CENG491

REQUIREMENT ANALYSIS REPORT

ANKA YAZILIM

November 2005

Aysun BAŞÇETİNÇELİK
C. ACAR ERKEK
Çağıl ÖZTÜRK
Mennan GÜDER

TABLE OF CONTENTS

1. INTRODUCTION 3
1.1. GOALS AND OBJECTIVES 3
1.2. STATEMENT OF SCOPE 3
1.3. CONSTRAINTS 5
2. PROJECT PLAN 6
2.1. PROCESS MODEL 6
2.2. TEAM ORGANIZATION 6
2.3. SCHEDULE 7
3. RESEARCH 7
3.1. MEETINGS 7
3.1.1. Asst.Prof.Dr. H. Fatoş (GÜR) AKINOĞLU 7
3.1.2. Assoc.Prof.Dr. Veysi İşler 8
3.1.3. Assoc.Prof.Dr. Kürşat CAĞILTAY 9
3.1.4. Prof.Dr.M.Yaşar ÖZDEN 9
3.2. ARTICLES 9
3.3. CURRENT SYSTEMS 9
4. PROJECT SPECIFICATIONS AND USE CASES 10
4.1. USER PROFILES 10
4.2. EDITOR MODE 12
4.2.1. Map Editor 12
4.2.2. Animation Editor 14
4.2.3. Game Editor 16
4.3. VIEWER MODE 18
4.3.1. Watching Animations 18
4.3.2. Playing Games 19
5. DATA MODELING 21
5.1. DATA FLOW DIAGRAM LEVEL 0 21
5.2. DATA FLOW DIAGRAM LEVEL 1 22
5.2.1. Processes 23
6. HARDWARE AND SOFTWARE REQUIREMENTS 24
6.1. INSTALLATION ENVIRONMENT 24

6.2. DEVELOPMENT ENVIRONMENT 24

7. CONCLUSION 26

8. APPENDIX 26

8.1. GANTT CHART 26

1. INTRODUCTION

In this document we will examine the educational perspective of our project topic, describe the final product that we will produce. Specify the requirements. Present the results of our researches. This document would also be a guide for us to make our design.

1.1. GOALS AND OBJECTIVES

It is known that traffic education is very important. Every year many people die because of traffic accidents and much money is wasted in those accidents. It is especially important to start traffic education from early ages. We are not the only people thinking this, thus in primary schools there is a “Traffic and First Aid” course.

This year, we will work on building a 3D educational software tool. We think that there is not a sufficient tool in use which responds to the needs of primary school teachers whose branch is traffic (Our researches and meetings also brought us to that point). Our product will meet with teachers’ expectations and it will be used to teach the correct behaviors of pedestrians’ especially for primary school children.

1.2. STATEMENT OF SCOPE

Educators want to measure students’ reactions in a real situation and see their behaviors. They also want to visualize real events for students. According to those our software tool will provide:

- An effective communication between educator and student by visualizing the course subject with 3D realistic environment.
- Events which will be defined by teachers.
- Two main capabilities:
    - Animation designing
    - Game designing

Those designs would be done by two main editors and would be very simple to use.

Educator could plan an event; she can design it as an animation or game. Set it up on an environment which we will provide her. The environment would not be fixed or it would not force the educator to design a new map. There would be some predefined maps. Limited changes on map could be done dynamically. Vehicles, people, or other movable objects will be assigned by user (as default they would be in application control). Those movables will have predefined Artificial Intelligence (AI). However, their predefined behavior could be changed independently. Also, teacher can state questions at a determined time and can get feedbacks.

A student could use our software, depending on his teacher's design, both as an animation viewer and a player for games. During animations, besides viewing the animation he will answer some integrated questions. These questions will also be prepared by the teacher. In addition to this, the student will be able to play games that teacher designed. He will get some messages within the game according to the choices he made.

Educational benefits:

- Visualization
- Safe conjectural traffic environment
- Teaching basic traffic rules
- Gaining correct behaviors
- Stiffen knowledge by repeating condition
- Increase participation

Focused technical points:

- Educational benefits
- 3D animation making
- 3D graphics engine and other required engines
- AI
- OpenGL
- Hardware and software requirements as summary

1.3. CONSTRAINTS

During this project we will not be able to do all of the things we want because of some constraints. These constraints can be listed as:

- *Experience Constraint:*

  We have not done such a project before that we are not sure of the situations we will face during our project. This will be the first time that we will interact with customer to learn their needs about the product we will make. We are not so sure how should we react to their needs.

  Moreover, there will be tools that we will have to use. In spite of the fact that we choose these tools, none of us has any experience for these tools. We will learn how to use them from scratch. This also limits us. Although we have some interesting ideas about the project, we are not sure whether we can do all of them so that our specifications are limited to the estimations of our capability.

- *Time Constraint:*

  We have limited time and fixed deadlines that we are not so flexible about this. These dates were planned before our project that it also limits our ideas. Different from a software company workers working fully on a project, we have other projects for different courses that we should manage them concurrently. Unfortunately, we will not be able to focus entirely on this project only.

- *Funding Constraint:*

  We will not be able to spend much money for this project. Because of the funding constraint, we try to prefer free software tools to use in our project or tools that are free for educational purposes.

- *User Constraint:*

  As users of our program are teachers and children at primary schools, our interface should be very simple. It may not be much problem with children using that software as they will not need to do complex things but for teachers, as they will need to use an editor, we should make the simplest one for them. We will not be able to train them and they will make animations and games using it. Because of that constraint, we had to eliminate some functions of our program.

2. PROJECT PLAN

After defining our scope of the project, we have also made some decisions about our project plan that it will help us to predict our next steps. Firstly, we have selected our process model parallel to the course schedule. Secondly, we have decided on the team organization. Then, we have come up with a rough schedule which can evolve according to our work in further weeks.

2.1. PROCESS MODEL

We have not chosen one model for our project that we combined a few models to satisfy our needs for the project. We have chosen a linear model till the prototype of our program and after that we will continue within an evolutionary model. After deciding on the requirements, we will focus ourselves to the design. Making a detailed design, we will prepare a prototype which will show some basic functions of our program. In the implementation phase we are thinking of using an evolutionary model that will include both coding and testing to show our mistakes or problems during implementation. We have chosen such a model because we have not done any graphics project before and we are not sure of our capabilities.

2.2. TEAM ORGANIZATION

As our team is composed of four people we have not done any specific division of labor. Everybody will work during all phases and everybody will make both documentation and coding. We have not chosen any leader that everybody can take this position in the group according to the needs. During weekly meetings, the work which members will do that week is decided depending on both suitable times and interests. This organization can be named as a democratic decentralized one.

As it is stated before, we are meeting at least once in a week and communicate both from internet and telephone. We have made a mail group from Yahoo! Groups to keep our files and communicate through. Everybody is responsible for following this group. Moreover, we have a website that we put the latest versions of our reports. (http://www.ceng.metu.edu.tr/~e1297522/ANKA/)

2.3. SCHEDULE

We have limited time that this project must be ended by June 2006. During this timeline we will go through every step. On January 17th, 2005 we will have released our prototype. Our Gantt chart showing these steps is at *Appendix 8.1*. This chart is subject to change according to our needs. We have not decided on the implementation phase till we know the details.

