A computer based approach on autism

Final Year Project
2009

Ævar. K. Karlsson
A computer based approach on autism

Final Report

Ævar. K. Karlsson
Supervisor: Tangming Yuan & Nik Whitehead

School of Computing
Faculty of Business and Science
University of Akureyri

Submitted April 2009, in partial fulfilment of
the conditions of the award of the degree BSc.

I hereby declare that this final report is all my own work,
except as indicated in the text:

Signature ______________________
Date _____/_____/_____

Abstract

This is the final report for, the final year project 2008 – 2009, a computer based approach on autism. Its main purpose is to describe the entire project.

My work began by studying autism in detail and the people who suffer from it. Before starting on programming the software I did background research on other similar software that has been made in this field. I analyzed specific software, which is an old final year project from an unknown student at the University of Leeds. The idea of this final year project originated from that software which is called Story builder. The layout of the software, its implementation and its evaluation are all described in detail in the second part of this report.
Table of Contents

Abstract ......................................................................................................................... 2

Table of Figures ........................................................................................................... 5

Chapter 1: Introduction ............................................................................................... 6
  1.1 Prelude .................................................................................................................. 6
  1.2 Project objectives ................................................................................................. 7
  1.3 Dissertation overview ....................................................................................... 7

Chapter 2: Autism ........................................................................................................ 8
  2.1 Characteristics and symptoms ........................................................................... 8
  2.2 Diagnosis and resorts ......................................................................................... 9
  2.3 Autism in Iceland ............................................................................................... 12
  2.4 History .............................................................................................................. 13

Chapter 3: Autism and computers ............................................................................ 15
  3.1 Teaching and computers .................................................................................... 15
  3.2 Similar software ................................................................................................ 17
    3.2.1 TeachingPix2 .................................................................................................. 17
    3.2.2 Teach Town .................................................................................................. 18
    3.2.3 Story builder ............................................................................................... 19

Chapter 4: Requirement analysis .............................................................................. 21
  4.1 The client ............................................................................................................ 21
  4.2 System requirements .......................................................................................... 22
    4.2.1 Use case diagram ....................................................................................... 22
    4.2.2 Board-Maker process documents ............................................................... 23
    4.2.3 Storyboards ................................................................................................ 27
    4.2.4 Storyboard revised .................................................................................... 29
  4.3 Task list with cognitive walkthrough ................................................................ 32

Chapter 5: System implementation .......................................................................... 40
  5.1 System design .................................................................................................... 40
  5.2 System architecture ........................................................................................... 40
  5.3 First cut class diagram ...................................................................................... 41
  5.4 Class diagram final version .............................................................................. 42

Chapter 6: Evaluation ................................................................................................. 43
# Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Teaching Pix2 main interface.</td>
<td>17</td>
</tr>
<tr>
<td>2</td>
<td>Teach Town student interface.</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>Teach Town matching identical cards game.</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>Story-builder story interface.</td>
<td>19</td>
</tr>
<tr>
<td>5</td>
<td>Story-builder story example.</td>
<td>20</td>
</tr>
<tr>
<td>6</td>
<td>Use case model.</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>The log in (main menu).</td>
<td>27</td>
</tr>
<tr>
<td>8</td>
<td>Select day screen.</td>
<td>28</td>
</tr>
<tr>
<td>9</td>
<td>See schedule screen (student view).</td>
<td>28</td>
</tr>
<tr>
<td>10</td>
<td>Create schedule (supervisor view).</td>
<td>29</td>
</tr>
<tr>
<td>11</td>
<td>The log in screen.</td>
<td>29</td>
</tr>
<tr>
<td>12</td>
<td>The student interface.</td>
<td>30</td>
</tr>
<tr>
<td>13</td>
<td>The supervisor interface part 1.</td>
<td>30</td>
</tr>
<tr>
<td>14</td>
<td>The supervisor interface part 2.</td>
<td>31</td>
</tr>
<tr>
<td>15</td>
<td>First cut UML diagram of my system.</td>
<td>41</td>
</tr>
<tr>
<td>16</td>
<td>The UML diagram of my system.</td>
<td>42</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

1.1 Prelude

Autism is a syndrome which has been on increase for the recent years and it doesn’t seem to respect any borders. It doesn’t matter if a person is from a healthy family or not, is black or white, from Europe or Africa. Everyone can expect to encounter or be affected by autism at any time during their lives. In fact everyone has probably met or at least seen an individual that suffers from autism. Autism has been studied for almost a century now and many treatments and solutions have been laid out. The estimated extent of autism spectrum disorder (ASD) in United Kingdom is around 1 out of 100 children. Autism also seems to affect more boys than girls; its ratio is believed to be 3 boys for 1 girl (University of Sunderland, 2008). Autism is a sub-class of autism spectrum disorder which is divided into five categories according to the severity of the symptoms, age of onset and association with other disorders. In the following dissertation, the focus is on the first two categories which are classic autism and Aspergers syndrome and when the word autism is used, it is only a reference to those two categories. Classic autism (infantile autism) is usually diagnosed when the child is between 18 – 36 months old. Persons who suffer from classic autism have a range of intellectual functioning from below to above average. Asperger syndrome is usually diagnosed later than classic autism since the symptoms are less severe. Person who suffer from Aspergers syndrome have in general intelligence from average to above average. People with Aspergers are harder to identify and are usually just considered eccentric (Organization for autism research, 2008). Autism will be examined and the living conditions here in Iceland, for the people who are in some way affected by autism. Most people who suffer from autism seem to be eager to escape from any situation which demands any communication with other persons at all. Most of them are being drawn more and more into the world of technology which can have both positives and negatives for them. The main aim of this project is to make contacts with people who are working in the field of autism. From there to see if it is possible to develop small software which is intended to help people with autism to live their lives and that can contribute to an improved social behavior.
1.2 Project objectives
The overall aim of the project is to seek to develop software for autistic children with special needs. The following objectives will contribute the achievement of the overall aim:

1) Make an investigation on the living conditions of children with autism and their parents here in Iceland.
2) Make contacts with people who are specialized in the field of autism and capture the requirements for the intended software.
3) Design and construct software against the requirements specified in 2)
4) Investigate appropriate means to evaluate the software.

1.3 Dissertation overview
The rest of this dissertation can be organized as follows. Chapter 2 provides the general background knowledge about autism. Since the intended user of the software are autistic people in Iceland, chapter 2 will also introduce their living conditions. Chapter 3 contains a discussion on how computer based approaches have used as one of the behavioral treatments. Chapter 4 is contains the investigation of what kind of software is suitable for autistic children and their parents. This chapter will also focus on the requirements. Chapter 5 contains all information about the implementation of the software and chapter 6 will focus on its evaluation. It is the idea to contact people who are experts in the field of autism and give them a chance to evaluate the software. The dissertation summary will be in chapter 7 which is also the final chapter.
Chapter 2: Autism

This chapter provides the necessary background material about autism e.g. characteristics, diagnosis and history, that is required before continuing.

2.1 Characteristics and symptoms

A person who suffers from autism has usually symptoms from all the three main categories as outlined below:

**Qualitative impairment in social interaction:** Autistic people often lack the intuition about other that the other people take for granted. These social impairments usually appear early in childhood and will follow the person throughout their life. A common autistic behaviour of a child is like less eye contact. When the children grow up, the symptoms become more visible. The children are much unlikely to show social understanding, approach others, imitate after other people, communicating nonverbally and take turns with other children during e.g. a game. An autistic person usually displays less attachment security than a normal person but it is dependent on how severe the ASD is. They have much trouble understanding how their behaviour can affect other people feelings. Autistic persons do however form attachment with the people that affect them the most e.g. their parents. Autistic people usually have a difficult time making friendships with other people and therefore the common belief is that they want to be alone. They tend to focus more on the quality of the friendship instead of the number of friends.

**Qualitative impairment in communication:** Around 30-50 % of autistic people don’t develop enough natural speech to meet the daily requirements communication needs. An example of those characteristics that develop early may include; children starts much later to babble than usual, unusual gestures are used and less responsiveness. Later characteristics are like; delays in spoken language, they tend to repeat the last word a sentence without considering its meaning and they often take things that other people say to them to literally e.g. if a person says “hey this is a cool game”, they think that he is actually meaning that this game is cold which is of course not the case.

**Restricted and repetitive behaviour:** The Repetitive Behaviour Scale-Revised (RBS-R) categories it into the following categories: stereotypy, compulsive behaviour, sameness,
ritualistic behaviour, restricted behaviour and at last self injury. A classic symptoms for each category (in the right order) are hand flapping, arranging objects in a certain way, refusing a removal of a furniture, dressing ritual, limited focus and at last biting himself in the finger. No single category is specific to autism but autism seems to have a special pattern of occurrence and severity of the categorical behaviours.

There exists however symptoms that fall outside the three main categories mentioned above. This can be symptoms like difficulty to fall asleep, frequent nocturnal awakenings and early morning awakenings. Unusual responses to sensory stimuli are also common in autistic children, this can be either form of under-responsivity (e.g. walking into things) and over-responsivity (being very sensitive to loud noises) where the form of under-responsivity seem to be more common (CARD, 2008).

2.2 Diagnosis and resorts
So far there are no genetic or medical tests for identifying whether an individual is suffering from an ASD or not. Clinicians have to rely on behavioural observations. The diagnosis is divided into two steps: screening and a comprehensive diagnostic evaluation.

The first step-screening is in form of parent observations, impressions and concerns which are usually supported by either family photos or videos. If concerns are raised, then parents are usually asked to complete one or more of the available standardized screening instruments. They are usually in form of filling out a checklist. Examples of such screening instruments are Checklist for Autism in Toddlers (CHAT) and the Screening Tool for Autism in Two-Year-Olds (STAT). If there continues to be concern after the screening process, then it is needed to move on to the next step in the diagnosis process. The screening tools today have not yet been validated and are currently lacking evidence for effectiveness. It is by those reasons that the general population is not being screened.

The second step-comprehensive diagnosis evaluation is used to rule in or rule out an ASD diagnosis which is done by a team of professionals. This evaluation usually contains neurological, in depth cognitive and language testing and a hearing test. At last there are special assessments that have the objective to make the researchers absolutely sure whether there is a presence of ASD or not. The most commonly used assessments are Autism Diagnostic
Observation Schedule (ADOS), Autism Diagnostic Interview-Revised (ADI-R) and the Childhood Autism Rating Scale (CARS) (Organization for autism research, 2008).

