Chemistry Learning In Progress

Final Document

Nathan Mikeska
Neil Alfredson
Richard Carney
Brian Navarro

Table of Contents

1. Introduction

1.1 Document Overview

This document encompasses all the information related to the C.L.I.P. system that was outlined in the Requirements Analysis Document, Design Specifications Document, and the Project Plan. It first provides an introduction to the project and the team. Following that is a description of the current system and the proposed system requirements as laid out in the Requirements Analysis Document. The design of the proposed system is then covered in full detail as it was in the Design Specifications Document. The document then covers the development process followed by a detailed test plan for the C.L.I.P. system. The deliverables are then listed. A detailed schedule for the project is then laid out along with descriptions of the schedule's tasks as can be found in the Project Plan Document. The document then concludes with measurements, risk management, and ethics. The final section of the document is a glossary covering specific terminology found in this document.

1.2 Purpose of the System

CLIP (Chemistry Learning In Progress) will be a new digital system intended to help chemistry students attain a better understanding and recognition of the organizational patterns related to the Periodic Table of Elements. The system will also provide functionality to assist professors with replaying the decisions made by a student who previously used the system. This part of the system relates to the organizational thought processes of students when it comes to pattern recognition.

The client is Prof. Susan Wiediger of the Chemistry Department. They have a current system on paper that involves the students moving around paper cards. Our client wants to make the current system electronic and available to students over the web. She also wants the students to be able to submit their results to her, possibly to be used as an assignment.

1.3 Scope of the System

The system should be developed with the scope objective of being available to many students within a professor's chemistry class and outside it. The client has suggested that this can be accomplished with applet, a web-application that has been used by many educational tools in the past.

The applet should meet the basic functions of the current system where it allows the user to use the program to organize the tiles and arrange them as they wish. The students also need to be able to submit their final arrangement to the professor. The professor will then be able to view the arrangement and see the choices a student made to reach it.

Furthermore, the system should allow for the expansion of functionality and data. Therefore, the system should be well documented to allow expanded functions to be added easily later. Also, we will need to allow the addition of new tile sets.

1.4 Objectives and Success Criteria of the Project

The objective of this project is to satisfy necessary requirements that are in current system and if possible, add new features. Such new features include allowing the teacher to check the end solution provided by the student against accepted patterns. Therefore, this project will be successful if it can replace the current paper system.

1.5 Definitions, Acronyms, and Abbreviations

C.L.I.P - Chemistry Learning In Progress

Professor (teacher) – A user that accesses the system from the perspective of a certain role. This role includes the use of the playback and creation functionality.

Student – A user that accesses the system from the perspective of a certain role. This role includes the use of the tile arrangement functionality.

User – Every user has access to the entire functionality of the system. Generally a user fulfills the role of a professor or student as defined above.

1.6 References

http://www.cs.siue.edu/SeniorProjects/f05g6/ C.L.I.P. Project page

2. Stakeholders

Client

The client for the project is Susan Wiediger, a chemistry professor at Southern Illinois University of Edwardsville. Along with defining the requirements she will also be a primary user of the system.

Users

The primary users of the clip system will be chemistry students and chemistry professors. The students will perform card sorting and arranging activities, save their results, and submit their results via email to their professor. The professors will then observe the students moves and analyze their thought process. The students will gain knowledge of the periodic table of elements and the professors will gain knowledge about the thought process of their students. However the system will not be restricted to only these two groups, anyone can use the system.

Development Team

The development team consists of four members; Nathan Mikeska, Neil Alfredson, Richard Carney, and Brian Navarro, who will receive grades in CS 425 and CS 499 Computer Science Senior Project Course depending upon the success of the project.

3. Project Organization

There are four team members working on the CLIP system. In order to maximize the effectiveness and organization of our work, each member will be taking certain roles in the project, although all members will still assist in most portions of the project.

Project Manager

Nathan is the Project Manager. As Project Manager, he will oversee all aspects of the project and keep it focused and on track. He will assist the other members in their assigned positions as well as lead any portion of the project not assigned to any other member. He will also handle communication between upper management and the team as well as communication with the client and the team.

Lead UI Designer & Lead Programmer

Brian Navarro will fulfill the role of Lead User Interface Designer as well as Lead Programmer. As the Lead User Interface Designer, he will be responsible for creating the look and feel of the user interface of the C.L.I.P. system, along with creating the tile sets.

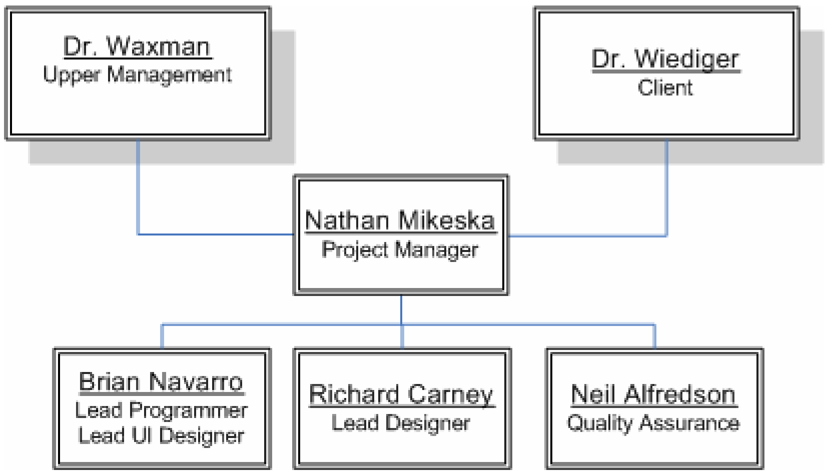
Furthermore, he has the responsibility of lead programmer since he has some experience Java programming. Due to the nature of being lead programmer, he will also take the role of Risk and Development Officer as described in section 9 with the responsibility of monitoring the risks created because of the new technologies being used by the team.

Lead Designer

Richard will be handling the role of Lead Designer. He will focus on the design of the system and take the lead in any addition or removal of features that require changes to the system design.

Lead Quality Assurance

Neil will take the role of Quality Assurance. He will be responsible for taking the lead in testing and bug finding.



4. Current System

4.1 Overview

The current system serves two purposes. Foremost, it acts as an educational tool for students, assisting them in becoming familiar with the various Periodic Table elements, relationships, and patterns. Furthermore, the system provides some insights into a student's thought process for a professor as they review the final arrangement of that student. Presently the system is non-computerized and focuses on the interaction between the professor and the student through a set of paper cards.

4.2 Functional Description

Paper Cards

The student is given a set of randomly sorted cards, each possessing a certain number of attributes. These attributes together are able to uniquely identify each card. However, the cards all share the same type of attributes, where the different values for them can be used to form the patterns that link them all together (ie: a boiling point attribute, where each card has a certain boiling point number and the student can arrange them in ascending or descending order).

.

<u>The Arrangement Area</u>

This can represent any number of surfaces, preferably flat, and is where the student lays out their cards to begin the Card Sorting Activity.

<u>Card-Sorting Activity</u>

After the student has received a set of cards, they are asked to arrange the cards on a table according to how they identify the relationships among the differing valued characteristics on each the card. The student is not limited to any particular form of the arrangement. Any number of cards in the set may be excluded, if the student does not identify the cards as having a position in the final arrangement. In addition to the discarding cards, the student could discover that there are cards not provided in the original set that complete the relationships among the organized set. When this happens, the student creates new cards and places them into the arrangement.

<u>Observation</u>

While the card sorting activity is occurring, the professor observes the student. The professor makes note of each move of the cards and the time spent thinking between the moves. During the process, the professor may ask questions about a student's decision and offer feedback and suggestions.