3. RESEARCH

In order to define the properties of our product in a detailed manner, we have performed mainly three research activities:

1. Meeting with professional people who can enhance our technical, systematic and pedagogic view
2. Examining articles which are helpful to reinforce the educational background of our product
3. Analysing current systems

3.1. MEETINGS

The meetings were done with the following professionals whose experience areas are:

- *Asst.Prof.Dr. H. Fatoş (GÜR) AKINOĞLU :* Traffic education and training
- *Assoc.Prof.Dr. Veysi İşler :* Modeling and simulation, traffic and road safety applications
- *Assoc.Prof.Dr. Kürşat CAĞILTAY:* Games for teaching
- *Prof. Dr. M. Yaşar ÖZDEN:* Instructional Technologies

3.1.1. Asst.Prof.Dr. H. Fatoş (GÜR) AKINOĞLU

Asst.Prof.Dr. H. Fatoş (GÜR) AKINOĞLU is a faculty member at Gazi Univetsity, traffic department. As a result of the meetings hold with Dr. Fatoş Akınoğlu, so many aspects of the project become more definite. These aspects are:

- Taking feedback which includes number and the type of the errors that students do is an important feature for a teacher in order to evaluate both the student and the learning activity.
- The most important problem of traffic education in current education system is that the level of detail that must be given to students is not well defined. In order to not to have the same problem, we have to consider the detail we will include, in a pedagogical manner. For example, there are big differences between secondary school and preschool students. While two color traffic lamps must be learned by preschool student, three color traffic lamps must be learned by the secondary school student.
- Students do not have to be only pedestrians, they are also drivers. For example there are rules for students while they are driving bicycles.
- Using a song related to traffic, during the activity of the student can enhance the motivation of the student.
- Competition can be used for better motivation since racing with others has an important effect on students. Thus, in order to enhance the learning activity, there can be a bonus mechanism. For example, if student has enough bonuses, he or she can drive bicycle in the environment.
- Definition of traffic is one of the most important aspects that must be learned by the student.

3.1.2. Assoc.Prof.Dr. Veysi İşler

Asst.Prof.Dr. H. Fatoş (GÜR) AKINOĞLU is a faculty member at Middle East Technical University, computer engineering department. Since he has experience in modeling, simulation and traffic applications, the meeting that we hold improved our view about both the subject and the technical details. His suggestions are:

- Property of on-line playable games with a multi-user choice can be added. However, since that is far from the educational manner, more related to the entertainment, we decided that it is not a basic property of our system.
- Using Renderware or Blender3d can be a good choice.
- Kids-plus has similar products related to the traffic education, examining can be helpful.

3.1.3. Assoc.Prof.Dr. Kürşat CAĞILTAY

Asst.Prof.Dr. H. Fatoş (GÜR) AKINOĞLU is a faculty member at Middle East Technical University, computer education and instructional technology department.

His main area is games for teaching. His suggestions are:

- He gave examples about his students' early works.
- He also advised using Renderware.
- He suggested some subjects for traffic education and other educational software before our decision about the subject.

3.1.4. Prof.Dr.M.Yaşar ÖZDEN

Prof.Dr.M.Yaşar ÖZDEN is chairmen of computer education and instructional technology department at Middle East Technical University. His suggestions are:

- Meeting with Dr. Fatoş Akınoğlu who is experienced in traffic training.
- The final tool must be working on the windows environment since the target is probably using windows.

3.2. ARTICLES

By examining articles written by Guney Akinoglu, Mustafa Candir, Cumhur Aydin and Ömer Kilic educational background of our project reinforced. The general considerations in the articles are:

- The current education systems expects student to know all the details about the traffic. For example a student is expected to know nearly all traffic rules.
- No interactive environment is available in traffic education.
- Traffic education programs must be continuous and must be able to provide students with the required skills, attitude and knowledge.
- Children are the most affected part of the community from accidents
- Because of the fact that educating children is easier than adults, traffic education must start at preschool.

3.3. CURRENT SYSTEMS

We have not found many similar systems to the one we are thinking about to make. We have found many driving simulator programs but none of them were for children. There are a few web pages for children to teach traffic but not much software programs. In Turkey, there is not much software for educational purposes interested in the area of traffic for children. We have found software from Halıcı Yazılımevi which is for driving schools.

Kids Plus

The only product we found which is for children is “Kids Plus Traffic and First Aid Education for Primary Schools”. It is made by Mobilsoft. It is an old product and only works in old operating systems like MS Windows 95/98. Thus, it was a problem for us to make this software work. It was not updated afterwards.

Main properties of the product which are written at the back of the package are:

- It includes both traffic and first aid education
- 3D simulations are used
- There are games related to each subjects of traffic courses in primary and secondary school level.
- There are test questions
- A help and print out property is available

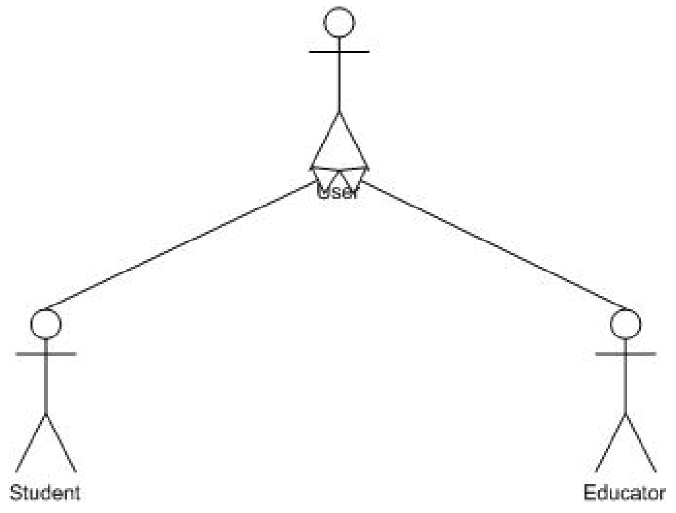
According to our examinations of the program, although it is said that 3D animations are used, they are not used in an effective manner. Only the town and a few objects were designed in 3D. However, modeling of the town was really good and attractive for a child. It is mainly focused on teaching the subjects in the book. Moreover, games that are said to be related are not very well thought. We do not think that a child needs to know every pieces of a bicycle.