If a person is to receive a positive identification of having ASD, the individual needs to have at least 6 symptoms in total from the earlier categorisation of autism. It must include at least two symptoms of qualitative impairment in social interaction, at least one symptom of qualitative impairment in communication, and at least one symptom of restricted and repetitive behaviour (Medscape, 2008).

When a child is diagnosed with autism, its family is faced with many questions e.g. will the person be able to live a normal life? What treatments should he receive? And so on. The main objectives of a treatment are to increase the quality of life for the person and his family, and to increase the person independence so that he will be able to live as normal life as possible. The first thing that the family of an autistic person should do is to seek out as much information as it can and from there they should be ready to make a decision on what treatment should best suit their child. The decision on what treatment should be chosen is an important one because each child is unique and a treatment that works well for one child might have the opposite effect on another child. The most commonly used treatments today are discussed as follows:

**Applied Behaviour Analysis (ABA):** This therapy is based upon the following theory: a reinforced behaviour is more likely to be repeated than a behaviour that is ignored. With the term reinforced behaviour it is meant that every time the child does something right, it will get some kind of a reward. When this method is used it is absolutely critical for its success to reinforce the behaviour every time that it occurs. This therapy is very effective in teaching a range of academic, social, communicative, motor and adaptive skills to the individual. The average work for each child is about 40 hours per week with a trained expert on board. The waiting list (waiting time) for an ABA therapist can be very long because of the ever increasing demand. This technique is probably the most used one today.

**Speech and language therapy:** Its main focus is on the individual communication skills. It uses the same principles as the ABA therapy where a positive behaviour is encouraged. A trained expert in speech and languages is assigned to the child. The child’s progress is assessed
by The Assessment of Basic Language and Learning Skills (ABLLS) which is an assessment which aim is to carefully track an individual speech and language skills.

**Treatment and Education of Autistic and Related Communication Handicapped Children (TEACCH):** its aim is to build upon the skills that children with autism already have. It is a structured approach which believes that the environment should be adapted to the children instead of trying to adapt the children to the ever changing daily life. Since this method doesn’t focus on acquiring new skills for the children, many families tend to not choose this one and instead choose one of the ABA therapies.

**Picture Exchange Communication Systems (PECS):** This method focuses on the communication aspects of autistic people. The main objectives are to get the language underway and the aid the ones that need it the most e.g. the ones that don’t speak at all. As with so many other methods, this one uses the same ABA principles as so many other therapies out there. This method lets the children have a lot of pictures and when the child wants something, it has to give the instructor a picture that points to the thing that it wants.

**Occupational therapy:** This method is usually used simultaneously with other therapies. An occupational expert works with the autistic individual and measures the effect of the disorder on the individual life e.g. how autism affects the individual at home or in school and so on.

**Physical therapy:** Just a normal physical expert helps the autistic individual to build up enough muscle strength, coordination and motor skills. Another kind of therapy is recommended to be used alongside with this one (Organization for autism research, 2008).

**Medical treatment:** There is in many cases that medications are chosen to treat autism. The most commonly used medicines are antidepressants, stimulants and antipsychotic. So far there are not many researches available that support the effectiveness or the safety of this kind of treatments. In many cases the medications can have the opposite effect on the individual and no known medication can relieve the autistic individual of his core symptoms which are the social and communication impairments (National Institute of Mental Health, 2008).
2.3 Autism in Iceland

Since the aim of this project is to produce software for people who suffer from autism in Iceland, it is important to examine them and their living conditions a bit first. This can give us a better and clearer idea of those who are expected to use the intended software directly or indirectly. The living conditions of the people who suffer from autism in Iceland vary from city to city. As soon as the individual has been diagnosed with autism, he receives support through kindergarten and primary school. The amount of support received depends on how serious the individual’s disability is. Many colleges in Iceland have started offering a 4 year long study that is specially designed for children who suffer from some kind of disability conditions. The aim of this study line is to give the individual a possibility to stay longer in school and get increased control and help with their lives. This 4 year long college study is becoming popular especially from the children parents point of view. After this 4 yearlong study, the individuals usually try to find some work to do, at least at some point. The labour market in Iceland isn’t believed to be receptive enough for the individuals with any kind of disability conditions. For those individuals who live in Reykjavik have the possibility to further their study in Institutes that are specialized in teaching children with any kind of disability conditions. Currently there is a shortage of places for children in institutes which offer them to stay for couple of days in a month and just to give their parents a little rest. A special residential centre for people with any disability conditions is becoming increasingly popular today. A residence in apartments, which are not part of special residence centre, is also becoming popular. The main concerns for autistic people today are the residence issues and the labour market which can be really tough for most of those individuals. The living conditions are believed to be worse in the capital Reykjavik than in the cities that are scattered around Iceland. This is mainly because the majority of Icelanders live there which means that the majority of autistic individuals are there also. In almost every city around Iceland, there is a special treatment centre that autistic people and their parents can go and get support and information about the disorder. This support can be crucial, as mentioned before in this dissertation, for the decision making what treatment should be chosen. For the years 2005, 2006 and 2007, 0.6% of the people in each year suffered from some symptoms of the autism spectrum disorder. For the last years, the numbers of diagnosed people have been on the increase. The reason is believed to be the same as mentioned in the end of chapter two, which is broader definition of autism and increased awareness (Umsjónarfélag einhverfra, 2008).
### 2.4 History

The word autism has been around for almost a century now and it was first defined in 1910 by the Swiss psychiatrist Eugen Bleuler. The word autism is derived from the Latin word *autismus*. Bleuler believed that he was describing symptoms of schizophrenia: a person that suffers from a morbid self-admiration which any outside influence or disturbance would become absolutely intolerable. Autism was first used in its modern sense in 1943 when it was introduced by the American doctor Leo Kanner. He labelled it as early infantile autism in a report where he described 11 children that seemed to have striking behavioural similarities (Autism Resources, 2008). His description of autism is still used today even though it has gone through many changes over time. During the time period 1950 – 1970, Kanner believed that autism wasn’t as inherited as he first thought. He and others thought that this disorder had its origins in the parent’s upbringing and this was very typical for people’s thoughts during this time period. In fact the parent’s were blamed for their children’s behaviour because scientists thought that they had failed in giving their children enough love, kindness and other emotional stimulation. The children were believed to have created a shell around them, for protection against their parent’s denial for them. This shell caused that they were closed inside themselves and they were unable to form a motional bonds with other people. This of course led to many questions such as how it would be possible for the parent’s of a child with autism, to be able to have normal children as well? This is in fact in contradiction to their beliefs in those days. This time must have been very difficult for the parents of children with autism. In the early 80’s, people were started to reject the earlier theories because researches were started to indicate that other factors would be the cause. Researchers claimed that the main cause would be because of a disturbance in the individual’s central nervous system. This theory has only got stronger in duration of time and it has become a known fact today that autism is because of a biological factors. It is known that the genetic combination of the individual is an important deciding factor; unfortunately no specific genes have yet been identified (Páll Magnússon, 1993). The number of people that have diagnosed with autism has skyrocketed during the past 30 years. People are not sure why there is so much increase but there have been several theories made. Some of them state that the improvements in diagnosis are responsible alongside with a broader definition of autism and increasing awareness among the people in the medical industry. Other theories claim that the
potential impact of many environmental factors might be responsible such as toxins (Organization for autism research, 2008).

This chapter introduced the term autism in a broad view as to looking into the living conditions of the autism population in Iceland. The chapter also provided general statistical results that were collected from various sources.
Chapter 3: Autism and computers

This chapter focuses on the impact that computer technology has been making in terms of helping autistic people in many various ways. Other software will be examined at the end of the chapter which have many things in common with the software that will be produced alongside this project.

3.1 Teaching and computers

So far all therapies that have been mentioned in the previous chapter have been quite successful and popular. Their success is quite difficult to measure in terms of efficiency and because of that there have been an ongoing debate regarding which one of them is the best approach for autistic children. One thing that all of the expert agree on about all of these therapies is also their main weakness as well. They all agree on that they all require intensive, often one on one training that can be in the form of many hours to even months. It is often said that the time spent working with one student with special needs is the time that is unavailable to all the others autistic children. One other thing is that due to the students social impairments, the traditional instructions in the form of social interaction can make it even more difficult for the student to actually learn (Connexions, 2008).

Computer technology is a fascinating subject for many people. They tend to get totally charmed by it and lose all interest for other things that is around it. Autistic people are no exception to that rule. Computer aided technology has been increasingly used for the last years, both for educational and medical purposes. Then there was the question whether it is possible to combine these two in the fight against autism and other disorders, and it has been done so far with really amazing results. There are mainly 5 aspects that computers can aid in the fight against autism. 1) diagnosis, 2) assessment of each individual learning skill, 3) aid in form of teaching, 4) assist in communication and finally 5) help with the development of research models which are meant to help researchers to understand autism (Autisme France, 1995). The main focus of this project is going to be using the computer as a teaching machine.

B.F. Skinner, a well known psychologist, was the first one that thought of the idea to use machines as teaching devices in the late 1950s. His ideas became very popular but unfortunately
in the early 1970s his dream came to an end since there was a lack of concrete evidence of the efficiency of computerized teaching. The cost of fairly good computer equipment in those days was also a major factor, but now a day’s good computer can be bought for a very reasonable price. Nevertheless, Skinner’s idea laid the groundwork for computer assisted instruction for the years to come (Connexions, 2008).

During the last couple of years computer aided teaching has become really popular in many fields and autism no exception of that. There are several reasons that backup that claim to use computers to help autistic children in one or many ways. Studies have shown that autistic children often find learning with computers much more interesting than learning with trained teachers. Many autistic children suffer from a sensory-overload; which means that the children have difficulties filtering out unneeded and irrelevant sensory input. By using computers, this problems impact is completely minimized since the computer displays only important and necessary information. The computer also has the ability to provide a very stable and regular work environment; it has the ability to produce, is without fear of fatigue or boredom and it provides predictable and immediate feedback as long as the child is willing to participate. Other good thing about the computer-based environment is that both goals and expectations are very clear to the child and the child can work on its own pace instead of e.g. the instructors pace. Some programs can also be customized to fit the child’s needs, and its structure can be designed to increase in both its scope and difficulty as the child progresses (Connexions, 2008).
3.2 Similar software

Many programs have been made so far for autistic people. In this section, several of those programs will be examined and their pros and cons will be identified.

3.2.1 TeachingPix2

TeachingPix2 is software that is used worldwide by many mainstream schools, specialized schools, language and speech therapists and parents that want to teach their child at home. This software includes 10000 flashcards in 65 categories that can be previewed and used to teach children speech, language, communication and social skills.