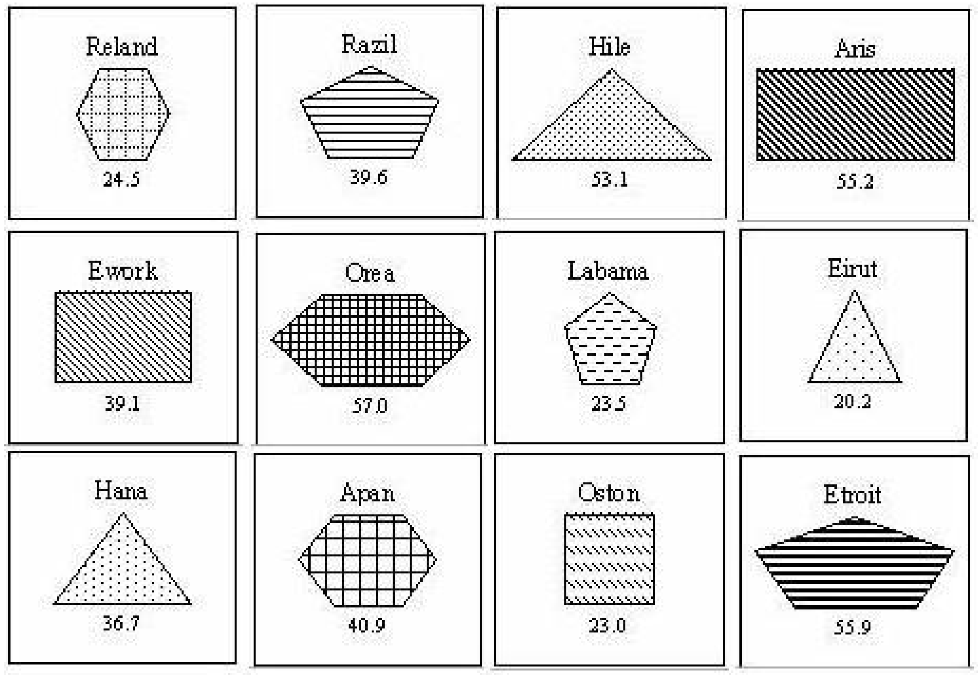
<u>Patterns</u>

A pattern is a final arrangement of the cards. While every card set has at least one best final arrangement, there exist other patterns that are acceptable.

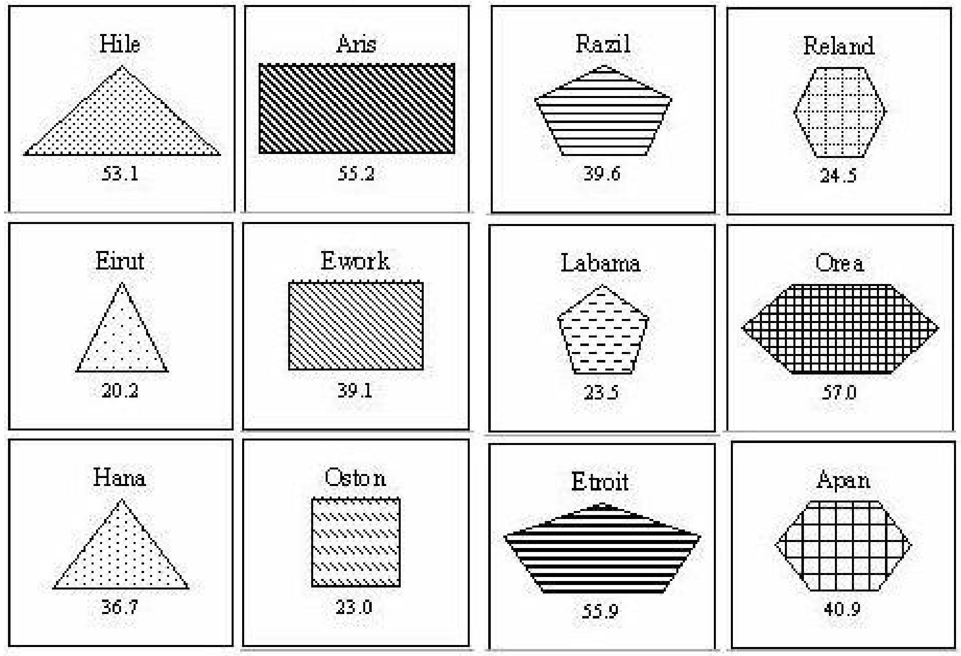
<u>New Sets</u>

Currently there are three sets of cards used in the system, but the professor can create new sets. Each set may focus on a different collection of relationships.

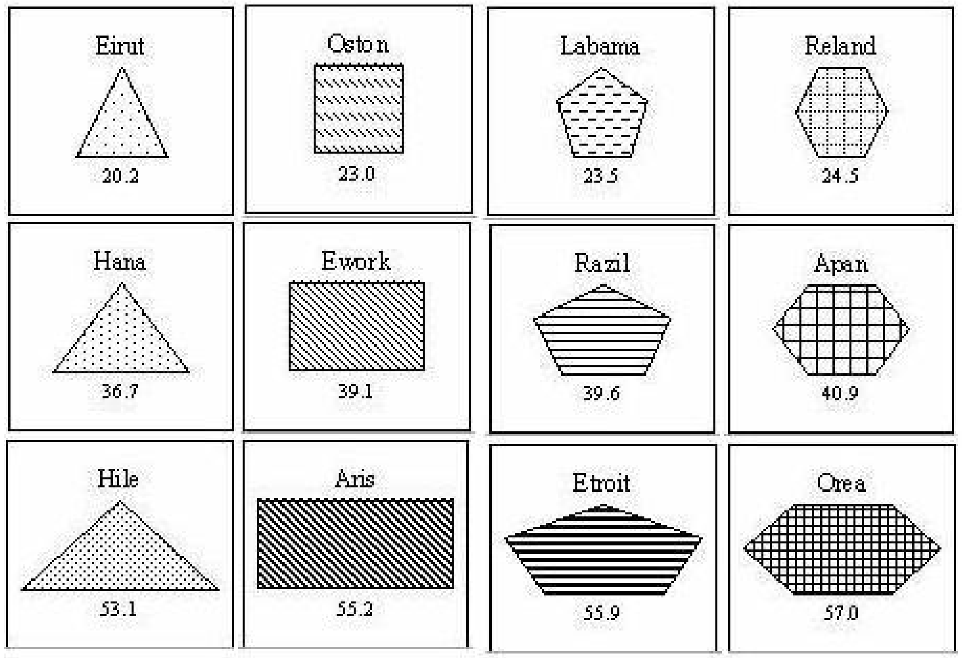
4.3 Example



A student is given a set of cards in random order.



The student decides to arrange the cards in columns of increasing number of sides.



The student sorts each column by increasing number.

The final figure represents the best possible pattern to sort the cards in. We see several patterns here: every column has one more side than the last, numbers increase from left to right, numbers increase from top to bottom, and the card patterns become darker from top to bottom.

5. Proposed System Requirements

5.1 Requirements Overview

The Chemistry Learning In Progress (C.L.I.P.) tool will be web-based. Its purpose is to replace the current paper version with an automated application that can handle the same card-sorting activities that are meant to mimic the organization of the periodic table of elements. The tool is educational based and will be used primarily by students and professors. The students will be presented with various sets of tiles that they must organize in best arrangement to create the pattern that links them all together. After the student has submitted their best answer, the professor will review the results.

5.2 Functional Requirements

<u>The Grid</u>

The Grid is where tiles are dragged to and organized to create the pattern that forms the final arrangement. It should provide a similar feeling to that of the Arrangement Area used in the current system. This means that the Grid should feel as natural as possible and allow for any spot on the grid to be filled or left empty.

The Play Area

The Play Area is the area to the right of the Grid. It will display a scrollable list of tiles that are currently unused. In this list, the tiles should be fully visible similar to the current system so that user can easily differentiate between items.

Mini-map

A mini-map will be available to provide an alternative viewing of the grid where the entire area is shown with a miniature version of each tile.

Playback Controls

After a user loads up an arrangement file, they will be allowed to control the viewing of it through a pause, play, stop, forward, and rewind button.

Dragged Tiles

Any tile of a set can be dragged from its current location to a spot either on the Grid or the Play Area.

Dropped Tiles

When a tile is dropped, it must be on an empty grid space in the Play Area, else it will be returned to the location it was originally dragged from.

Creating a Final Arrangement

The student will be presented with a set of tiles that start out in the Play Area. They will have the task of forming the overall relationship (several patterns together) that creates the best final arrangement. To do this, they will drag and drop all tiles they feel follow the overall pattern, leaving out any in which they feel does not adhere to the suggested pattern. Tiles are considered left out when they remain in the Play Area. Upon submitting the pattern, for each tile left out, the student will be asked to explain their reasoning for that by means of a dialog box. Likewise, the student will also be able to add blank tiles to take the place of missing tiles. For every blank tile added, they will need to include a description of how that tile fits in the pattern. They will do this via a dialog box that appears when they click the add blank tile button.

Tile Movement Recorded

As soon as the first tile is moved, the recorder will begin tracking the movements until the student is satisfied with his/her arrangement pattern and has saved the pattern for submission. Every tile move will be stored in memory and then written to a log file upon saving. The starting position and ending position of a moving tile will be what is recorded in memory. The professor will have the ability load a file into memory and to fast-forward, rewind, pause, etc. all of the recorded movements. This will allow the professor to carefully analyze how a student came up with a final arrangement pattern for a set.