4. PROJECT SPECIFICATIONS AND USE CASES

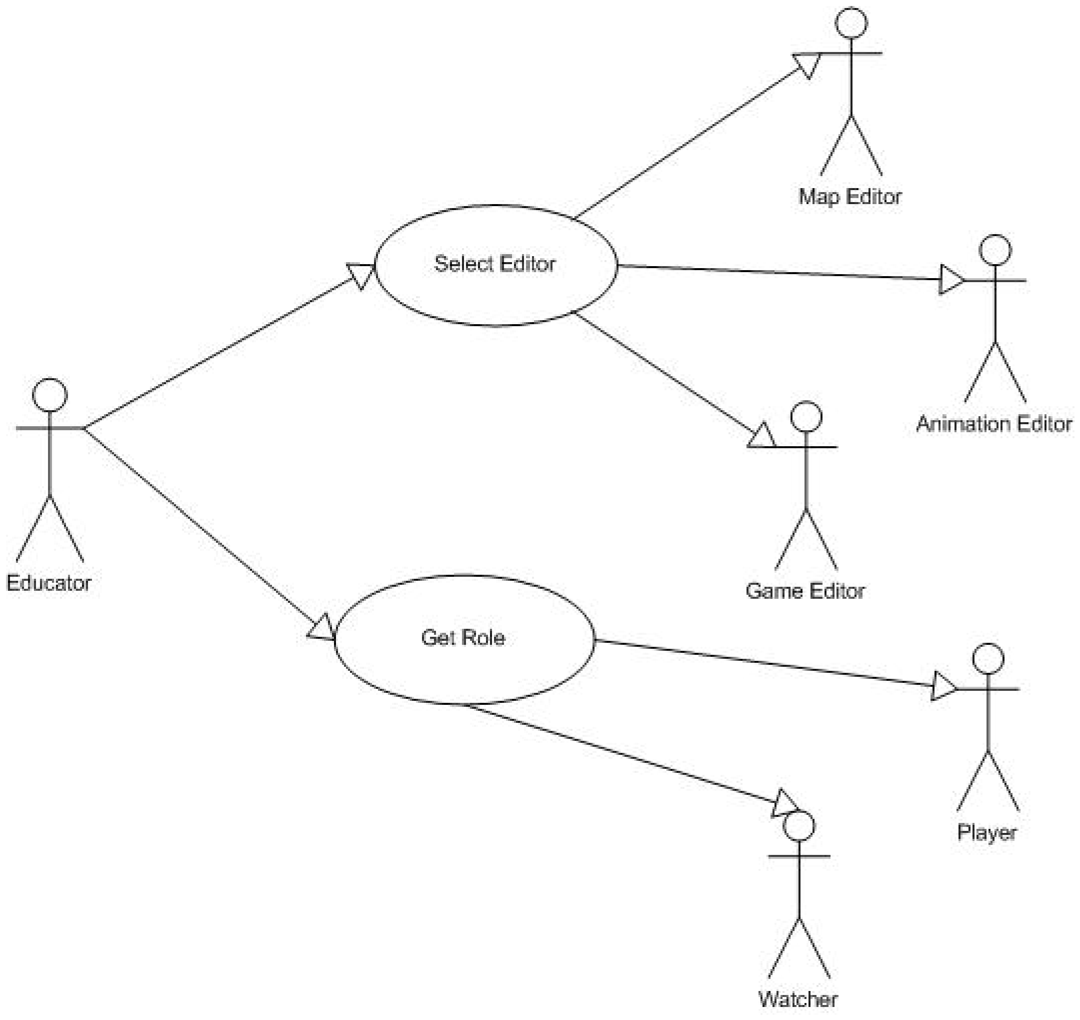
4.1. USER PROFILES

Our software is intended for two potential user groups. These groups are educators and students. Educators are users who want to teach traffic to students by our software (for example primary school teachers, traffic instructors, parents, etc.).

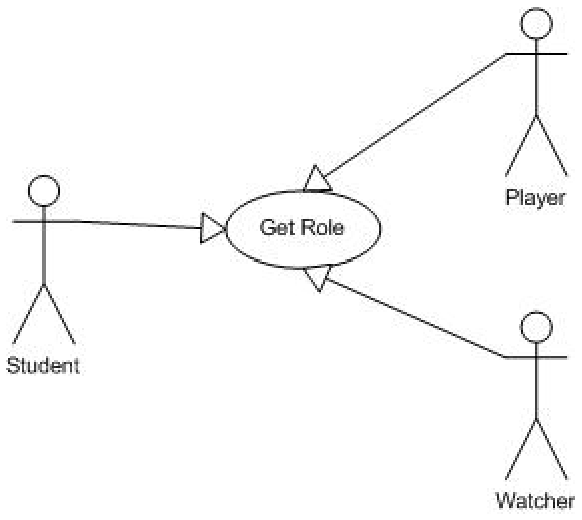
Students are the ones who will learn something by our program (any kind of traffic learners, but our program is intended for primary school students especially).



Educators will use our program in editor mode. They will prepare necessary content for speeding up the education process. These contents will be animations and games.



Students will use our program in viewer mode. They will use the content which was prepared by their educator, namely watching animations and playing games.



4.2. EDITOR MODE

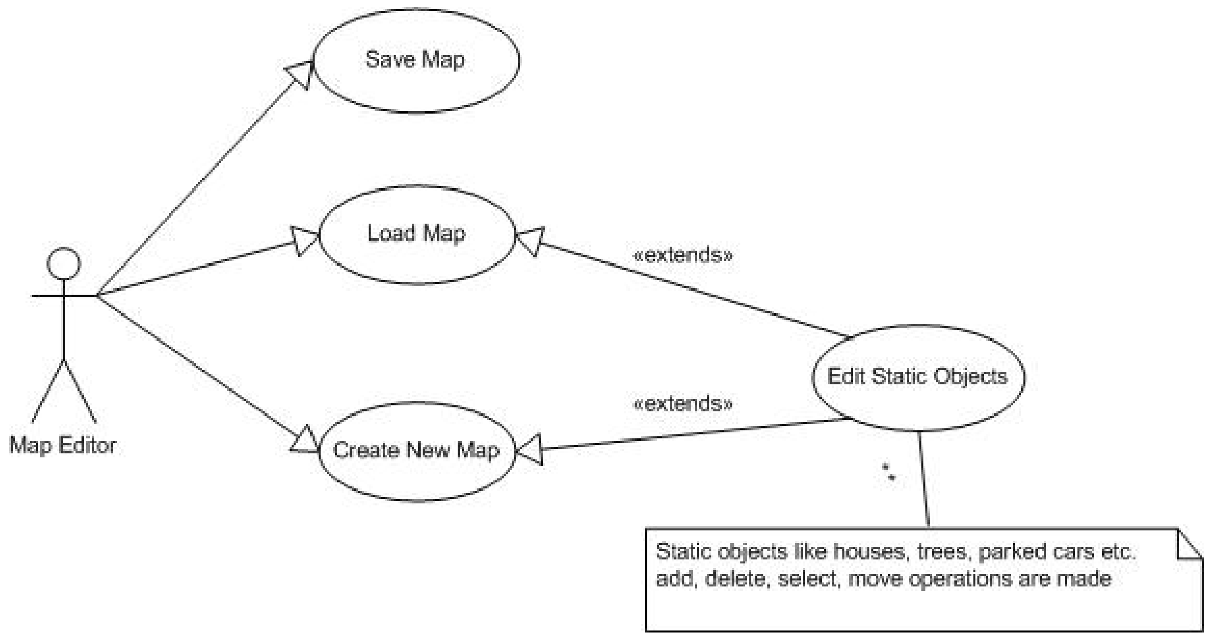
Editor mode is the core part of our software since it is the tool for preparing educational content. Editor mode has three sub-modules:

- Map Editor
- Animation Editor
- Game Editor

4.2.1. Map Editor

Usage scenarios

Map editor is used by educators for preparing the static environment to use in their animations and games.