![Teaching Pix2 main interface](image)

**Figure 1:** Teaching Pix2 main interface.

As can be seen in figure 1, this program works mainly as a database for images and a description of that image. This software is ideal for teaching autistic children since many known therapies involve pictures and using them to explain desires and behaviors. The bad thing about this software is that it is not designed to be simple. It is almost impossible for an autistic child to study this software on its own and it doesn’t involve any exercises for the children. This means that they don’t receive any feedback except having one of its parents to praise it for its achievements. This software should mainly be used by parents to print out or/and let the child study them and learn the expressions that follow each picture. This software contains several bugs but nonetheless if it is used correctly, it should make a good addition to other software that focus on teaching autistic children (Early learning site, 2008).
3.2.2 Teach Town
Teach Town is software that is specially developed for autistic children and one or more supervisor. A parent needs to register as a supervisor and can then supervise a certain number of autistic children. The program is linked to a special server via the internet and recommends for each child appropriate exercises. These exercises are both computer lessons and off computer activities, it is recommended to focus on them equally. The recommended exercise menu is displayed in figure 2.

![Teach Town student interface.](image)

Figure 2: Teach Town student interface.

The software has two different interfaces, one for the supervisor and one for the student. The student interface is made simpler and more accessible than the supervisor interface. The idea and the design behind the software are really good and there don’t seem to be many flaws in it. The computer lessons are really well laid out and practical for the teaching of an autistic child.

![Teach Town matching identical cards game.](image)

Figure 3: Teach Town matching identical cards game.
An example of one of the computer lessons is displayed in figure 3. The only thing that might be a little problem with the software is that; in the student interface, the child might get overloaded with information that it might not understand because of its disorder. In the whole this software is really good and it seems like the developers have put a lot of effort in which makes it a good choice for a parent of an autistic child (Teach Town, 2008).

3.2.3 Story builder

Story builder is software that was specially designed with the autistic people’s needs in mind. The user can both view previously made stories and create new ones by the interface that can be seen in figure 4.

![Story Builder Interface](image)

**Figure 4: Story-builder story interface.**

Creating a new story is very simple in story builder. The user simply needs to name his story and then he can immediately start working on the next page. For each page, a text is inserted and a picture that follows the text can be inserted as well. A created sample story page can be seen in figure 5.
Story builder is simple software that does exactly what it should do and nothing else. Its design is simple and it is not hard for a normal person to navigate in its environment. As with all other software it contains minor bugs but none of them affect the final outcome. This software is the inspiration and groundwork for the project that will be designed and it will probably contain the same core functions (University of Leeds, 2008).

Computer technology has been slowly making its way into the medical world and it has finally become a recognized way as a therapy for patients. All kinds of software are available for treating autistic people today and almost all of them rely on a supervisor or an adult to help the autistic child getting started. The story builder which is the inspiration software for this project was also examined.
Chapter 4: Requirement analysis

The emphasis of this chapter is on getting an idea of what the requirements are for the intended software. A profile of the general user is made at the beginning following with a description on how the requirement gathering was done. A revised storyboard is also made alongside with cognitive walkthrough on it and those two can give some idea of how the software will look like.

4.1 The client

The potential users of the intended software are people that suffer from, or are affected by autism in any way. To gather the requirements for the intended software, some possible users needed to be contacted. Since the clients are mainly the autistic people, the aim was set to make contact with people that work with them in the daily life. There is an elementary school in Akureyri that is called Síðuskóli. It has a department that is specialized in working with autistic children. The age range of these children is from 6 – 16 years old, which is for normal children in Iceland who are in elementary school. The head of the department, who won’t be mentioned by name due to privacy reasons, was contacted in October 2008 and a meeting was following. The main idea was to interview her and her colleagues and gather a rough idea on what kind of program would be good for autistic children and their parents. The interview was set up in the way that the interviewer (the author of this dissertation), the head of the department and her colleagues sat around one table, had some general conversation and everyone was allowed to express their thoughts when they wanted. The best thing about this method is that it is able to get most of the positive results as a normal interview would do, but it also manages to unify their general thoughts and ideas, that might be too different otherwise. Subjects all agreed that the software should be designed with the needs of the autistic children parents in mind rather than the teacher’s or school’s needs. After interviewing the teachers and sharing my thoughts with them we agreed on the kind of software that would be quite useful for the average autistic child and their family. The software requirements and main ideas will be discussed in detail next.
4.2 System requirements

4.2.1 Use case diagram

Use case 1: The autistic child will be able to see the daily schedule that the supervisor created for him.

Use case 2: The supervisor (parent) can create a schedule for an autistic child.

Use case 3: The supervisor (parent) can save his previously created schedule for future use.

Use case 4: Both the autistic child and the supervisor can load a previously created schedule.
4.2.2 Board-Maker process documents

Board-Maker process document 1 – Schedule the week for an autistic child

<table>
<thead>
<tr>
<th>Version</th>
<th>Description</th>
<th>Changed By</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Schedule a week for an autistic child.</td>
<td>Æ.K.K.</td>
<td>10/04/09</td>
</tr>
</tbody>
</table>

Brief Description: The process for creating a schedule for autistic children.

Preconditions: One grown up person that has supervisor access.

Business Trigger: One grown up person who acts as a supervisor for one or more autistic children.

Performed By: Individual supervisor.

Related Processes: See schedule.

Basic Flow: The supervisor creates a schedule. The supervisor can do everything that the student can. The supervisor can add a task with picture on the selected day. He can also delete tasks and erase the schedule if he wants.

1. Create schedule

1.1 Arrange activities for the days of the week.

1.1.1 Create activities.

1.1.1.1 Select day of the week.

1.1.1.2 Insert the name of the task.

1.1.1.3 Select appropriate picture for the task.

1.1.1.4 Press add schedule button.

1.1.2 Delete activity from a day.

1.1.2.1 Select day of the week.

1.1.2.2 Select the number associated with the task that should be deleted.

1.1.2.3 Press the delete button.

1.1.3 Erase activities.

1.1.3.1 Select day of the week (optional, only if erase day is chosen).

1.1.3.2 Choose either day or week to be erased.

1.1.3.3 Press the erase button.

1.2 Be able to load a previously created schedule (See process document 3).

1.3 Be able to save a created schedule (See process document 4).

1.4 See the schedule of the day just as the supervisor is making it.

1.5 Be able to exit the system.
Board-Maker process document 2 – See schedule (student view)

<table>
<thead>
<tr>
<th>Version</th>
<th>Description</th>
<th>Changed By</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>View a week created by the supervisor.</td>
<td>Æ.K.K.</td>
<td>10/04/09</td>
</tr>
</tbody>
</table>

**Brief Description:** The process of an autistic child or a student viewing his schedule.

**Preconditions:** Perform load schedule.

**Business Trigger:** Is triggered when the autistic child wants to view his schedule for the week.

**Performed By:** Individual autistic child/student.

**Related Processes:** Schedule the week for an autistic child.

**Basic Flow:** The week is viewed each day at a time.

1. *See schedule.*
   
   1.1 Load a previously created schedule (See process document 3).
   1.2 View each day.
      
      1.2.1 Select day of the week.
      1.2.2 Look at the task for the selected day.
   1.3 Be able to exit the system.
Board-Maker process document 3 – Load schedule

<table>
<thead>
<tr>
<th>Version</th>
<th>Description</th>
<th>Changed By</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>A person loads a created schedule.</td>
<td>Æ.K.K.</td>
<td>10/04/09</td>
</tr>
</tbody>
</table>

**Brief Description:** The process when either supervisor or student loads a previously created schedule.

**Preconditions:** None.

**Business Trigger:** Is triggered when a student or the supervisor wants to view an existing schedule.

**Performed By:** Student or the supervisor.

**Related Processes:** Save schedule.

**Basic Flow:** The student loads a schedule that his supervisor has previously created for him.

1. *Load schedule.*
   1.1 Log in to the system as either student or supervisor.
   1.2 Press the load button.
      1.2.1 Search for the schedule.
      1.2.2 Select the schedule.
      1.2.3 Press the open button.
   1.3 The schedule is loaded into the system.
Board-Maker process document 4 – Save schedule

<table>
<thead>
<tr>
<th>Version</th>
<th>Description</th>
<th>Changed By</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>A supervisor saves a created schedule.</td>
<td>Æ.K.K.</td>
<td>10/04/09</td>
</tr>
</tbody>
</table>

Brief Description: The process when the supervisor saves a created schedule.

Preconditions: Schedule a week for an autistic child (process document 1).

Business Trigger: Is triggered when the supervisor has finished creating a schedule for one of his students.

Performed By: Supervisor.

Related Processes: Load schedule.

Basic Flow: Once the supervisor has created a schedule that he is happy with, he saves it so that it can be accessed later.

1. Save schedule.
   1.1 Log in to the system as either student or supervisor.
   1.2 Press the save button.
      1.2.1 Create a valid name for the schedule.
      1.2.2 Press the ok button.
   1.3 The schedule is saved to your computer.
4.2.3 Storyboards

Here below are images that can give some idea of how the system will finally look like. Note that it is possible that some changes will happen on this look during time.

Here below on figure 7 is the log in screen. A person can either log in as a student or a supervisor by clicking on one of the two available buttons that are displayed on the screen.

![Log in screen](image)

*Figure 7: The log in (main menu).*

Here below on figure 8 is the screen which appears when a person has either logged in as a student or a supervisor. The person can select a day of the week by clicking on a button. There is one button for each day of the week. The person can also load an old schedule by clicking on the load button below.
Figure 8: Select day screen.

Here below on figure 9 is the screen which appears when the student has selected a day of the week. The student sees a list of actions that his supervisor has created for him for some particular day of the week. Next to each action there is a picture which represents the action in some way. This window screen can be scrolled down if there happens to be too many actions scheduled for the day.

Figure 9: See schedule screen (student view).

Here below on figure 10 is the screen that appears when the supervisor has finished selecting a day. This is the screen where he can create a schedule for a student. There are two windows on the screen. One window is for the day which is currently being scheduled and the other contains a list of many activities which can be moved to the other window and vice versa. There is a dropdown list on top of the screen and the supervisor can select an activity group and every
action which the group contains will be displayed in the window below. At the bottom of the screen are three buttons. The supervisor can save and load his schedule and one confirm button which can be pressed when the day is ready. After clicking the confirm button, the select screen will be displayed which is shown in figure 8.

![Create schedule](image)

Figure 10: Create schedule (supervisor view).

### 4.2.4 Storyboard revised

Here is a revised version of the storyboard which will introduce some graphical changes to the software. The reason for most of these changes is because of programming limitations. Some ideas had to be changed because it was much easier in terms of programming and it was also more efficient in some cases. A cognitive walkthrough was made on this version of the storyboard and it can be seen in the next section of this chapter.

