04][Rose04][Gadegast05]. With the help of Squeak Etoys, a graphical scripting language simplified in Squeak (as shown in Figure 4.3), one can create simple or complex animations and simulations, even without major programming. Children can drag and drop blocks of scripts to produce programmes and test them simultaneously, since all changes are immediately noticeable. Thus, it is possible to quickly and easily try out different ideas. The simple handling of the Squeak environment and the easy to understand language script in Squeak Etoys allow children to learn mathematics or physics in school in a playful way by visual simulations [Conn03][Denker04]. Beside Squeak Etoys, another example of application designed on top of Squeak is Plopp [Schuster06].

c) Tweak

While Morphic offers a user-friendly graphical interface to directly manipulate objects, to understand what happens behind the program codes can cause problems for programmers, particularly for Squeak beginners. Not all Morphic functions are described in the available tutorials. There are serious inconsistencies in the entire system which makes the Morphic system hard to debug [Impara09]. A new form of framework called Tweak was developed. The basic idea behind Tweak is to integrate the benefits of Morphic and MVC into Squeak. Morphic is a great architecture when it comes to direct manipulation. But it has unsuitable architecture to create reusable and flexible components. It is not impossible to do this, but it is very difficult, since there are no Morphic abstractions. The MVC architectural pattern on the other hand, has some excellent features with abstraction. Since MVC uses a viewing architecture, it promotes a better separation of relations between the model and the view.

Unfortunately, the possibilities of MVC became exhausted. Tweak approaches these problems and tries its best to cater them. It is also directly accessible as Morphic and provides a similar view of available MVC architecture.

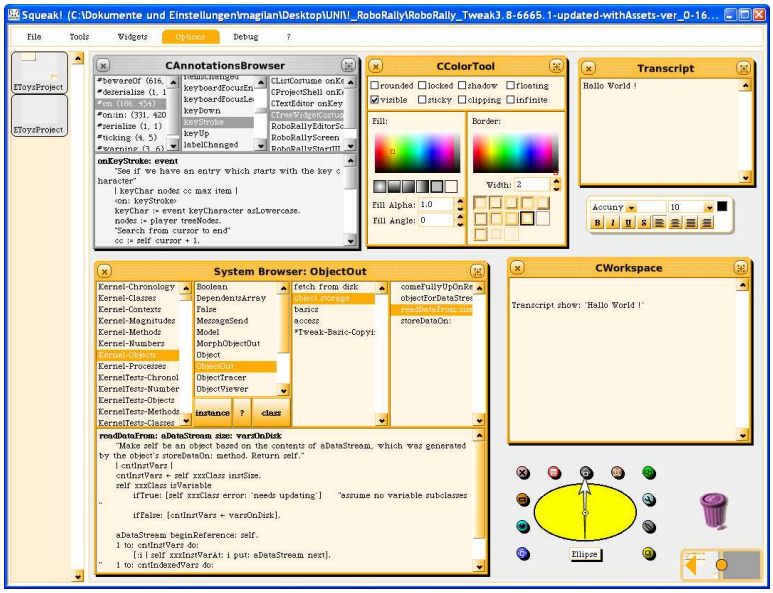


Figure 4.4 A mix of the Squeak-Tweak working environment [Roth07].

In addition to a revised user interface, Tweak brings along a number of new concepts such as a scripting environment, fields, asynchronous messages, as well as players and costumes. The equivalent to morph into Squeak is the pair of objects in Tweak which are the player and the costume. Each represents a different graphical component. Another key idea in Tweak is the asynchronous event handling. In Morphic, the morphs communicate among themselves and react to events with the help of polling. Polling indicates that a program checks in a continuous loop if an event occurs. Tweak on the other hand, provides a different mechanism. A class definition known as fields is used instead of ordinary instance variables. If the value of the instance object changes, all attached objects will also change via asynchronous messages. The scripting in Tweak was motivated by Squeak Etoys and uses its semantics. A similar concept described above is a stepping into Morphic mechanisms with the help of polling work that is also available in Tweak. In addition, the Tweak implementation offers a number of widget templates, such as simple geometric objects and various additional buttons, as shown in Figure 4.4. It is believed that by assembling graphical components using drag and drop programming, novice programmers will not face any difficulties. Widgets can be easily selected from the menu and integrated into the chosen layouts. Nowadays, Tweak has become a stable user interface framework, which, with its new concepts, can more and more act as an alternative to Morphic. Its clear architecture enables a relatively easy creation and the use of reusable components.

d) iEngine

A game engine named iEngine was designed using Squeak by the company Impara [Impara09] in Magdeburg. It provides support for single- and multi-player games from different genres. The core of the engine is an object-oriented world with different physics,

collisions and rendering engines that can be docked. Cartoons, motions and physically correct physics simulations give the play its characteristic object behavior [Masuch05]. The iEngine was successful in game projects like Pirates, Steamers Battle, Buccaneer Inc. [Masuch05] and Robo Rally [Roth07]. The technical functionality and the conceptual view of the iEngine are shown in Figure 4.5 [Nacke05]. The iEngine is a universal game engine designated for designing different game projects. It provides many useful tools that enable the development of much easier games, including the implementation of multiplayer online games and animations [Berger07]. Thanks to its platform independence, the underlying development environment Squeak or Tweak makes games in various developed operating systems possible. The functionalities of iEngine are constantly improved and expanded towards a 3D implementation in order to fully support a 3D interface and environment.

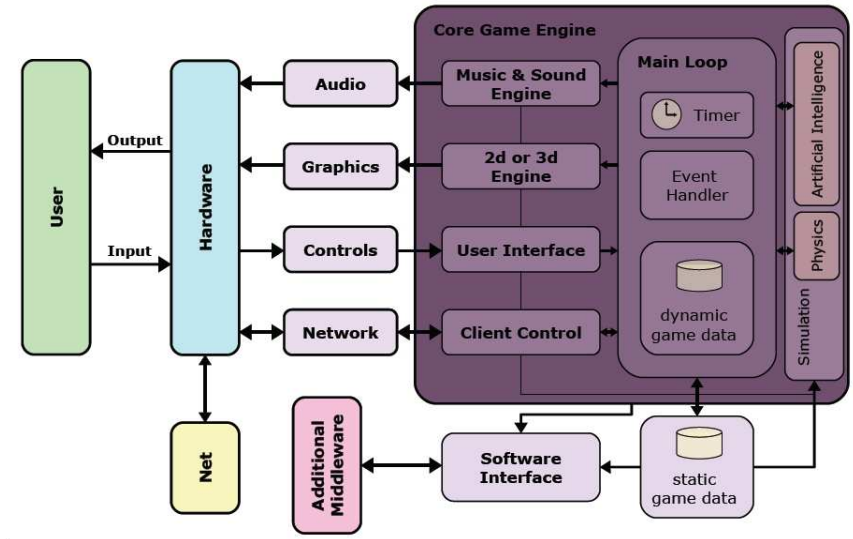


Figure 4.5 A conceptual view of the iEngine [Nacke05].

The integration of the tools Squeak, Tweak and iEngine was used for the design of the intended game authoring tool named Gatelock. Gatelock is implemented in Squeak, because the targeted platform of the tool is an OLPC XO laptop. The laptop was designed by the OLPC (One Hundred Dollars Laptop per Child) initiative whose emphasize is on design and software technology in the field of children's development in the classroom. By keeping this intention in mind, various OLPC XO revisions have been used for testing during the creation and development of Gatelock. The prototype was used at the earlier stage of development together with the adaptation of the hardware restrictions in the OLPC XO laptop. The OLPC XO laptop is a laptop especially developed for children in developing and emerging countries [OLPC09]. The laptop is a supportive assistance for learning in schools, but also for self-learning. The laptop has been revised many times and went through many innovations [Perry07]. For example, Revision 1 and Revision 2 were equipped with an AMD Geode processor with 366MHz and 128MB memory. Until the date Gatelock was last tested, Revision 4 was equipped with 433MHz, 128KB L2, 32KB L1 Cache and 256MB of RAM. In addition, it was equipped with an operating system based on a Linux version with an interface implemented in Python specifically developed for the OLPC. Figure 4.6 shows the main physical features of the OLPC XO laptop and its user interface known as Sugar.

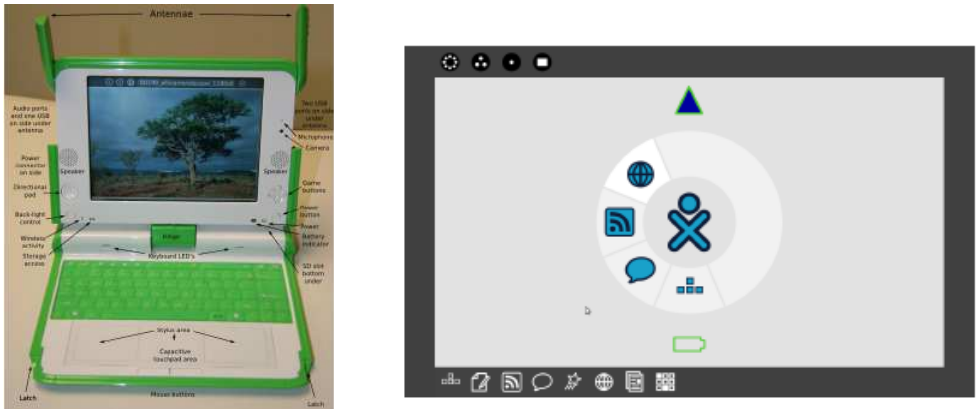


Figure 4.6 OLPC XO and its user interface known as Sugar. (*Left*) Main physical features of the OLPC XO including Wi-Fi antennas, LCD screen, game buttons, microphone, keyboard and so on. (*Right*) Screenshot of the user interface, known as Sugar, displaying a world of collaboration capabilities and abandoning the normal desktop metaphor.

The development of Gatelock was based on several works. Hinze's project contributed to the capability aspects of the Tweak implementation [Hinze03]. Nacke contributed the idea of designing a tool for rapid game prototyping named LeGaCy which exercised the usage of the iEngine [Nacke05]. Schmidt and Roth contributed to the testing of the iEngine and implemented the idea of having the Gatelock working on the OLPC XO platform. At the end of the work, Gatelock was proven to work successfully on the OLPC [Schmidt08]. However, one problem occurred: Gatelock could not be used for testing with children due to the limited number of OLPC XOs at that time. During the development of Gatelock, the OLPC XO Revision 4 was used and the production of the latest version was still in process. The Gatelock developers had to switch the implementation of Gatelock from the OPLC XO to a normal PC. At the end, these issues could be resolved and the main research was carried out up to the evaluation of Gatelock by the children. The flexibility of Gatelock makes the evaluation and testing during the field study easier, since the evaluation was carried out in remote locations and in two different countries. The flexibility of the integration of Squeak, Tweak and the iEngine shows the positive possibilities to implement Gatelock to a PC platform. From now on, all explanations regarding the implementation of Gatelock are referred to the use of the tool on a Windows-based PC. Gatelock consists of two main parts where different tools are used. Tweak is used for presenting the game content in the development phase where it is possible to add new game content or pictures to a new or existing project. During the playing phase, Gatelock uses the iEngine as main game engine for presenting and executing the contents in the game project to the children, and for user interaction with the game. The relationships can be seen in Figure 4.7. The two main parts of Gatelock are the development of game content mode and the playing area mode. While designing their game contents, the children actually use Tweak. While playing or testing their game creation, the children use the iEngine and can eventually see the outcomes of their game design.

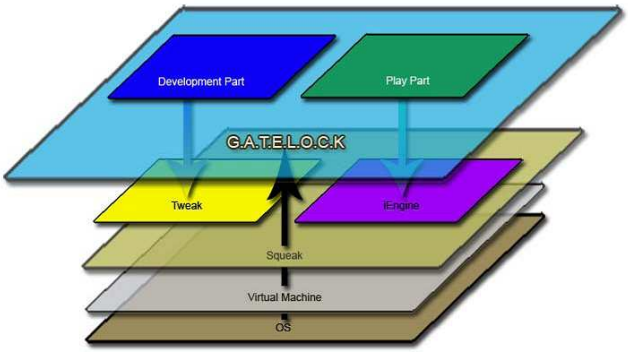


Figure 4.7 Relationships and flows between Squeak, Tweak and the iEngine [Schmidt08]

4.4.1 Activity Design

It was agreed that the requirements analysis carried out earlier sets the planning scene for the game authoring tool. Next, during the design activity, these requirements will transform the children's activities by having new technologies, new tasks and new experiences. As mentioned before, the design activities will be based on powerful ideas in making a fun toolkit so that children can explore the design space and utilize it as a way to learn while designing. On top of that, the design activities will attract children to become enhanced in an interesting and challenging learning activity. The design activities will be supported by functionality features that are easy to remember and understand, including graphics, animation and control of designed objects, as shown in Figure 4.8. The activity design for the game authoring tool will emphasize on what is being designed; this aspect is also referred to as task-level design. In addition, the three activity design concern guidelines are *effectiveness* (designing tasks that meet the children's needs), *comprehension* (designing concepts that children can predict and understand in accordance to their mental model), and *satisfaction* (designing tasks that motivate children and lead to the feeling of accomplishment and satisfaction). One thing to bear in mind is that the game authoring tool will allow children to design simple 2D arcade games using a simple game programming style that fits them. To achieve this, an analysis of activity scenarios was conducted to capture key ideas to build design rationales and to document problems that need to be addressed during the user interface design by conducting brainstorming sessions with the children in the preliminary study. This approach will nurture the participating children to share and elaborate rough ideas before designing their game authoring tool.

Figure 4.8 Interface screenshot of the high-fidelity prototype.

4.4.2 Screen Design

The screen design ideas are concentrating on visually meaningful metaphors, languages, tools, design environments and instructions. The interactivity between the tools and the children focused on direct manipulation, possible course of action, instant feedback and assistance when needed. For example, the cursor was designed larger for a better visual communication and the rollover was used to highlight and to indicate the right functionality, as shown in Figure 4.9. Finally, the design of the tool stressed the need to combine interface design with programming metaphors from the children's point of view. The aspects of user interface and interaction design will be explained below.

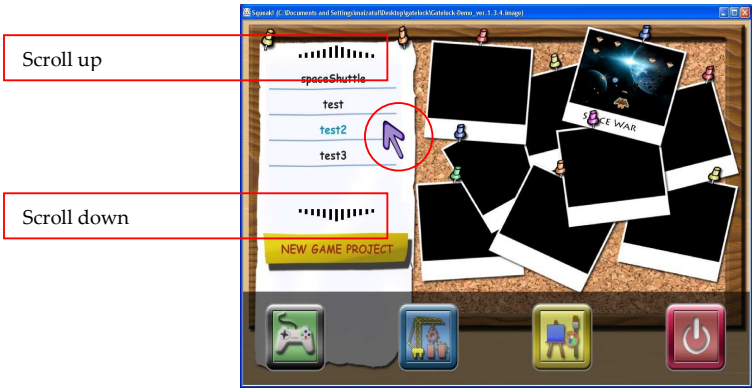


Figure 4.9 Large cursor and scrolling method used in the game authoring tool.

4.4.2.1 User Interface

In this section, the design of the interface is described in relation to a simple handling by the children, as mentioned before, including the needs to establish an attractive easy-to-use design package for children. The complexity of the development environment will be reduced through a menu management, which also helps to prevent errors. The entry into the development environment should be as simple as possible, so that first positive experiences with the development of games can be gained early. Firstly, the interface of Gatelock is the most important element that the children will see. If the interface is poorly designed or inconsistent with the children's needs, they are likely to reject it. Thus, the aim is to make game programming for children more like thinking. There are three ways to embrace the children's programming: by having an enactive representation, an iconic representation or a symbolic representation [Smith96]. For this research, images that stand for a concept without fully defining it were used to fulfil the aim by having meaningful icons, links, menus, layout, navigation, labels and feedback. In the following, the design of the user interface will be described in detail according to three different screens in Gatelock, which are the main menu, the development area and the play-test area.

a) Main Menu

The main menu screen for the development work is designed to display the selection of existing projects, the creation of a new project and the selection of the appropriate module. Figure 4.9 (Left) shows a subdivision of the screen into three areas. In the first area, an existing project can be selected from a pick list, and a button with a new project can be

deployed. The projects will be visible in the selection list, whereas a unique name specifies a new project. If the amount of projects exceeds the size of the retention area, a scrolling technique will be used that consists of an up-and-down button and a *pusher* for faster scrolling. The scrolling technique will allow the children to expand all existing projects in an organized way. Figure 4.9 (Right) shows the three sections of the user interface of the main menu which consists of the listed projects area, project preview area and main functions selection area.

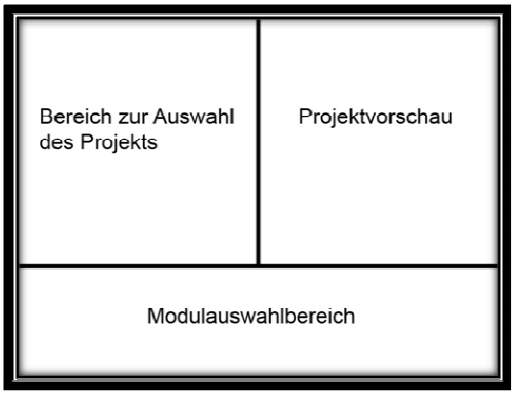


Figure 4.10 Three sections of the main menu screen.

Figure 4.11 Screenshot of the main menu with its features. (a) The selection of existing projects can be made by choosing the name of a project. The thumbnails of the project will appear in the preview area. (b) The four main functions of Gatelock are symbolized by four icons (play, develop, draw and exit).

The second area is the preview of a project selected by the children. This facilitates the option to recognize older projects, and gives a brief overview of the nature of the game that was exchanged between the children. Children are enabled to select the Preview button to switch the appropriate project records, as shown in Figure 4.1 (a). The preview area resembles a Polaroid picture and shall clarify that it is a photograph of the current development status of the project. In addition to the scroll buttons, a *Clear* button was added in order to delete or to remove a selected project. The button is marked differently compared to the other buttons, including a visual difference between the elements to show. The third

and final area of the screen are the side functions play, develop, draw and exit, as shown in Figure 4.1 (b). Each element has a clickable mouse-over image as visual feedback. The bulletin board in the background is used to display games created with the help of the development environment. These are part of the picture and are not dynamically adapted to each child.

b) Development Area

The development area starts with a white, empty space similar to a blank sheet of paper to symbolize a drawing canvas for the children to design their game. All objects related to the game project can be placed in this empty field. All necessary development tools are available as attached extendable menus, as shown in Figure 4.12. The extendable menu has several advantages, for example that no scaling is necessary between the development area and the play-test area. The available presentation of the game corresponds to the presentation in the development area. This mean, the game will have exactly the same design that the children created in the development area. Furthermore, it facilitates the convenience for children, since only related menus will be displayed. The design concept is similar to the surface of the operating system. The entire outer edge (the dark grey area in the image) can either be hidden by touching the outer edge with the mouse or appears back by pressing any keys.

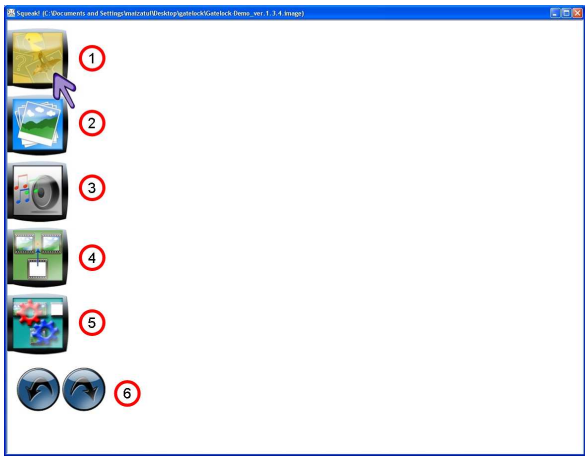


Figure 4.12 A white space indicates the area where children can insert game objects and level backgrounds. On the left side of the screen, children can choose between six elements for designing their game project: (1) game content, (2) pictures including background, (3) sound, (4) level list management, (5) Option Editors, and (6) Undo-Redo options.

c) Play-Test Area

The play-test area is the screen where children can test and play their designed games. As shown in Figure 4.13, the screen has pop-up menus for playing, developing and main menu, while the drawing function is disabled at this time. The area allows the children to try out their game concepts or game rules created before. The design of the play-test area focuses on the simplicity for children to switch from the development area to the play-test area and vice versa.

Figure 4.13 Screenshot of the play-test area.

4.4.2.2 Interaction Design

In plain words, interaction design is the creation of a dialog between a user and a system. In addition, the interaction design for the game authoring tool marks the next step in the design process right after several user studies have been conducted and evaluated. It was implemented directly after the collection and analysis of the findings from the studies with the existing game authoring tools. It started with drafting the user interface and forming a vision of the new game authoring tool environment using a high-fidelity prototype. This includes determining and selecting the type of interaction and the assignment of suitable interaction elements. For this purpose, the designers conducted a special discussion session to create a physical model of the game authoring tool by determining the required platform, interaction type (mouse, keyboard and so on), and elements in the user interface based on the usability guidelines as described in the previous chapter. Above all, the interaction design of Gatelock will emphasize on two common mouse interaction styles which are *drag and drop* and *point and click*. Other interaction devices such as joysticks and other platforms such as tabletops have been tested with Gatelock. During the development, Gatelock has shown its potential to be used with a game pad as game control device and a tabletop with pen interaction, as shown in Figure 4.14. However, due to technological changes, still some more work is needed, including the requirements analysis. For example, for the game authoring tool to be used with the tabletop, the perception of children towards the interface design of a tabletop should be considered. In other words, designing particular software for the tabletop is different compared to other technological devices such as computers or mobile phones. Subsequently, the interaction design will be explained according to the Gatelock features and functionalities that the children are using. The explanation also involves the interface design implemented in the three different areas referred to as main menu area, development area and play-test area. Each function serves different tasks and purposes, as simplified in the following (whereas the details of the Gatelock functionalities and features can be found in the Gatelock Tutorial).

Figure 4.14 Testing Gatelock with tabletop and pen interaction.

a) Drawing

In the studies conducted earlier, it was observed that the children love to draw and like their drawings to be included in their games instead of using the predefined images in the software library. For this purpose, the drawing function was included in Gatelock and the children can show and save their creative drawings. The drawing area has many nice options. For example, the children can draw by using different pens, choose their favorite colors and much more, as shown in Figure 4.15. The features have colorful icons that enable the children to understand the graphical actions they are representing. This includes typical drawing commands such as lines, refill, erase, pick color and so on.

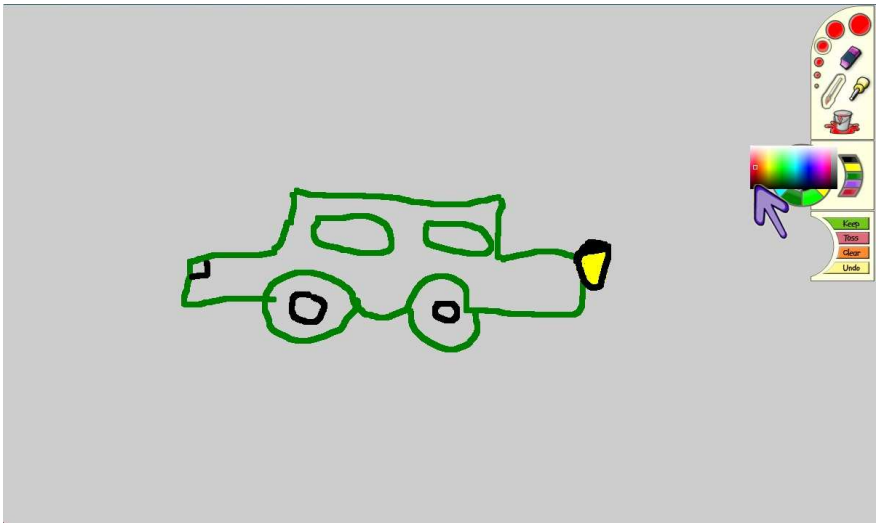


Figure 4.15 Drawing features.

b) Adding Game Elements

Gatelock will provide the children with a question mark, which indicates that they can freely select their preferred game objects and the background. Some of the examples of game objects and the background can be found in sub-folders and each folder classified different types and images. Figure 4.16 shows an example of a game object to create a new spaceship. An *Add* button will appear in the view after the children selected an intended object from

various existing objects listed in categorized folders. The *Add* button uses a *question mark* to indicate where the children need to act and select the intended images. Henceforward, the children can choose between various options that are required to create game characters and their behavior.

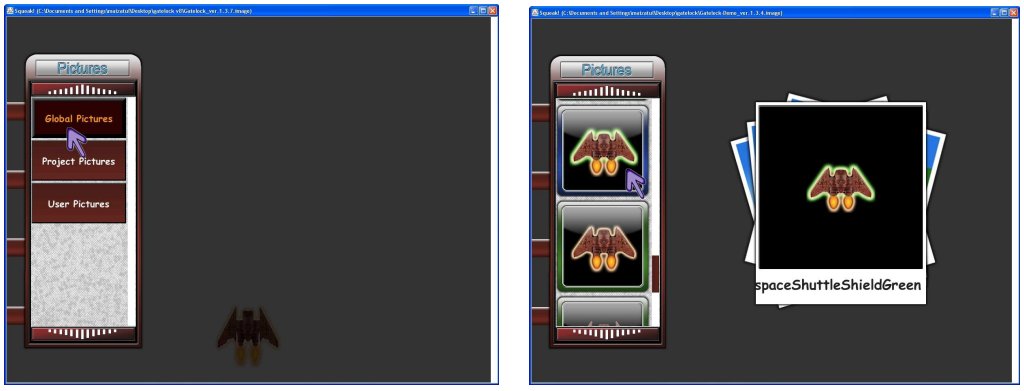


Figure 4.16 Selecting and choosing a game object. (*Left*) Selection of existing pictures organized by using created folders. (*Right*) When the children choose a certain picture, it will be shown in a larger preview.

c) Changing characteristics or functions of game elements

All elements in the folders have a mouse-over image that gives a direct visual feedback to the children. The scroll areas run across the entire width of the dialog to facilitate the scrolling process. For every object, there are different object controlling functions or characteristics (known as halo) distinguished by using different colors. When the object appears, it carries seven halos, an ID and an arrow. Each halo has a specific function. The description of thirteen halos for an object including an ID and an arrow that allow children to control the object's functions and characteristics can be found in the Gatelock Tutorial on the attached CD.

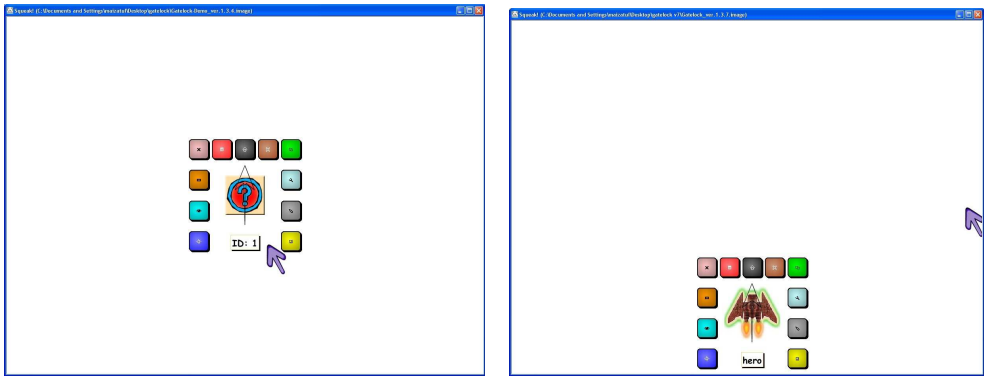


Figure 4.17 Halos. (*Left*) Through the *Add* button with question mark sign the children can choose their next action for the selected object. (*Right*) If the children have chosen the right object, it can be put at the intended location or it can be resized.

These halos are the rapid adjustment of properties that are appropriate for a mouse-based interaction such as size, appearance, rotation and position that also appear in the Object Editor windows. Several objects can be duplicated by clicking on the selected object and choosing the *Duplicate* halo on the selected object (see Figure 4.17). Duplication means that the duplicated object will inherit all properties and characteristics of the main object. Since Gatelock uses Smalltalk and Squeak paradigms of object-oriented programming, the duplication is no problem. For example, a new object can be added by moving the cursor to the left side of the screen. Now, the dialog for the selection of new elements appears and the children can choose a new object from the list of objects.

d) Changing the background scene

In each game level, game backgrounds are essential to support the objects and to give them a context. The children can select existing backgrounds from the Gatelock library or design own backgrounds by using the drawing tool of Gatelock or any other usable drawing applications. Most game backgrounds use a bitmap image to represent the backdrop of a game. The children need to choose the appropriate game background for the game they want to design, as shown in Figure 4.18 (Left). If the children have more than one game level and want to have different game backgrounds for each level, the *Option Editor* is needed. In *Option Editor*, the children can either select different game backgrounds for different game levels or all game levels will have the same game background. Figure 4.18 (Right) shows the selection of one game background for all existing game levels. The features allow the children to choose their own game background for each game level and they can also draw their own game background using the drawing features provided in the game authoring tool.

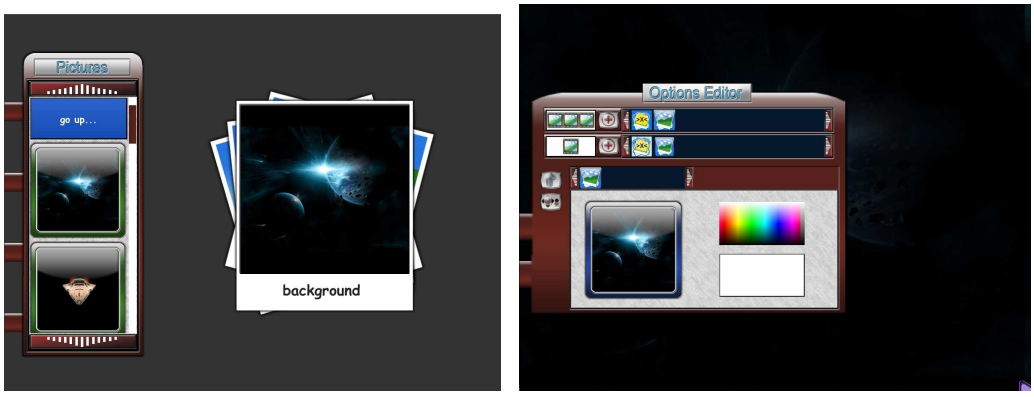


Figure 4.18 Selecting and choosing the background. (*Left*) List of backgrounds to be selected from the predefined library, including the preview area. (*Right*) To design different backgrounds for different game levels, children can use the Option Editor function.

e) Adding sound

Gatelock possesses a list of predefined sound. Nevertheless, the children can create their own sound by recording them using a microphone or an audio CD. Together with available sound or audio recorder applications, children can input their own choice of sound to the game project. The selection of sound can be seen in Figure 4.19. Adding sound to the game

level is the same as adding a new game object. Upon clicking on the sound icon, the children get a list of available sound tracks to be chosen from, and all predefined sounds are playable. Before the decision-making, the children can also listen to the sound tracks.

Figure 4.19 Predefined sound. (*Left*) List of predefined sound tracks that can be selected by children. (*Right*) List of playable sound examples facilitating the selection process.

f) Managing the game levels

The management of game levels is involved with three different functions: adding a new game level, switching between game levels, and deleting a game level (see Figure 4.20). Firstly, the A*dd Game Level* menu allows adding a new game level. The second menu item is *Switch Game Level*. This allows exchanging two game levels. The third menu item is *Delete Game Level* which allows deleting any existing game level. Thus, the children can freely manage their own game levels.

