LEARNING GAMES FOR PROGRAMMING

A MASTER THESIS

by

Henrike Lode

Giuseppe Enrico Franchi

Niels Gamsgaard Frederiksen

Supervisors: Prof. Rilla Khaled and Pippin James Barr

A Thesis Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science of

Media Technology and Games

at IT University Copenhagen, September 2012

Acknowledgments

We would like to thank

Samatha Fern and Kristian Jespersen
from Copenhagen International School
for enabling tests with our target audience

and

Peter Blanchfield, James Paul Gee, Peter Mozelius, Ole Pedersen,
Ilse Schmiedecke and Thomas Wernbacher
for their feedback on our game Machineers

and

Café Analog for the coffee and the comfy sofas.

ABSTRACT

Many scholars, talking about both educational games and general educational practice, argue that principles such as 'agency', 'immersion', 'intrinsic motivation' and 'fantasy' are paramount for the pupils to actively engage with the learning activity. Games have the power of stripping the pressure coming from judgement, evaluation, intra-classroom comparison and competition, creating a safe environment where the principles mentioned above can enable the player to undergo a deep and valuable learning experience. And yet, educational games often fall short when it comes to apply those principles and engage their players.

Especially learning games that teach programming often lack immersive settings, are weak in terms of compelling game-play and sometimes even require prior knowledge of syntax or coding. The more that computers and programming have become a part of everyday life, the more a profound literacy in computer science has become necessary and therefore it should be introduced at an early age, using suitable visual metaphors and providing adequate intrinsic motivation.

In this thesis we present the design and evaluation of a learning game based on the concept of 'stealth learning' and the theory of constructivism, which provides an early introduction to basic programming concepts and procedural literacy to children from 10 years upwards, attempting to provide the highest possible degree of immersion. The evaluation involves a number of tests verifying usability, immersion and motivation, and an attempt to test the transferability of the in-game acquired knowledge to reading pseudocode, the results of which, triggered further reflection on the possibility of adding new layers of complexity and embedding evaluation of learning within the game experience.

Content

1 Introduction

1.1 Research Question(s)

The low production quality in edutainment titles compared to commercial video games as well as the less engaging gameplay has lead to a general public aversion to the term ‘edutainment’ and a negative mindset during play which possibly affects the learning experience and results in a resistance to pretests and posttests (Egenfeldt-Nielsen 2007, Honey & Hilton 2011). It seems like many children, who have experienced disappointments in the classroom develop a negative attitude towards learning in general and dismiss learning games as a ‘trick’ to fool them into enjoying something they know for a fact is not fun (Honey & Hilton 2011). On the other hand, concealing the learning purpose of the game can hinder the learning experience and the transformation of spontaneous concepts into scientific concepts (Papert 1998, Egenfeldt-Nielsen 2007).

> *Can we create a positive mindset for the learner and provide a base for experiential learning, if we present a learning game that does not advertise its purpose and is not distinguishable from commercial games in terms of gameplay and visual quality?*

After some initial attempts in the 80s, there is still only marginal teaching of computer literacy and programming in schools. Specifically, there are only a few learning games teaching programming, none of which are free from shortcomings in the gameplay design. Not only is it important for children to achieve a proficient literacy in reading, writing, math and natural sciences, but considering the increasing use of computers and programming in different professions, it is vital for them to develop procedural literacy, “*the ability to read and write processes, to engage procedural representation and aesthetics, to understand the interplay between the culturally-embedded practices of human meaning-making and technically-mediated processes*” (Mateas 2005, p.1) as early as possible.

> *This project is aimed at identifying and practically applying a series of design principles relevant to our case, then analyzing the process of children dealing with procedures in an immersive and engaging learning environment created for this specific purpose.*

1.2 Motivations for this work

The lack of experience-based learning is evident in the educational system when we consider history, geography, citizenship or religion. We are not actually basing the learning in school on concrete experiences with a given topic, but are primarily relying on students reading or hearing about topics mostly represented by abstract information and concepts with little connection to an actual experience base. This is not a criticism of reading or hearing, as these are strong teaching tools, but rather a challenge of the balance between these and other approaches in education. The educational system seems to entertain the fantasy that we can skip the concrete experiences altogether. (Egenfeldt-Nielsen 2007, p.94-95)

Learning games are generally unhesitating about their purpose, which makes them easy to advertise to parents and schools, but not necessarily to children, the intended target audience. As we will further explain in section 2.5.6 Clear Goals, attaining game goals is a tangible need for children when they are engaged with them, while receiving verbal and out-of-context information when learning about abstract topics is seen as something with little impact on their life, and therefore not really relevant. If the learning content is not carefully woven into the design of a learning game, the tangible need of playing the game might be overshadowed by the feeling of practicing some abstract and irrelevant skill in just a different form. For these reasons, our learning content is not explicitly presented to the player, but embedded in the game challenge.

Investing into a positive mindset by enabling the player to immerse themselves into a game world gives educational games the potential to overcome many great learning obstacles. We assume that players are more likely to engage themselves deeply with the learning content if they can chose the learning pace which they are most comfortable with without being compared to their classmates or being observed and possibly evaluated by their teachers.

Defending and resisting learning may be as much social as it is cognitive; children who see themselves as failing may "drop out" of the enterprise because they no longer see themselves as members of the class or group. Whether they are sidelined by themselves or by others the result may be a complete unwillingness to try. (Olson 2007, p.41)

Following the line of thought that attempting to measure and compare learning outcomes of their students restrains the learning process, we steer away from the traditional school approach. In order to create fertile ground for learning, the learners must first and foremost immerse themselves in a playful and secure learning environment, build up self-esteem and a sense of accomplishment and start perceiving themselves as empowered learners. To accomplish this within our project we want to use the *stealth learning* approach, which means that we attempt to engage the learner in a puzzle adventure game world to establish intrinsic motivation and absorb the players attention so much, they would almost not realize that they are learning. We anticipate that once the player has the right motivation and a real purpose for applying their knowledge, they will understand the need to learn the content, so the learning purpose can be slowly revealed in order to enable transferability of knowledge. This way we would integrate stealth learning with experience-based learning.

Commercial games are good at delivering a learning experience in a limited time period that is individualized by each player (Papert 1998). It is important that not only regular gamers are taken into account as potential players, but also people who never play videogames at all and who might need more confirmation and help. Learning games need to work for every player type to be used successfully in a classroom or as a tool designed to help children on their own. It is important to make sure that girls feel comfortable playing the game as well as boys.

Learning is related to self-esteem and personal interests in a way that children avoid possible situations of failure as it lowers their self-esteem and refrain from engaging with content they have no personal relation to or interest in. But failure is a part of learning - it shows the learner the flaws in their mental model and helps them to adjust it. Learners must "*experience success and failure not as reward and punishment, but as information*" (Bruner 2006, p.62). Failing helps us to fully understand how something works in its specific way, therefore learning games need to present failures as a learning opportunity without judgment, punishment, or other negative attributes. Carefully balancing game challenges and rewards is an important part of this. If a game is not challenging enough, the players will not experience a feeling of great achievement which could potentially feed intrinsic motivation; on the other hand, if the game is too hard there is a chance players can get stuck, experiencing personal failure and eventually leave a game with a deteriorated self-image and confidence.

We chose to use programming concepts as inspiration for our game-design to show that any subject, no matter how abstract, can be integrated into good game design and engaging gameplay suitable for any player type without placing initial barriers (e.g. prior knowledge to play). Most people start to learn programming in college and experience it as a very tedious and time-consuming process. But learning how to program simultaneously teaches how to learn, how one's thought processes work and how they differ from computers, and to embrace failure as means to deeper understanding. The world we live in relies to a large extent on the use of computers, and virtually all professions of higher education involve computer technology and programming to a certain degree. Considering the great impact computer technology has on our daily lives, procedural literacy needs to be introduced to students as early as possible.

The product we create is aimed to demonstrate how good game design can constitute a beneficial "preparation for future learning" (Bransford & Schwartz, 2001). With this project we also want to establish an appropriate balance for a broad audience by providing the players with a selection of on-demand scaffolds in order to cater for the more ambitious player, who might try to solve problems without accessing any help options, while more cautious and self-aware players will be guided as closely as possible, to avoid game-stopping experiences.

1.3 Synopsis

The contents of this thesis are presented in three parts: the theoretic background to this project, the design of a learning game that combines good game design principles with learning content, and in the third part the test and evaluation of this learning game.

The theoretic chapter focuses on existing literature. In the first part we cover general research about educational games and existing programming-teaching tools. We outline the foundations of constructivism and scaffolding, then we proceed exposing what procedural literacy is and why it is relevant in modern education. We proceed outlining a number of prominent design principles that emerge from play theory, as well as how we can use play theory to understand how these principles can facilitate the learning process.

The next chapter discusses in detail how we used the learning principles described in the first part to design an educational computer game, its scaffolds and intended learning curve, as well as how we use high qualitative graphics and an enthralling storyline to provide a proper level of immersion and motivation.

The third part describes in detail how we planned and conducted tests to evaluate our learning game, discusses how some design features needed to be adjusted to improve usability and immersion, and concludes with a thorough evaluation of successes and failures of our approach.

2 Background

2.1 Previous research

This chapter deals with previous work concerning learning games, learning theories in general as well as procedural literacy and game design principles that can affect the learning experience in educational titles.

2.1.1 Three Generations of Learning Games

Egenfeldt-Nielsen (2007) is the father of the three generations of educational computer games model, which we will use as a base to start the discussion. The first generation mainly involves drill-and-practice games that use a behaviorist approach to condition the player, while the second generation games enhance the motivation of the learner by incorporating player attention, focus, curiosity and fantasy. They require the player to develop deeper understanding in accordance with the cognitivist approach. Egenfeldt-Nielsen presents then a third generation of instructional games, which are based on a constructivist approach, meaning the player creates knowledge through experiences and interaction with social communities.

> *The construction of knowledge, as meaningful through orientation in a social context, becomes paramount in the third generation. Instead of conceiving content, skills and attitudes as residing within the user, knowledge is transferred to culture, tools and communities. [...] One learns new things by participating in these communities, appreciating and negotiating what counts as knowledge, skills and attitudes. (Egenfeldt-Nielsen 2007, p.88)*

In '*Overview of research on the educational use of computer games*' (2006), Egenfeldt-Nielsen analyzes edutainment titles based on a behaviorist approach, usually using a drill-and-practice approach and relying on extrinsic rewards. Discussing the game '*Math Missions Grades 3-5: The Amazing Arcade Adventure*' he demonstrates an example of this approach and its gap between gameplay and educational content. In this game, players earn points for correctly answered math questions which they can spend on playtime in an arcade. Egenfeldt-Nielsen grants success to this approach e.g. for health games, talking about a study from Debra Lieberman that compared enjoyment and learning outcome (everyday health management) after watching a 30-minute long video or playing a 30-minute video game about asthma and found that the testers learned the same while enjoying the game more, therefore repeating it multiple times (Lieberman, 2001). Still, these type of games are strongly criticized by children, parents, educators and researchers for non-satisfactory gameplay, learning principles and graphics. The nature of drill-and-practice games lead to mechanical memorization of content or actions, not to a deeper understanding or transferable knowledge (Egenfeldt-Nielsen, 2006).

Educational games that use the cognitivist approach attempt to integrate learning and game experience and therefore establish intrinsic motivation, which is a person's self-motivation to pursue a task for which no external reward is necessarily granted. For the cognitivist approach it is important to present the learning content in different ways, while considering limitations and potentials of the human mind by providing multiple meaningful contextualized activities, visualization, manipulation, feedback, etc. Cognitivism focuses more on learned skills than content, as well as on meta-skills like problem-solving which have received much attention from researchers. Egenfeldt-Nielsen argues that transfer of these skills to other areas then computer games is difficult. Studies from a cognitivist perspective are limited, but the math teaching games *Super Tangrams* and *Phoenix* have been proven very effective in motivating and teaching math to students (Egenfeldt-Nielsen 2006).

The constructionist approach to teaching has been brought forward by Seymour Papert, building up on Jean Piaget's constructivism, as demonstrated by the programming language Logo (Feurzeig & Papert 1967), which is used to enable students to computer-generate drawings using mathematical concepts. For constructionism it is important that the learner actively creates knowledge using external artifacts. Educational games using the constructivist approach are commonly called microworlds and constitute open-ended universes or sandboxes, more simulations than games. Through manipulation of and interaction with objects, the learner discovers their properties, connections and applications. The focus lies again not so much on content, as on skills like problem-solving, critical-thinking, sequential planning and creativity. Rather than transfer, educational games should focus on helping the player to engage with material, talk and think about it. Constructivism ultimately aims to turn the learners into creators of content by enabling them to e.g. design games and thus, acquire knowledge about math and programming (Egenfeldt-Nielsen, 2006). The need to integrate the learning content into the world makes it impossible to use a universal design formula, but instead learning games following this approach have to be designed anew for every topic.

The socio-cultural approach relies on the Vygotskian concept of proximal development which will be explained in more details in a later section. In the "proximal development" metaphor, the learner is guided "*from an actual point of development to a potential point of development*" (Egenfeldt-Nielsen 2006, p.199), thus focusing on the activity and on "*the possibility for the player to invest something of himself in the game*" (Egenfeldt-Nielsen 2006, p.200). The learning goal should be reached by exploration of relationships between objects. Scholars recognize the power of constructivist and experience-based approaches applied to learning games: for example Gee (2005c, p.14) when arguing that humans *"think through experiences they have had and imaginative reconstructions of experience"*. Games are tools for constructing experiences, as "*video games are interesting not for their content but for the way new explorations initiate negotiations, constructions, and journeys into knowledge*" (Egenfeldt-Nielsen 2006, p.200).

2.1.2 Further research on learning games

In '*Learning Science Through Computer Games and Simulations*' (Honey & Hilton, 2011) the National Research Council presents a review of the research work on educational games and simulations. The report identifies a number of goals and design features that influence science learning and need to be considered when creating computer games teaching science topics. Motivation of players is mostly based on identity, play, immersion, social relationships, and a strong narrative. To date there is little research concerned with the question if computer games actually can affect the players ability to lead scientific discourse, but it indicates that when proper scaffolds are provided, they are well suited for enabling players to engage in serious argumentations and reasoning, and to understand and use scientific terminology correctly. Further research claims that computer games can even provide a new identity for players with low self-efficacy in terms of science skills which alters their mindset in a way that enables deeper engagement and motivation (Barab & Dede, 2007). This is consistent with Gee's concept of *identity* and *authentic professionalism* (Gee, 2005b). However, there is no proof that this identification with a scientific professional persists for a longer time.

As a general critique on existing research about computer games, Honey and Hilton state that it is mostly insufficient, as it often has not been conducted very systematically. Frequent and significant changes in technology and platforms lead to games and simulations changing fast, making it hard to establish a shared research approach. Also the context is not always clear - what is unique about that game, how does it contribute to the learning process, is it designed for a home or school setting? Studies hardly ever inquire about prior knowledge of the testers and experiences with computer games, the tested groups often have little diversity, so results cannot be generalized. Studies show a wide range of focus, methods and theoretical learning perspectives, that they are based on. There is no ground for a coherent base of research across studies and over time. Egenfeldt-Nielsen also strongly criticized basically all of the studies he analyzed for a lack of proper test methods (e.g. some studies tried to compare 40 minutes of teaching to 20 minutes of gameplay) and not basing their studies on sufficient prior research, thus constantly "re-inventing the wheel".

In '*Does Easy Do It? Children, Games and Learning*' Seymour Papert (1998) argues that edutainment titles often replace the positive features of games with the negative ones of traditional schooling, resulting in a product which convinces maybe parents and teachers, but not the children whose learning experience is supposed to be 'easier' through playing those titles. His point is that the term 'easy' is never a good selling point for games, as the challenge is what makes them fun. He further argues that such an approach is immoral, "*to trick children into learning and doing math when they think they are just playing an innocent game*" (Papert 1998, p.2), as well as contradicting, because their learning experience would be much more effective if they were aware of the hidden agenda. As mentioned by Simon Egenfeldt-Nielsen, a conflict arises here between a predetermined mindset that might interfere with the receptiveness of the learner and the transferability of learned content (Egenfeldt-Nielsen, 2007). To circumvent both problems, we suggest to keep the learning purpose of the game hidden until the intrinsic motivation of the player is strong enough to value the learning content, as the learner then has a useful application for their knowledge available. By slowly revealing the learning content embedded, the risk of losing players who are biased against learning games can be reduced.

Papert's message to learning game developers is "forget about making games to teach children multiplication or spelling or any of those old-fashioned basic skills. The really basic skill today is the skill of learning, and the best use of games is to leverage their tendency to enhance it."

(Papert 1998, p.2). He further recommends to debrief children after game experiences and engage them in conversations about learning as much as possible, as well as turn them into game designers themselves, and advises game designers to empower children as independent learners.

2.1.3 Programming-teaching tools

The Logo programming language (Feurzeig & Papert, 1967) is considered the first real attempt at teaching computer programming to a young audience. Children would use Logo to draw shapes by programming a cursor to move around on a display leaving a trail wherever it went. This cursor was referred to as the *'turtle'*, and would later be implemented as a physical robot, drawing lines on the floor. Being able to see exactly how the turtle behaved made it easy for children to debug their program and they could easily verify their code visually. Logo also encourages users to think procedurally since many shapes require multiple executions of the same lines of code. For example drawing a pentagram would require the turtle to go forward a set amount and turn 144 degrees five times.

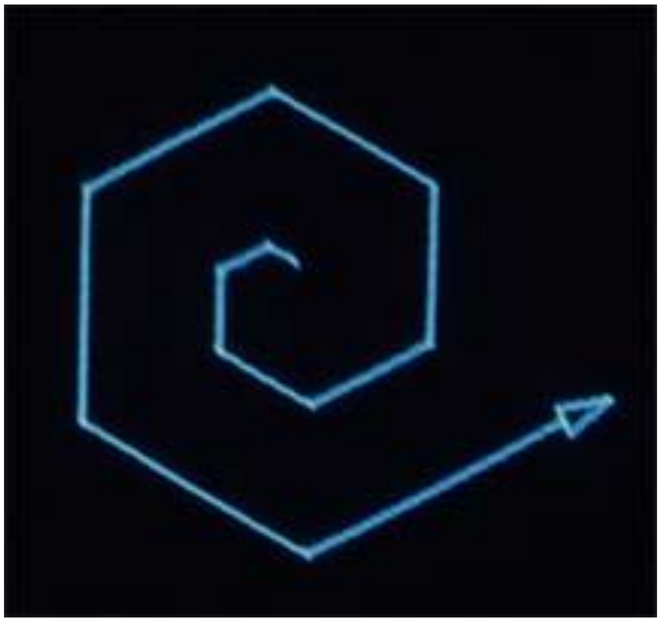
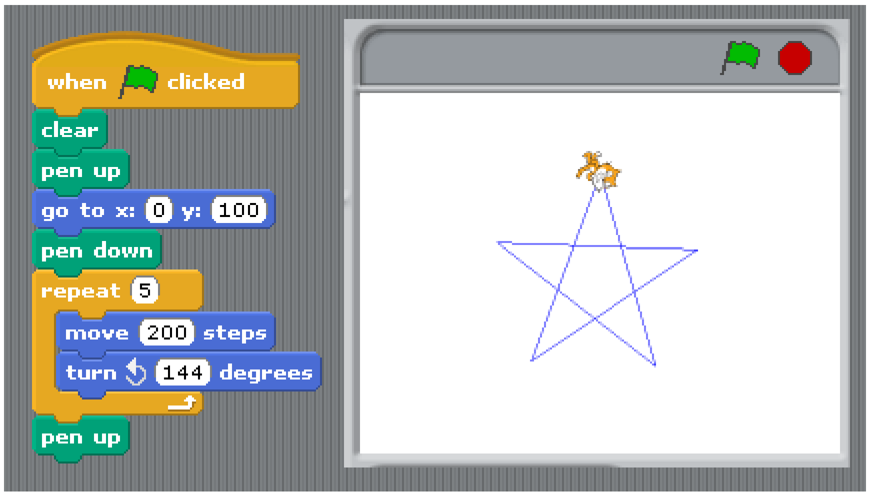


Image 2.1 and 2.2 - The Logo turtle graphics and its implementation as a robot.

Even though Logo syntax is not case-sensitive it still requires the user to learn the correct syntax in order to begin programming. Later, games like Scratch (Lifelong Kindergarten Group, 2006) and Alice (Carnegie Mellon University, 1999) have taken what is good about Logo but have exchanged the writing of code with a drag and drop visual programming language, thereby eliminating the need for understanding syntax.



```
when [flag] clicked
clear
pen up
go to x: 0 y: 100
pen down
repeat 5
    move 200 steps
    turn ↺ 144 degrees
pen up
```

Image 2.3- Scratch visual programming

The toy company Lego also created a line called Lego Mindstorms (The LEGO Group, 1998) named after the book "*Mindstorms: Children, Computers, and Powerful Ideas*" by Seymour Papert. With Lego Mindstorms children can build robots out of lego and then program them on a computer much like the old Logo robots, but Mindstorms uses a visual programming language similar to that of Alice and Scratch.

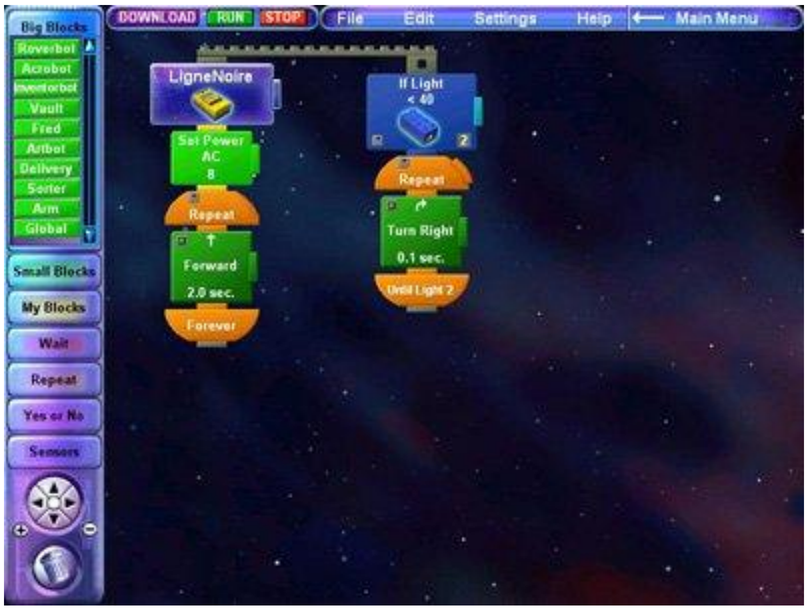


Image 2.4 - The Lego Mindstorms visual programming

Even though Lego Mindstorms, Scratch and Alice are good tools for teaching programming to children, we cannot really call them games. They do not have a specific goal and they act more like an environment for the children to play around in.

Lightbot (Armor Games, 2008) is a puzzle game that takes the drag and drop programming concept and puts it in a game with specific goals. In this game the player has to direct a robot through a series of levels using a limited set of commands. The player drags a series of commands into the main method and starts the program, and the robot then executes these commands in the order they were placed.

As the player progresses in the game, new commands become available, and eventually the player also gets the ability to define and call up to two functions. The player defines what the functions do, but since the main method can only hold a finite number of commands the player is forced to think of ways to place repeating series of commands into functions to save space in the main method. In the sequel Lightbot 2.0 (Armor Games, 2010) they added a feature showing where, in the series of commands, the robot is currently at which helps a lot in understanding flow control. Lightbot 2.0 also introduces concepts like recursion and conditionals.

Joe Rheaume talks about LightBot in his article: "*Light Bot: Learning Objectives and Game Elements"* where he claims that LightBot is does a good job at teaching programming because "*Learning should come from the game elements themselves*" (Rheaume 2009) and this is exactly what LightBot does. The game seems to have received little attention from scholars.

Learning games for programming keep evolving and a new game called Code Hero is currently under development by Primer Labs. Code Hero will be a first person shooter in which the player has to navigate and manipulate the game world using code. Equipped with a "code gun", the player can write the code that the gun will apply to the objects it hits, manipulating them in funny and creative ways. The learning potential is still unknown since the game is currently under development but it is interesting to see new and innovative approaches to the genre.

2.2 Constructivist learning and scaffolding

Pedagogy must be oriented not to the yesterday, but to the tomorrow of the child's development. Only then can it call to life in the process of education those processes of development which now lie in the zone of proximal development (Vygotsky 1993, p.251-252).

Developed from Jean Piaget's model of intellectual development, the constructivist approach to learning assumes that the nature of knowing relies in adapting received knowledge with experience. In a process that is active rather than passive, learners adapt their inner mental models to accommodate new informations received, constructing their own inner knowledge (Bodner 1986). Heavily influenced by Piaget, Jerome Bruner, a founder of Constructivism, clearly states that "*a principal task of intellect is in the construction of explanatory models for the ordering of experience*" (Bruner 2006, p.113). Olson summarizes Bruner's view in saying that "*knowledge and intelligence are indistinguishable. Intelligence is not thought of as an inborn talent but rather as a description of an educated mind*" (Olson 2007, p.35). Acquiring new structures allows for a better basis for understanding and therefore learning new concepts, in a virtuous circle. Knowledge is more viewed as process rather than content.

Born in 1896, the same year as Jean Piaget, Lev Vygotsky's interest was focused around the social system in which learning takes place and how it influences the learning process (Daniels, 2001), but he was also an influential constructivist and the creator of the metaphor of the Zone of Proximal Development (ZPD).

The ZPD is one of the most fascinating and renowned concepts of constructivist learning, "*often cited as Vygotsky's most profound contribution to educational debate*" (Daniels 2001, p.56). Such a metaphor represents the distance between the set of tasks the learner is able to handle alone and the ones they cannot complete by themselves yet, but which are just within reach if supported by a tutor. As defined by the author, the ZPD is the "*actual developmental level as determined by independent problem solving and the higher level of potential development as determined through problem solving under adult guidance or in collaboration with more capable peers*" (Vygotsky 1978, p.33).

This concept, as later explained by the author, implies that "*instruction is only useful when it moves ahead of development. When it does, it impels or wakens a whole series of functions that* *are in a stage of maturation lying in the zone of proximal development*" (qtd. in Daniels 2001, p.58). It is only when learners get instructed in something they are not yet capable of mastering on their own that their skills mature and potential knowledge becomes actual knowledge: in order to develop skills, learners tackle tasks that are beyond their current capabilities (Verenikina 2003).

One of the most important roles of the teacher, according to Vygotsky, is therefore to support the learners in their problem-solving tasks. A major pedagogical implication of this concept is that the focus of the educational effort and assessment should be placed on the potential level rather than the actual level of performance (Daniels, 2001). The concept of an expert - an adult or a more capable peer - guiding an apprentice into mastering new skills will be the foundation of Bruner's theory of scaffolding. Vygotsky was more interested in "*assessing the ways in which learners make progress*" (Daniels 2001, p.58) rather than providing precise guidelines about how to help learners achieve such a result.

Simon Egenfeldt-Nielsen contributes in explaining constructivist learning process in more detail by mentioning Vygotsky's differentiation between scientific concepts and spontaneous concepts. These two terms refer to "*concepts learned through education and concept that emerge more spontaneously in any setting*" (Egenfeldt-Nielsen 2007, p.100). "Spontaneous concepts", in fact, defines concepts that form "*from the bottom-up, the spontaneous ordering of experience"*, while "*scientific concepts make the student capable of higher mental functioning and are characterized by being systematic, general, abstract and organized*" (Egenfeldt-Nielsen 2007, p.100). When marking the difference between spontaneous and scientific concepts, Egenfeldt-Nielsen stresses how there is no difference in relevance between the two, and how Vygotsky found "*scientific concepts to be ahead of spontaneous concepts if instruction is appropriately based on an experience base*" (Egenfeldt-Nielsen 2007, p.100). In this sense, the effort of the teacher following the constructivist approach is to facilitate the transformation, and avoid forcing the mere scientific facts upon the students:

> *With no experience base to build the scientific concepts, students will merely learn 'parrot-like repetition'. [...] The important point is that it is not hard to teach a student the word mercantilism or to give some rudimentary understanding of it relating to trade.*

However, the underlying concept and, in particular, the transfer to other situations becomes hard. (Egenfeldt-Nielsen 2007, p.100)

The assistance to be given in relation with Vygotsky's concept of ZPD was labeled *scaffolding* by Jerome Bruner. As in the construction of a building a scaffold is an external support that allows a structure to be raised or improved, an instructional scaffold is defined as follows:

Scaffolding consists essentially of the adult "controlling" those elements of the task that are initially beyond the learner's capacity, thus permitting him to concentrate upon and complete only those elements that are within his range of competence. (Wood et al. 1976, p.90)

Guidelines for the scaffolding process are then outlined. Recruitment is introducing the task and getting the learner interested in it. *Reduction in degrees of freedom* is defined as "s*implifying the task by reducing the number of constituent acts required to reach solution*" (Wood et al. 1976, p.98). It is important to underline how this does not imply simplifying the task as a whole. Rather, a scaffold allows the learner to initially have a smaller subset of tasks to focus upon, that can be autonomously solved (Daniels, 2001). The adult provides flexible support, and control has to be gradually passed on to the learner when making progress. *Direction maintenance*, keeping the learner on track, both keeping the learner focused towards the goal and providing motivational support. *Marking critical features* helps the learner identifying the crucial points of the task, while *frustration control* keeps motivation high, but with the necessary precaution of avoiding the learner to depend too much on the instructor. Finally, the last principle is *demonstration*, which is modelling and presenting a task's solution.

The scaffolding technique applied to computer-based learning environment has received an increasing amount of attention from scholars and researchers in the last decade (Azevedo & Hadwin 2005). Providing external scaffolding is listed amongst the design features that influence learning in Honey and Hilton (2011), given that clear learning goals are outlined. Identifying clear learning goals is in fact the prime condition for applying good scaffolding techniques, as reinforced by Gee (2005a) when calling for well-ordered problems instead of giving complete freedom to explore a broader problem space. A good learning game will not leave the player unguided in their exploration of the game but present them with well-ordered problems so they are optimally scaffolded during their learning experience. Help options in a learning game should be provided just-in time (when the player can understand and use it) and on-demand (when they want it). Keeping in mind the constructivist approach, Gee also calls for meaningful scaffolds, warning about providing support in the form of '*words for words*'. Describing a concept and its implications in mere words leads to a poor understanding if the learner cannot connect them to their experiences. Good video games explain terminology with images, actions and provide a context for situated meaning, which is explaining what a concept stands for and implies by placing it in a meaningful situation.

2.3 Experiential Learning

Experiential learning focuses more on the process and feedback than the outcome of learning. "In its simplest form, experiential learning means learning from experience or learning by doing." (Lewis & Williams 1994, p.5). Learners are encouraged to immerse themselves in an experience, then develop new skills by reflecting on the process.