Save Map
Load Map
«extends»
Edit Static Objects
«extends»
Map Editor
Create New Map
Static objects like houses, trees, parked cars etc. add, delete, select, move operations are made

User will be able to create new maps, load and edit existing maps. For users who don't want to spend their time while creating maps, some basic maps will be provided with the module. So they will be able to edit those maps or even use without editing.

While preparing maps, user will be able to add predefined objects. Those predefined objects can be grouped like:

- Buildings (schools, markets, houses, etc.)
- Roads (one ways, multiple lanes, sidewalks, etc.)
- Traffic primitives (traffic signs, road lines, lights, overpasses, etc.)
- Environmental objects (trees, parked cars, gardens, etc.)

User will be able to select, delete or move previously added objects.

All user interaction for the map will be easy to use mouse drag and drop actions. A menu in the GUI will be provided for file actions (save, load, create new, etc.)

Other specifications

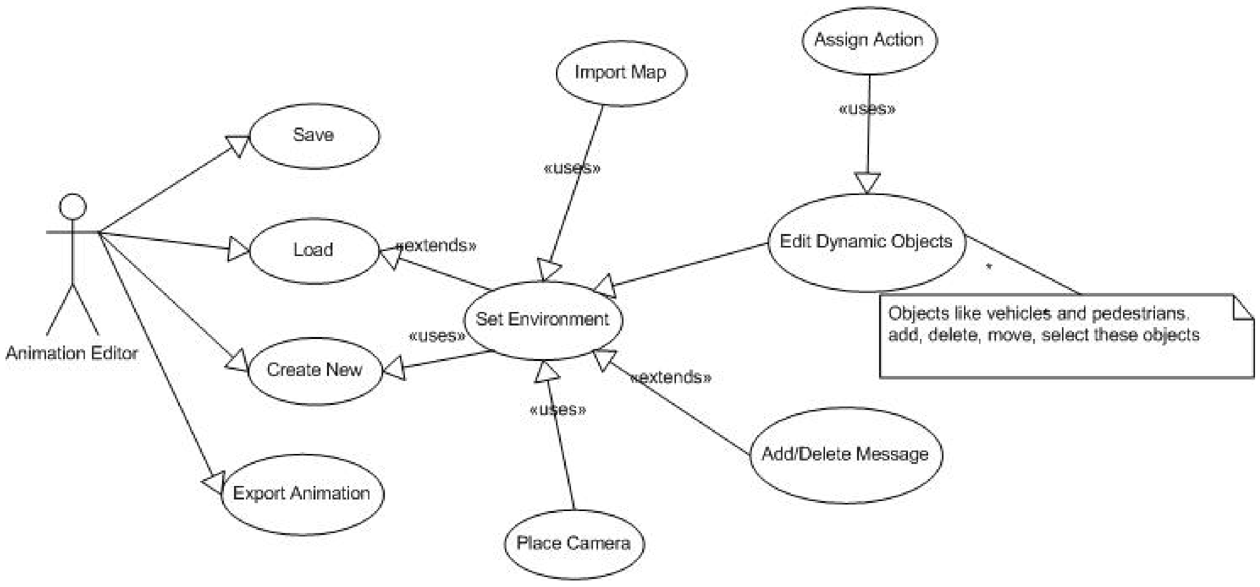
Maps will be saved in its own file format. This format will be recognizable by all other modules of the program.

Predefined addable objects will have their own file format, and will be packaged in a modular manner so that after the release of the program; extensions, upgrades or corrections can be applied easily. For example, other cars or buildings can be added in post-release upgrades. Since we have limited time, we will only provide limited car and house models.

4.2.2. Animation Editor

Usage scenarios

Animation Editor is used by educators for preparing interactive animations for students. Watching these animations are not different than watching a movie, except interactions of the students. Interactions in the animations will be screen messages for interrupting the animation and instructing students.



Firstly, Animation Editor needs a map (which was previously created with Map Editor). So user will have to import a map.

Then, user will add the dynamic objects to the scene. Dynamic objects can be:

- Pedestrians (children, adults)
- Vehicle (cars, buses, light vehicles)

Animation Editor will have a time-line. By this time-line, user can go to any frame of the animation, so user can make modifications to that frame. These modifications will be giving orders to dynamic objects.

The orders which can be given to objects will be predefined in the program. So, user will not have to deal with these orders. In addition, every dynamic object will have specific orders which can be applied only that specific object kind. For example:

- Pedestrians: walk, run, etc.
- Vehicles: move, stop, turn, etc.

In addition to these basic orders, dynamic objects will have traffic oriented orders which will increase the reality of the animation and educational benefits. For example, a pedestrian will be able to look right or left, or a car will turn on its lights or horn.

By the help of time-line, user can change the camera angle at any frame. In addition user may choose the camera to be locked on some dynamic object so that it changes its place automatically in the animation. Moreover, camera can be placed on a static object for a general view of the place.

User will be able add screen messages to the animation at a chosen frame. The purpose of these messages is to increase the educational benefit. These messages can be static screen messages, or multiple choice questions for the student. Users have to determine all content of the screen messages. If a question is asked via screen messages, the correct answer and a value (in the means of grading) should be assigned to that question. Also, user can record and bound sound clips to screen messages. These sound clips will be played while showing the message in the screen.

When user finished creating animation, it can be saved. Also, previously saved animations can be loaded for editing. For the platforms, where our software cannot be installed (video players, etc.), user can export the animation in a popular video file format (for example .avi, .mpg, .mov). But in this case, the animation will lose its interaction capability although screen messages can still be shown for a certain time period.

All user interaction for the objects will be mouse actions. For changing camera settings, user will use the keyboard. And for file options, a menu will be provided. Also time-line will be implemented in the GUI.