![Log in](image)

Figure 11: The log in screen.

This is the revised log in screen of the software. It works just as the previous version of the log in screen but a password field has been added. This means that if a user wants to get access to the
supervisor part of the software, he is going to have to insert the correct password into the password field.

Figure 12: The student interface.

This is the revised student interface of the software. A Combo Box has been added to it which allows the user to select a day within the interface itself.

Figure 13: The supervisor interface part 1.

This is the supervisor interface part 1 of the software. It has the same Combo Box added to it as the student interface. Now it has only one screen for the schedule making and next to it is a JPanel that contains some available editing commands. Above this JPanel are two buttons called modify and clean. These buttons allow the user to switch between JPanels that contain different editing commands. The modify JPanel is displayed on the part 1 and it contains an insert and a delete option. The insert option allows the user to create a task by writing it into a text field and then clicking the insert button will insert it into the schedule. The delete option allows the user
to delete an existing task in the schedule. The task number is selected in a Combo Box and is deleted by clicking the delete button. If no task with the selected number exists, then a pop-up window appears saying that it is impossible to delete a non-existing task.

**Figure 14: The supervisor interface part 2.**

This is the supervisor interface part 2 of the software. It is identical to part 1 except it contains the clean JPanel which is displayed if the user selects the clean button above. This clean option allows the user to totally empty one whole day or the entire week of tasks. The user selects an option via a Combo Box and then pressing the erase button right next to it.
4.3 Task list with cognitive walkthrough

The four questions which are used to evaluate each action of the software

1. Is the user likely to use this action?
2. Is the user likely to see the action?
3. Is the user likely to understand what the action will do before he uses it?
4. Is the user likely to understand if the action worked or not (does he gets any feedback)?

The name of each task is written in bold and actions are in italic. This cognitive walkthrough is performed on the final look on the software.

Log in as a student.

Click the student button

1. The user (both student and supervisor) is very likely to use this action since it is the only way to use student interface of the software.
2. It is very easy to see the button since it is both big and is placed on a very visible place on the screen.
3. The button is labelled with the word “student” on it. It should in most cases give the user the idea that this should log him in as a student.
4. When the user clicks on the button, he will get another screen (the student screen) in front of him.

Log in as a supervisor.

Enter password in the password text field

1. If the user wants to enter the supervisor interface of the software, then it is essential to enter a password into this field.
2. The text field is placed on a visible place, above the supervisor button.
3. The text field is labelled with the word password so it should give the user the idea that a password is required.
4. If the user has entered the correct password in the text field he will enter the supervisor interface. If wrong password, a warning box is displayed and nothing else happens.

*Click the supervisor button*

1. The user (the supervisor) is very likely to use this action since it is the only way to use supervisor interface of the software.
2. It is very easy to see the button since it is both big and is placed on a very visible place on the screen.
3. The button is labelled with the word “supervisor” on it. It should in most cases give the user the idea that this should log him in as a supervisor.
4. When the user clicks on the button, he will get another screen (the supervisor screen part 1) in front of him. This happens only if he has entered the correct password in the field above the button. If wrong password, a warning box is displayed and nothing else happens.

*View different day of the week (in both student and supervisor mode).*

*Select a day with the dropdown box*

1. The user is likely to use this button. Every time when he want to look at another day on the schedule display.
2. The box is displayed high on the screen right next to the schedule display. It is the best possible location that is left on the screen. The location should be good enough since there aren’t many actions and items on the screen.
3. The box is labelled with the word day which should give the user some idea about its functionality. Inside the box is also the current day that is being viewed.
4. Once the user selects another day, the day will become selected in the box and the schedule display will change to the selected day.
Look at the schedule in the schedule display

1. The user is going to use this action in the only way possible which is to look at his schedule for the selected day.
2. It is very easy to see the schedule since it takes the most part of the whole screen.
3. It is easy to understand that this is only the display of the created schedule since it is the main objective of the software.
4. The user doesn’t get any feedback in other form that he can look at the schedule of each day of the week.

Add a task.

Write the name of the task in the insert text field

1. If the user is going to insert a task into the schedule, then it is necessary to enter a task into this field.
2. The text field is right next to the insert button. All insert task items are located together which makes it easier to find them.
3. The text field is right next to the insert button and marked as insert task which means that it should give a good idea of its functionality.
4. Once the insert button is pressed, the task that was entered into the text field will appear at the bottom of the schedule display at the centre of the screen.

Select a picture with the picture dropdown box

1. The user is likely to use this button. Every time when he wants to insert other picture than the default selected picture.
2. The box is displayed high on the screen right above the insert text field. It is not hard to find since it is grouped with the insert task actions.
3. The box is labelled with the word picture which should give the user some idea about its functionality. Inside the box is also a descriptive name of the selected picture.
4. Once the insert button is pressed, the task appear on the schedule display along with the selected picture.

Select a day with the dropdown box

1. The user is likely to use this button. Every time when he want to look at another day on the schedule display.
2. The box is displayed high on the screen right next to the schedule display. It is the best possible location that is left on the screen. The location should be good enough since there aren’t many actions and items on the screen.
3. The box is labelled with the word day which should give the user some idea about its functionality. Inside the box is also the current day that is being viewed.
4. Once the user selects another day, the day will become selected in the box and the schedule display will change to the selected day.

Click the insert button

1. The users will in most cases use this button since it handles the creation of a new task in the schedule.
2. The button should be relatively easy to see.
3. The button is labelled with insert which should give a good idea of its functionality. There is also a sign above this feature which is labelled insert task.
4. Once the button is pressed, the inserted task will appear at the bottom of the schedule display at the centre of the screen.

Delete a task.

Select a day with the dropdown box

1. The user is likely to use this button. Every time when he want to look at another day on the schedule display.
2. The box is displayed high on the screen right next to the schedule display. It is the best possible location that is left on the screen. The location should be good enough since there aren’t many actions and items on the screen.
3. The box is labelled with the word day which should give the user some idea about its functionality. Inside the box is also the current day that is being viewed.
4. Once the user selects another day, the day will become selected in the box and the schedule display will change to the selected day.

*Select task number to delete from the delete dropdown box*

1. This action will only be used by those users who need to correct a mistake in their schedule.
2. The buttons location is appropriate given the likelihood of use.
3. Inside the dropdown box is the number of a task that is in the schedule. The dropdown box is right next to the delete button. This should give a good idea of its functionality.
4. Once the delete button is pressed, the selected task will be deleted from the schedule.

*Click the delete button*

1. This button won’t be used as much as the insert feature, that is the reason why it is below it. This action will only be used by those users who need to correct a mistake in their schedule.
2. The buttons location is appropriate given the likelihood of use.
3. The button is labelled with delete which should give a good idea of its functionality. There is also a sign above this feature which is labelled delete task.
4. Once pressed, the selected task will be deleted from the schedule.
Erase a whole day/week.

Select a day with the dropdown box

1. The user is likely to use this button. Every time when he want to look at another day on the schedule display.
2. The box is displayed high on the screen right next to the schedule display. It is the best possible location that is left on the screen. The location should be good enough since there aren’t many actions and items on the screen.
3. The box is labelled with the word day which should give the user some idea about its functionality. Inside the box is also the current day that is being viewed.
4. Once the user selects another day, the day will become selected in the box and the schedule display will change to the selected day.

Select method to erase in the erase dropdown box

1. This feature is not likely to be used in every run. It will only be used by those users who want to erase their existing schedule.
2. The user is not very likely to see this action. Only those who are willing to explore the software in more detail. The location of the dropdown box is appropriate given the likelihood of use.
3. Right next to it is the erase button and above it is a label clean selected. Inside the dropdown box, the user can choose from selecting day or week. This feature can maybe represented in a better way but this feature is not likely to be used.
4. Once the erase button is pressed, the selected day or the whole week will be erased from the schedule.

Click the erase button

1. This feature is not likely to be used in every run. It will only be used by those users who want to erase their existing schedule.
2. The user is not very likely to see this action. Only those who are willing to explore the software in more detail. The location of the button is appropriate given the likelihood of use.
3. The button is labelled with erase which should give a good idea of its functionality.
4. Once pressed, the selected day or the whole week will be erased from the schedule.

**Load a schedule.**

*Click the load schedule button*

1. The user might want to use this action so that he can load a schedule that was created when the software was run at a different time.
2. The button is big and easy to see.
3. The button is labelled load schedule which should give the user a good idea of its functionality.
4. Once pressed, the user will get an open file screen and there he can select a previously made schedule.

**Save a schedule.**

*Save schedule button*

1. The user might want to use this action so that he can save his schedule for another time.
2. The button is big and easy to see.
3. The button is labelled save schedule which should give the user a good idea of its functionality.
4. Once pressed, the user will get an save file screen and there he can save his schedule.
Exit the system (in both student and supervisor mode).

Confirm button

1. The user might use this action when he has finished looking at his schedule. The user doesn’t have to use it because he can always use the X button to close the window as well.
2. The button is big and easy to see.
3. The button is labelled confirm which should give the user some idea of its functionality. There is a possibility whether some other label might be more appropriate like exit or ok, but they can be a little misleading as well.
4. Once pressed, the student interface will close and the user will see the login screen once more.
Chapter 5: System implementation

This chapter emphasizes on how the intended software is going to be implemented in terms of system design, architecture and class modeling.

5.1 System design
The software that is currently under construction will be programmed in the programming language java. Java was selected since the developer had already acquired most of the knowledge that will be needed to program the software. Java is also one of the most used programming language that are used today for software design, which means that any additional knowledge needed for the software implementation should be easy to acquire. The software was programmed in the jGRASP programming environment because of my good previous experience with it. The software was later on moved into the Eclipse programming environment, I fixed all programming bugs with the find bugs plug in and most of the coding style was fixed with the help of the check style plug in.

5.2 System architecture
The system will be split into two components (two tier architecture): 1) model and 2) view. It is an architecture that is very similar to the well known MVC (three tier architecture) and is in fact a simplified version of it. The MVC architecture is split into three components which are the following: 1) Model – These are the classes which provide the functionality of the system, 2) View – These are the classes which provide the graphical user interface to the user and 3) Controller – These are the classes which handle the input from the user and send messages to the other two components (Bennet, Simon and others. 2006).

The intended software will be divided into the same components as the MVC architecture but instead of having special class which handles the control classes, they will be joined to the model classes for simplicity reasons.
5.3 First cut class diagram

The view components are marked as boundary classes. Their role is to handle the GUI of the system.