Kolb defines learning as "*the process whereby knowledge is created through the transformation of experience*" (Kolb 1984, p.38) and shapes a four-stages model called the '*experiential learning model'*. Such model starts from the concrete experience and continues with observation and reflection, abstract conceptualization, and active experimentation of the new concept in new situations. "*This in turn leads to another set of concrete experiences and another round of learning at a more sophisticated level*" (Lewis & Williams 1994, p.6).

In this context, educational games can offer experiences for subjects which usually do not allow for experiential learning in school settings, such as history, geography or religion, where learning activities are mostly passive and involve reading or hearing.

2.4 Procedural Literacy

Michael Mateas defines procedural literacy as "*the ability to read and write processes, to engage procedural representation and aesthetics, to understand the interplay between the culturally-embedded practices of human meaning-making and technically-mediated processes.*" (Mateas 2005, p.1). Ian Bogost suggests that "procedural literacy entails the ability to reconfigure basic concepts and rules to understand and solve problems, not just on the computer, but in general" (Bogost 2005, p.32).

Learning how computers read and execute code and thinking in terms of procedures are becoming important skills for more and more people as computers are continually being introduced into professional workflows. Today we use a wide range of technologies in our everyday lives and if we want to be comfortable with them and use them in an optimal way, then we have to become procedurally literate. Unfortunately the education of procedural literacy is not keeping up with this constantly increasing demand for procedurally literate people and Mateas has argued that we should start teaching procedural literacy as a mandatory part of the curriculum earlier in the educational process and as a part of all higher education. Mateas suggests that: "*To achieve a broader and more profound procedural literacy will require developing an extended curriculum that starts in elementary school and continues through college"* (Mateas 2005, p.14).

The term 'procedural literacy' comes from a presentation by B. A. Sheil at the Xerox Palo Alto Research Center in 1980. Here Sheil talked about how beginners in programming have an easier time understanding certain concepts of programming because the concepts have similar properties to concepts they know from other aspects of life.

> *The beginning programming student brings a rich structure of semi-procedural knowledge, based on previous experience with instructions, directions for getting places, recipes, etc. which can be used as a basis for teaching procedural skills." (Sheil 1980, p.125)*

Similarly Ian Bogost talks about how children's toys like Lego and Playmobil can also help teach the basis of procedural literacy to children at a very young age.

> *[...] playing with Legos gives children exposure to procedural literacy. Lego bricks are a bit like the "basic building blocks" Wise and Bauer relegate exclusively to the terrain of language. Legos recombine in multiple multiples to create new, previously unpredictable meaning. (Bogost 2005, p.35)*

Both Sheil and Bogost seem to think that early exposure to procedural thinking can help a child develop the groundwork for procedural literacy and improve their learning process when they get older. In his paper "*Procedural Literacy: Problem Solving with Programming, Systems, & Play*" Bogost looks at procedural literacy in a new light. Procedural literacy is not only the ability to program a computer, but instead an essential set of skills that can be used to solve any logical problem. Bogost compares it to the study of Latin which is encouraged in classical education because of its strict set of rules which train the mind to think in a more structured and systematic way. He also talks about how certain games can teach procedural literacy.

> *[some games] suggest that procedural literacy can be cultured not only through authorship, such as learning to program, but also through the consumption or enactment of procedural artifacts themselves. In other words, we can become procedurally literate through play. (Bogost 2005, p.34-35)*

Papert observed that children can have a very good understanding of the subject even when they are lacking the proper vocabulary to express themselves, which is often superior to knowledge learned at school (Papert 1998). Jerome Bruner as well believed that a topic can be approached and learned by a student of any age. In a famous statement he talks about “Readiness for Learning”, claiming that “*any subject can be taught effectively in some intellectually honest form to any child at any stage of development*” (Bruner 1960, p.33), acknowledging this to be a bold hypothesis, although “*no evidence exists to contradict it*”. There is no intrinsic difficulty in a particular topic: specifically, Bruner argues that delaying the teaching of certain subjects seems arbitrary and even incorrect: fundamental notions of Euclidean geometry or physics are “*perfectly accessible to children of seven to ten years of age*” (Bruner 1960, p.33) given the fundamental premise that “*they are divorced from their mathematical expression and studied through materials that the child can handle himself*” (Bruner 1960, p.33). The teaching material has to be based upon the learner’s experience; the right questions - what he calls “medium questions”, need to be presented to the child. Intuitive understanding of the logical foundations proper of a certain subject can to be established, or at least introduced before the body of knowledge is presented to the learner. If we accept Bruner’s point of view, it is easy to see a strong argument in promoting procedural literacy. The logical foundations of computer programming can be introduced at an early age, given that the learning material is put in a form that the learner can grasp.

2.5 Design Principles

In this section we are going to outline some of the design principles for educational games we found particularly relevant in our design approach.

2.5.1 Agency

James Paul Gee suggests that in order to create a basis for a good learning experience the players have to connect with the character they are playing in a way that allows them to take on a new identity in order to internalize the actions and achievements of their avatar. All actions and terminology are situated in an "*interactive relationship between the player and the world*" (Gee, 2005a, p.5). The player should be encouraged to experiment and take risks without having to face the severity of consequences those actions would have if they failed in the '*real world*' (Gee, 2005a).

In '*What would a state of the art instructional video game look like*', Gee (2005b) reinforces this idea by explaining the term '*authentic professionalism'* as a play experience where the learner is able to connect to their virtual character in a manner that allows them to take on their identity and internalize their skills, knowledge and values in order to understand and empathise with the way they think, behave, and solve problems from their perspective as a professional.

> *With authentic professionalism, 'knowing' is not merely the mastery of facts; rather knowing involves participation in the complex relationships between facts, skills, and values in the service of performing a specific identity. Here, word and deed are united and the knower is a knower of a specific kind - a type of active professional, not just a generic recipient of knowledge. (Gee, 2005b)*

Gee explains that games can give the player more realistic experiences than e.g. role-play exercises within a school setting. For example the experience of campaigning to become a mayor in an online multiplayer game world enables the player to realistically experience a powerful identity. Furthermore, the player should be enabled to customize the game according to their needs and preferred learning and playing styles. Being able to identify with their character and having control of their actions creates a strong feeling of agency, which facilitates the learning process (Gee, 2005b).

> *Video games based on the training of socially-valued practitioners let us begin to build an educational system in which students learn to work (and thus think) as doctors, lawyers, architects, engineers, journalists, and other important members of the community - not in order to train for these pursuits in the traditional sense of vocational education but rather because developing those epistemic frames provides students with an opportunity to see the world in a variety of ways that are fundamentally grounded in meaningful activity and well aligned with the core skills, habits, and understanding of a postindustrial society. (qtd. in Gee 2005a, p.12)*

Allowing the player of a video game to assume the role of a professional and to work on 'real' tasks, or tasks with relevance in the context of the game world, partly reflects the cognitive apprenticeship approach. This approach is defined by Daniels as follows.

> *An instructional model informed by the social situation in which an apprentice might work with a master craftsperson in traditional societies. [...] The cognitive apprenticeship approach proposes that learners should engage in meaningful learning and problem solving whilst working on authentic tasks. (Daniels 2001, p.116)*

Obviously, playing an educational game cannot be considered equivalent to working with a 'master craftsperson'. The lack of such a figure could be partially compensated with the presence of a teacher and/or good automatic scaffolds, but the principle that game scholars tend to emphasize is working on 'authentic tasks'. In fact, this principle is also suggested in '*Video games and the future of learning*', where Shaffer (et al., 2005) talk about how computers games can be used as a constructive force in schools, homes, and at work. A learning activity is most effective when it is meaningful, experiential, social, and epistemological at the same time. Games let players think, talk, and act in a rich virtual world in a way that lets them inhabit roles which would otherwise not be accessible.

> *In virtual worlds, learners experience the concrete realities that words and symbols describe. Through such experiences, across multiple contexts, learners can understand complex concepts without losing the connection between abstract ideas and the real problems they can be used to solve. (Shaffer et al. 2005, p.4)*

The balancing between agency and learning content needs to be careful. In '*Learning Science Through Computer Games and Simulations'* it is stated that the optimal degree of user control is closely related to reaching intended learning goals. To some degree autonomy is engaging and motivating, but if it is open-ended, learners tend be confused and overwhelmed by it, an effect which is similar to Gee's 'deep learning paradox' which will be explained later. A certain amount of control and guidance through feedback during the play experience enhances learning of sciences processes and science content. Navigating according to personal preferences leads to higher cognitive outcome and positive attitudes (Honey & Hilton, 2011).

A strong feeling of control is often originated in a highly responsive environment, providing many choices and effective feedback for the user (Malone & Lepper 1987, Egenfeldt-Nielsen 2007). This does not necessarily imply actually providing the highest possible degree of choice, but giving players the feeling of choice by providing alternatives with tangible effects.

2.5.2 Immersion

We are going to talk about two factors that relate to immersion in computer games, both important in captivating the imagination of players: flow and identification.

There is an established interest in Csikszentmihalyi's theory of *flow*. According to Csikszentmihalyi's research, flow is a condition that comes in many forms.

> *Some people reported to Csikszentmihalyi that they reached flow through the rigors of perfecting an assembly line work task, or through the immersive problem-solving of law library research. Others say they achieved flow during the solitary exertion of rock climbing or through the exacting vocation of surgery. [...] [Flow] is a feeling of being in control of our actions, masters of our own fate. Although rare, when we achieve a state of flow we are deeply exhilarated. Csikszentmihalyi refers to this phenomenon as an optimal experience. (Salen & Zimmerman 2004, chapter 24)*

Flow is a state of deep engagement in the activity that brings a sense of achievement, accomplishment, even a "*greater sense of self*" (Salen & Zimmerman, 2004). In order to provide the most motivating mindset that would help players reach this state, a game needs to offer the player clear goals, fast feedback, a sense of control and concrete possibility of completing the task at hand. Concentration, an altered sense of time and disappearing self-awareness are also conditions for reaching the state of flow (Egenfeldt-Nielsen, 2007). A good balance of challenge and frustration, so that the game will be a *pleasantly frustrating* experience, is paramount for reaching this state of deep engagement (Gee, 2005a).

Gee also argues that identification with a virtual character is beneficial for immersion in games for learning (Gee 2005a). This does not mean that players need to directly and completely identify with their avatar, *becoming* the in-game character. Players are in fact "*fully aware of the character as an artificial construct*" (Salen & Zimmerman, 2004), but nonetheless they can and will use the character as a projection to access the gameworld and its narrative.

2.5.3 Narrative/Fantasy

In *'Learning Science Through Computer Games and Simulations*' narrative or fantasy of a computer game is considered extremely important, as it is the feature that determines how engaging and immersive an experience is. The narrative provides often the mission, question or problem that requires the learner to learn the content. On the other hand, there is a concern that 'too much' narrative will distract from learning content, because this makes it difficult to tell fantasy and learning content apart. This means, the designers have to carefully balance context and content (Honey & Hilton, 2011). If the learning content is properly integrated in the learning game, virtually all of the fantasy should be part of the learning content.

Wilson (et al. 2009, p.13) indicate fantasy to be an attribute that removes "*fear of real-life consequences but also has the added benefit of making users feel immersed in the game*", while Malone and Lepper believe that a fantasy environment, "*one that evokes mental images of physical or social situations not actually present*" (Malone & Lepper 1987, p.240) contributes greatly to the intrinsic motivation of the player. They differentiate between endogenous and exogenous fantasies, whereas an exogenous fantasy is one-sidedly dependent on the performance of the player, while the endogenous fantasy and player performance influence each other likewise. Malone and Lepper suppose that the former is more interesting and more educational than the latter.

2.5.4 Intrinsic Motivation

Bruner is an advocate for intrinsic motivation: love for learning must be "*based as much as possible upon the arousal of interest in what there is to be learned"* (Bruner 1960, p.80). Teachers are also responsible for keeping such interest "*broad and diverse in expression*"

(Bruner 1960, p.80). The importance of intrinsic motivation when facing an intellectual challenge is emphasized by many other scholars (Egenfeldt-Nielsen 2007, Gee 2005a, Malone 1980). Sedighian and Sedighian (1996) argue that intrinsic motivation is best achieved when the task has some relevance for the child. Abstract topics such as mathematics and procedural literacy have almost no impact on the child's everyday life: they are not perceived as something having any useful application in the short term. Therefore the difficulty to encourage intrinsic motivation: the task at hand has to be invested with some tangible importance for the child to *care* about learning it. One way to achieve such a result is by "*placing children in situations in which learning mathematics becomes a tangible need*". (Sedighian & Sedighian 1996, p.2).

> *When playing well-designed CBMGs [Computer Based Mathematical Games] (i.e., ones in which the mathematics is used as a continual and natural part of the game rather than as incidental diversions from the main activity), children gradually develop the need to learn the embedded mathematical content in order to satisfy their need to play the game. (Sedighian & Sedighian 1996, p.2)*

In '*What makes things fun to learn?*' (Malone 1980) and '*Making learning fun: A taxonomy of intrinsic motivations for learning*' Malone and Lepper (1987) provide a framework intended to enable designers of instructional computer games developing more engaging learning games using 'common-sense' principles and a general taxonomy of 'intrinsic motivation'. Interesting challenges can be maintained by providing an appropriate level of difficulty through meaningful short-term and long-term *goals*, gameplay with an uncertain outcome and encouraging performance feedback (Malone 1980).

A good game should inspire the players curiosity by presenting complex and unknown information in order to encourage exploration and organization, while maintaining a level of familiarity that intrigues the player to make assumptions on what outcome their actions will provoke (Egenfeldt-Nielsen 2007, Malone 1980).

Malone also underlines the difference between intrinsic and extrinsic motivation. Intrinsic motivation is induced by the task itself, while extrinsic motivation comes from expectation of an extrinsic reward, which can under certain circumstances contradict the former (Lepper et al. 1973). Educational games that are purely based on extrinsic motivation are assumed to be less effective as they lead the player to focus more on the reward than to engage in the learning activity itself.

> *Malone and Lepper pointed out that a long list of educational computer games has extrinsic game elements which can get in the way of the learning experience. These correspond to the heavily criticized edutainment titles, where there is no connection between the computer game and the learning part. The game is mainly used as a reward for doing some learning activity. (Egenfeldt-Nielsen 2007, p.60)*

Egenfeldt-Nielsen criticises existing edutainment titles for failing to create intrinsically motivating learning experiences because of a lacking connection between rewards and learning content and therefore creating weak learning experiences like memorization of practices instead of deep understanding of skills or content. On the other hand, the strong focus on making educational games mainly a pleasant and fun experience might be counterproductive to the goal of improving learning, as frustration is part of learning. A conflict arises between stealth learning and transferability of knowledge - giving away the learning purpose might deter children from playing the game with an open mind, while hiding it could results in inferior learning experience, as the player is not aware of what they learned (Egenfeldt-Nielsen 2007).

2.5.5 Visual Stimuli

Following Honey and Hilton (2011), many studies are concerned with how different people benefit from different visual stimuli (e.g. images compared to text; detailed/realistic compared to stylized/abstract). General findings indicate that "*idealized graphics enhance learning and transfer when compared with highly realistic graphics.*" (Honey & Hilton 2011, p.49). Furthermore, the form of representation is connected to learning goals: "*when designing simulations, it is important that the salient features of the simulation are ones that will be most productive in terms of the targeted learning goals*" (Honey & Hilton 2011, p.49). This means that features that are visually more prominent will catch the player's focus resulting in a higher chance that they will be learned.

A young audience is especially less prone to overlook the shortcomings of a game lacking in visual and auditory stimuli: "*any effort to introduce games designed for informal science learning will have to compete with the production and marketing of commercial games for young* *people's attention*" (Honey & Hilton 2011, p.76). The principles of association through pleasure and attraction argued by Sedighian and Sedighian (1996) justify the amount of work necessary for the process of polishing visual and audio in an educational game. Children need sensorial stimuli to associate the learning of a concept with a pleasant memory and to keep engagement high.

> *At the initial stage of our research several of the computer games which we installed for the children had minimal sensory stimuli. Many children were not particularly approving of these games because they had no fancy graphics, their images were in black and white, their animations were very simple, their sound effects were primitive, and they had no background music. We have found that for children such sensory stimuli add to the fun of playing the game and make the learning of mathematics more enjoyable and memorable. (Sedighian & Sedighian 1996, p.6)*

2.5.6 Clear Goals

In '*Learning and Games*', Gee (2008) describes how experiences can be used to facilitate learning. Experiences need to be structured in specific goals, so the learner can measure and compare if and how well those goals were achieved. The experiences need to be interpreted by relating the goals to the thought process that led to the actions taken to achieve them. Immediate feedback helps the player to adjust their thought process and expectations, and should provide encouraging alternatives to point the player in the right direction (Gee 2008).

The same conclusion is reached by Honey and Hilton (2011). As already mentioned both when introducing the conditions for providing good scaffolding and for keeping players in *flow*, game goals need to be clearly defined in the game design and clearly presented to the player. Optimally, goals are defined before other design features and in accordance with the learning objectives, while avoiding information that distracts or possibly misleads the player. Test results show less progress when a game draws focus on other content than the intended learning goal (Plass et al. 2009).

Another crucial feature in design that helps directing the learner towards the learning goals is feedback. Graphical feedback with short explanations leads to better results than textual only.

Explanatory feedback better contributes to the learning experience than only corrective feedback, as it proves to evoke increased retention and transfer (Honey & Hilton 2011).

2.5.7 Community

One strong characteristic of gamers is that they commonly participate in big social gaming communities, often from all over the world, which can be an advantage for the learning process. As a matter of fact students can and should assume a more proactive role, searching for information over the internet and getting in touch with classmates. Encouraging students to interact and communicate with a broad social group sharing interests and questions, as an example of the principle 'effective social practices' enables learners to help each other. In player communities players share strategies, exchange files, collaborate, compete, advise each other, etc. and thereby develop *shared values* (Shaffer et al. 2005).

In Honey and Hilton (2011) emerges the argument that in a classroom is often only the teacher guiding and mentoring to the learners, while when using educational games, mentorship can be distributed among other adults, family members, and peers. A study using the game Digital Zoo found that "*players who reported that the adult mentors (design advisors and clients) helped them to think about their design or themselves and their job differently were significantly more likely to demonstrate an increased understanding of the engineering frame*" (Honey & Hilton 2011, p.73). Another advantage of distributed mentorship is the opportunity to differentiate roles and expertise of different players: "*Informal science learning contexts can support the co-construction of learning goals between learners and designers*" (Honey & Hilton 2011, p.74). These games can cater for learners with different levels of prior knowledge and offer different learning goals and roles in collective participation. Players evolve on their learning trajectory from beginner to expert and can take on unique identities.

Malone and Lepper (1987) state that interpersonal motivations such as competition against, collaboration with, as well as recognition from peers greatly enhance intrinsic motivations. Enabling those forms of interactions with other children involves either a classroom setting, a multiplayer mode within the game, or an online platform which connects the players over forums and social networks. The other side of the coin is the possible discouragement coming from competition and comparison: not being able to solve the problem as fast as their peers, or not being able to solve it at all might have a negative effect on motivation and self-esteem of the learners.

Bruner already warns about the “*danger signs of meritocracy and a new form of competitiveness*” (Bruner 1960, p.80). Echoing Bruner’s warning about the extensive use of extrinsic motivation in the ecosystem of the classroom, Sedighian and Sedighian also argue that *“children want and need to be successful in the social environment of their class.* (...) *Therefore, we should place them in environments of learning mathematics that provide this sense of success.”* (Sedighian & Sedighian 1996, p.4), explicitly indicating computer games as this environment. Malone and Lepper (1987) recognize the tradeoff between these two aspects, admitting that competition can be encouraging for self-confident or more skilled players, but also discouraging for players who are less confident or skilled.

2.6 Design Challenges

In this section we discuss problematic conflicts that commonly arise when developing and testing educational games and need to be considered during design.

2.6.1 Paradox of deep learning

Gee describes domains of knowledge, within which players can integrate facts and information into an immersive activity or experience, as a balancing act which leads to the ‘*central paradox of all deep learning*’ (Gee 2005b). Since any knowledge domain will be unlocked only through observation, actions, and actualisation, providing too much information will undermine the learning process, while providing too little will leave them unguided and force them to ‘re-invent the wheel’. Gee claims that games can resolve this paradox by providing an appropriate immersive setting and supporting the player with knowledge distributed in other characters, tools, objects, and the game world itself, as well as by providing information just-in-time and situated in the proper context (Gee 2005b). This way the player will be guided and steered towards the learning goal without losing control over their avatars development and being deprived of the learning experience itself.

Simon Egenfeldt-Nielsen (2007) explains a similar problem, namely ‘*learning vs. playing*’ which concerns balancing user control and guidance. The more the player is guided, supported, scaffolded and debriefed, the better they learn, but the less control they experience. The author then raises another concern about learning games in general: the more we push the game to be fun, the more we send out the message that learning always has to be fun and is not worth sweating for. (Egenfeldt-Nielsen, 2007) This interesting statement seems to imply that whenever one is having fun there can be no difficulty involved, opposing “fun” and “sweat”. While this might be true for many activities, it is an unsafe assumption if applied to computer games. Csikszentmihalyi's theory of Flow explains that games have to be difficult and challenging but at the same time interesting enough, so players will be intrinsically motivated. Lazzaro (2004) also explains that certain categories of players finds enjoyment in hard challenges. Many players go through a lot of effort to learn and actually do “sweat” to become better at playing games, when they are following a higher goal.

2.6.2 Informal Learning Context

Honey and Hilton emphasize the benefits of using educational computer games in an ‘informal learning context’ outside of school setting (after school clubs, individually at home, etc.). They discuss which opportunities are opened up in informal learning contexts as well as constraints, and how to overcome those constraints, “*so that simulations and games can serve as a bridge, linking science learning across and between informal and formal contexts*” (Honey & Hilton 2011, p.69). The great diversity in learning goals over different projects provides opportunities as well as challenges.

One advantage is the freedom to pursue *diverse learning goals*, as demonstrated by games that link for example biology and ecology. Their nature makes it difficult to place the game in one school subject (Honey & Hilton, 2011). Egenfeldt-Nielsen (2007) calls this the ‘*one subject vs. cross-subjects*’ problem. Linking across diverse topics is easier done in an informal learning setting. On the other hand, assessment is more difficult, as students who voluntarily engage in gaming activities in their spare time are less inclined to answer pretests and posttests than when assessed in school settings where tests are regular activities, as Honey and Hilton point out listing a number of studies from DeVane, Durga and Squire, Hayes and King, Steinkuehler and King (as in Honey & Hilton 2011, p.78). Because of this, developers put more focus on keeping the player’s interest and motivation than on the educational content, but this also gives them the opportunity to focus on very different goals than formal educators. Individualized learning is encouraged in informal learning settings: players often develop deep interest and expertise in specific areas, participate in or even create communities (a common practice in gaming culture), where members are valued on their expertise, not on grades or academic credentials (Honey & Hilton 2011).

Egenfeldt-Nielsen mentions the conflict of '*depth vs. superficiality*', which compares the information load contained in a book or a video to a computer game. Games cannot present the same material without game play and rules suffering from it, whereas a richer learning experience may be facilitated through the social context. Games can deliver what the student is able to fully understand and support information in multiple ways, which is more valuable than providing a large amount of information that the learner cannot fully engage in (Egenfeldt-Nielsen, 2007).

Another tension area explained by Egenfeldt-Nielsen is '*Teacher intervention vs. no teacher intervention*'. Teachers are an important part of the learning process as they direct the learner and can debrief them to ensure a more effective learning experience. In edutainment the role of the teacher is mostly neglected, although the socio-cultural approach stresses the importance of teachers (Egenfeldt-Nielsen, 2007). As Vygotsky claims, all the social parts of learning, competition, explanations from peers, etc. are important. In order to properly process learned content, children need to be debriefed and engaged in conversation about the topics encountered in the game (Papert, 1998).

Honey and Hilton emphasize the importance of so-called *middle spaces*: recreational spaces or after school programs, that can help integrating informal learning games because they are less rigid than schools, but provide more guidance and structure than home settings. Children can be empowered as learner when e.g. modding a game or simulation. Those environments can provide more control which leads to more engagement and better learning. Further, they can provide better technical infrastructure than available in many home settings. Content learned at school can be individually explored and delved into while more advanced students can mentor others, collaborate and share their experiences (Honey & Hilton, 2011).

2.6.3 Constraints

There are of course also a number of constraints for the use of games in informal learning settings, e.g. social, cultural, and technical constraints. Informal learning environments are for most children found with their family and at home, rarer at gaming clubs and communities, and in this context they often play games that are popular among their peers. “*The presence of educational games or other types of learning software in their homes does not enhance the social standing of children and youth in their peer networks. [...] Generally, both parents and children view gaming as an activity in opposition to academic learning*” (Honey & Hilton 2011, p.76) The industry itself is a constraint, as informal educational games will have to compete with purely recreational commercial games - there is no robust market for public interest games. Games like chess or scrabble are regarded as valuable *enrichment activities*, and since the 80’s also some edutainment titles, e.g. *Civilization*. This is a game which is entertainment-oriented but has been approved by parents and educators, and then crossed over to be used in schools. Some children welcome their parents guidance, but some, especially when they become teenagers, resist their parents attempt to dictate media choices. This is why the market usually targets a younger audience, and games that have a strict learning agenda are less accepted among adolescents (Honey & Hilton, 2011).

Development constraints are based on the fact that most science games are developed and tested in classroom settings, and not played differently in informal settings. To enable this there should be more open-ended play options that invite exploration, and more collaborative problems that link the game universe with real science world. One common problem is the matter of successfully integrating interest and learning. A number of studies from the 90’s found that “*players rarely oriented to the scientific content of the game without the explicit intervention of an educationally minded adult.*” (Honey & Hilton 2011, p.80). When playing a game outside of a learning context, players tend to focus more on beating the game or engaging with incidental content than learning content.

Alternative approaches involve addressing constraints by using mobile technology (e.g. internet, mobile phones) to overcome social, cultural and technical constraints, familiarizing teachers with technology, re-integrating those games in the classroom, providing secure online platforms for children to interact while being monitored by teachers and parents, so they can use them in their spare time and in informal settings while guidance and external control is provided. Additionally they provide voluntary activities for children who want to pursue personal interests. Students learn because they know that their knowledge and skills is directly related to their game identity and has an impact on the game world (Honey & Hilton, 2011).

2.7 Summary

In preparation for designing an educational game, we covered prior research and the influence of relevant learning theories on existing educational titles, the term *procedural literacy* and its relevance to this thesis, as well as a selection of design principles that have been proven to affect the learning experience in educational games.

Mateas' call for introducing procedural literacy at an early age, is compatible with Bruner's principle of *readiness for learning*. While keeping in mind the constructivist and experiential learning theories, as well as the scaffold theory to support learners in their process, we marked some relevant and synergetic design principles that emerge from the literature.

Agency is central to the principle of authentic professionalism and to *flow*. *Immersion* is also beneficial for flow and identification. *Narrative and fantasy* can be used as a reinforcing frame for agency and for *intrinsic motivation,* which in turn is emphasized as being greatly beneficial for the learning process. *Visual stimuli* are necessary to keep the attention of a young audience. *Clear goals* are a basic condition for flow and for providing adequate scaffolds.

Some challenges for the design process are underlined. These are the paradox of deep learning and the pros and cons of informal learning settings, including the absence of a teacher that could debrief the learning content after the play activity. The latter is especially related to our production, as our game is designed for an informal learning setting.

3 Design

In this chapter we will delve into the design process of Machineers, the vertical slice of the learning game that was developed for this thesis. At first we will expose the early design decisions: game genre, setting, and general design principles such as immersion, agency and positive feedback. Then we will outline the evolution of our design approach, followed by the description of the game elements and their mapping to procedural literacy concepts, the game puzzles and the difficulty curve, a description of the scaffolds provided to the player, and a brief excursion into the UI design.

3.1 Game Genre

> *Students are to be led to see that the great ideas in a discipline are solutions to intellectual puzzles and that solving them is an extremely satisfying experience. Scientists routinely describe the intellectual excitement that accompanies insights into and understandings of how things work and there is no need for children to be deprived of those experiences. (Olson 2007, p.39)*

Olson’s metaphor of intellectual challenges as puzzles alone could give a good idea of the reasons why we chose this production to be a puzzle game, but there is more to it. Being able to present the player with a well ordered series of increasingly complex problems (Gee 2005a) is a fundamental condition not only for them to develop skills, but also for us to provide adequate scaffolds. The ability of ‘reading processes’, as we explained above, is a central goal in learning procedural literacy. In this sense, we created a series of processes with a limited amount of constituent elements and clear goals for the player to explore and complete.

Framing those challenges within a virtual world allows us to give importance to the activity of facing and solving them. Therefore, we settled for a puzzle-adventure game. Presenting the player with this game type also enables us to easily incorporate the stealth learning approach.

We aimed for creating a modular puzzle structure, for implementing a sandbox mode for children to play in after completing the game, to allow them exploring the relationships between components as the constructivist approach would suggest. Unfortunately, such a feature turned out to be unfeasible within the scope of this project.