Other specifications

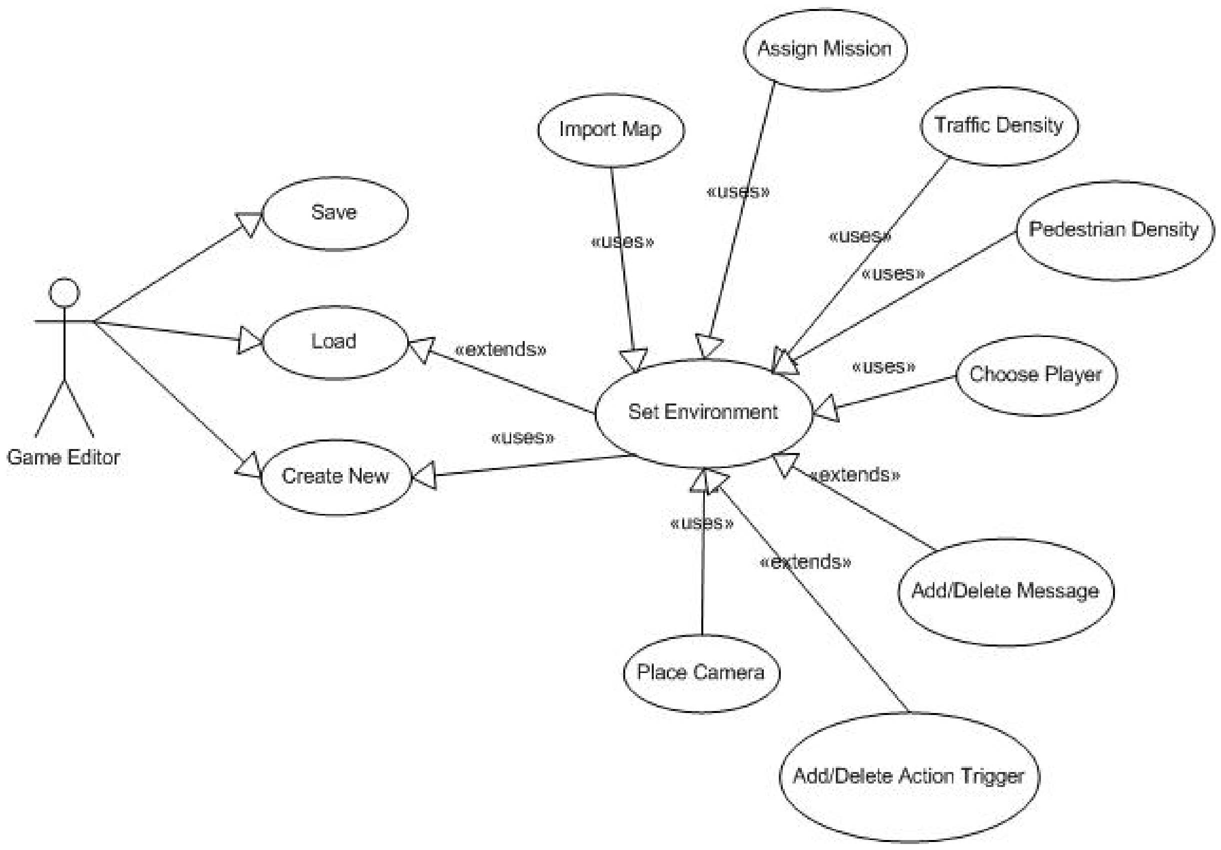
Animations will be saved in its own file format. This file format will be recognized by viewer module of our program.

Dynamic objects will have their own file format, and will be packaged in a modular manner so that after the release of the program; extensions, upgrades or corrections can be applied easily.

4.2.3. Game Editor

Usage scenarios

Game Editor is used by educators for preparing games for students. These games later can be played with viewer module of our software. Purpose of these games is giving student a chance to apply the knowledge gained from other sources (animations created by this software or other sources like in-class education).



Firstly, Game Editor needs a map (which was previously created with Map Editor). So user will have to import a map. On this map user may define several traffic routes, in which cars will randomly drive. The traffic density of these routes can be set to different amounts. Similarly, several walking routes can be defined, on which pedestrians wander.

Then, user must specify the playable character. A playable character can be a child on foot, or a child with a vehicle. Since children aren't allowed to drive motor vehicles, only a bicycle, a skateboard or a roller skate will be provided as vehicles.

After choosing the playable character, user must choose the starting location and the ending location for the character. Ending location is a tool to create missions in the game. When student reaches the ending location, the game will end successfully.

In every game, user may create a starting screen which will tell the student about the mission of that game. In addition to this screen, user may add tutorial characters to the game. In fact, these characters are nothing but message screens (like in Animation Editor), which will be activated by student by getting close enough to them.

Although game will create error messages for violation of traffic rules, educator may want to teach something that is an informal traffic rule, or a good habit. Game Editor will provide a useful tool for this purpose. User will be able to add invisible action triggers to the game, and bind custom directional or error messages to these triggers. For example, a trigger can be placed anywhere, so that student will get an error message when student step on it.

After game creation is ended, user can save the game. Later, user can load previously saved game to edit.

All user interaction for the objects will be mouse actions. And for file options, a menu will be provided.

Other specifications

Games will be saved in its own file format. This file format will be recognized by viewer module of our program.

4.3. VIEWER MODE

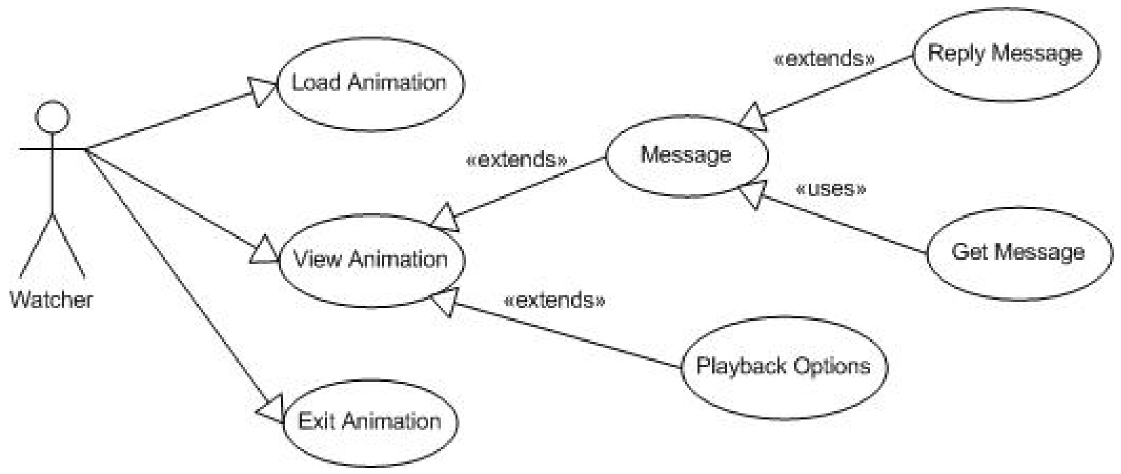
Viewer mode is the second part of our software. It will be used by both educators (for testing or demonstrations), and students (for watching or playing). User may choose one of the following actions:

- watching an animation
- playing a game

4.3.1. Watching Animations

Usage Scenarios

This module of the software allows users to watch the animations created by Animation Editor of our software.



First, user must choose the animation to watch. Then usual playback options will be provided to user (play, pause, stop, rewind, etc).

Once user start playing the animation, all user should do is to sit and watch, until a message screen appears in the animation. Then, user should interact with the screen messages.

If there is a static screen message, it will stay until user read it. Then, animation will resume. If the screen message is multiple choice question, then user should select an answer to continue (otherwise his grade will reduce).

When the animation ends, an information screen will appear which will tell the user about the points taken from questions.