The model components are marked as entity classes. Their role is to contain the system overall functionality.

The control components will be placed both in the model and view classes. Their role is to handle all input from the user.

Figure 15: First cut UML diagram of my system.
5.4 Class diagram final version

As can be seen from the two class diagrams, the structure of the software changed dramatically. The biggest change is that the software went from 4 entity classes to only 2. This change was possible with the use of having linked list for each day and then one that acted like a temporary storage while the list was being used in some way. The class MainController does almost all the work in this program from being the main storage for the linked lists to containing all the methods to manipulate the linked lists. The class Task works exactly like the class Action was meant to do except that it contains now a url for a picture as well. As long as the main class exists, the user can create as many instances of the other classes as he wants. There is no particular change in the boundary classes except with a deletion of the class supervisorInterface2.
Chapter 6: Evaluation

This chapter emphasizes on all the different types of evaluation that were made on the program. The evaluation was split into two forms; first a black box test (which led to some bug fixing) and then the user evaluation.

6.1 Black box testing

Before the actual user evaluation, the program was tested by using a simple form of black box testing. The black box testing was tested in the way that couple of people studying on the computer science department were selected and asked to test my software and do everything in their power to make it give the wrong output. They were also asked to state their opinion on the layout of the software. I received the following feedback from the black box testing (some of these features were fixed before the user evaluation):

- *It is possible to create an (empty) task that contains only white space:* This has been fixed.
- *Set limits on the input field:* not fixed
- *Insert more confirmation windows e.g. when saving:* not fixed
- *When in open a file, have a filter that displays only text files:* I knew about this already, there just wasn’t enough time to implement this feature.
- *If a user is saving a schedule and then wants to continue working on it, the contents of the day that was selected gets lost:* This has been fixed.
- *Have something to show in what file the user is working on:* This has been partially fixed. After opening a file, the user gets its URL in the header of the program.
- *When in the student interface, maybe skip the combo box and have big buttons for each of the days:* not convinced if this is a problem or not.
- *The cereal picture is too complicated:* I was pointed at specific standard of simple pictures for kids. I inserted a simpler picture.
6.2 User testing

I contacted the teachers at Síðuskóli, got a permission to visit them so that they would have the opportunity to evaluate the produced software. These are the same teachers that the user requirements were based upon which means that the results should be quite interesting. The evaluation was performed during the teachers work time and it was done by trying to simulate a real life scenario as much as possible. The teachers were allowed to be in the same room at the same time. All of them as a group were given a couple of minutes (5 – 10) to play with the software. Next they were given a questionnaire containing around 10 multiple answers questions and 1 section in the end containing couple of lines so that if there are some other things that they want to point out then they can do it there in their own words. The multiple answers questions will each have 5 options (e.g. Excellent, Good, Ok, Bad, Awful) and these options should provide them with as much variety needed in order to answer those questions without any difficulty. The reason why exactly 5 options are given for each multiple answer question is to make sure that the teacher doesn’t feel forced to pick as side. Each teacher is on the other hand encouraged to choose something else than neutral. I stayed close to the group while they were testing the software so that if any questions arised, then they could always ask me. When the group had spent a sufficient time doing the testing, I went away and waited while the teachers answered the questionnaire. This was done in order to prevent that my presence would effect their answers. The questionnaire didn‘t include any names so there is not possible to relate the questionnaires to each of the teachers.
6.3 The results

The evaluation went according to plan and the results were similar to what I expected. They all agreed that the user interface is very simple and all actions that are available to the user are straightforward. The thing that seemed to bother them the most is that the possibility to print out the schedule is not there. This note didn’t surprise me at all since I already knew about this and wanted to implement it but it was left out due to lack of time. They also wanted to have the option to view the whole week which would then give them a better overview of the selected week. There were not many other things that they felt were missing in the program except that it would be nice to have more pictures that could then be categorized according to similarities. They don’t think that the program is ready for the public yet but emphasized on that it was looking very promising, I am on the right path and they encouraged me to continue on it by looking at the three things mentioned earlier.

This chapter looked at the 2-step evaluation process of the program. The evaluation began with a simple black box test and after some minor bug fixing the program was evaluated by some possible users (teachers of the autism department in Síðuskóli). The program got good remarks but it even so they felt that it wasn’t ready for the public.
Chapter 7: Final words

This chapter will summarize the project and go discuss all possible future works that can be made from it. Finally there will be a sub-chapter called personal reflections which will go into the impact that this project has had on me and what I have gained from it.

7.1 Conclusions

I managed to do a detailed background reading on the subject which helped me to fully understand the task at hand. The list of requirements for the software was gathered successfully from the teachers and the implementation went better than I hoped. A proper user evaluation test was done and the results were similar to what I expected. The software is currently on its beta version stage and is in fact close to be finished.

The teacher at the specialized department for autism at Síðuskóli liked my software but thought that it would still need some further work. I intend to do some further work on the software on my own just to increase my JAVA knowledge and hopefully reach the final version.

7.2 Future work

I encountered multiple obstacles while implementing the software but I managed to overcome most of them. These obstacles that I encountered did however change the final layout of the software but not in the manner that it affected the user experience. It did put some limits on the software. The things that could be worth adding to the software are the following;

- Allow the user to print out the schedule.
- Allow the user to view the whole week.
- Allow the user to add pictures to the existing collection of pictures.
- Allow the user to make a schedule that is not associated with some particular day e.g. create a schedule for going to the library.
- Storing the data in a database could be a good idea instead of text files.

As the list above suggests, there can definitely be some more work done on the software in terms of programming. Other work that could be interesting is to contact parents that have an autistic child and focus more on them than the teachers since the software is intended for them. The user
evaluation test could then be made in cooperation with the parents or even the institute here in Iceland that is focused on helping autistic people: Umsjónarfélاغ einhverfra.

7.3 Personal reflections

I learned a lot from this project in the way that I gained some real life working experience and in working individually on a project where I was responsible for all major decisions. I gained some valuable programming experience in programming software with all the knowledge in JAVA that I had previously gained and learning individually the extra bits required to create working software. It was also really interesting in working on software in “cooperation” with people outside of the University who was in this case the specialized department of Síðuskóli for kids with autism. All of this can benefit me on the employment market and this could make me a valuable asset for companies that are looking for people with that have experience in working individually and in cooperation with other people.

If I were supposed to do this project all over again, then there are few things that I would do differently. One thing that would have been good to do is to have used the Christmas break better in terms of trying to gain some more programming experience that would then have given me some more time to work on the software and maybe reach an acceptable version for the public market.

In the end I am very satisfied with how the project went and I think that all my time spent on it was worth the effort. The expectations were maybe a little bit high during the first months of the project but after scaling it down to a reasonable size; I think that I managed to produce software that fulfilled most of what was expected of it.
References


Appendix A: Project Outline first draft (made in September 2008)

Until the end of September:

- Background reading along with collection on relevant materials.
- A rough plan on how the chapters will be constructed into subchapters.
- Start on the introduction (maybe!).

Throughout October:

- Work on chapter 1, which will be about autism in general and chapter 2, which will be about autism here in Iceland and the help that people who suffers from autism can get.
- Do the references, alongside I use them.
- Hand my supervisor a draft on my first and second chapter.

Throughout November:

- Finish the chapter 1 and 2 if needed.
- Do and document the design analysis.
  - Use case diagram
  - The architecture

Christmas vacation:

- Look at the feedback that I got from my chapters and use them to make the final version (hopefully).
- Get the interim report ready.
- Start coding (hopefully I will have a clear idea on what exactly I will be coding at that point).
Throughout January:

- Do the coding and hopefully finish it by the end of January.
- Update my report.
- Class diagram.
- Declare my design decisions.

Throughout February:

- Focus on the evaluation of my program.
  - HCI expert.
  - Expert evaluation (expert in the field of autism).
  - User trial (if there is time available!!).
- Make some final words and conclusions.

Throughout March:

- Prepare for the presentation.
- Time available to finish everything that is unfinished.
- Finish the final report.
Appendix B: Project Chapter’s Layout

- Abstract.
- Table of contents.
- Introduction: A couple of words and facts about autism and followed by a few words of how this report is structured so the person (who is reading this report) can get a clear view of things. The project objectives will be presented.
- Chapter 1 (what is autism): A brief introduction to this chapter.
  - How does autism affect the lives of the individual: Does there exists one or more kinds of autism, what are the symptoms of each one. Maybe put in some examples of behaviour of people with autism and in the end, who is affected by autism?
  - Why do people get autism: What causes autism?
    - What is the spread of autism: what people are more likely to get autism?
  - Is there a cure? : Is it possible to cure autism or at least give the one suffering from autism some kind of help? What kind of help is available e.g. treatments, medications. What help has been giving the best results and so on.
    - Diagnosis: How are people diagnosed?
    - Treatments:
- Chapter 2 (autism in Iceland): A brief introduction to this chapter.
  - What is the situation here in Iceland: Is it similar to other countries or do we use some other methods.
  - Where can people that are affected with autism go and get some help or information about it.
    - What institutions are here in Iceland that handle autism and in what way?
  - Autism environment: How is it to work with people with autism here in Iceland?
    I was thinking of this part of the chapter to contain something from the interview with Unnur.
  - Discussion about the usefulness of the computer-based approach.
• **Chapter 3 (the coding progress):** A brief introduction to this chapter.
  
o A description of the program:
  ▪ How does it work?
  ▪ How is it constructed? (Requirement analysis in form of use case diagrams).
  ▪ Why did I make it exactly this way?
  
o How is this program expected to help people with autism?
  
o Some technical information about the program:
  ▪ UML diagram.
  ▪ Technical description.
  
o What did the experts say? Did I get some comments from an expert?
  
o What can be added to the program or can there be some future work made on this it?

• **Chapter 4 (The design of the system):**
  
o System design.
  
o Architecture: in form of a package diagram.
  
o Class diagram.

• **Chapter 5 (Evaluation):**
  
o Usability: Maybe get a HCI expert to evaluate the program.
  
o Expert walkthrough: Get someone who is an expert on autism to go through the whole program and evaluate it.
  
o User trial: Maybe a controlled experiment (Only if time allows!!).

• **Final words:** A conclusion, some final thoughts about the program and discussion about if it can in fact help people with autism or be some kind of a foundation for a future work in this field.

• **A list of references.**

• **Appendices:**
  
o The code.
  
o Project outline.
  
o Gant chart.
  
o And there is probably some more to come.
Appendix C: Interview core points (main focus is on those points)

The interview was made 14th October 2008 with four teachers who are working in the faculty that is specialised for autistic people in Akureyri

General information about the faculty

- What is the age of the autistic people that are in this faculty?
- What methods do you use for teaching people with autism?
- Is the faculty using any kind of software today?
- Is that software specialised for autistic people?