3.2 Setting and Narrative

Learning a new domain, whether it be physics or furniture making, requires the learner to take on a new identity: to make commitment to see and value work and the world in the ways in which good physicists and good furniture makers do. Good video games capture players through identity. (Gee 2005a, p.34)

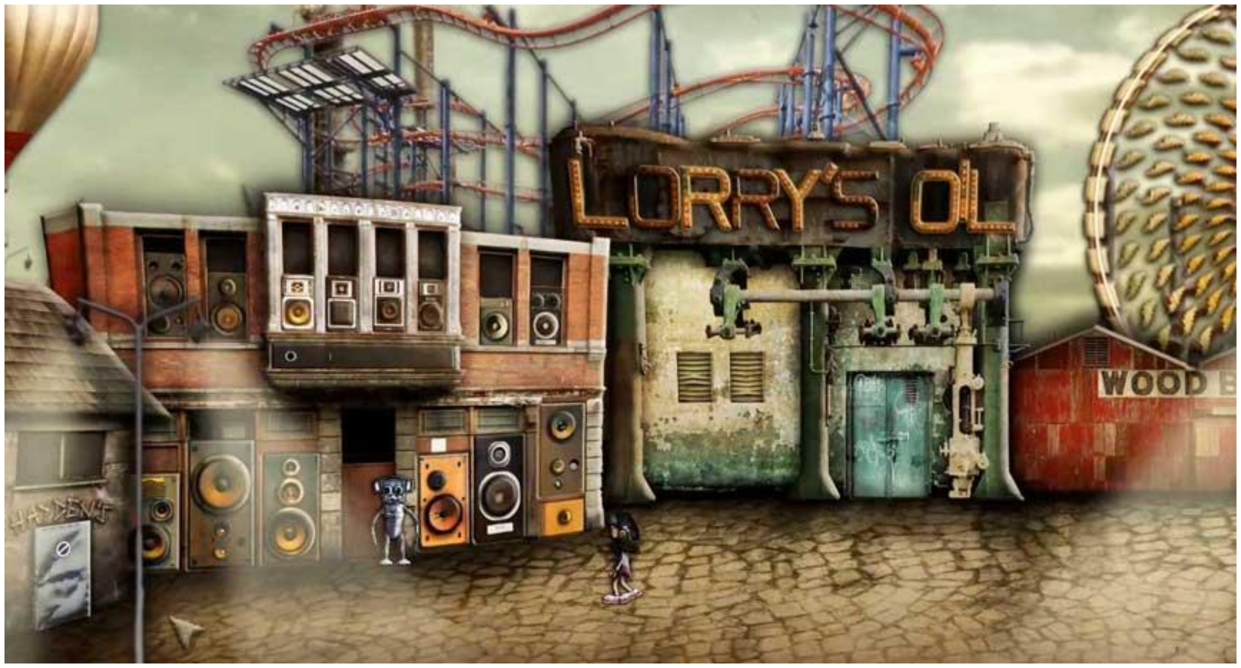


Image 3.1 - in-game screenshot - town center

Machineers' backdrop is quirky and industrial. We decided on a world ruled by machines with a life of their own, using them as a metaphor for programs and software. The city in which the game takes place is on the outskirt of an amusement park, visible in the background, and its inhabitants are all robots. By placing robot characters in an amusement-park setting we aim to create a familiar game world which possibly stimulates experimentation and exploration. Every house and object is artificial, metallic and a bit worn out: the feeling that this world is a little old and rusty, subtly emphasizes the importance of the player's role, which is going to be a repair technician. Engineers and repair technician are the 'programmers' of this world. Initially, we had a metaphor for the learning process as well, a journey from town to town, each one as a different game level; the production of this vertical slice was boiled down to a single location for scope issues, but ideally the concept of Machineers still aims for a journey between different levels.

The player assumes a protagonist's role: the young girl Zola, a repair shop apprentice. Zola is a flat character, in the sense that she has no strong distinctive traits. Some personality will transpire from the available answer options in the dialogue system, from polite, light-hearted, to sometimes sarcastic; but we avoid imposing a personality frame on players, allowing them to play freely - as Frasca argues, "*the more freedom the player is given, the less personality the character will have*" (Frasca 2001, p.2). The intent is to letting players identify with the role of the apprentice, reducing the main character to "*a "cursor" for the player's actions*" (Frasca 2001, p.2). Still, Zola's appearance is happy and spirited, as an invitation for players to enjoy the process and identify with the praise she will receive.

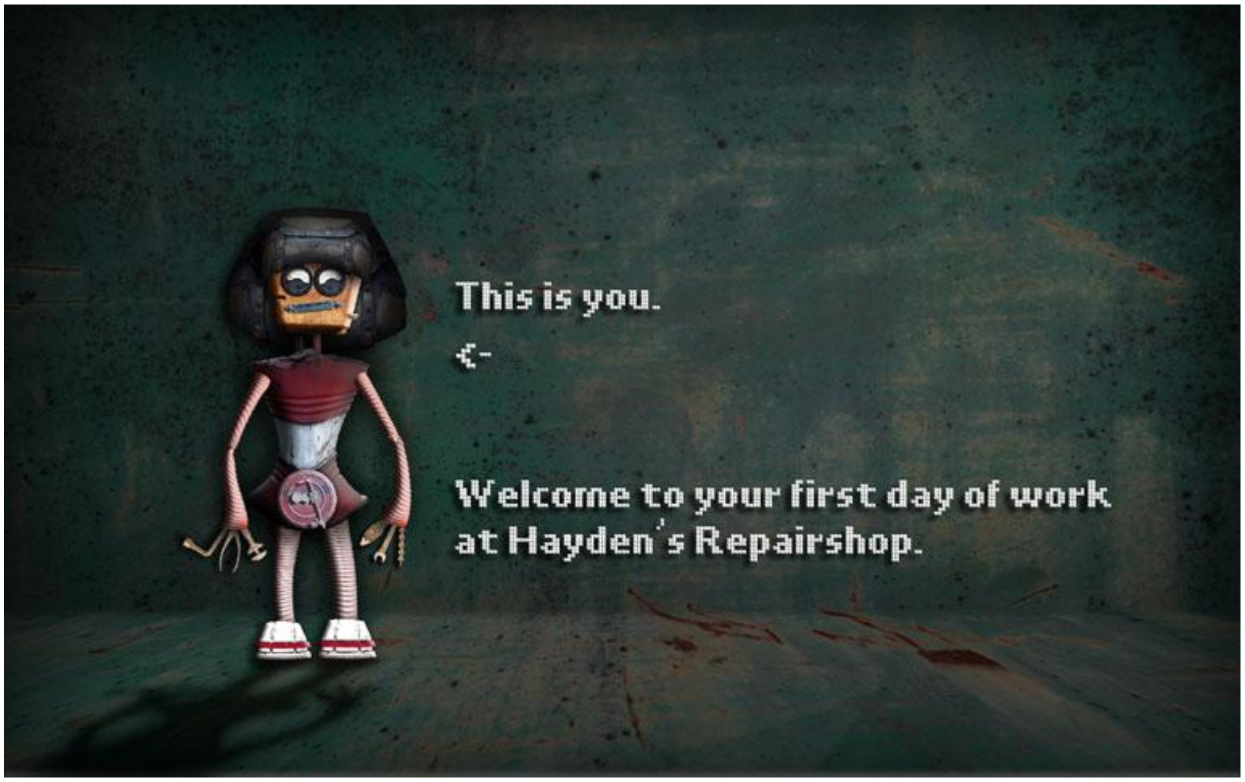


Image 3.2 - Introductory screen to the game.

The game narrative is introduced in a very simple way: an initial screen (image 3.2) shows the avatar, giving players basic information about 'who' and 'where' their digital identity is going to be. This small piece of information helps the player not to feel alienated when the first game scene appears (image 3.3). No further information is automatically provided. Control is left entirely in the players' hands, who is in charge of finding out what this place is and what they are supposed to do.

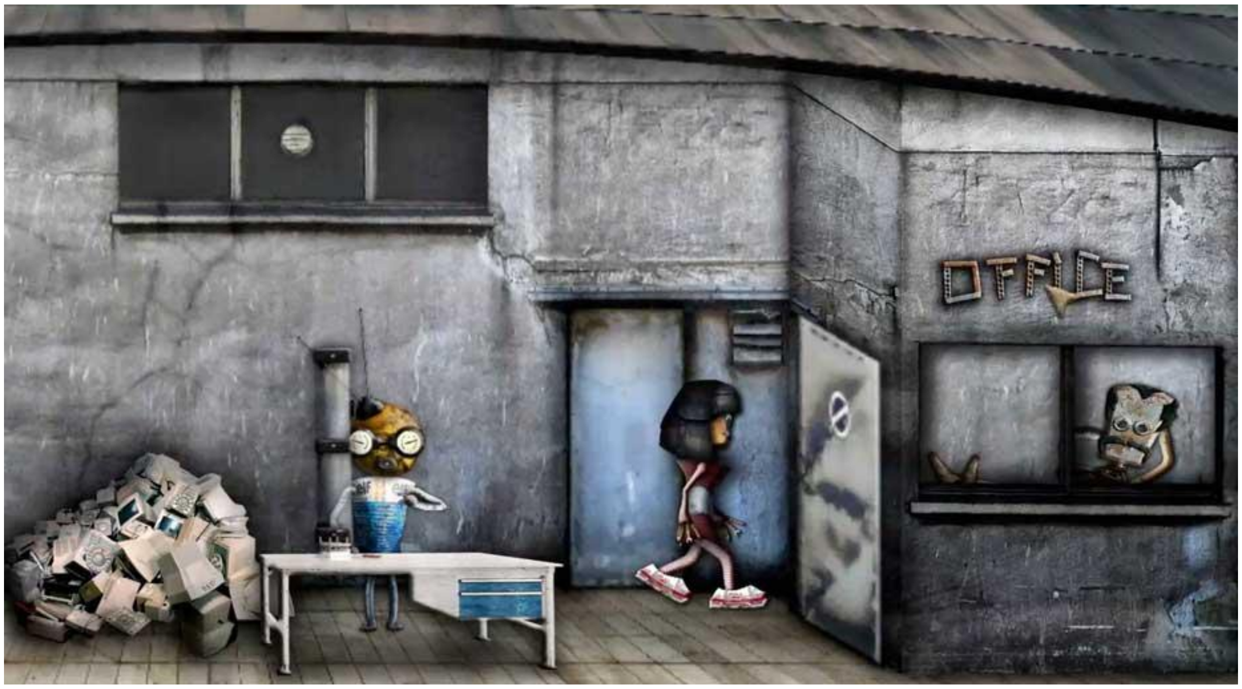
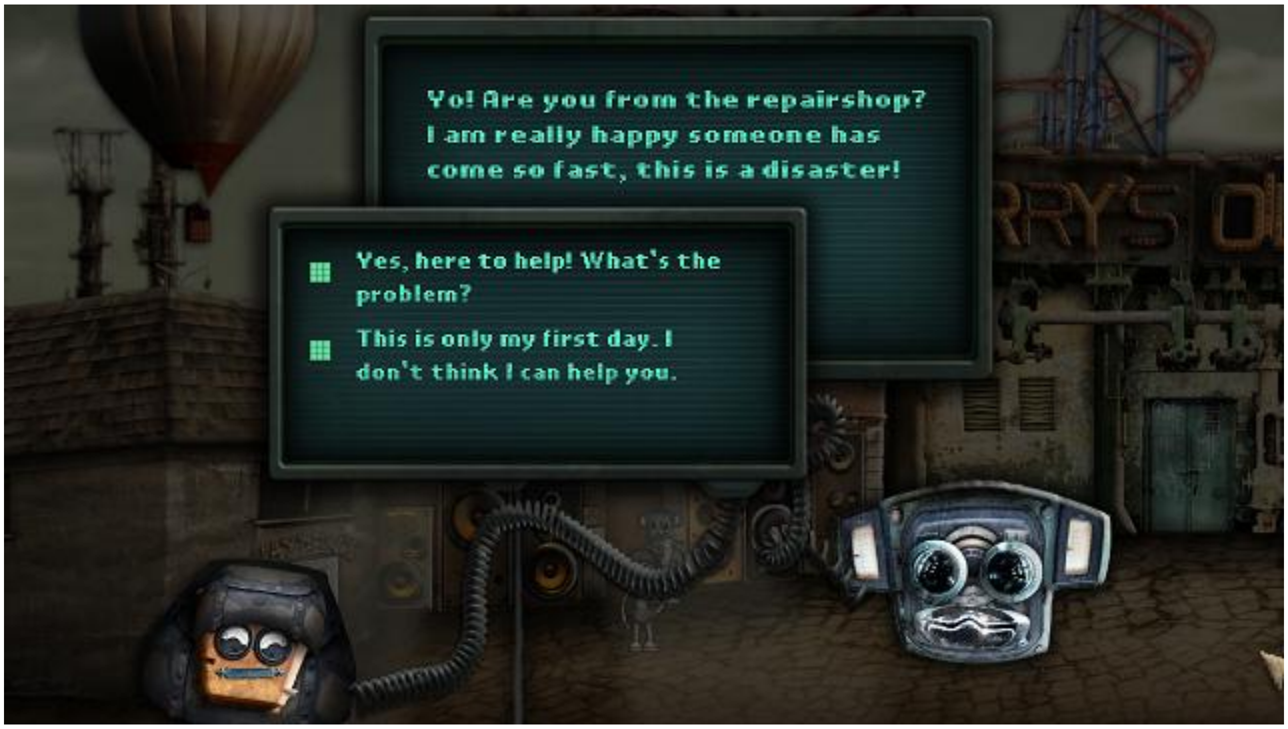


Image 3.3 - The first scene: the repair shop

Polishing everything, from visuals to outsourcing to a professional sound designer for the audio, has been an important part of the process. As we emphasized in section 2.5.5 Visual Stimuli, a young audience is less prone to overlook the shortcomings of a game lacking in visual and auditory stimuli (Honey & Hilton 2011, Sedighian & Sedighian 1996): to make sure our game would captivate their attention and let our educational content pass through, especially given the informal context in which Machineers is meant to be played, we needed to make an effort in this direction.

3.3 Play agency

The game is divided into two different modes: adventure and puzzle. In the adventure mode, the player moves the avatar in the virtual world by clicking on points of interest (doors, objects or other robots) and on the ground. The avatar will walk up to the selected location by moving on a predefined path. When exploring, the player can interact with other robots, gets tasks assigned, discovers informations about the game world and the narrative through a dialogue system. When clicking on other robots, Zola will start a conversation with them; an animation of the robot face pops up, providing sound and visual stimuli.


Yo! Are you from the repairshop? I am really happy someone has come so fast, this is a disaster!

Yes, here to help! What's the problem?

This is only my first day. I don't think I can help you.

Image 3.4 - Dialogue with answer options

Oftentimes the player can choose between a different set of possible answers to give, triggering various responses. Several options in the dialogues, including timid, absurd, and comical replies, give the player the chance for identification with the character-apprentice, and some choice. The consequences of such choices are minimal, limited to one or two different lines in the next part of the dialogue. Even so, giving players a choice about what to say prevents the dialogues to be a passive process, something that unfolds while the player is relegated to the role of mere witness. Having alternatives empowers agency and a feeling of control (Gee 2005a, Honey & Hilton 2011).

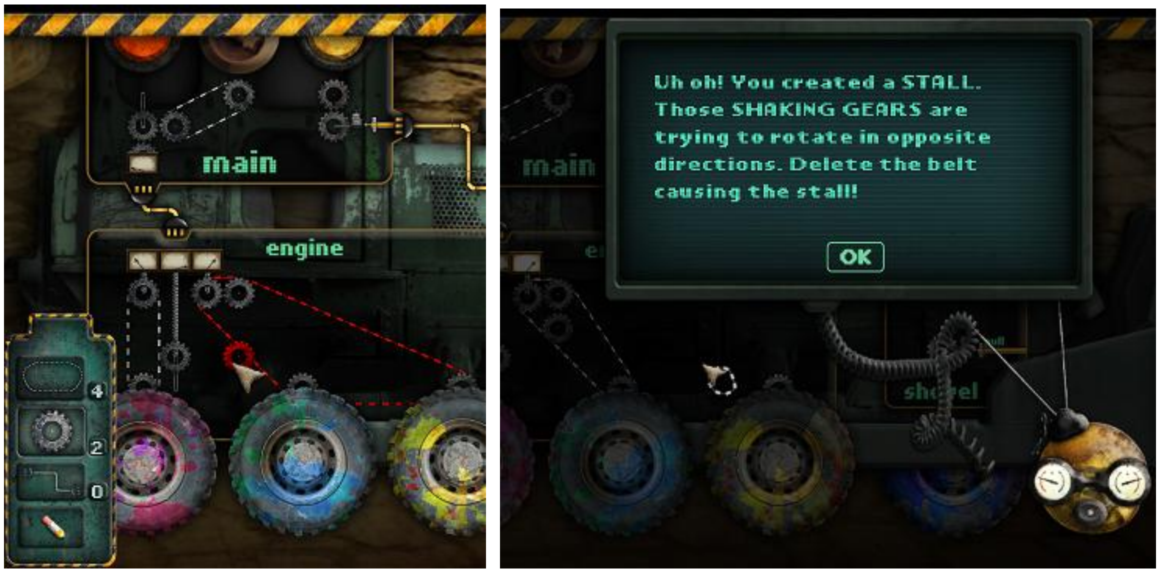
When a puzzle starts, the set of possible interactions changes completely. This is the puzzle mode, where the player actually faces a challenge. We will discuss interactions and puzzles in section 3.7.

3.4 Positive feedback

Image 3.3 shows Hayden in his office; he is the owner of the repair shop and Zolas employer. He is initially very cold: he does not believe Zola has what it takes to become a good Machineer and dismisses her (“*Geary or Bent might have some work for you*”). This approach makes it easier to give more positive feedback later on: as puzzles are solved, Hayden will become more and more aware of Zola’s skills and eventually will start treating her better. This is a principle we apply to all characters: the customers will thank the player and recognize her skill upon puzzle completion. The more advanced the puzzle, the more the dialogues underline how complicated the task was. Dialogues aim to give positive reinforcement to the activity of solving other robots problems, which in turn transmits a sense of mastery and advancement.

Robots will ask for help in many different ways: the DJ will literally beg the player to fix his machine, Lorry will show perplexity because of Zola’s young age and gender, Ivan will treat her as a professional, while Ruby will simply assume that she will fix her problem right away. Still, the importance of her role will always be emphasized: the characters do care about their problem and need someone to solve them. Repair works are not mere drill and practice but are invested with importance in the game narrative, in the attempt of giving the feeling of working on real tasks and fostering a sense of success every time a challenge is beaten, following Gee’s (2005b) principle of authentic professionalism explained in section 2.5.1 Agency. This should provide the proper intrinsic motivation for the learner to understand relations between the machine parts.

Mistakes are not underlined, unless strictly necessary. Negative feedback is provided in two scenarios only, in both cases to prevent a usability issue. The first case is when players try an illegal element placement, like overlapping gears or belts. The UI will already signal that the action is not possible before it is attempted. For example, when players try to place a gear in the scene, a preview will follow the mouse around, showing them where the gear would be placed if they clicked in that moment. For an illegal placement, the preview would turn red (image 3.5). A low-tone buzz sound accompanies the click, indicating that the desired action is not possible. The second case is a stalling gears situation, which happens when two adjacent gears are connected with a belt. The belt transmits movement to rotate the gears in the same direction, while the cogs transmit movement to rotate in opposite directions. When set in motion, the gears will start shaking and a dialogue will pop up, warning the player. This is the only time we give explicit feedback about a wrong action, but also providing the player with a solution to it (image 3.6).



Uh oh! You created a STALL. Those SHAKING GEARS are trying to rotate in opposite directions. Delete the belt causing the stall!

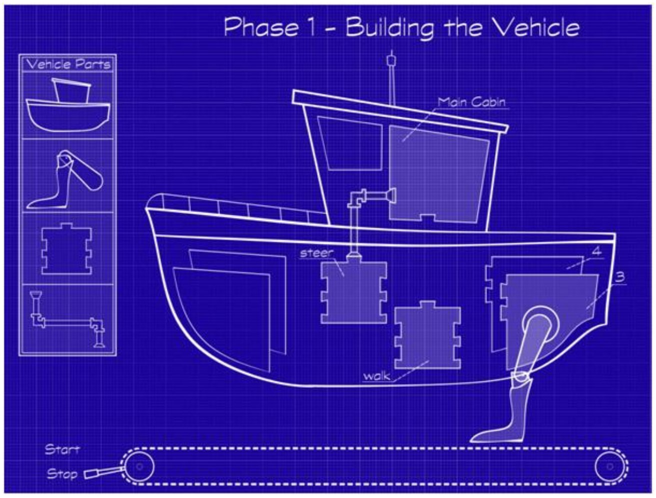
Image 3.5 and 3.6 - Negative feedback scenarios

Following Sedighian and Sedighian, “an important factor in feeling successful is how children perceive their mistakes. We found that since children could recover from their mistakes in the game without forfeiting much they would not feel threatened by making mistakes” (Sedighian & Sedighian 1996, p.4). Trial and error is a perfectly acceptable strategy for solving puzzles and it is even encouraged in some of them, as it would be for a student engaging in a programming exercise. The mindset we try to set the player into is to embrace failure as means to deeper understanding. Puzzles are not meant to be easy, but nonetheless we intend to cater for interesting challenges within our game by offering different help options which can be accessed on-demand: following the principle of Individual Learners Differences, it is important “to include adaptive features that modify the pace and type of informations, based on user response” (Honey & Hilton 2011, p.52). Players in Machineers are able to set their own goals and level of difficulty by choosing how fast and with how many hints to solve the puzzle, while not facing the danger of being completely discouraged.

3.5 First design iteration

The first step in designing our puzzle game, after the robotic setting had been decided upon, was to find out which building blocks were the constituent components of our puzzles. Tightly coupling puzzle elements with programming logics was initially a central focus in the design activity: following Honey and Hilton, we tried to find meaningful metaphors to associate each puzzle element with a programming concept, as "*all the elements of a simulation (game) should be directly related to the learning goals, avoiding extraneous informations that could distract the learner, disrupt the learning process, or seduce them in incorrect understanding"* (Honey & Hilton 2011, p.46). Ideally, this would mean that every interaction with the game is directed towards a learning goal. If every puzzle element represents a programming concept, then, solving puzzles would mean solving actual programming problems, a certain degree of abstraction given, depending on how close the metaphors and the concepts are.

To enhance gameplay we increased the abstraction level of our metaphors over several design iterations. In fact, we found that strictly adhering to the principle mentioned above was leading us to design a tool for drawing flowcharts, or a visual programming language - which should not be surprising, given the premises. Initially, all machine parts would be placed and connected, then their internal logics, featuring loops and conditional structures, would be specified. Different machine parts would contain little robots operating on and exchanging variables in different forms (oil cans, numeric values, lengths, angles, etc.) that would in turn empower other machine parts. Figure 3.7 shows an early interface mockup for building a walking vehicle. Such a machine provides space for 'instantiating' four legs, and a 'walk' method for controlling them. In this method, a small robot would calculate the number of steps necessary to cover a certain 'input' distance. Then a loop would activate the four legs 'counter' times.



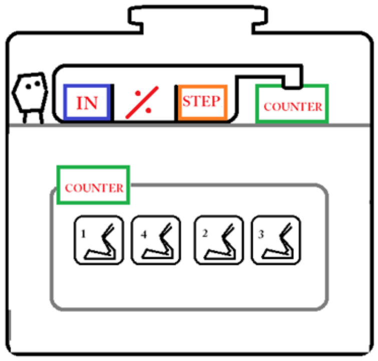


Image 3.7 and 3.8 - Early interface mockup and drawing for a method design

While the coupling of game mechanics and educational content was strong, this design had two major flaws. First and foremost, the prior knowledge required in order to approach the game included elements of mathematics that we were not certain our intended target audience would already master. The second reason is, mathematical concepts were not part of our learning goals, in contrast with the very principle we tried to follow.

If we take another look at image 3.8, given an overall distance to be covered and the length of a single step, the goal is to calculate a counter that would specify how many times the legs need to be activated to cover that distance. It is but a division, and that could be very simple if the length of a single step was unitary; nonetheless, this design carries the legacy of mathematics. Bruner (1960), as mentioned above, underlines that the material has to be in a form the learner can understand, based on his previous experience. Players need to understand that formula: what the purpose of that operation is and how the resulting value is calculated. Providing scaffolds for them to make sure they would understand that structure would deviate from our teaching purposes.

The concept of variables, upon which we initially put much effort, implies storing data for later manipulation and use, which in turn implies that the player has to be exposed to some data to be manipulated. We decided to remove this layer of complexity by sacrificing variables, in favor of a simpler puzzle structure: one of the focus points of teaching procedural literacy is the ability "*to read and write processes*" (Mateas 2005, p.1), so we gave priority to exposing logical processes rather than data manipulation.

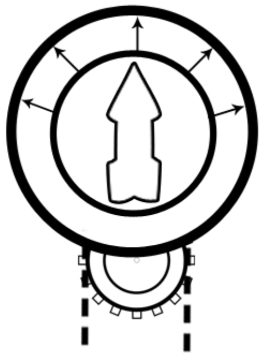
3.6 Puzzle elements and concept mapping

As we translated programming concepts using the machine metaphor, we needed to create a completely new, simple and visual terminology, which would constitute the building blocks of our puzzles. The intention is to map puzzle elements to "*the more general tropes and structures that cut across all languages*" (Mateas 2005, p.1).

3.6.1 User Input - Value

Given the premise that we did not want the user to manipulate data, we still wanted machines to have different statuses according to some form of user input.

On any kind of device, a knob usually allows the user to select a single option out of many, instructing an electronic or analog device to perform different operations (like the direction of the air flow on a car's air conditioner), or influencing an operation by controlling a parameter (like the heat level on a stove, a radiator, or the volume knob on a computer's speakers). Likewise in Machineers, where knobs are used to set different values in an intuitive form. The value is represented by the inclination of the arrow.

 *Image 3.9 - Puzzle element - Knob*	`// Pseudocode representation` `Knob.Value = Input;`

This actually means that even if we did not implement the metaphor of variable as such, we still have some information handled inside the machine's logics. This data is purely instrumental in attaining an immediate result: it is not stored anywhere, but it exerts control on the machine - much like an input value, which has little meaning if it is not used to exert some control on the process. Therefore, a knob has to be connected to another component, a case switch in order to function. The case switch is explained below.

3.6.2 User Input - Do While

Another important concept in the control flow of a computer program is a *do-while* loop. This statement keeps executing a series of instructions as long as a logical condition is met. This condition usually comes in the form of a boolean variable, or a comparison between values. Given that in Machineers there is no storing of data, that condition must come straight from a user's action. An *on/off button* is the metaphor we chose to represents this concept. As long as the button is activated, we consider the logical condition to be true, and keep powering a part of the machine.

One can argue that this is still a choice between two different values (on/off), and wonder why we did not use the same metaphor as above, with a two values knob. While that is certainly true, there are two good reasons for using a different metaphor. First and foremost, buttons are more common than knobs for activating modern electronic devices. That is not a secondary aspect, given the young age of our target audience. Another reason is that an on/off button allows for two different behaviours: a machine could be *continuously* activated or triggered for a *one-shot* action. The two different buttons are distinguished by color: a do-while button is bright red, while a one-shot activation button is yellow.

While the button is pressed, the gear beneath keeps rotating. A one-shot activation removes the *while* condition, reducing the logical structure to a single activation instruction.

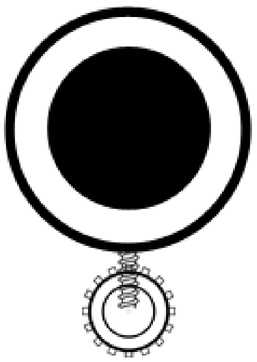


Image 3.10 - Puzzle element - Button

```
version (A) - start/stop button
while (Button.IsActive)
{
     Gear.Activate()
}


version (B) - perform once button
Gear.Activate()
Button.IsActive = false;
```

3.6.3 RobOmeter: Case Switch

The knob, as explained above, allows the user to specify different values without being forced to handle any data. This value can be checked through a robOmeter, the in-game representation of a *switch* structure.

As the value set by a knob is represented by the inclination of the knob arrow, the switch shows a number of meters with different arrow inclinations. When connected properly, a knob can control which gear placed underneath a robOmeter will be activated.

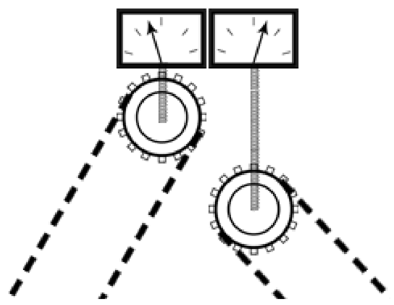


Image 3.11 - puzzle element - robOmeter

```
switch (RobOmeter.Value){
     case left:
     LeftGear.Activate()
          break
     case right:
    RightGear.Activate()
          break
}
```

There is an apparent inconsistency between the behaviour of the knob and the button that we need to clarify. In fact, the knob will *keep transmitting* its parameter to any robOmeter connected to it, causing the matching robOmeter gears to *continuously* rotate. In programming, a *switch* instruction normally checks for a value, executes the corresponding action, then moves on with the code. Our puzzle element behaves as if the switch was included in a continuous while loop.

```
while (true){
   switch (RobOmeter.Value){
     case left:
       LeftGear.Activate()
       break
     case right:
       RightGear.Activate()
       break
   }
}
```

We justify the omission of the loop in terms of better usability and reduced complexity. We could have placed an activation button at the root of every machine to simulate the loop, but in all the puzzles we implemented this would have been an unnecessary formalism, as the controls are always situated at the root. The knob is the representation of the user giving an input value to the machine: by automatically changing the machine status in response of this operation, implicitly assuming the loop, we can immediately show to the user the consequences of their action.

There is a particular type of gear that can be connected only to a robOmeter: the gear on rail. This gear is placed on a structure allowing it to slide: when activated, the gear will slide to the other side of the rail and start rotating there. This is not intended as the representation of a specific programming concept; rather, it is a structure born out of necessity from the constituent elements of the puzzles, necessary to allow machines to perform some operations. For example, if we look at the pseudocode related to picture 3.12. It is a switch activating Gear1 for one value, Gear2 for another, and both gears for the middle switch value. To represent this logic in one of our machines we needed a component that would activate Gear1 and Gear2 for the middle value. Simply connecting them together would not have sufficed: the basic concept upon which we base the entire game is that connected parts move together. Therefore, Gear1 and Gear2 had to be separated. We needed a removable link, something that would slide into place, connect and activate the other moving parts only when necessary. The sliding gear was created to solve this specific problem.

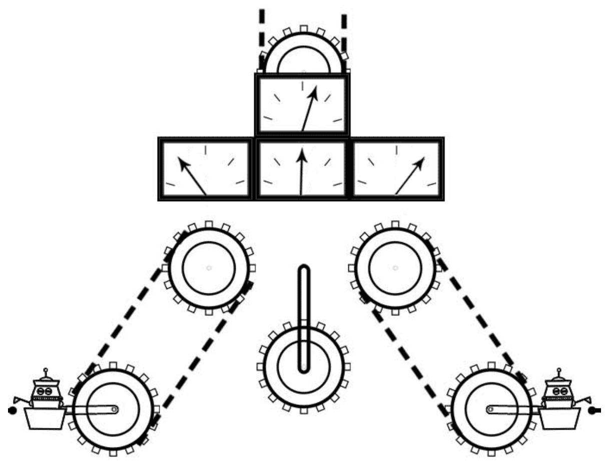


Image 3.12 - Gear on Rail

```
switch (RobOmeter.Value){
     case one:
          Gear1.Activate()
          break
     case two:
          Gear2.Activate()
          break
     case three:
           Gear1.Activate()
           Gear2.Activate()
           break
}
```

3.6.4 Method Call - With Parameter: gearOmeter

Given their nature of value-setter and value-checker, a knob and a robOmeter are meaningful only when connected. This connection might be direct, or across two different methods. Different methods are represented by different machine parts. Machine parts are linked by a wire that represents a method call.