Creation of New Sets of Tiles

The professor will be allowed to create new sets of tiles that can be added to the available sets. In all cases, the tiles will display images. These images will be created outside the system. The professor will also have the ability to create acceptable patterns as well as set rules for each new tile set. These rules include tile set name, description, size of tiles, ability to add blank tiles, ability to check pattern, ability to leave tiles out of the pattern arrangement, a show grid option.

Additional Sets of Tiles

An initial set of tiles will be preloaded by the application every time it is used. However, this is limiting, so the ability for the professor to add additional sets of tiles will be available.

Modification of Existing Sets of Tiles

The professor will be allowed to modify any existing set of tiles given that all the necessary information is properly provided.

Results Submitted

After the application is used with any set of tiles, those results may be submitted online. All the necessary data related will be sent to the professor through email is outside the scope of the system. Therefore, when the user saves his/her final pattern, they will be instructed that they simply must attach that file as an attachment in an email.

5.3 Non-Functional Requirements

User Interface and Human Factors

The user interface will be intuitive and easy to use. Since the UI mimics the physical card sorting activity, it should feel as natural to the user as the physical version. Little user training should be necessary and new users should quickly come to understand how the interface works.

Documentation

A set of online tutorials will be available to help explain the program to any users who require a better understanding of how the system works. A full manual will be made available with thorough explanations of every bit of functionality that the program provides. On the same page as the web application, a set of tutorials and documentation on the system will be provided through a series of html links.

Software Considerations

Since the system will be web based, users will need an internet connection and a standard internet browser that supports the Java Virtual Machine. The system will be designed for systems that support the Java Virtual Machine, specifically Windows 2000, Windows XP, and Mac OS X.

Hardware Considerations

The system will require a web server. The web server will need the bandwidth to support multiple file transfers for when students load up the applet initially. During this initial stage, the file transfers are simply the loading up of image data from the web server hosting the applet.

Performance Characteristics

The system should respond to user input with minimal delay once the applet has loaded onto the user's machine. There will be an initial delay that is dependent upon the user's connection as the applet class files that make up the system will be downloaded to their computer during that time.

Since most of the user interaction will involve dragging tiles with the mouse, the system should not exhibit any sort of lag so as not to frustrate or annoy the user. Users with dial-up connections should be able to load the system within a reasonable amount of time.

Error Handling and Extreme Conditions

In the event that the user loses connection to the internet, the system will support the ability to continue the user's session without any loss of work since the applet will be loaded on the user's local PC.

Quality Issues

The user interface will need to be easy to use and consistent throughout, only changing in form when expected and necessary. This system is primarily a front-end system with a small back-end for minimal processing, computing, and configuration. Therefore, a considerable amount of effort must be spent on testing and maintaining an understandable, useable, useful, and robust interface.

System Modifications

The system should support an easy and reliable way to create and use additional tile sets. The users should be able to use these additional tile sets with minimal effort. The teachers should be able to easily create or modify tile sets and provide them for students to use.

Physical Environment

No specific physical environment is required for the system.

Security Issues

The primary security issue is keeping the user safe while they use the applet. This means that we will only be allowed to read/write their local machine with their permission which is asked for before the applet starts loading. Therefore, the applet must be digitally signed by the team verifying it is safe to use.

5.4 Pseudo Requirements

Our client has indicated two constraints:

1. The system be web-based, preferably as an applet since she has viewed many educational tools take this form.
2. The User Interface needs to demonstrate a similar ease of use that is accomplished by hand with the paper cards.

.

5.5 System Models

5.5.1 Scenarios

Scenario: Tile Arrangement

Participating Actors: John: student user

Goal: Move the tiles into the desired arrangement and submit results.
Conditions: John is at a computer and ready to use the program.
Outcome: John arranges the tiles, saves his results and submits them using email.

1) John first opens up his web browser and types in the web address that the program is located at. John loads up the program.
2) The program asks John which tile set he would like to use for this session. John selects one of the three pre-loaded sets or a set stored on his own machine.
3) The program gives John all the tiles for the set that he selected and places them in the playarea. John begins examining and arranging the tiles in the playarea. When he has a decent idea of how he is going to try arranging the tiles, he starts moving his tiles from the playarea to the grid area. John makes many changes to the arrangement before he is satisfied.
4) John decides that one of the given tiles does not belong in his arrangement and he opts to leave it in the playarea excluding it from his arrangement.
5) John also decides that the set needs another tile to complete his arrangement. He selects the option to add a blank tile to the set. A blank tile is added and the system then prompts him to enter information about it regarding the values of its attributes. John enters this information.
6) John feels that he has completed his tile arrangement to the best of his abilities and decides that it is time to save his results. He selects the option to save his arrangement. The system prompts him to enter the reasoning why he left that specific tile out of the arrangement. The system then stores a log file onto his computer. John then sends this file to his professor via email.

Scenario: Results Playback
Participating Actors: Jane: teacher

Goal: To examine the arrangement and final solution of a student.
Conditions: A student has sent Jane an email containing a log file and Jane is ready to examine them.
Outcome: Jane thoroughly examines the student's results until satisfied.

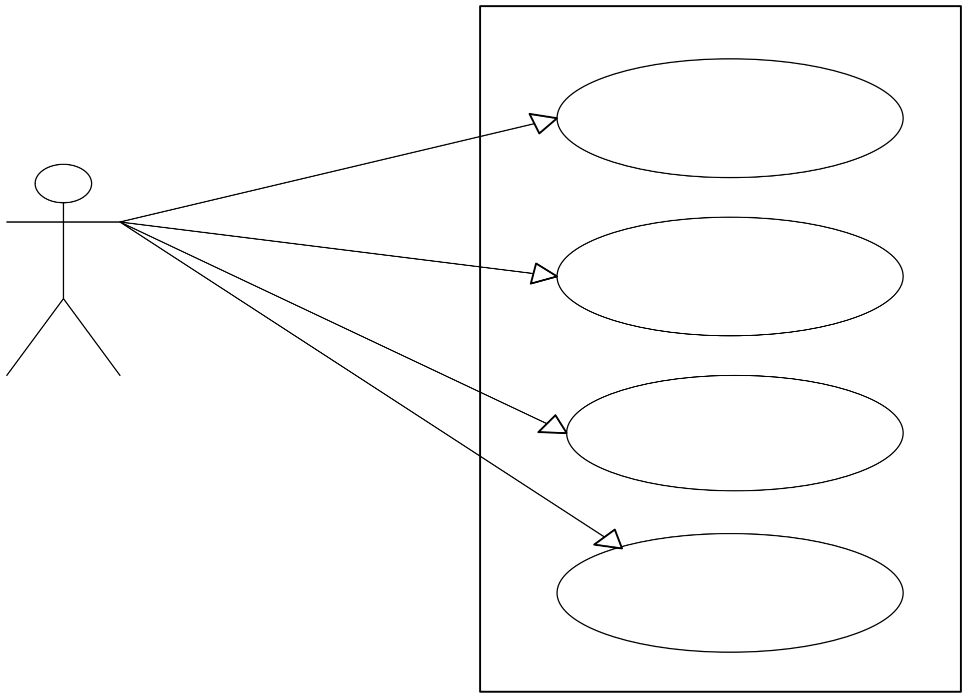
1) Jane first opens up her web browser and types in the web address that the program is located at. Jane loads up the program.
2) Jane selects the option to view a results file. She then browses to the location of the students file she wishes to examine and selects it. The program loads up the file.
3) Jane then watches the tile movements that the student made. She often rewinds the playback or fast-forwards it. She eventually reaches the end and then closely examines the result of the student. At the end Jane is also shown the students reasoning behind the left out tile. Jane feels she has sufficiently examined the student's results.

Scenario: Create Tile Set
Participating Actors: Jane: teacher
Goal: To create a new tile set for the program.
Conditions: Jane has the program loaded up and is ready to design a new tile set.
Outcome: Jane has created and saved a new tile set.

1) Jane has the program running and decides to create a new tile set. She selects the option to create a tile set located in the tools file menu.
2) A form is displayed that has three tabs on it. Jane is on the tile editor tab. Jane now has an empty set of tiles. She selects the option to add a new tile. She browses to the location of the images she created with a third party graphics program and selects one. The name of the text file is now displayed in the tile set list and the image is shown below in the image preview box.
3) Jane repeats step two until she has created all the tiles that she wishes the new tile set to contain. She then hits save.