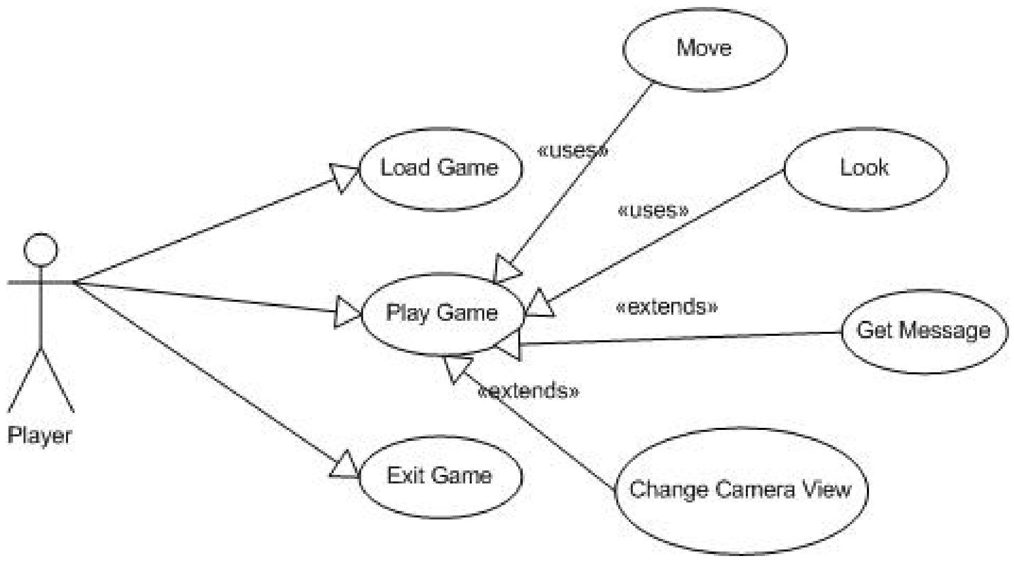
Other specifications

Answers to the questions during the animation will be kept in a file for future reference.

4.3.2. Playing Games

Usage Scenarios

In this module, user is able to play the games created by game editor of our software.



Firstly, the user chooses the game he wants to play and loads it. Then, with mouse and keyboard he moves the character. These movements are very basic movements like turn right/left, look right/left, run/walk, etc. Also, depending on the choice of the educator (who prepared the game), user can change the camera angle.

In a game, user must finish a given mission; this mission can be going to school from home or from home to a market, etc. During the game, user also faces some messages prepared with the game editor, these messages are to direct the user as a tutorial. When user makes a general traffic rule violation or a violation of educator defined error, he gets a message.

Other specifications

Objects other than the user have predefined AI. According to this AI, as an example, cars move obeying the traffic rules, stop by the red light; slow down when coming to a crossroad, etc.

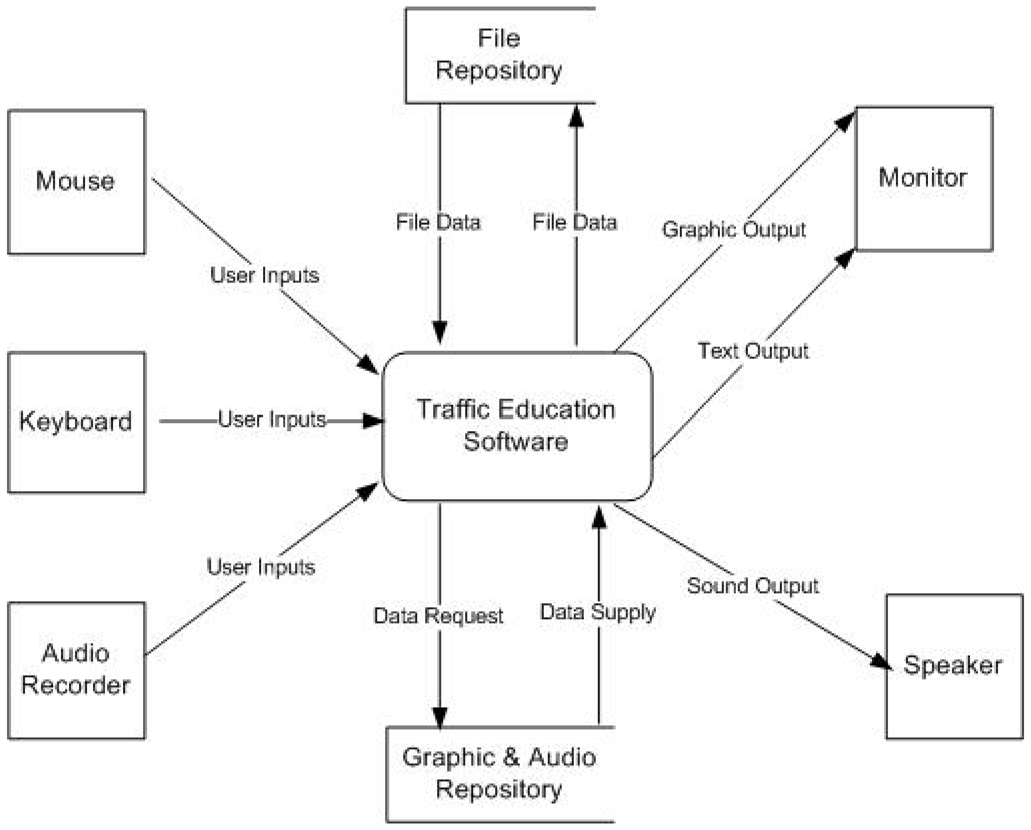
At the end of the game, the user log will be kept in a file. This file contains the mistakes the user did during the game.

5. DATA MODELING

In our data model, user can interact with system through mouse, keyboard and audio recorder (it is just for educator). User requests are processed; then, output data is accessed by monitor or speaker.

5.1. DATA FLOW DIAGRAM LEVEL 0

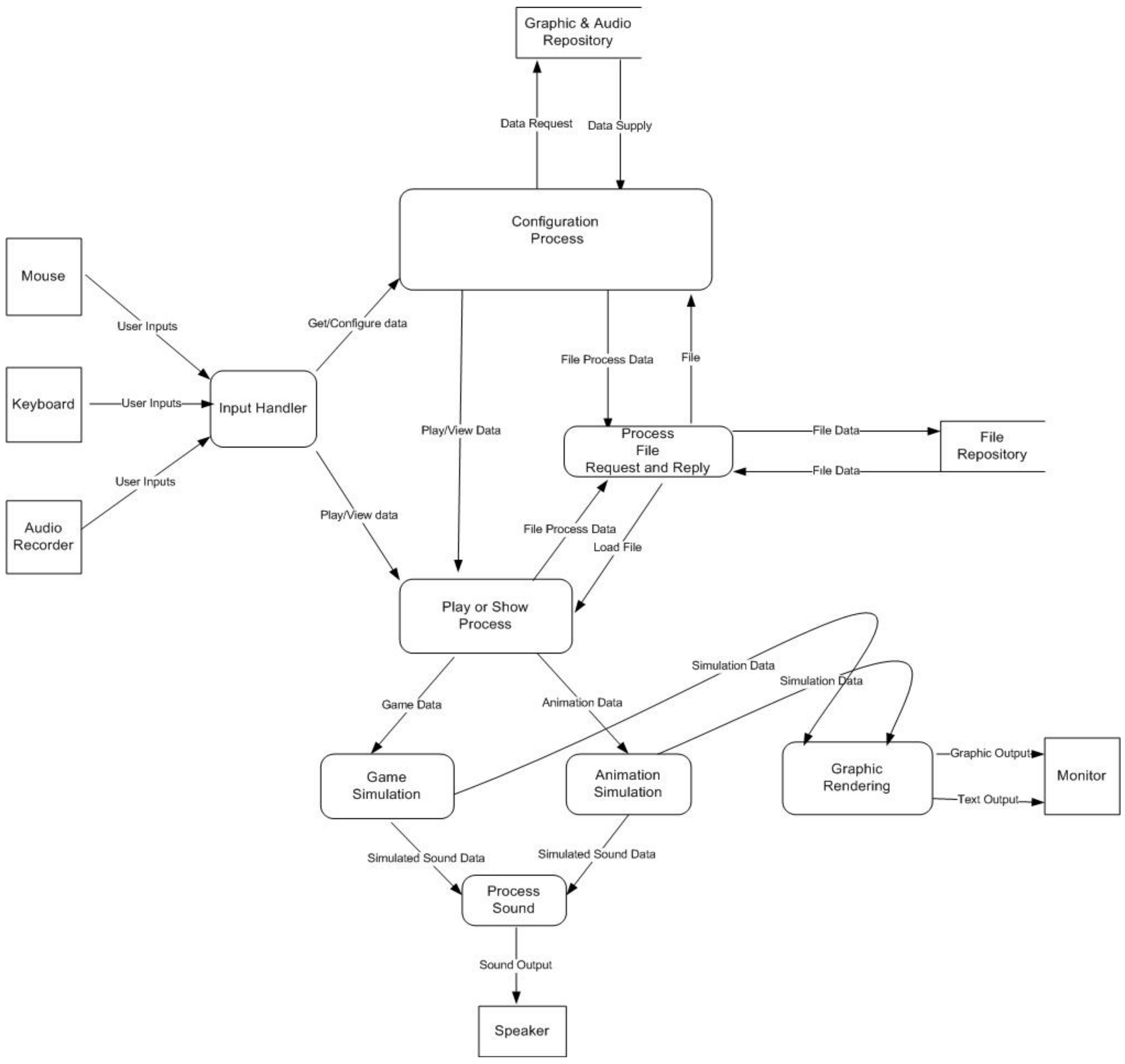
We figured out general input-output behavior of Traffic Education Software on Level 0 DFD. Also we placed the basic structure of system data that are 3D models, sound effects, games, and animations.