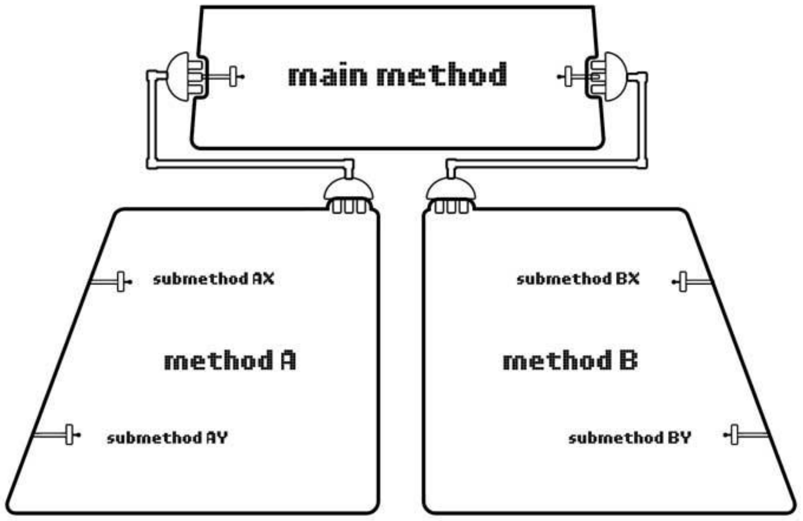


Image 3.13 - Method hierarchy

Connecting a knob and a robOmeter across two methods simulates a method call with parameter. A value is set in one method, then passed on to another method which will perform different actions accordingly.

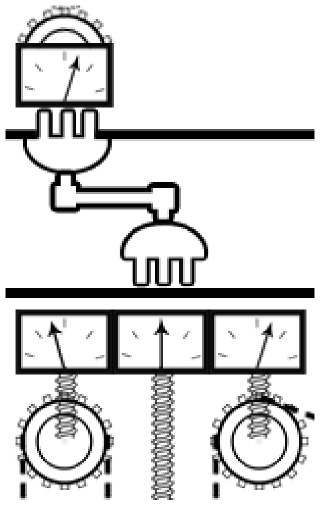
<table>
<tr>
<td>

Image 3.14 - Method call with parameter</td>
<td>

```
public Cart () {
   function main (){
      engine(Knob.Value)
   }
   function engine (valueParameter){
      switch (valueParameter){
         case left:
            LeftGear.Activate()
         case right:
            RightGear.Activate()
         case middle:
            MiddleGear.Activate()
      }
   }
}
```

</td>
</tr>
</table>

The element on the top end of the wire (the gear integrated with a meter) is specifically created for passing a knob value to another function. We called this component the gearOmeter. The gearOmeter has the same appearance as the robOmeter, to emphasize the link between these two components. To visualize the connection, the gearOmeter will light up when it is properly connected with a knob set to a corresponding value. The corresponding value on the robOmeter will light up as well.

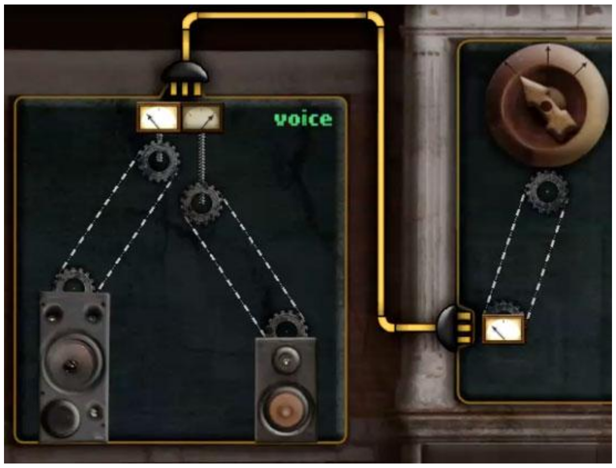


Image 3.15 - gearOmeter transmitting a value to a robOmeter

3.6.5 Method Call - Without Parameter

In computer programming, one method can call another without the need of passing a parameter. Therefore, the combination of knob and robOmeter becomes unnecessary to represent this concept. We already discussed a metaphor for a single activation instruction, namely the one-shot activation button: at this point, we need to implement the same logic, but with different actors. It is not the user activating a command anymore, but a function calling another function.

We had different design possibilities here. One of them would have been having an output gear in the calling method, an input gear in the called method, and a wire connecting them. As long as the output gear would keep turning, the gear on the other end of the wire would keep turning as well. This very straightforward and simple approach would have suited well the puzzle structure, but it was too mechanical: it lacked the idea of *calling* a method as an explicit action. Given that the player could not (and should not) activate method calls manually, we decided to create another entity that would perform this action: a little robot placed on a gear.

Since the robot is rotating on a gear, the most natural device we could envision for it to activate was a lever. The robot pulls the lever to call the function on the other end of the wire. A method call without parameter can either activate a gear that will start rotating, or activate an external machine part - for example, the ‘wing up’ action in the first puzzle.

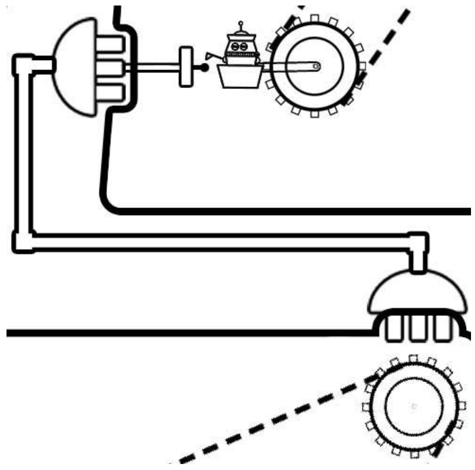


Image 3.16 - Method call without parameters, activating a gear

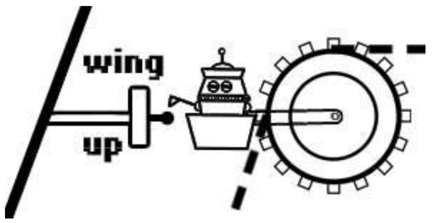


Image 3.17 - Method call without parameters, activating an external machine part

```
public Owl {
      function main () {
      activateLeftWing()
}


       function
activateLeftWing() {
            Gear.Activate()
            // if the belt
is connected
            WingUp()
      }
}
```

This latter case is especially powerful in representing a call to an *external* function, as for example a call to a function included in a library. The user has not programmed said function and does not need to understand or even access its inner logics. He just needs to know how to call that function, and what the output will be. Equivalently, when activating the ‘wing up’ machine part, our player does not know and does not need to know the inner functioning of the mechanism raising the wing. Knowing that activating that function will produce the desired output is enough.

3.6.6 For Loop

A for loop is represented by a counter placed on a method call. To execute an action a specific number of times, the design had to fall back to showing numbers.

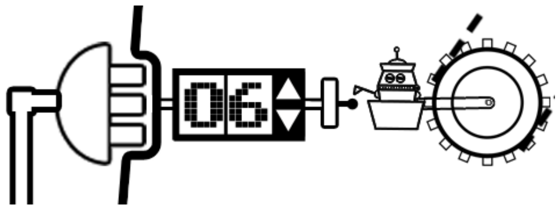


Image 3.18 - puzzle element - counter

```
int counter = 6   // set by user
for (int x = counter; x>0; x--;)
{
    functionXY();
}
```

In the image above, the counter will progressively decrement each time the robot pulls the lever. When the counter reaches zero, the method on the other side of the wire will not be activated anymore. This visual representation is actually quite close to the programming concept.

3.6.7 Sequence / Loop

This element is perhaps a more direct and easier noticeable representation of a programming structure. A programmable knob is a mix of machine parts in a monolithic and rather complex structure, which is by no coincidence introduced in the last puzzle of the game.

A programmable knob consists of an auto-incrementing knob, which will automatically change its value over time. A ‘power on’ button activates it. The knob is connected to a row of robOmeters, to check its value and perform different operations accordingly. We decided to use existing components in the hope that by aggregating elements already known to the players, the functioning of this new machine part would be easier to grasp for them.

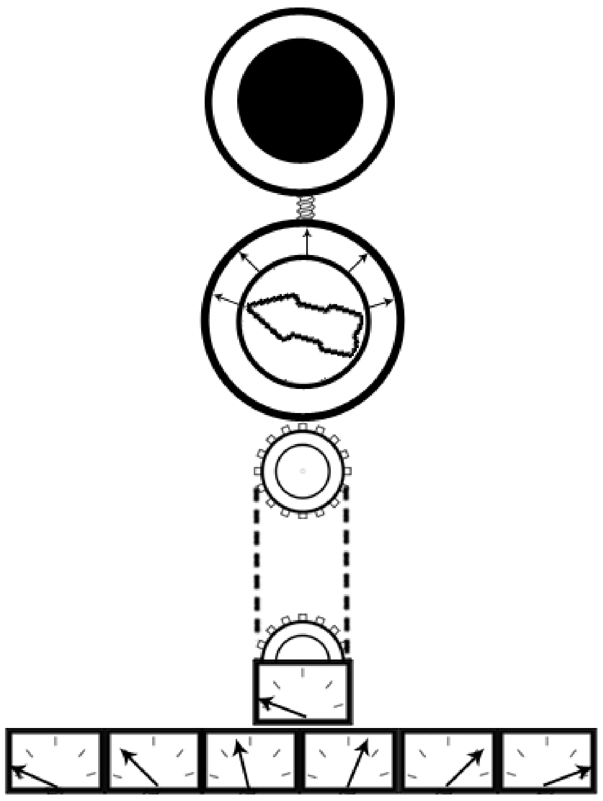


Image 3.19 - puzzle element - programmable knob

```
while (Button.IsActive) {
   Knob.Value++
   if (Knob.Value > max)
     Knob.Value = 1;
   switch(Knob.Value){
      case 1:
         Gear1.Activate()
      case 2:
         Gear2.Activate()
      case 3:
         Gear3.Activate()
      case 4:
         Gear4.Activate()
      case 5:
         Gear5.Activate()
      case 6:
         Gear6.Activate()
   }
}
```

This is the first element introducing a control that is not executed in real-time. The knob and the on/off buttons, operate in fact under the user’s direct control, while the programmable knob executes a series of predefined instructions. This machine part is designed to introduce the concept of sequentiality. A computer program like a game is obviously made to react to user’s real-time input, but it still executes a series of predefined instructions. Especially in the early stages of learning how to program, the process often consists of specifying a short series of commands, launch them, and see what happens: that is the process we tried to recreate when the player approaches this machine part.

3.6.8 Synchronization

The representation of the different order in which commands can be executed, on the other hand, is perhaps the most abstract metaphor of the game. The small robots placed on fixed gears will rotate with them, activating a method call through a lever when a certain position is reached. If the gear keeps rotating, that method call will be activated repeatedly.

Although quite far from its pseudocode representation, this approach has two major positive aspects. First and foremost, the possibilities for synchronizing the commands are limited to four options. Once the player understands that robots starting at different positions execute commands at different times, synchronizing the robots inside a single method is very intuitive. Obviously the other way around is also valid: robots starting in the same position will activate the levers at the same time, allowing for simultaneous method calls.

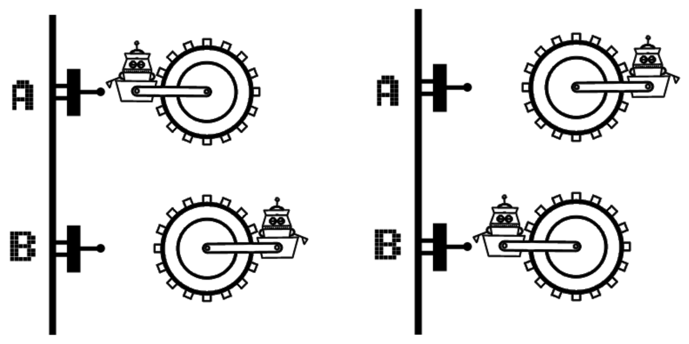


Image 3.20 and 3.21 - Robots on gears - v(1) left and v(2) right

```
version (1)
FunctionA()
FunctionB()

version (2)
FunctionB()
FunctionA()
```

3.7 Puzzles and learning curve

The principles in the skill development category focus on the player's developing mastery of a skill. This is an important component in a gamer's positive game experience. It is not, however, merely the development of a skill, but rather it is the pacing of learning that skill that divides a good game from a bad one (Desurvire & Wiberg 2009, p.8)

We designed and implemented five puzzles, each focusing on a different challenge and increasing in difficulty. The aim of the puzzle design is to allow children to play inside their zone of proximal development, giving them a challenge they would hardly be able to complete without any help, while keeping them from falling into frustration; the design aims for the 'pleasantly frustrating' principle to keep the flow.

To almost all of [the children] being challenged in a game meant that they would not be bored. In contrast, they frequently referred to school mathematics as boring. [...]

Children need to be constantly challenged and seem to thrive on it. (Sedighian & Sedighian 1996, p.5)

Scholars argue that a good level of challenge is not only beneficial for children's engagement, but also for the transfer of educational concepts. Wilson (et al. 2009, p.36) for example, propose that "*as the challenge feature in a game increases, so will declarative knowledge and learner's retention of that knowledge*", but at the same time warn against the counterproductive effects of an excessively high level of challenge, resulting in hindrance for the learning progress. Adequate scaffolding can potentially prevent such a scenario from happening, as it can cater for the needs of different players.

The number of elements players need to understand is constantly increasing, as we introduce something new in every puzzle. The number of actions available to the player is quite limited (placing and removing gears, belts and wires, reposition robots, operate buttons and knobs) but constantly changing scenarios and the introduction of new elements prevent the feeling of repetitiveness.

Following Gee's principle of 'system thinking' (2005c, p.14) claiming that "people learn skills, strategies, and ideas best when they see how they fit into an overall larger system to which they give meaning", each puzzle is designed as a complete machine, to show how relationships between game components work. Even if players do not need to understand the functioning of the entire machine in the first puzzles, when they are confined to work on smaller sub-tasks, they will always be able to see the bigger picture, so to have the chance to understand the interplay of all the puzzle elements and how their interactions affect the machine as a whole.

3.7.1 Zima, the Owl Puzzle

Rummaging through a pile of scrap metal while waiting for work, Zola will find Zima, a robotic owl with a broken wing. Bent will challenge the player to fix it: the goal of the puzzle is to get Zima's wings flapping at the same time.

In this first puzzle we introduce the controls and the game interface. A very short scripted tutorial given through the standard dialogue system precedes the actual problem-solving phase. Bent will guide the player in the very simple tasks of moving the visual around and using the zoom function, allowing them to familiarize with the controls.

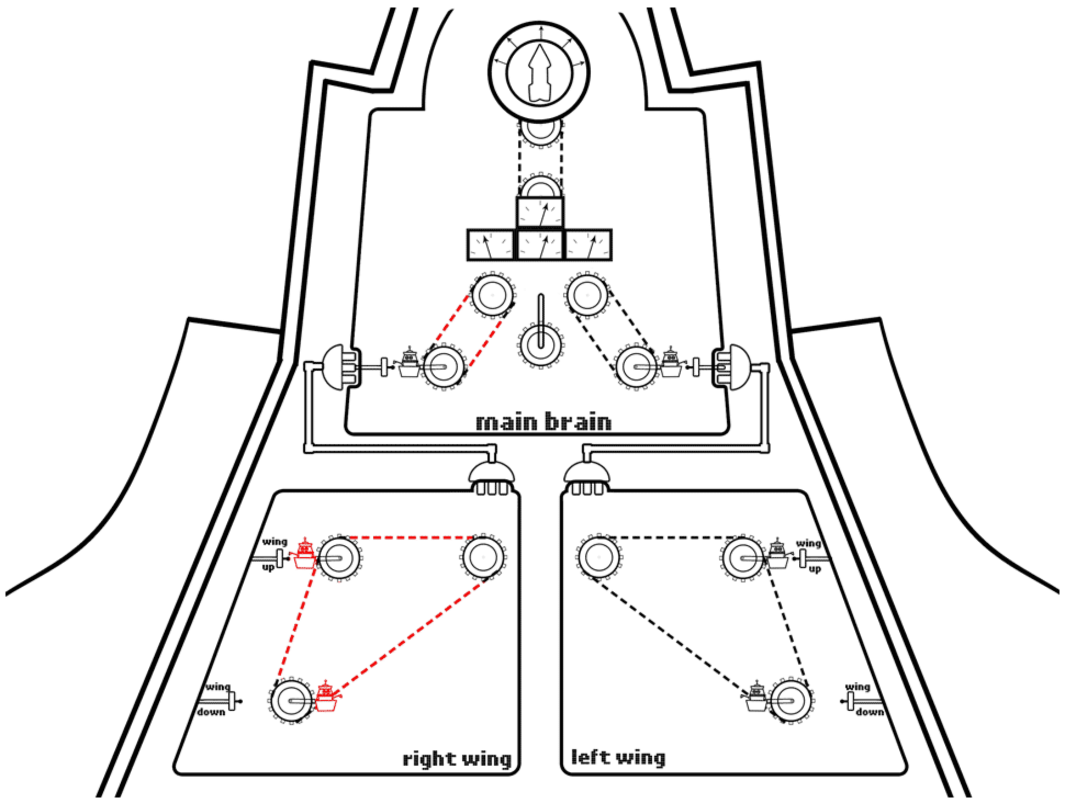


Image 3.22 - Owl Puzzle - solved state (parts initially missing or misplaced are marked red)

To solve this puzzle, the player needs to learn how to place elements like gears and belts, and interact with fixed elements in the scene (the top knob and the robots on gears). By operating on the knob, players will also be exposed to its functioning logic, although they do not need to fully grasp it to solve the puzzle. What they do need to understand are the following concepts. Belts connect gears to transmit movement. Two belts are missing, the player needs to connect the isolated gears. Robots on gears can be moved around to synchronize actions. The robots on gears in Zima's right wing are misplaced and the two wings don't flap together. The player needs to synchronize them. The knob activates the machine and different positions allow for different actions.

3.7.2 The DJ Machine

The second puzzle is the first actual task assigned by the repair shop. The DJ robot will complain about some missing parts: in fact, this puzzle is all about placing belts and wires. The structure of the machine is centered around the main control method and its four knobs. One knob allows selecting which one between two or three speakers in another method will be powered. The goal of the puzzle is to connect the knobs to the speakers so that for different knob positions different sound loops play.

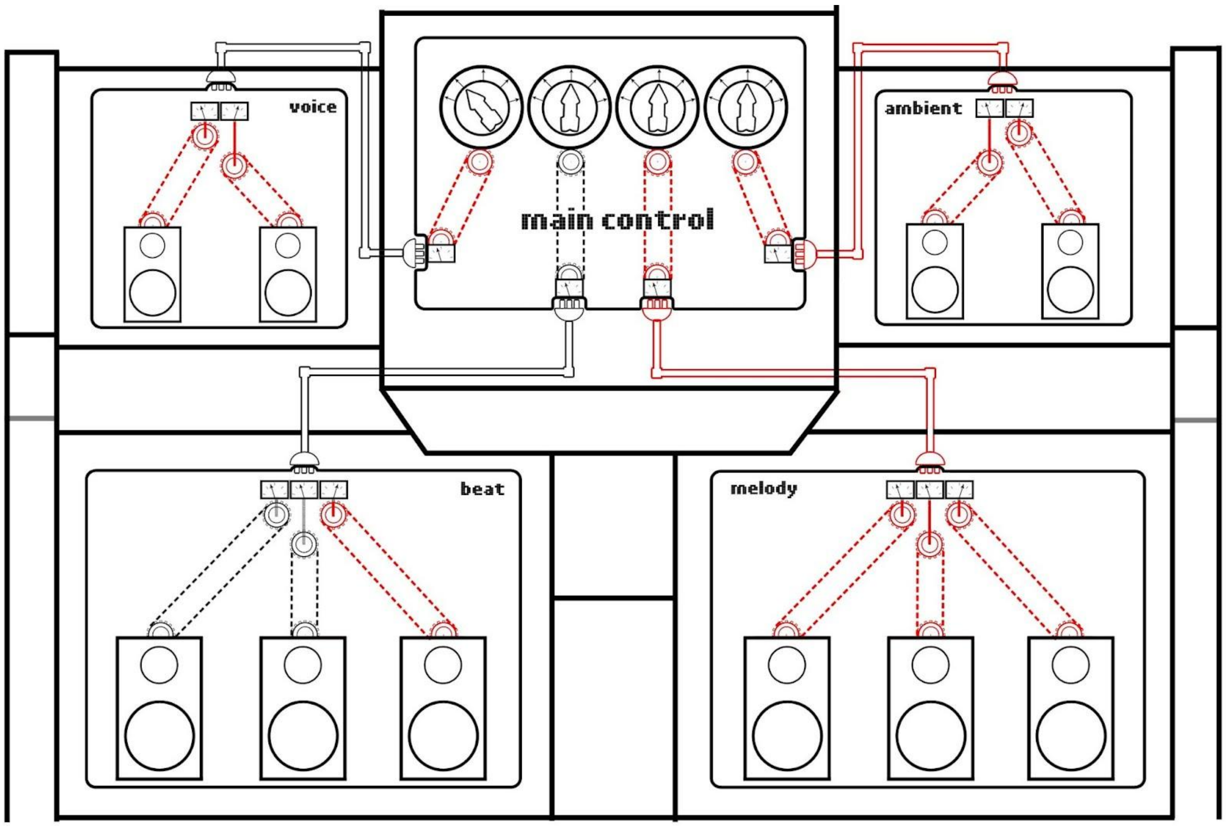


Image 3.23 - DJ Puzzle - solved state

The concepts players need to understand to solve this puzzle are the following: Knobs are meaningful when connected to a gearOmeter. Wires allow communication between a gearOmeter and a robOmeter placed in another part of the machine, effectively connecting two methods. Once the machine is fixed, the player can play around with the knobs to mix different loops of voice, beat, melody and ambient sound.

3.7.3 Lorry, the Bar Puzzle

The next puzzle takes place in Lorry's bar. Lorry, the owner, will ask Zola to fix the blinking signboard, setting it so all letters will blink at the same time.

This puzzle is all about synchronism inside a single method and between methods calling each other sequentially. All the belts are initially missing, but by now players know what a belt does: placing them gives players a sense of familiarity and success to start the puzzle with.

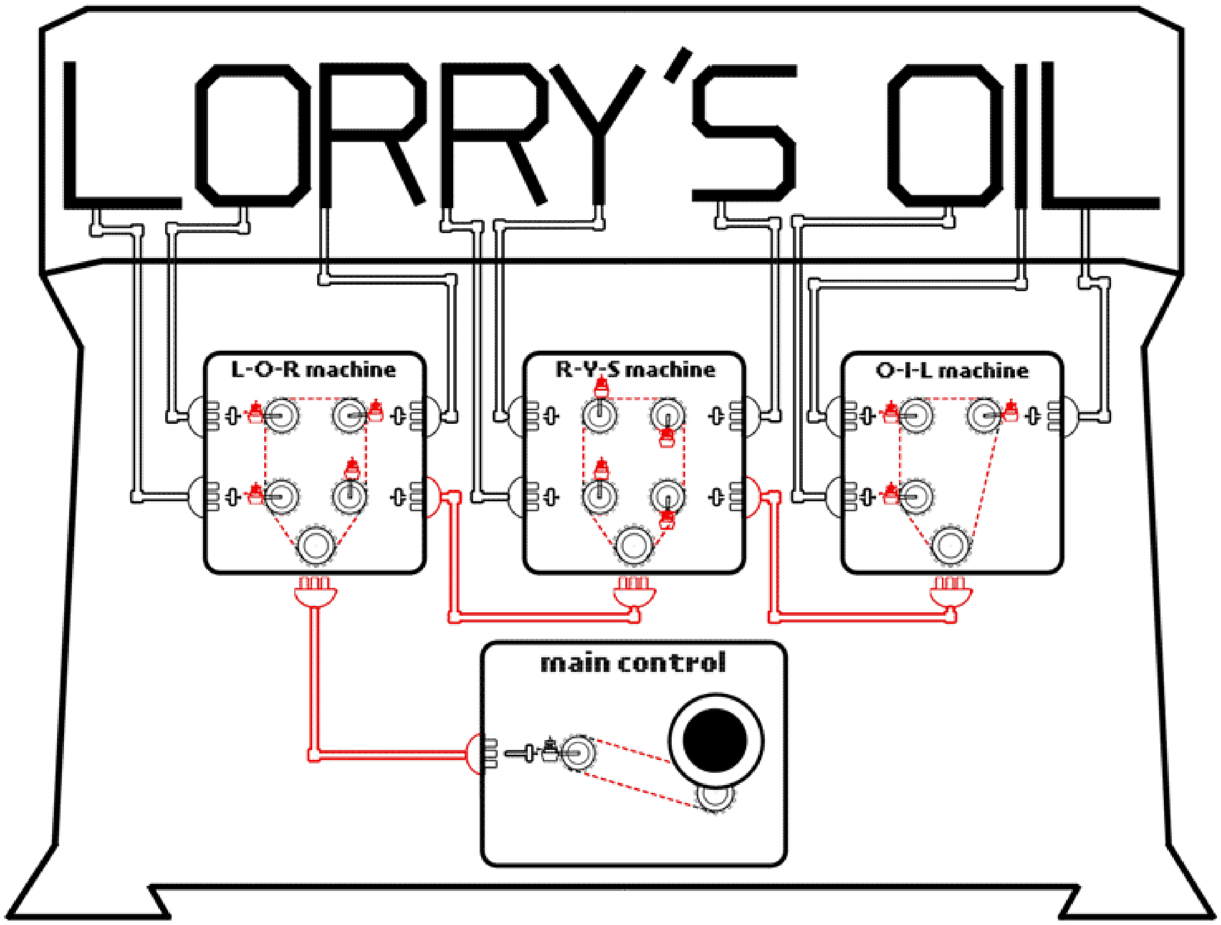


Image 3.24 - Bar or Glowing Letters puzzle - solved state

We introduce a new element, the activation button that starts the machine. A series of methods controlling the blinking of different letters must be connected with a wire. The synchronism between actions we introduced in the owl puzzle is taken to a higher order of difficulty. In fact, the player has to coordinate 9 action calls spread across 3 different methods. Those methods are called at different times: the robot on the gears calling the next method in the L-O-R and the R-Y-S methods are in fact fixed, in order to draw the player's attention to how the wires determine the order in which the methods are called, and also to increase the puzzle difficulty.

3.7.4 Ivan, the Cart Puzzle

The fourth puzzle is a repair commissioned by Ivan, the owner of the roller coaster. Zola will be asked to fix the self-propellent roller coaster cart. The cart has an engine and can travel at different speeds: the knob is supposed to activate a different number of wheels for each distinct value. A shovel is installed in the front, to push potential obstacles out of the way. Another red button represents an emergency off button: when activated, it removes the gear connecting the knob and the engine.

When this puzzle was originally designed, the intent was allowing the player, once the repairs were made, to drive the cart on the rollercoaster track in a mini game scenario: therefore we integrated the emergency stop button and the shovel. Unfortunately, implementing the mini game turned out to be out of scope, and it had to be dropped.

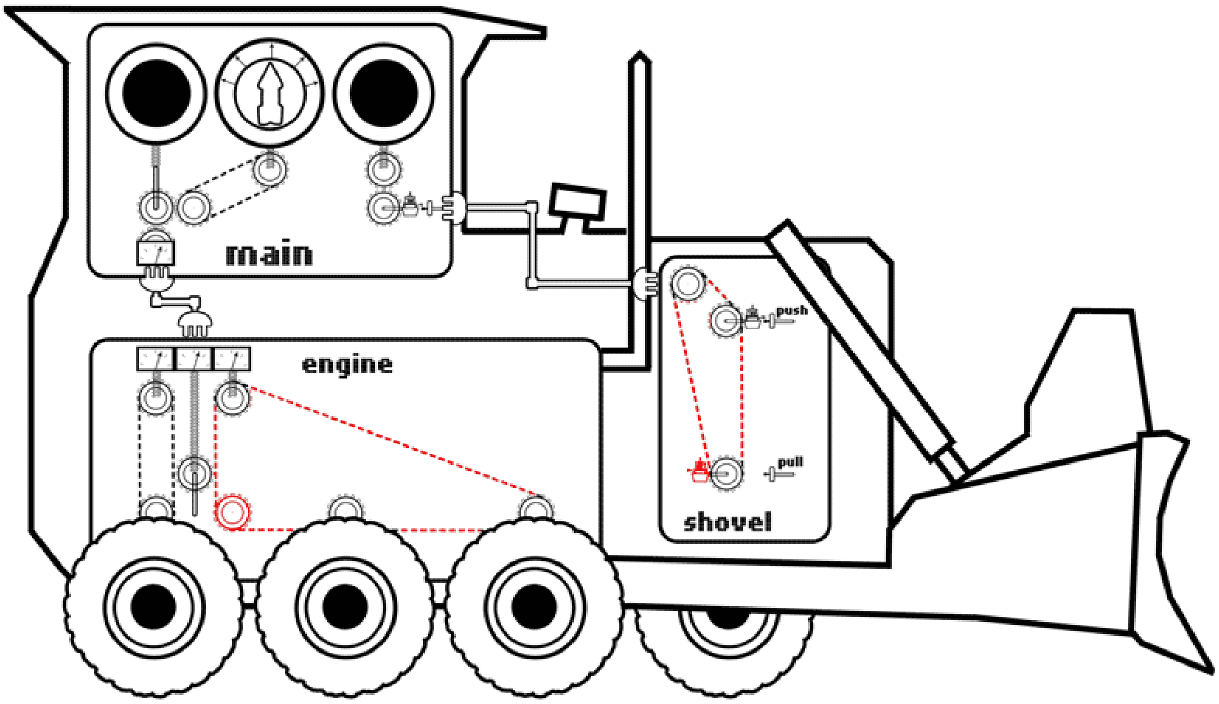


Image 3.25 - Rollercoaster cart puzzle - solved state

In this puzzle we introduce a new element, the shovel activation button, which is a *one-shot* button. The complexity of this puzzle does not consist of understanding the functioning of this new element: in fact, the one-shot button is a simple variation of a very natural concept - the activation button). The shovel itself is a repetition of the owl wing structure. The real challenge of this puzzle resides entirely in how the elements in the engine are arranged.

To separately activate one, then two, then all three wheels at a time the player is forced to understand the functioning of the gear on rail. The same elements form a combination that the player has never seen before, forcing them to think ‘outside the box’: a precious skill in problem solving.

3.7.5 Ruby, the Crane Puzzle

The last puzzle is the repair of a robotic crane. The goal of the puzzle is to fetch a teddy bear with its claws and drop it in a shaft. The machine has many sub methods that need to be coordinated to achieve this, each taking care of a different action: moving the crane left and right, up and down, opening and closing the claws.

Specifically for the purpose of triggering different actions in a predetermined sequence, we designed and introduced the programmable knob. Because of this component, this is the puzzle that gets the closest to an actual programming exercise, as it has sequentiality.

The player can connect the robOmeter gears to different outputs and place the wires in the right part of the machine, deciding the order of the actions; but they have no control over the fact that the crane has to be programmed in advance to execute a series of actions, then activated in order to observe the outcome.