Use case analysis

- What are the basic requirements for a program that is built with the needs of people with autism?
- What factors should the program focus most on e.g. reinforced behaviour?
- What age group should the program focus most on e.g. children, teenagers?
Appendix D: Project plan
Appendix E: The source code

Class MainWindow

/******************************************************************************
 * Program: MainWindow.java
 * This is a part of the final year project "A computer based approach on autism"
 * for the computer department of the university of Akureyri
 * Author: Ævar. K. Karlsson
 * Date: 17 April 2009
 * The driver program. The program's login GUI interface
 ******************************************************************************

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class MainWindow extends JFrame implements ActionListener
{
    private static final long serialVersionUID = 1L;

    //**********************
    //*   GUI components   *
    //**********************
    private JPanel panelA, northPanel, centerPanel;
    private JPanel leftCenterPanel, rightCenterPanel;
    private JButton student, supervisor;
    private JLabel labelA, labelB;
    private JPasswordField passwordField;
    private static final String REAL_PASSWORD = "jaws";
public static void main(final String []arg) {
    new MainWindow();
}

public MainWindow() {

    // All panels initialized
    panelA = new JPanel();
    northPanel = new JPanel();
    northPanel.setBackground(Color.lightGray);
    centerPanel = new JPanel();
    centerPanel.setBackground(Color.lightGray);
    leftCenterPanel = new JPanel();
    leftCenterPanel.setBackground(Color.lightGray);
    rightCenterPanel = new JPanel();
    rightCenterPanel.setBackground(Color.lightGray);
    this.setTitle("Board-Maker");
    panelA.setLayout(new BorderLayout());

    // GUI components are created
    labelA = new JLabel("Vinsamlegast veldu það sem við á");
    labelB = new JLabel("Aðgangsorð: ");
    passwordField = new JPasswordField(6);
    student = new JButton("Nemandi");
    student.addActionListener(this);
    supervisor = new JButton("Umsjónarmaður");
    supervisor.addActionListener(this);

    // GUI components are added to the panels
    northPanel.add(labelA);
    leftCenterPanel.setPreferredSize(new Dimension(100, 100));
}
leftCenterPanel.add(Box.createRigidArea(new Dimension(0, 110)));
leftCenterPanel.add(student);
rightCenterPanel.setPreferredSize(new Dimension(180, 100));
rightCenterPanel.add(Box.createRigidArea(new Dimension(5, 30)));
rightCenterPanel.add(labelB);
rightCenterPanel.add(passwordField);
rightCenterPanel.add(Box.createRigidArea(new Dimension(30, 40)));
rightCenterPanel.add(supervisor);
centerPanel.add(leftCenterPanel);
centerPanel.add(rightCenterPanel);
panelA.add(northPanel, BorderLayout.NORTH);
panelA.add(centerPanel, BorderLayout.CENTER);

// The size of the JFrame along with other options
setPreferredSize(new Dimension(350, 170));
this.add(panelA);
this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
this.pack();
this.setVisible(true);
this.setResizable(false);
}

//--------------------------------------------------------------------------------------------------------
// Control components
//--------------------------------------------------------------------------------------------------------
public final void actionPerformed(final ActionEvent loginEvent) {
    if (loginEvent.getSource() == student) {


MainController control = new MainController();
StudentInterface sFace = new StudentInterface(control);
control.addObserver(sFace);
}
if (loginEvent.getSource() == supervisor) {
String password = "";
// getPassword() returns password in char array
char [] input = passwordField.getPassword();
// get the total characters of the password
int passwordLength = input.length;
for (int i = 0; i < passwordLength; i++) {
    // Combining the char array into string
    password = password + input[i];
}

// if password is correct
if (password.equals(REAL_PASSWORD)) {
    MainController control = new MainController();
    SupervisorInterface sVFace = new SupervisorInterface(control);
    control.addObserver(sVFace);
} else {
    JOptionPane.showMessageDialog(null,
"Vinsamlegast skráðu inn rétt aðgangsorð", ",",
JOptionPane.ERROR_MESSAGE);
}
}
import java.util.Observable;
import java.util.Observer;
import java.util.*;
import java.io.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class StudentInterface extends JFrame implements Observer, ActionListener {
    private static final long serialVersionUID = 1L;

    // Components are initialized
    private JPanel mainPanel, schedulePanel, schedulePanelA, eastPanel, southPanel, labelPanel;
    private JButton confirmButton, loadButton;
    private JComboBox dayBox;
    private String[] days = {
        "Mánudagur", "Þriðjudagur",
        "Miðvikudagur", "Fimmtudagur",
        "Föstudagur", "Laugardagur", "Sunnudagur"};
private String selected = "Mánudagur";
private JLabel dayLabel, selectedDayLabel, pictureLabel;
private MainController control;
private File loadFile;
private LinkedList<JPanel> tempList = new LinkedList<JPanel>();

public StudentInterface(final MainController tempControl) {
    //******************************
    // individual GUI components    *
    //******************************
    this.control = tempControl;

    dayBox = new JComboBox(days);
    dayBox.addActionListener(this);

    Font timesA = new Font("Times New Roman", Font.BOLD, 18);
    dayLabel = new JLabel(" Dagur: ");
    selectedDayLabel = new JLabel("Mánudagur");
    selectedDayLabel.setFont(timesA);
    pictureLabel = new JLabel("Mynd " + "Númer " + "Nafn");
    pictureLabel.setFont(timesA);

    confirmButton = new JButton("Hætta");
    confirmButton.addActionListener(this);

    loadButton = new JButton("Ná í viku");
    loadButton.addActionListener(this);

    //******************************
```java
// panels and their items
//************************
mainPanel = new JPanel();
mainPanel.setLayout(new BorderLayout());

labelPanel = new JPanel();
labelPanel.setBackground(Color.WHITE);
labelPanel.setPreferredSize(new Dimension(420, 80));
labelPanel.add(pictureLabel);

schedulePanelA = new JPanel();
schedulePanelA.setBackground(Color.WHITE);
//MAX 10 tasks are allowed + the labelPanel
schedulePanelA.setLayout(new GridLayout(11, 1));
schedulePanelA.add(labelPanel);

JScrollPane scrollPanel = new JScrollPane(schedulePanelA);

schedulePanel = new JPanel();
schedulePanel.setBackground(Color.lightGray);
schedulePanel.add(selectedDayLabel);

eastPanel = new JPanel();
eastPanel.setBackground(Color.lightGray);
eastPanel.setPreferredSize(new Dimension(250, 80));
eastPanel.add(Box.createRigidArea(new Dimension(0, 50)));
eastPanel.add(dayLabel);
eastPanel.add(dayBox);

southPanel = new JPanel();
```

southPanel.setBackground(Color.lightGray);
southPanel.setPreferredSize(new Dimension(60, 60));
southPanel.add(confirmButton);
southPanel.add(Box.createRigidArea(new Dimension(80, 0)));
southPanel.add(loadButton);

mainPanel.add(schedulePanel, BorderLayout.NORTH);
mainPanel.add(scrollPanel, BorderLayout.CENTER);
mainPanel.add(eastPanel, BorderLayout.EAST);
mainPanel.add(southPanel, BorderLayout.SOUTH);
setPreferredSize(new Dimension(700, 600));

// The size of the JFrame along with other options
this.setTitle("Vikan mín");
this.add(mainPanel);
this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
this.pack();
this.setVisible(true);
this.setResizable(false);

// set default, starting day as Monday
control.setDay("Mánudagur", 0);
}

//**********************************************************************
//                    Control components
//**********************************************************************
public final void actionPerformed(final ActionEvent loginEvent) {
    if (loginEvent.getSource() == confirmButton) {
        this.dispose();
    }
if (loginEvent.getSource() == loadButton) {   // a schedule is loaded
    JFileChooser chooser = new JFileChooser("D:/Háskólinn "
+ "á Akureyri/Vorönn 2009/LOK0283/The Project/The software");
    int status = chooser.showOpenDialog(null);

    if (status == JFileChooser.APPROVE_OPTION) {
        loadFile = chooser.getSelectedFile();
        dayBox.setSelectedIndex(0);
        control.cleanSelected(1);
        control.readInfo(loadFile);
    }
}

if (loginEvent.getSource() == dayBox) {   // the action
    comboBox
    int day = dayBox.getSelectedIndex();
    switch (day) {
        case 0: selected = "Mánudagur"; break;
        case 1: selected = "Þriðjudagur"; break;
        case 2: selected = "Miðvikudagur"; break;
        case 3: selected = "Fimmtudagur"; break;
        case 4: selected = "Föstudagur"; break;
        case 5: selected = "Laugardagur"; break;
    }
case 6: selected = "Sunnudagur";
break;
default:
break;
}
control.setDay(selected, day);
control.printDay();
}

// this method updates the GUI whenever a change occurs
public final void update(final Observable observable, final Object arg1) {
    String day = control.getDay();
    selectedDayLabel.setText(day);

tempList = null;
tempList = control.getListOfPanels();

    schedulePanelA.removeAll();
schedulePanelA.add(labelPanel);

    for (int i = 0; i < tempList.size(); i++) {
        schedulePanelA.add(tempList.get(i));
    }
schedulePanelA.repaint();
schedulePanelA.revalidate();

this.setTitle("Skipuleggja viku: " + loadFile);
}
Class SupervisorInterface

//*****************************************************************************
//  Program: SupervisorInterface.java
//  This is a part of the final year project "A computer based approach on autism" for the computer department of the university of Akureyri
//  Author: Ævar. K. Karlsson
//  Date: 17 April 2009
//  The program's Supervisor GUI
//*****************************************************************************

import java.util.Observable;
import java.util.Observer;
import java.util.*;
import java.io.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class SupervisorInterface extends JFrame implements Observer, ActionListener {
    
    private static final long serialVersionUID = 1L;