Audio recorder here is optionally used by educator when an audio message like a warning is needed. Messages could be sent through speakers. We used this option to strengthen the effect of messages which are sent to students. Also some realistic sound effects will be given through speaker.

5.2. DATA FLOW DIAGRAM LEVEL 1

Level 1 DFD keeps the balance of Level 0 DFD. Same interactions with outside world still exist. We just cut Traffic Education Software into parts that we gave major sub processes of our main system. We have eight sub processes; those are Input Handler process, Configuration Process, Process File Request and Reply, Play or Show Process, Game Simulation process, Animation Simulation process, Process Sound, and Graphic Rendering process.



File Repository: This repository is under user control. It will hold some predefined maps initially. User can add animations, games, and maps after configuring them. All those can be created changed, deleted, loaded, and uploaded. And those operations are under File Request and Reply Process's control.

Graphic & Audio Repository: Graphic and audio components are kept here while compositions of those components are kept on file repository as an animation, game, or a map. This repository is under system control. User will have no right to make any change on it. She can use this repository through editors. Get predefined graphic and audio data to make their editing.

5.2.1. Processes

Input Handler:

This process separates users, an educator or a student, directs them to different modules. If our user is a student this process just directs the user to Play or Show Process, sends required data; than, Play or Show Process could know which to play or show. If our user is an educator, Input Handler directs the user to Configurations process. It also differentiates between keyboard, mouse, and recorder. It sends data with respect to the incoming input type. For example an audio input can only be sent to Configuration Process through the data line named as Get/Configure Data from the figure. This is because only Configuration Process operates on such an editing.

Configurations process:

This process is used by Educators only. Educator can examine game or view animation through this process by sending required information to Play or Show Process. Configurations process includes map configurations, game configurations, and animation configurations. We summed them up in one process to simplify Level 1. It incorporates our editors (all editing processes). It also controls Process File Request and Reply (where file create, load, save, and delete are processed).

Play or Show Process:

By the data flow, Play/View Data, it processes on playing or animation show. It requests required files to play or show by sending data through File Process Data arrow; gets file through the data flow on Load File arrow.

Process File Request and Reply:

This process communicates with file repository where all animations and games are kept. A new file can be created. An existing file can be uploaded. A file can be loaded. All file requests and replies are taken from this process. The data flow from here to Configurations is to send file to Configurations. Configurations process gets file, make editing, and resend it to this process. The incoming data through the File/Process Data arrow could be a file or just a command to direct Process File Request and Reply work.

6. HARDWARE AND SOFTWARE REQUIREMENTS

6.1. INSTALLATION ENVIRONMENT

Our software is intended to run on Microsoft Windows operating systems. Since our program includes 3D animations, a new generation graphics card will be needed for good performance (a card with hardware 3D support).

6.2. DEVELOPMENT ENVIRONMENT

Since we have little experience on 3D programming, we made a detailed research about potential tools that we may use while developing our software. And we decided to use several different possibilities. These are:

- *Blender3D (www.blender3d.org)*

Blender3D is an open source tool for 3D environment developing with community support. That's the main reason that we chose Blender3D. It has nearly all features that we need: modeling, animation, physics, rendering, sound synchronization, file import/export, scripting support, etc. But it has a disadvantage that it is hard to use, since it is programmed for functionality not ease of use. Although we'll mainly use that software, we will use other softwares for backup. It is platform independent.

- *3D Studio Max (www.autodesk.com/3dsmax)*

It is another modeling tool. It is very popular in 3D modeling applications. We will use this software for character modeling and skeletal animations. It requires Microsoft Windows.

- *Ogre3D (www.ogre3d.org)*

It is a high level, cross platform, object oriented, 3D graphics engine. Again, it is an open source software. It supports OpenGL and Direct3D support. It includes many useful features that are commonly used in 3D programming, which makes programming easier and faster. Also it is designed to easily integrate with other libraries (sound, network, AI, etc). It supports all popular file formats (including 3ds Max files).

- *OpenAL (*www.openal.org*)*

It is a high level, cross platform sound engine. It is very popular in video games industry. It allows user to add composite sound effects.

- *Microsoft Visual Studio*

We will heavily use C++ and C. So we chose to use MS Visual Studio as our compiler. The main reasons are: most of the libraries we will use are successfully tested on Visual Studio, its user interface is simple, and it works very efficient for Windows environments. It requires Microsoft Windows.

Most of the time we will be using Blender3D and 3DS Max for modeling our 3d objects and later using other libraries, we will combine these objects with other sources (physics, sound, etc) in Visual Studio. So we just need Windows PCs with above software installed. In addition, we'll need video cards with 3D support in our development computers for testing our software.