The idea behind the design of this puzzle came partly from looking at Logo’s turtle graphics. One of the great features about Logo was that user was easily able to observe if the program was performing as expected, just from looking at the drawing the turtle was making. This made it possible for children to debug their own code without the help of a teacher. We believe the crane puzzle shares this feature, but instead of turtle graphics it uses the movement and action sequence of the crane. By observing how the crane behaves we anticipate that the children can understand how the machine works and how to fix it.

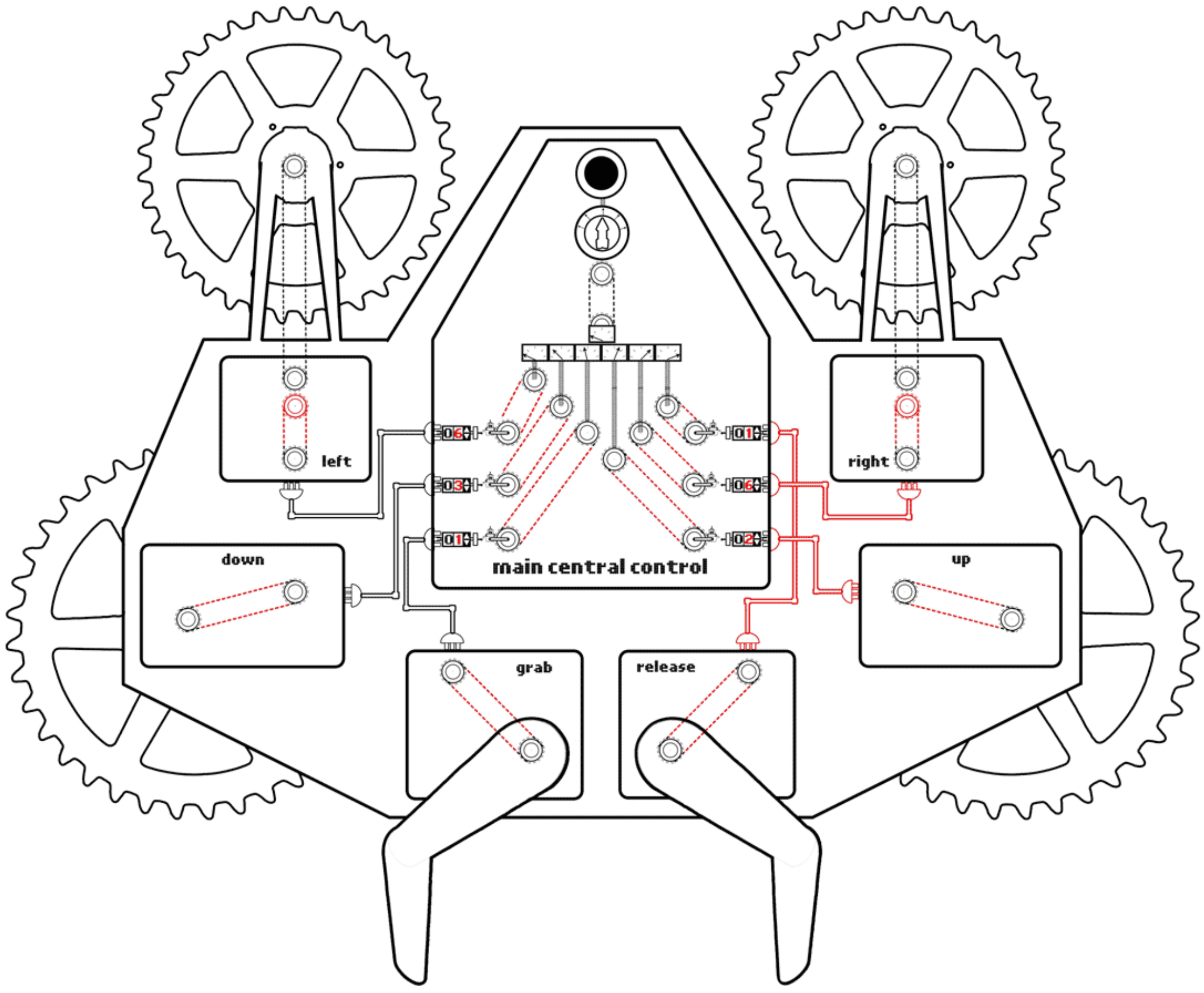


Image 3.26 - Crane puzzle - solved state

The mirrored puzzle structure as a scaffold for solving the puzzle that we implemented in the first two machines here works against the player. The wires on the left side of the main method are in fact already linked to the commands in the right sequence: at first the machine will go left, then down, then the claw will grab. If the player does not pause to think about how the cables on the right side should be connected, they might just replicate the same structure: top output to top method (‘right’), middle output to middle method (‘up’), bottom output to bottom method (‘release’). If so, and assuming that the belts in the main method are connected as in the image, the machine will release the object right after grabbing it, making it impossible to carry the teddy bear to the shaft.

In the main control machine we also introduce decrementing 'for loops'. Movements need to be repeated a certain number of times to reach exactly the position for grabbing the teddy bear, then to move back to the shaft. As in Lorry's puzzle, this forces a trial and error approach.

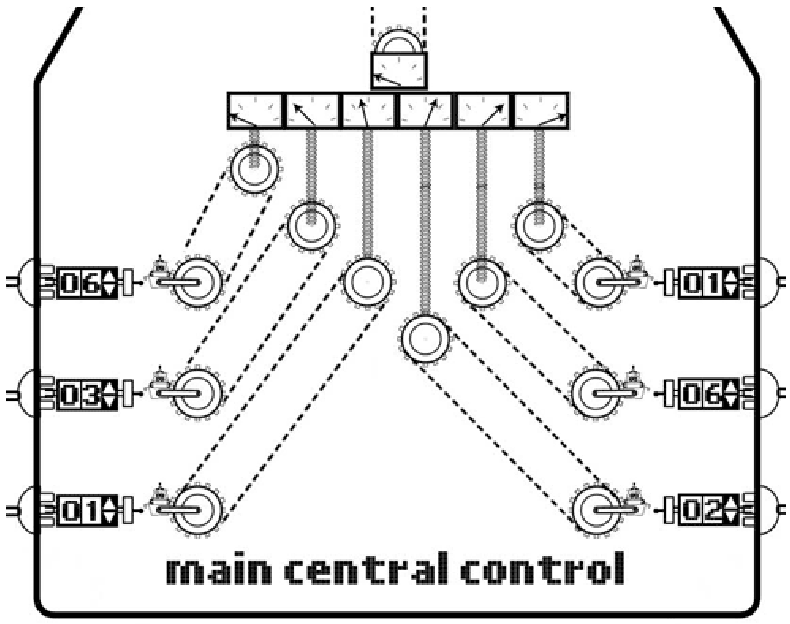


Image 3.27 - Detail Crane Puzzle - Main method

3.8 Scaffolds

Designers should always consider the target audience when designing and include adaptive features to adjust pace and type of information, according to individual learner differences (Honey & Hilton, 2011). Given that we cannot assume that any teacher or tutor would necessarily be present in an informal playing setting, the following help options play a crucial role in the overall balance of the game. The '*instant feedback*' system. When the player performs a particular action for the first time, like clicking a button or a particular component, a dialogue speech bubble will pop up providing introductory information about it. This will happen only once to avoid a constant interruption of the game flow; we designed another on-demand help option (the Owl) in case they need more informations later on about a particular component. The list of actions designed to trigger an instant feedback include clicking on interface controls (gear, wire, belt and delete buttons), elements in the scene (gears, gearOmeters, robOmeters, belts, wires, buttons, knobs, programmable knob and loop counters) and some particular cases, like when the player creates a group of stalling gears.

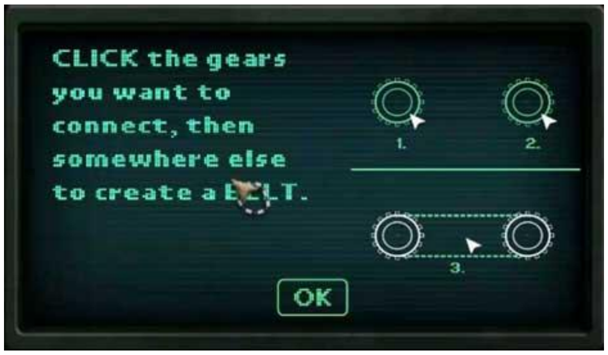

CLICK the gears you want to connect, then somewhere else to create a BELT.
1.
2.
3.
OK

Image 3.28 - Instant feedback - Belt mode

Another help option is represented by Zima the owl, also called the information-on-demand system. By clicking on the owl in the top bar, then on some machine part in the puzzle scene, Zima will give some information about it. The list of parts for which help is designed includes gear, gear on rail, robOmeter, belt, wire, gearOmeter, robot on gear, and knob. This option will be available only after the first puzzle is completed, in which the player fixes Zima the owl. Ideally, Zima knows exactly the same as the player, as it cannot solve puzzles or give hints about how to do so, but provides some informations about isolated parts. Providing the player with ‘smart tools’ enables them to store the knowledge they acquired in-game within game artifacts and characters; in other words, distribute their knowledge (Gee 2005c).

Bent, and occasionally Hayden will appear in the top bar with hints for the player. These are predefined, short sentences that will be given through a speech bubble after a certain amount of time has passed from the start of the puzzle. Providing the player with ‘smart tools’ enables them to store the knowledge they acquired in-game within game artifacts and characters (Bent and Hayden in our case). Once they are provided, these hints remain accessible through the same interface. Given the problems we observed in the early testing phases for players to notice this help option, the availability of a new hint is indicated through a blinking light bulb, a sound and a sparkle effect. Players can see another smaller light bulb next to the character with the new hint and read it by clicking on them.

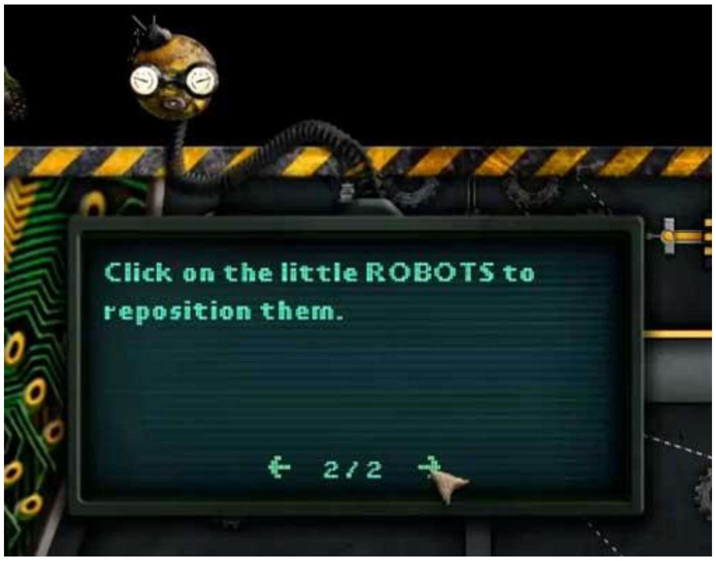

Click on the little ROBOTS to reposition them.

Image 3.29 - Hint system

The hints in the first puzzles are as short and clear as possible, aiming to instruct the player on how to interact with the puzzle elements: "*Click the arrow KNOB on top of the machine to make the wings move*", "*Click on the little ROBOTS to reposition them*". When proceeding into the game, hints orient more towards directing the player's focus on a subset of the puzzle, limiting the range of the task at hand ("*It is best if you start fixing the OIL machine on the right and work your way over the middle to the left*"), clarifying the puzzle ultimate goal ("*The cart needs to speed up, break and push things out of the way with the shovel to be allowed on the roller coaster*") and providing encouragement ("*Don't give up yet, I'm sure you will figure it out*").

The symmetric setup of the first puzzles is also a scaffold. In the first puzzle, the wing that needs to be repaired is an exact copy of the other one, but mirrored. We point this out immediately through the hint system in a subtle way: Zima will offer a hint after 10 seconds of playing time ("*my LEFT WING moves fine, but I can't move the RIGHT WING at all!*"). Later on, after five minutes, we explain this in a more explicit way through Bent ("*Copy the structure of the left wing into the right wing. That should do the trick.*"). This intrinsic help is also present in the second puzzle, where not all the belts are missing. The player can observe and mimic the structure of the existing parts. Once again, the first hint points out this aspect - "*Check out the parts that seem to be working. There are BELTS connecting the GEARS and the SPEAKERS.*" Although the indication of what to do is quite clear, this support cannot be considered sufficient for every player to solve the puzzle. In the first puzzle especially, players might still have difficulties interacting with the interface, trying to understand how to place belts, reposition robots on gears, activate the knob, and understanding the consequences of their actions. The hint might reach them at the wrong time, when they are focused on something else, so they might decide to ignore it altogether. This scaffold is removed from the third puzzle on.

After all hints have been given, a certain amount of time passes so players have a chance to keep trying and eventually solve the puzzle. If that is not the case, Hayden will provide the machine's *blueprint*. The blueprint is a schematic representation of the puzzle in its solved state that overlaps the current scene. Players can see the solution, then close the blueprint and implement it in the machine. The blueprint is accessible through an icon appearing in the top bar.

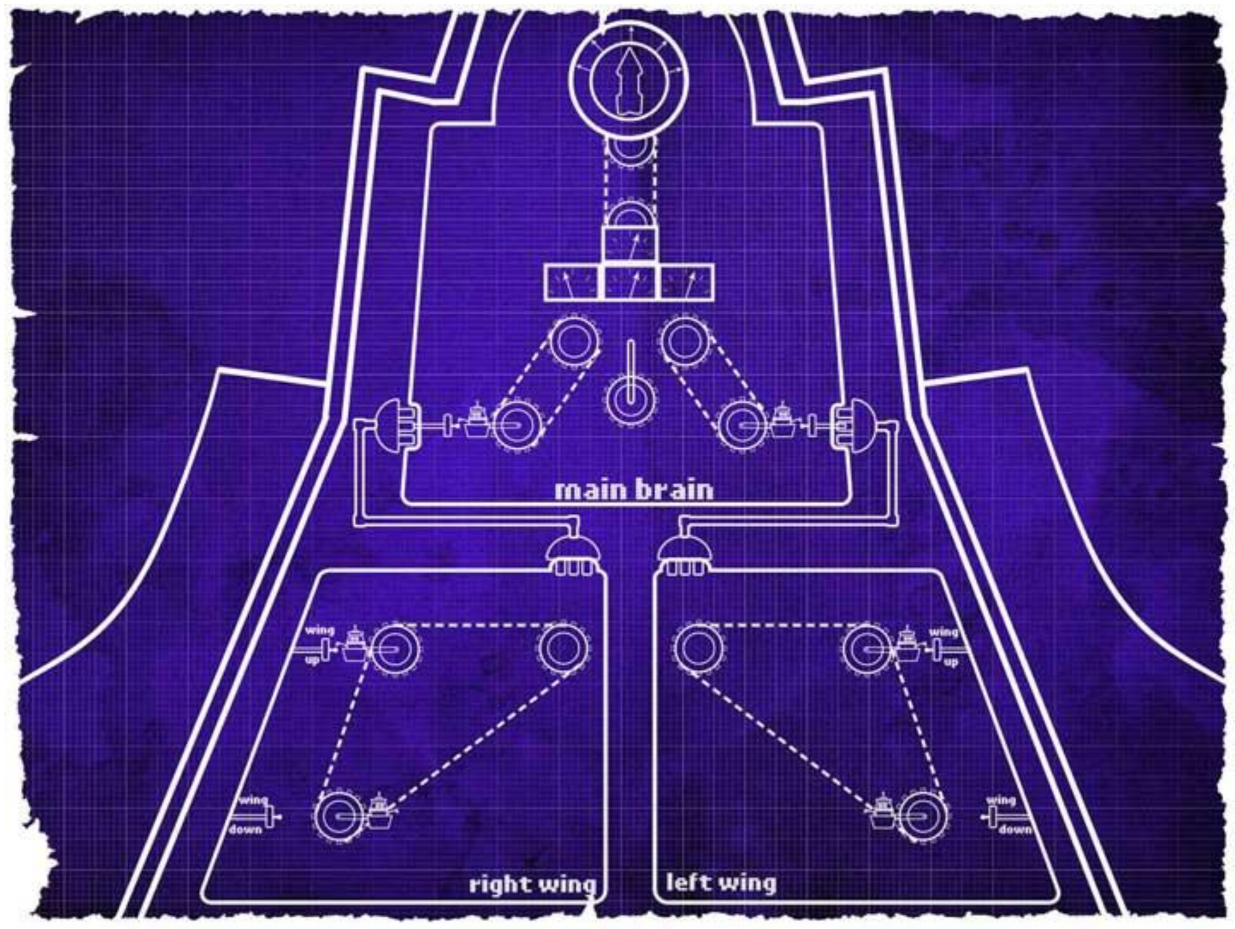


Image 3.30 - Blueprint for the owl puzzle

Ideally, children are playing in their ZPD throughout the entire game, therefore we never fade away the scaffolds. An exception is made for the *instant feedback,* which is triggered just-in-time; the other help options (once available) can be accessed on demand and they are not progressively taken away. The lightbulb is meant to attract the player’s attention, notifying them that a new hint is available, but it does not force the player to read it. Following Gee’s principle of customization (2005a), players can choose the scaffolds and help options they want to use.

Talking about customized styles of play, Machineers allows for some freedom. Many of the puzzles have multiple ways of solving them; often players can also quit a puzzle if they find it too hard, go back to the repair shop and ask Hayden for another task to unlock the next puzzle. Originally we designed two different main characters with the idea of letting the player choose between a boy’s (Mylo) and girl’s (Zola) avatar. In the end, we decided to only use Zola, focusing our development time on other features. The intention was that boys would not be annoyed by playing a girl, while girls would be encouraged in identifying with their new virtual identity and her successes.

3.9 User Interface

Preventing usability issues was the focus of the whole interface design process. At the same time, an effort was made to provide a good number of visual stimuli as well, with further focus on those elements of the user interface related to the help scaffolds.

3.9.1 Adventure mode

In adventure mode, the only element of the user interface is the mouse. The top priority is to communicate which elements in the scene can be interacted with, and which ones cannot. When hovering over a door, a robot, or any object with a function, the mouse cursor will change and the object itself will be highlighted. When the dialogue system pops up, players need to be given two basic informations. The first is which robot is talking, which is represented by an animation. The second is, when multiple answers appear on the screen, which answer is actually selected, which is represented by an arrow appearing next to the selected answer on mouseover.

3.9.2 Puzzle mode

Inspired by the simple user interface of the game Machinarium (Amanita Design, 2009), we decided to use a pop-down menu on top of the screen, called *top bar*, which is partly visible and extends to full size once the mouse cursor hovers over it. The top bar contains icons of the other game characters who provide access to hints and blueprints, as well as a button for leaving the puzzle. A small light bulb next to the characters head indicates that a new piece of information is available. To further indicate the availability of hints when the top bar is closed, a bigger light bulb appears on the edge of the closed top bar.

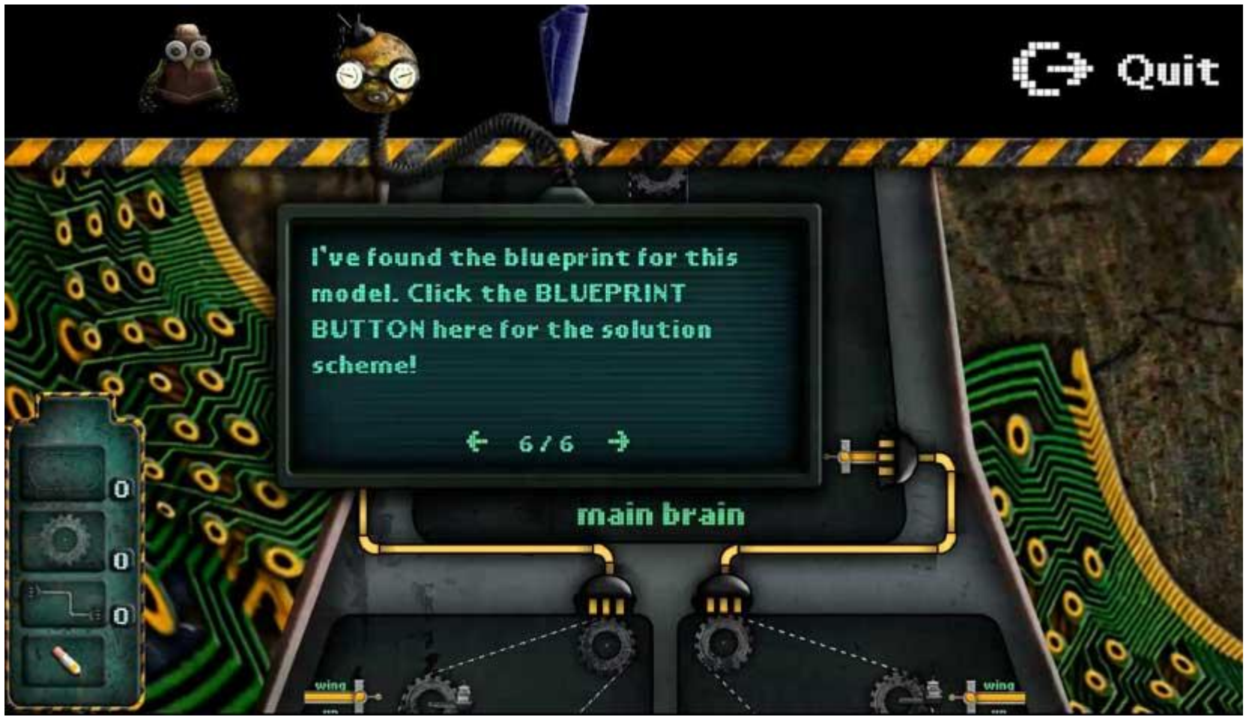

I've found the blueprint for this model. Click the BLUEPRINT BUTTON here for the solution scheme!

Image 3.31 - In-game screenshot - Owl puzzle

For installing puzzles elements, players need a tool menu. To keep the screen as clean and well-arranged as possible, we initially decided not to integrate the context menu in the interface, but instead bring it up on right-click. This would further enable the free positioning of the menu according to the players preference. Due to usability issues we later decided to go back to a static interface. The on-screen menu is placed on the lower left corner on the screen, from where players can select the elements to be placed in the scene, and access the erase mode to remove them.

3.9 Summary

In this chapter we covered the main focus areas of the design of Machineers. First, capturing the player's attention by framing the challenge in a rich adventure world, providing plenty of stimuli, play agency, and positive feedback in accordance with the design principles outlined in chapter 2 Background. Second, translating procedural concepts into game puzzle elements and creating meaningful scenarios where players could get introduced to and explore the metaphors and their logical connections. Third, providing an adequate number and variety of help options and anticipating usability problems to prevent as many players as possible from being discouraged and dropping the game prematurely. The choice of implementing a girl as the main character was also made in the attempt to facilitate identification of female players.

The exclusion of variables from the procedural literacy concepts included in the design was based on the decision to spare the young audience from the logics of data manipulation. In the shaping of the basic puzzle elements, a major focus was put on the play experience. More complex parts, with a more direct link to the programming concept they represent, such as the programmable knob and the for loop counter, are introduced later in the game.

The informal learning context in which Machineers is meant to be played, informed design decisions regarding graphics and sound polishing, and scaffolds. We were planning on implementing a sandbox mode for children to play in after completing the game, in accordance with Egenfeldt-Nielsen's approach to constructivism in educational games, but such a feature turned out to be unfeasible within the scope of this project.

4 Usability Test

The key value of the playtest, however, lies in identifying problem areas. We think of playtests as a tool to help designers understand an issue, so they can think up a workable solution. Importantly, if a designer makes substantial changes to the game based on playtest results, we can iterate to ensure the changes were effective in improving the game. (Davis et al. 2005, p.10)

This chapter deals with the first of three test phases, which were conducted at the end of the first production phase when all the basic game elements (adventure mode, dialogue system, hints) and the DJ puzzle were implemented. This evaluation mainly focuses on interface elements, controls and overall usability issues. After solving them we will be able to analyze problems concerning immersion and learning process in further tests unbiased.

4.1 Preparation

For the first testing phase we conducted a qualitative assessment with four other game design students, who were asked to playtest the early version of our game while following the Think-Aloud protocol (Hoonhout 2008). Since the testers we chose are well accustomed to participating in playtest sessions, they delivered a very thorough examination of the game, e.g. attempting to break it and perform undesired actions, while expressing their intentions and annoyances in as much details as possible. All issues were recorded with pen and paper or noted down digitally.

We intended to use a new unconventional test method called *peer play*, which we designed for the next test phase, in order to obtain as much information as possible from our younger and possibly more hesitant testers. We tested this approach with two students who were less accustomed to gaming and playtesting in a comfortable environment and screen-recorded the session.

The peer play method is inspired by Jettie Hoonhout, talking about a number of studies that used an alternative to the Think-Aloud protocol. They let their testers play in pairs while recording all utterances: "*the participants, children between nine and eleven years old, were not specifically asked to think aloud, but since they were playing in pairs, and depended on each other for good game results, it was inevitable that they had to discuss game options and tactics*" (Hoonhout 2008, p.71).

4.2 Navigation

At this point the navigation for both modes involved moving the mouse cursor to the sides of the screen, which would then start panning. We realized during this test session that players did not find it very intuitive to navigate the way we intended them to. Instead, players would e.g. try to click-drag to move around - obviously a habit from navigation in other games. We discussed several different options of how to solve this issue, but did not have the time to implement any of them before the next test phase two weeks later. We wanted to observe how our target audience would behave with the current controls in the next test session, because even if they were as comfortable using computers as our fellow student testers, they might have different expectations and reactions towards navigation and controls in games.

4.3 Interface

We initially used a radial context menu which appeared on right-click, because we wanted to keep the screen as clean as possible in order to not overwhelm the player. Players showed difficulties when trying to access the context menu per right-click: keeping the screen clean, in fact, came at the cost that players needed at least three clicks to place an element: open the menu, select the component, place it. This been observed, we decided to fall back to a static on-screen control panel. By keeping the menu on the screen not only did we streamline the most recurring game function, we also made the interface more intuitive, as controls for element placing are immediately visible. Knowledge is placed ‘in the world’ so there is no need for a tutorial explaining players how to bring the toolbar on screen (therefore placing the knowledge ‘in their head) (Norman 2002). Also, by removing this functionality from the right mouse button we were able to use that control for another basic function, namely navigation.

As it turned out that players did not perceive the screen as too crowded, but rather did not notice the toolbar immediately. This is why we decided to give the tool menu a hazard-striped frame to visually emphasize it. After the preliminary test with the target audience which is described in the next chapter, we decided to limit the number of puzzle elements (gears, belts, electric wires) to help directing the player towards the correct solution, we added a label next to the puzzle elements displaying the number of available elements (image 4.3). At this point we also removed puzzle elements from the toolbar that the user was not able to manipulate or include (robOmeters, rails), although for further designing the game we were still planning on enabling the player to control these features. These design decisions reflect Gees approach to tackle the ‘paradox of deep learning’ (2005b) mentioned in Chapter 2.

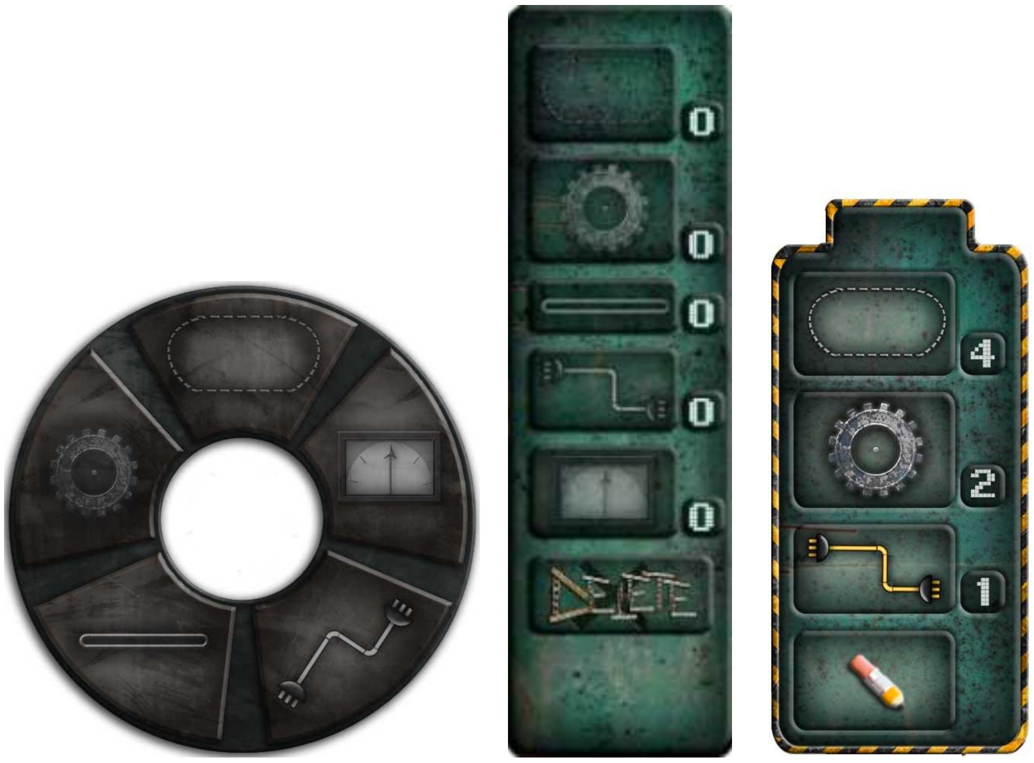
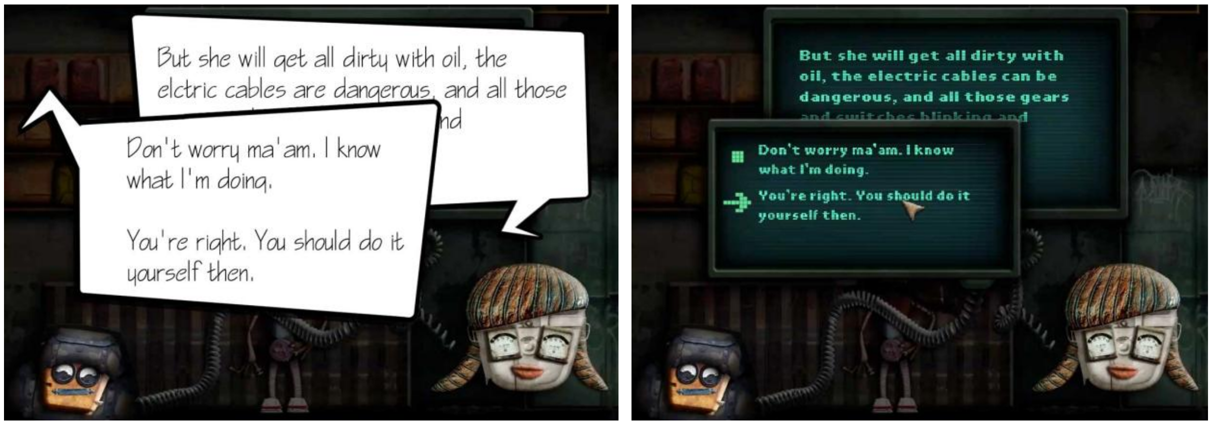


Image 4.1,4.2 and 4.3 - Context Menus - Initial version, after Test Phase 0, and Phase 1

Another issue the players mentioned, concerned the dialogue system. One player pointed out that the visual appearance of the dialogue boxes was not conform with the general visual style of the game, so we agreed to replace them with an old-school computer-inspired screen and use a light-green, retro computer font. The different answer options were displayed without a clear visual indicator, so by many testers they were perceived only as text. We added dots to emphasize the different options and an arrow to indicate which answer was selected.