    // Components are initialized
    private JPanel mainPanel, schedulePanel, schedulePanelA, eastPanel, southPanel;
    private JPanel deletePanel, cleanPanel, labelPanel;
    private JButton confirmButton, loadButton, saveButton;
    private JButton insertButton, deleteButton, eraseButton;
    private JTextField insertField;
}
private JComboBox dayBox, pictureBox, deleteBox, cleanBox;
private String[] days = {"Mánudagur", "Þriðjudagur", "Miðvikudagur", "Fimmtudagur", "Föstudagur", "Laugardagur", "Sunnudagur"};
private String[] clean = {"Dag", "Viku"};
private String[] numbers = {"1", "2", "3", "4", "5", "6", "7", "8", "9", "10"};
private String selected = "Mánudagur";
private JLabel dayLabel, insertLabel, deleteLabel, cleanLabel, selectedDayLabel, pictureBox;
private JLabel taskNameLabel, pictureNameLabel;
private File loadFile;
private MainController control;
private LinkedList<JPanel> tempList = new LinkedList<JPanel>();

public SupervisorInterface(final MainController tempControl) {
    //**************************************************************
    //*   individual GUI components   *
    //**************************************************************
    this.control = tempControl;
    dayBox = new JComboBox(days);
    dayBox.addActionListener(this);
    pictureBox = new JComboBox(picture);
    deleteBox = new JComboBox(numbers);
    cleanBox = new JComboBox(clean);
Font times = new Font("Times New Roman", Font.BOLD, 17);
Font timesA = new Font("Times New Roman", Font.BOLD, 18);

dayLabel = new JLabel(" Dagur: ");
insertLabel = new JLabel("Nýtt verkefni");
insertLabel.setFont(times);
deleteLabel = new JLabel("Eyða verkefni");
deleteLabel.setFont(times);
cleanLabel = new JLabel("Stroka út");
cleanLabel.setFont(times);
selectedDayLabel = new JLabel("Mánudagur");
selectedDayLabel.setFont(timesA);
pictureLabel = new JLabel("Mynd
 + "Númer
 Nafn");
pictureLabel.setFont(timesA);
taskNameLabel = new JLabel("Nafn:");
pictureNameLabel = new JLabel("Mynd:");

confirmButton = new JButton("Hætta");
confirmButton.addActionListener(this);

loadButton = new JButton("Ná í viku");
loadButton.addActionListener(this);

saveButton = new JButton("Vista viku");
saveButton.addActionListener(this);

insertButton = new JButton("Búa til");
insertButton.addActionListener(this);

deleteButton = new JButton("Eyða");
deleteButton.addActionListener(this);

eraseButton = new JButton("Stroka út");
eraseButton.addActionListener(this);

insertField = new JTextField(10);

//*****************************/
//*   panels and their items  *
//*****************************
mainPanel = new JPanel();
mainPanel.setLayout(new BorderLayout());

labelPanel = new JPanel();
labelPanel.setBackground(Color.WHITE);
labelPanel.setPreferredSize(new Dimension(420, 80));
labelPanel.add(pictureLabel);

schedulePanelA = new JPanel();
schedulePanelA.setBackground(Color.WHITE);
//MAX 10 tasks are allowed + the labelPanel
schedulePanelA.setLayout(new GridLayout(11, 1));
schedulePanelA.add(labelPanel);

schedulePanel = new JPanel();
schedulePanel.setBackground(Color.lightGray);
schedulePanel.add(selectedDayLabel);

JScrollPane scrollPanel = new JScrollPane(schedulePanelA);

deletePanel = new JPanel();
deletePanel.setBackground(Color.lightGray);
displayPanel.setPreferredSize(new Dimension(250, 100));
deletePanel.add(deleteLabel);
displayPanel.add(Box.createRigidArea(new Dimension(110, 30)));
deletePanel.add(deleteButton);
displayPanel.add(Box.createRigidArea(new Dimension(90, 0)));
displayPanel.add(deleteBox);

cleanPanel = new JPanel();
cleanPanel.setBackground(Color.lightGray);
cleanPanel.setPreferredSize(new Dimension(250, 100));
cleanPanel.add(cleanLabel);
cleanPanel.add(Box.createRigidArea(new Dimension(110, 30)));
cleanPanel.add(eraseButton);
cleanPanel.add(Box.createRigidArea(new Dimension(40, 0)));
cleanPanel.add(cleanBox);

eastPanel = new JPanel();
eastPanel.setBackground(Color.lightGray);
eastPanel.setPreferredSize(new Dimension(250, 80));
eastPanel.add(Box.createRigidArea(new Dimension(0, 50)));
eastPanel.add(dayLabel);
eastPanel.add(dayBox);
eastPanel.add(Box.createRigidArea(new Dimension(40, 70)));
eastPanel.add(insertLabel);
eastPanel.add(Box.createRigidArea(new Dimension(140, 0))){}
eastPanel.add(pictureNameLabel);
eastPanel.add(Box.createRigidArea(new Dimension(30, 0)));
eastPanel.add(pictureBox);
eastPanel.add(Box.createRigidArea(new Dimension(30, 30)));
eastPanel.add(taskNameLabel);
eastPanel.add(Box.createRigidArea(new Dimension(20, 0)));
eastPanel.add(insertField);
eastPanel.add(Box.createRigidArea(new Dimension(40, 30)));
eastPanel.add(insertButton);
eastPanel.add(Box.createRigidArea(new Dimension(130, 50)));
eastPanel.add(deletePanel);
eastPanel.add(cleanPanel);

southPanel = new JPanel();
southPanel.setBackground(Color.lightGray);
southPanel.setPreferredSize(new Dimension(60, 60));
southPanel.add(confirmButton);
southPanel.add(Box.createRigidArea(new Dimension(80, 0)));
southPanel.add(loadButton);
southPanel.add(Box.createRigidArea(new Dimension(80, 0)));
southPanel.add(saveButton);

mainPanel.add(schedulePanel, BorderLayout.NORTH);
mainPanel.add(scrollPanel, BorderLayout.CENTER);
mainPanel.add(eastPanel, BorderLayout.EAST);
mainPanel.add(southPanel, BorderLayout.SOUTH);
setPreferredSize(new Dimension(700, 600));

// The size of the JFrame along with other options
this.setTitle("Skipuleggja viku");
this.add(mainPanel);
this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
this.pack();
this.setVisible(true);
this.setResizable(false);

// set default, starting day as Monday
control.setDay("Mánudagur", 0);

}//********************************************
//                    Control components
//********************************************
public final void actionPerformed(final ActionEvent loginEvent) {
    if (loginEvent.getSource() == confirmButton) {
        this.dispose();
    }
    if (loginEvent.getSource() == loadButton) { // a schedule is loaded
        JFileChooser chooser = new JFileChooser("D:/Háskólinn á Akureyri" + "/Vorönn 2009/LOK0283/The Project/The software");
        int status = chooser.showOpenDialog(null);

        if (status == JFileChooser.APPROVE_OPTION) {
            loadFile = chooser.getSelectedFile();
        }
dayBox.setSelectedIndex(0);
// the existing schedule is deleted before a new
one is loaded
control.cleanSelected(1);
control.readInfo(loadFile);
}
}
// the schedule is being saved
if (loginEvent.getSource() == saveButton) {
    String fileName =
        JOptionPane.showInputDialog(null,
"Vinsamlegast gefðu vikunni nafn.",
"Vista viku", JOptionPane.QUESTION_MESSAGE);
    if (fileName.trim().equals("") ) {
        JOptionPane.showMessageDialog(null,
"Þú verður að gefa vikunni þínni nafn!", ",",
        JOptionPane.ERROR_MESSAGE);
    } else {
        try {
            // Only acceptable filenames are saved
            control.saveFile(fileName);
            JOptionPane.showMessageDialog(null,
"Skráin þín hefur verið vistuð.", ",",
            JOptionPane.INFORMATION_MESSAGE);
        } catch (IOException e) {
        }
    }
}
}
if (loginEvent.getSource() == insertButton) { // task
    String taskName = insertField.getText();
    if (taskName.trim().equals("") ) {

JOptionPane.showMessageDialog(null, "Þú getur ekki búið til tómt verkefni!", ", ", JOptionPane.ERROR_MESSAGE);

} else {
    if (taskName.trim().length() > 20) {
        JOptionPane.showMessageDialog(null, "Hvert verkefni getur aðeins haft 20 stafi að lengd!", ", ", 
        JOptionPane.ERROR_MESSAGE);
    } else if (control.countTasks() == 10) {
        JOptionPane.showMessageDialog(null, "Dagurinn er yfirfullur!", ", ", 
        JOptionPane.ERROR_MESSAGE);
    } else {
        int id = control.countTasks();
        String picUrl = "";
        switch (pictureBox.getSelectedIndex()) {
            case 0: picUrl = "myndir/none.jpg";
            break;
            case 1: picUrl = "myndir/brush_teeth.jpg";
            break;
            case 2: picUrl = "myndir/sleep.jpg";
            break;
            case 3: picUrl = "myndir/bus.jpg";
            break;
            case 4: picUrl = "myndir/homework.jpg";
            break;
            case 5: picUrl = "myndir/car.jpg";
            break;
            case 6: picUrl = "myndir/cereal.jpg";
            break;
        }
    }
}
case 7: picUrl = "myndir/family.jpg";
break;
case 8: picUrl = "myndir/bathroom.jpg";
break;
case 9: picUrl = "myndir/kitchen.jpg";
break;
case 10: picUrl = "myndir/playRoom.jpg";
break;
case 11: picUrl = "myndir/þvo_hendur.jpg";
break;
default:
    break;
}
// only acceptable task names are created
control.insertTask(taskName, id, picUrl);
}
//the whole day/week is cleaned
if (loginEvent.getSource() == eraseButton) {
    int taskNumber = cleanBox.getSelectedIndex();
    control.cleanSelected(taskNumber);
}

if (loginEvent.getSource() == dayBox) {    // the action
    int day = dayBox.getSelectedIndex();

    switch(day) {
        case 0: selected = "Mánudagur"; break;
        case 1: selected = "Þriðjudagur"; break;
        case 2: selected = "Miðvikudagur"; break;
        case 3: selected = "Fimmtudagur"; break;
        case 4: selected = "Föstudagur"; break;
        case 5: selected = "Laugardagur"; break;
        case 6: selected = "Sunnudagur"; break;
        default: break;
    }

    control.setDay(selected, day);
    control.printDay();
}
}
// this method updates the GUI whenever a change occurs
public final void update(final Observable observable, final Object arg1) {
    String day = control.getDay();
    selectedDayLabel.setText(day);

    tempList = null;
    tempList = control.getListOfPanels();

    schedulePanelA.removeAll();
    schedulePanelA.add(labelPanel);

    for (int i = 0; i < tempList.size(); i++) {
        schedulePanelA.add(tempList.get(i));
    }
    insertField.setText(""");
    schedulePanelA.repaint();
    schedulePanelA.revalidate();

    this.setTitle("Skipuleggja viku:   " + loadFile);
}
Class MainController

//******************************************************
// Program: MainController.java
// This is a part of the final year project "A computer
// based approach on autism" for the computer department
// of the university of Akureyri
// Author: Ævar. K. Karlsson
// Date: 17 April 2009
// This class has all the controls and stores all tasks
//******************************************************
import java.util.*;
import javax.swing.*;
import java.io.*;
import java.awt.*;

public class MainController extends Observable {
    private LinkedList < Task > MondayList = new LinkedList < Task >();
    private LinkedList < Task > TuesdayList = new LinkedList < Task >();
    private LinkedList < Task > WednesdayList = new LinkedList < Task >();
    private LinkedList < Task > ThursdayList = new LinkedList < Task >();
    private LinkedList < Task > FridayList = new LinkedList < Task >();
    private LinkedList < Task > SaturdayList = new LinkedList < Task >();
    private LinkedList < Task > SundayList = new LinkedList < Task >();
}
private LinkedList < Task > SelectedList = new LinkedList < Task >();

private LinkedList < JPanel > ListOfPanels = new LinkedList < JPanel >();

private JPanel taskPanel;
private JLabel pictureLabel, idLabel, taskLabel;
private Task TaskObject;
private String selected_day;
private int dayNumber, previousDay = 0;
private String whatDay = "";