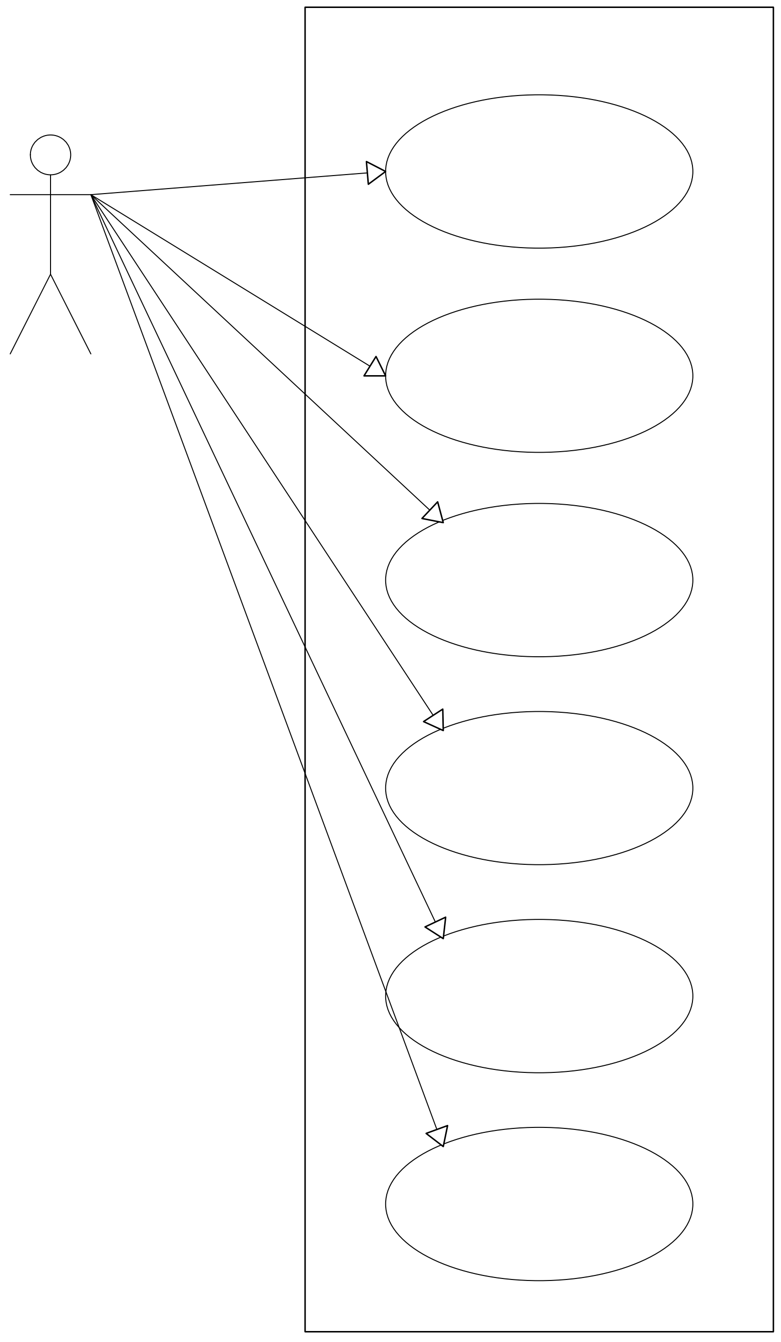
4) Jane then selects the rule editor tab. There Jane enters the name of the new tile set and a description about the tile set. A set of rules for the tile set (such as to allow addition of blank tiles or not) is also located here. Jane edits these rules until she is satisfied and hits the save button.
5) When Jane has finally created all of her tiles, she then begins to arrange them in different patterns. When complete with each pattern, Jane selects the option to add the pattern to the tile set. With each pattern, the program asks Jane to set the 'correctness' level of each. When satisfied she hits the save button.

5.5.2 Use Cases



Use Case Name:	**StartProgram**
Participating Actors:	User
Entry Conditions:	The user is at a pc with internet access and ready to begin using the system.
Exit Conditions:	The user is connected to the system and ready to begin using it.
Flow of Events:	

1) The user opens a web browser and browses to the page where the system is located.
2) The user starts the applet up.
3) The user is greeted with an empty grid screen and empty playarea screen.
4) The user then clicks on the file menu. There he or she has the option to load a tile set or open a recording.
5) If the user opts to begin with a preloaded set, he or she is then prompted to select one of the preloaded sets to begin using it. The system enters Tile Arrangement Mode. If the user selects to create or modify a set, the system enters Creator Mode. If the users opts to view a results file, the system enters Playback Mode



Use Case Name: **ArrangeTiles**

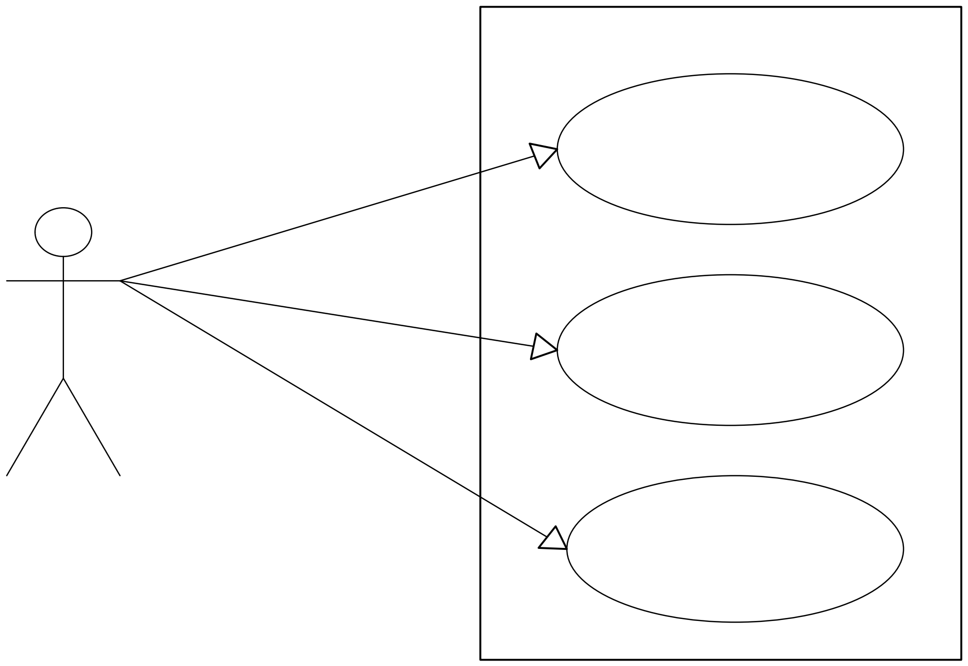
Participating Actors: User

Entry Conditions: The user is in Tile Arrangement Mode

Exit Conditions: The user is ready to save final arrangement.

Flow of Events:

1) The system loads up a grid for a large portion of the screen. Off to the right, the system loads up the playarea. The system then dumps all the tiles in the current set into the playarea in a random arrangement.
2) The user can now begin to arrange tiles. In the playarea, the user can drag and drop tiles rearranging the list how they wish. The user can also drag and drop tiles from the playarea onto the grid area. Tiles in the grid area can be moved from any grid position to any other grid position or back into the playarea. The user's goal is to eventually arrange the tiles into a specific arrangement depending on the tile attributes.
3) If the user feels that they need any additional tiles to complete their set, they can select the option to add a blank tile, which calls AddBlankTile. Likewise, if the user feels that they have certain tiles that do not belong in their set then they can leave them in the playarea excluding them from their final arrangement.
4) When the user decides that he has arranged the tiles correctly he can save his final arrangement.