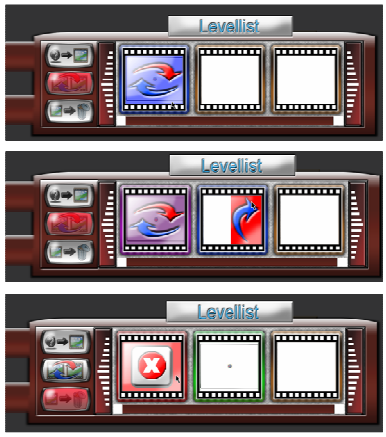


Figure 4.20 Managing the game levels. (*Upper*) Add Game Level. (*Middle*) Switch Game Level. (*Below*) Delete Game Level.

g) Managing the game options

Managing game options, such as collision detection, can be done with the Option Editor which is symbolized by entangled gears. The Option Editor function has multiple features, as shown in Figure 4.21 (Left). The dialog represents the fundamental characteristics of the objects, including the setting of one background for all game levels or a different

background for each level. The children can also identify the best background by either choosing it from the predefined background library or by creating a personal background. The *Options Editor* is divided into two different features: *Global Options* and *Local Options*. Both option features have a separate *Add* button to add elements accordingly. Furthermore, they are simultaneously displayed to the children with an overview of currently used objects in the project. Each project receives a standard element with the ability to freely adjust the game background scene of the levels.

Figure 4.21 Option Editor windows. (*Left*) The *Option Editor* shows the conditions *before* defining a collision detection function. (Right) The *Option Editor* shows the conditions *after* setting two defined collisions.

For example, to set collision detection features, they should be defined in both the *Global* and the *Local Option*, as shown in Figure 4.21 (Right). The figure shows two fully defined collisions. Four collision types can be chosen: 1) a collision between two objects, 2) a collision between groups of objects, 3) a collision between one object, as well as 4) a collision with one or more border. As shown in the example in Figure 4.22, the collision of two objects was defined by selecting two objects that will be involved in a collision. Then, the selected objects will be declared and described based on the intended actions. In this case, several actions can be defined using the *Option Editor*.

Figure 4.22 Collision detection. (*Left*) Defining a collision between two objects. In this example, an event occurs when the object Enemy Missile knocks against the object Hero Missile. Setting the event requires a scripting procedure.

h) Managing Undo and Redo

The last entries in the main menu serve to undo and restore operations, as shown in Figure 4.23. Children can go to a previous action by clicking the white arrow. If no changes were made or no action is reversible, or recoverable, this procedure can be done by clicking the

redo arrow. As described in the study, the *Undo* and *Redo* operations are important (see Chapter 3 for further details).

Figure 4.23 Undo and redo function.

i) Managing the help system

The help dialog system uses two methods. The first is the tooltip method which is similar to the tooltip in any Microsoft application. The tooltip defines the name of the selected buttons or icons. The second method uses more detailed description dialogs for the selected buttons or icons. When the children select a button or icon and wait for a few seconds, a brief explanation of the button appears. For further investigation, the appropriate length of waiting time until the help system dialog appears will be described in Chapter 5. Figure 4.24 shows the difference between the two methods used in the help system. The help system aims to provide the children with an overview of the functionality for each button or icon. The used texts or words are meaningfully selected and appropriate to the age and language skills of the children. When the cursor is moved away from the button or icon, the help system is hidden from the children's view.

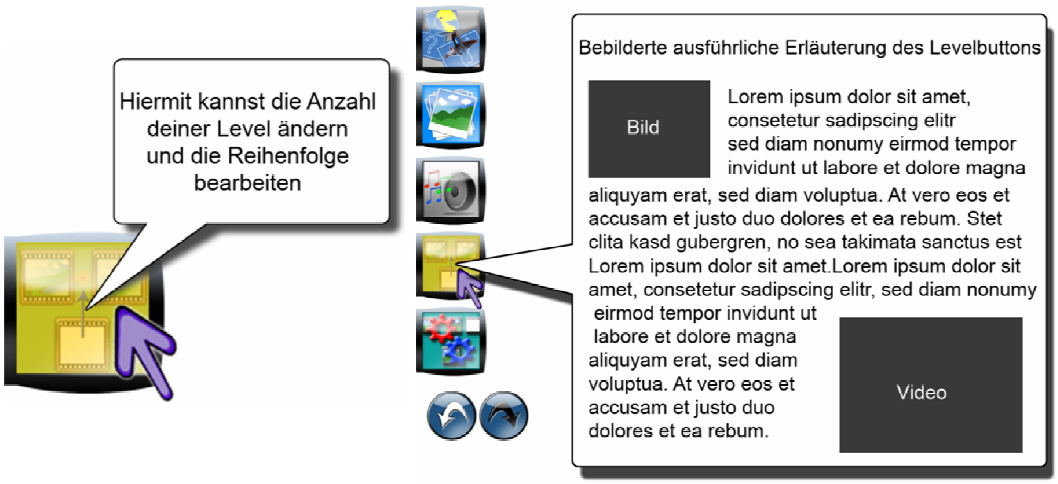


Figure 4.24 Help system. (*Left*) Tooltip method. (*Right*) A description dialog used in the help system.

4.5 Child-Level Description of Gatelock

This section will describe how a game is created using Gatelock. The game design activity will have an ideal flow structure for children to be used in designing games. It starts with the game plan. The children need to think carefully about the game objects and all elements

related to the design of the game project. They can create pictures related to the game. This can be done by using the drawing tool provided in Gatelock, or by using any available drawing applications. All pictures should be saved in **.png* format in one folder for further actions. The pictures can either be game objects or one or more backgrounds for the game levels. The planning phase also involves additional materials that are going to be used in the game project, such as sounds or animations.

4.5.1 Game Design by a 10-Year-Old Boy

A boy aged 10 was asked to design a game. He decided to design a one level game called *SpaceShuttle*, which is similar to the game from the 80s known as *Space Invaders*. The boy planned his game as shown in Figure 4.25 (Left) and used the given Gatelock Tutorial (see the attached CD). The following section describes each step of the game design workflow, as shown in Figure 4.25 (Right).

A hero ship will be controlled by a child (gamer) through a two-dimensional static level while five enemy ships will constantly fly from the top to the bottom of the screen. On top of that, the gamer must try to avoid the enemy ships as the ships disappear to the bottom of the screen. The gamer can also shoot the enemy ships using the spacebar to fire a bullet or missiles. If the missiles hit the enemy ship, the hero ship will get stronger. But if the hero ship collides with an enemy ship, this will eventually end the game.
– A 10-year-old boy –

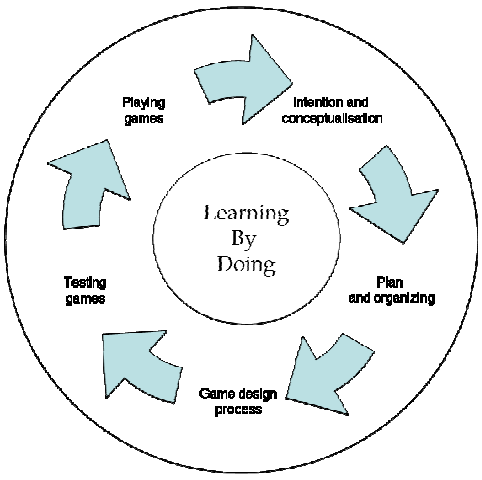


Figure 4.25 Game design workflow used by a child. (*Left*) Description of the boy's game design plan. (*Right*) A game design workflow mediated according to the constructivist learning approach.

Intention and conceptualization: The boy described the game that he wanted to design in writing. He identified the characters and elements involved in the game and discussed them with his teacher and with other children.

Planning and organizing: The boy carefully planned the game design. During this process, his teacher acted as facilitator to guide and assist the game design process. The discussion between the boy and his teacher was more focused on the needs of the boy on learning how to solve game design related problems with a slight introduction of a new concept of designing games by the teacher. Now, the boy had a rough idea about the game he wanted to design, including the drawing and finding of appropriate images for the game characters, a game background, a storyline, and so on.

Game design process: After designing all necessary pictures, the boy set the game background by selecting one of the backgrounds listed in the specified folder. Afterwards, he

located the suitable game character images or objects to their exact location and position in the development area (also known as the canvas). Since these images need to be declared, the *Option Editor* is involved for specifying the characteristics of each relevant object. The second declaration involves the movement of the objects. The editor decides how the interaction of the game will look like. In addition, whenever the enemy fires the ship with missiles (declared in a given time interval) the child should move the ship to a safe place. When the ship of the enemy collides with the ship of the child, its second shield will be reduced to one shield. When no shields are left, the game level will be reset. Since this is only a demonstration, there is no game goal. New enemy spaceships are generated and the player always restarts. The overall flow of designing the game can be seen in Figure 4.26.

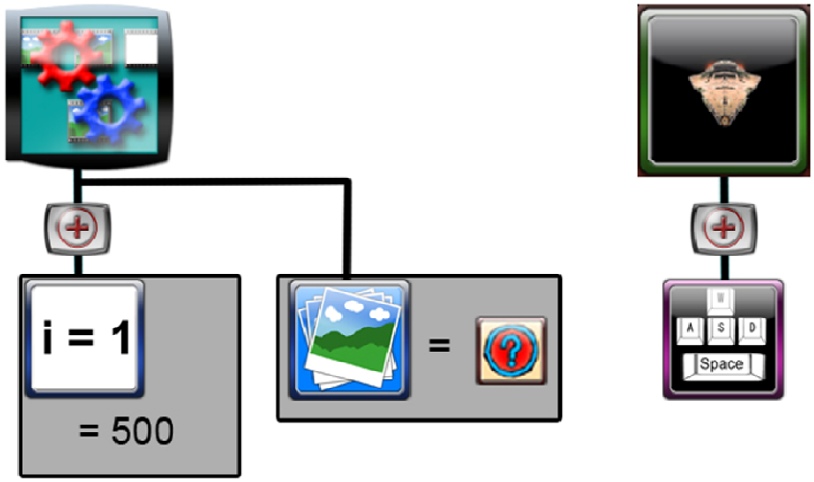


Figure 4.26 Configuration of a game object starting with adding the object up to defining its characteristics and behavior.

Testing the game The boy repeated testing and designing the game until he was really satisfied with the results. In fact, the testing process also involved the teacher who was asked about the work. Here, it could be observed that the availability of the play-test features really helped the boy to understand the relationship between the design process and the arising outcomes. The teacher was impressed on how the boy criticized his own work and redesigned the game in order to achieve his goal.

Playing the game When the boy was satisfied with his work, he called his friends to play his game and they accepted the challenge without hesitation. It could be observed that the boy was proud about his work and accepted his friends' opinions and comments. The children collaborated with each other in order to fulfill their game design objectives. This cycle continued with other children and other game design projects. Some examples of game design projects can be seen in Figure 4.27.

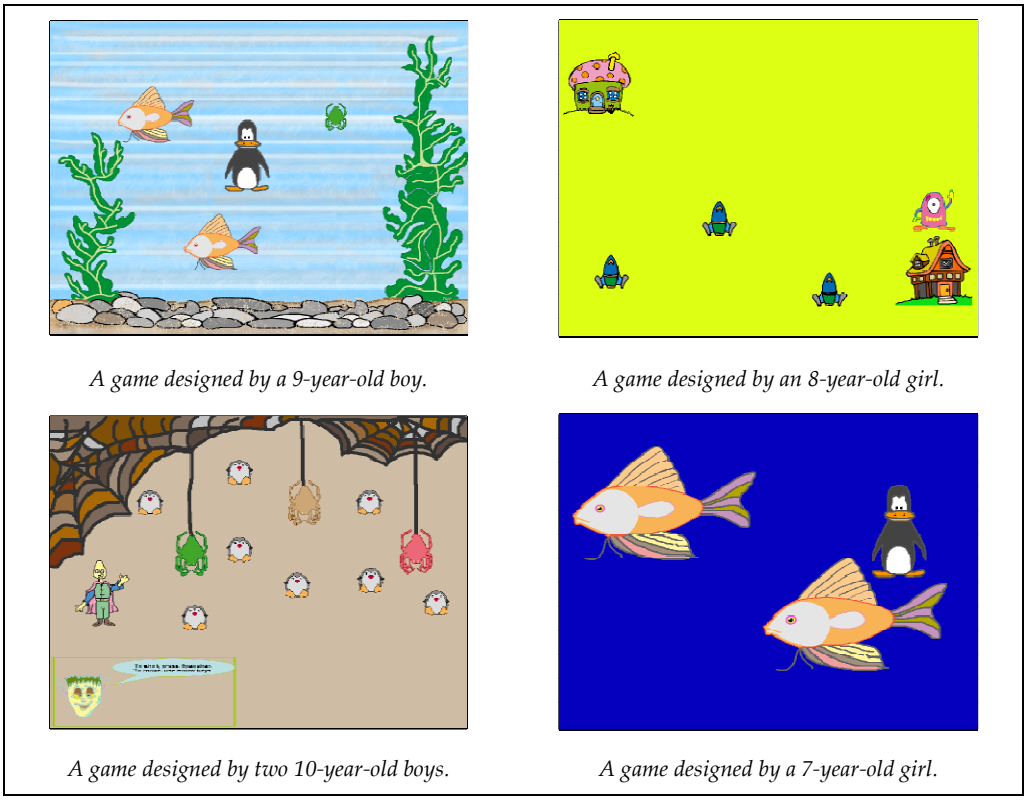


Figure 4.27 Screenshots of games designed by children.

4.6 Summary

The implementation of the design of the game authoring tool was carried out smoothly under consideration of several steps. The first attempt was to test the game engine itself. Afterwards, the preliminary study was taken into consideration, especially the aspects of children's requirements, game design requirements, and the elements of learning for children. In this chapter, the implementation of the design approach for designing a game authoring tool for children between seven to twelve years was described. The chapter described the implementation of the game authoring tool called Gatelock in detail. The implementation was based on the prototyping methodology with the integration of three main development tools called Squeak, Tweak and iEngine, including an explanation of the game authoring tool interface and interaction design. The chapter ended with an example of a simple game designed by a group of children. Since this is an early stage of development, a usability test is required. In addition, it is assumed that there will be many areas lacking of functional elements. Thus, the first usability evaluation of Gatelock and understanding its usage will be conducted. The children will be enabled to design a simple game by using the working features of the main menu screen and the development screen, and they can also design a game with one or more playable game levels. These items will be described in the next chapter.

5
Usability Studies of Gatelock with Children

The previous chapter reported on the implementation of the game authoring tool called Gatelock. The implementation was based on the prototyping methodology with the integration of the three main development tools Squeak, Tweak and iEngine. The chapter ended with an example of a simple game designed by children. As mentioned, since this is an early stage of development, a need to conduct a usability test was urged. In this chapter, two usability studies will be discussed beginning with the formative evaluation. Prior to this study, the process of redesigning Gatelock will be described. Afterwards, the summative evaluation will be discussed with different groups of participants (children). Lastly, the findings from these evaluation studies will be discussed in detail at the end of this chapter.

5.1 Pilot Study for the Formative Evaluation

Before starting the formative evaluation at a school in Malaysia, a pilot study was conducted with five children between nine and twelve years. The aim was to measure the instruments that are going to be used in the real formative evaluation, including the language used, the answer scales given and the questions asked. All children participating in this study possessed basic computer knowledge. Hence, they were familiar with computer terms and had basic experiences in using standard application software and playing computer games. In addition, the formative evaluation was conducted in the selected school. After finishing the formative evaluation, Gatelock was redesigned and the summative evaluation study was conducted between August and November 2008.

5.1.1 Participants

The five participating children already had basic computer knowledge. Merely five children were selected since this was only a pilot study to improve the design of the instruments that are going to be used in the real study. The sample covered the normal range of ability including basic computer literacy. All children had English as second language and did not have any language problems, since the tool provided visual understanding and most of them received guidance from the instructor.

5.1.2 Procedures

The study was conducted in the presence of a researcher. Gatelock was installed on one notebook and each child had two hours of time to use Gatelock assisted by the researcher. Each child was asked to design a game. This included a game plan, using Gatelock to design it, trouble shooting, decision-making based on numerous possible solutions, and testing. A specific time limit or perfect outcomes were not defined, but the children were asked to solve given tasks. The researcher was accompanied by an assistant, who noted the children's reactions and engagement with the tasks. At the end of the study, a set of usability questions was distributed to be answered by the children.

5.1.3 Instruments and Data Collection

Questions regarding the three main usability aspects—effectiveness, efficiency and satisfaction—were posed. Some of the questions were structured into the organization of information, highlighting the easy way to perform tasks, understanding the information on the screen, sequences of the screen, messages on the screen, easy understanding of the scripts and structuring of visual information. To measure the fun aspects, questions concerning creativity work, styles of artwork, pleasant surprises and enjoyment were asked. To get answers to these aspects, the questionnaire method was used, which is established for data collection since a long time. The questionnaire method was also reported to be used with children [Bogdan98]. Furthermore, upon following a question-answer procedure, normally the Likert scale is selected by conventional researchers. Others also use visual analogue scales to ask questions to children, such as Wong-Baker's Pain Rating Scale [Wong98] and the Smileyometer Scale [Read06]. Both methods use different smiley faces, since it is believed that younger children have difficulties in using traditional measurement scales with "*a number, a ranking concept or unfamiliar words*" [Airey02]. For the usability and fun questions in this study, the Smileyometer method was adapted [Sieber07], replacing the traditional discrete Likert-type scale. The Smileyometer has been used for different research before and is said to be one of the most appropriate indicators to be used when the testers are children [Read06]. The Wong-Baker Pain Rating Scale was not selected due to two reasons. First, it is aimed for pain assessment [Airey02], and second, the smiley faces scale is in a totally different direction. As most traditional scales increase from left to right, the Wong-Baker uses the opposite way. The children received the smiley emoticons (emotion icons) in form of stickers and were asked to stick or paste them at the appropriate scale that shows their true emotion and feeling about the tool. This method was revised from the Fun Sorter tool [Read06] which allows children to rank items against one or more items, and was intended to record the children's opinions about the game authoring tool. Furthermore, the Smileyometer and the Fun Sorter tool can be used to measure the children's engagement (see Figure 5.1).

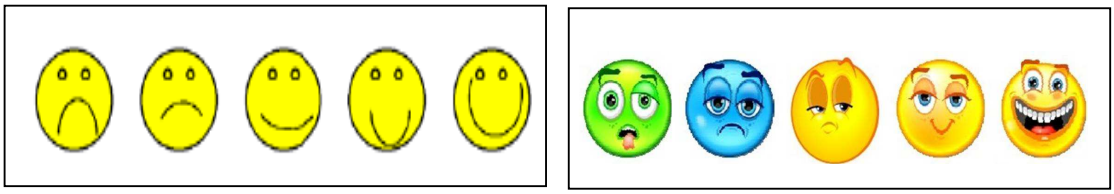


Figure 5.1 Smileyometer. (*Left*) A Smileyometer [Read06]. (*Right*) The adapted Smileyometer used in this study. Both approaches indicate the children's emotions in using the game authoring tool. From left to the right (for each version) it means awful, not good, good, very good, and brilliant.

One researcher was involved in conducting all five usability test sessions, while one assistant observed and noted the children's reactions and engagement with the tasks. The assistant used a checklist to simplify his tasks. The checklist consisted of behavior and acting reactions during the use of Gatelock, including affective reactions such as gestures and facial expressions, laughing or frustration, expression of pride, verbal communication, self-exploring and self-discovering (refer to Appendix B for the Observation Checklist). The behavioral indicators were marked and analyzed for any frequent actions occurred and notified according to a given time scale.

5.1.4 Results

The usability and fun instruments were analyzed qualitatively and compared to the observation data. Since the number of samples is small, generalizations are not that appropriate. The test involved only five participants; therefore, the data analyses are limited to descriptive statistical analyses. Due to the limited number of children participating in the pilot study, qualitative results are very important. Above all, Table 5.1 shows the personal data from the participating children to help the researchers understand the cause for an observed problem, and to make the analysis more reliable to provide heuristics or guidelines.

Table 5.1 shows the children's personal data (B=boy and G=girl).

Gender	Age	Having own computer	Hours per day playing games	Most favorite game genre	Person influencing the playing of games	Features in games you like most	Interest to develop own game
G1	10	Yes	< 2	strategy	brother, father	challenge	Yes
B2	9	Yes	> 3	action, sport	friends	challenge, graphic	Yes
G3	12	Yes	< 2	puzzle, strategy	friends	story, graphic	Yes
B4	11	Yes	2 – 3	simulation, racing	uncle	challenge	Yes
B5	9	Yes	2 – 3	action, racing	brother	challenge, music	Yes

As a result, computer games are more embedded into the leisure culture of boys than girls. Boys are more in favor of games that are associated with their other interests such as sports. They spend more than two hours per day playing computer games and they like to play challenging games. Regarding the question about the children's leisure activities, boys considered game playing as first choice activity, whereas girls were more likely only to play games when they were bored or had nothing else to do. When the children were asked to identify the best features of their favorite games, they stated features such as challenge, story, graphics and music.

The findings also revealed several problems of the instruments used. Some problems resulted from the usage of computer jargon within the instructions and questions, the need to reorganize the structure and procedure in answering the questionnaire to make it more fun to use, the need to reduce the number of questions and to select those which are really connected with the study, and the elimination of redundant questions (if any). Since only five children were involved in the study, it was much easier to implement the think-aloud method to evaluate the tool with the children. They were asked to think aloud while performing some given tasks. Through this activity, the children gained and developed a capacity to verbalize their thoughts and to make suggestions upon using the tool or the instrument given for data collection. Figure 5.2 shows the original instruments used in the pilot study, including the Gatelock tutorial and the questionnaire (refer to Appendix B for details).

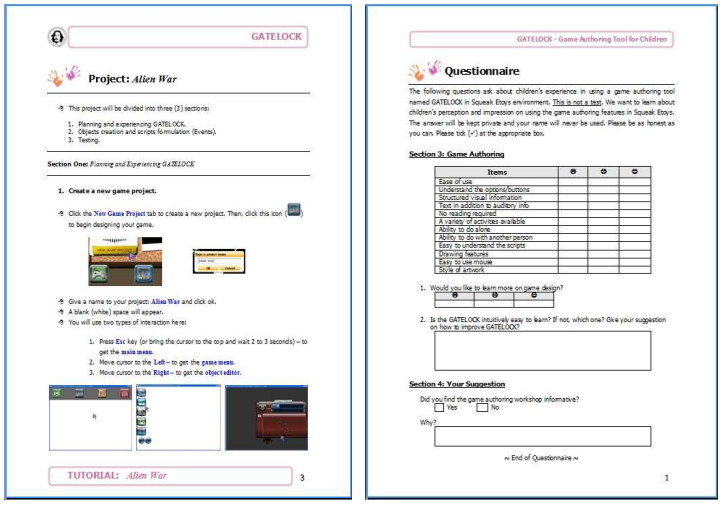


Figure 5.2 Instruments used in the study. (*Left*) Gatelock Tutorial. (*Right*) The first impression of the questionnaire used in the pilot study for formative evaluation (refer to Appendix B for details).

Furthermore, the children were asked questions concerning their interest level in game making activities in the classroom, and especially about what they have learned from Gatelock. It is common to use an interview method for children where they are asked to contribute their ideas and suggestions for partially completed or future designs. Below are some quotes from the children at the end of the test:

I wanted to know how a game can be developed. Before this, I only knew how to play...Girl1
I have learned something new and I feel good about it. And it is fun...Boy2
I wanted to develop my own games and asked my brother to play them...Girl3
If I know how to make a game, I do not have to spend my money buying one...Boy4
I think we need more games that relate to learning so that we will sit in front of the computer all day in school...Boy5

All children were keen to take part and seemed to enjoy the experience of designing games with the given tool and to be involved in the test session. The children were also asked about the appropriateness of visual representations and icons used in the tool. For this purpose, a set of selected icons was given to the children who had to identify the meaning of the icons within their own visual perception (refer to Appendix B for details). The children received a mix-and-match questionnaire on icons as well as their meaning. All children managed to answer 89% of the questionnaire accurately [Yatim09e]. The overall results indicated that all icons except one were found appropriate to be used in the software. It was observed that the children's level of experience plays an important role in perceiving the representation and meaning of the icons.

5.2 Conducting the Usability Study: The Formative Evaluation

The formative evaluation was carried out to spot usability issues in the early phases of a design project. The evaluation provides a possibility to test Gatelock's design ideas and ways to implement changes before completing the entire working software. It was also carried out to test the interface and interaction design and the content delivery of the game authoring tool. The evaluation was intended to allow the designers to assess Gatelock during its development, and to test an early version of the tool with selected children to identify weaknesses in areas such as functionality and instructional effectiveness.

5.2.1 Participants

The sample consisted of 46 children (30 boys and 16 girls) between eight and twelve years from a primary school in Malaysia. The specific goal of this study was to detect as many usability problems as possible and to improve the interaction design and user interface of Gatelock. All children had English as second language and they did not have any language problems since the tool provided a visual understanding and most of them received guidance from the instructor. All students participating in this study possessed basic computer skills and thus fulfilled the requirements. The students were familiar with the computer terms and had basic experience in using standard application software and playing computer games. Figure 5.3 shows groups of children participated in the game design activities.

Figure 5.3 Groups of children involved in the game design activities. At the end of the study, the children were asked to answer a questionnaire.

5.2.2 Procedures

The study was conducted in the presence of researchers. Gatelock was installed on all computers in the computer lab. The children were divided into a small group of three, whereas each group had two hours of time to use Gatelock assisted by the researchers. Each group was asked to design a game using Gatelock, including trouble shooting and decision-making based on numerous possible solutions, and testing the game continuously. The children were informed that this activity does not require any perfect outcomes, but that the given tasks should be solved. An assistant noted the children's reactions and engagement with the given tasks. All children were keen to take part and seemed to enjoy the experience. At the end of the study, a set of usability and fun questions was distributed and had to be answered by the children.

5.2.3 Instruments and Data Collection

For the evaluation protocol, guidelines for usability testing with children were adapted especially on greeting, stressing the importance of the participants, explaining the purpose of the study, and making sure they know that they are not treated as test objects [Hanna97]. As mentioned above, a set of usability and fun questions was handed out to the children. The questionnaire was a mixture of several usability methods, since evaluations on the usability of existing game authoring tools such as Alice, Stagecast Creator and Game Maker are still unclear, as described in the introduction section. The questions were formulated regarding the three main usability aspects effectiveness, efficiency and satisfaction. Some of the questions were structured into the organization of information, highlighting the easy way to perform tasks, understanding the information on the screen, sequences of the screen, messages on the screen, easy understanding of the scripts and structures of visual information. Again, to measure the fun aspect, questions concerning creativity work, styles of artwork, pleasant surprises and enjoyment were asked. Furthermore, for both instruments, the usability and fun questionnaire, the Smileyometer method was adapted replacing the traditional discrete Likert-type scale.

Regarding the usability questions, the children were asked to rank their answers. The answers needed to be picked by selecting and scrolling to the appropriate answer (*Select-and-Scroll*). One child received one question at a time and answered it by pushing the red indicator to the emoticon that showed his or her feelings towards the used tool. Afterwards, another child answered the same questions and the procedure was repeated with other children and other questionnaire sheets. Regarding the fun questions, the children received Smileyometer emoticons (emotion icons) in form of stickers and had to stick or paste them at the appropriate scale that shows their true emotions and feelings about the tool (*Pick-and-Paste*). This method was revised from the fun sorter that allows children to rank one item against one or more items and to record the children's opinions, as shown in Figure 5.4. Both procedures showed the fun side of the evaluation process and the children were observed to enjoy participating in the test.

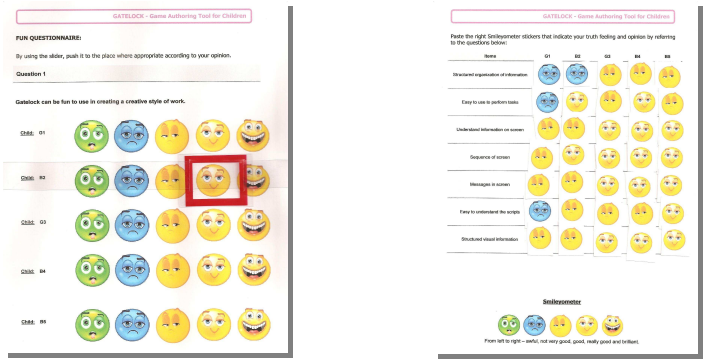


Figure 5.4 Instruments used in the study. (*Left*) Snapshot of the satisfaction ranking called *Select-and-Scroll* for one question in the questionnaire. (*Right*) A fragment of a table showing one question in the fun questionnaire called *Pick-and-Paste* (refer to Appendix B for details).

One researcher conducted all usability test sessions while another research assistant observed and noted the children's reactions and engagement with the software. In order to perform the qualitative approach, the assistant used a checklist (Appendix B) to simplify this

task. The checklist consists of behavior and acting reactions during the use of Gatelock, including affective reactions such as gestures and facial expressions, laughing or frustration, expression of pride, verbal communication, self-exploring and self-discovering. The behavioral indicators were marked and analyzed for any frequent actions occurred and notified according to a given time scale (refer to Appendix B for Observation Checklist). The usability and fun questions in the questionnaire were analyzed and compared to the observation data. The findings of the study were divided into the four categories of effectiveness, efficiency, satisfaction (the first three categories are related to usability aspects) and fun.

a) To measure *effectiveness*, the number of errors made by children when working through the given tasks was counted. The errors were treated as problems identified during the software testing with children. The findings highlighted that 69% of the participating children made more than 10 errors while using the tool. 20% of the total errors were repetitive errors. A total of 58 errors were found in the formation study. The errors made by the children were identified and it was concluded that these errors involve two parts: (1) a wrong or different interpretation of the iconic interface used, and (2) common technical programming problems in the tool itself (software bugs).

b) To measure *efficiency*, two indicators were used: the learning time and the speed of performance. Another way to measure efficiency is by looking at the retention over time which requires the same group of children to participate in tests more than once. However, since the study had a time restriction because it was conducted in a school environment, the measurement of retention over time has to be kept for future work.

    i. To measure the *learning time*, first it has to be determined when the software was learned. This includes the children's ability to perform certain tasks. To achieve this purpose, only children who have never used the software before were selected. They received a list of 15 tasks and the time they spent to learn the tasks, was recorded. The recording of learning time was carried out by one researcher. The findings revealed that 67% of the children spent more than 5 minutes to learn and understand how the tool works, including its functions and interaction features. The remaining children spent less than 5 minutes for the whole tasks to grasp the meaning of the functions and interaction features to settle the given tasks. It was also found that children who spent more than 5 minutes to finish one task at a time spent less than 1 hour per day with the computer and made most of the errors.

    ii. Measuring the *speed of performance* was carried out by specifying a list of benchmark tasks and asking the children to perform these tasks using the tool. This activity was done after a 15 minutes recess. First of all, the time was set to a maximum of 10 minutes to find out how many tasks can be completed by then. Again, the same 15 tasks were used and the children's speed of performance in finishing all tasks at once was recorded. Nearly half of the children needed less than 5 minutes to complete the tasks and some of them managed to repeat the same tasks while others were still working on the first cycle of tasks.

c) *Satisfaction* is best measured by asking the children if they like the tool or not. For this purpose, the satisfaction ranking questionnaire (*Select-and-Scroll*) was used to find out

what the children think, as shown in Figure 5.4 (Left). All children agreed that they are satisfied with the tool; 70% chose *most likely* while the rest chose *likely*. None of them reported any dislike about the tool, even if they faced several errors and problems with the software.

d) *Fun* is also best measured by asking the children. This may rely on self-report responses, a technique not commonly associated with children. But it is believed that the children told the truth and were honest when giving answers or opinions. For this purpose, a special tool called *Pick-and-Paste* was developed, as shown in Figure 5.4 (Right). The majority of children agreed that the tool is fun to use. However, some children stated that the tool is not fun to use due to the errors they had to face and deal with before fulfilling the given tasks. Another reason was the technical problems of the tool itself, since 1/3 of the computers were not operating properly, i.e. they staggered and had a slow performance. The problems arose due to the limitation of the hardware and were considered to slightly discourage the children to use the tool. A further problem occurred during the installation of Gatelock a day before the actual evaluation. According to the teacher responsible for administration of the computer lab at school, *"…all computers in the lab fulfill the minimum hardware requirements for operating, but some of them need to be repaired and upgraded due to the installation of new software for the school."*

5.2.4 Results

The possibilities to integrate any subject lesson into game making activities are endless, especially when it comes to the idea of using a program such as Gatelock in the classroom. Creativity is not just a question of creating new solutions, but creating better solutions and this requires a critical judgment. By understanding this, learning while making games can connect motivation, creative thinking and learning in any lessons, including sciences and mathematics.

a) *Finding 1: Usability problems can be detected by measuring the effectiveness, efficiency and satisfaction of the game authoring tool as early as possible*. The usability study for evaluating the game authoring tool revealed problems regarding its interaction and interactivity aspects. The problems were defined by considering and measuring the effectiveness, efficiency and the children's satisfaction with the tool, as discussed in the previous section.

b) *Finding 2:* The f*un aspect is a must in designing a game authoring tool for children*. Fun in the positive sense can include real exertion when designing games. Fun plays an important role with regard to intrinsic motivation and engagement in learning how to design and play a game. Therefore, it is crucial for developers of any game authoring tools or construction tools to include and further measure the fun aspects of their tools.

c) *Finding 3: Evaluating children's software with children offers positive impacts for designing the game authoring tool*. The children were committed to state their opinions and criticisms about the used tool. These aspects were considered for perfecting and redesigning the tool for the next version. The children knew how to comment on the tool regarding interface aspects, interaction, and understanding of functions and features of the tool, and they made several new suggestions.