But she will get all dirty with oil, the elctric cables are dangerous, and all those nd

Don't worry ma'am. I know what I'm doing.

You're right. You should do it yourself then.

But she will get all dirty with oil, the electric cables can be dangerous, and all those gears and switches blinking and

Don't worry ma'am. I know what I'm doing.

You're right. You should do it yourself then.

Image 4.4 and 4.5 - Dialogue Boxes - before and after Test Phase 0

4.4 Puzzle Elements

At this stage, players were allowed to adjust the arrows on the robOmeters by click-dragging them. The robOmeter, as explained in section 3.6.3, is the device that checks the input value for a method call with parameter. By dragging the arrow around, players were able to manually specify the activation value for each robOmeter.

As we found out, this created some issues. Not only did players not notice the possibility of doing so, we also realized that this gave them the opportunity to modify the puzzles in a way that could make them very hard to solve later on, so we decided to keep the meters at a fixed state. This meant we had to take away some control from the player to prevent them from wasting time and motivation on a problem that is not relevant for the learning experience. Once again, this decision reflects a counter-strategy to avoid the ‘paradox of deep learning’ (Gee, 2005b) and also the concept from Honey and Hilton (2011) that content not instrumental for the learning experience can and should be removed from the game.

Initially the robOmeters were simply named as *switches* according to their use as a programming concept - a name which indicated to the user that they were interactive, while their visual appearance did not invite for interaction. This conflict confused players, so we renamed the switches to *robOmeters* (many meters connected with screws to gears that can activate different parts of the machine) and differentiate them from *gearOmeters* (indicating that the meter is combined with and could be manipulated by a gear).

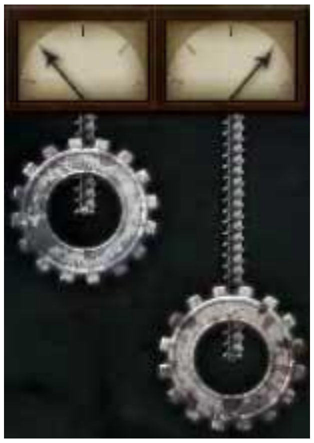


Image 4.6 and 4.7 - RobOmeters and GearOmeters

4.5 Help Options and Feedback

As most players were overwhelmed with the information from the hints available in the top bar, we chose to reduce the number of hints and rewrite them to make sure they were as short and precise as possible. We also felt players would benefit from a clear statement at the start of each puzzle, specifying the objective of the puzzle, and providing a clear goal (Gee 2008).

We further noticed that some players chose not to access the hints at all, partly because they tried to solve the puzzles without any help, and we agreed that there was some information about the puzzle elements and the use of the tools which was crucial and therefore decided to implement another scaffold we called '*instant feedback*'. Information would appear when the player enters another mode (e.g. belt mode, erase mode) or clicked an important puzzle element for the first time. This scaffold was intended to prevent impatient players from getting stuck entirely, prioritizing the 'just-in-time' over the 'on-demand'.

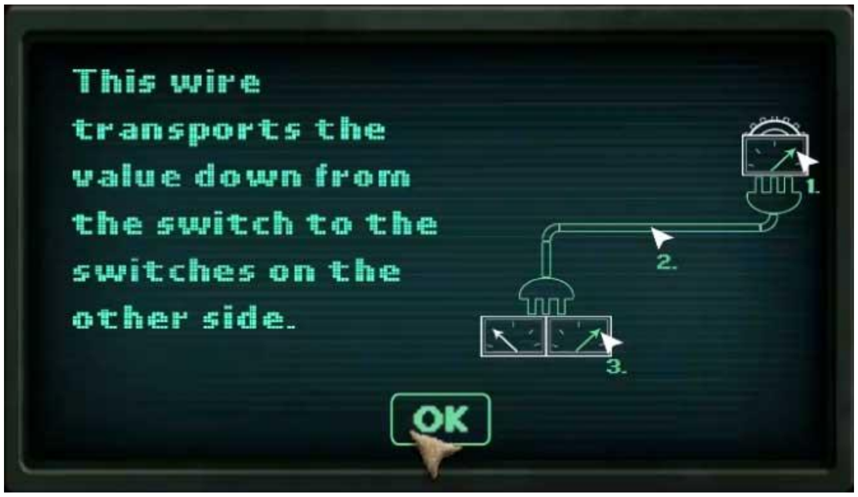

This wire transports the value down from the switch to the switches on the other side.

Image 4.8 - Instant Feedback for electric wires

Other feedback we received after all test phases were completed, indicates that a few number of players would still ignore all written information and get stuck in puzzles because e.g. they did not know that they could reposition the robots on gears or that they could delete elements.

4.6 General User Feedback

Although we did not specifically look for feedback on motivation and immersion, our players reported that they enjoyed the game very much and that they were looking forward to solving more puzzles, as soon as they were available. They expressed a great deal of appreciation towards funny and absurd dialog options even though most players generally picked the most polite answer option.

The DJ puzzle relies heavily on the player understanding the audio feedback, which is a much harder task for some people than for others. One playtester who was less musical than the others could not tell merely from the audio feedback if he was performing well. We realized that displaying the name of each method visually would help players like him understand the information from hints and the objective much better, and we were able to implement this feature after the next test phase.

4.7 Conclusion

We were positively surprised by the test results, as most of the problems that occurred were issues we were already aware of. The playtesters feedback gave us an idea of the best way to amend them, as they performed the action that seemed most natural to them and we were able to use this for developing solutions. This evaluation provided us with the necessary information to eliminate a number of major usability issues and helped us to improve our scaffolds so they would support the player better when solving puzzles. The data that had been gathered was then used to inform the design of further puzzles, dialogues, and scaffolds.

This test also confirmed the effectiveness of the peer-play approach for playtesters who are not used to speak out their mind while playing - with the peer-play method they were forced to discuss strategies, which also helped them understand the objective better and how to solve the puzzle. Unfortunately it also requires unanimous cooperation and a similar level of prior knowledge in order to provide a beneficial and balanced experience for both players, as long as it is applied to games like Machineers, that are designed purely as a single-player game. Possible risks when using this technique include that the less skilled or outgoing player might be dominated by the other one, and that communication might die down when one player takes over control. This problem was luckily not very prominent during our next test phase.

5 Preliminary Test with Target Audience

This chapter concerns the second test phase, which was conducted about two weeks after usability evaluation with a build containing two puzzles and improved usability. The main focus of this test phase was to examine how well the children were immersed into and motivated by the game when looking at dialogues, narrative and graphical style. Since usability issues, as well as game-stopping factors affect immersion and motivation, we also included them within our focus, assuming that our target group might show different preferences than our previous testers.

We contacted the director of Copenhagen International School about a collaboration for our tests early on and were able to acquire 12 testers from our target audience (ages 10 - 14) from an after school activity and the friday computer club.

5.1 Preparation

At the first attempt we were working with four children, two boys and two girls around the age of 10, who stayed at school for a regular after school activity. We introduced ourselves and our project properly and stated clearly that the goal of this test was to improve the game, not to evaluate how well they were playing it. Due to technical problems we could only use one computer to test the game using the peer-play method with the two boys while taking notes. One of the girls had to leave before they were finished, so we attempted to playtest the game with the remaining girl using the Think-Aloud protocol, but we noticed that our presence and intense observation were making her uncomfortable which affected the play experience negatively.

We arranged for another test session during Friday computer club where we could test with another 9 players, this time exclusively boys. We screen-captured the session so we would not miss any information. Again we introduced ourselves as Master students from ITU Copenhagen and asked them to help us to "*make learning fun*" by testing an educational game for us. Openly admitting to the purpose of the game led to an interesting reaction of one of the boys, who kept complaining about the graphics and setting of the game ("*Why is this robots? Why?*", "*This game should* not *have those type of graphics.*"), and expressed he felt tricked because "*learning will never be fun*". His behaviour fortunately did not seem to affect the other players.

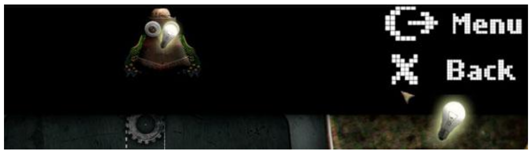
5.2 Navigation

During this test phase we observed once again that players had trouble navigating in adventure mode as well as in puzzle mode. Our approach did not turn out to be very intuitive to use, so we decided to exchange this control scheme in adventure mode by implementing a path for the main character that could be clicked. In the puzzle mode we decided to replace panning with navigation per right-click and dragging, as some of the testers had intuitively attempted.

Also, the players did not realize how to zoom in and out during puzzle mode. When the puzzle started, we had zoomed into the puzzle in order not to overwhelm the player with too much visual information. Once they had examined all the single parts up close, they would need to zoom out and try to grasp the whole picture of the puzzle in order to understand how the different parts were connected. We decided to solve this issue by including a little initial tutorial, inviting the player to exercise the basic controls before entering the puzzle.

5.3 Interface

The problem with navigation also influenced the perception of the *top bar*: A small part of it would be visible on the top of the screen, and to expand and collapse it again the player would need to click it. While moving the mouse towards the screen, the panning navigation would be activated. We solved this issue by reversing the controls as mentioned above, and expanding the top bar when the mouse hovers over it. During the play session it turned out that most players did not even notice the top bar even though we used a light bulb icon on it to attract the players attention. Our solution was to add black and yellow, so called ‘hazard stripes’, to make it more noticeable, as well as animate the lightbulb to appear blinking. Since we changed the controls for navigation, we decided to expand the top bar on mouse hover instead of left-click to simplify its access.



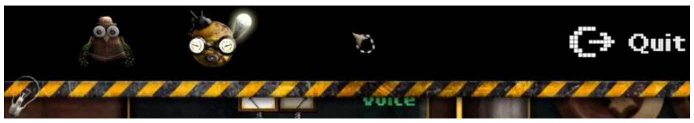


Image 5.1 and 5.2 - Top bar - early and later version

Since many players had trouble using the eraser tool, we decided to exchange its icon and increase the size of the colliders on belts in order to alleviate the usability.

Image 5.3 and 5.4 - Eraser mouse icon - early and later version

5.4 Dialogues and Rewards

With the initial instructive dialogues the players were well informed of what they had to do, but some complained afterwards that they did not have the feeling that they achieved anything and were generally lacking a proper reward for their efforts and achievements. We agreed to add a rewarding dialogue after solving each puzzle, displaying respect and gratitude from the person that was helped, as well as generally improving all other dialogues gradually, the more puzzles are solved until Hayden promotes Zola to be employed as a proper Machineer.

5.5 Interaction with Puzzle Elements

Players had trouble identifying which objects could be interacted with. To solve this we replaced the mouse cursor from an arrow to a wrench in adventure mode, and highlighted elements in puzzle mode, whenever the mouse hovers over interactable objects or characters.

One element that players had difficulties understanding was the gear on the rail, as it moved relatively slowly due to a programming restriction and did not necessarily catch the players attention fast enough. We decided to emphasize it by adding visual effects resembling sparks and adding an appropriate sound effect.

Each puzzle contained partly broken as well as fully functional parts, which were initially set to the correct state and provided a visual reference to guide the player. We observed during this session that sometimes when players were uncertain what to do to solve the puzzle, they started to manipulate all the puzzle elements in a way that made it harder or impossible for them to solve the puzzle. To protect them from making their challenge more difficult we decided to lock all the puzzle elements that were initially set so the user could not delete or manipulate the original puzzle elements.

5.6 Conclusion

Some players took a long time to realize that they were looking at the insides of the owl during the owl puzzle, so it seemed important to add another image of the closed body of the owl on top, which the player would open by clicking on it. This should also help keeping the player immersed during the transition from adventure mode to puzzle mode. Generally players seemed to be confused about the purpose of the different machine parts and invisible parts that were activated by levers, so we agreed to label all machines and levers accordingly, as well as surround machine parts with a yellow frame to emphasize them visually.

None of the players got completely stuck in any of the puzzles, but this might have been because our presence prevented the situation. After discussing the chance of this happening and possible ways to avoid it, we decided to implement the blueprint scaffold. As described in the design chapter, the blueprint provides a complete solution for each puzzle after a certain amount of time has passed, giving the players a chance to solve the puzzle without this extensive help. Not being able to solve the puzzle at all would detach the players from their immersion and interrupt the feeling of *flow*. This scaffold is not to be compared with the option to skip the puzzle entirely - performing the correct actions without fully understanding them can still have an impact on the players learning experience the same way as memorization is used in school settings. If the actions are repeated in different contexts with different stimuli and explanations, we can assume that the learner will eventually grasp the meaning of the action.

The most important finding of this test, as we found out, is that conducting a pilot test is not enough, but we as testers needed to have a test run as well. Inviting the children to help us ‘make learning fun’ was very well intended, but gave away the ‘stealth’ of our ‘stealth learning’ concept and provoked at least one player’s strong resistance. Furthermore, all of us were physically too present, standing right behind the players, observing their behaviour closely and helping out with problems too soon in some cases. We noted that for the final test in order to produce valid data we would have to let the testers play in as much of an uninfluenced manner as possible.

6 Final Test with Target Audience

Our final testing session took place once again at Copenhagen International School during their summer school program. With the help of the director we were able to test the game with 21 children aged from 9 to 14 years, 12 boys and 9 girls. The children played the final version of Machineers, with changes implemented from the two previous test phases and including all five puzzles. The main focus of this test phase was to analyze learning process of the participants and also to determine whether they had fun, were motivated and immersed while playing.

6.1 Preparation

The game was installed on all of the computers in the schools computer lab, and the computers were equipped with headphones in order to enable players to play individually with minimal outside interference. We had prepared an online questionnaire for the participants to fill out after playing the game, in which the children had to answer a few personal questions and were asked to solve some logic problems relating to the game. Using a timestamp and computer identification number we were able to link the information of the questionnaire with the metrics data. Before the test started, we introduced ourselves shortly, instructed the children not to talk to and help each other during the playtest, and raise their hand once they were finished so we could direct them to the questionnaire. We did not give away the learning purpose of the game.

Before actually carrying out the test we conducted a pilot-test with a 12 year-old boy to make sure the questionnaire was understandable, the programming test was not too complicated, and that there were no major bugs left in the game. The pilot test worked out better than expected; our tester solved all the puzzles without even once consulting the blueprints, and fixed some of the machines without testing them. He enjoyed playing the game very much, and did not criticize any aspect of it, but we have to consider the fact the he plays a lot of computer games in his spare time and might be more advanced than the majority of the children in our target group.

The different methods we used and their results will be described in detail in the next sections.

6.2 Metrics

We implemented a metric data collection system that would log some of the player's actions in a file while they were playing. With this system we were able to measure how long the testers took to complete the different puzzles, and how many times they accessed hints and blueprints. We used the data to get an overview of how differently player would use scaffolds and if they catered to their needs in the way we designed them. The outcome is discussed in section 6.5 Results.

6.3 Questionnaire

The final dialogues in the game instructed the playtesters to raise their hand so we could direct them to the online questionnaire, which had been created using Google forms.

We asked the players about general personal information: age, gender, and how often they played computer games and analog games (board or card games) to gather some general information about them which might have an effect on how they perceive the game.

Furthermore we inquired about how much the test subjects enjoyed playing Machineers, whether they thought it was fun, too challenging or too easy, and if and under which circumstances they would like to keep playing. We also asked them which puzzle they thought was most fun/most challenging and which help options our players found helpful, used often or if they were even aware of them.

We encouraged the children to state aspects they did not like about the game and asked them to mark any puzzle element they were confused about or did not exactly know how to use, as well as state any usability problems they had with the top bar. Further questions were concerned with the programming test and will be specified in the section below.

6.4 Programming Test

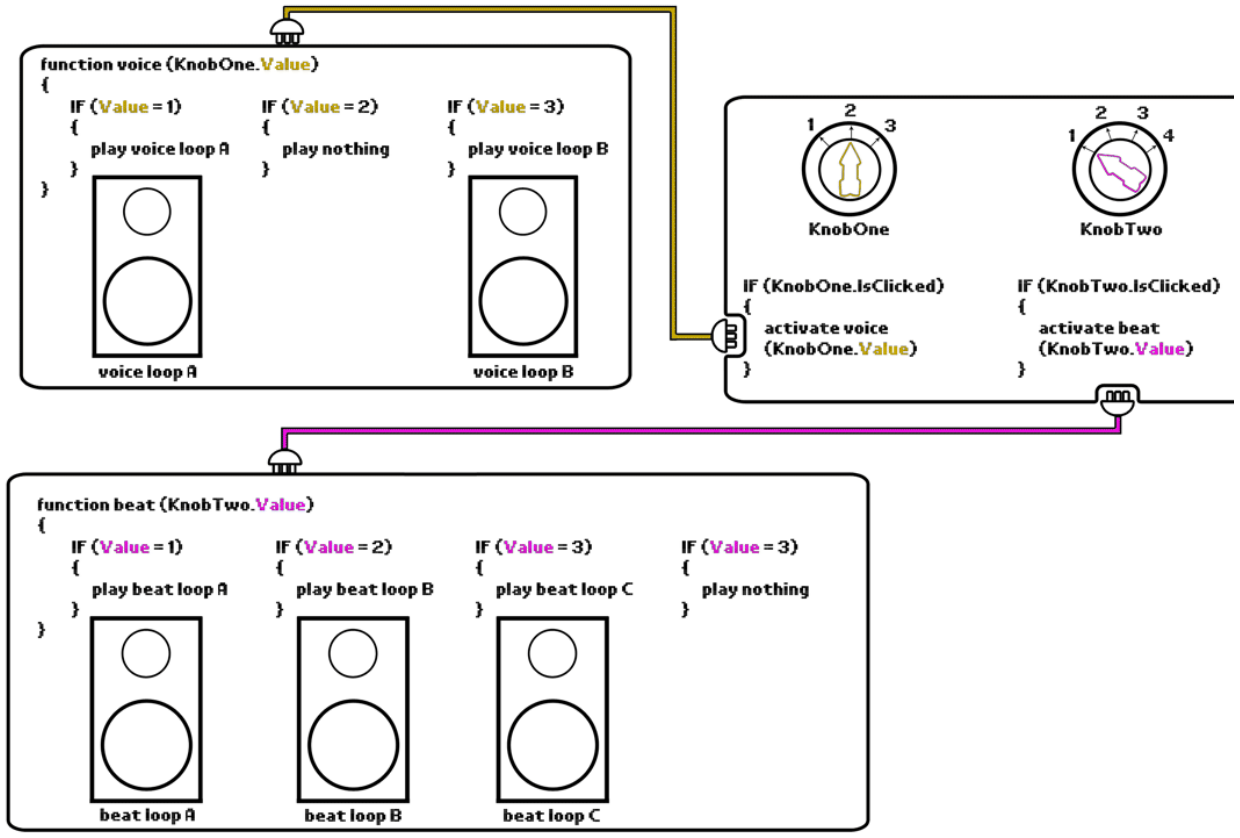
The programming tests we presented to the children consisted of a number of pictures showing machines similar to the ones they had encountered within the game, as well as pseudocode examples. The children were then prompted to link the pictures to the correct piece of pseudocode or the other way around.

The aim of this test was to determine whether the children would be able to understand and solve logical problems that were thematically connected to the game they had just played, but presented differently, therefore analyzing the *transfer value* of their problem-solving skills. Instead of using the visual representations from the game we presented them with a pseudocode text version of those same programming concepts.

> *a 'transfer' of ideas from one context to another that is elusive, rare, and powerful. It happened not because the students learned more information, but because she learned it in the context of a new way of thinking - an epistemic frame - that let her see the world in a new way. (Shaffer et. al 2005, p.15)*

Since children of this age group are usually not familiar with pseudocode, and because we did not want to provoke a hostile attitude towards our game, we decided against presenting the children with a pretest similar to this posttest. This makes us unable to show exactly how much the game improved the children's ability to interpret pseudocode.

We attempted to create a smooth transition from the game to the test by taking a puzzle most players enjoyed and understood well in the previous test phases, the DJ machine - simplifying it by cutting it in half and translating it partly to pseudocode. The code snippets were placed exactly where the puzzle elements had been, but there was no visual representation of the DJ machine to view since the game was closed, so they had to recall that visual from their memory.



```
function voice (KnobOne.Value)
{
    IF (Value = 1)
    {
        play voice loop A
    }
    IF (Value = 2)
    {
        play nothing
    }
    IF (Value = 3)
    {
        play voice loop B
    }
}
```

```
IF (KnobOne.IsClicked)
{
    activate voice
    (KnobOne.Value)
}
IF (KnobTwo.IsClicked)
{
    activate beat
    (KnobTwo.Value)
}
```

```
function beat (KnobTwo.Value)
{
    IF (Value = 1)
    {
        play beat loop A
    }
    IF (Value = 2)
    {
        play beat loop B
    }
    IF (Value = 3)
    {
        play beat loop C
    }
    IF (Value = 3)
    {
        play nothing
    }
}
```

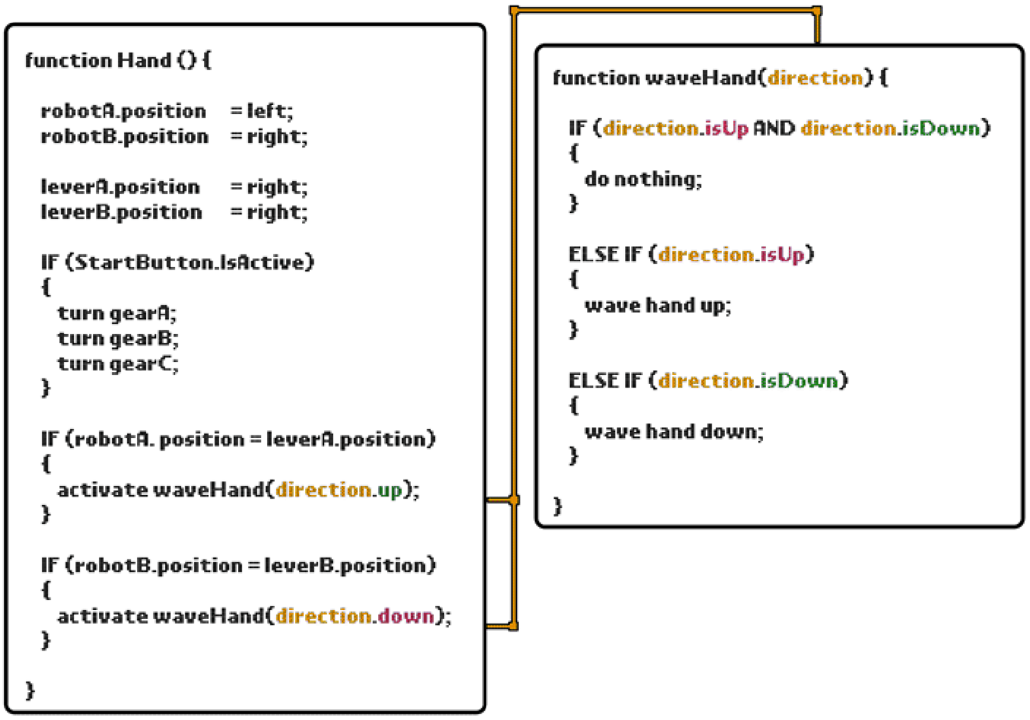
Image 6.1 - Programming test 01 - DJ machine

The question related to image 6.1 is as follows: “Look at the knobs and try to figure out what kind of music is playing at the moment.”

The answer options are 1A “voice loop A and beat loop A”, 1B “no voice loop and no beat loop (no music at all)”, 1C “no voice loop and beat loop A”, 1D “voice loop A and no beat loop” we also included an “I don’t know” option to make sure that tester would not randomly select an answer and falsify the test results. The correct answer is 1c), “no voice loop and beat loop A”.

The second question involves a machine that the players had not interacted with during the game, the *waving clown* machine. In this question we presented a complete pseudocode version of the machine (image 6.2) and asking the testers to select the matching visual representation from a choice of pictures (image 6.3).

“Which of the images is an exact match of the text version of the Waving Clown machine?” The answer options are 2(A), 2(B), 2(C), 2(D) and “I don’t know”. The correct answer is 2(B).



```
function Hand () {

  robotA.position   = left;
  robotB.position   = right;

  leverA.position   = right;
  leverB.position   = right;

  IF (StartButton.IsActive)
  {
    turn gearA;
    turn gearB;
    turn gearC;
  }

  IF (robotA. position = leverA.position)
  {
    activate waveHand(direction.up);
  }

  IF (robotB.position = leverB.position)
  {
    activate waveHand(direction.down);
  }

}
```

```
function waveHand(direction) {

  IF (direction.isUp AND direction.isDown)
  {
    do nothing;
  }

  ELSE IF (direction.isUp)
  {
    wave hand up;
  }

  ELSE IF (direction.isDown)
  {
    wave hand down;
  }

}
```

Image 6.2 - Programming Test 02 - Waving Clown machine - pseudocode

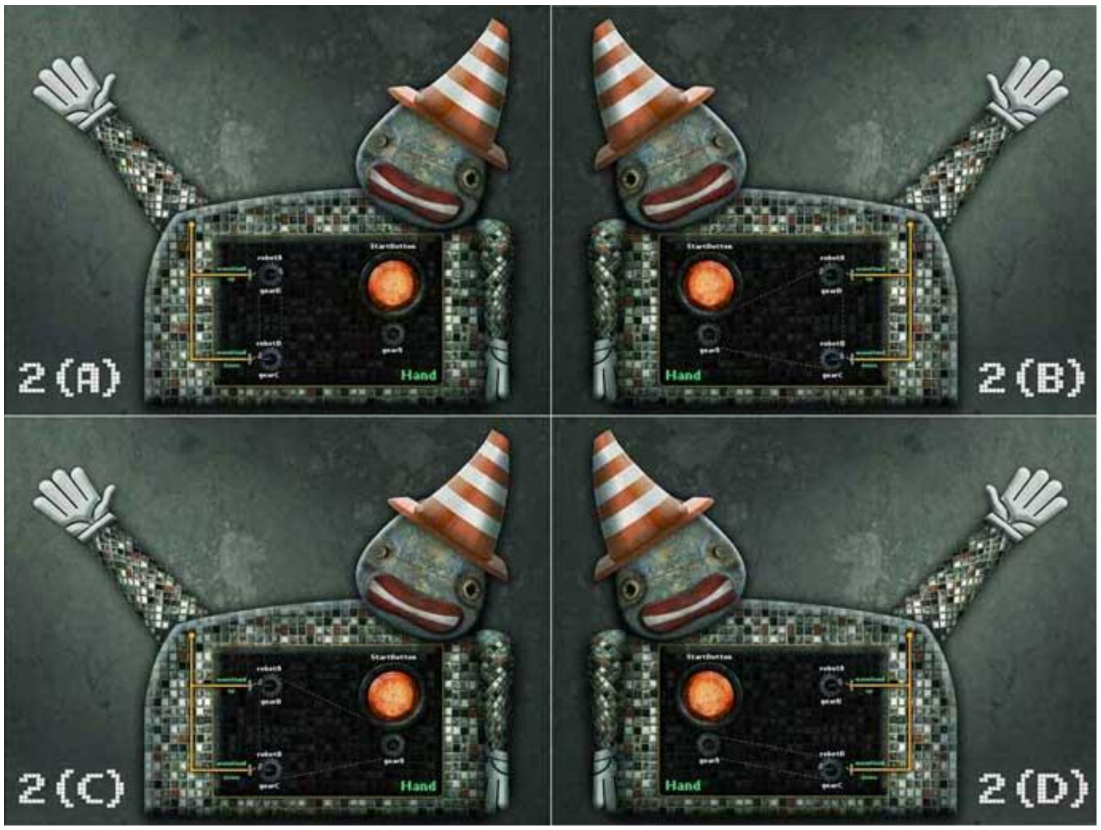


Image 6.3 - Programming Test 02 - Waving Clown machine - answer options

The third question involved a more complex machine, composed of similar elements as the crane machine, but simplified to some degree, the *hot dog* machine. For this part of the test the children were provided with three images: one showing the hot dog machine in the visual representation as it would have been used within the game (image 6.4), another image showing the simplified pseudocode representation of the same machine but partly broken (image 6.5), and a third image with a selection of pseudocode pieces that could replace the broken part in order to exactly resemble the first image (image 6.6).

The question is "*Which piece of the text fits into the text version of the hot dog machine?*' and the answer options are Text 3(A), 3(B), 3(C), 3(D) and "*I don't know*". The correct answer is 3(D).