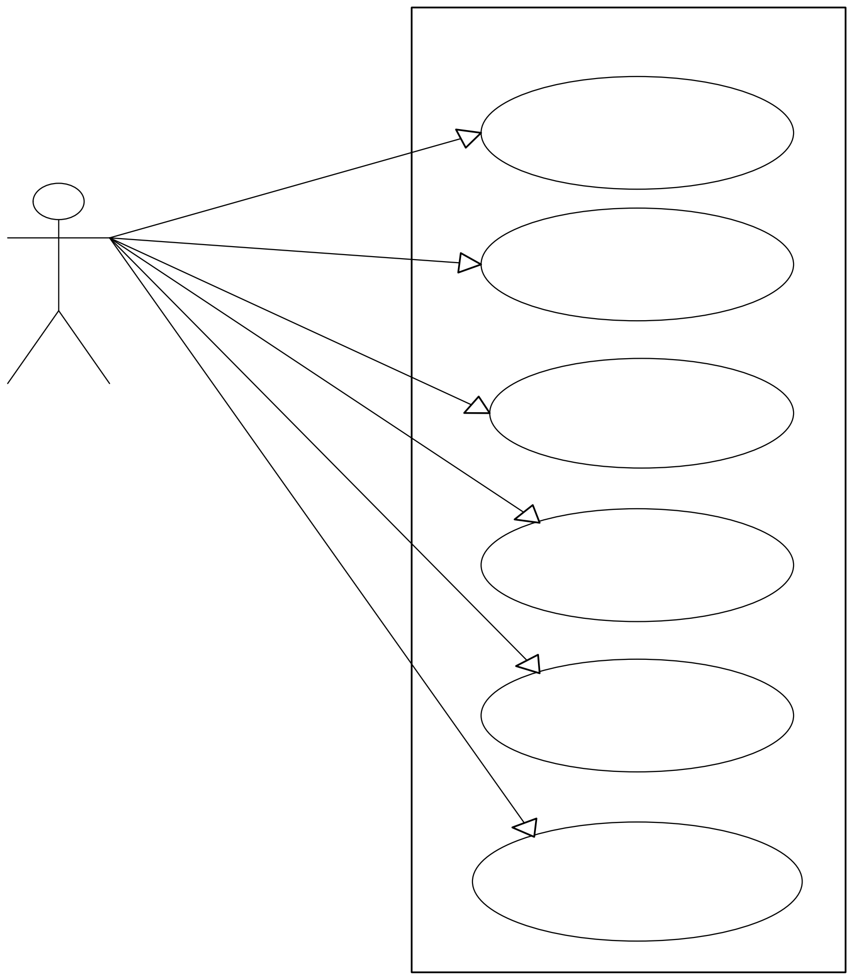


Use Case Name: **AddBlankTile**

Participating Actors:	User
Entry Conditions:	The user is in Tile Arrangement Mode and selects the option to add a blank tile.
Exit Conditions:	The blank tile has been added to the current set or the user has canceled.

Flow of Events:

1) The add blank tile dialog box appears.
2) The user is prompted to give a description of the tile (give information concerning the attributes of the tile, which vary depending on the set).
3) Once the user feels he or she as given a good enough description of the blank tile, they click the add button and the blank tile is added to the free-floating area. Alternatively, the user can click cancel to abort the blank tile addition.



Use Case Name: **PlaybackResults**

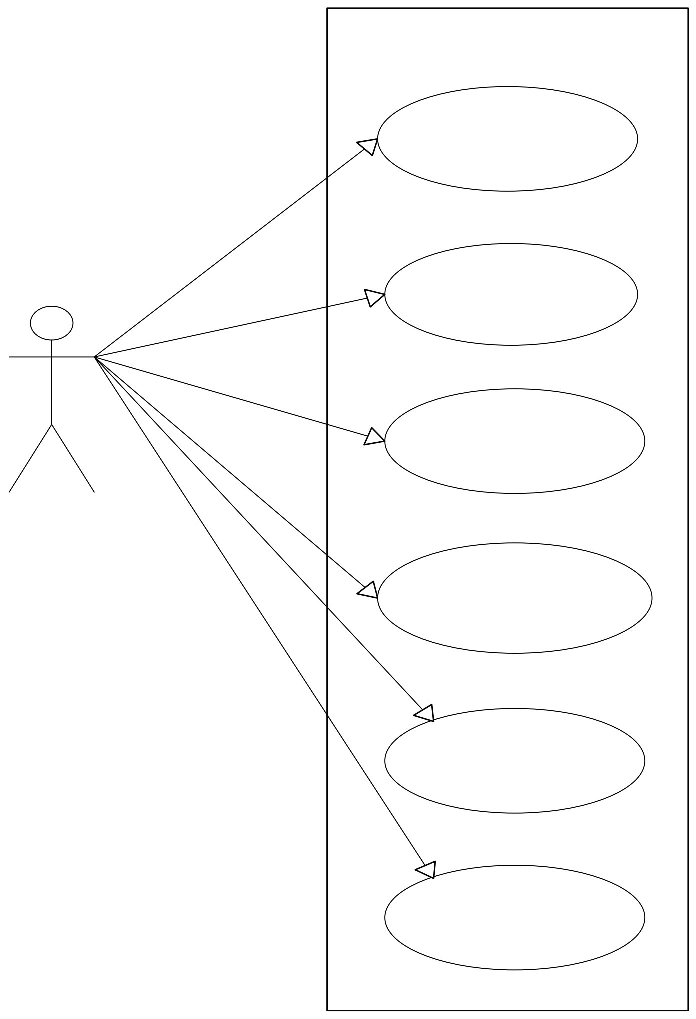
Participating Actors:	user
Entry Conditions:	The system enters Playback Mode
Exit Conditions:	The system leaves Playback Mode

Flow of Events:

1) When the system enters Playback mode, it prompts the user to select the file to view.
2) The user browses to the file he or she wishes to view and selects it. The user can also cancel which leaves Playback Mode and returns the system to the starting menu.
3) The system loads up and analyzes the file.
4) The system displays the tile arrangement starting from the first move. The user can move forward or backward one move at a time with the forward or backward buttons. The user can also skip forward and

backward a specific number of moves. The user can put it in auto mode, which plays the moves one after another without input from the user. Auto mode can move forward or in reverse at a few different speed levels. The user can also opt to skip to a specific move number. The system displays the total time that has passed since the last move as well as the total time that has passed since the first move.

5) When the user has viewed the results file as much as he or she wishes, they can select the option to leave Playback Mode and then the system takes them back to the starting menu.



Use Case Name: **TileSetEditor**

Participating Actors: user

Entry Conditions: The system enters Creator Mode.

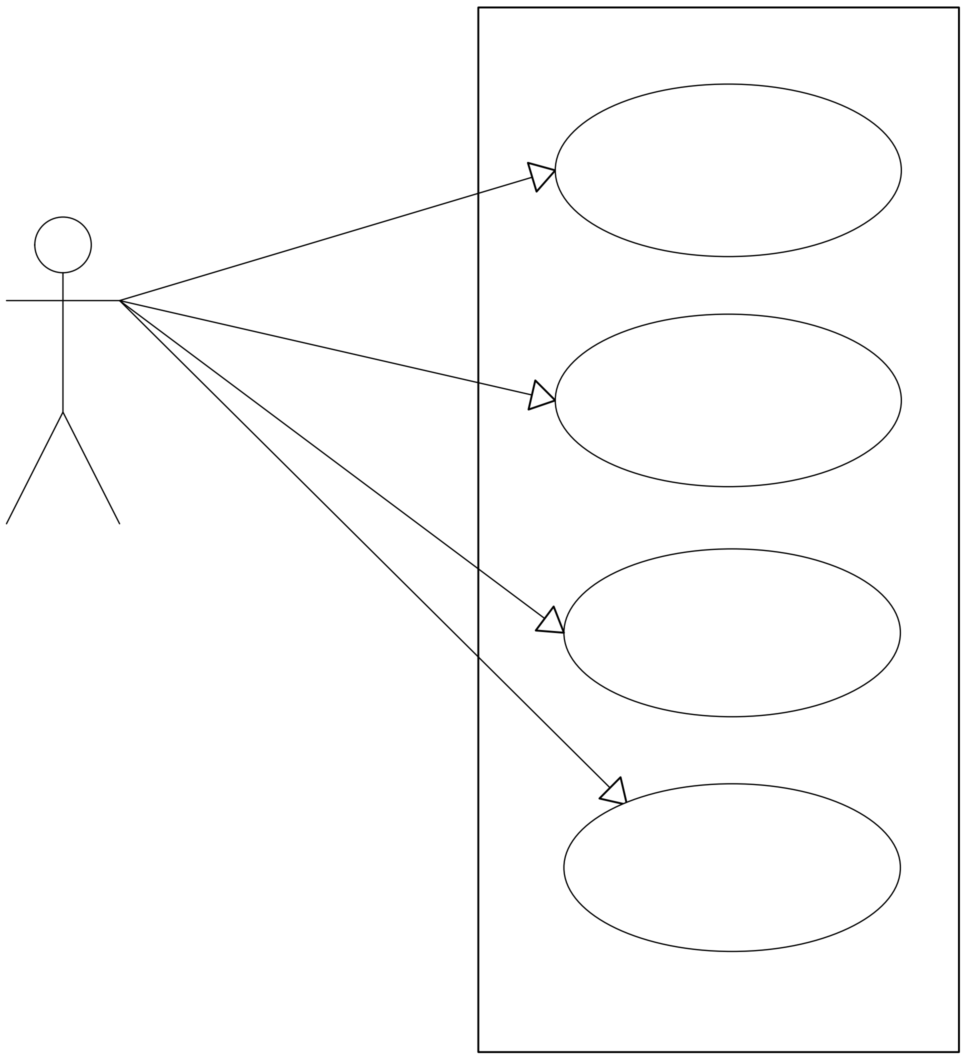
Exit Conditions: The system leaves Creator Mode.