The overall results of the study were encouraging. A significant amount of usability aspects in the tool was expected, but also pleasantly surprising was the low number of limitations and errors in the tool. It is understood that the tool is still in its development phase and may provide some defects that will eventually be treated as important information to redesign the tool. In general, all children found the designed game authoring tool easy to use with a little instruction. By running the test, some key problems were identified at an early stage and various usability issues could be fixed prior to the actual deployment of the game authoring tool. There were several high-ranking usability problems that needed to be solved:

- 48 out of 64 buttons did not work constantly.
- The children did not pay attention to navigation buttons, thus the buttons need to be more noticeable.
- The children were confused by different operation modes (object-action-object-model).
- The children frequently clicked on objects and expected something to happen.
- A more simple language that is easier to understand by the children, iconic layout and interface design for a better understanding are required.
- 73% of the participants think that labels placed on the left side are user-friendly for understanding their functionality.
- 88% of the participants find icons helpful and useful.
- 26% of the participants prefer a horizontal navigation bar compared to a vertical navigation bar.

As mentioned before, these problems will become the guidelines to redesign Gatelock. Then, further tests will be carried out with other children for the summative evaluation. As mentioned, Gatelock purposely aims to engage children with game design materials in a non-threatening and enjoyable way. The study showed that the children enjoyed the use of Gatelock and that they were able to think about and develop their own creative work. They enjoyed the game design activities. Sometimes they were frustrated because they did not get what they intended. But they were willing to share their thoughts, asked for assistance and communicated their ideas by drawing them on paper.

5.3 Redesign of Gatelock

After getting input from the formative evaluation of Gatelock, a major step was needed to redesign Gatelock. This step is based on several factors. Firstly, the formative evaluation revealed that the functionality of several features was not working. Thus, the children became frustrated using the software. Secondly, the metaphor and icon design used for Gatelock seemed to be confusing for the children and they had problems to interpret their meanings. Thirdly, some features in Gatelock should be eliminated from the software, since they have no relation to children as designers (however, these functions are important for the designers of Gatelock). The idea was to hide the features from the children's eyes but to retrieve them whenever the designers need them. Many of these problems should have been retrieved earlier, but during the work with the children unexpected ideas came up, for example, that the space for drawing is too large and that the drawing should be full-scale. During the observation, the children were eager to draw features by drawing all game characters as one drawing, instead of really understanding the concept of *object* in designing the game character. Upon asking, the children complained about the drawing space that was too wide and they thought they had to draw within the whole area. In the latest design, the drawing space was reduced to solve this problem.

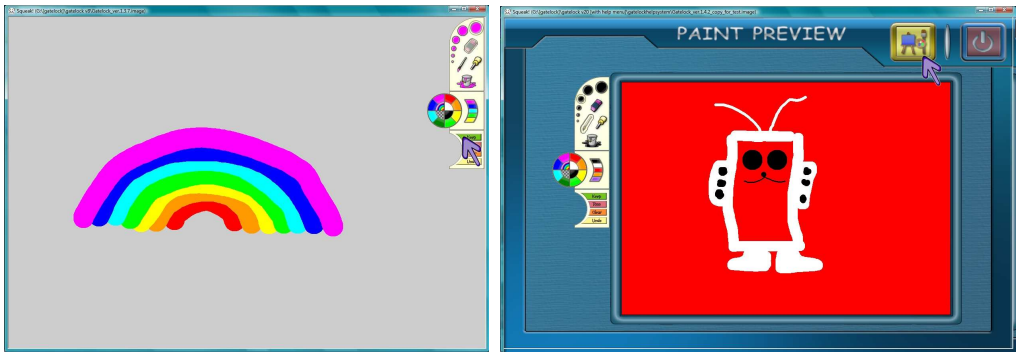


Figure 5.5 Drawing features. (*Left*) The button for saving the artifacts designed by children (inscribed with 'Keep') is not working. (*Right*) All drawing features in the tool were redesigned and the save button problem was fixed. In addition, a draw button was created for the children to draw the cover page of their game project.

Functionality of Features: The features that did not work before were fixed during the process of redesigning Gatelock. One option included the deletion of features that were suspected to cause problems. For example, the drawing feature only allowed children to draw, but not to save. The saving and some drawing features such as the *refill* function did not work before and had to be fixed during the process of redesigning Gatelock, as shown in Figure 5.5.

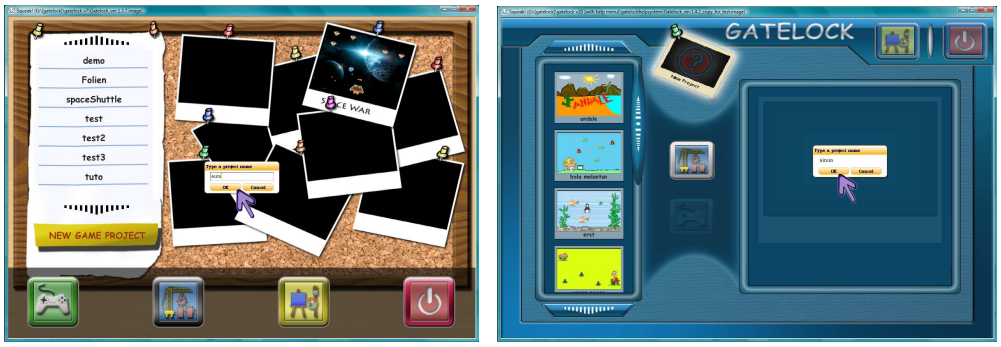


Figure 5.6 Gatelock's main menu. (*Left*) The main menu interface appearance *before* the redesign process. (*Right*) The main menu interface appearance *after* the process of redesigning.

Metaphor: The interface design of the main and development menu also faced a major redesigning process, as shown in Figure 5.6 and 5.8. The changes were implemented due to the difficulties in the interaction between the children and the tool including the misperception of certain metaphors and the icon design. The redesign process of Gatelock was aimed at a clear interface for the main and development menu. Although the earlier design was based on metaphors found in daily life, such as noticeboard and note list, it was observed that the representation of the metaphors was insufficient. Furthermore, screenshots of existing projects were found to be too disarranged and the children were confused about the design.

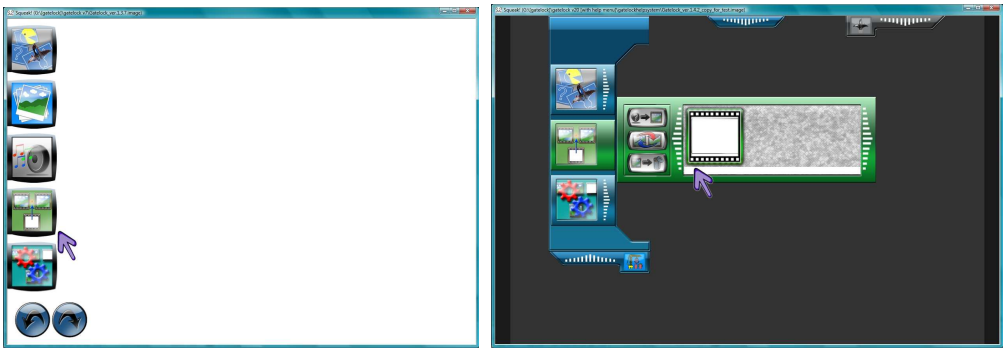


Figure 5.7 Gatelock's development menu. (*Left*) The appearance of the development menu interface *before* the redesign process. (*Right*) The appearance of the development menu interface *after* the process of redesigning.

Features to be eliminated: Some of the features were eliminated from the software, since they were not widely used. For example, the thirteen halos allowing the users to control the functions and characteristics of objects were reduced to nine halos, as shown in Figure 5.8.

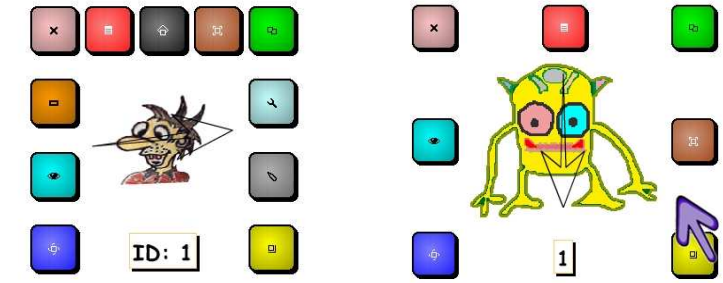


Figure 5.8 Gatelock's halos. (*Left*) Thirteen halos in the first version of Gatelock used during the formative evaluation study. (*Right*) The remaining nine halos after the redesign of Gatelock.

The game authoring tool in this study uses the mouse interaction as primary input device. During the user studies conducted, it was observed that the children did not have any difficulties in using the mouse since they had prior computer skills and experiences. Additionally, the mouse interaction devices were available during the field study which involved more than 40 participating children. For future work, which will be explained in Chapter 7, the game pad was suggested as another input device. During the design process of the game authoring tool the game pad could successfully be used with Gatelock, but only for playing purposes. To design a game that uses a game pad, virtual keyboard features need to be installed. Finally, the developers needed almost eight months to redesign Gatelock and to prepare it for the next study, the summative evaluation.

5.4 Pilot Study for the Summative Evaluation

Once again, before starting the summative evaluation, a pilot study was conducted with two children aged seven and eight. The reason was to measure the instruments, i.e. a list of tasks

and a questionnaire that were planned to be used during the real summative evaluation. Both children possessed quite impressive computer skills, i.e. they were familiar with computer terms and had basic experiences in using standard application software and playing computer games. Figure 5.9 shows the children participating in this study.

5.4.1 Participants

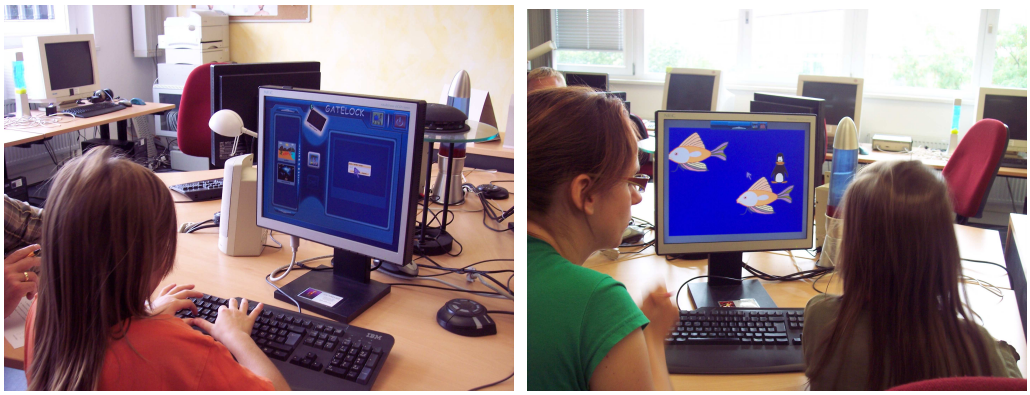


Figure 5.9 Children involved in the pilot study.

The two participating children already possessed basic computer skills. Their first language was German, but they were also familiar with English, since they learned the language in school and also practiced it at home. The eight year old boy and the seven year old girl were very excited to take part in the pilot study and they were eager to use the tool.

5.4.2 Procedure

The study was conducted in the presence of a researcher and two assistants who acted as facilitators during the study. This means, they supported the two children during the use of Gatelock and answered questions at the end of the study. Gatelock was installed on two PCs and each child had one hour to use it. Each child received a list with 15 tasks related to the game design activities (as shown in Figure 5.1) that had to be finished. A certain time limit or perfect outcomes were not specified, but the children were asked to solve as many tasks as possible.

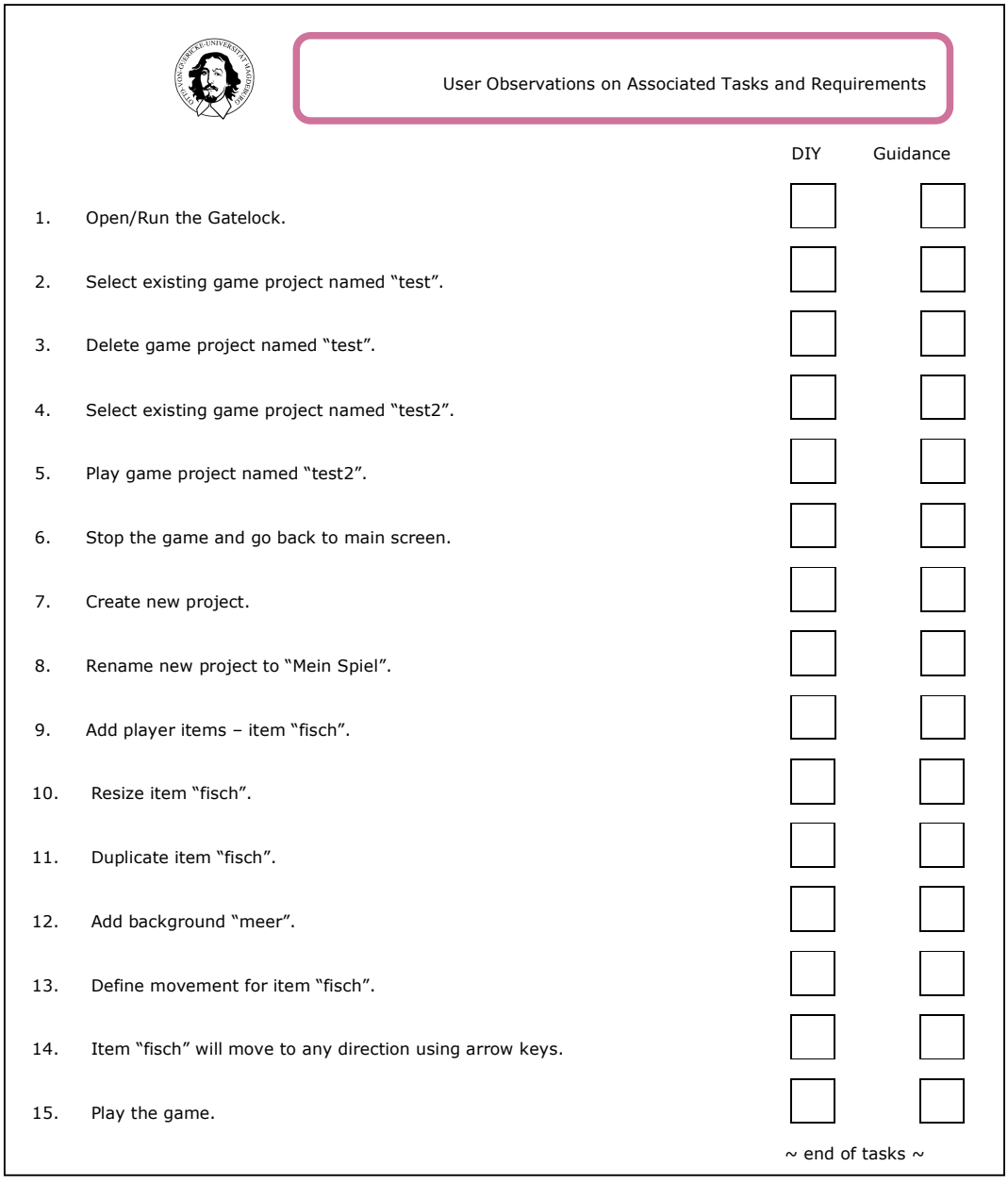


Figure 5.10 Snapshot of a list of tasks given to the children for the pilot study.

5.4.3 Instruments and Data Collection

As mentioned above, smiley emoticons were used as answer scales for the questionnaire. However, for the summative study, the implementation was different. Based on the inputs from the formative evaluation carried out before, the answer scales for the questionnaire were reduced from five to three. Another example is the procedure to answer the given questions. For the formative evaluation, two types of instruments were used: *Select-and-Scroll* and *Pick-and-Paste*, as described in Section 5.2.3. At the end of the study, the children received a questionnaire called *Scroll-Me.* The questionnaire consisted of questions regarding usability and fun and was designed with scrolling methods. The children needed to scroll

the indicator to the appropriate emoticon that indicated their opinion as shows in Figure 5.11. Contrary to the formative study carried out before, this time, only three smiley emoticons were used, as shown in Figure 5.12. The number of smiley emoticons was reduced from five to three due to the experiences gained when conducting the formative evaluation. First, within the original scale with five points, the children had difficulties to select the most accurate response. For example, they found it difficult to differentiate between the smiley emoticons representing 'awful' and 'not good' and between the smiley emoticons representing 'really good' and 'brilliant'. Second, the children seemed to prefer selecting a 'neutral' answer (i.e. the smiley emoticon representing 'good') whenever they were stucked finding a suitable response. The main reason of introducing the *scrolling* activity was to gain the children's opinions by creating a fun data collection environment that is different from the traditional method of *ticking* the answers.

Figure 5.11 Scroll-Me. (*Left*) Usability and fun questions constructed in a questionnaire called *Scroll-Me*. (*Right*) *Scroll-Me* used by the participant and her facilitator.

Figure 5.12 Smiley emoticons. (*Left*) Five smiley emoticons used in the formative evaluation. (*Right*) Three smiley emoticons used in the pilot study in the summative evaluation.

In the summative evaluation, these instruments were combined and the numbers of questions asked were reduced based on several factors. First, the summative evaluation aimed to measure the performance of Gatelock with the listed tasks to be fulfilled by the participants. Therefore, the questions mainly focused on the aspects of performing the given tasks and the usability of the tool. Furthermore, based on experiences gained after conducting several studies with children, it is required to formulate the questions as short and easy as possible. This aspect was also concurrent with findings from related studies carried out by other researchers [Bruckman02][MacFarlane03].

5.4.4 Results

Two major findings arose from the pilot study concerning the list of tasks and the questionnaire. Since both children finished the tasks in less than 20 minutes, whereas they

were expected to complete them in one hour, there is a need to add more tasks. Firstly, from the list of tasks, several tasks are actually repeated purposely to evaluate how much the children learned from the activities. Secondly, the three Smileyometers used showed some inconsistencies. When answering the questions, the children seemed to be confused and distracted by the variety of colors in the Smileyometer. Hence, these inputs will be taken into consideration for the summative evaluation. It is argued that the colors affect the children's decisions, because they tend to choose the colors they like most instead of choosing them according to the meaning of the respective smiley emoticon. Therefore, the earlier hypothesis that colors can help children to decide more accurately had to be reconsidered. Consequently, the final smiley emoticons involved in the summative evaluation will be unicolored.

5.5 Conducting the Usability Study: The Summative Evaluation

After redesigning Gatelock and conducting a pilot study, another usability test was conducted called summative evaluation. The aim was to measure the usability performance of Gatelock, and the benchmark tests will allow creating a strategy to revise the usability guidelines according to the strengths and weaknesses of the Gatelock design. The goal of this study is *to identify the context use of Gatelock and the required measures of usability.* This includes an explanation of how the usability of Gatelock can be specified and evaluated. It also explains how measures of user performance and satisfaction can be used to realize how any functions will affect the whole system. Does Gatelock meet the objectives in terms of usability (effectiveness, efficiency and satisfaction) and fun adequacy? Does Gatelock serve the training and education needs of learners in terms of its effect on their knowledge, skills and attitudes. Children are expected to find Gatelock easy to use and they can create a simple game using functions such as game control, collision detection and logical gameplay. Gatelock will be able to entertain children to learn game programming. The study can reveal minor usability and fun-related problems with children as software evaluators.

5.5.1 Participants

Fourteen children between seven and ten years participated in the summative evaluation. Seven boys and seven girls were found to be familiar with computer terms and spend more than one hour per day playing computer games and using the computer. The children never used Gatelock before or did not have any experiences in programming languages or construction tools for children. However, all children were experienced in using drawing tools or similar. Figure 5.13 shows one of the children involved in the study.

Figure 5.13 A child involved in the study.

5.5.2 Procedures

Before starting, the children received a set of questions regarding their background information. Subsequently, a list of 25 tasks was given to them to be completed within 30 minutes. After the successful completion of the tasks, the children were involved with questions on usability and fun. For this purpose, a special questionnaire was designed.

5.5.3 Instruments and Data Collection

As mentioned before, the children received a list of tasks. But this time, the list was extended from 15 to 25 tasks, as shown in Figure 5.1. The tasks were designed according to the research needs, i.e. they include usability and fun aspects. The list can also be treated as tutorial to understand the operation and functionalities of Gatelock. Some of the tasks were repeated by using different game objects and characters to evaluate the children's understanding and learning performance. At the end of the study, the children received a *Scroll-Me* questionnaire consisting of questions on usability and fun. But as a result from the pilot study carried out before, this time, only three instead of five smiley emoticons were used and all of them had the same yellow color instead of three different colors. The final smiley emoticons involved in the summative evaluation will be unicolored (yellow), as shown in Figure 5.1.


User Observations and Associated Tasks and Requirements

		Info	Guidance
1.	Open/Run the Gatelock.	☐	☐
2.	Select existing game project named "test".	☐	☐
3.	Delete game project named "test".	☐	☐
4.	Select existing game project named "test2".	☐	☐
5.	Play game project named "test2".	☐	☐
6.	Stop the game and go back to main screen.	☐	☐
7.	Create new project.	☐	☐
8.	Rename new project to "Mein Spiel".	☐	☐
9.	Add player items – item "fisch".	☐	☐
10.	Resize item "fisch".	☐	☐
11.	Duplicate item "fisch".	☐	☐
12.	Re-locate the duplicate "fisch".	☐	☐
13.	Add background "meer".	☐	☐
14.	Define movement for item "fisch".	☐	☐
15.	Item "fisch" will move to any direction using arrow keys.	☐	☐
16.	Play the game.	☐	☐
17.	Go back to development screen.	☐	☐
18.	Add player items – item "pinguin".	☐	☐
19.	Duplicate item "pinguin".	☐	☐
20.	Delete one of the "pinguin".	☐	☐
21.	Define movement for item "pinguin".	☐	☐
22.	Item "pinguin" will move to any direction using arrow keys.	☐	☐
23.	Play the game.	☐	☐
24.	Go back to main screen.	☐	☐
25.	Exit Gatelock.	☐	☐

~ end of tasks ~

Figure 5.14 Snapshot of a list of tasks given to the children during the pilot study (refer to Appendix C1 for details).

Figure 5.15 The questionnaire *Scroll-Me* was revised by using only three smiley emoticons which have the same color.

5.5.4 Results

The process of making observations, formulating hypotheses, and figuring out the rules governing the behavior of a dynamic representation through a trial and error process is basically a cognitive process of inductive discovery. It is a process by which individuals learn much about the world, and, at a more formal level, it is the thought process behind scientific thinking and discovery. After transcribing the data, the results of the evaluation

study could be divided into two factors of usability and fun. Further the results discussed the implementation of the instruments used during the study. The results are based on observations of the children and their answers in the given questionnaires. All results will be presented in the following section:

Usability: Within the frame of this research, the three main usability aspects effectiveness, efficiency and satisfaction were tested.

a) *Effectiveness* was measured by looking at the number of errors made by children as they worked through the given tasks. These errors as well as the accuracy and completeness of the solved tasks were identified and recorded. The study revealed several problems the children had to deal with. These mainly involved certain features and functionalities that did not work as intended. Eight children indicated that they faced more than 12 errors, while others reported less than 12 errors. Some of these errors were based on the misunderstanding of conceptualizing and visualization aspects of the interface design itself. Nevertheless, the number of errors could be reduced compared to the number of errors revealed during the formative evaluation.

b) To measure *efficiency*, the three indicators of learning time, speed of performance and retention over time were used.

    i. To measure the *learning time*, it was determined which functions of Gatelock needed to be learned for being able to perform certain tasks. The children who had never used Gatelock before received certain tasks, and the time they needed to fulfill these tasks was recorded. In other words, the children were evaluated by making games based on a list of given tasks. The children faced programming challenges before they were able to become familiar with the Gatelock environment. But all children needed less than 5 minutes to understand the specific rules for making a game. After setting the same tasks in a different context, the children seemed to work quickly and accurately without asking for help.

    ii. To measure *speed of performance*, a benchmark task or tasks were specified and the children were asked to perform these tasks using Gatelock. Their actions were recorded to see how long they took to complete the given tasks. When evaluating their ability, the children had similar results. All children spent less than 15 minutes to complete the tasks and kept repeating them just for self-amusement.

    iii. To measure *retention over time*, the same group of children was required to participate in the tests more than once. Since the summative evaluation only involved fourteen children within a very restricted time, it was difficult to repeat the tests with the same children. Therefore, it is suggested to conduct the study to measure retention over time on Gatelock during a longitudinal study which is focusing on a longer research duration.

c) *Satisfaction* is best measured by asking the children to rate Gatelock, since only they know whether they like the tool or not. By using questionnaires and surveys, the children's opinions can be discovered. The children often responded slightly more positive, especially when they knew that the person administering the questionnaire has a vested interest in the outcomes. Thus, the children were strongly asked to answer the questions honestly. The *Scroll-Me* questionnaire was used in conjunction with the Smileyometer to indicate the answers. The findings revealed that all children were satisfied with Gatelock, including the rich features embedded in it. But some children also suggested several improvements on Gatelock, especially regarding the management of the program with own artifacts, including game characters, game elements, and background game scenes, and they wished to program in their native language.

d) *Fun* it is best measured by observation and by asking the children. The fun survey relies on self-report responses, a technique not commonly associated with children. Designing a game for children is fun, but also difficult. By means of facial and body language expressions (such as smiling and laughing with other children participating in the design process), it could be observed that the children enjoyed the game making. Their facial expressions and feelings were honest and they provided valuable input for the study. In fact, some of the children showed a strong interest in saving the software on their own computer and they were very proud about their work and started to brag in front of their families and friends.

Another outcome of the evaluation study concerns the implementation of instruments used throughout the evaluation. It was a good practice to conduct a pilot study before the real usability evaluation. In both pilot studies, some problems of the instruments that are going to be used could be found directly with the children. This enabled the researchers to design more meaningful, appropriate and useful instruments.

5.6 Findings of the Two Usability Studies

Two usability studies were conducted to enable an effective involvement of users in the early stages of the game authoring tool development. Furthermore, these studies concentrated on the external design of the tool which rapidly offered created and modified versions of its user interface. The user interface design of the game authoring tool is dependent on the children's characteristics, the functional requirements of the tool, and the usability guidelines. Both usability evaluation studies succeeded in establishing children's specifications that consisted of interaction requirements and needs. The studies reduced the problems revealed in the game authoring tool and in the methods, procedures and instruments used along with the studies. In both the formative and summative evaluation, issues such as usability, functionality, learning and social impacts became the primary focus of the evaluation. The formative evaluation assessment concentrated more on the potential users of the tool—the children—and the environment to test the tool. During the summative evaluation, the designers of the tool could either decide to continue designing the tool, to throw it away, or to extract the main requirements from the prototype and use them for the real tool. Either way, by repeating the design process through several usability evaluations, positive impacts can direct the design activities for the game authoring tool together with the usability guidelines used as benchmarks. However, the primary limitation of these studies was the small sample size, particularly for the summative evaluation. The small size harms the power of the analyses which can naturally affect the significance of the testing.

Thus, a larger number of samples would be required together with a good plan and an appropriate amount of time and human resources.

The game authoring tool used by the children in this research was observed to get them involved in the design process. When the children were engaged in exploration, they were actively working. Scholars believing in the constructionist learning approach totally agreed that children can learn by doing [Papert93][Gray98] and having hands-on experiences [Harel90][Kafai96b] in understanding certain contexts instead of simply memorizing the facts and steps on how to conclude the given tasks. Below are several examples of how the game design activities can embrace children's learning following the constructivist approach.

Learning by doing – Game design can serve as motivator for children to acquire a deep understanding of the subject matters, either by designing and programming games or by integrating related subjects such as mathematics and sciences. Some children were observed to start asking why they could not do certain things during the game design in the first place, such as changing bullets (missiles) to new images like sweets or candies. Observing children's reactions and thinking in real life showed that they could nurture their creativity during the game design. On top of that, the children shared their skills in designing games and continuously communicated with other children. Here, teachers can play a role as facilitators rather than instructors to assist the game design activities. It is believed that for the benefit of learning by designing games, the game authoring tool excels to provide a rich creative environment that children love to explore and discover. This creates '*ah-ha effects*' of learning to children, especially when they gain new experiences based on their reflections on the causes and effects during the design and construction of games in order to achieve their intended goals.

Learning by making mistakes – During the design of their games, the children started posing questions and seeking for answers by observing their own interaction with the simulation and finding a sense in the things they discovered (*trials and errors*). This can help children to better understand the planning and designing aspects. Once they were familiar with the content and the ways to program and design games, they showed their skills and capabilities in fulfilling the given tasks. The children also learned how to solve certain tasks by observing their friends and instructors.

Learning by watching – By watching others, children can comment on other children's games and decide which game is more compelling and convincing for them to try playing. This time, the children acted as critics with a detailed understanding of specific actions carried out by others. An observation revealed that children with prior experiences in playing games were engaged much faster than those without any experiences, and that these children shared the idea of technology transfer with others.

Learning by playing – The children who played other children's games were observed to show critical thinking skills by giving comments and opinions about the respective game. They were also able to make suggestions on how to improve the games and sometimes they demonstrated how to do it. Here, the children showed their capability to share skills and knowledge collaboratively and cooperatively.

Without a doubt, the game authoring tool designed in this research has met the purpose of designing simple 2D games and nurturing learning for children aged seven to twelve. It is

suggested to use the designed tool to measure learning according to the constructive approach by using suitable measurement methods, such as methods used in the usability engineering field. In other words, to measure learning in game design activities, researchers can incorporate a holistic approach by integrating suitable and appropriate evaluation methods such as observation, think-aloud, interview, questionnaire or heuristic usability.

5.7 Recommendations from Children

During the game making process, the children had to solve some problems, whereas each step involved to find solutions and to get along with the process of refinement. Regarding the games, the children were not told any rules of play in advance. They had to figure them out themselves by observation, trial and error, and a process of hypothesis testing. The rules go beyond the decoding of the meaning and usability of individual icons on the screen [Grobelny05]. Besides figuring out what the symbols mean, the children had to discover how they act. However, all children attempted to ask questions and they were directly guided before proceeding to the next level.

The children also recommended the game design activities to be integrated into subjects such as sciences, history, geography, and so on. This shows that the children were motivated to learn any lesson while making games. It is believed that the possibilities to integrate any subjects or lessons into game making activities are endless, especially when it comes to the idea of using a program such as Gatelock in the classroom. Creativity is not just a question of creating new solutions, but creating better solutions, and this requires a critical judgment. By understanding this, learning while making games can really connect motivation and creative thinking with learning in any school subjects. These issues go back to the prior knowledge and preparation of teachers and educators to implement Gatelock and to put it into practice.