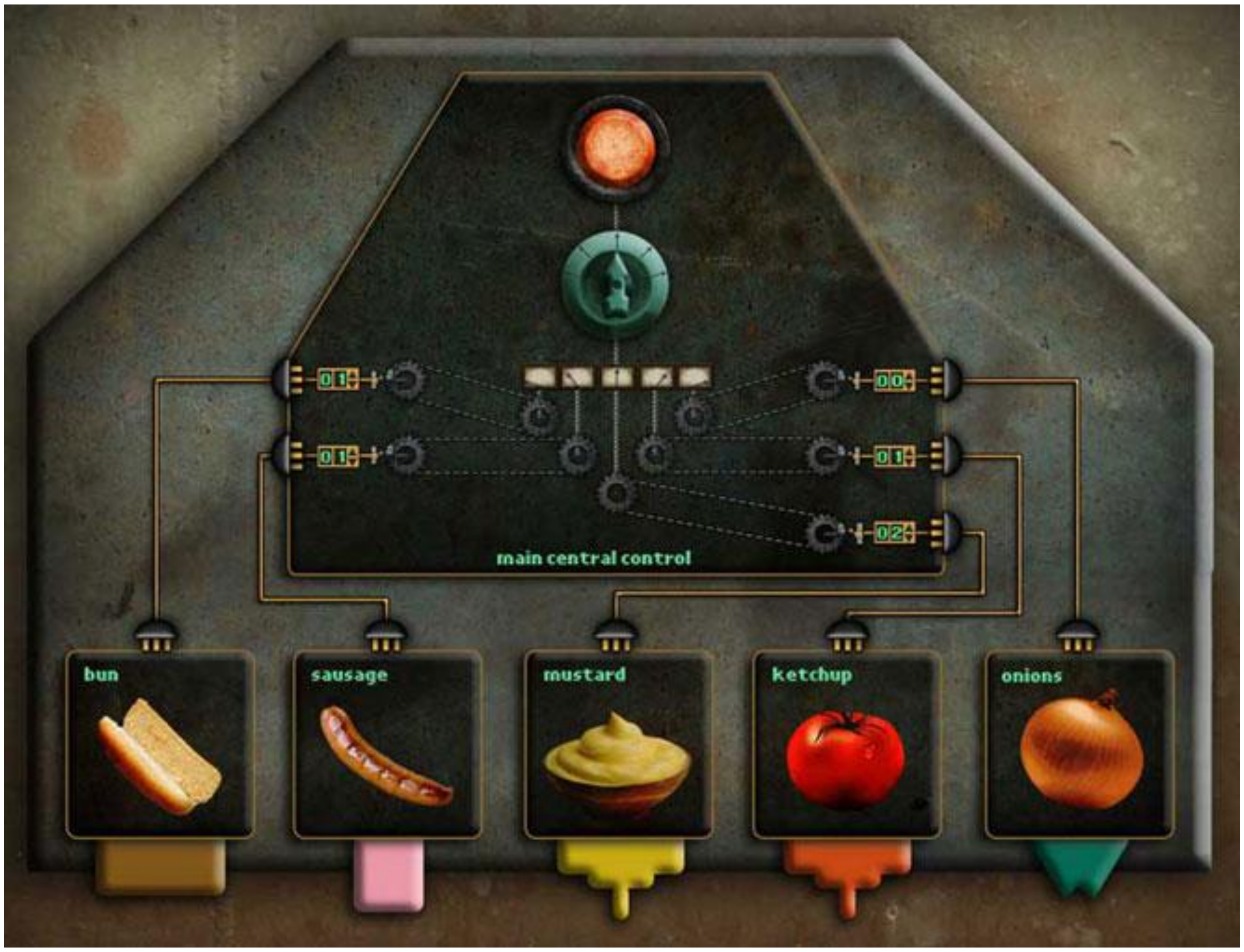
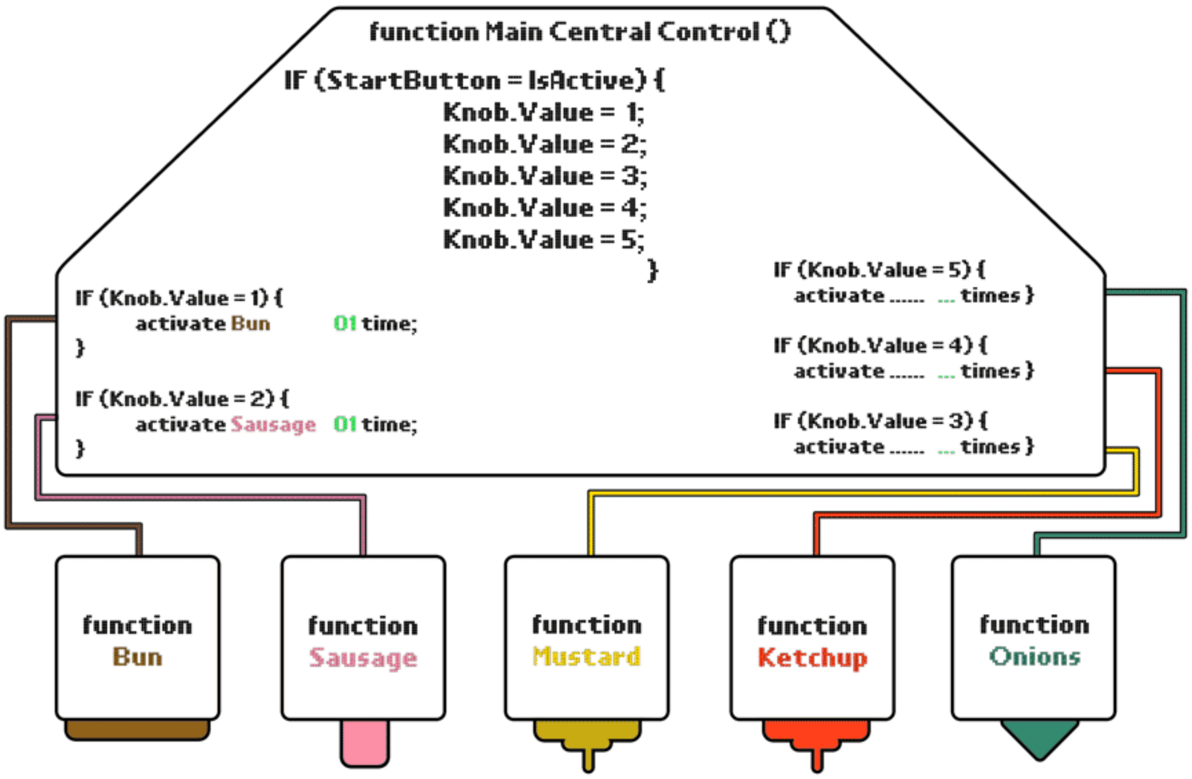


Image 6.4 - Programming Test 03 - Hot Dog machine - textured version

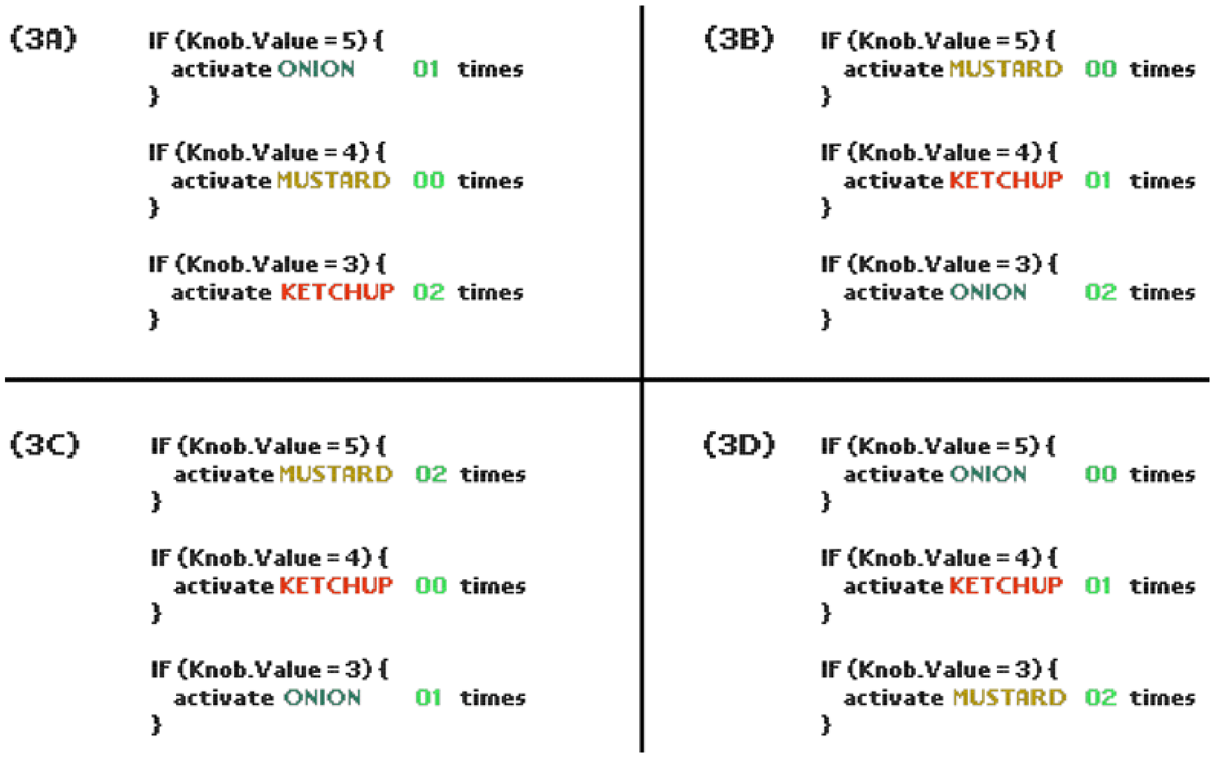


```
function Main Central Control ()
IF (StartButton = IsActive) {
            Knob.Value = 1;
            Knob.Value = 2;
            Knob.Value = 3;
            Knob.Value = 4;
            Knob.Value = 5;
}

IF (Knob.Value = 1) {
    activate Bun        01 time;
}
IF (Knob.Value = 2) {
    activate Sausage    01 time;
}

IF (Knob.Value = 5) {
    activate ...... ... times }
IF (Knob.Value = 4) {
    activate ...... ... times }
IF (Knob.Value = 3) {
    activate ...... ... times }
```

Image 6.5 - Programming Test 03 - Hot Dog machine - pseudocode



```
(3A)  IF (Knob.Value = 5) {
        activate ONION     01  times
      }
      IF (Knob.Value = 4) {
        activate MUSTARD   00  times
      }
      IF (Knob.Value = 3) {
        activate KETCHUP   02  times
      }

(3B)  IF (Knob.Value = 5) {
        activate MUSTARD   00  times
      }
      IF (Knob.Value = 4) {
        activate KETCHUP   01  times
      }
      IF (Knob.Value = 3) {
        activate ONION     02  times
      }

(3C)  IF (Knob.Value = 5) {
        activate MUSTARD   02  times
      }
      IF (Knob.Value = 4) {
        activate KETCHUP   00  times
      }
      IF (Knob.Value = 3) {
        activate ONION     01  times
      }

(3D)  IF (Knob.Value = 5) {
        activate ONION     00  times
      }
      IF (Knob.Value = 4) {
        activate KETCHUP   01  times
      }
      IF (Knob.Value = 3) {
        activate MUSTARD   02  times
      }
```

Image 6.6 - Programming Test 03 - Hot Dog machine - pseudocode answer options

As a final part of the test and the questionnaire we asked the children answer the open-ended task: *"Pick the picture of your favorite machine and explain how it works in as much detail as possib*le". Unfortunately most children strongly resisted completing this task or possibly misunderstood it, as many just stated the name of their favorite machine ("*DJ machine*") or typed in random letters. We were able to assist some testers in answering this question producing quite good results, but because this process was very time-consuming and our resources limited, it was not possible to collect enough data this way for a proper analysis.

6.5 Results

Relying on our experience from the immersion test phase, where all children had a excellent english literacy whether they were native english speakers or not, we assumed that the attendees of the summer school would have equally good english reading skills, which our game Machineers requires. As it turned out, the children attending summer school represented a broader variety of age, gender and nationality than our previous testers, but unfortunately their english proficiency was considerably lower.

This meant that many of the children had problems understanding the text based information which our game relies on to a large extent. Surprisingly this fact did not imply that they were unable to play the game; instead they had to be supported with more explanations from us, but performed unexpectedly well regardless.

6.5.1 General results

Out of 21 participants, 15 completed the game within the time we had available for the test, with an average completion time of 65 minutes and 14 seconds.

Number of participants	**21**
Completion rate	15 (71,43%)
Average completion time:	65 min 14 sec

Table 6.1 - Average completion time of all players

The fact that our game was capable of capturing the attention of the children for such a long time is remarkable and points towards a deep immersion of the players. During this time the majority of players were deeply engaged with the game and only a few of our testers showed signs of boredom, being overwhelmed and the desire to quit.

We had 12 male and 9 female testers in the ages 9 to 14. The male testers showed a better completion rate than the females with 10 out of 12 boys (83%) completing the game against only 5 out of 9 girls (55%) doing the same.

	boys	**girls**
Participants:	12	9
Completion ratio:	10 (83,33%)	5 (55,56%)

Table 6.2 - Average completion ratio divided by gender

When looking at the completion rate divided on the different ages we can see a slight increase for the older testers, but for unknown reasons none of the three 14 year old testers were able to complete the game. The participants aged 11-13 did particularly well with only 1 person unable to complete in time, and their average completion time was much lower than 9-10 year olds.

Ages:	**Participants:**	**Completion rate:**	**Average completion time:**
9 or younger	2 (9,52%)	1 (50,00%)	72 mins 55 secs
10	3 (14,29%)	2 (66,67%)	77 mins 49 secs
11	2 (9,52%)	2 (100,00%)	66 mins 29 secs
12	5 (23,81%)	4 (80,00%)	60 mins 57 secs
13	6 (28,57%)	6 (100,00%)	60 mins 48 secs
14 or older	3 (14,29%)	0 (0,00%)	-

Table 6.3 - Average completion time ratio divided by age

With the metric data we collected, we were able to calculate the average completion times of the different puzzles along with the amount of hints and blueprints used.

Puzzle:	**Completion rate**	**Completion time (average)**	**# of hints in puzzle**	**Hints used (average)**	**Blueprint use**
Own Puzzle:	100,00%	9 min 37 sec	8	5,57	23,81%
DJ Puzzle:	100,00%	7 min 25 sec	6	3,95	52,38%
Bar Puzzle:	95,24%	9 min 32 sec	6	5,05	65,00%
Cart Puzzle:	90,48%	9 min 16 sec	8	6,89	47,37%
Crane Puzzle:	71,43%	17 min 4 sec	6	4,13	86,67%

Table 6.4 - Average completion time and hint/blueprint access per puzzle

The average completion time for the owl puzzle seems too long considering it is the first puzzle in the game. Due to the language barrier, many of the children did not understand that they could zoom and scroll inside the puzzle, and this may have further increased the completion time of the puzzle.

Gaming habits, both digital and analog, were one of the aspects we inquired about in the questionnaire, and the male testers answered that they were playing video games more often than the female testers. More frequent video game habits showed a higher completion rate, but surprisingly, of the children who completed the game, the ones who answered that they played video games only a few times a month had the lowest average completion time.

Video Game Habits	**Answers**	**Completion rate**	**Boys**	**Girls**	**Completion time (average)**
never	3	2 (66,67%)	1	2	74 min 24 sec
a few times a month	6	3 (50,00%)	1	5	56 min 32 sec
a few times per week	11	9 (81,82%)	9	2	65 min 52 sec
everyday	1	1 (100,00%)	1	0	58 min 59 sec

Table 6.5 - Video Game Habits in relation to gender and completion time

On the other hand, analog gaming habits did not seem to have any effect on completion time and since only one of our testers was playing analog games on a daily basis we cannot make assumptions from the completion rate statistics of that habit group. Both completion rate and time was very similar over the different analog gaming habits and there was an even distribution across genders.

Analog Game Habits	**Answers**	**Completion rate**	**Boys**	**Girls**	**Completion time (average)**
never	3	2 (66,67%)	1	2	63 min 29 sec
a few times a month	10	7 (70,00%)	5	5	65 min 51 sec
a few times per week	7	5 (71,43%)	5	2	64 min 2 sec
everyday	1	1 (100,00%)	1	0	62 min 6 sec

Table 6.6 - Analog Gaming Habits in relation to gender and completion time

6.5.2 Immersion-Related Results

In the following we inquired about how much the test subjects enjoyed playing Machineers, whether they thought it was fun, too challenging or too easy, and if and under which circumstances they would like to keep playing (note that participants were able to choose more than one answer).

Answer:	**Answers**	**% of testers**
I would like to keep playing and solve more puzzles	11	52,38%
I would like to build my own machines	3	14,29%
I would ask my parents to get this game for me	5	23,81%
I would not play this game at home	4	19,05%
I would enjoy playing this game in school	8	38,10%

Table 6.7 - Questionnaire answers to the question "Keep on playing?"

Just over half of our players answered that they would like to keep playing, and this is after they had already been playing for one hour. This shows that the game was fun and engaging for many of the children and the general attitude towards the game was good. 4 of the 21 test subjects answered that they would not play the game at home which we consider to be quite a low number taking into account the diversity of the testers.

Answer:	**# of answers**	**% of testers**
It was very tricky and challenging	10	47,62%
It was ok.	9	42,86%
It was a lot of fun	8	38,10%
It was too hard for me	2	9,52%
It was a little boring	1	4,76%
I did not like it at all	0	0,00%

Table 6.8 - Questionnaire answers to the question

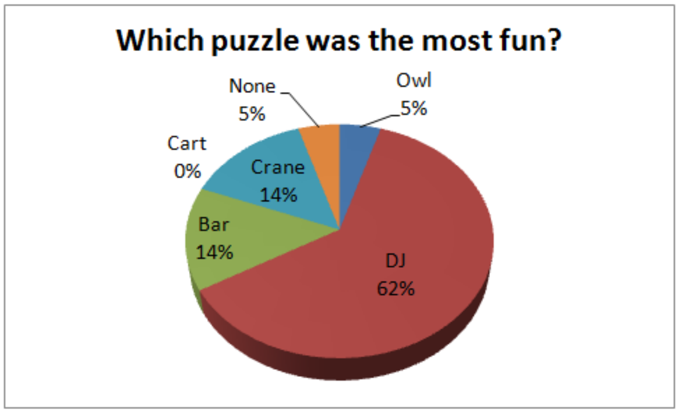
"How would you describe how you felt when playing this game?"

In response to the question how they felt when playing Machineers, most testers answered positively. This question provided a list of answers were the testers could choose as many as they liked. Almost half of the players described the games as being tricky and challenging and only a few found it too hard or boring. None of the children showed complete dislike of the game. The answer "*It was very tricky and challenging*" can be viewed as both positive and negative depending on the tester.

The fact that out of 10 players who selected *"it was very tricky and challenging*" five also selected "*it was a lot of fun*", indicates that they appreciated the challenge. This is oriented towards the 'hard fun' concept expressed by Lazzaro (2004). Players appreciating this kind of gameplay thrive under compelling challenges that allow for multiple strategies and the expression of skill and mastery. "*The challenge focuses attention and rewards progress to create emotions such as Frustration and Fiero (an Italian word for personal triumph)*" (Lazzaro 2004 p.3). The body postures of the children during the play session ranged from holding their fingers crossed, over raised shoulders, to hands covering mouth or eyes and after solving the puzzles almost all the children had both arms raised over their heads as a sign of victory.

Image 6.7 - Children solving the crane puzzle

In the following we asked in more detail about which puzzle they thought was most fun and which one was most challenging. Almost two thirds of the children thought the DJ machine was the most fun, and as shown in Table 6.4, this was also the puzzle that the children solved the fastest. It should also be noted that the DJ puzzle had a nice audio feedback compared to the other puzzles, and that this might have made it more memorable to the children. Only a single tester answered that none of the puzzles were fun.

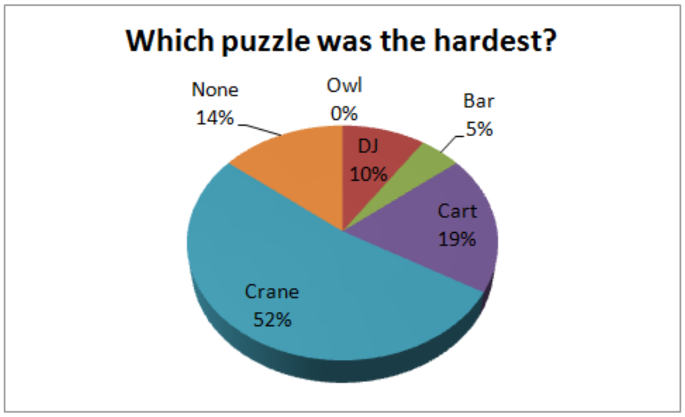


Graph 6.1 - Pie Chart showing percentage of most fun puzzle

We were surprised that none of the testers chose the cart puzzle as the most fun, since Table 6.4 shows a decline in blueprint use, which could have indicated a greater sense of accomplishment. We suppose that because the visual feedback in the cart puzzle was not as exciting as in the other puzzles, it may have been less memorable to the testers.

The hardest puzzle was the Crane puzzle, which the children spent an average of 17 minutes 4 seconds completing. It should be noted however, that solving the puzzle relied very much on trial and error, and that the crane spent plenty of time moving back and forth while the children could only observe if the crane performed correctly. This must have added to the completion time of the puzzle. From a design perspective, this puzzle and the Bar puzzle are the most trial and error oriented: the complexity of the structure of those two puzzles forces players to approach them one step at a time. Based on our observation of test subjects, players tend to focus on single methods, then proceed to attempt synchronization between methods. This behavior is exactly what we had in mind when designing those puzzles and we feel that a this approach fits well and resembles how many approach problem solving and programming.

Time is not the only factor that indicates the Crane as the most complex puzzle: in fact, this is where we can see the highest percentage of blueprint use, which indicates that the children did in fact have a hard time solving it. The Crane puzzle is wider in terms of the number of actions needed to solve it, and also introducing the most complex component of the game, the programmable knob.



Graph 6.2 - "Which puzzle was the hardest and most annoying?"

Three testers answered that none of the puzzles were hard, but interestingly two of these testers did not manage to complete the game in time which puts the seriousness of their answers in question. No one answered that the owl puzzle was the hardest which is unexpected considering it had the second highest average completion time of all five puzzles. Testers may have hesitated choosing the owl because it was the first puzzle and they felt that they should not have struggled with it.

6.5.3 Usability and Scaffolds

We encouraged our testers to state aspects they did not like about the game. This was also a multiple-answers, multiple-choice question.

Answer:	**# of answers**	**% of testers**
there was too much text	6	28,57%
the puzzles were too hard	4	19,05%
I did not like that I was playing a girl robot	3	14,29%
there was not enough help	2	9,52%
I did not like the robots	1	4,76%
It was too boring	1	4,76%
Nothing (no option selected)	7	33,33%

Table 6.9 - "Things I did not like about the game"

The most chosen reply was "*There was too much text*". 6 testers or ~28% chose this answer which may point back to the aforementioned lack of fluency in english. We also offered them the option to state other observations, which was only used by one tester, stating: "*I LIKE ALL THE PUZZLE*" (sic).

We further inquired which help options our players found helpful, used often or if they were even aware of them. This question allowed only for a single answer.

Answer:	**# of answers**	**% of testers**
The best help was the blueprints	12	57,14%
The best help was the tips and hints from the robot Bent	4	19,05%
The best help was the owl explaining things to me	3	14,29%
The best help was when information just appeared when I clicked on things	2	9,52%

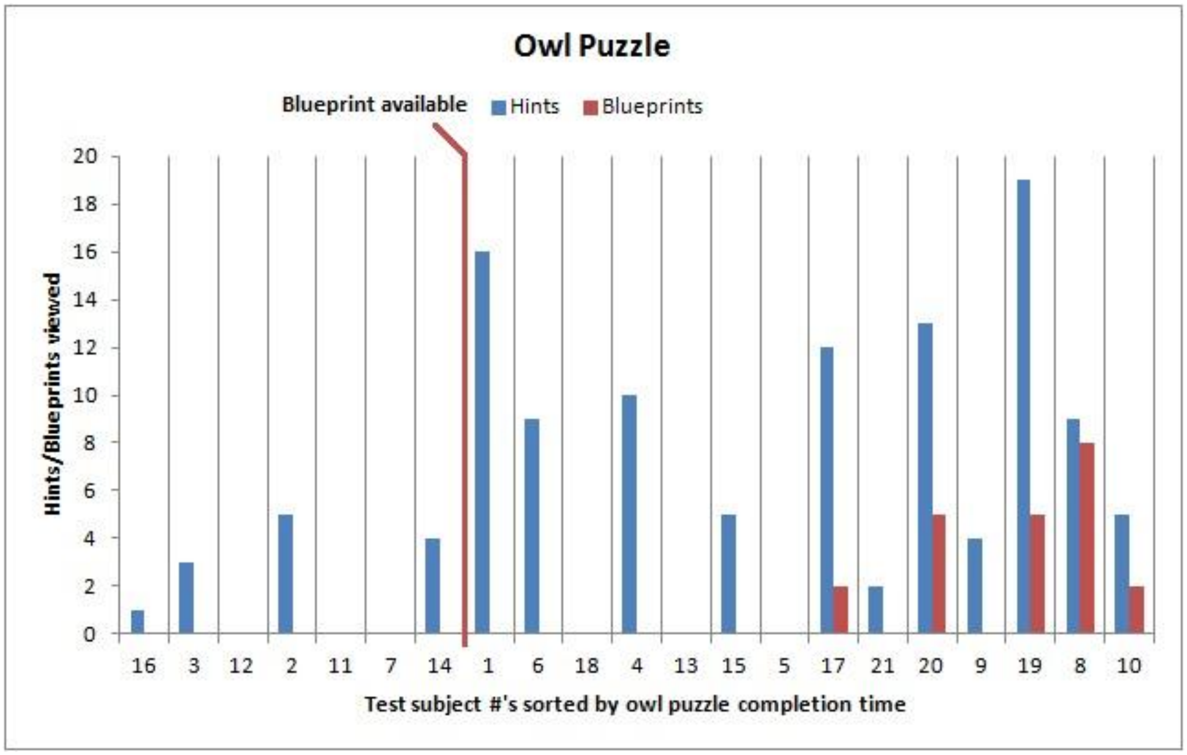
Table 6.10 - "What help option worked best for you?"

Our metrics data shows that the owl scaffold has been used much less and perceived as less helpful than other help functions, which can be caused by several reasons. Answers to the questionnaire showed that many players were simply not aware of this help option, others did not find it very useful since we had written many explanations for different puzzle elements, but not implemented all of them. Having to read a lot of text that contains general information but no specific hints towards solving puzzles is clearly not very appealing to our target audience.

Three players indicating that "*the best help was the owl*" is an unexpected result, given the virtually nonexistent usage of the on-demand owl help emerged from the metrics. The question might have been misunderstood: perhaps players were thinking of the owl as a character here, rather than a help option. More than half the testers found the blueprint to be the best help feature which is not surprising. The blueprint consisted of a picture of the machine in its solved form and therefore the language problems would have had no influence on this scaffold in contrast to the other help methods.

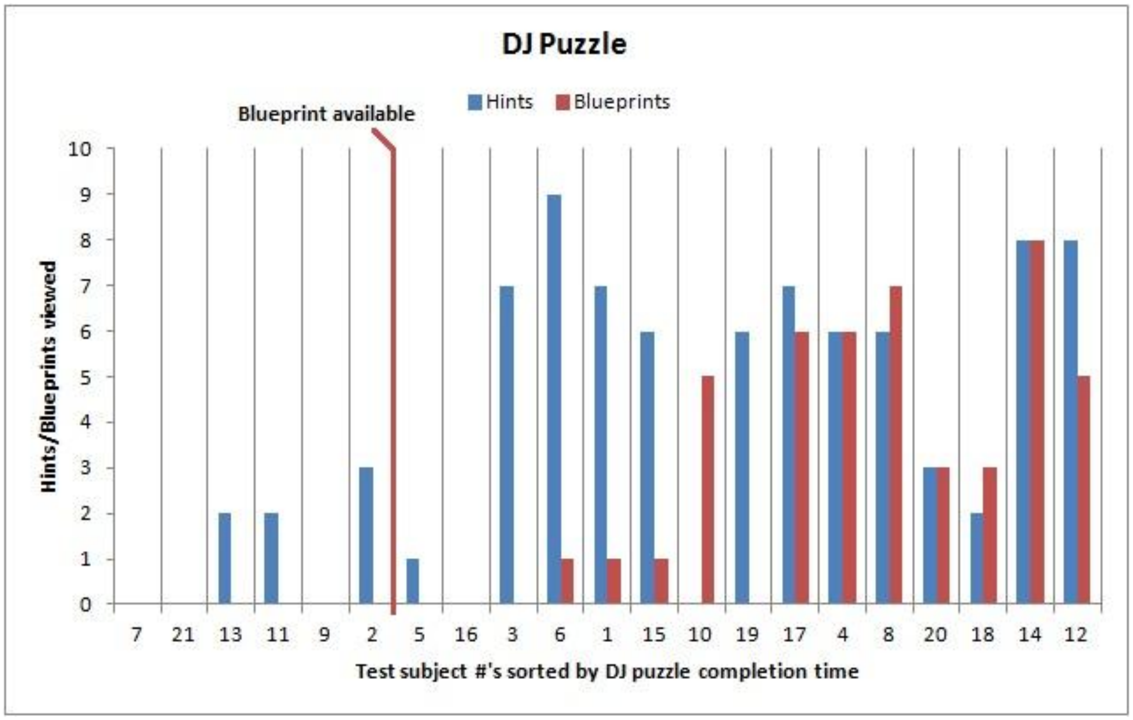
To evaluate our game externally we asked a number of professionals in the field to watch a commented walkthrough (which can be found on the DVD) and received the following feedback, confirming good usability of the game: "*The design is simple, but effective and consistent. I had no problems while navigating through the game. I encountered no usability issues or game stoppers.*" (Thomas Wernbacher, personal communication, August 14, 2012)

In order to obtain more information about how the different players used scaffolds individually and how their behaviour developed over the course of the game, we crossed individual hint and blueprint access data for each puzzle and all players. Below is a graph for each puzzle showing how many hints and blueprints the individual test subjects used. The graphs are sorted by the test subject's completion time of the puzzle and a red line shows the point in time where the blueprint became available.



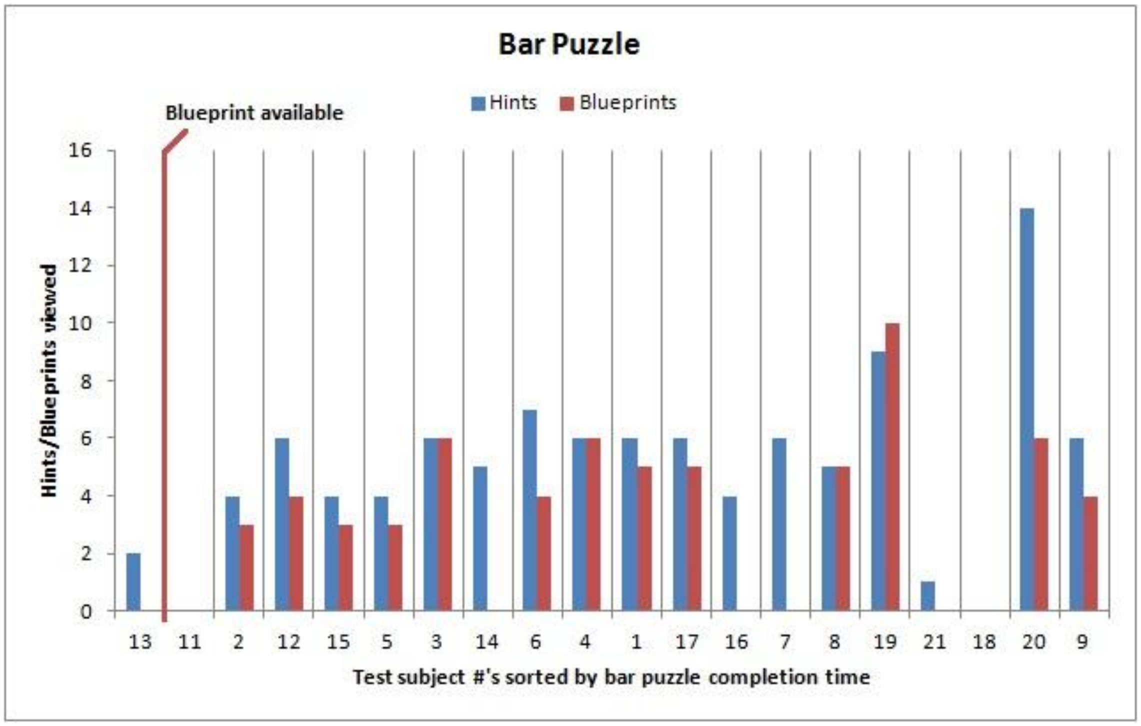
Graph 6.3 - Hint and Blueprint use of all players in the Owl puzzle

Looking at the owl puzzle we can see that only a few players used the blueprint and many accessed neither hints nor blueprints. Since this was the first puzzle, many of the testers may not have been aware of the existence of the blueprint. It could be argued that the low hints and blueprint use was the result of the players own desire to complete the puzzle without help, but the increasing use of help in the later puzzles (which had a lower average completion time) suggests otherwise.