Flow of Events:

1) After entering Creator Mode, the user has the option to open an existing set to edit or to create a new set.
2) Selecting to create a new set calls EditTileAttributes. Selecting to modify a set prompts the user to select an existing set to modify. When the user selects a set, it

loads up all tiles associated with that set into the playarea area.

3) From here, the user has several options:
    - Delete a tile – Calls DeleteTile
    - Create a new tile – Calls CreateNewTile
    - Add a solution pattern to the set – Calls AddPattern
    - Edit the rule set for a set – Calls EditSetRules
4) The Instructor has the option to save the tile set or return to the starting menu forfeiting all changes he or she has made.
5) If the user chooses to save the tile set, the system will prompt him or her to enter a filename and choose the location (on the local PC) to save the tile set.
6) The user then has the option to continue in Creator Mode or exit to the starting menu.



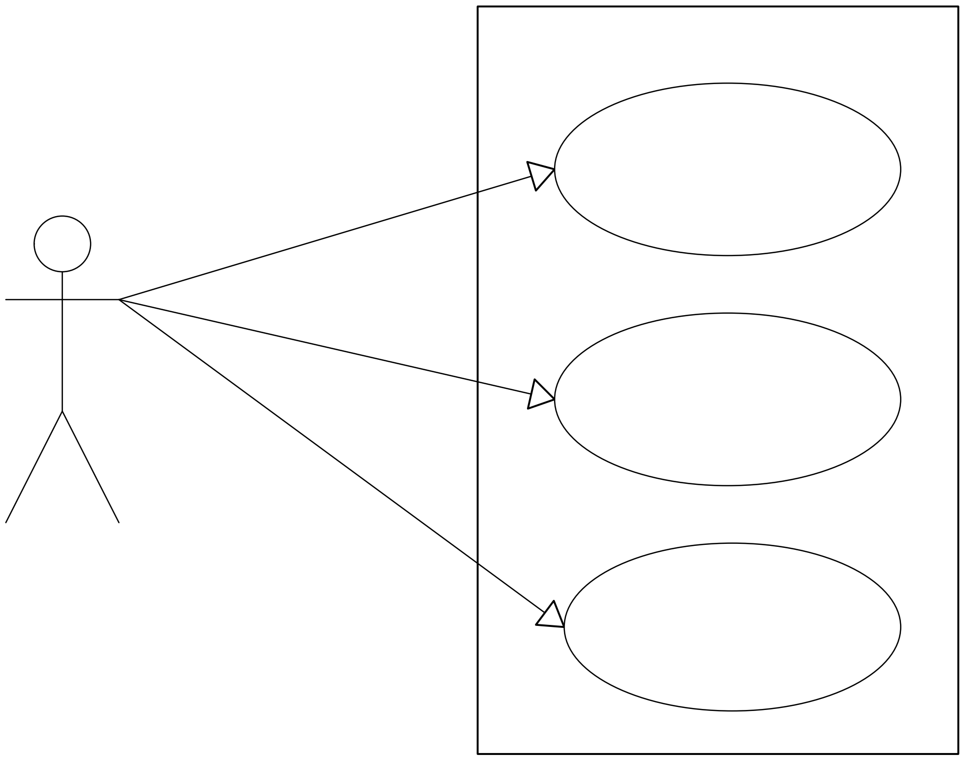
Use Case Name: **DeleteTile**

Participating Actors:	user
Entry Conditions:	The system is in Creator Mode and has a tile to be deleted.
Exit Conditions:	A Tile has been deleted or the operation is canceled

Flow of Events:

1) The user selects the tile to be deleted from the list of tiles in the set.
2) The user selects the delete option.
3) The system then prompts the user asking if he or she is certain that they want to delete the tile.

4) If the user selects yes, the tile is removed from the tile set. If the user selects no, the operation is aborted. The user can then save or cancel.

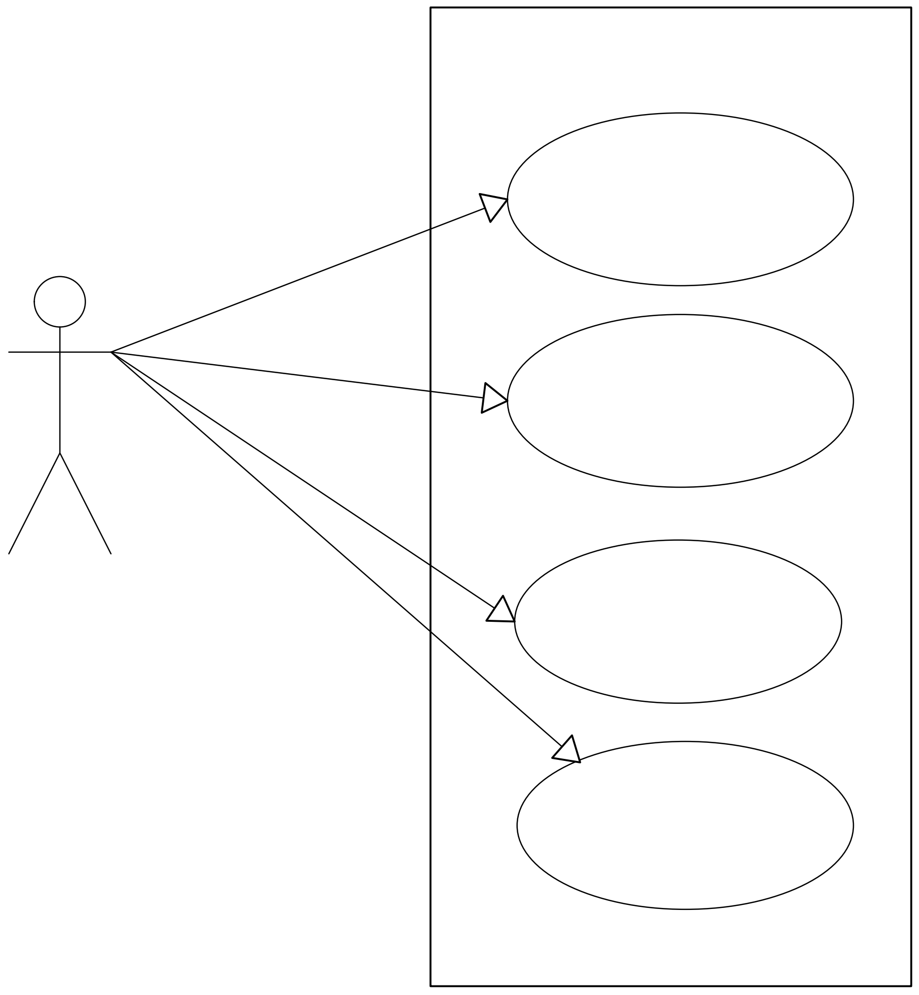


Use Case Name: **CreateNewTile**

Participating Actors:	user
Entry Conditions:	The system is in Creator Mode.
Exit Conditions:	A new tile has been created or the operation is aborted.

Flow of Events:

1) The user selects the option to add a new tile to the set.
2) The user uploads an image to serve as the tile.
3) User saves or cancels tile set.