5.8 Revision of the Usability Guidelines

In Chapter 2, a set of guidelines was formulated for designing a game authoring tool for children. The results from the survey were analyzed based on literature and were used in the preliminary study described in Chapter 3. The usability guidelines were also refinded again, due to a number of interesting aspects found in the preliminary study regarding the appropriate environment of the game authoring tool. The findings were separated into general and specific ones. General findings address usability issues that apply to the user interface as a whole, such as children having problems to understand how to apply game rules within the Gatelock environment. Even though the children easily learned how to program rules, enjoyed using the environment and built creative simulations within a reasonable complexity, the problem of understanding the environment can cause frustration. Specific findings addressed usability problems associated with specific user interface elements, such as an ambiguous icon. Especially in case of a formative evaluation it is helpful to present the problems in a format that describes them in detail, recommends how to solve them, and prioritizes the solutions. As a whole, it is important to note that a formative evaluation outcome should include positive as well as negative feedback. The summative evaluation often only reports the data, excluding design recommendations, because it is too late to make changes in the development schedule. Again, the refined usability guidelines were used for two usability studies, i.e. the formative and the summative evaluation. After considering the outcomes from these studies and understanding the relationship or the impacts on the earlier usability guidelines, the latest

usability guidelines were simplified in Chapter 4. The enhancements can be seen in Figure 5.16.

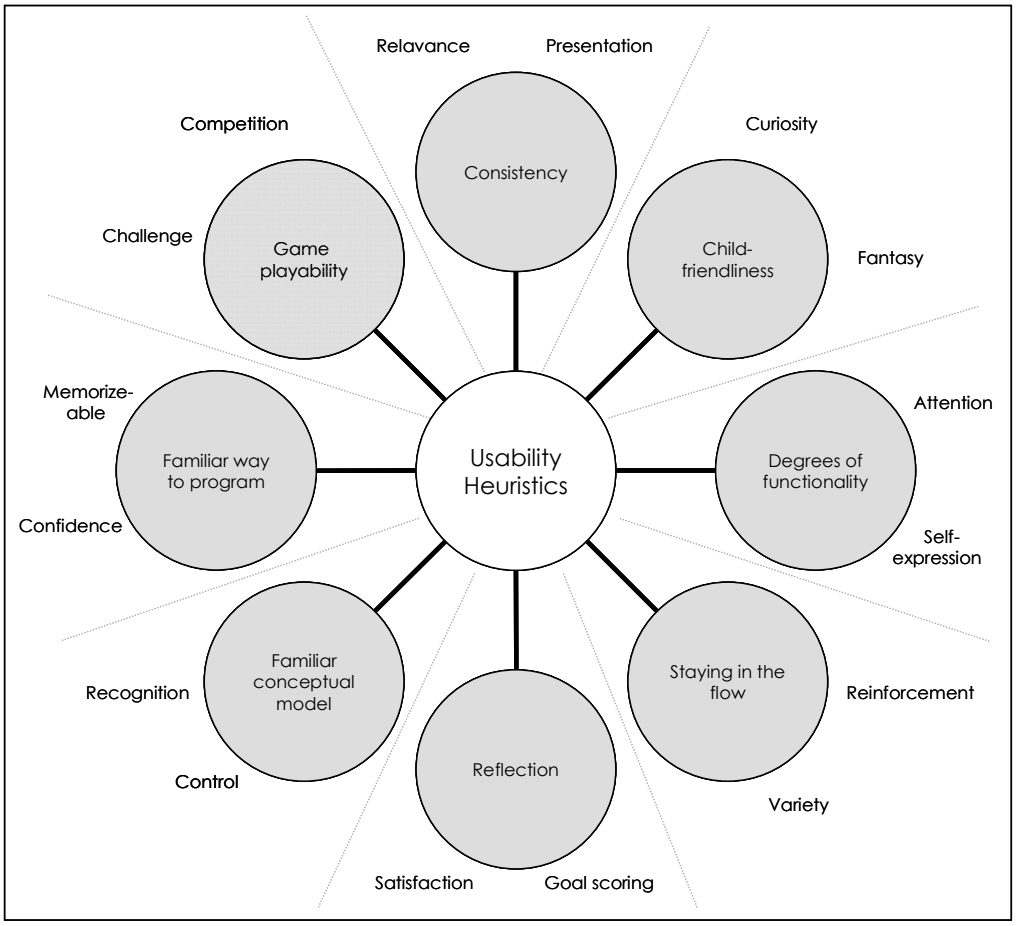


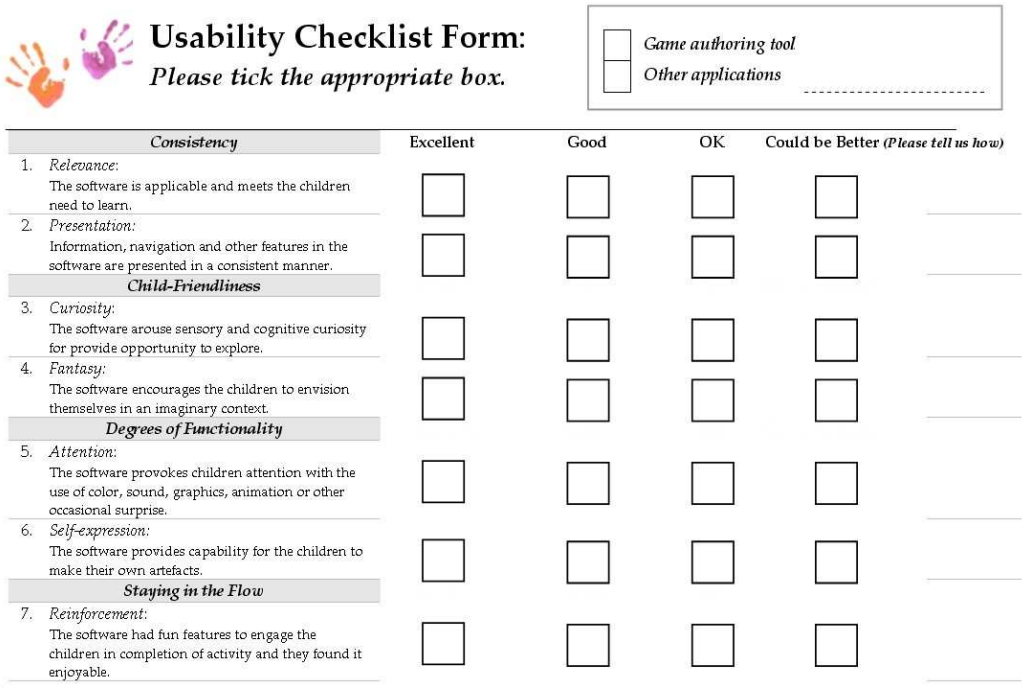
Figure 5.16 Usability guidelines after several refinements based on the preliminary study and two usability studies conducted throughout this research.

Figure 5.17 summarizes the revised usability guidelines with an enhanced explanation based on the findings and feedbacks from the formative and summative evaluation. This time, each heuristic has been specified according to the related criteria to simplify the usability guidelines to be used in developing and evaluating a game authoring tool for children, or for developing and evaluating other types of software for children.

5.9 Summary

In this chapter, findings from two field studies (formative and summative evaluation) have been revealed. As mentioned before, the developed usability guidelines faced a number of changes based on the outcomes of the conducted studies. Before conducting the evaluation studies, pilot studies were carried out to test the instruments before using them in the real studies. Basically, the pilot studies helped to identify minor problems with regard to the instruments used and became important for conducting the evaluation studies. The formative and summative evaluation also offered positive input for the design and

development of Gatelock. The inputs were treated as main players in designing the usability guidelines. Again, the usability guidelines have been refined in order to grasp important facts upon designing a game authoring tool for children. The refinements will be described in the next chapter together with a thorough discussion on Gatelock and the usability guidelines. This includes explanations about the achievements, but also limitations, of Gatelock throughout this research. Subsequently, the computational power of Gatelock in boosting children's understanding on how to program will be described, including explanations on future works to improve Gatelock, and reflections on the usability guidelines.



Usability Checklist Form:

Please tick the appropriate box.

☐ *Game authoring tool*
☐ *Other applications* ----------

	Excellent	Good	OK	Could be Better *(Please tell us how)*
Consistency				
1. *Relevance:* The software is applicable and meets the children need to learn.	☐	☐	☐	☐
2. *Presentation:* Information, navigation and other features in the software are presented in a consistent manner.	☐	☐	☐	☐
Child-Friendliness				
3. *Curiosity:* The software arouse sensory and cognitive curiosity for provide opportunity to explore.	☐	☐	☐	☐
4. *Fantasy:* The software encourages the children to envision themselves in an imaginary context.	☐	☐	☐	☐
Degrees of Functionality				
5. *Attention:* The software provokes children attention with the use of color, sound, graphics, animation or other occasional surprise.	☐	☐	☐	☐
6. *Self-expression:* The software provides capability for the children to make their own artefacts.	☐	☐	☐	☐
Staying in the Flow				
7. *Reinforcement:* The software had fun features to engage the children in completion of activity and they found it enjoyable.	☐	☐	☐	☐

Figure 5.17 Specific criterias in form of a checklist for the usability guidelines for designing a game authoring tool for children. It can also be used to evaluate the appropriateness of other type of children's software (refer to Appendix D for a clear and detail view of checklist form with its details).

6
Gatelock and Usability Guidelines

In this chapter, a thorough discussion on Gatelock and the usability guidelines will be held. This includes an explanation of the achievements, but also limitations, of Gatelock throughout this research. Furthermore, a description of the usage of Gatelock for boosting children's understanding of how to program will be included. The chapter will end with an explanation on future works to improve Gatelock, and with a reflection on the usability guidelines. These guidelines will be discussed in detail by comparing them to the design and development of Gatelock and the studies that were conducted in the beginning of the research right up to the summative evaluation study. The usability guidelines are expected to be advantageous for those who are involved with the design of children's software products and especially with game authoring tools for children.

6.1 Achievements and Limitations of Gatelock

Since Gatelock was fully presented and used by the children in several studies, believable achievements could be made regarding its features and capabilities. However, Gatelock also showed limitations that were identified by the studies carried out throughout this research. In this section, the features of Gatelock and its achievements and limitations will be described.

6.1.1 Features of Gatelock

Some issues are involved in the research, especially to identify design features that could make the game authoring tool more fun and motivating. To address these design issues, four elements that can reflect the cognitive structure of a child, will be listed below.

Dynamic Representation – In this tool, the choice of appropriate metaphors is a vital design decision. Visual programming can provide an ideal testing basis for the visualization of object-oriented concepts. By using iconic authoring environments, the tool presents a program structure in a flow chart metaphor. Together with multimedia techniques, especially animation, the tool will provide a way to illustrate the nature of building a program by children.

Drawing Activity – The tool includes a drawing activity, where children can draw what they intend to and turn their intention into visualization. By using drawing sketches and images of the program, the children can identify the behavior of their objects according to the story of the game they want to produce. They use drawings as a creative way of learning and expressing themselves.

Game Workflow – The tool provides a typical workflow on how to create a game derived from the game production pipeline. The objects such as game characters and backgrounds can either be created by selecting them from the predefined library or by drawing them from scratch. After drawing these objects, the whole designing process will

take place by placing the related objects in the development screen. Afterwards, each object will be defined according to its intended behaviors by using the option menu. The option menu is responsible for elements in the game or in the related level. These elements can be programmed to perform certain actions, thus exhibiting a gaming behavior.

Rich Media Manipulation – Initial activities in traditional programming environments typically involve the manipulation of objects or simple graphics. In contrast, the tool manipulates images, animations and sound through attribute manipulation. By giving children the possibility to control rich media, the tool supports programming activities that strongly correspond to the children's interests.

6.1.2 Achievements

Several usability studies were conducted during this research. Within the fields of CCD's practice and research, a number of usability evaluation methods and techniques have been developed and used, for example a combination of think-aloud with active intervention method, as well as questionnaires and observations to evaluate the usability of the game authoring tool. The think-aloud technique was implemented to capture events from usage situations, problems, expectations, and so on. The generated data recorded the cognitive processes of children during the software usage. Another method was questionnaire which followed the inquiry method. In this method, the data from children's opinions and suggestions were recorded using either quantitative or qualitative approaches. The method usually focuses on children's likes and dislikes, needs and understanding of the tool. Furthermore, a number of observations were made throughout this research. For this purpose, guidelines for usability testing for children [Hanna97] were implemented. The usability evaluation methods and techniques were mainly chosen because they represent a process-oriented and not a product-oriented method [Wiberg03]. In addition, the data collection has been accomplished with the creation and usage of evaluation instruments such as questionnaires, *Pick-and-Paste*, *Select-and-Scroll*, an observational checklist and a Smileyometer rating scale. All these methods were practically used with children in their natural environment (field research). The scope of evaluation instruments and the developed prototype interfaces have been used and tested, and recent discussions about the need for new types of measures have been raised.

Gatelock was designed based on the usability guidelines created earlier, on surveys, and on the preliminary study carried out in the first place. The redesign process of Gatelock also followed results from the studies conducted with children. The iterative process of redesigning the game authoring tool allows conducting more tests and evaluations with children. The approach facilitates and encourages discussions and communication between the designers and the children. Hence, it can be stated that the approach encourages the children to actively participate in designing the tool. The main advantages of Gatelock have been summarized as follows:

Gatelock is flexible enough to implement a wide variety of simple arcade games despite its limited ability to program games using its drag-and-drop functions and visual representation.

Gatelock provides easy-to-use features for children of all ages. The children, who used Gatelock, commented on this aspect, as stated in Chapter 5. However, the studies were intended for children between seven to twelve years. Further studies need to be conducted for other age groups.

Gatelock offers a wide range of objects that can be viewed and manipulated. Furthermore, it offers unlimited access for children who want to create their own objects. Furthermore, the objects can also be designed in any application software and be easily retrieved by using Gatelock.

Gatelock has a simple entry interaction where children do not need to drag and drop the intended objects to a specific location. The interaction is simple, since it is just based on looking at and selecting (point-and-click) the intended objects or events. To do more sophisticated things, the children need to know and use the scripting editor that is hidden and still in working process.

Gatelock aims to maintain enjoyment and give children a degree of freedom when creating their own games individually or in groups.

Gatelock encourages children to see and feel the experience of designing games and learning aspects that are related to their subjects.

Gatelock's rules to program games are represented by visual images that show the effects of the scripts or programs. In Chapter 5, it was shown how easy the children can use them.

Gatelock allows a simple modification of objects. But it does not allow children to create their own objects. They have to design these elements creatively based on their gameplay experiences. The limitations and recommendations to improve Gatelock are going to be described in the next sections.

With the overall achievements and further implementation of Gatelock, the tool is expected to have a great prospect to be a teaching and learning tool in the classroom. The studies with children showed that the game design activities really attracted them in learning how to use computers and incorporate subjects such as mathematics, sciences, languages, and so on. The entire research has shown that learning is most effective when the following four fundamental characteristics are present: (1) the active engagement of children, (2) group participation activities, (3) frequent interaction and feedback by children and the designed game authoring tool, and (4) the connection of game design activities to the real world.

6.2 Missing Capabilities of Gatelock

Even though Gatelock has some better capabilities than other game authoring tools or construction tools, there are many shortcomings. This is due to the condition of Gatelock as a prototype which has limited functions that are necessary for a game authoring tool. Below are listed some features and functionalities that Gatelock does not have:

Game genres – Due to its limitations, not all types of games can be designed using Gatelock. In the first place, Gatelock was developed as a construction tool that allows children to design simple arcade games. However, it can have multiple game levels.

3D capabilities – 3D games cannot be designed, since Gatelock does not have any 3D features. Gatelock was not designed for 3D applications, as it is related to the ability of children to perceive and accept the 3D environment according to their age, mental model and experience. But further work can be done to improve Gatelock, as done before with the implementation of 3D features in the iEngine [Masuch07].

Normal features – Gatelock is different to other construction tools such as drawing software. It does not allow children to print or send their work by email. However, the designed games are playable and can be redesigned as many times as desired.

Collaboration features – Gatelock is not able to communicate through networks. This limits the collaborative and cooperative activities among the children. However, the children can be involved in collaborative and cooperative activities by designing their games in groups.

6.3 Improvements of Gatelock

Due to its limitations, this section describes several parts of Gatelock that can be improved in the future. Firstly, verbal instructions should be considered towards a greater pictorial or symbolic approach. This feature can assist children in form of an interactive character offering guidance. The most important thing is that the agent should be audible or muted according to the children's choice. Another suggestion is to enable teachers and students to compose more choices of their game contents and activities with the familiar ease of desktop publishing. The design of the game authoring tool should consider the needs of non-technical authors, particularly of teachers. Moreover, the teachers should be able to add content components that can be embedded in a variety of choices. For example, the teachers could add additional content to an activity in a variety of media types such as movies, sounds, pictures or game background scenes.

Gatelock could also improve another feature which allows teachers to explore the use of scripting languages to support their programming, where desirable. Simple scripting programming languages in Gatelock involving textual scripts can allow children to gradually move towards learning real programming topics. This type of embedded scripting language offers a fine degree of control over the environment. Furthermore, it offers teachers or even students to learn how to program. However, teachers can also work with the Gatelock programmers to add scripting where their activities require an additional customization. Furthermore, both visual programming and scripting features could be useful to support educational authoring during the game making process.

During the design and development process, Gatelock faced numerous changes. Since the interaction design served as starting point for designing Gatelock, the focus was on the process of use and interaction. Designing Gatelock was a big challenge, but redesigning it was an even bigger challenge. These problems were addresses by breaking up the whole task and focusing on details for improvement. This led to several ideas that were later tied together in the redesign concepts for the implementation of the whole environment, as described in Chapter 5.

6.4 Reflections on the Usability Guidelines

Usability testing is often an intense experience when the users explore the interface design for the first time. Employing actual users to test the developed software can harness their fresh outlook to help analyze the usability of the interface, navigation and interactive controls, and the overall appeal of the software. Designers of children's software face many challenges regarding design principles used for adult interfaces that cannot be applied to children's software due to their different skills, needs and expectations. Based on the design and development of a game authoring tool, the following section will reflect the usability guidelines designed throughout this research. These guidelines were collected and organized from several sources and findings of several studies. They can be treated as heuristic usability in designing a game authoring tool for children with a thorough description for each given heuristic. Together with HCI theories, principles and standards, the designed guidelines can help designers and researchers to produce good and better software products, respectively [Yatim07d]. The guidelines developed during this research also provide a clear understanding of the principles of a game authoring tool. The guidelines can support and consolidate the research in two ways. First, the guidelines can be used for the creation of a game authoring tool. Second, the guidelines can be used for the evaluation of a usable game authoring tool.

The usability guidelines presented in this research consolidate a variety of disparate studies on children's software and provide a first collection of guidelines specifically oriented towards designing a game authoring tool for children. They can be used by designers as formative design guidelines or as a basis for evaluating similar software for children. However, there are also substantial differences that call for a thorough consideration to combine usability with the games field. This is basically handled by balancing the gameplay and game design features to meet children's needs in terms of effectiveness, efficiency and satisfaction within the element of fun.

The usability guidelines can be improved and refined by repeated studies during a longitudinal study in many ways. First, the age group of children should be further studied. Since many design principles are age-specific, more research and studies need to be carried out to test the guidelines with different age groups. In software design, age-specific means that children are going to be grouped according to their age. The categorization is based on things that children like to buy and use in terms of marketing and consumer behavior [Hanna97]. Furthermore, the categorization is also based on children's cognitive development [Piaget83], as explained in Chapter 2. Second, a study on usability guidelines in terms of cognitive and physical development should be carried out. Due to the availability of technology and the increased amount of tangible software for children [Weevers04], special treatment in designing usability guidelines (or heuristics) for tangible aspects should be clearly defined and can be matched with interface design principles [Barendgret03].

Lastly, the usability guidelines presented in this research provide guidelines based on the design of a game authoring tool. However, as technologies steadily develop, so will the usability guidelines. This means, with upcoming new technologies and gadgets, the usability aspects of these technologies will definitely change and grow based on their specific features. For example, introducing tangible interaction devices or tangible user interfaces to children may change at least one or two heuristics in the existing guidelines or may add more heuristics to the guidelines such as natural interaction and navigation [Field01]. In

certain cases, specific usability guidelines will cater for a specific design or software evaluation. However, not all heuristic usability guidelines for specific software can be used for other types of software.

Therefore, further work is required to revise the software design and evaluation methodologies. Despite the need for more research in various aspects, the usability guidelines in this research have gathered a valuable set of information for designers to design a game authoring tool or similar construction software. However, it became clear that some usability issued cannot be fixed. Some designers were uncertain whether proposed solutions will successfully fix the usability problems [Medlock02]. This research also encountered constraints in producing a list of usability guidelines involving children. Fixing usability problems takes time and resources, but when performed, it often produces useful results.

6.5 Lessons Learned in Conducting Usability Studies with Children

With all studies described, this section explores the experiences gained in involving children in the software design from the initial stage of information gathering to the testing and evaluation of the software. The experiences are classified according to several aspects which are initial plan and requirements, methods used for evaluation, instruments used for data gathering and other related points [Yatim09b].

a) Initial Plan and Requirements

Early Preparation – Advanced research preparation is a must, especially for those who want to include children into their studies. This step is even more crucial if researchers choose to conduct a field study in schools [Sauvé05]. There is a need to identify which school to include in the study, and which school (rural or urban) really meets the study requirements. This includes, for example, minimum hardware requirements in school computer labs. In addition, researchers need to find out if there are any procedures to be followed to conduct the study in the particular school [Ferrari06]. For example, in Malaysia, any research involving children in schools must go through several steps. Requesting permissions from various authorities, such as the Ministry of Education, the State Education Department, the Economic Planning Unit and the school headmaster/mistress, are necessary to gain access to the research setting. Usually, the permission takes approximately 3 to 4 months. Then, before arriving at the school, the researchers are responsible to be fully prepared with questions to be answered, especially for the school authorities. Questions, such as the place where the study is going to be conducted (whether in a classroom or a computer lab), the duration of the study, the age group or class to work with, the time required to observe or to work with each child, and any limitations on activities in order to complete the study, need to be answered before conducting the study. Sometimes, the answers for these questions together with the earlier plan can be critical factors for choosing research methods for the study. For example, the constraints of time and access could influence the fieldwork, and sometimes the sample size necessitates either to use a qualitative or a quantitative approach for the study.

Age – The findings of the studies differed when evaluating the same software with children of different ages. In one study, children from third grade (approximately nine to ten years old) still needed some specific advice and had different ideas about what to do with the tested software. A twelve year old child suddenly stopped using the software after a certain time and started to do other things. The child stated that it had apparently already done everything possible with the software and got bored. This was the only incident where an older child interrupted the given evaluation tasks, whereas the younger children kept on using the software until the end of the evaluation session because they had many ideas, and some of them even wanted to take the software home. The younger children did not wanted to use the predefined pictures in the software. Instead, they preferred their own drawings, since they were nicer according to them. While older children wanted to explore more, the younger ones wanted to get external input, whereas there are no features in the tool allowing them to implement it. By experience, the same point also came up in other studies indicating that the age matters a lot [Druin99a].

Experience and knowledge – Computer literacy of children played the most important role for handling the software, and directly influenced the evaluation results. Another important aspect is, for example, the children's previous knowledge (mental model) of drawing. The more computer literate children are, the more they will ask for special functions and compare a program with existing features in software they are familiar with. For example, in one study, the children understood the drawing function after tying to draw several times with and without asking for help. The children's understanding of the metaphor used in the software was beyond our expectation. But sometimes the children were not able to relate some functions in Gatelock, for example, with things they used in everyday life. A simple and brief explanation had to be given to them, especially on how to use different functions such as adding new game characters or coding intended events. This may be related to the design of the metaphor itself which can be claimed as inauspicious and ambiguous. Works on eliciting the mental models of children with research methods such as interviews and questionnaire were also done aiming to find the relationship between mental models of children and computer literacy [Jensen05]. During the study it was concluded that both methods can be used to elicit the mental model of children, especially for children with moderate language skills. The researchers also suggested a combination of both in getting sensible research findings.

Psychomotor skills – Handling the mouse as input device was first expected to be more problematic. But this turned out to be wrong. Due to the use of their parents' computers, children (of all ages) often already had prior experiences in using the computer, and especially the mouse. The actual process of painting with the mouse turned out to be difficult, since the drawing pen movements with the mouse are different compared to pen movements (this was revealed in the study of the drawing tool, since drawing with a pen offers more degrees of freedom) [Read01][Dachselt08]. But the children, especially the younger ones, did not care about these matters. In fact, they continued drawing and practicing. After a certain period of time, the children mastered to draw with the mouse because they were encouraged and motivated and always found new things to try. This approach was remarkable, especially during the evaluation of the drawing tool software with the children. They hardly needed a creative impulse for painting, drew ideas from everyday life and included them into their pictures. But, some of the children had difficulties in handling the mouse, since its size is not appropriate to their hands and the design is

not ergonomic for them [Inkpen97b]. Thus, for future works, other interaction devices such as graphics tablets with pens or even fingers as text or input entry for children could be used to replace the traditional mouse interaction [Browne00][Read02].

Types of Software – Multiple types of children's software were involved in the user studies conducted. The categories for children's software also contribute to the nature of research settings. Most of the software uses visual representations that are only used on normal computers, whereas some of them need special devices for interaction purposes, especially software that is combined with tangible interaction in tabletop environments. Gatelock has been tested in a tabletop environment and it was proven to work with minor changes such as adding keyboard and mouse as text and input entry. This was necessary because the direction design of the game authoring tool is based on the personal computer in the first place. Therefore, to use the same game authoring tool with other technologies such as a tangible tabletop, computing for children will need different or additional interface and interaction features. Nevertheless, to implement a user study for this purpose, children need to come to the specified location or lab, since the tabletop is not removable-friendly. But new technologies are introduced such as the SMART table that is expected to be appearing and accepted in the classroom in the future. Another point is that the game authoring tool itself is more like an application to build other applications (games). This means, the evaluation processes are slightly different from evaluating other application software because they involve two types of software testing, which are testing the application used and testing the application designed. Evaluating computer game concepts with children also includes additional aspects, such as involving children in the further development of a given concept, evaluating the concept itself, and not forgetting fun and storytelling as important elements in designing a game [Hourcade04][Smart09]. Another point of view is that most interactive software tools are the result of adults' imaginations and not children's [Montemayor01]. Therefore, a way to easily program this kind of environment, without taking the child away from their physical world and the act of storytelling has become an important agenda in choosing and evaluating software for children.

b) Methods Used for Evaluation

Research methods and data gathering – Selecting and implementing the appropriate research methods can be a tedious task. A classification of existing research methods was described for designing technologies for children [Jensen05]. Some research methods that are mostly suitable for gathering initial information even before starting developing software for children can be used as early as possible. Some research methods can be implemented while conducting the software evaluation and testing. The important thing is to select the most appropriate method that can be implemented in a suitable place and at the right duration. On the other hand, evaluation methods used, for example, in tangible computing for children also need special attention, especially in selecting the best and appropriate evaluation methods. This research implemented observation techniques in all user studies in order to grasp the children's perception on the usage of the tools with their gestures, facial expressions and behavior. Nevertheless, the researchers need to be trained to observe these aspects, including how to interpret the children's behavior and how to record the test data.

Group or individual work – Collaborating and sharing ideas among the children offered benefits and feedback for our software development and design. During the evaluation phase, the children wanted to share their ideas on how to do certain drawings with friends. This collaboration among children indicated that they are not only learning to use the computer or the software, but that they are also learning with their peers and share their knowledge. Therefore, working alone or in groups can contribute to the findings of user studies.

c) Instruments

Types of instruments – After choosing the research methods, the right research instruments need to be considered to achieve the research objectives. To a large extent, the quality of research depends on the quality of the data collection tools. Interviewing and administering questionnaires are probably the most commonly used research techniques. Therefore, designing good questioning tools or instruments forms an important and time-consuming phase in the development of most research proposals. This includes concentrating on face-to-face interviews or designing questionnaires and types of questions used in the interviews. In one study conducted during the formative evaluation of Gatelock, it is useful to use pictures or drawings when asking certain questions, especially regarding a small-scale study. In case of illiterates, a questionnaire may even consist exclusively of pictures. Therefore, it is wise to conduct a pilot study to explore the appropriateness of the instruments before actually using them in a real study.

Questions asked – The children did not care about the question which software features such as input devices would be more suitable for them to use [Read05b]. They really want their opinions to be heard and considered in the design and evaluation of the software. For example, they did not have problems in using the mouse and drew their ideas free from any restrictions or boundaries while creating game characters or backgrounds in the game design activity. For them it was not about the perfect look of the drawing, but about the freedom to express their ideas, and drawing was the most valuable experience.

Rating scale – The number of scales in rating an opinion or an input for given questions should also be considered as important in data gathering. Although the 5-point rating scale system works fine for most questions, there are always questions that cannot be answered using the scale, which leads to the incorrect but only viable answer of 3 (undecided). The *middle value* is usually chosen by the respondents for a neutral answer (maybe) [Breakwell95]. Sometimes, forced-choice response scales are used with an *even* number such as 4-point so that the respondents lean more towards agreement or disagreement (yes or no). This approach is said to deny the respondents' right to give undecided answers. Many researchers commonly use a 5-point rating scale format [Infosurv06], and others agree that there must not be more than 7 points [Miller56]. The number of points used in the rating scales leads to another issue: the use of numbers, for example, marked "1 to 5". The numbered scales are less accurate than scales marked with labels such as "good" or "satisfying". However, problems with quantifying labeled scales will occur when it comes to decide between "strongly agree" and "slightly agree". It is assumed that two issues—the number of points in scales and labeled scales—are needed to be revised, especially when children act as respondents. Therefore, in this research, the

number of point scales was reduced from 5 to 3, and pictorial images (Smileyometer) were used instead of labeling the scales with confusing words. Another factor is that the children do not take the time to be accurate in their rating; usually they chose the numbers randomly according to how much they liked or disliked the questions without taking care of reading or even understanding them. As described above, this also had a huge impact on the questions asked. Another aspect that can also contribute to the misinterpretation of true meanings is the usage of colors for the rating scale, as shown in Figure 6.1.

Figure 6.1 Smileyometer replacing the Likert-type scale. (*Left*) Smileyometer with different colors indicating different scales used. (*Right*) Smileyometer using the same color referring to different scales used.

Language effects – Children have varying speaking and writing abilities, including the use of different languages. They have their own interpretation of words, and sometimes, when computer jargon was used, the children seemed to be lost. The problem got worse when the jargon was translated into words they had never heard before. For example, in one study, a joystick was used as interaction device for Gatelock. To ask questions on the usage of the device, Malay was used, since the research was conducted in Malaysia. However, some children did not understand the translation and rather used the English word *joystick*.

d) Other Related Points

Venue of the study – There was a huge difference when conducting the evaluations in primary schools. Some differences became obvious within the evaluation turns between nursery and primary schools, i.e. the different educational learning concepts seemed to have a strong influence on the children's way to approach given tasks. Based on the experience with children, it is suggested to consider a proper setting of the computing environment for children. Due to different kinds of research settings (nursery schools, private and public schools), it was discovered that children in different settings exhibited different ideas, creativity and fantasy. This may be due to their respective school environment where lots of drawings as well as mural and creative paintings surrounded them. It could be observed that the children felt more at ease and focused with the user study when it was conducted in the school computer lab. No user study was conducted in the usability lab, since it was intended to involve the children in their actual learning environment. Nevertheless, in general, the classroom was a hectic and often unpredictable environment for the field study.

Researcher-Child Ratio – Conducting a user study with children involves human relationships and management. Researchers should consider the number of children and the number of researchers or assistant researchers participating in the study. Too many

children and only a few assistants could make the study imprecise and random. On the other hand, too many researchers could scare the children and make them act passively. Again, this factor contributes to the research method applied in the study. An implementation of a 1:5 researcher-child ratio for children aged between seven and twelve is recommended, but needs to be studied in detail.