Graph 6.4 - Hint and Blueprint use of all players in the DJ puzzle

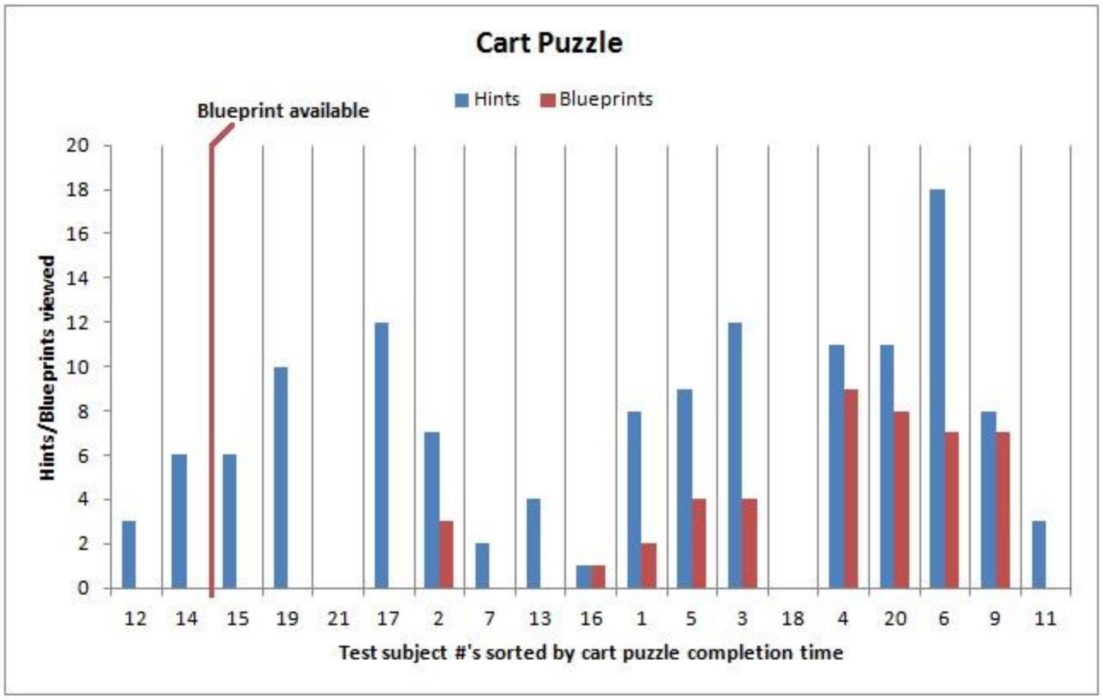
The DJ puzzle shows an increase in blueprint use compared to the owl. The two fastest solvers did not use any hints in contrast to the others but since the hints were made available sequentially after certain time intervals, some subjects may have completed the puzzle before all hints were available. It is interesting to see that test subject #9 and #21, who took a long time to solve the owl, were so fast solving the DJ puzzle. We can see that they both solved the owl without the use of the blueprint and this may have helped them understand the puzzle elements better. Following subject #21 we can see that he does not use the blueprint until the last puzzle which makes it seem like a conscious decision to try and beat the game without help.



Graph 6.5 - Hint and Blueprint use of all players in the Bar puzzle

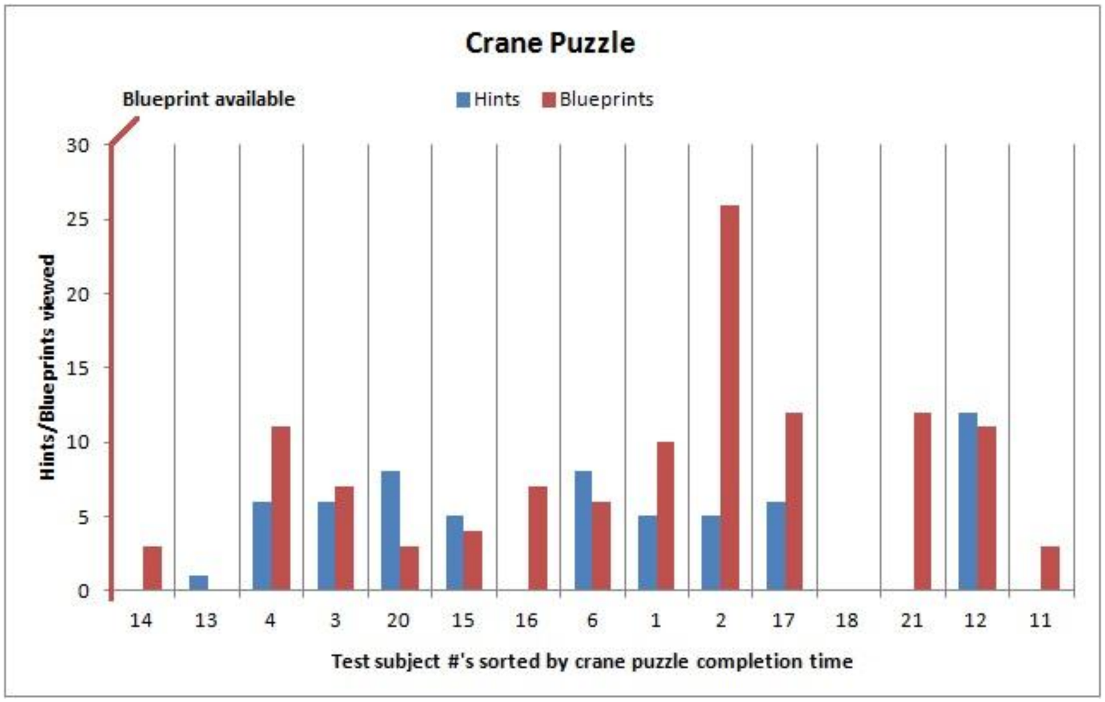
The bar puzzle has the second highest blueprint use after the crane puzzle and all the test subjects who used the blueprint had to consult it several times. This makes sense considering that the puzzle required setting the position of nine different gear robots correctly, and remembering all the settings from just one look at the blueprint would be hard. It also indicates that all testers who used the blueprints failed to solve the logic task this puzzle involved and probably did not fully understand what was being taught. In this case, presenting the blueprint at a later point might give the player more time for another attempt to solve this puzzle using hints and logic.

It is interesting to see that test subject #18 refused to use any help in this and the following two puzzles. Test subject #18 used both hints and blueprint in the DJ puzzle so he must have been aware that help options were available.



Graph 6.6 - Hint and Blueprint use of all players in the Cart puzzle

The use of scaffolds in the cart puzzle was a little surprising. The number of players accessing the blueprint is lower than in the previous puzzle, and the average completion time is also lower. According to these results, the difficulty curve seems to decrease instead of increasing, although this is the second most difficult puzzle by number of votes (if we look at graph 6.2). This can be caused by many factors. The Cart puzzle requires a small number of components to be placed, while the Bar puzzle requires to manipulate a higher number of elements, which could potentially introduce an underestimated factor of complexity. Another possible explanation could be that the hints in the Cart puzzle are clearer, reducing the need for consulting the blueprint. The hints in this puzzle were directly instructing the player on how to fix the engine and this may have inspired some testers to try and solve the puzzle without the blueprint.



Graph 6.7 - Hint and Blueprint use of all players in the Crane puzzle

The Crane puzzle was the one most of our testers chose as the most challenging puzzle. Looking at the graph we can see that almost all testers used the blueprint and subject #2 consulted it 26 times before completing the puzzle. During testing we noticed that some of the players did not understand how the wires in the crane were crossed on the right side in contrast to the left side, even when looking at the blueprint, and this may been the reason for subject #2 consult it so many times.

6.5.4 Programming Test Results

As mentioned earlier, the data from the programming test will have to be analyzed with great cautiousness since the missing pretest does not allow us to make any assumptions regarding learned content and the quality of the test itself. Exactly like anticipated by Honey and Hilton (2011), the informal learning setting our game was tested in and its combination with the language barrier possibly resulted in a strong resistance of players to complete the programming tests, assuming that players who answered “I don’t know” to all test questions were resisting the test. Honey and Hilton’s (2011) suggestion to use the technology available and incorporate the test within the game, was not feasible to implement within the scope of this project. The test clearly showed us that this approach would be significant for playtests in informal learning settings.

	Correct:	**Incorrect:**	**Don't know:**
Picture 1:	14,29%	33,33%	52,38%
Picture 2:	19,05%	4,76%	76,19%
Picture 3:	28,57%	19,05%	52,38%

Table 6.11 - Programming test results

Many of the children had problems understanding the questions, and when it came to writing, even more of them gave up.

6.5.5 Interesting Player Profiles

In this section we have selected a few interesting representatives of our test group to get a general idea of how different testers experienced the game.

Test subject #13

Test subject #13 is interesting since she was the only tester to answer all three programming test questions correctly. At the same time she also had the fastest completion time of all the testers with 48 minutes and 54 seconds. When asked to describe one of the machines from the programming tests in the questionnaire she gave a good description of the hot dog machine: "*Hot Dog machine: you click on the red button, and the knob starts on gear 1 once (gives buns), and then it goes to the next gear once (gives sausages) and then the rest.*" This description shows that she completely understood how the hot dog machine worked and also points towards good english literacy. She thought the best help was the hints from the robot Bent but said she only used the help when she was stuck, which corresponds with what the metric data shows. Her computer gaming habits were "few times a month" so it is unlikely this is the cause of her success.

Test subject #11

Test subject #11 solved the first 3 puzzles very fast but had the slowest completion time in the last two puzzles. She spent 16 minutes in the cart puzzle and almost 25 minutes in the crane, and only used the blueprint in the last. She answers that she only used the help when she was stuck which is backed up by the generally low hint use in the metric data, but as her favorite help feature she choose the owl even though the metrics show she never used it. She may be confusing the owl's on demand help with when the owl sometimes speaks to the player in the beginning of a puzzle.

In the programming test she answered "I don't know" to all three questions and she did not try to explain any of the machines. This indicates to us that she resisted answering the test.

Test subject #21

Test subject #21 is interesting because he is a 9 year old boy who used almost no help but still managed to complete the game. He only used the blueprint in the crane puzzle and in the other puzzles he used almost no hints. His answers in the questionnaire show that this was conscious decision, but since he also said the game had too much text, he might have had trouble reading and understand the hints.

In the programming test he tried to answer question 1 and 3 but got none of them right. He was unable to write his own explanation of one of the machines, but with one of us writing while he explained he managed to show he had a good understanding of the hot dog machine.

6.6 Reflection

Our results show that the low english literacy we encountered during the third test phase did not have a harmful effect on immersion and enjoyment of our test subjects during playtest. However, as most of our scaffolds involved textual information, it is difficult to determine how helpful they were to our players. This triggered the assumption that we could expand our target audience to include younger and less literate children if we reduce the text to a minimum and provide more explanatory pictures as well as audio recordings as a support for textual information. More feedback from a professional, this time an assistant professor for programming, Ilse Schmiedecke, indicated another option of expanding our target group:

> *there is another group of kids i am considering, namely those from so-called 'precarious' homes. this game could be so good for them to train their creativity and logic, but even when they are older, they oftentimes lack the reading skills required. apart from that, the mood and graphics of the game would definitely appeal to them, and i would love to*

> *introduce that to such schools. (Ilse Schmiedecke, personal communication, July 23, 2012)*

We can assume that the amount of puzzles available were not sufficient to establish a circle of repetition and mastery. The initially planned sandbox mode would have provided better grounds for determining the development of skills and enable learners to actively construct knowledge, than the task to verbally describe one of the machines. Implementing the sandbox mode would have been the final step to incorporate the constructivist approach (Egenfeldt-Nielsen 2007) into our game design, but this was unfortunately not feasible within the scope of this production.

From the reactions of the players towards the programming test and the resulting data from the questionnaire, we can assume that the informal learning context that we tested our game in, resulted in a strong resistance towards the test, as predicted by Honey & Hilton (2011). Due to an inadequate introduction of testers to the programming test, an insufficient level of assistance from us during the task, and the absence of a pretest, no meaningful conclusions can be drawn about the learning outcome. Judging from the results we achieved when asking a few children to verbally describe the hot dog machine, a more qualitative approach including a post test interview would be a better approach to determine how well players understood them. We presume that implementing Honey and Hilton's recommendation to use the technology available and incorporate the assessment within the game in a playful way would produce better results and simultaneously eradicate conflicts with the stealth learning approach.

> *Assessments are often designed to measure conceptual understanding alone, rather than other learning goals, and generally rely on paper and pencil tests, rather than taking advantage of digital technology to embed assessments in simulations or games (Honey & Hilton 2011, p.53).*

The game in its current state was not meant to give its learning purpose away just yet, but to prepare the player for this step by providing them with a positive mindset and an interesting context for applying the learned content, using stealth learning. Through interaction and manipulation of puzzle elements representing the procedural concepts, the players should have been able to form spontaneous concepts, which later need to be transformed into scientific concepts (Egenfeldt-Nielsen, 2007). Without proper debriefing, engaging and conversation within a social group, the learned content of the game is unlikely to be fully understood and transferable to other contexts.

As mentioned earlier, we received feedback from professionals about the practical use of our game Machineers, amongst others from James Paul Gee, who stresses the need for providing a network of other media and peers: "*For me, a game needs to be embedded in a whole learning system connecting to other media, curriculum, and social media. Games like this one, it seems to me, are good for what has been called 'preparation for future learning' (a form of transfer). [...] The game does look promising*" (Gee, personal communication, July 31, 2012).

What our results do show, considering the extensively long playtime and the high amount of players stating that they would want to keep playing, is that our testers were deeply immersed into the activity and enjoyed the game, indicating that they were under the impression of playing a commercial title that was not related to any learning purpose. The expressions of emotions of the children during the playtest (Image 6.7) resemble very much the atmosphere we were trying get across from an actual programmer or programming learner whose ways of solving problems similarly consists of a mixture of experimenting, guessing, calculating and predicting. Solving puzzles in Machineers involves observing the system, making assumptions and hypotheses on how placing a certain component will affect it, observing the outcome, and adjusting one's mental model with every experience, constructing "*explanatory models for the ordering of experience*" (Bruner 2006, p.113) which in turn allow for solving later more complicated puzzles. The more difficult the problem, the greater is the sense of achievement is after it has been solved ('hard fun').

The data shows that our audience was well targeted. The very low number of players who stated that they were bored or that the challenge was too hard for them compared to the large amount of players stating that the game was fun, ok and/or tricky and challenging implies that the majority of the children were challenged, well-guided and in a state of flow. This is encouraging in terms of the design points concerning agency, immersion, narrative, intrinsic motivation and stimuli we outlined and followed. A few children also produced interesting solutions to the cart puzzle, which we did not anticipate when designing the puzzle, pointing towards a stimulation of creativity for some players.

A minor percentage of our players stated that they did not like the protagonist to be a girl. Surprisingly enough, two of them are boys and the third one is a girl herself. We can only speculate about the reasons behind her dislike. Being able to choose between a boy and a girl to play as protagonist as we initially planned would encourage customization, removing this possible source of annoyance.

Data shows that scaffolds have been used differently by different player types, as we intended them to do when designing them (despite the problem that they were not as literate in english as expected). This shows that our scaffolds are suitable for catering to a broad range of player types. Some players, as described in section 6.5.4 Interesting Player Profiles, are more ambitious and try to solve puzzles with as little help as possible, while other players access the help options more often. When trying to correlate other data about the different learner groups, we could not find any commonalities in age, gender or gaming habits. This leads us to the assumption that, in analyzing the target audience for a learning game, emphasis should be put on their playing behaviour rather than their personal data or gaming habits. In this sense, a taxonomy of learning game players might help in order to ensure that all types of players will be catered for when designing scaffolds.

7 CONCLUSION

7.1 Future Work

In order to provide an actual circle of repetition and mastery, Machineers would benefit from implementing more puzzles, while gently increasing challenge and complexity, and reducing or delaying help options. Also, the learning curve should be softened in the beginning, giving the younger players more time to familiarize with the controls. It became clear to us after the test that the game needed more introductory tutorials on how the different elements worked and how to interact with them. Using the owl puzzle as their first introduction to the puzzle system was too overwhelming, as too many components were introduced at the same time. Some very simple puzzles, going through all the basic interactions step by step without all the interface and different elements that might confuse them, would have been a better way to ease them into the system and reduce confusion and frustration.

To build upon the logical skills introduced in the initial levels and emphasize the connection with the actual programming concepts laying behind the game, more levels would need to be designed, introducing different layers of abstraction. For example, we speculated about introducing a pseudocode layer, where the matching code parts are positioned exactly where the visual parts are, and eventually an actual code layer, where everything is written top-down. To integrate this in the game, our blueprint scaffold with the solution could be replaced by a pseudocode document and the player could either change the visual layer or the real code layer according to the pseudocode layer.

Placing counters on activation buttons could open up for a design revision of the do-while button, eliminating the distinction between red button (do-while) and the yellow button (do-once), allowing to manually specify the number of calls that would follow the activation of the button - from 1 to infinite. This not only to ensure a more consistent design, but also to emphasize the ‘for loop’ metaphor.

Testing confirmed that some players would appreciate a sandbox mode where they can build their own machines, as in the early phase of this production we were hoping to be able to implement, but eventually were lacking the time. Since the idea of a built-in mini-game worked so well in the DJ puzzle, we would implement this concept in every puzzle, so that players receive audiovisual feedback on how well the machine behaves. It would be very interesting to examine how this game could be used in a context of a social community of programming learners, connected to other activities and peers as stated by Malone, Gee, and Egenfeldt-Nielsen. Enabling competition using the sandbox mode to find which player builds the most effective, fastest, most complicated, simplest, or prettiest machine would put further emphasis on the constructivist approach to procedural literacy.

To cater for individual player differences as suggested in ‘Learning Science Through Simulations and Games’ (Honey & Hilton 2011), the optimal solution would be to present the players with customized scaffolds. As of now, we provide hints based only on the time spent on the puzzle, which is sub-optimal in terms of “just in time” (Gee 2005a). The ideal approach would be to “read” the puzzle situation before giving out the hint: the game should be able to point out specific situations, like when the player is missing a certain piece, or a machine part is wrongly assembled or isolated.

The test of transferability would have to performed before the beginning of the actual game, as well as afterwards, in order to produce comparable data and assess the learning outcome. One possible solution on how to integrate testing into the game without interrupting the game play could be Hayden asking Zola to perform tests on a regular basis, claiming to determine what kind of salary she deserves or if she is ready to be promoted, but only after properly introducing the player to the story and other characters to make sure that this test will not interrupt the immersion or create a biased mindset. Those tests would work as a quick multiple-choice-with-images assessment and give the player only vague and encouraging performance feedback.

7.2 Conclusion

Investing in creating a positive mindset and fostering intrinsic motivation in the learner are proven to be paramount factors for the learning process. The first purpose of this work was to explore whether or not such an approach is encouraged if we present a young audience with a learning game that does not advertise its purpose and is not distinguishable from commercial games in terms of gameplay, visual and sound quality.

The subject of procedural literacy has been chosen for several reasons: First, because of its relevance in modern society; second, because this subject is commonly not taught to children younger than 16 years; third, because we wanted to introduce the topic to learners who perceive science negatively and as irrelevant, e.g. girls, minorities, children from single parent homes, etc. (Honey & Hilton 2011).

Computer based educational games can offer a good platform for empowering beneficial learning principles related to immersion and motivation. This work also aimed at identifying and applying a series of design principles such as 'agency', 'immersion', 'narrative' and 'fantasy', 'visual stimuli', and 'clear goals'.

With this in mind, we produced Machineers, the vertical slice of a learning game based on the constructivist and experiential learning theories and analyzed usability, immersion and learning process in various test phases using different qualitative methods.

By analyzing the process of children playing Machineers we found both positive and negative aspects resulting from our approach, our design decisions and testing methodologies. The biggest difficulty was to evaluate the transferability of the learned skills, both resulting from our choice not to introduce a pretest, and from the 'stealth learning' approach.

The results show positive aspects of Machineers in terms of immersion and intrinsic motivation. Given that players were immersed, this is encouraging in terms of the design points we identified and followed. Playtesters generally developed a positive mindset and by stealthily familiarizing them with programming concepts, they formed spontaneous concepts which later can be transformed into scientific concepts (Egenfeldt-Nielsen 2007) with the help of an "*educationally minded adult*" (Honey & Hilton 2011, p.80).

In the light of this we can encourage other projects aiming to create educational games as 'preparation for future learning' in an informal learning environment, to use stealth learning, align content, mechanics and narrative. Such an approach can captivate the players' attention and introduce them stealthily to the learning experience. The educational content embedded in the gameplay can then be made progressively more explicit to facilitate transfer. Not only would this prevent players from having a biased mindset towards the content to be learned; on the contrary, experiencing possible scenarios for the application of such knowledge might increase motivation to learn it.

BIBLIOGRAPHY

Amanita Design (2009). *Machinarium*. Daedalic Enternainment.

Armor Games (2008). *Lightbot.* http://armorgames.com/play/2205/light-bot

Armor Games (2010). *Lightbot* 2.0. http://armorgames.com/play/6061/light-bot-20

Azevedo, R. Hadwyn, A. F. (2005). *Scaffolding self-regulated learning and metacognition - Implications for the design of computer-based scaffolds.* Instructional Science (2005) 33: 367–379

Barab, S.A., and Dede, C. (2007). *Games and immersive participatory simulations for sience education: An emerging type of curricula.* Journal of Science Education and Technology, 16(1), 1-3.

Bodner, G. M. (1986*). Constructivism: A theory of knowledge*. Journal of Chemical Education, 63, 873-878.

Bogost, I. (2005). *Procedural Literacy: Problem Solving with Programming, Systems, & Play*. *The* Journal of Media Literacy 52, no. 1-2.

Bransford, J.D. & Schwartz, D.L. (2001). *Rethinking transfer: A simple proposal with multiple implications.* In Iran-Nejad, A. & Pearson, P. D., Eds. Review of Research in Education. (24) pp. 61-100. American Educational Research Association (AERA): Washington, DC.

Bruner, J. (1960). *The Process of Education*. Harvard University Press.

Bruner, J. (1966). *Toward a Theory of Instruction*. Harvard University Press.

Bruner, J. (2006). *In Search of Pedagogy Volume 1: The Selected Works of Jerome Bruner, 1957-1978.* Routledge.

Carnegie Mellon University (1999). *Alice.* http://www.alice.org/

Daniels, H. (2001). *Vygotsky and Pedagogy*. Routledge.

Davis, J. P., Steury, K. & Pagulayan, R. (2005). *A survey method for assessing perceptions of a game: The consumer playtest in game design*. Game Studies, Vol. 5, Issue 1.

Desurvire, H., Wiberg, C. (2009). *Game Usability Heuristics (PLAY) for Evaluating and Designing Better Games: The Next Iteration*. In Proceedings of the 3rd International Conference on Online Communities and Social Computing: Held as Part of HCI International 2009 (OCSC ,09), A. Ant Ozok and Panayotis Zaphiris (Eds.) (557 - 566). Berlin, Heidelberg: Springer-Verlag.

Egenfeldt-Nielsen, S. (2006). *Overview of research on the educational use of video games*. Digital Kompetanse, 3-2006, Vol. 1, p.184 - 213.

Egenfeldt-Nielsen, S. (2007). *Educational Potential of Computer Games*. Continuum International Publishing Group Ltd.

Feurzeig, W., Papert, S. (1967). *The Logo Programming Language. http://el.media.mit.edu/logo-foundation/*

Frasca, G. (2001). *Rethinking agency and immersion: video games as a means of consciousness-raising.* Essay presented at SIGGRAPH 2001. http://www.siggraph.org/artdesign/gallery/S01/essays/0378.pdf (accessed Aug 27, 2012)

Gee, J. P. (2005a). *Good Video Games and Good Learning.* Phi Kappa Phi Forum, Vol. 85, No. 2.

Gee, J. P. (2005b). *What would a state of the art instructional video game look like?*. Innovate 1 (6). http://www.innovateonline.info/index.php?view=article&id=80 (accessed Aug 26, 2012).

Gee, J.P. (2005c). *Learning by Design: good video games as learning machines.* E–Learning and Digital Media, Volume 2, Number 1 (5-16).

Gee, J. P. (2008). *Learning and Games*. The Ecology of Games: Connecting Youth, Games, and Learning. Edited by Katie Salen. The John D. and Catherine T. MacArthur Foundation Series on Digital Media and Learning. Cambridge, MA: The MIT Press (21 - 40).

Honey, M. A. & Hilton, M. L. (Eds). (2011). *Learning science through computer games and simulations*. Committee on Science Learning: Computer Games, Simulations, and Education. Board on Science Education, Division of Behavioral and Social Sciences and Education. Washington, D.C.: The National Academies Press.

Hoonhout, H. C. M. (2008). *Let the game tester do the talking: think aloud and interviewing to learn about the game experience.* In: Isbister, K. & Schaffer, N. (Eds), Game Usability: Advice from the Experts for Advancing the Player Experience (65 - 77). Burlington, MA: Morgan Kaufman Publishers.

Kolb, D. A (1984). *Experiential Learning: Experience as the Source of Learning and Development*. Prentice Hall.

Lazzaro, N. (2004). *Why we play games: Four Keys to More Emotion Without Story*. XEODesign, Inc. www.xeodesign.com/xeodesign_whyweplaygames.pdf (accessed Aug, 29).

LEGO group (1998). *Lego Mindstorms*. http://mindstorms.lego.com

Lewis, L. H. & Williams, C. J. (1994) *Experiential learning: past and present*. New Directions for Adult and Continuing Education, 62, pp. 5–16.

Lepper, M. R., Greene, D., Nisbett, R.E. (1973). *Undermining Children's Intrinsic Interest with Extrinsic Reward: A Test of the "Overjustification" Hypothesis.* Journal of Personality and Social Psychology, 28, 1, 129-137.

Lieberman, D. A. (2001). *Management of Chronic Pediatric Diseases with Interactive Health Games: Theory and Research Findings*. Journal of Ambulatory Care Management, 24(1), 26–38.

Lifelong Kindergarten Group (2006). *Scratch*. MIT Media Lab. http://scratch.mit.edu/

Malone, T. W. (1980). *What makes things fun to learn? Heuristics for designing instructional computer games*. In Proceedings of SIGSMALL '80 Proceedings of the 3rd ACM SIGSMALL symposium and the 1st SIGPC symposium on small systems. New York, NY, USA: ACM.

Malone, T. W., Lepper, M. R. (1987). *Making Learning Fun: A Taxonomy of Intrinsic Motivations for Learning*. Aptitude, Learning, and Instruction. Volume 3: Conative and Affective Process Analyses. Lawrence Erlbaum Associates. Hillsdale, New Jersey.

Mateas, M. (2005). *Procedural Literacy: Educating the New Media Practitioner*. On The Horizon. Special Issue. Future of Games, Simulations and Interactive Media in Learning Contexts, v13, n1 2005.

Moll, L. C. (1992). *Vygotsky and Education: Instructional Implications and Applications of Sociohistorical Psychology*. Cambridge University Press.

Norman, D. (2002). *The Design of Everyday Things.* Basic Books.

Olson, D. R. (2007). *Jerome Bruner: The Cognitive Revolution in Educational Theory*. London: Continuum International Pub. Group.

Papert, S. (1998). *Does Easy Do It?* Children, Games, and Learning. Game Developer, pp. 88.

Plass, J.L., Homer, B.D., Milne, C., Jordan, T., Kalyuga, S., Kim, M., & Lee, H.J. (2009). *Design Factors for Effective Science Simulations: Representation of Information*. International Journal of Gaming and Computer-Mediated Simulations, 1(1), 16-35.

Primer Labs. *Code Hero.* http://primerlabs.com/codehero

Rheaume, J. (2009). *LightBot: Learning Objectives and Game Elements.* http://www.gamescanteach.com/games/light-bot-learning-objectives-and-game-elements (accessed Aug 28, 2012)

Salen, K. & Zimmerman, E. (2004). *Rules of Play: Game Design Fundamentals.* The MIT Press.

Sedighian, A. & Sedighian K. (1996). *Can Educational Computer Games Help Educators learn About The Psychology of Learning mathematics in Children?* 18th Annual Meeting of the International Group for the Psychology of Mathematics Education, Florida, USA.

Shaffer, D. W., Squire, K. R., Halverson, R. & Gee, J. P. (2005). *Video games and the future of learning*. Phi Delta Kappan, 87 (2), (104 - 111). H.W. Wilson Company.

Sheil, B. (1980). *Teaching Procedural Literacy.* Proceedings of the ACM 1980 annual conference. New York: ACM Press, 1980, pp. 125 – 126.

Son, J.Y., Goldstone, R.L. (2009). *Fostering general transfer with specific simulations*. Pragmatics and Cognition, *17*, 1-42.

Valsiner, J. (1988). *Developmental psychology in the Soviet Union*. Brighton: The Harvester Press.

Verenikina, I. (2003). *Understanding Scaffolding and the ZPD in Educational Research*. Conference papers of AARE/NZARE, Auckland.

Vygotsky, L (1978). *Interaction between learning and development.* From: Mind and Society (pp.79-91). Cambridge, MA: Harvard University Press.

Vygotsky, L. (1993). *The collected works of L. S. Vygotsky, Vol. 2: 77 fundamentals of defect.* Rieber, Robert W.; Carton, Aaron S. (Eds.)

Wilson, K.A., Bedwell, W.L., Lazzara, E.H., Salas, E., Burke, C.S., Estock, J.L., Orvis, K.L., and Conkey, C. (2009). *Relationships between game attributes and learning outcomes: Review and research proposals*. Simulation Gaming, vol. 40 no. 2 217-266.

Wood, D, Bruner, J.S. Ross, G.. (1976). *The Role of Tutoring in Problem Solving*. Pergamon Press.