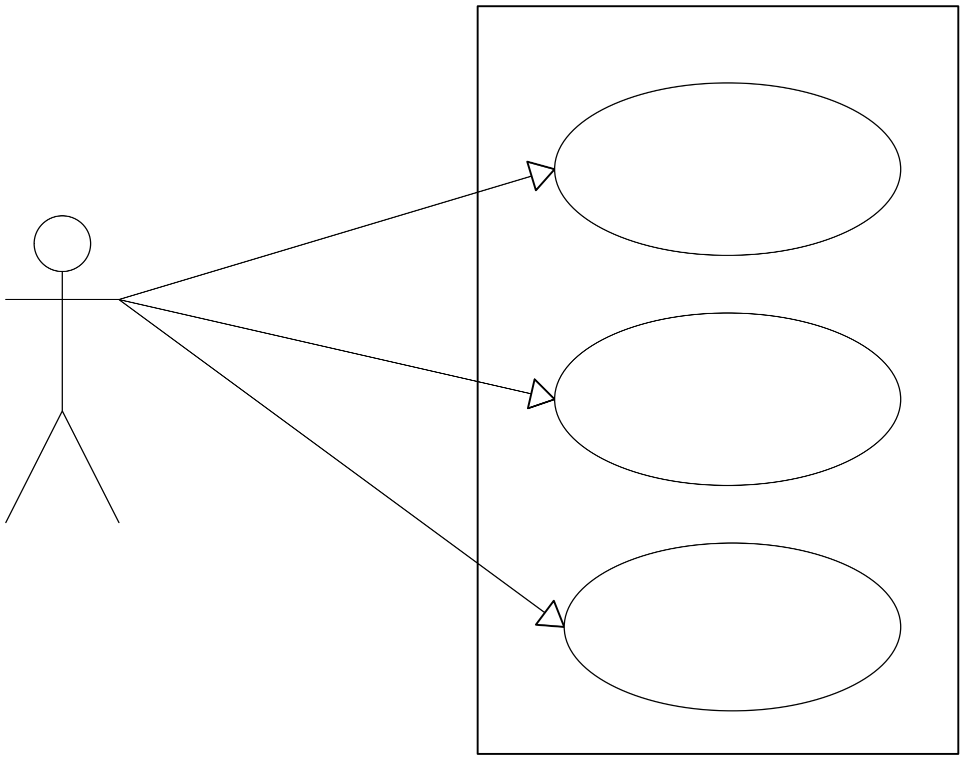
Use Case Name: **AddPattern**

Participating Actors:	user
Entry Conditions:	The system is in Creator Mode and at least 1 tile is in the set.
Exit Conditions:	A new pattern has been added to the solution set for the current tile set or the operation has been canceled.

Flow of Events:

1) The user arranges the tiles in the grid area until they form the pattern they wish to add to the tile sets solution set.
2) The user then selects the option to add the pattern to the solution set.

3) The system prompts the user to enter the ‘correctness’ rating of the current pattern as well as a name for the pattern.
4) The system then prompts for confirmation. If save is selected, the pattern is saved in the solution set for the tile set. If cancel is selected, the pattern is not added and the system returns to creator mode.



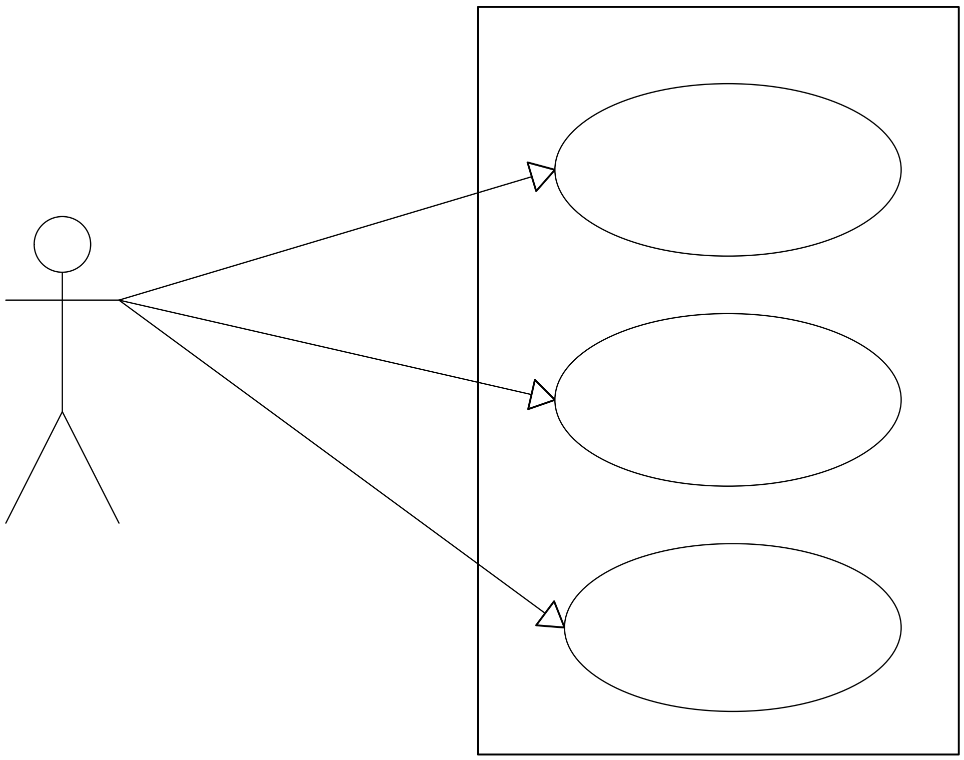
Use Case Name:	**RemovePattern**
Participating Actors:	user
Entry Conditions:	The system is in Creator Mode.
Exit Conditions:	A pattern has been removed from the set’s solution set, the operation has been canceled, or the user opts to view a pattern.

Flow of Events:

1) At any time in Creator Mode, the user can select the option to remove a pattern from the set’s solution set.
2) A list of patter names is then displayed. The user has the option to cancel and return to Creator Mode or they can select any one pattern and select to view the pattern

or remove the pattern. If the view pattern is selected, RemovePattern exits and the tiles are arranged in the grid of Creator Mode to match the pattern that was selected.

3) If the user opts to remove a pattern, a prompt asks for confirmation. No will return to step 2. Yes will remove the selected pattern from the set and then return to Creator Mode.



Use Case Name:	**ModifySetRules**
Participating Actors:	user
Entry Conditions:	The system is in Creator Mode
Exit Conditions:	Set rules have been modified or the operation has been canceled.

Flow of Events:

1) The user can select the option to modify the tile sets rules while in Creator Mode.
2) This brings up a list of tile set rules and their current settings. The user can edit the rule settings at will.

3) The user then has the option to cancel and return directly to Creator Mode or to save the changes to the set rules and then return to Creator Mode.

6. Proposed System Design

6.1 Design Goals

6.1.1 Dependability Criteria

Reliability

The system must accurately handle and provide information to the end user.

Security

The system must be secured against unauthorized users modifying a user's local hard drive and it must inform the user of the need to read/write their hard drive by providing a signed certificate.

Robustness

The system should perform error checking on all inputs in order to catch invalid inputs and deal with them in such a manner to prevent the program from crashing or disrupting the user's work. An example would be preventing the user from loading up a file for playback that is not actually a log file created by the system.

Availability

The system should be accessible to students and professors. It may also be used by other teachers and possibly common non-related users.

6.1.2 Performance Criteria

Response time

The system should take less than one minute to download to the user's PC on a dialup connection. Once the system is downloaded to the user's local PC, it should respond to all user input without any visible delay.

6.1.3 Maintenance Criteria

Modifiability

The system needs to be designed and implemented in an efficient object oriented manner in anticipation of any interface or functional application updates to the system.

Readability

The system layout, design, and code needs to be easy to navigate and understandable by any developer who would later be modifying the code or adding additional features. This can be accomplished by utilizing clear coding standards and module descriptions in the header part of the code.

6.1.4 End-user Criteria

Usability

The system should be usable to a wide range of users from novice to expert, and should be intuitive.

6.1.5 Design Trade-offs

This C.L.I.P. system as it is being designed and developed has several design tradeoffs to deal with.

Tile size vs. Screen size.

A main User Interface tradeoff that we needed to consider and work around is tile size vs. screen size. We needed to keep the tiles large enough so that all of the information on each tile could be seen at one time. This caused some conflict because the larger and clearer the tiles were, the less of them we could display on the screen at once.

The size of the tiles also brought up conflicts with the size of the grid area. We needed the grid area to be large enough to accommodate the patterns of the set. The patterns could be several tiles long or high. We could not afford to make the tiles smaller so we needed to add the functionality of the scrolling grid area to accommodate all different kinds of patterns.

The tradeoff to make the grid larger then can be displayed at once forced us to trade the simplicity of our program and add a way to see the entire pattern at once by means of a mini-map.

Using Java

The use of java is another design tradeoff that our group needed to make. We needed to consider the difference between functionality and the learning curve associated with learning the new language. The time that it takes to learn java was weighed against the need for additional functionality and the deadline we had for our project.

After deciding to use the java language we had to determine the pros and cons of using an applet vs. application. The applets ability to be online-based and used from a web browser helped us determine that it would be better suited for our program. It seemed much better to allow the user to use the applet online instead of forcing to download an application for them to use.

Security

Security in the read-write of files was an important Access and Control Security consideration for the group. We needed to determine who we should allow to use the system and what we would allow them to do. The security of the system was sacrificed in order to make the program available and easy to use. We decided that instead of trying to keep the applet secure by requiring a logon or some password, we would allow all users to use the program.