Time – Researchers need to consider a few aspects during the initial planning stage. They need to consider the school calendar to find the best time to conduct the study. For example, Malaysia is a multi-ethnic society with many festivals and public holidays, and this may affect research time and duration. Second, every school has its own activities such as fire drills, teachers' days or unexpected events conducted on the same day as the study. Therefore, researchers need to agree the school activities with the headmaster/mistress so that the study will not be affected.

Involvement of other people – Even though most of the user studies conducted for children involve children, sometimes also the participation of their teachers may be required. In one study, the involvement of teachers was tried to avoid, since they could not help their students. But since the study was conducted in the school, this type of intervention was inevitable. In such a case, it is up to the researchers to handle the situation politely and with reasonable apologies. There are also some rules about getting the parents' permission before incorporating their children in the study. Researchers should follow this procedure (if any) and briefly explain the study and its workflow to the parents. In some cases, parents wanted to be involved in the process. In terms of even better understanding the children's behavior and expectations towards the software it could have also been helpful to include teachers and parents in the study.

6.6 Summary

This chapter addressed the achievements and limitations of Gatelock based on the outcomes from several studies conducted throughout the research. It also explained the capabilities that Gatelock has, and the ones that it does not have. Furthermore, some suggestions to improve Gatelock were described. On top of that, the chapter also made some reflections on the usability guidelines iteratively established during the research. The chapter ends with a description on the lessons learned in conducting usability studies with children. The next chapter will conclude the whole research with emphasize on the major conclusions of the research and a discussion about the reviews of each chapter, including overall contributions and future perspectives in the field of children and interaction design.

7
Conclusions

To convey children's intentions into realization still remains one of the main challenges in the field of children and interaction design. This dissertation discussed the detailed aspects of game authoring environments especially designed for children according to the usability research. The design of the tool concentrated on a flexible interaction and construction activity for children. In this research, the field of children and programming was discussed and the implementation for the design and development of the game authoring tool for children was presented. Finally, this chapter concludes with future work to be added to the study.

7.1 Major Conclusions Drawn from the Research

The usability guidelines designed throughout this research are guidelines that emphasize on the principles which predetermine the aspects of applicability of the life cycle of a game authoring tool development. These guidelines can serve as evaluation for other game authoring tools or any educational systems. From the research, these usability guidelines are in many aspects similar to existing usability evaluations with adults. But there are still a few differences that need to be solved in the context of children's comfortable feelings when using the software and having fun with it. First of all, different children have different requirements and needs associated with their age, education level, capabilities and knowledge. Second, different types of educational software have different learning approaches and methods such as education, training, practical, theoretical, abstract or concrete knowledge. Therefore, different lists of usability guidelines are required to cater to these issues. Following these guidelines may doubtlessly avoid many common errors and minimize harmful effects on the software. The existing usability guidelines discussed in Chapter 2 were mainly established after testing the respective software that has been designed [Resnick05][Begel97]. But in this research, compared to other common approaches, the usability guidelines were established throughout the whole software development phase. It is reasonable to look at the initial requirements of usability guidelines before starting to design the software instead of putting the usability guidelines at the end of the design process. In addition to these guidelines, Table 7.1 describes some broader lessons about children and learning aspects, including the interaction design and the game design perspective.

Table 7.1 shows lesson learned throughout the research and its description.

Lessons	Description
Design is a key determinant in building trust with children.	For motivated users of a constructive software tools, a bad design (busy layout, small print, too much text) hurts more than a good design helps. The application of making games gives children the power to modify and extend any aspects of the game user interface and allows them to share those modifications.

Lessons	Description
Promising application for improving 'what' and 'how' children learn.	The idea of *what* and *how* children learn is separated, because the application cannot only help children to learn, but also help them to learn better. The question *what* eventually addresses the problem of making and designing a game and enables the children to resolve it. On the other hand, the question *how* addresses the problem of enhancing the things that children had already expected to learn.
Layout of the interface may not influence performance, but it does influence satisfaction.	Regarding the interfaces of children's software, it needs to be considered that children may not yet understand abstract concepts. The children seemed to be comfortable with direct manipulation interfaces where the actions were more directly mapped to the actions on the screen. If other styles are adapted, the children require training.
Children are exposed to computer games at a much younger age than their older siblings.	At different ages, the relation of children towards interactive technologies varies reflecting their interests, contexts and attitudes. Games are perhaps the most controversial area of programming for children. By including children as game designers, they can relate to characters and actions similar to their age, gender and social interaction status. Although children are inherently dependent on others, they are empowered when they feel in control of their environment.
Experience matters.	The children's experiences are closely related to their mental model. For example, the icon design in children's software should be similar to the children's level of familiarity. Children as design partners in the development of children's software could have positive impacts on the ease and consistency of the software [Guha04]. Furthermore, children can process pictures faster than text, which offers significant advantages.
Rules of thumbs for icons.	Create them as large as possible, place frequently used icons in a persistent task bar, and arrange them either in a square (first choice) or in a horizontal layout. Applications for children should be designed to teach children a variety of skills such as the visual association of images, shapes and patterns, cause-and-effect relations and learning how to program.
Drag-and-drop programming style is still convenient for children.	Children interacting with conventional drag-and-drop programming styles were faster in understanding how to program and design games. This shows that children use their perception ability to recognize and understand the way of dragging and dropping scripts and game characters rather than reading and memorizing syntax or scripts [Yatim08].
At all age levels boys and girls show similarities and differences in choosing and using technology.	Although gender was not of interest when carrying out the research, it was not possible to ignore its influence in analyzing the results. Even though boys play more often and regularly than girls [Gorriz00][Thomas00], there is a significant difference in the kind of games they like to design [Cassell98]. Girls usually show an interest in designing beautiful games with social relations, while boys like to show their technical interests and skills in designing games [Yatim06b].

One question remains: *What do children learn when designing games*? Most of the children participating in this research stated that they had learned how to design a game. Many of them deemed they had learned how to better use the computer or application software. They also liked to work together and help each other to achieve their intended design. Although this was not necessarily a goal in this research, it was yet an unintended benefit. On the other hand, some children found it difficult to work in groups. This was due to the fact that children in a certain age range always wanted their ideas to be used by others. However, the children did not appear to have strong arguments on this matter. The children's game

design activities can be classified into three domains of educational activities: cognitive, psychomotor and affective matters. These three domains can be implemented in the game design activities due to the nature of the activities themselves. In the *cognitive* domain, the game design and game programming contributes to the aspect of how to create intended games. The children can learn how to specify, to code and test the games they designed, either by communicating with their educators or with other children. This can embrace their way of thinking and lead to more discovery learning. In the *psychomotor* domain, the children spend most of the time designing games by using the computer and intended software. Here, they can directly interact with the hardware and software, including experiences and skills in handling and using the mouse and other peripherals. In the *affective* domain, the children will learn about cause and effect involved in their decision-making. This research has also given details about the environment that might be used in schools to avoid children getting disengaged from seeking knowledge. Consequently, a fun environment would help to attract children and keep them interested in learning. On the other hand, children can spend most of the time with construction activities to prepare for their future. This is important, since ICT has changed the way how children learn, play, socialize and participate in their life.

7.2 Discussions

One of the major difficulties in the field of children and programming is to express the intention of the programming language itself. Yet, the programming activity intends to reinforce creative thinking and problem-solving skills that nowadays become more important [Smith96]. Although children are no programmers, they still have the desire to create and develop their own programs; especially those that have to do with games and simulations. With a suitable environment, the Gatelock bears a great potential to engage children in learning and enjoying their interaction with technology. Through constructive learning theories, children learn while actively being engaged in problem-solving activities [Kafai98]. Constructivists have noted that, instead of embedding the educational part directly into the games, children can learn more and better by creating the games themselves [Kafai06]. The opportunity to construct their own games will facilitate their motivation to design and develop games, and eventually learn how to program. In game making activities, the children do not only act as players, but also as game producers. Furthermore, they begin to develop computer literacy and knowledge about software tools, and they develop new and creative ways of thinking.

Typically, children can polish their creative thinking and problem-solving skills with the help of appropriate tools. But developing suitable tools for children is not easy. Nearly all commercial and powerful programming environments are clearly inappropriate because they have been designed for formal learning rather than for exploration and discovery learning [Scheider96] and are far too complicated for children [Soloway86][Repenning93]. Therefore, authoring activities are more suitable for children compared to natural programming [Yatim09c]. Although these tools focus on simplification rather than complexity, as Tholander noted, hidden complexity means hiding learning possibilities from children [Tholander02]. Programming is a perfect exercise for the brain. The programming skills will enable children to acquire new knowledge when learning simple series of instructions and computer programs which include the design of games. This research approach showed multiple sources of ideas of children's behavior by analyzing the children and also the tasks given to them during the iterative design of the game authoring tool.

Again, the approach has created an opportunity for clinical innovation and endorsed a method to study children in the field of interaction design. It also challenged the theoretical assumptions in this field, particularly in CCD. Furthermore, the approach acted as tentative support for the child psychology theory and the constructivist learning theory. Finally, the approach was in accordance with the study of children using the game authoring tool.

However, the approach also had some identified limitations. There was a problem of generalizing the conclusions based on a few subjects chosen as samples of the studies. The difficulties of drawing the cause and effect conclusion also had been faced in several studies. Furthermore, the possible biases in data collection and in interpretation were treated as limitation of the approaches. Another limitation of this approach is the level of children participating in the software evaluation. The children were brought into the process as evaluators. Additionally, they acted as informants who had an impact on the tool from the beginning of the development process. Here, the researcher still had the main control, since the researcher decided when to involve each child. But if the role of a design partner is given to the children, they have the same stake in the entire design and research process. When the children were directly involved in the research without any interference by adults, they usually slowed down the development process [Montemayor00]. The slow-down was due to the discovery of better ways of doing things, or new ideas/features that should be included in the technology because the children were able to recognize things that adults did not notice.

As the approaches used in this research are more about designing a game authoring tool iteratively within several usability evaluations, one danger of these ongoing evaluation studies is to get caught in a loop of refinement after refinement. To avoid this, it must be ensured that a list of project goals and specifications will be achieved. This helps to determine which features and interface problems really need to be fixed before releasing the tool to the public. A possibility would be to reuse the most successful features in a new project or to develop a new version. However, these aspects offer a great opportunity to correct any problems and to overcome the initial learning curve, since most of the technological aspects to design the tool have been familiarized and experienced during the planning and implementation process. After being involved in the research using different methods and techniques to evaluate the game authoring tool with children, it is up to the researcher and the development team to choose the role of the participating children. Depending on the team's philosophy, resources and time constraints, different roles make more sense. Educational researchers may prefer a role as user or tester, because observations and evaluations are much easier. If the children are involved as informants or design partners, the researchers should be prepared for longer development times. If children act as testers, they need to be users in order to test the designed tool. Both, user and tester offer feedback to the designers, either directly or through observations by researchers about the usefulness, effectiveness and correctness of the tool. The major difference is that the users are testing the tool after it is completed and unchangeable, while testers use intermediate pre-released prototypes as well as final products.

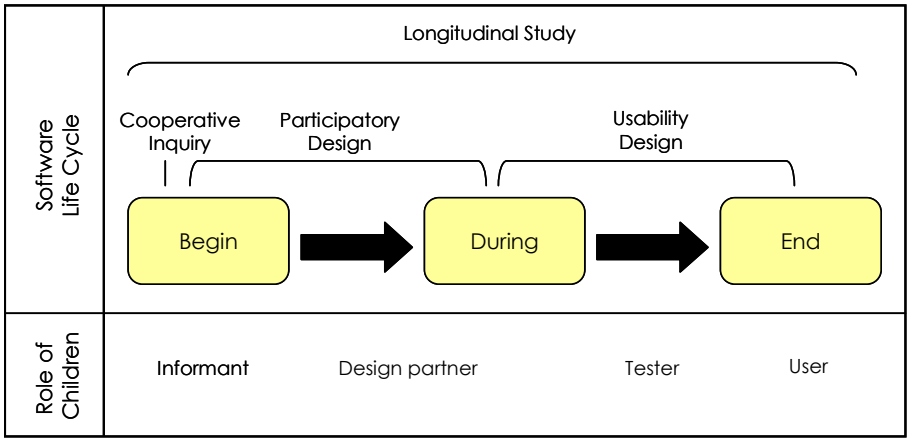


Figure 7.1 Role of children according to different phases in the software development life cycle and the suggested usability methods.

By looking at the role of children in evaluating software (as shown in Figure 7.1) the limitations mentioned above can be solved or reduced. The figure is based on the four roles that children play in the design process, as discussed in Chapter 2 [Druin99a]. The children can keep these roles throughout the whole software development life cycle, especially with regard to the usability methods called cooperative inquiry, participatory design and usability evaluation. All of these methods can be used and exploited with appropriate research resources such as time and money, human resources such as researchers and adequate respondents (in terms of number and availability), as well as related individuals such as teachers and parents. Therefore, a longitudinal study with iterative tests is suggested. Conducting a long-term study of more than 6 months can help researchers to receive more findings and results of a particular treatment with the application software involved [Kafai98b]. The longitudinal study of the effects of game authoring and programming can show and reveal children's potential cognitive abilities and learning achievements. Nevertheless, the longitudinal study can only be carried out if resources such as time, money and human work force are available and manageable during the research period. The following section summarizes the discussions of each chapter to recap the whole research.

Chapter 2 discussed the usability aspects of children's software in terms of the relationship between programming for children and the measurement of its usability. Furthermore, the relationship between usability and games was described, including the aspects of fun and flow theory. Since children often use technology to perform certain tasks, usability and capability to fulfill the purposes are crucial. In order to understand the premise that the elements of enjoyment are universal, especially in nurturing children's skills on how to program, the chapter touched the importance of an educational programming language for children for the further design plan of an ideal game authoring tool whose usability will be measured. Two major topics discussed in the chapter dealt with design for children in age-specific interaction styles (such as how to structure menus, the size of the screen objects, fonts, the suitability of input devices, and so on) and the involvement of children in the design process.

Chapter 3 illustrated the importance of knowing the real users—children aged from seven to twelve. The selected children were involved in a preliminary study of two tools called Squeak Etoys and Game Maker. Squeak Etoys represented a simulation authoring tool while Game Maker represented a game authoring tool. Findings from the study were used as benchmarks for designing the prototype of a game authoring tool within the frame of this research along with a refinement of the usability guidelines.

Chapter 4 introduced the implementation of the theoretical model approach for designing a game authoring tool for children from seven to twelve. It was described based on the following three interface elements: (1) an educationally appropriate interface representation, (2) an interaction protocol that naturally shifts children's attention from intuitive interaction to focusing on the structure and operation of the representation, and (3) a gradual elimination of representation components so that children are required to take over cognitive responsibilities. In detail, the chapter reported on the implementation of the game authoring tool called Gatelock. The implementation was based on the prototyping methodology with the integration of the three main development tools Squeak, Tweak and iEngine. The chapter ended with an example of a simple game designed by a group of children.

Chapter 5 showed the findings from two field studies, the formative and the summative evaluation. Both evaluation studies were started with pilot studies to test the instruments first before using them for the real studies. Basically, the pilot studies helped to identify minor problems of the instruments used and became important for conducting the evaluation studies. The formative and summative evaluation also brought positive input to the design and development of Gatelock. The input was treated as main player in designing the usability guidelines. Again, the usability guidelines were refined in order to grasp important facts upon designing a game authoring tool for children. The refinements of the usability guidelines were described in the next chapter together with a thorough discussion on Gatelock. This includes an explanation of the achievements of Gatelock throughout this research, and also its limitations.

Chapter 6 addressed the achievements and limitations of Gatelock based on outcomes from several studies conducted throughout the research. It also explained the capabilities that Gatelock has, and the ones that it does not have. Furthermore, some suggestions to improve Gatelock were described. On top of that, this chapter made some reflections on the usability guidelines iteratively developed throughout the research. This chapter ends with a description of the lessons learned when conducting usability studies with children.

7.3 Main Contributions of the Research

The research was expected to deliver both a product-based (a game authoring tool and evaluation instruments) and a process-based approach (a set of usability guidelines) for those who have an interest in developing a similar game authoring tool for children that focuses on fun in a learning environment. The results from this research contribute to usability design, interaction design, and a game authoring tool for children. More important, through this study, the environment of the game authoring tool conforms to children's needs and desires by using a set of usability guidelines throughout the whole development process. Finally, the outcomes are technologies, applications and guidelines aiming at

supporting children's creativity and learning. By integrating game design, game programming and playing activities, the specific research contributions are:

a) A preliminary study of creating a new game authoring tool with usability as the first objective.

The research started by looking at and surveying existing tools such as Alice and Stagecast Creator or similar tools such as ComiKit and ToonTalk to get an overview of what to expect when developing a game authoring tool. Another stage was to conduct a preliminary study using other existing tools such as Game Maker and Squeak Etoys which offers a great opportunity to study for children, especially in terms of their learning experiences and their understanding of programming using different types of software and tools. Most of these tools can be found in the market, and some of them are free such as Game Maker and Alice, while others fall into the category of simulation tools but can also be used for designing simple games such as Scratch and Comikit. A thorough survey on certain aspects such as programming styles, features and capabilities could reveal ideas and solutions for certain aspects before designing the intended software.

b) Empirical evidences characterizing the preliminary study with results developed by children, which can be used to help designers generate and select the appropriate authoring features.

Conducting a preliminary study with children can identify what they really need and means to be independent of what the designers actually want. The study was conducted with two software tools for children and upon using them, the children were asked questions about the features of the tool they liked or disliked. The findings were treated as valuable information towards designing the prototype of a game authoring tool. The main findings were: (1) children with better computer experience had a higher ability to learn game programming, (2) children preferred a game authoring tool compared to a simulation tool, due to its easier handling and usage, (3) there are no different gender preferences when designing games, (4) there are differences between the usability guidelines for game authoring software and other children's software, (5) understandable metaphors, languages, tools, design environments and instructions, (6) a direct manipulation interactivity between the tools and the children, including instant feedback and assistance, (7) a need for powerful ideas in making the software fun for children, (8) functionality tools that have a variety of resources supporting graphics, animations and controlling of objects that are easy to use and understand by children, and 9) a need to combine a uniform interface and programming metaphors from the children's point of view.

c) A game authoring tool prototype with a unique set of features due to its child-centered design.

The iterative design and development of the prototype throughout this research brought many benefits. Due to its child-centered design, the prototype of the game authoring tool has a unique set of features. The prototyping method addressed the problem of communication between designers and children, getting children's acceptance or even managing change requests, delivering early proof of a game authoring tool concept and also increasing the constructive participation of children. The most important fact is that the prototype be can easily changed or even discarded, but sometimes the prototype can also encourage an excess of change requests. Engagement, participation and involvement of children are related to each other and are required for the CCD approach. Instead of being passive, children can be more engaged and provide requirements for the functionality of the game authoring tool. In

addition, this approach can demonstrate a high level of usability to both children and designers. Children have a decision-making power and appropriate mechanisms for communication or even negotiation with the designers. The iterative and evolutionary prototype designed can be considered as a socio-technical design process where priority is given to the user interface design. Here, designers are enabled to react to changing needs in functionality, and to manage and control the development and evaluation of the game authoring tool prototype.

d) Formative and summative studies demonstrating how to iteratively design and create usability guidelines for a game authoring tool.

Implementing both formative and summative evaluation studies can cause major changes in the design of the game authoring tool. The approach is helpful in several aspects such as providing a rich set of data to be used to plan changes to the game authoring tool and to highlight short-term and long-term effects. The work of both evaluation studies is necessary, but also expensive and requires a careful planning, especially in terms of time and resource management. The results from these studies can be used to help designers generate and select the authoring features. The studies also showed several aspects of the usefulness of methods, tools and techniques used for the evaluation process with children. The formative and summative studies that are demonstrating the usability of the game authoring tool have improved the design of it. The underlying logic of this approach is that, if an accurate application of a game authoring tool is created, any weaknesses that are found in the application may reflect weaknesses in the model.

e) Empirical evidences of the usefulness of methods, tools and techniques used for the evaluation process with children.

For both formative and summative evaluation, each evaluation unveiled different aspects of the software that the developers did not discover. Both identified usability flaws within the evaluations that were revealed by children's views. The combined techniques form a valuable tool to make the correct decisions concerning the design of a game authoring tool for children to be used in the classroom. Different methods, tools and techniques have been carried out during different user studies, which help to understand the different practices of the methods and their effectiveness. Each of them was conducted in a different way, according to the design aim, context, and age of the children. This includes the usage of qualitative research (such as observation, survey and questionnaires), the creation of data gathering instruments (such as *Pick-and-Paste*, *Select-and-Scroll* and the refinement of the Smileyometer adapted from other researchers), as well as implementation procedures and protocols carried out during the studies. The main concept behind this method was to involve the children and to use their outputs for the following design sessions.

f) A list of usability guidelines for designing a game authoring tool for children.

To create applications that are good, effective and highly usable in terms of the interface design, the involvement of users is strongly required to develop children's software. It is necessary to design a game authoring tool by which children can express their ideas more easily. Overall, the main contributions are the list of usability guidelines, as simplified and discussed in Chapter 5. The list was also involved in several refinements and it was

iteratively changed based on the inputs from several studies conducted throughout the prototype design and development phases.

7.4 Future Work Perspectives

Based on this research and the studies conducted, this section will describe further ideas on the development and improvement of Gatelock. Future work will concentrate on finding solutions for problems that can be solved by implementing several suggestions and proposals. This research will continue on two fronts: mirroring the structure of this thesis to research about new technologies, and research about new applications and ways of learning. These two activities are intertwined and interdependent; a new application idea can lead to a new technology, and vice versa. Furthermore, the research presented has led to a number of open questions which are interesting to be addressed in future work.

First it has to be looked at several controversial findings. The need for having a tool to structure and support the integration of game play and learning requires the designers of the game authoring tool to master pedagogy [Hinostroza00], and educators to master game design and development. To design a game authoring tool for children, it is not sufficient to involve only the children but also the teachers. Since CCD aims to give a voice to children during the process of designing a game authoring tool, teachers should also be introduced into the design, especially with those tools that are going to be used in the classroom [Anning94]. Teachers can contribute to the scheduling of the design sessions, select participating children, assist the researchers in forming groups of children (if required), evaluate the potential of educational tools in practice and, based on their experiences, help to identify learning goals, and identify the children's difficulties and capabilities [Kafai98][Robertson02][Pardo05]. For future work, the game authoring tool can be tested and evaluated on a good platform for children and teachers to perform computational activities especially in the classroom.

Second involves the design of a collection of activities for computer systems so that learners engaged in these activities have to achieve given educational goals. The measure of success is not considered to be simple, but a more complex evaluation of the effects of performing the given tasks [Zaman05b]. For example, if children complete an educational activity given quickly without value, they will not learn anything from it. By using the game authoring tool, it is emphasized that for effective technology integration teaching the curricular content is more important than using the technology. On the other hand, the use of technology should support the teaching of contents and make learning more meaningful for students [Guzdial00]. Hopefully, the game authoring tool can be used with lessons that are already included in an integrated curriculum guide, or it can be used as a tool for the design of new lessons. Giving the children a project-based design activity for classroom usage will eventually integrate the game design activity as one of the teaching tools for teachers.

Third involves the nature of the game authoring tool itself. A game authoring tool has different design considerations and usability issues than other types of software. Regarding the usability of a game authoring tool, the elements of effectiveness, efficiency and satisfaction must be accompanied with the element of fun. The usability test was conducted to get some input from the children and it revealed

some interface problems. These findings can be used by the designers of the tool to create a better and acceptable game authoring tool for the actual users, the children. The usage of programming language to design the game authoring tool including the game engine can be expanded and its scalability can be tested in future works, such as an implementation into 3D environment or Web-based game authoring environments.

Forth involves the research approach itself. The problem of generalizing the conclusions based on a few subjects chosen as samples of the studies caused difficulties in drawing the conclusions. Furthermore, it effected the duration of the evaluation studies that needed to be longitudinal in order to get more detailed and precise findings. The idea was to explore the game design activities of children from seven to twelve. To measure the learning effects, more time and a larger number of participating children are needed and are suggested for future work. For this purpose, the study can adopt a quantitative research approach for getting measurable data. At the end it will be verified if the game design activity is suitable for learning purposes, which requires an accuracy of content delivery and an adaptation to the curriculum.

Hopefully, by embedding computational power in game design activities, in the future children will be provided with new ways of connecting to scientific and mathematical concepts. Furthermore, it is important for the next generation of children to gain a sense of control, ownership, and empowerment, and to become actively involved in understanding and designing own creations. The game authoring tool should enable children to become authors of rich and amazing technological artifacts. On top of that, the game authoring tool can make use of today's amazing and upcoming technological gadgets. Numerous research started to look upon the reliability and functionality of these innovative technologies such as the tabletop [Scott02], electronic whiteboard [Ashfield03], tangible interaction devices [Wyeth06][Horn07] and pen interaction [Read07]. Children can better appreciate the magic of technology when they do not only use it, but also create it. In the current game industry gamers are given the opportunity to be involved in designing games. The game design activity by the end users is known as *game modding* or modifying off-the-shelf games in the market such as Neverwinter Nights and Half-Life [El-Nasr06][Prensky06]. To mod a game means the modification of a game that changes its properties and allows the gamers to customize their games with little or no programming experiences. The modification of these games can be carried out by mod making tools such as the Auroro Toolset for Neverwinter Nights. The future of modding games is well-accepted by gamers, but the game industry faces a tough decision of either accepting the idea of allowing gamers to mod their games or enclosing their games with copyright policies due to the ownership of the games. It is the right time to encourage children to design games so they can gain experiences and knowledge in modding them or even designing their own games in the future. This will dynamically broaden the field of end-user computing.

Research on the impact of introducing inappropriate software for children has generally been scarce due to the relatively early adoption of the curricula [Said04]. Studies have begun to concentrate on the training of younger generations and teachers to shed some light on the implementation of technology courses particularly focusing on pedagogical and socio-cultural aspects [Yatim06a] [Yatim07a]. Currently, public schools in Malaysia are provided with computers either in a specialized computer lab or in the classroom. The schools are steadily and increasingly accessing new technologies. The teachers are advised to use the

computer technology as latest pedagogical instruction. In fact, the Ministry of Education (MOE) of Malaysia has implemented a new educational reformation to revise the current curriculum due to the convergence of Information and Communication Technology [MOE09]. The Education Technology Department under MOE is also responsible for designing and providing appropriate software for children and for training teachers by using these software technologies. One of the critical knowledge domains that attracts a lot of attention is computer literacy of children in the future. Their computer skills must be used and enhanced by providing the right resources for teaching and learning. Therefore, a lot of future work has to be carried out, especially in Malaysia.

The first task is to use and implement the game authoring tool in selected schools for a longer research period. The experience of creating games in the school can enable children to develop a deeper, more systematic, and critical understanding of this medium. It can also enable them to become creative users in a game culture, constructing imaginative representations and expressing their own creativity. The second step is the evaluation of the game authoring tool in schools. This involves a quantitative approach and a longitudinal study to get more precise results, including the implementation of pre- and post-evaluation studies. Third, the usage of the game authoring tool as well as the practicality and the involvement of teachers that are important in the classroom shall be explored. When introducing new technology to teachers, they sometimes do not feel entirely comfortable with using this technology. Nevertheless, teachers are also curious and eager to learn how to use new technologies and how to apply them to their teaching [Yatim07a]. On top of that, teachers are also willing to join the design process of new technology to be used in the classroom [Druin02b], or to be involved in usability testing to contribute to the research. Last but not least, this research aims to show that a list of usability guidelines is involved in designing and developing a game authoring tool, and that this list gradually changes due to the findings of the evaluation studies. Considering these points, the research was able to implement the concept of having an ongoing project with outcomes towards the theoretical aspects of HCI in general and CCI in particular. To bring it to the point, more work still needs to be done.

7.5 Personal Remarks

Playing interesting and attractive games has become an everyday activity for children of all ages. While there is a lot of controversy surrounding computer games and their negative impacts on the society, I do believe that games can actually be educative, beneficial and helpful for the learning and development of children. The need to know how computer games can be beneficial for children has become my main commitment throughout this research. Before the research started, I was highly interested in children's software and its evaluation process. I mainly wanted to concentrate on the usability of children's software used in the classroom and taught some related courses at the Sultan Idris University of Education in Malaysia. During this research, I quickly decided to integrate game design by using a specific tool for children in the classroom. Fortunately, Marco Schmidt and Sergei Roth helped me a lot with the implementation of these ideas and they always inspired me by sharing their experiences. The result of this work can be considered as a novel approach for the process of learning by doing and learning by thinking about what children do, or in other words, making games in a fun and creative game authoring tool environment. On the other hand, the work also contributed to find usability guidelines for developing a game authoring tool for children. Of course, the implementation was not entirely faultless.

Although through learning and involving students in various projects dealing with Squeak during the development, many obstacles and problems had to be faced. With numerous tests, the research received positive and negative opinions and comments from many people. But this is the true challenge that must be faced. On top of that, the work has been presented and published in several conference proceedings, posters and demonstrations.

I think that the idea of developing the game authoring is the right approach supplementing the constructivist approach of learning by doing. Younger children have the opportunity to quickly create simple games. Through the collection of experiences within the game authoring tool environment, more content modules can be created; thus, more games can be created using the content elements. Even after finishing the research, I will continue to work with Gatelock including fixing and adding missing features and continuously testing the tool with the actual users, the children. Upon designing the usability guidelines for a game authoring tool, there is no doubt that this approach takes practice and patience. On the other hand, by following these usability guidelines, many common errors can be avoided, and this will fortunately minimize harmful effects and frustrations.

Appendix A

Instrument used in the preliminary study

This section shows instrument used for the preliminary study which is:

A1. Questionnaire

Questionnaire:
Children Basic Information and Game Preferences

The following questions will ask on children's experience in using two (2) programming environments –Squeak Etoys and Game Maker. This is not a test. We want to learn about children's perception and their impression on using these tools. The answer will be kept private and your name will never be used. Please be as honest as you can. Please tick (✓) at the appropriate box.

Section 1: Personal Information

1. What grade are you in? _____

2. How old are you? _____ years old

3. Are you a boy or a girl?
   ☐ Boy ☐ Girl

Section 2: Game Preferences

1. Have you ever played video games?
   ☐ Yes ☐ No

2. How interested are you in playing games?
   ☐ Not Interested
   ☐ Interested
   ☐ Very Interested

3. Which of the following you use to play games? (You can tick more than once)
   ☐ PC / Computer
   ☐ Nintendo
   ☐ PS2
   ☐ GameBoy
   ☐ Others (Please specified): _____
   ☐ Sega
   ☐ PS1
   ☐ GameCube
   ☐ Xbox

4. Last month, how many hours did you spend playing video games?
   ☐ None
   ☐ Less than 5
   ☐ 5 – 10
   ☐ More than 10

5. Have you ever played Online games?
   ☐ Yes ☐ No

6. Last month, how many hours did you spend playing online games?

- ☐ None
- ☐ Less than 5
- ☐ 5 – 10
- ☐ More than 10

7. How interested are you in the following genres of video games? Please tick (✓).

Genre	Not Interested … Really Enjoy				
	1	2	3	4	5
Action (e.g. Tomb Raider/Doom/Half-Life)					
Driving (e.g. Gran Turismo/Formula 1)					
Sports (e.g. Football/Olympics)					
RPG (e.g. Everquest/Raknarog)					
Strategy (e.g. Chess / Sim City)					
Flight Simulation / Emulators					

8. What game do you play most now?

_____

9. What game have you played the most?

_____

10. Name five (5) of your favourite games.

1. _____
2. _____
3. _____
4. _____
5. _____

Section 3: Programming Tools Please tick (✓).

Items	Squeak Etoys			Game Maker		
	☹	😐	☺	☹	😐	☺
Ease of use						
Understand the options/buttons						
Structured visual information						
Text in addition to auditory info						
No reading required						
A variety of activities available						
Ability to do alone						
Ability to do with another person						
Easy to understand the scripts						
Drawing features						
Easy to use mouse						

Section 4: Your Suggestion

4. Would you like to learn more? Please tick (✓).

Squeak Etoys			Game Maker		
☹	😐	☺	☹	😐	☺

4. Give your opinion on these game programming environments. Write your opinions.

Squeak Etoys	Game Maker

4. Write your opinions. What do you dislike about:

Squeak Etoys	
Game Maker	

4. Which is the best?

☐ Yes ☐ No

If yes, why?