//                        The constructor                        
// ****************************************************************

public MainController() {
 ...
}

// ************************************************************************
// *                       Getters & setters                        *
// ************************************************************************

public final void setDay(final String tempDay, final int tempDayNumber) {
    selected_day = tempDay;
    this.dayNumber = tempDayNumber;
    controlDays();
    printDay();
    setChanged();
    notifyObservers();
}

public final String getDay() {
    return selected_day;
}
public final int countTasks() {
    int numbers = SelectedList.size();
    return numbers;
}

public final LinkedList<JPanel> getListOfPanels() {
    returnListOfPanels;
}

// *******************************************************
// *    The controls for the switching of the lists      *
// ********************
// *******************************************************
public final void controlDays() {
    switch(previousDay) {
        case 0: MondayList = SelectedList;
               break;
        case 1: TuesdayList = SelectedList;
               break;
        case 2: WednesdayList = SelectedList;
               break;
        case 3: ThursdayList = SelectedList;
               break;
        case 4: FridayList = SelectedList;
               break;
        case 5: SaturdayList = SelectedList;
               break;
        case 6: SundayList = SelectedList;
               break;
        default:
               break;
    }
    switch(dayNumber) {
case 0: SelectedList = MondayList;
    previousDay = dayNumber;
break;
case 1: SelectedList = TuesdayList;
    previousDay = dayNumber;
break;
case 2: SelectedList = WednesdayList;
    previousDay = dayNumber;
break;
case 3: SelectedList = ThursdayList;
    previousDay = dayNumber;
break;
case 4: SelectedList = FridayList;
    previousDay = dayNumber;
break;
case 5: SelectedList = SaturdayList;
    previousDay = dayNumber;
break;
case 6: SelectedList = SundayList;
    previousDay = dayNumber;
break;
default:
    break;
}
// ********************
// *        A new task is created         *
// ********************

public final void insertTask(String taskName, int id, String picUrl) {
    Task TaskObject = new Task(taskName, id, picUrl);
    SelectedList.add(TaskObject);
printDay();
setChanged();
notifyObservers();
}

// ****************************************
// *    The selected day is printed      *
// ****************************************

public final void printDay() {
    Font times = new Font("Times New Roman", Font.BOLD, 15);
    String idString;
    ListOfPanels.clear();

    for (int a = 0; a < SelectedList.size(); a++) {
        ImageIcon icon = new ImageIcon(SelectedList.get(a).getTaskPicUrl());
        pictureLabel = new JLabel(icon);
        idString = Integer.toString((SelectedList.get(a).getTaskID() + 1));
        idLabel = new JLabel(idString);
        idLabel.setFont(times);
        taskLabel = new JLabel(SelectedList.get(a).getTaskName());
        taskLabel.setFont(times);

        taskPanel = new JPanel();
        taskPanel.add(pictureLabel);
        taskPanel.add(Box.createRigidArea(new Dimension(60, 0)));
        taskPanel.add(idLabel);
        taskPanel.add(Box.createRigidArea(new Dimension(60, 0)));
        taskPanel.add(taskLabel);
    }
}
taskPanel.setPreferredSize(new Dimension(420, 150));
taskPanel.setBackgroundColor(Color.WHITE);

ListOfPanels.add(taskPanel);
}

// ****************************
// *    A task is deleted    *
// ****************************

public final void deleteTask(int id) {
    TaskObject = SelectedList.get(id);
    SelectedList.remove(TaskObject);
    TaskObject = null;
    for (int counterA = id; counterA < SelectedList.size(); counterA++) {
        SelectedList.get(counterA).setTaskID(id);
        id++;
    }
    printDay();
    setChanged();
    notifyObservers();
}

// ****************************
// *    Cleaning method       *
// ****************************

public final void cleanSelected(final int taskNumber) {
    SelectedList.clear();
    if (taskNumber == 1) {
        MondayList.clear();
        TuesdayList.clear();
        WednesdayList.clear();
        ThursdayList.clear();
    }
FridayList.clear();
SaturdayList.clear();
SundayList.clear();
}
printDay();
setChanged();
notifyObservers();
}
// ******************************************************************************
// * A file is loaded/read   *
// ******************************************************************************
public final void readInfo(final File loadFile) {
    try {
        Scanner fileScan = new Scanner(loadFile);
        whatDay = "";
        while (fileScan.hasNext()) {
            String currentString = fileScan.nextLine();
            if (currentString.equals("NEWDAY")) {
                whatDay = fileScan.nextLine();
            } else {
                fillArray(fileScan, currentString,
                        whatDay);
            }
        }
        printDay();
        setChanged();
        notifyObservers();
    } catch (IOException e) {
        JOptionPane.showMessageDialog(null, "Skráin er ekki til!", ", ",
                JOptionPane.ERROR_MESSAGE);
    }
}
public final void fillArray(final Scanner fileScan, final String currentString, final String whatDay) {
    String url = currentString;
    String idString = fileScan.nextLine();
    int id = Integer.parseInt(idString);
    String fullTask = fileScan.nextLine();
    TaskObject = new Task(fullTask, id, url);
    if (whatDay.equals("Monday")) {
        MondayList.add(TaskObject);
        printDay();
    }
    if (whatDay.equals("Tuesday")) {
        TuesdayList.add(TaskObject);
        printDay();
    }
    if (whatDay.equals("Wednesday")) {
        WednesdayList.add(TaskObject);
        printDay();
    }
    if (whatDay.equals("Thursday")) {
        ThursdayList.add(TaskObject);
        printDay();
    }
    if (whatDay.equals("Friday")) {
        FridayList.add(TaskObject);
        printDay();
    }
    if (whatDay.equals("Saturday")) {
        SaturdayList.add(TaskObject);
        printDay();
    }
}
if (whatDay.equals("Sunday")) {
    SundayList.add(TaskObject);
    printDay();
}
}

// **********************************
// *         A schedule is saved   *
// **********************************

public final void saveFile(String fileName) throws IOException {
    if (fileName != null) {
        fileName += ".txt";
        String newline = System.getProperty("line.separator");
        String day = "";
        Writer output = new BufferedWriter(new FileWriter(fileName));
        try {
            int prevDay = dayNumber;
            for (int counterA = 0; counterA < 7; counterA++) {

                switch (counterA) {
                    case 0: day = "Monday";
                        SelectedList = MondayList;
                        break;
                    case 1: day = "Tuesday";
                        SelectedList = TuesdayList;
                        break;
                    case 2: day = "Wednesday";
                        SelectedList = WednesdayList;
                        break;
                }
            }
        } finally {
            output.close();
        }
    }
}
case 3: day = "Thursday";
    SelectedList = ThursdayList;
    break;

    case 4: day = "Friday";
    SelectedList = FridayList;
    break;

    case 5: day = "Saturday";
    SelectedList = SaturdayList;
    break;

    case 6: day = "Sunday";
    SelectedList = SundayList;
    break;

    default:
    break;

    }

    output.write("NEWDAY");
    output.write(newline);
    output.write(day);
    output.write(newline);

    for (int counterB = 0; counterB <
    SelectedList.size(); counterB++) {

        output.write(SelectedList.get(counterB).getTaskPicUrl());
            output.write(newline);
        int id =
    SelectedList.get(counterB).getTaskID();
            String stringID = Integer.toString(id);
        output.write(stringID);
        output.write(newline);

        output.write(SelectedList.get(counterB).getTaskName());
output.write(newline);
}
}
output.close();

// this part is done to maintain the schedule after doing the saving process

switch (prevDay) {
    case 0: day = "Monday";
        SelectedList = MondayList;
        break;
    case 1: day = "Tuesday";
        SelectedList = TuesdayList;
        break;
    case 2: day = "Wednesday";
        SelectedList = WednesdayList;
        break;
    case 3: day = "Thursday";
        SelectedList = ThursdayList;
        break;
    case 4: day = "Friday";
        SelectedList = FridayList;
        break;
    case 5: day = "Saturday";
        SelectedList = SaturdayList;
        break;
    case 6: day = "Sunday";
        SelectedList = SundayList;
        break;
    default:
        break;
}
} catch (IOException e) {
JOptionPane.showMessageDialog(null,
"Villa kom þegar verið var að vista vikuna!",
"", JOptionPane.ERROR_MESSAGE);
}
public class Task {
    private int id;
    private String name, picUrl;

    //********************************************************
    //  Program: Task.java
    //  This is a part of the final year project "A computer
    //  based approach on autism" for the computer department
    //  of the university of Akureyri
    //  Author: Ævar. K. Karlsson
    //  Date: 17 April 2009
    //  This class represents an object of a task
    //  that is created for each day in the week
    //********************************************************

    public Task(final String tempName, final int tempID, final String tempUrl) {
        this.name = tempName;
        this.id = tempID;
        this.picUrl = tempUrl;
    }

    public final String getTaskName() {
        // Getters & setters
    }
}
public final int getTaskID() {
    return id;
}

public final void setTaskID(final int tempID) {
    this.id = tempID;
}

public final String getTaskPicUrl() {
    return picUrl;
}
Appendix F: Special thanks to

**Páll. Rúnar. Gíslason:** For black-box testing the software.

**Rósa. Dögg. Jónsdóttir:** For black-box testing the software.

**The teachers at the autism department at Síðuskóli:** For helping me build the system requirements and evaluating my software.

**Unnar. Freyr. Hlynsson:** For looking at my code when I was completely stuck.

**To the rest of the Computer Science Department of the University of Akureyri:** For just being, who they are.