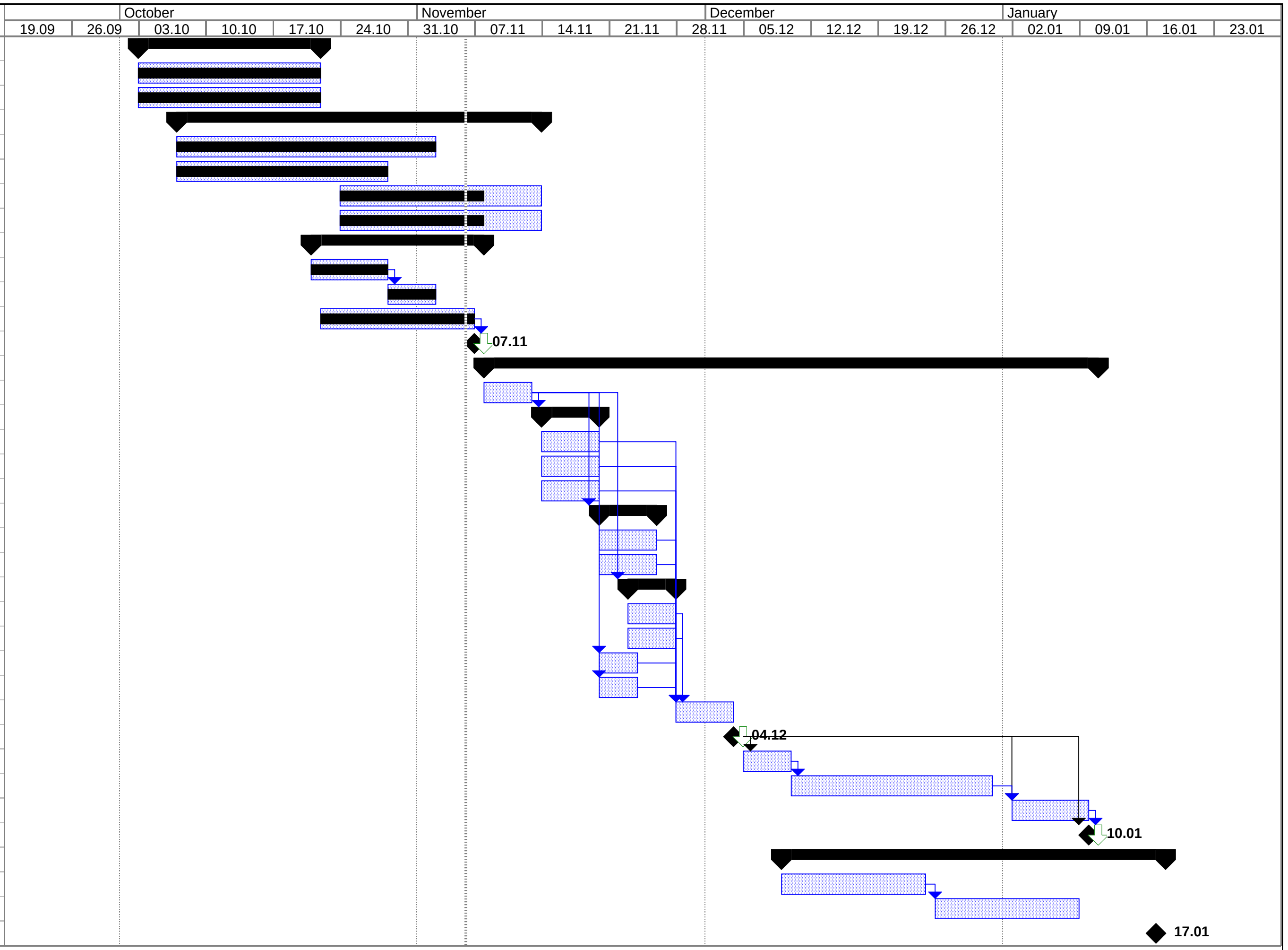
7. CONCLUSION

The current education system for traffic education is not based on an interactive and developmental method. Students have to learn so many subjects which include more than their need without any application about that subject. That was the main reason for us to choose this subject. Also we have seen that there are not enough and well qualified software products in the market for the traffic education. We believe that if we get the correct connections with the right people, we can fulfill this need in the area.

8. APPENDIX

8.1. GANTT CHART

ID	Task Name	Duration	Start	Finish
1	**Project Definition**	**15 days**	**Mon 03.10.05**	**Fri 21.10.05**
2	Problem Definition	15 days	Mon 03.10.05	Fri 21.10.05
3	Project Scope Determination	15 days	Mon 03.10.05	Fri 21.10.05
4	**Research and Survey**	**29 days**	**Fri 07.10.05**	**Sun 13.11.05**
5	Literature Survey	20 days	Fri 07.10.05	Wed 02.11.05
6	Meetings	16 days	Fri 07.10.05	Fri 28.10.05
7	Technical research	18 days	Mon 24.10.05	Sun 13.11.05
8	Development tools research	18 days	Mon 24.10.05	Sun 13.11.05
9	**Requirement Analysis**	**13 days**	**Fri 21.10.05**	**Mon 07.11.05**
10	Deciding on specifications	6 days	Fri 21.10.05	Fri 28.10.05
11	Making Use Cases	4 days	Sat 29.10.05	Wed 02.11.05
12	Report preparation	11 days	Sat 22.10.05	Sun 06.11.05
13	*Requirement Analysis Report*	1 day	Mon 07.11.05	Mon 07.11.05
14	**Design**	**64 days**	**Tue 08.11.05**	**Tue 10.01.06**
15	Specifications review	5 days	Tue 08.11.05	Sat 12.11.05
16	**Editor Design**	**6 days**	**Mon 14.11.05**	**Sat 19.11.05**
17	Map Editor	6 days	Mon 14.11.05	Sat 19.11.05
18	Game Editır	6 days	Mon 14.11.05	Sat 19.11.05
19	Animation Editor	6 days	Mon 14.11.05	Sat 19.11.05
20	**Viewer Design**	**6 days**	**Sun 20.11.05**	**Fri 25.11.05**
21	Game player	6 days	Sun 20.11.05	Fri 25.11.05
22	Animation watcher	6 days	Sun 20.11.05	Fri 25.11.05
23	**Data design**	**5 days**	**Wed 23.11.05**	**Sun 27.11.05**
24	Defining file types	5 days	Wed 23.11.05	Sun 27.11.05
25	File interactions	5 days	Wed 23.11.05	Sun 27.11.05
26	GUI design	4 days	Sun 20.11.05	Wed 23.11.05
27	Multimedia design	4 days	Sun 20.11.05	Wed 23.11.05
28	Report preparation	6 days	Mon 28.11.05	Sat 03.12.05
29	*Initial design report*	1 day	Sun 04.12.05	Sun 04.12.05
30	Revision of initial design	5 days	Mon 05.12.05	Fri 09.12.05
31	Implementation plan	21 days	Sat 10.12.05	Fri 30.12.05
32	Report preparation	8 days	Mon 02.01.06	Mon 09.01.06
33	*Detailed design report*	1 day	Tue 10.01.06	Tue 10.01.06
34	**Prototype**	**40 days**	**Fri 09.12.05**	**Tue 17.01.06**
35	Prototype design	15 days	Fri 09.12.05	Fri 23.12.05
36	Prototype implementation	15 days	Sun 25.12.05	Sun 08.01.06
37	*Prototype release*	1 day	Tue 17.01.06	Tue 17.01.06



Project: ANKA_SCHEDULE
Date: Sun 06.11.05

Task	▨	Progress	▬	Summary	▬	External Tasks	▨	Deadline	⇩
Split	┈	Milestone	◆	Project Summary	▬	External Milestone	◆		