If no, why?

~ End of Questionnaire ~

Appendix B

Instruments used in the formative evaluation study

This section shows instruments involved in the formative evaluation study which includes:

B1. Questionnaire
B2. Icons and Their Meaning
B3. Pick-and-Paste
B4. Select-and-Scroll
B5. Observation Check Sheet

Questionnaire:

Children Basic Information and Game Design Preferences

The following questions will ask about children's basic information and the game design preferences prior to the usage of a game authoring tool named Gatelock. This is not a test. The answer will be kept private and your name will never be used. Please be as honest as you can. Please tick (✓) at the appropriate box.

Section 1: Personal Information

1. Are you a boy or a girl?
   - ☐ Boy
   - ☐ Girl

Section 2: Game Playing Information

2. Have you ever played any computer/video games?
   - ☐ Yes
   - ☐ No

2. How interested are you in playing games?
   - ☐ Not Interested
   - ☐ Interested
   - ☐ Very Interested

3. Which of the following medium you use to play computer/video games? (You can tick more than one)
   - ☐ PC / Computer
   - ☐ Nintendo / GameBoy
   - ☐ Others (Please specify): _____
   - ☐ XBox / Xbox 360
   - ☐ Sony PS1 / PS2 / PS3 / PSP

4. Last month, how many hours did you spend playing computer/video games at a time?
   - ☐ Up to 30 minutes
   - ☐ Up to 1 hour
   - ☐ Up to 2 hours
   - ☐ More than 2 hours

5. Last month, how many times per week do you play computer/video games?
   - ☐ Once a week
   - ☐ 2-3 times per week
   - ☐ 4-6 times per week
   - ☐ Every day

6. Have you ever played online games?
   - ☐ Yes
   - ☐ No

7. Do you play video/computer games at school?

☐ Yes ☐ No

8. Do you play games with other members of your family?

☐ Mother ☐ Father
☐ Older brother ☐ Younger brother
☐ Older sister ☐ Younger sister
☐ Others (Please specify): _____

9. What do you play? Please tick (✓).

Genre	Not Interested … Really Enjoy				
	1	2	3	4	5
Action (e.g. Tomb Raider/Doom/Half-Life)					
Driving (e.g. Gran Turismo/Need for Speed)					
Sports (e.g. FIFA/WWF Smackdown)					
RPG (e.g. Everquest/Raknarog)					
Strategy (e.g. Chess / Sim City)					
Flight Simulation (e.g. Blazing Angels)					

10. Name three (3) of your favourite games.

1 _____
2 _____
3 _____

11. Identify the features of your favourite games that you like best.

☐ Graphic
☐ Colour
☐ Characters
☐ Fun
☐ Challenge
☐ Music
☐ Story
☐ Others (Please specify): _____

12. What do you learn through game playing?

☐ Working as a team
☐ Decision making
☐ Planning
☐ School based learning
☐ Spelling
☐ Reading
☐ ICT literacy
☐ Others (Please specify): _____

13. Do you want to develop a game? ☐ Yes ☐ No

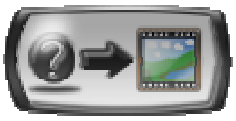
Why?

~ End of Questionnaire ~

Match the Icons with Their Meaning:

Padankan ikon dengan maksud yang bersesuaian.

	O	O	Reject *Salah atau tidak diterima*
	O	O	Draw *Lakar*
	O	O	Design *Rekabentuk*
	O	O	Audio, Sound *Bunyi*
	O	O	Play/Test Game *Main/Uji permainan*
	O	O	Home *Kembali ke menu utama*
	O	O	Yes *Betul*
	O	O	Game Content *Kandungan permainan*
	O	O	On/Off *Tutup/Buka*

	O	O	Collision *Perlanggaran berlaku*
	O	O	Use AWSD keys, space bar to play *Main guna kekunci AWSD, space bar.*
	O	O	Built new object *Bina objek baru*
	O	O	Background *Latarbelakang*
	O	O	Use arrow keys & space bar to play *Main guna kekunci anak panah, spacebar.*
	O	O	New game level *Tahap permainan baru*
	O	O	Insert game level *Selit tahap permainan*
	O	O	Delete game level *Buang tahap permainan*
	O	O	Undo *Undur*
	O	O	Switch game levels *Tukar tahap permainan*

~ End of Assessment / *Tamat* ~

Pick-and-Paste:

Instruments used to indicate children feedbacks on usabilityaspects.

GATELOCK - Game Authoring Tool for Children

Paste the right Smileyometer stickers that indicate your truth feeling and opinion by referring to the questions below:

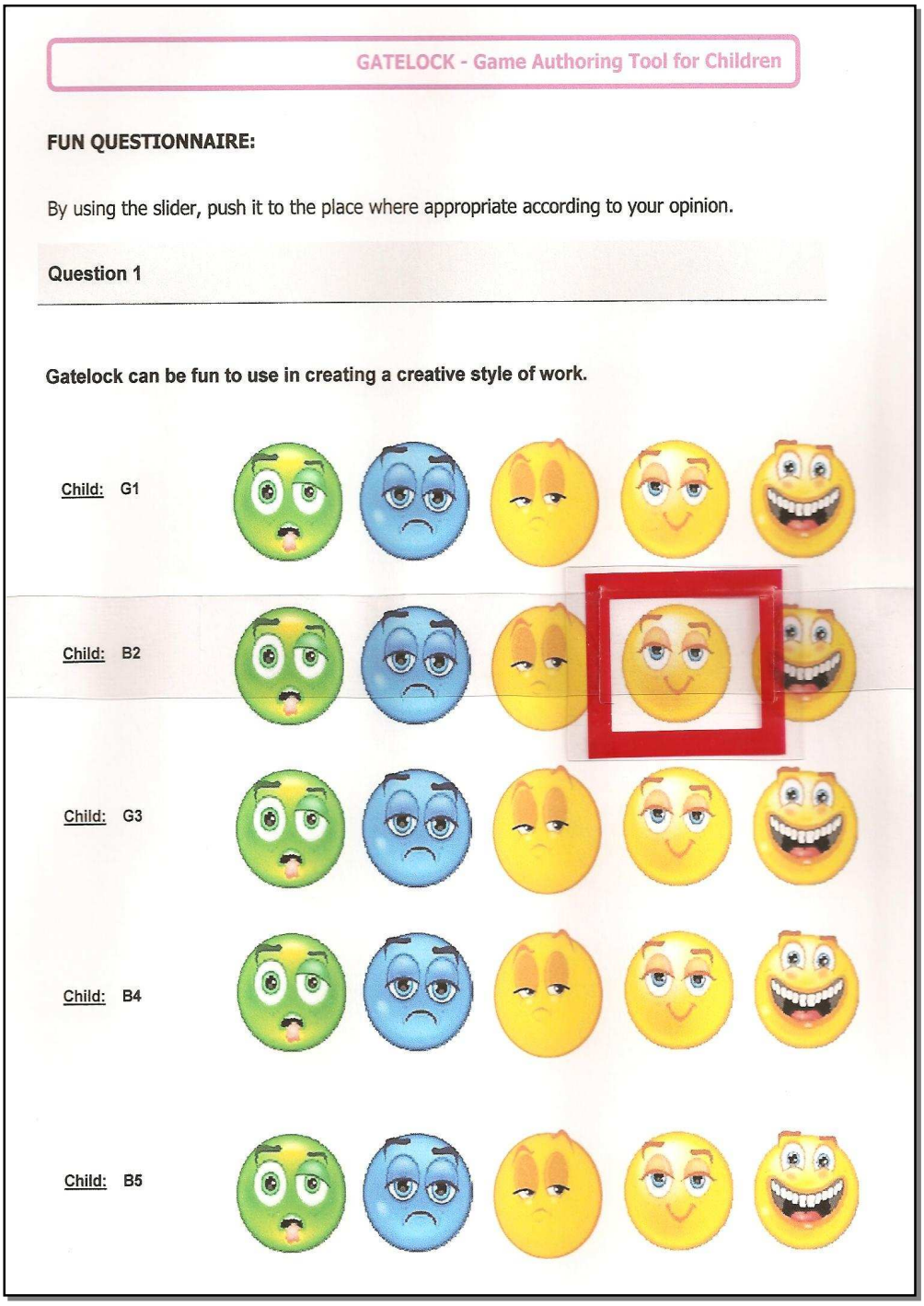
Items	G1	B2	G3	B4	B5
Structured organization of information					
Easy to use to perform tasks					
Understand information on screen					
Sequence of screen					
Messages in screen					
Easy to understand the scripts					
Structured visual information					

Smileyometer

From left to right – awful, not very good, good, really good and brilliant.

Select-and-Scroll:

Instruments used to indicate children feedbacks on fun aspects.



Observation Check Sheet:

Instruments used to indicate children behavior and gestures.

GATELOCK - Game Authoring Tool for Children

Check Sheet (Observational Data)
AFFECTIVE & PSYCHOMOTOR SKILLS

Week	1 / 2 / 3
Date	1 / 2 / 3
Group	

Happened / Can be notified / Frequency occurrences.

Mark (X) at the appropriate indicator.	Start →								End
	5 min	15 min	30 min	45 min	60 min	75 min	90 min	105 min	120 min
Affective reactions (expressed frustration)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Laughter / fun	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Sense of engagement	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Express pride	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Verbal aggression (at the game/at other people)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Physical aggression (at the game /at other people)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Verbal communication (self/within group members)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Verbal communication (within other groups)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Self-Exploring and discovering	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Thinking beyond expectation	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Shows the ability to solve problems	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Ability to give ideas on 'how-to'	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Ability to remember information given by facilitator	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Ability to imitate facilitator's behaviour	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Relate game design with skills needed	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Relate game design with other subjects	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L

1

GATELOCK - Game Authoring Tool for Children

Happened / Can be notified / Frequency occurrences.

Mark (X) at the appropriate indicator.

	Start → End								
	5 min	**15 min**	**30 min**	**45 min**	**60 min**	**75 min**	**90 min**	**105 min**	**120 min**
Listen to others with respect	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Participates in group discussion	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Propose a plan to social improvement	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Accepts responsibility for one's behaviour	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Shows self-reliance when working independently	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Ability to use sensory cues to guide motor activity	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Knows and acts upon a sequence of steps in a process	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Adequacy of performance is achieved by practicing	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Performed tasks with confidence and proficiency.	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Skilful performance of motor acts	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Skills are well-developed (adaptation)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Create new patterns to fit a certain situation/problem	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Coordinating a series of actions (articulation)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Reactions that are not learned (reflex movements)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Has a value system that controls their behaviour	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L
Effective body language (gestures & facial expressions)	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L	H N L

[H N L (High-Normal-Low)]

Notes / Anecdote (any other aspects of child's behaviour/game use including problems encountered):

~ end of check sheet ~

Appendix C

Instruments used in the summative evaluation study

This section shows instruments used for the summative evaluation study which includes:

C1. List of Tasks

C2. Questionnaire

C3. Usability Checklist Form

List of Tasks:

A list of tasks to be completed.

User Observations and Associated Tasks and Requirements

		DIY	Guidance
1.	Open/Run the Gatelock.		
2.	Select existing game project named "test".		
3.	Delete game project named "test".		
4.	Select existing game project named "test2".		
5.	Play game project named "test2".		
6.	Stop the game and go back to main screen.		
7.	Create new project.		
8.	Rename new project to "Mein Spiel".		
9.	Add player items – item "fisch".		
10.	Resize item "fisch".		
11.	Duplicate item "fisch".		
12.	Re-locate the duplicate "fisch".		
13.	Add background "meer".		
14.	Define movement for item "fisch".		
15.	Item "fisch" will move to any direction using arrow keys.		
16.	Play the game.		
17.	Go back to development screen.		
18.	Add player items – item "pinguin".		
19.	Duplicate item "pinguin".		
20.	Delete one of the "pinguin".		
21.	Define movement for item "pinguin".		
22.	Item "pinguin" will move to any direction using arrow keys.		
23.	Play the game.		
24.	Go back to main screen.		
25.	Exit Gatelock.		

~ end of tasks ~

Questionnaire:

Instruments used to indicate children behavior and gestures.

AN EVALUATION OF GATELOCK: A GAME AUTHORING TOOL FOR CHILDREN

Section 1

Gender		Age	≤ 6	7	8	9
			10	11	12	≥ 13

Do you like to play video games?	✓	⊘

Why?

Section 2

To play games, I used...		Others (Please specify):

Do you ever play online games?	✓	⊘

Last month, how many hours did you spend playing games at a time?	30 minutes
	Up to 1 hour
	Up to 2 hours
	More than 2 hours

Last month, how many times in a week did you spend playing games?	Once a week
	2 – 3 times per week
	4 – 6 times per week
	Everyday

Section 3

Do you play games at school?	✓	⊘

Whom do you play games with?	Friends	Mother	Father	Others (Please specify):
	Sister	Brother	Uncle	

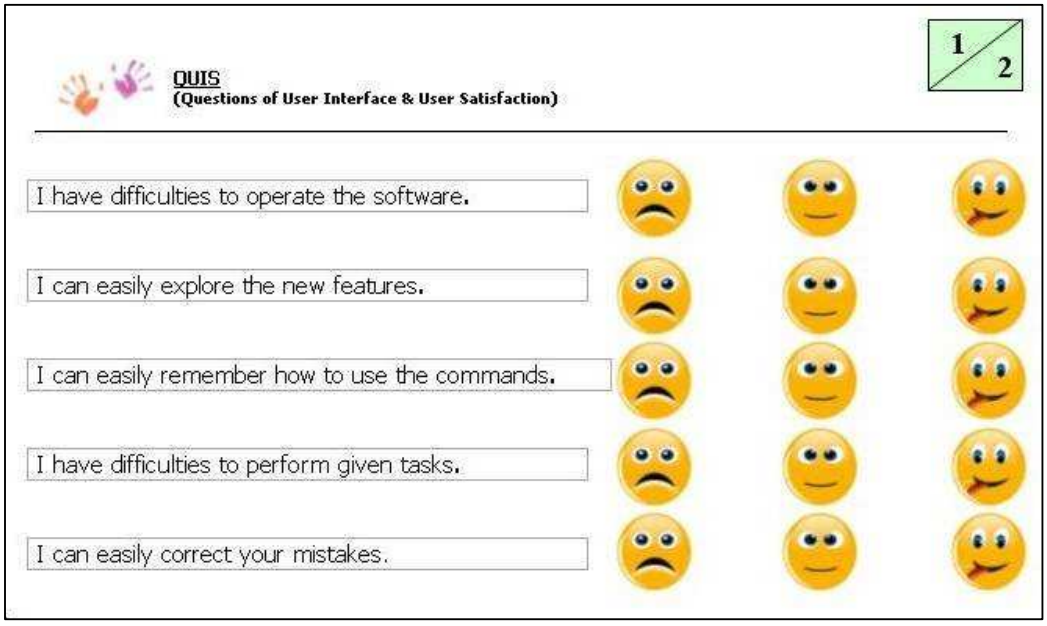
Which genre do you like to play?	Action	Driving	Sports	RPG	Strategy	Simulation	Shooting
							

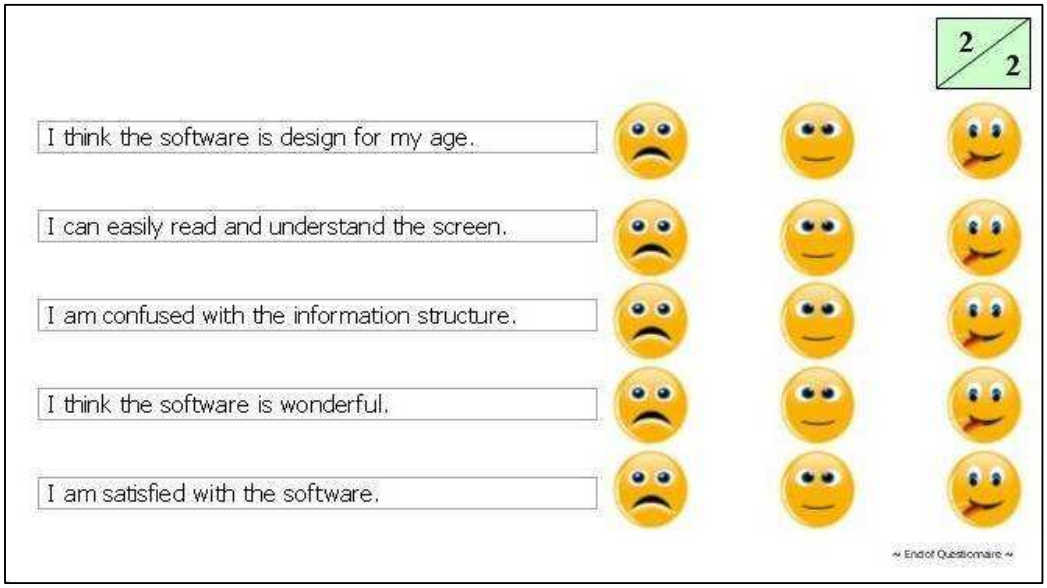
Others (Please specify):

Which features you like most? (You can circle more than one)	Graphic	Color	Challenge	Game characters
	Music	Story	Others (Please specify):	

Do you want to develop games?	✓	⊘

Why?


QUIS
(Questions of User Interface & User Satisfaction)
1/2
I have difficulties to operate the software.
I can easily explore the new features.
I can easily remember how to use the commands.
I have difficulties to perform given tasks.
I can easily correct your mistakes.


2/2
I think the software is design for my age.
I can easily read and understand the screen.
I am confused with the information structure.
I think the software is wonderful.
I am satisfied with the software.
~ End of Questionnaire ~

Usability Checklist Form:

Please tick the appropriate box.

☐ *Game authoring tool*
☐ *Other applications* .....

	Excellent	Good	OK	Could be Better *(Please tell us how)*
Consistency				
1. *Relevance*: The software is applicable and meets the children need to learn.	☐	☐	☐	☐
2. *Presentation:* Information, navigation and other features in the software are presented in a consistent manner.	☐	☐	☐	☐
Child-Friendliness				
3. *Curiosity*: The software arouse sensory and cognitive curiosity for provide opportunity to explore.	☐	☐	☐	☐
4. *Fantasy:* The software encourages the children to envision themselves in an imaginary context.	☐	☐	☐	☐
Degrees of Functionality				
5. *Attention*: The software provokes children attention with the use of color, sound, graphics, animation or other occasional surprise.	☐	☐	☐	☐
6. *Self-expression:* The software provides capability for the children to make their own artefacts.	☐	☐	☐	☐
Staying in the Flow				
7. *Reinforcement*: The software had fun features to engage the children in completion of activity and they found it enjoyable.	☐	☐	☐	☐
8. *Variety*: The software has various input and output modes such as mouse, keyboard, joysticks etc. to occupy the children with the design activity.	☐	☐	☐	☐
Reflection				
9. *Goal scoring*: The software clearly set learning or design goals and provides indication on how the children can proceed gradually.	☐	☐	☐	☐
10. *Satisfaction*: The software maintains child's satisfaction by giving supportive feedbacks and encouragement.	☐	☐	☐	☐
Familiar Conceptual Model				
11. *Control:* The software interface provides the children sufficient control and freedom to accomplish tasks.	☐	☐	☐	☐
12. *Recognition:* The software has familiar characters and gives the children a familiar way to locate and use information.	☐	☐	☐	☐
Familiar Way to Program				
13. *Confidence:* The software provides logical and reasonable thinking for children to solve design problems.	☐	☐	☐	☐
14. *Retain Information*: The software has features and functions that are easy to remember and memorize.	☐	☐	☐	☐
Game Playability				
15. *Challenge:* The games designed with the software can provide and design appropriate levels of challenge.	☐	☐	☐	☐
16. *Competition*: The games designed with the software have fair level of difficulties.	☐	☐	☐	☐

Please comment on how we could do better or what we have done well.

Thank you for helping us to evaluate our software performance.

References

[Abdullah08] Abdullah, N. A., Kamaruddin, R.H.R.A., Razak, Z., and Yusoff, Z.M. A Toolkit Design Framework for Authoring Multimedia Game-Oriented Educational Content. *In proceedings of the 8th IEEE International Conference on Advanced Learning Technologies*, pp. 144-145, 2008.

[Abras04] Abras, C., Maloney-Krichmar, D., and Preece, J. User-Centered Design. In W. Bainbridge (Ed.), *Encyclopedia of Human-Computer Interaction*, Thousand Oaks, Sage Publications, pp. 763-768, 2004.

[Ackermann09] Ackermann, E.K. Piaget's Constructivism, Papert's Constructionism: What's the Difference. http://learning.media.mit.edu/content/publications/EA.Piaget%20_%20Papert.pdf Last accessed: May 2009.

[Ahmad98] Ahmad, R. Educational Development and Reformation in Malaysia: Past, Present and Future. *Journal of Educational Administration*, Vol. 36 (5), pp. 462-475, 1998.

[Airey02] Airey, S., Plowman, D., Connolly, D., and Luckin, R. Rating Children's Enjoyment of Toys: Games and Media. *In proceedings of the 3rd World Congress of International Toy Research on Toys, Games and Media (ITRA),* London, 2002.

[Al-Imamy06] Al-Imamy, S., Alizadeh, J., and Nour, M.A. On the Development of a Programming Teaching Tool: The Effect of Teaching by Templates on the Learning Process. *Journal of Information Technology Education,* Vol. 5, pp. 272-283, 2006.

[Alice09] Alice by Carnegie Mellon University. http://www.alice.org/ Last accessed: May 2009.

[Als05] Als, B.S., Jensen, J.J., and Skov, M.S. Comparison of Think-Aloud and Constructive Interaction in Usability Testing with Children. *In proceedings of Interaction-Design and Children Conference (IDC),* Colorado, USA, pp. 9-16, 2005.

[Anderson01] Anderson, C.A. and Bushman, B.J. Effects of Violent Video Games on Aggressive Behavior, Aggressive Cognition, Affect, Physiological Arousal and Prosocial Behavior. *Psychological Science*, Vol. 12, pp. 353-359, 2001.

[Anderson08] Anderson, C.A., Sakamoto, A., Gentile, D.A, Ihori, N, Shibuya, A, Yukawa, S., Naito, M., and Kobayashi, K. Longitudinal Effects of Violent Video Games on Aggression in Japan and the United States. *Pediatrics*, Vol. 122(5), pp. 1067-1072, 2008.

[Ang08] Ang, C.S., Avni, E., and Zaphiris, P. Linking Pedagogical Theory of Computer Games to their Usability. *Journal of Computer Assisted Learning*, Vol. 7(3), pp. 533-558, 2008.

[Annetta08] Annetta, L.A., Holmes, S.Y., Cheng, M., Sears, M., Ogren, C., and Simmons, P. Engaging Teachers and Students in Science through Video Games: Experiences from the HI FIVES Project. *Paper presented at the annual meeting of the Association for Science Teacher Education*, St. Louis, MO, USA, 2008.

[Anning94] Anning, A.J.E. Technological Capability in the Primary School Classroom. *Paper presented at the Annual Meeting of American Educational Research Association*. New Orleans, LA, USA, pp. 1-20, 1994.

[Apple92] Apple Computer, Inc. *Macintosh Human Interface Guidelines.* Addison-Wesley: Boston, MA, 1992.

[Ashfield03] Ashfield, J. and Wood, R. The Use of the Electronic Whiteboard to Enhance Teaching and Learning within Whole Class Mathematics Lessons in the Primary Classroom. In C. Crawford et al. (Eds.), *In proceedings of Society for Information Technology and Teacher Education International Conference (SITE),* pp. 2263-2266, 2003.

[Baauw05] Baauw, E., Bekker, M., and Barendregt, W. A Structured Expert Evaluation Method for the Evaluation of Children's Computer Games. In M.F. Costabile and F. Paternò (Eds.), *INTERACT 2005 Lecture Notes in Computer Science 3585*, Springer, pp. 457-469, 2005.

[Baldwin00] Baldwin, L.P. and Kuljis, J. Visualization Techniques for Learning and Teaching Programming. *In proceedings of the 2nd International Information Technology Interfaces*, Croatia, pp. 83-90, 2000.

[Barendgret03] Barendregt, W., Bekker, M., and Speerstra, M. Empirical Evaluation of Usability and Fun in Computer Games for Children. In M. Rauterberg et. al. (Eds.), *Human-Computer Interaction*, IOS Press, pp. 705-708, 2005.

[Barendgret04] Barendregt, W. and Bekker, M. Towards a Framework for Design Guidelines for Young Children's Computer Games. In M. Rauterberg (Ed.), *ICEC 2004 Lecture Notes in Computer Science 3166*, Springer, pp. 365-376, 2004.

[Barendgret05] Barendregt, W. and Bekker, M. Extended Guidelines for Usability (and Fun) Testing with Children. In J. Deen and E. Kok (Eds.), *HCI Close 2U*, SIGCHI NL, ACM Press, 2005.

[Barendgret06] Barendregt, W., Bekker, M., Bouwhuis, D., and Baauw, E. Identifying Usability and Fun Problems in a Computer Game During First Use and After Some Practice. *International Journal of Man-Machine Studies,* Vol. 64(9), pp. 830-856, 2006.

[Beavis05] Beavis, C. Pretty Good for a Girl: Gender, Identity and Computer Games. *In online proceedings of DiGRA 2005,* http://www.digra.org/dl/db/06276.30483.pdf Last accessed: May 2009.

[Becker01] Becker, K. Teaching with Games: The Minesweeper and Asteroids Experience. *Journal of Computing Sciences in Colleges,* Vol. 17(2), pp. 23-33, 2001.

[Becker05a] Becker, K. and Jacobsen, D.M. Games for Learning: Are Schools Ready for What's to Come? *In online proceedings of DiGRA 2005,* http://www.digra.org/dl/db/06278.39448.pdf Last accessed: May 2009.

[Becker05b] Becker, K. and Parker, J.R. All I Ever Needed to Know About Programming, I Learned from Re-writing Classic Arcade Games. *Paper presented at Future Play: The International Conference on the Future of Game Design and Technology*, Michigan State University, East Lansing, Michigan, USA, 2005.

[Begel97] Begel, A. *Bongo: A Kids' Programming Environment for Creating Video Games on the Web*. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, USA, 1997.

[Bekker04] Bekker, M., Barendregt, W., Crombeen, S., and Biesheuvel, M. Evaluating Usability and Fun during Initial and Extended Use of Children's Computer Games. *In proceedings of the BSC-HCI*, Leeds, 2004.

[Berger07] Berger, T. and Danneberg, M. *Implementation of Animation Features Using iEngine for Creating Serious Games*. Lab Work Report, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2008.

[Bergman05] Bergman, P. A Computer Game Used for Moral Education: Some Empirical Results. *In proceedings of DiGRA 2005,* Vancouver, British Colombia, Canada, 2005.

[Beyer98] Beyer, H. and Holzblatt, K. *Contextual Design: Defining Customer-Centered Systems*. Morgan Kaufman, San Francisco, USA, 1998.

[Bijvank09] Bijvank, M.N., Konijn, E.A., Bushman, B.J. and Roelofsma, P. Age and Violent-Content Labels Make Video Games Forbidden Fruits for Youth. *Pediatrics,* Vol. 123(3), pp. 870-876, 2009.

[Blythe03] Blythe, M.A., Overbeeke, K., Monk, A.F., and Wright, P.C. (Eds.). Funology: From Usability to Enjoyment. *Human-Computer Interaction Series,* Vol. 3, 2003.

[Bogdan98] Bogdan, B.C. and Biklen, S.K. *Qualitative Research for Education: An Introduction in Theory and Methods*. Allyn and Bacon, Boston, USA, 1998.

[Breakwell95] Breakwell, G., Hammond, S.M., and Fife-Schaw, C. *Research Methods in Psychology*. Sage Publications Ltd, London, 1995.

[Browne00] Browne, H., Bederson, B., Druin, A., Sherman, L., and Westerman, W. Designing a Collaborative Finger Painting Application for Children. *Technical Report HCIL-2000-17,* 2000.

[Bruckman97] Bruckman, A. *MOOSE Crossing: Construction, Community, and Learning in a Networked Virtual World for Kids*. PhD Dissertation. MIT, USA, 1997.

[Bruckman02] Bruckman, A. and Bandlow, A. Human-Computer Interaction for Kids. In A. Sears and J.A. Jacko (Eds.), *Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications.* Lawrence Erlbaum Associates Inc., Mahwah, NJ, 2002.

[Brusilovsky97] Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A., and Miller, P. Mini-Languages: A Way to Learn Programming Principles. *Education and Information Technologies*, Vol. 2(1), pp. 65-83, 1997.

[Buckleitner99] Buckleitner, W. The State of Children's Software Evaluation: Yesterday, Today, and in the 21st Century. *Information Technology in Childhood Education Annual,* pp. 211-220, 1999.

[Carroll04] Carroll, J.M. Beyond Fun. *Interaction*, Vol. 11 (5), pp. 38-40, 2004.

[Cassell98] Cassell, J. and Jenkins, H. (Eds.) *From Barbie to Mortal Kombat: Gender and Computer Games*. MIT Press, USA, 1998.

[Cavallari92] Cavallari, J., Hedberg, J., and Harper, B. Adventure Games in Education: A Review. *Australian Journal of Educational Technology*, Vol. 8(2), pp. 172-184, 1992.

[Chiasson05a] Chiasson, S. and Gutwin, C. Design Principles for Children's Technology. *Technical Report HCI-TR-05-02*, Computer Science Department, University of Saskatchewan, Canada, 2005.

[Chiasson05b] Chiasson, S. and Gutwin, C. Testing the Media Equation with Children. *In proceedings of CHI,* Portland, Oregon, USA, 2005.

[Clements99] Clements, D.H. The Effective Use of Computers with Young Children. In V.J. Copley (Ed.), *Mathematics in the Early Year,* National Council of Teachers of Mathematics, Reston, VA, USA, 1999.

[CTR09] Children's Technology Review. http://www.childrenssoftware.com/ Last accessed: May 2009.

[Cockburn98] Cockburn, A. and Bryant, A. Cleogo: Collaborative and Multi-Metaphor Programming for Kids. *In proceedings of the 3rd Asia Pacific Conference on Computer Human Interaction (APCHI)*, Kanagawa, Japan, 1998.

[Conn03] Conn, B.J. and Rose, K. *Powerful Ideas in the Classroom*. Viewpoints Research Institute, Los Angelas, CA, 2003.

[Conway99] Conway, M., Audia, S., Burnette, T. et. al. Alice: Lessons Learned from Building a 3D System for Novices. *In proceedings of Conference on Human Factors in Computing Systems,* The Hague, Netherlands, pp. 486-493, 1999.

[Costabile07] Costabile M., Roselli T., Lanzilotti R., Ardito C., and Rossano V. A Holistic Approach to the Evaluation of E-Learning Systems. *LNCS 4556*. Springer, Berlin, Germany, pp. 530-538, 2007.

[Crawford84] Crawford, C. *The Art of Computer Game Design.* McGraw-Hill, USA, 1984.

[Crawford03] Crawford, C. *Chris Crawford on Game Design.* New Riders, USA, 2003.

[Csikszentmihalyi94] Csikszentmihalyi, M. Flow: The Psychology of Optimal Experience. *Games and Culture*, Vol. 1(1), pp. 36-40, 1994.

[Csikszentmihalyi96] Csikszentmihalyi, M. *Creativity, Flow and the Psychology of Discovery and Invention*. Harper Collins, New York, USA, 1996.

[Dachselt08] Dachselt, R., Frisch, M., and Decker, E. Enhancing UML Sketch Tools with Digital Pens and Paper. *In proceedings of the 4th ACM Symposium on Software Visualization*, Ammersee, Germany, 2008.

[Dann05] Dan, W.P., Cooper, S., and Pausch, R. *Learning to Program with Alice*. Prentice Hall, Upper Saddle River, New Jersey, USA, 2005.

[Denker04] Denker, M. and Gälli, M. Squeak: Lernumgebung und Smalltalk-System für Kinder und Erwachsene. 2004. http://www.heise.de/ct/04/07/216/ Last accessed: May 2009.

[Denner05] Denner, J., Bean, S., and Werner, L. Girls Creating Games: Challenging Existing Assumptions about Game Content. *In online proceedings of DiGRA 2005,* http://www.digra.org/dl/db/06278.43275.pdf Last accessed: May 2009.

[Derryberry07] Derryberry, A. Serious Games: Online Games for Learning. 2007. http://www.adobe.com/resources/elearning/pdfs/serious_games_wp.pdf Last accessed: May 2009.

[Desurvire04] Desurvire, H., Caplan, M., and Toth, J.A. Using Heuristics to Evaluate the Playability of Games. *In proceedings of the CHI on Human Factors in Computing Systems,* Vienna, Austria, 2004.

[Diezmann00] Diezmann, C.M. and Watters, J.J. Identifying and Supporting Spatial Intelligence in Young Children. *Contemporary Issues in Early Childhood,* Vol. 1(3), pp. 4-9, 2000.

[diSessa87] diSessa, A.A. Reference and Data Construction in Boxer. In M.J. Tauber and P. Gorny (Eds.), *Visual Aids in Programming,* Springer, Heidelberg, Germany, pp. 151-162, 1987.

[Dix98] Dix, A., Finlay, J., Abowd, G.D. and Beale, R. *Human-Computer Interaction*. Prentice Hall, Hillsdale, New Jersey, USA, 1998.

[Dix03] Dix, A. Being Playful: Learning from Children. *In proceedings of Interaction-Design and Children Conference (IDC),* Preston, UK, pp. 3-9, 2003.

[Donker01] Donker, A. and Markopoulos, P. Assessing the Effectiveness of Usability Evaluation Methods for Children. In N.M. Avouris and N. Fakotakis (Eds.), *Advances in Human Computer Interaction I*, Typorama Publications, Greece, pp. 409-410, 2001.

[Druin94] Druin, A. and Solomon, C. Designing Educational Computer Environments for Children. *In proceedings of CHI,* Boston, Massachusetts, USA, pp. 403-404, 1994.

[Druin99a] Druin, A. *The Design of Children's Technology.* Morgan Kaufmann Publishers Inc., San Francisco, CA, 1999.

[Druin99b] Druin, A. Cooperative Inquiry: Developing new Technologies for Children with Children. In M. W. Altom and M.G. Williams (Eds.), *In proceedings of the ACM CHI Human Factors in Computing Systems Conference,* ACM, New York, USA, pp. 592-599, 1999.

[Druin02a] Druin, A. Age Matters. *ACM SIGCHI Bulletin: A Supplement to Interactions,* Vol. Sept/Oct, pp. 5, 2002.

[Druin02b] Druin, A. When Teachers are Involved in the Design of New Technology. *ACM SIGCHI Bulletin: A Supplement to Interactions,* Vol. March/April 2002, pp. 6, 2002.

[Duda98] Duda, S. Design and Empirical Testing of a Checklist for the Evaluation of Multimedia Software for Children. In A. Sutcliffe, J. Ziegler and P. Johnson (Eds.), *Designing Effective and Usable Multimedia Systems*, Kluwer Academic Publishers, Boston, USA, 1998.

[Dyck03] Dyck, J., Pinelle, D., Brown, B., and Gutwin, C. Learning from Games: HCI Design Innovations in Entertainment Software. *In proceedings of the Conference on Graphics Interface,* Halifax, North Carolina, USA, 2003.

[Edwards01] Edwards, E., Elliot, J., and Bruckman, A. AquaMOOSE 3D: Math Learning in a 3D Multi-User Virtual World. *In proceedings of Conference on Human Factors in Computing Systems*, Seattle, Washington, USA, pp. 259-260, 2001.

[Egenfeldt-Nielsen04] Egenfeldt-Nielsen, S. Practical Barriers in Using Educational Computer Games. *On the Horizon: Special Issue*, Vol. 12(1), pp. 18-21, 2004.

[Egenfeldt-Nielsen05] Egenfeldt-Nielsen, S. *Beyond Edutainment: Exploring the Educational Potential of Computer Games.* PhD Dissertation, IT-University Copenhagen, Denmark, 2005.

[Egenfeldt-Nielsen06] Egenfeldt-Nielsen, S. Overview of Research on the Educational Use of Video Games. *Digital Kompetanse*, Vol. 1, pp. 184-213, 2006.

[Egloff04] Egloff, T.H. Edutainment: A Case Study of Interactive CD-ROM Playsets. *ACM Computers in Entertainment*, Vol. 2(1), pp. 1-22, 2004.

[El-Nasr06] El-Nasr, M.S. and Smith, B.K.. Learning through Game Modding. *ACM Computers in Entertainment,* Vol. 4(1), Article 3B, 2006.

[Ericson05] Ericson, E. Computer and Video Games as Learning Tools in the Classroom. *Technical Report ISU-HCI-2005-07,* Iowa State University, USA, 2005.

[Federoff02] Federoff, M.A. *Heuristics and Usability: Guidelines for the Creation and Evaluation of Fun in Video Games*. Master's thesis, Department of Telecommunications, Indiana University, USA, 2002.

[Fernaeus03] Fernaeus, Y. and Tholander, J. Games to Explore Programming. *In proceedings of Interaction-Design and Children Conference (IDC),* Preston, UK, pp. 163, 2003.

[Fernaeus06] Fernaeus, Y., Kindborg, M., and Scholz, R. Rethinking Children's Programming with Contextual Signs. *In proceedings of Interaction-Design and Children Conference (IDC),* Tampere, Finland, pp. 121-128, 2006.

[Ferrari06] Ferrari, L., Addessi, A.R., and Pachet F. New Technologies for New Music Education: The Continuator in a Classroom Setting. In Baroni et al. (Eds.), *Proceedings of the International Conference of Music Perception and Cognition (ICMPC),* Bologna, Italy, 2006.

[Feurzeig69] Feurzeig, W., Papert, S., Bloom, M., Grant, R., and Solomon, C. Programming Languages as a Conceptual Framework for Teaching Mathematics. *Final Report on the Logo Project*, National Science Foundation, 1969.

[Field01] Field, M. *Designing for Tangible Interaction*. PhD Dissertation. Zurich University of Applied Science, Austria, 2001.

[Fisch05] Fisch, S.M. Making Educational Computer Games Educational. *In proceedings of Interaction Design and Children Conference (IDC)*, Boulder, Colorado, USA, pp. 56-61, 2005.

[Freisleben08] Freisleben, M. *Die Konzeption eines als MMOG erhöht sowohl Lernmotivation als auch Lernerfolg bei Schülern*. Master's thesis, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2008.

[Fromme03] Fromme, J. Computer Games as a Part of Children's Culture. *The International Journal of Computer Game Research*, Vol. 3(1), pp. 1-23, 2003.

[Gadegast05] Gadegast, P. *Objekt-Orientierte Programmierung von Entertainmentumgebungen in Squeak*. Internship Report, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, Germany, 2005.

[Gaelli06] Gaelli, M., Nierstrasz, O., and Stinckwich, S. Idioms for Composing Games with Etoys. *In proceedings of the 4th International Conference on Creating, Connecting and Collaborating through Computing (C5),* 2006.

[GameMaker09] Game Maker by YoYo Games. http://www.yoyogames.com/gamemaker/ Last accessed: May 2009.

[Gelderblom04] Gelderblom, H. Designing Software for Young Children: Theoretically Grounded Guidelines. *In proceedings of Interaction Design and Children Conference (IDC),* Maryland, USA, pp. 121-122. 2004.

[Gee03] Gee, J.P. *What Video Games have to Teach Us about Learning and Literacy*. Palgrave/Macmillian, New York, USA, 2003.

[Gee05] Gee, J.P. Learning by Design: Good Video Games as Learning Machines. *E-Learning*, Vol. 2(1), pp. 5-16, 2005.

[Gentile04] Gentile, D.A. and Anderson, C.A. Violent Video Games: The Newest Media Violence Hazard. In D.A. Gentile (Eds.), *Media Violence and Children*, Westport, CT, Praeger Publishing, 2004.

[Gibson03] Gibson, J.P. A Noughts and Crosses Java Applet to Teach Programming to Primary School Children. *In proceedings of the 2nd International Conference on Principles and Practice of Programming in Java*, Kinkenny City, Ireland, UK, pp. 85-88, 2003.

[Gilutz02] Gilutz, S. and Nielsen, J. Usability of Websites for Children: 70 Design Guidelines. 2000. http://www.nngroup.com/reports/kids/ Last accessed: May 2009.

[Gómez-Albarrán05] Gómez-Albarrán, M. The Teaching and Learning of Programming: A Survey of Supporting Software Tools. *The Computer Journal*, Vol. 48(2), pp. 130-144, 2005.

[Good04] Good, J. and Robertson, J. Computer Games Authored by Children: A Multi-Perspective Evaluation. *In proceedings of Interaction-Design and Children Conference (IDC),* College Park, Maryland, USA, pp. 123-124, 2004.

[Google09] Google's Software Principles. http://www.google.com/press/ Last accessed: May 2009.

[Goolnik06] Goolnik, S., Robertson, J., and Good, J. Learner Centred Design in the Adventure Author Project. *International Journal of Artificial Intelligence in Education,* Vol. 16(4), pp. 415-438, 2006.

[Gorp01] Gorp, M.J.V. and Grissom, S. An Empirical Evaluation of Using Constructive Classroom Activities to Teach Introductory Programming. *Computer Science Education,* Vol. 11(3), pp. 247-260, 2001.

[Gorriz00] Gorriz, C.M. and Medina, C. Engaging Girls with Computers through Software Games. *Communications of the ACM*, Vol. 45(1), pp. 42-49, January, 2000.

[Gottschlag07] Gottschlag, D. *Interaction Design for a Spherical Motion-Sensitive Input Device*. Master's thesis, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2007.

[Gray98] Gray, J., Boyle, T., and Smith, C. A Constructivist Learning Environment Implemented in Java. *In proceedings of the 6th Annual Conference on the Teaching of Computing and the 3rd Annual Conference on Integrating Technology into Computer Science Education: Changing the Delivery of Computer Science Education (ITiCSE)*, pp. 94-97, 1998.

[Greenbaum91] Greenbaum, J. and Kyng, M. *Design at Work: Cooperative Design of Computer Systems*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, USA, 1991.

[Grobelny05] Grobelny, J., Karwowski, W., and Drury, C. Usability of Graphical Icons in the Design of Human-Computer Interfaces, *International Journal of Human-Computer Interaction*, Vol. 18(2), pp. 167-182, 2005.

[Guha04] Guha, M.L., Druin, A., Chipman, G., Fails, J.A., Simms, S., and Farber, A. Mixing Ideas: A New Technique for Working with Young Children as Design Partners. *In proceedings of Interaction-Design and Children Conference (IDC),* College Park, Maryland, USA, pp. 35-42, 2004.

[Guzdial00] Guzdial, M. Using Squeak for Teaching User Interface Software. *In proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education,* pp. 219-223, 2000.

[Guzdial01a] Guzdial, M., and Rose, K. *Squeak: Open Personal Computing and Multimedia*. Prentice Hall, New Jersey, USA, 2001.

[Guzdial01b] Guzdial, M. *Squeak: Object-Oriented Design with Multimedia Applications*. Prentice Hall, New Jersey, USA, 2001.

[Guzdial04] Guzdial, M. Programming Environments for Novices. In S. Fincher and M. Petre (Eds.), *Computer Science Education Research*, Taylor & Francis Group, UK, pp. 127-154, 2004.

[Habgood05] Habgood, M.P.J., Ainsworth, S.E., and Benford, S. Intrinsic Fantasy: Motivation and Affect in Educational Games Made by Children. *Artificial Intelligence in Education*, 2005.

[Habgood07] Habgood, M.P.J. *The Effective Integration of Digital Games and Learning Content*. PhD Dissertation, LSRI, Computer Science Department, University of Nottingham, UK, 2007.

[Halverson05] Halverson, R. What Can K-12 School Leaders Learn from Video Games and Gaming? *Innovate: Journal of Online Education*, Vol. 1(6), August/September, 2005.

[Hanna97] Hanna, L. Risden, K. and Alexander, K. J. Guidelines for Usability Testing with Children. *Interactions*, September/October 1997, pp. 9-14, 1997.

[Hanna04] Hanna, L., Neapolitan, D., and Risden, K. Evaluating Computer Game Concepts with Children. *In proceedings of Interaction-Design and Children Conference (IDC),* College Park, Maryland, USA, pp. 49-56, 2004.

[Harel90] Harel, I. and Papert, S. Software Design as a Learning Environment. *Interactive Learning Environment*, Vol. 1(1), pp. 1-32, 1990.

[Haugland97] Haugland, S.W. and Wright, J.L. *Young Children and Technology: A World of Discovery*. Allyn and Bacon, Boston, USA, 1997.

[Hayes05] Hayes, E. Women and Video Gaming: Gendered Identities at Play. *In proceedings of Games, Learning and Society Conference*, Madison, WI, USA, 2005.

[Henry00] Henry, P. *User-Centered Design for Improved Software Usability.* Artech House, Boston, USA, 2000.

[Hinostroza00] Hinostroza, J. and Mellar, H. Considering Pedagogy in the Design, Development and Evaluation of Educational Software. In J. Bourdeau & R. Heller (Eds.), *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications (Ed-Media)*, Chesapeake, VA: AACE, pp. 454-459, 2000.

[Hinze03] Hinze, J. *3D-Animations-Skripting für Nicht-Professionelle Benutzer*. Master's thesis, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2003.

[Horie03] Horie, Y. Psycho-Physiological Reactions in Children Using Computer Games. *In online proceedings of International Ergonomics Association Conference (IEA),* http://www.iea.cc/ergonomics4children/pdfs/iea2003horie.pdf Last accessed: May 2009.

[Horn07] Horn, M.S. and Jacob, R.J.K. Designing Tangible Programming Languages for Classroom Use. *In proceedings of Tangible Embedded Interaction,* ACM Press, Baton Rouge, LA, USA, 2007.

[Hourcade04] Hourcade, J.P. Building KidPad: An Application for Children's Collaborative Storytelling. *Software: Practice and Experience,* Vol. 34(9), pp. 895-914, 2004.

[Hoyles04] Hoyles, C. and Noss, R. Making Rules in Collaborative Game Design. In Siraj-Blatchford, J. (Ed.), *Developing New Technologies for Young Children.* Trentham Books, Stoke on Trent, UK, pp. 55-73, 2004.

[Huber07] Huber, C., Röber, N., Hartman, K., and Masuch, M. Evolution of Interactive Audiobooks. *In proceedings of Audio Mostly Conference,* Ilmenau, Germany, 2007.

[Hull03] Hull, R. and Reid, J. Designing Engaging Experiences with Children and Artists. In Blythe, M.A. et. al. (Eds.), *Funology: From Usability to Enjoyment*, Kluwer Academic Publishers, pp. 179-187, 2003.

[Hussain03] Hussain, H., Embi, Z.C., and Hashim, S. A Conceptualized Framework for Edutainment. *In proceedings of Informing Science and Information Technology Joint Conference,* pp. 1077-1083, 2003.

[Hussain06] Hussain, S., Lindh, J., and Shukur, G. The Effect of LEGO Training on Pupils' School Performance in Mathematics, Problem Solving Ability and Attitude: Swedish Data. *Educational Technology and Society,* Vol. 9(3), pp. 182-194, 2006.

[Höysniemi04] Höysniemi, J., P. Hämäläinen, and L. Turkki. Wizard of Oz Prototyping of Computer Vision Based Action Games for Children. *In proceedings of Interaction-Design and Children Conference (IDC),* College Park, Maryland, USA, pp. 27-34, 2004.

[Ibrahim04] Ibrahim, A.A.A. and Salim, S.S. Designing Software for Child Users: A Case Study of a Web Page Construction Kit for Children. *Malaysian Journal of Computer Science,* Vol. 17(1), pp. 32-41, 2004.

[IGDA08] International Game Developers Association. IGDA Curriculum Framework: The Study of Games and Game Development. http://www.igda.org/wiki/images/e/ee/Igda2008cf.pdf Last accessed: May 2009.

[Impara09] Impara GmbH. http://www.impara.de/projekt_gameengine.html Last accessed: May 2009.

[Infosurv06] Infosurv White Paper. 5-point vs 6-point Likert Scales. http://www.infosurv.com/images/Likert_Scale_Debate.pdf Last accessed: May 2009.

[Ingalls97] Ingalls, D., Kaehler, T., Maloney, J., Wallace, S. and Kay, A. Back to the Future: The Story of Squeak - A Usable Smalltalk Written in Itself. *In proceedings of the International Conference on Object Oriented Programming, Systems, Languages and Applications (OOPSLA)*, pp. 318-326, 1997.

[Inkpen95] Inkpen, K., Booth, K. S., Klawe, M., and Upitis, R. Playing Together Beats Playing Apart, Especially for Girls. *In proceedings of the Computer Support for Collaborative LearningConference (CSCL),* Bloomington, Indiana, 1995.

[Inkpen97a] Inkpen, K. Three Important Research Agendas for Educational Multimedia: Learning, Children and Gender. *In proceedings of Educational MultiMedia*, Calgary, AB, Canada, pp. 521-526, 1997.

[Inkpen97b] Inkpen, K., McGrenere, J., Booth, K.S., and Klawe, M. The Effect of Turn-Taking Protocols on Children's Learning in Mouse-Driven Collaborative Environments, *In proceedings of the Conference on Graphics Interface*, Kelowna, British Columbia, Canada, pp. 138-145, 1997.

[Inkpen99a] Inkpen, K. Designing Handheld Technologies for Kids. *Personal Technologies Journal*, Vol. 3(1&2), pp. 81-89, 1999.

[Inkpen99b] Inkpen, K., Ho-Ching. W-L., Kuederle, O., Scott, S. D., and Shoemaker, G. B. D. This Is Fun. We're All Best Friends and We're All Playing! Supporting Children's Synchronous Collaboration. In C. M. Hoadley and J. Roschelle (Eds.), *Proceedings of the Computer Support for Collaborative LearningConference (CSCL),* Lawrence Erlbaum Associates, Mahwah, NJ, USA, pp. 252-259, 1999.

[Ioannidou03] Ioannidou, A. Programmorphosis: A Knowledge-Based Approach to End-User Programming. *In proceedings of INTERACT,* pp. 152-159, 2003.

[Ismail07] Ismail, A., Yatim, M.H.M and Fathil, N. (2007). Potensi Squeak bagi Menggalakkan Proses Pembelajaran dan Pembinaan Kemahiran Diri di Peringkat Sekolah Rendah (The Potential of Squeak in Nurturing Learning Process and Skills Development for Primary Schoolers). *Poster presented at Ekspo Penyelidikan dan Inovasi UPSI 2007*, Tg. Malim, Malaysia, 2007.

[ISO98] ISO. Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs) - Part 11: Guidance on Usability. ISO 9241-11. 1998. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16883 Last accessed: May 2009.

[Jegers01] Jegers, K. and Wiberg, C. Evaluating Entertainment: Implications for Usability Tests Conducted on Entertainment Web Sites. *In proceedings of the 24th Information Systems Research Seminar*, Scandinavia, 2001.

[Jegers03] Jegers, K. and Wiberg, C. FunTain: Design Implications for Edutainment Games. *In proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications (Ed-Media)*, Chesapeake, VA, AACE, 2003.

[Jenkins06] Jenkins, H. and Squire, K. A Game that Will Make us Cry. *Computer Games Magazine*, February, 2006.

[Jensen05] Jensen, J.J., and Skov, M.B. A Review of Research Methods in Children's Technology Design. *In proceedings of Interaction Design and Children Conference (IDC)*, Boulder, Colorado, USA, pp. 80-87, 2005.

[Jörgenson98] Jörgenson, L. and Sinclair, N. Involving Middle School Students in Research Design. *Submitted to NCE-TeleLearning 1998*. http://www.cecm.sfu.ca/~loki/ Last accessed: May 2009.

[Kafai95] Kafai, Y. Making Game Artifacts to Facilitate Rich and Meaningful Learning. *Technical Report ED 388 682*, University of California, Los Angeles, USA, 1995.

[Kafai96a] Kafai, Y. Software by Kids for Kids. *Communication of the ACM*, Vol. 39(4), pp. 38-39, 1996.

[Kafai96b] Kafai, Y. and Resnick, M. *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Lawrence Erlbaum, Mahwah, New Jersey, USA, 1996.

[Kafai98a] Kafai, Y., Franke, M., Ching, C., and Shih, J. Games as Interactive Learning Environments Fostering Teachers' and Students' Mathematical Thinking. *International Journal of Computers for Mathematical Learning*, Vol. 3(2), pp. 149-193, 1998.

[Kafai98b] Kafai, Y., Franke, M.L., Ching, C.C., and Shih, J.C. Game Design as an Interactive Learning Environment for Fostering Students' and Teacher' Mathematical Inquiry. *International Journal of Computers for Mathematical Learning*, Vol. 3, pp. 149-184, 1998.

[Kafai01] Kafai, Y. The Educational Potential of Electronic Games: From Games-to-Teach to Games-to-Learn. *Paper presented at the Playing by the Rules*, Cultural Policy Center, University of Chicago, USA, 2001.

[Kafai05] Kafai, Y. *Minds in Play: Computer Game Design as a Context for Children's Learning*. Lawrence Erlbaum Associates, New Jersey, USA, 2005.

[Kafai06] Kafai, Y. (2006). Playing and Making Games for Learning: Instructionist and Constructionist Perspectives for Game Studies. *Games and Culture*, Vol. 1(1), pp. 36-40, 2006.

[Kahn01] Kahn, K. Generalizing by Removing Detail: How any Program can be Created by Working with Examples. In Lieberman, H. (Ed.), *Your Wish is My Command: Programming by Example.* Morgan Kaufmann, San Francisco, USA, pp. 21-43, 2001.

[Kaiser02] Kaiser, H.J. Key Facts: Children and Video Games. *The Henry J. Kaiser Family Foundation.* Fall 2002, pp. 1-4, 2002.

[Karuno03] Karuno, H. and Konomi, S. Squeak Workshop Experiences in Kyoto. *In proceedings of Creating, Connecting and Collaborating through Computing (C⁵),* pp. 218-221, 2003.

[Kay72] Kay, A. A Personal Computer for Children of All Ages. *Paper presented at the ACM National Conference*, Boston, USA, pp.2, 1972.

[Kay77] Kay, A. and Goldberg, A. Personal Dynamic Media. The New Media Reader, pp. 391-404, 1977. http://www.newmediareader.com/book_samples/nmr-26-kay.pdf Last accessed: May 2009.

[Kay05] Kay, A. Squeak Etoys, Children and Learning. *VPRI Research Notes RN-2005-002,* Viewpoints Research Institute, 2005.

[Kelleher05] Kelleher, C. and Pausch, R. Lowering the Barriers to Programming: A Taxonomy of Programming Environments and Languages for Novice Programmers. *Journal of ACM Computing Surveys*, Vol. 37(2), pp. 83-137, 2005.

[Kerawalla05] Kerawalla, L. and Crook, C.K. From Promises to Practices: The Fate of Educational Software in the Home. *Technology, Pedagogy and Education,* Vol. 14(2), pp. 107-125, 2005.

[Kharrazi05] Kharrazi, H.H.K. Usable Guidelines for Usability Testing with Children. 2005. http://flame.cs.dal.ca/~kharrazi/courses/HF/Midterm_revised_final.pdf Last accessed: May 2009.

[KIM2008] Medienpädagogische Forschungsverbund Südwest: KIM-Studie 2008. http://www.mpfs.de/fileadmin/KIM-pdf08/KIM08.pdf Last accessed: May 2009.

[Kindborg03] Kindborg, K. *Concurrent Comics: Programming of Social Agents by Children.* PhD Dissertation, Department of Computer and Information Science, Linköpings Universitet, Sweden, 2003.

[Kirriemuir02] Kirriemuir, J. and McFarlane, A. *The Use of Computer Games in the Classroom.* Coventry, Becta, 2002.

[Kirriemuir03] Kirriemuir, J. and McFarlane, A. *Literature Review in Games and Learning.* Nesta Future Lab, 2003.

[Klawe95] Klawe, M.M. and Phillips, E. A Classroom Study: Electronic Games Engage Children as Researchers. Paper presented at the *International Conference on Computer-Supported Collaborative Learning*, Bloomington, Indiana, USA, 1995.

[Klopfer05] Klopfer, E. Playing to Learn: State-of-the-Art Computer Games Go to School. *Access Learning*, July/August, pp. 10-11, 2005.

[Knuth84] Knuth, D. Literate Programming. *The Computer Journal*, Vol. 27(2), pp. 91-111, 1984.

[Ko04] Ko, A.J., Myers, B.A., and Aung, H.H. Six Learning Barriers in End-User Programming Systems. *Paper presented at IEEE Symposium on* Visual Languages and Human-Centric Computing *(VL/HCC),* pp. 199-206, 2004.

[Kristensen03] Kristensen, J.F., Eriksen, M.A., Iversen, O.S., Kanstrup, A.M., Nielsen, C., and Petersen, M.G. Young People in Old Cars: Challenges for Cooperative Design. *In proceedings of the 26th Information Systems Research Seminar*, Scandinavia, 2003.

[Lamb06] Lamb, A. and Johnson, L. Educational Software Evaluation Tool. 2006. http://eduscapes.com/earth/management/softevaluation.html Last accessed: May 2009.

[Larssen04] Larssen, A.T., Loke, L., Robertson, T., and Edwards, J. Understanding Movement as Input for Interaction: A Study of Two Eyetoy Games. *In proceedings of the 17th Annual Conference of the Australian Computer-Human Interaction (OZCHI),* Wollongong, Australia, 2004.

[Latif07] Latif, R.A. Understanding Malaysian Students as Gamers. *In proceedings of the 2nd International Conference on Digital Interactive Media in Entertainment and Arts (DIMEA),* pp. 137-141, 2007.

[Lin05] Lin, J.M.C., Yen, L. Y., Yang, M.C. and Chen, C.F. Teaching Computer Programming in Elementary Schools: A Pilot Study. *In proceedings of the National Educational Computing Conference (NECC),* Pennsylvania, USA, 25-30 June, 2005.

[MacFarlane03] MacFarlane, S., Read, J., Höysniemi, J., and Markopoulos, P. Half-Day Tutorial: Evaluating Interactive Products for and with Children. In M. Rauterberg et. al. (Eds.), *Interact 2003*, IOS Press, pp. 1027-1028, 2003.

[MacFarlane05a] MacFarlane, S. and Pasiali, A. Adapting the Heuristic Evaluation Method for Use with Children. *In Workshop on Child Computer Interaction: Methodological Research at Interact*, Rome, Italy, 2005.

[MacFarlane05b] MacFarlane, S., Sim, G., and Horton, M. Assessing Usability and Fun in Educational Software. *In proceedings of Interaction Design and Children Conference (IDC),* Boulder, Colorado, USA, pp. 103-109, 2005.

[Magnussen03] Magnussen, R., Misfeldt, M., and Buch, T. Participatory Design and Opposing Interests in Development of Educational Computer Games. *In online proceedings of DiGRA 2003,* http://www.digra.org/dl/db/05150.36589.pdf, Last accessed: May 2009.

[Malone80] Malone, T. W. What Makes Things Fun to Learn? Heuristics for Designing Instructional Computer Games. *In the 3rd ACM SIGSMALL Symposium at Palo Alto*, CA, New York, pp. 162-169, 1980.

[Malone81] Malone, T.W. Heuristics for Designing Enjoyable User Interfaces: Lessons from Computer Games. *Association for Computing Machinery*, pp. 63-68, 1981.

[Malone83] Malone, T. W. Guidelines for Designing Educational Computer Programs. *Childhood Education*, Vol. 59 (4), pp. 241-47, 1983.

[Malone87] Malone, T. W. and Lepper, M. R. Making Learning Fun: A Taxonomy of Intrinsic Motivations for Learning. In R. E. Snow & M. J. Farr (Eds.), *Aptitude, Learning, and Instruction III: Cognitive and Affective Process Analysis*, Lawrence Erlbaum Associates, Hillsdale, NJ, USA, pp. 223-253, 1987.

[Malone04] Malone, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., and Resnick, M. Scratch: A Sneak Preview. *In proceedings of Creating, Connecting and Collaborating through Computing (C5)*, pp.104-109, 2004.

[Mano04] Mano, A. and Campos, J.C. A Study about Usability Criteria on Computer Interfaces for Children. *In proceedings of the first Portuguese Forum of Experimental Psychology*, Braga, Portugal, 2004.

[Markopoulus08] Markopoulus, P., Read, J., MacFarlane, S., and Höysniemi, J. *Evaluating Children's Interactive Products: Principles and Practices for Interaction Designers*. Morgan Kaufmann Publishers, Amsterdam, Netherland, 2008.

[Masuch05] Masuch, M. and Rueger, M. Challenges in Collaborative Game Design Developing Learning Environments for Creating Games. *In proceedings of the Third International Conference on Creating, Connecting and Collaborating through Computing (C5)*, pp. 67–74, 2005.

[Masuch07] Masuch, M., Yatim, M.H.Y., and Gadegast, P. Developing Software for Children: Experiences from Creating a 3D Drawing Application. *In proceedings of Mensch und Computer 2007*, Bauhaus-Universität Weimar, Germany, pp. 179-188, 2007.

[Matsuzawa04] Matsuzawa, Y., Sugiura, M., and Ohiwa, H. A Trial Course of Programming with Squeak. *In proceedings of Creating, Connecting and Collaborating through Computing (C5)*, pp. 188, 2004.

[Mattila06] Mattila, J. and Väätänen, A. UbiPlay: An Interactive Playground and Visual Programming Tools for Children. *In proceedings of Interaction-Design and Children Conference (IDC),* Tempere, Finland, pp. 129-136, 2006.

[Mayhew99] Mayhew, D.J. *The Usability Engineering Lifecycle: A Practitioner Handbook for User Interface Design*. Morgan Kaufmann, 2002.

[McFarlane02] McFarlane, A., Sparrowhawk, A., and Heald, Y. Report on the Educational Use of Games: An Exploration by TEEM of the Contribution Which Games can Make to the Education Process. *Department of Education and Skills Report*, Cambridge, UK, pp. 1-26, 2002.

[McGee05] McGee, K. and Kindborg, M. Research on Partner Technology. *In proceedings of Artificial Intelligence and Learning Systems*, Västerås, Sweden, 2005.

[Medlock02] Medlock, M.C., Wixon, D., Terrano, M., Romero, R., and Fulton, B. *Using RITE Method to Improve Products: A Definition and a Case Study*. Usabillity Professionals Association, Orlando FL, USA, 2002.

[Michael05] Michael, D. and Chen, S. *Serious Games: Games that Educate, Train, and Inform*. Course Technology PTR, Boston, USA, 2005.

[Microsoft95] Microsoft. *Windows Interface Guidelines for Software Design*. Microsoft Press, Redmond, WA, USA, 1995.

[Miller56] Miller,G.A. The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *The Psychological Review,* Vol. 63, pp. 81-97, 1956. http://www.musanim.com/miller1956/ Last accessed: May 2009.

[MOE09] Ministry of Education Malaysia. The Official Curriculum Development Centre. http://www.ppk.kpm.my/english/index.htm Last accessed: May 2009.

[Monk02] Monk, A., Hassenzahl, M., Blythe, M., and Reed, D. Funology: Designing Enjoyment. *In proceedings of CHI,* Minnesota, USA, pp. 924-925, 2002.

[Montemayor00] Montemayor, J., Druin, A. and Hendler, J. PETS: A Personal Electronic Teller of Stories. In A. Druin, J. Hendler and J. Neilsen (Eds.), *Robot for Kids: Exploring New Technologies for Learning*. Morgan Kaufmann Publishers, San Francisco, USA, pp 73-108, 2000.

[Montemayor01] Montemayor, J., Druin, A., Farber, A., Simms, S., Churaman, W., and D'Amour, A. Physical Programming: Designing Tools for Children to Create Physical Interactive Environments, *Technical Report HCIL-2001-21.* 2001.

[Morgado06] Morgado, L., Cruz, M. and Kahn, K. Radia Perlman: A Pioneer of Young Children Computer Programming. *In proceedings of the FORMATEX,* 2006.

[Moskal00] Moskal, B., Lurie, D. and Cooper, S. Evaluating the Effectiveness of a New Instructional Approach. *In proceedings of the Technical Symposium on Computer Science Education*, Virginia, USA, pp. 75-79, 2000.

[Muda05] Muda, Z. and Basiron, I.S. Multimedia Adventure Game as Edutainment Application. *In proceedings of the International Conference on Computational Intelligence for Modelling Control and Automation (CIMCA-IAWTIC),* 2005.

[Myers86] Myers, B. A. Visual Programming, Programming by Example and Program Visualization: A Taxonomy. *In proceedings of CHI*, pp. 59-66, 1986.

[Myers90] Myers, B.A. Creating User Interfaces Using Programming by Example, Visual Programming and Constraints. *ACM Transactions on Programming Languages and Systems,* Vol. 12(2), pp. 143-177, 1990.

[Myers92] Myers, B. A. and Rosson, M.B. Survey on User Interface Programming. *In proceedings of CHI,* pp. 195-202, 1992.

[Nacke05] Nacke, L. *Facilitating the Education of Game Development*. Master's thesis, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2005.

[Nakagawa99] Nakagawa, H., and Itabashi, S. A Qualitative Study of Children's Exchanging Activity in the Group Work of Software Making. *In proceedings of the 9th Annual Conference of the Internet Society (INET)*, San Jose, California, USA,1999.

[Nielsen90] Nielsen, J. and Molich, R. Heuristic Evaluation of User Interfaces. *In proceedings of ACM CHI*, Seattle, WA, USA, pp. 249-256, 1990.

[Nielsen94] Nielsen, J. Heuristic Evaluation. In J. Nielsen and R.L. Mack (Eds.), *Usability Inspection Methods*, John Wiley & Sons, New York, USA, 1994.

[Nielsen00] Nielsen, J. Jakob Nielsen's Alertbox: Why you only Need to Test with 5 Users. 2000. http://www.useit.com/alertbox/20000319.html Last accessed: May 2009.

[Nielsen07] Nielsen, J. Usability of Websites for Children: 70 Design Guidelines Based on Usability Studies with Kids. 2007. http://www.nngroup.com/reports/kids/ Last accessed: May 2009.

[Noss96] Noss, R. and Hoyles, C. *Windows on Mathematical Meanings: Learning Cultures and Computers*. Kluwer Academic Publishers, London, UK, 1996.

[Oblinger04] Oblinger, D. The Next Generation of Educational Engagement. *Journal of Interactive Media in Education*, Vol. 2004(8), pp. 1-18, 2004.

[OLPC09] One Laptop per Child (OLPC). http://laptop.org/ Last accessed: May 2009.

[Overmars04] Overmars, M. Teaching Computer Science Through Game Design. *IEEE Computer: Institute of Electrical and Electronics Engineers*, Vol. 23, pp. 11-13, 2004.

[Pagulayan03] Pagulayan, R. J., Keeker, K., Wixon, D., Romero, R., and Fuller, T. User-Centered Design in Games. In J. Jacko and A. Sears (Eds.), *Handbook for Human-Computer Interaction in Interactive Systems*, Mahwah, NJ, Lawrence Erlbaum Associates, pp. 741-758, 2003.

[Pane98] Pane, J.F. Designing a Programming System for Children with a Focus on Usability. *In proceedings of CHI*, pp. 62-63, 1998.

[Pan09] Pan European Game Information. http://www.pegi.info/en/index/ Last accessed: May 2009.

[Pane02] Pane, J. *A Programming System for Children that is Designed for Usability*. PhD Dissertation, School of Computer Science, Computer Science Department, Carnegie Mellon University, USA, 2002.

[Papert80] Papert, S. *Mindstorms: Children, Computers and Powerful Ideas*. Basic Books, New York, USA, 1980.

[Papert93] Papert, S. *The Children's Machine: Rethinking School in the Age of the Computer*. Basic Books, New York, USA, 1993.

[Papert96] Papert, S. Computers in the Classroom: Agents of Change. *In online article appeared in The Washington Post Education Review October 1996*, http://www.papert.org/articles/ComputersInClassroom.html, Last accessed: May 2009.

[Papert99] Papert, S. and Caperton, G. Vision for Education: The Caperton-Papert Platform. *In online paper presented for the 91st Annual National Governors' Association Meeting 1999*, http://www.papert.org/articles/Vision_for_education.html, Last accessed: May 2009.

[Paras05] Paras, B. and Bizzocchi, J. Game, Motivation and Effective Learning: An Integrated Model for Educational Game Design. *In proceedings of the DiGRA Conference: Changing Views – Worlds in Play,* 2005.

[Pardo05] Pardo, S., Vetere, F. and Howard, S. Broading Stakeholder Involvement in UCD: Designers' Perspectives on Child-Centered Design. *In proceddings of the 17th Australia Conference on Computer-Human Interaction*, Canberra, Australia, pp. 1-9, 2005.

[Parker92] Parker, L.E. and Lepper, M.R. Effects of Fantasy Contexts on Children's Learning and Motivation: Making Learning More Fun. *Journal of Personality and Social Psychology*, Vol. 62, pp. 625-633, 1992.

[Pausch94] Pausch, R. What HCI Designers can Learn from Video Game Designers. *In proceedings of Computer-Human Interaction,* Boston, Massachusetts, USA, pp. 177-178, 1994.

[Pelletier05a] Pelletier, C. The Uses of Literacy in Studying Computer Games: Comparing Students' Oral and Visual Representations of Games. *English Teaching: Practice and Critique,* Vol. 4(1), pp. 40-59, 2005.

[Pelletier05b] Pelletier, C. Studying Games in School: A Framework for Media Education. *In online proceedings of DiGRA 2005,* http://www.digra.org/dl/db/06278.32248.pdf, Last accessed: May 2009.

[Peppler05] Peppler, K.A. and Kafai, Y. Creative Coding: Programming for Personal Expression. *In article retrieved from MIT Media Lab website 2005,* http://pimarsc.pbworks.com/f/CreativeCoding.pdf, Last accessed: May 2009.

[Perry07] Perry, T.S. The Laptop Crusade. *IEEE Spectrum: April*, pp- 24-29, 2007.

[Piaget83] Piaget, J. Piaget's Theory. In P. Mussen (Ed), *Handbook of Child Psychology*. 4th Edition, Vol. 1, Wiley, New York, USA, 1983.

[Pierce03] Pierce, J. Alice in a Squeak Wonderland. *In proceedings of Creating, Connecting and Collaborating through Computing (C5)*, pp. 40-43, 2003.

[Pleva04] Pleva, G. Game Programming and the Myth of Child's Play. *In proceedings of Consortium for Computing Sciences in Colleges,* Northwestern, Illinois, USA, pp. 125-136, 2004.

[Pouyioutas08] Pouyioutas, P., Paschali, P. and Laghos, A. Design and Evaluation of the User Interface of the Hala Sultan Tekke Multimedia Application. *In proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications (Ed-Media)*, Chesapeake, VA, AACE, pp. 1478-1483. 2008.

[Prensky01] Prensky, M. *Digital Game-Based Learning*. McGraw-Hill, New York, USA, 2001.

[Prensky06] Prensky, M. *Don't Bother Me Mom - I'm Learning: How Computer and Video Games are Preparing your Kids for Twenty-First Century Success*. Paragon House, Minnesota, USA, 2006.

[Prince04] Prince George's County Public Schools. Instructional Software Evaluation Rubric. http://www.pgcps.org/~support/RubricSoftwareEvaluation.pdf Last accessed: May 2009.

[QUIS09] QUIS from HCIL. http://www.cs.umd.edu/hcil/quis/ Last accessed: May 2009.

[Rader97] Rader, C., Brand, C., and Lewis, C. Degrees of Comprehension: Children's Understanding of a Visual Programming Environment. *In proceedings of CHI,* 1997.

[Raffle06] Raffle, H. The Future of Interface Design, through a Child's Eyes. *Paper presented at CHI Workshop: What is the Next Generation of Human-Computer Interaction,* Montreal, Canada, 2006.

[Randel92] Randel, J.M., Morris, B.A., Wetzel, C.D., and Whitehill, B.V. The Effectiveness of Games for Educational Purposes: A Review of Recent Research. *Simulation and Gaming*, Vol. 23(3), pp. 261-276, 1992.

[Rankin08] Rankin, Y., Gooch, A., and Gooch, B. The Impact of Game Design on Students' Interest in CS. *In proceedings of Game Development in Computer Science Education Conference*, Cozumel, Mexico, pp. 31-35, 2008.

[Ray04] Ray, S.G. Gender Inclusive Game Design: Expanding the Market. *Women in Games Conference,* University of Portsmouth, USA, 2004.

[Read01] Read, J., MacFarlane, S., and Casey, C. Measuring the Usability of Text Input Methods for Children. *In proceedings of HCI,* Lille, France, Springer-Verlag, 2001.

[Read02] Read, J., MacFarlane, S., and Casey, C. Pens Behaving Badly – Usability of Pens and Graphics Tablets for Text Entry with Children. *In proceedings of ACM User Interface and Software Technology,* Paris, France, pp. 21-22, 2002.

[Read04] Read, J. C. Designing Evaluations to Evaluate Designs. http://www.chici.org/references/designing_evaluations_to_evaluate.pdf Last accessed: May 2009.

[Read05a] Read, J. C. The ABC of CCI. *Interfaces*, Vol. 62**,** pp. 8-9, 2005.

[Read05b] Read, J.C. *Handwriting Recognition Technology, Children, and the Writing Process*. PhD Dissertation, Department of Computing, University of Central Lancashire, UK, 2005.

[Read05c] Read, J.C. and Fine, K. Using Survey Methods for Design and Evaluation in Child Computer Interaction. *Workshop on Child Computer Interaction: Methodological Research at Interact*, Rome, Italy, 2005.

[Read06] Read, J.C. and MacFarlane, S. Using the Fun Toolkit and Other Survey Methods to Gather Opinions in Child Computer Interaction. *In proceedings of Interaction Design and Children Conference (IDC)*, Tampere, Finland, pp. 81-88, 2006.

[Read07] Read, J.C. Children using Digital Ink for Writing. *In proceedings of the First International Workshop on Pen-Based Learning Technologies,* Vol. 00, IEEE Computer Society, pp. 1-5, 2007.

[Repenning93] Repenning, A. *Agentsheets: A Tool for Building Domain-Oriented Dynamic, Visual Environments*. PhD Dissertation, Department of Computer Science, Faculty of the Graduate School, University of Colorado, USA, 1993.

[Resnick04] Resnick, M. Edutainment? No Thanks! I Prefer Playful Learning. *Associatzione Civita*, Vol. 1(1), pp. 2-4, 2004.

[Resnick05] Resnick, M. and Silverman, B. Some Reflections on Designing Construction Kits for Kids. *In proceedings of Interaction Design and Children Conference (IDC)*, Boulder, Colarado, USA, pp. 117-122, 2005.

[Rieber01] Rieber, L.P., Davis, J., and Grant, M. Children as Multimedia Critics: Middle School Students' Motivation for and Critical Analysis of Educational Multimedia Designed by Other Children. *Paper presented at the Annual Meeting of the American Educational Research Association,* Seattle, 2001.

[Robertson94] Robertson, J. Usability and Children's Software: A User-Centered Design Methodology. *Journal of Computing in Childhood Education,* Vol. 5, pp. 257-271, 1994.

[Robertson01] Robertson, J. *The Effectiveness of a Virtual Role-Play Environment as a Story Preparation Activity*. PhD Dissertation. School of Informatics, College of Science and Engineering, University of Edinburgh, 2001.

[Robertson02] Robertson, J. Experiences of Designing with Children and Teachers in the StoryStation Project. In M. Bekker, P. Markopoulos, and M. Kersten-Tsikalkina (Eds), *In proceedings of Interaction Design and Children Conference (IDC),* Eindhoven, Netherlands, pp. 29-41, 2002.

[Robertson04] Robertson, J. and Good, J. Children's Narrative Development through Computer Game Authoring. *In proceedings of Interaction Design and Children Conference (IDC),* Maryland, USA, pp. 57-64, 2004.

[Robertson05] Robertson, J. Children's Narrative Development through Computer Game Authoring. *TechTrends*, Vol. 49(5), Springer Boston, pp. 43-59, 2005.

[Rosas03] Rosas, R., Nussbaum, M., Cumsille, P., Marianov, V., Correa, M., Flores, P., Grau, V., Lagos, F., Lopez, X., Lopez, V., Rodriguez, P., and Salinas, M. Beyond Nintendo: Design and Assessment of Educational Video Games for First and Second Grade Students. *Computers and Education*, Vol. 40(1), pp. 71-94, 2003.

[Rose04] Rose, K. Amusing Ourselves to Life. *ACM Computers in Entertainment,* Vol. 2(2), Article 7, January, 2004.

[Roth07] Roth, S. *Squeak Game Engine Framework for OLPC Project*. Lab Practical Report, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2007.

[Roussou04] Roussou, M. Learning by Doing and Learning through Play: En Exploration of Interactivity in Virtual Environments for Children. *ACM Computers in Entertainment,* Vol. 2(1), Article 1, January, 2004.

[Rowland05] Rowland, D. Game Development Experiences with Spatial Audio. *In proceedings of FuturePlay*, Michigan State University, Michigan, USA, 2005.

[Röber08] Röber, N. *Interaction with Sound: Explorations beyond the Frontiers of 3D Virtual Auditory Environments*. PhD Dissertation, Department of Simulation and Graphics, Otto-von-Guericke University Magdeburg, Germany, 2008.

[Said04] Said, N.S. An Engaging Multimedia Design Model. *In proceedings of Interaction Design and Children Conference (IDC)*, ACM, New York, USA, pp. 169-172, 2004.

[Sauvé05] Sauvé, L., Villardier, L., Probst, W., Boyd, G., Kaufman, D., et Sánchez Arias, V.G., and Power, M. Playing and Learning without Borders: A Real-time Online Play Environment. *In online proceedings of DiGRA,* http://www.digra.org/dl/db/06276.15565.pdf, Last accessed: May 2009.

[Scaife99] Scaife, M. and Rogers, Y. Kids as Informants: Telling Us What We Didn't Know or Confirming What We Knew Already? In A. Druin (Ed.), *The Design of Children's Technology*, Morgan Kaufman, pp. 27-50, 1999.

[Scheider96] Scheider, K.G. Children and Information Visualization Technologies. *Interactions*, Sept/Oct 1996 Issues, pp. 68-73, 1996.

[Schmidt08] Schmidt, M. and Roth, S. *Projekt Gatelock: Eine Spieleentwicklungsumgebung zum Bildungserzieherischen Lernen für OLPC-Kinder*. Master's thesis, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2008.

[Schuler93] Schuler, D. and Namioka, A. *Participatory Design: Principles and Practices*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, USA, 1993.

[Schuster06] Schuster, G. and Freudenberg, B. Plopp: A Squeak Application on the Shelf. In Baroni et al. (Eds.), *European Smalltalk User Group's 14th International Smalltalk Conference (ESUG),* Prague, Czech Republic, 2006.

[Schwartz06] Schwartz, J., Stagner, J., and Morrison, W. Kid's Programming Language (KPL). *In proceedings of International Conference on Computer Graphics and Interactive Techniques* (SIGGRAPH), Boston, Massachusetts, USA, Article No. 52, 2006.

[Scott02] Scott, S.D., Grant, K., Sheelagh, M., Carpendale, T, Inkpen, K.M., Mandryk, R.L., and Winograd, T. Co-Located Tabletop Collaboration: Technologies and Directions. *In proceedings of ACM Conference on Computer Supported Cooperative Work*, New Orleans, USA, 2002.

[Sedighian96a] Sedighian, K. An Investigation of Design Factors of Game-Based Electronic Learning Environment. *In proceedings of the International Conference on Learning Sciences,* Illinois, USA, pp. 589-590, 1996.

[Sedighian96b] Sedighian, K. and Sedighian, A. Can Educational Computer Games Help Educators Learn About the Psychology of Learning Mathematics in Children? In E. Jakubowsi et. al. (Eds.), *Proceedings of the 8th Annual Meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education,* Vol. 2, pp. 573-578, 1996.

[Sedighian96c] Sedighian, K. Super Tangrams: A Child-Centered Approach to Designing a Computer Supported Mathematics Learning Environment. *In proceedings of the International Conference on Learning Sciences*, Evanston, Illinois, USA, pp. 490-495, 1996.

[Seth96] Sethi, R. *Programming Languages: Concepts and Constructs*. Addison-Wesley, California, USA, 1996.

[Shaffer04] Shaffer, D. W. Video Games and the Future of Learning. *WCER Working Paper No. 2005-4*, 2004.

[Shaffer05] Shaffer, D.W. and Gee, J.P. Before Every Child is Left Behind: How Epistemic Games Can Solve the Coming Crisis in Education. *Working Paper No. 2005-7,* September, 2005.

[Shaffer06] Shaffer, D.W. *How Computer Games Help Children Learn*. Palgrave Macmillan, New York, USA, 2006.

[Shariff07] Shariff, A.S. and Yatim, M.H.Y. Introducing Squeak Etoys to Enhance Science and Mathematics Learning. *In proceedings of the 1st International Malaysian Educational Technology Conference*, Johor Bharu, Malaysia, pp. 526-532, 2007.

[Sheehan03] Sheehan, R. *The Icicle Programming Environment for Children*. PhD Dissertation, Faculty Computer Science, University of Auckland, Australia, 2003.

[Shneiderman97] Shneiderman, B. and Plaisant, C. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, Boston, USA, 1997.

[Shneiderman02] Shneiderman, B. *Leonardo's Laptop: Human Needs and the New Computing Technologies*. MIT Press, 2002.

[Shneiderman04] Shneiderman, B. Designing for Fun: How can We Design User Interfaces to be More Fun? *Interactions*, Vol. 11(5), Sept/Oct 2004 Issues, pp. 48-50, 2004.

[Sieber07] Sieber, J. *Interactive Storytelling in Multiplayer-Role-Playing Games*. Master's thesis, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2007.

[Sim04] Sim, G., Horton, M., and Strong, S. Interfaces for Online Assessment: Friend or Foe? *In proceedings of HCI Educators Workshop,* Preston, UK, pp. 36-40, 2004.

[Skalski06] Skalski, P., Lange, R., and Tamborini, R. Mapping the Way to Fun: The Effect of Video Game Interfaces on Presence and Enjoyment. *Paper presented at the 9th Annual International Workshop on Presence*, Cleveland, Ohio, USA, pp. 63-64, 2006.

[Slaughter01] Slaughter, A.H. and Strohecker, C. A Framework for Microworld-Style Construction Kits. *In proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications (Ed-Media)*, Chesapeake, VA, AACE, 2001.

[Smart09] SMART Table. http://www.gadgetsfan.com/computer-gadgets/smart-table.html Last accessed: May 2009.

[Smith94] Smith, D.C. KidSim: Programming Agents without a Programming Language. *Communications f the ACM,* Vol. 37(7), pp. 55-67, 1994.

[Smith96] Smith, D.C., Cypher, A., and Schmucker, K. Making Programming Easier for Children. *Interactions,* September/October, pp. 60-67, 1996.

[Smith00] Smith, D.C., Cypher, A., and Tesler, L. Novice Programming Comes of Age. *Communications of the ACM*, Vol. 43(3), pp. 75-81, 2000.

[Soloway86] Soloway, E. Learning to Program = Learning to Construct Mechanisms and Explanation. *Communications of ACM*, Vol. 29(9), pp. 850-858, 1986.

[Sotamaa05] Sotamaa, O. Creative User-Centered Design Practices: Lessons from Game Cultures. In L. Haddon (Ed.), *Exploring Users*, pp. 104-116, 2005.

[Spencer86] Spencer, M. Choosing Software for Children. *ERIC Clearinghouse on Elementary and Early Childhood Education ED267914*, Urbana, IL, USA, 1986.

[Stellmach08] Stellmach, S. *A Pschophysiological Logging System for a Digital Game Modification*. Internship Report, Department for Simulation and Graphics, Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Magdeburg, 2008.

[Stewart03] Stewart, T. and Travis, D. Guidelines, Standards, and Style Guides. In J.A. Jacko and A. Sears (Eds.), *The Human-Computer Interaction Handbook: Fundamental, Evolving Technologies and Emerging Applications*, Lawrence Erlbaum Associates Inc., New Jersey, USA, 2003.

[Subrahmanyam00] Subrahmanyam, K., Kraut, R.E., Greenfield, P.M., and Gross, E.F. The Impact of Home Computer Use on Children's Activites and Development. *The Future of Children: Children and Computer Technology,* Vol.10(2), Fall/Winter, pp. 123-144, 2000.

[Squire02] Squire, K. Cultural Framing of Computer/Video Games. *International Journal of Computer Game Research*, Vol. 2(1), 2002. http://gamestudies.org/0102/squire\ Last accessed: May 2009.

[Squire05] Squire, K. Changing the Game: What Happens When Video Games Enter the Classroom. *Innovate: Journal of Online Education*, Vol. 1(6), August/September, 2005.

[Stagecast09] StageCast Creator by Stagecast Software, Inc. http://www.stagecast.com/ Last accessed: May 2009.

[Steiner06] Steiner, B., Kaplan, N., and Moulthrop, S. When Play Works: Turning Game-Playing into Learning. *In proceedings of Interaction-Design and Children Conference (IDC),* Tampere, Finland, pp. 137-140, 2006.

[Strohecker99] Strohecker, C. Construction Kits as Learning Environments. *In proceedings of the International Conference on Multimedia Computing and Systems*, IEEE Computer Society, pp. 1030-1031, 1999.

[Strohecker00] Strohecker, C. and Slaughter, A.H. Approaches to Processes of Constructing in Software Construction Kits. *Technical Report TR2000-28*, Mitsubishi Electric Research Laboratories, 2000.

[Sutherland63] Sutherland, I.E. Sketchpad: A Man-Machine Graphical Communication System. *In proceedings of Spring Joint Computer Conference*, pp. 507-523, 1963.

[Sweetser05] Sweetser, P. and Wyeth, P. GameFlow: A Model for Evaluating Player Enjoyment in Games. *ACM Computers in Entertainment*, Vol. 3(3), pp. 1-24, 2005.

[Sword00] Sword, L. I Think in Pictures, You Teach in Words: The Gifted Visual Spatial Learner. *Gifted*, Vol. 114(1), pp. 27-30, 2000.

[Trushell01] Trushell, J., Burrell, C., and Maitland. Year 5 Pupils Reading an `Interactive Storybook´ on CD-ROM: Losing the Plot. *British Journal of Educational Technology,* Vol. 32(4), pp. 389-401, 2001.

[Tholander02] Tholander, J., Kahn, K., and Jansson, C.G. Real Programming of an Adventure Game by an 8 Year Old. *In proceedings of the 5th International Conference of the Learning Science,* Seattle, Washington, USA, 2002.

[Thomas00] Thomas, A. and Walkerdine, V. Girls and Computer Games. *In proceedings of the 4th European Feminist Research Conference,* Bologna, Italy, 2000.

[Uden00] Uden, L. and Dix, A. Iconic Interfaces for Kids on the Internet**.** *IFIP World Computer Congress,* Beijing, China, pp. 279-286, 2000.

[Unterhaltungs09] Unterhaltungssoftware Selbstkontrolle. http://www.usk.de/ Last accessed: May 2009.

[UPA09] Usability Professionals' Association. http://www.usabilityprofessionals.org/usability_resources/about_usability/what_is_ucd.html Last accessed: May 2009.

[Valente05] Valente, L. and Conci, A. Guff: A Game Development Tool. *In Workshop of Theses and Dissertations in the XVIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI),* 2005.

[Weevers04] Weevers, I., Fitrianie, S., Sluis, W., van Schijndel, C., Kolos-Mazuryk, L., and Martens, J.B. Read-It: A Multi-model Tangible Interface for Children who Learn to Read. In M. Rauterberg (Ed.), *ICEC 2004 Lecture Notes in Computer Science 3166*, Springer-Verlag Heidelberg, pp. 226-234, 2004.

[Wiberg03] Wiberg, C. *A Measure of Fun: Extending the Scope of Web Usability*. PhD Dissertation, Department of Informatics, Umeä University, Sweden, 2003.

[Williamson05] Williamson, B. The Participation of Children in the Design of New Technology: A Discussion Paper. *Future Lab September,* 2005.

[Wolfe98] Wolfe, J. and Crookall, D. Developing a Scientific Knowledge of Simulation/Gaming. *Simulation and Gaming*, Vol. 29(1), pp. 7-19, 1998.

[Wong98] Wong, D. and Baker, C. Pain in Children: Comparison of Assessment Scales. *Pediatric Nursing,*Vol. 14(1), pp. 9-17, 1998.

[Wook07] Wook, T.S.M.T. and Salim, S.S. A Conceptual Design for Children's WebOPAC Interface: Graphic Design Issues. In: M. J. Smith and G. Salvendy (Eds.), *Human Interface Part II HCII 2007 LNCS 4558*, Springer-Verlag Berlin Heidelberg, pp. 785-791, 2007.

[Wright05] Wright, T. and Cockburn, A. Evaluation of Two Textual Programming Notations for Children. *Research and Practice in Information Technology,* Vol. 40, pp. 55-62, 2005.

[Wright06] Wright, T. PatternProgrammer: Yet Another Rule-Based Programming Environment for Children. *In proceedings of Seventh Australasian User Interface Conference*, Vol. 50, pp. 91-96, 2006.

[Wyeth03] Wyeth, P. and Purchase, H.C. Using Developmental Theories to Inform the Design of Technology for Children. *In proceedings of Interaction-Design and Children Conference (IDC),* Preston, UK, pp. 93-100, 2003.

[Wyeth06] Wyeth, P., Diercke, C., and Viller, S. Design for Inspiration: Children, Personal Connections and Educational Technology. *In proceedings of Australasian Computer-Human Interaction Conference,* 2006.

[Yamaguchi08] Yamaguchi, Y., Yamaguchi, H., Kawasaki, Y. and Sasaki, H. Difference by School Age in Icon Search Recognition in VDT Work in Young Learners. *In proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications (Ed-Media)*, Chesapeake, VA, AACE, New York, USA, pp. 1634-1643, 2008.

[Yatim06a] Yatim, M.H.M. Computer Games for Supporting and Facilitating the New Education Era. *Paper presented at Malaysian Research Group International Conference (MRGIC),* Salford, UK, 2006.

[Yatim06b] Yatim, M.H.M., Nacke, L., and Masuch, M. Improving Game Design by Understanding the Gender Differences: The Cognitive Approach. *In proceedings of International Conference on Gender in Educational Games and Gender Sensitive Approaches to E-Learning*, Donau University Krems, Austria, pp. 279-290, 2006.

[Yatim07a] Yatim, M.H.M. and Ibrahim, M. Teaching Future Teachers to Use Digital Toys in the Classroom. *In proceedings of Society for Information Technology and Teacher Education International Conference (SITE)*, Chesapeake, VA, ACM, New York, USA, pp. 2168-2173, 2007.

[Yatim07b] Yatim, M.H.M. and Masuch, M. Gatelock: A Game Authoring Tool for Children. *In proceedings of Interaction Design and Children Conference (IDC)*, ACM, New York, USA, pp. 173-174, 2007.

[Yatim07c] Yatim, M.H.M. and Masuch, M. Educating Children through Game Making Activity. *Paper presented at Game in' Action*, Göteborg University, Sweden, 2007.

[Yatim07d] Yatim, M.H.M. Principles of Interface Design for Edutainment. *Paper presented in Interface Design for Edutainment Seminar*, Otto-von-Guericke University of Magdeburg, Germany, 2007.

[Yatim08] Yatim, M.H.M. Evaluating Usability and Fun in a Game Authoring Tool. *In proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications (Ed-Media)*, Vienna, Austria, 2008.

[Yatim09a] Yatim, M.H.M., Sharif, A.S., and Fathil, N.S. Game Authoring Tool for Teaching and Learning: A Study on Usability and Fun. *In proceedings of National Conference on ICT in Education*, Ipoh, Malaysia, 2009.

[Yatim09b] Yatim, M.H.M. and Dachselt, R. Lessons Learned in Conducting User Studies for Children' Software. *To be submitted to the Interactive Computer Aided Learning (ICL)*, 2009.

[Yatim09c] Yatim, M.H.M. and Dachselt, R. A Survey of Educational Programming Languages for Children. *To be submitted to AACE Journal of Interactive Learning Research*.

[Yatim09d] Yatim, M.H.M. Designing a Game Authoring Tool for Children: A Child-Centered Approach. *To be submitted to the 21st Annual Conference of the Australian Computer-Human Interaction (OZCHI)*, 2009.

[Yatim09e] Yatim, M.H.M. and Fathil, N.S. A Study of Children and Icon Recognition: Evaluating Icon Design for a Game Authoring Tool. *To be submitted to the 21st Annual Conference of the Australian Computer-Human Interaction (OZCHI)*, 2009.

[Ye04] Ye, J. and Ye, D. HCI and Game Design: From a Practitioner's Point of View. Found without any publishing note. 2004. http://www.ye-brothers.com/documents/HCIGAMEDESIGN.pdf Last accessed: May 2009.

[Yokokawa06] Yokokawa, K. Script Synthesis Tool for Non-Experienced Programmers. *In proceedings of Creating, Connecting and Collaborating through Computing (C5)*, pp. 218-221, 2006.

[Yoo06] Yoo, S., Kim, K., Yeurn, Y., Kanemune, S., and Lee, W. Empirical Study of Educational Programming Language for K12: Between Dolittle and Visual Basic. *International Journal of Computer Science and Ntework Security*, Vol. 6(6), pp. 119-124, 2006.

[Yusof02] Yusof, R.J.R. and Singh, S.K.K. Guidelines for Designing a Usable Educational System. *In proceedings of the IEEE International Conference on Advanced Learning Technologies, Media and the Culture of Learning*, Kazan, Russia, pp 402-407, 2002.

[Zaman05a] Zaman, B. Evaluating Games with Children. *In proceedings of Interact Workshop on Child Computer Interaction: Methodological Research*, Rome, 2005.

[Zaman05b] Zaman, B. and Geerts, D. Gender Differences in Children's Creative Game Play. *In Young People and New Technologies*, UK Northampton, 2005.

Acronyms

ACM	Association for Computing Machinery (http://www.acm.org)
CBL	Computer-Based Learning
CCD	Child-Centered Design
CCI	Child-Computer Interaction
EPLs	Educational Programming Languages
FPS	First-Person Shooter
GML	Game Maker Language
GUI	Graphical User Interface
HCI	Human Computer Interaction
HEP	Heuristic Evaluation for Playability
ICT	Information and Communication Technology
IEEE	Institute of Electrical and Electronics Engineers (http://www.ieee.org)
ISO	International Standardization Organization (http://www.iso.org)
LCD	Liquid Crystal Display
MVC	Model-View-Controller
OLPC	One Hundred Dollars Laptop per Child
OOP	Object-Oriented Programming
PC	Personal Computer
QUIS	Questionnaire for User Interaction Satisfaction
TPS	Third-Person Shooter
UCD	User-Centered Design
UI	User Interface

Declaration

I declare that this dissertation has been written by me. All published or unpublished materials used here have been given full acknowledgement.

Magdeburg, June 2009

_____

Maizatul Hayati Binti Mohamad Yatim

CD

This CD contains the following:

/ DISSERTATION	A digital version of this dissertation (in *.pdf format).
/ DOCUMENTS	Compilation of documents (in *.pdf format).
/ IMAGES	Compilation of photographs and images.
/ GATELOCK	Two versions the game authoring tool designed throughout the research: (1) *GatelockOld* presenting the old interface design of Gatelock and (2) *GatelockNew* presenting the new interface design of Gatelock .