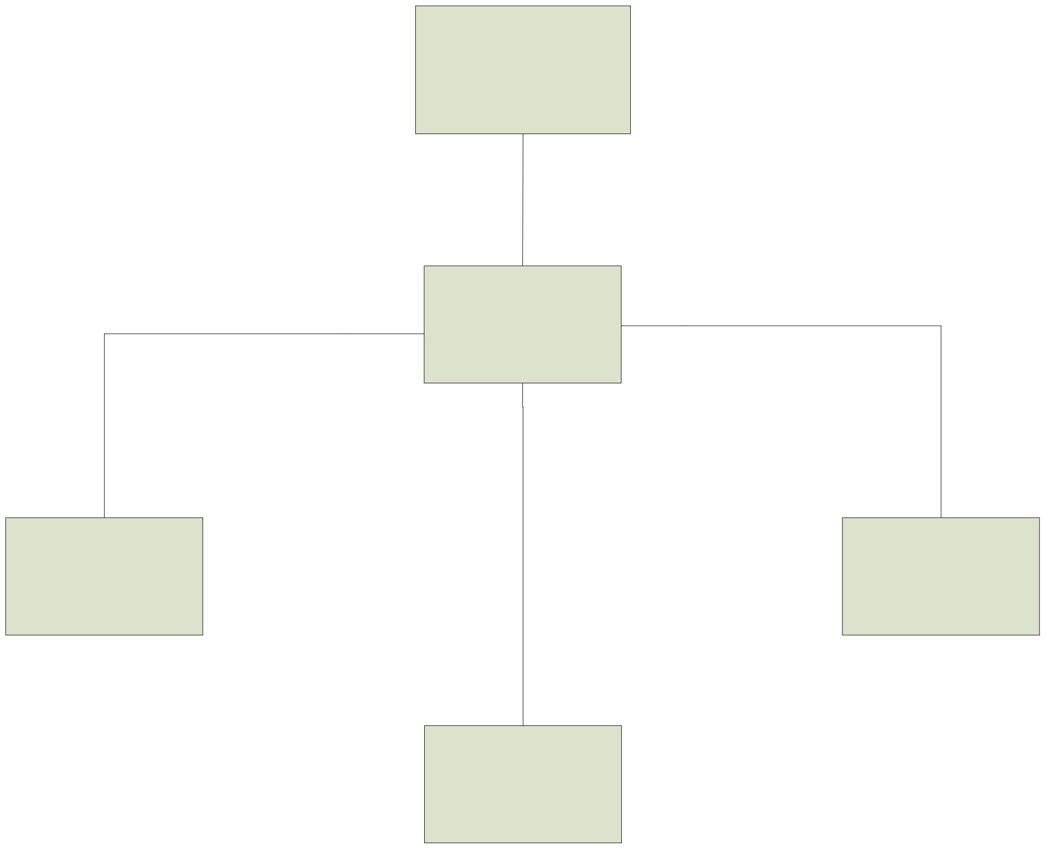
We also decided that we would allow the students and teachers to have the same access to the functionality of the program. This allowed us to only worry about developing a single interface for all users.

Another security tradeoff that we needed to worry about was the submission of the results. We could have allowed the program to submit all the results to the professor, however this could lead to many complications for the professor due to users of the system that are not in her class. This would also complicate allowing other professors to use the system. So we decided that it would be better to have the submission of the results to the professor separate from the program. This means that the student would need to email the recoded file to the professor or give it to her on a disk.

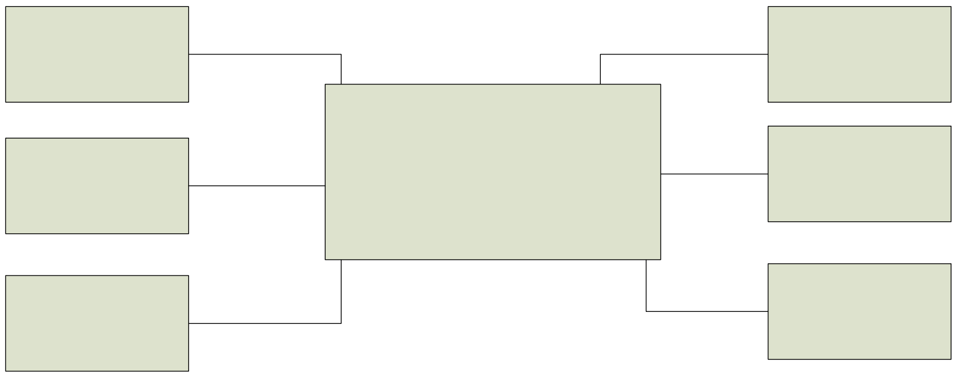
6.2 System Overview

6.2.1 System Decomposition

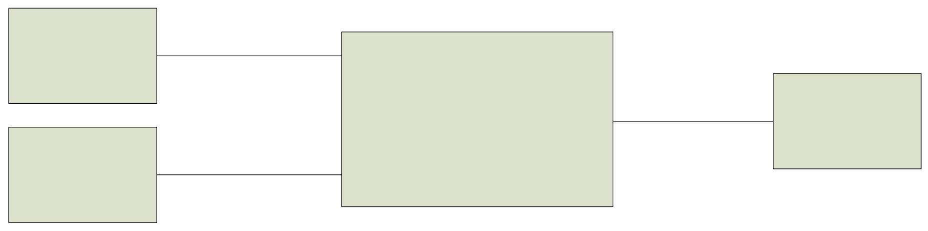
Complete System



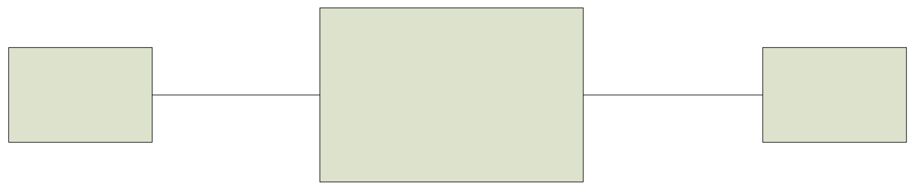
Tile Arrangement Module



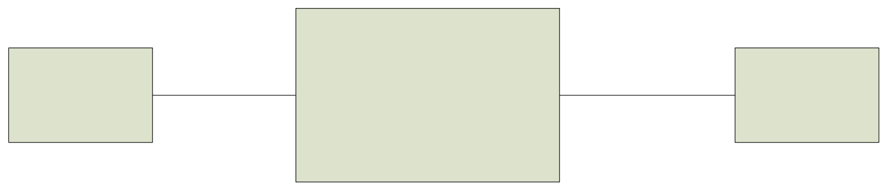
Data Module



Grid Module

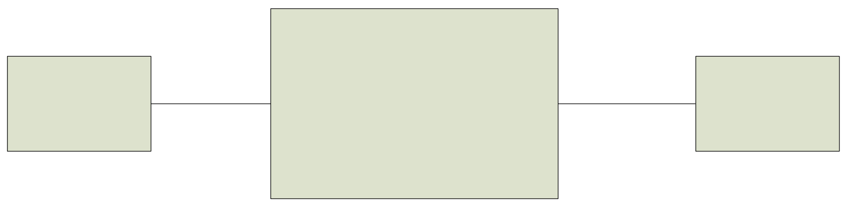


Play Area Module



Ti

User Interface Module



There are 5 modules that make up our system. They are the User Interface module, Tile Arrangement module, Grid module, Play Area module, and Data module.

Grid Module

The Grid Module is responsible for displaying and storing the tile placements that occur by the user as they carry out the task of tile arrangement.

Play Area Module

The Play Area module is responsible for displaying and storing a listing of tiles not yet used by the user.

S

Data Module

The Data Module deals with the creation and modification of the set of tiles used by the system.

Tile Arrangement Module

The Tile Arrangement Module brings together the data module, grid module, and play area module allowing for the data created and loaded from the data module to move between the grid module and play area module freely. This also allows the tile positions in both the grid and play area modules to be logged for use by the playback component. The action of logging the positions is handled by the record component. Lastly, the mini-map component handles displaying a miniature version of the entire grid.

User Interface Module

The User Interface module handles the visual placement of components that the user can interact with. This includes the visual component of the grid, play area, playback controls, and mini-map. It also provides a way to load, save, create, and modify (through the menus) all of the data used by the system.

6.2.2 Subsystem Interfaces

Tile Arrangement Module

The Tile Arrangement module is the main module of the system that ties the all the modules together. Communication between the other four modules is handled by the Tile Arrangement Module. This module receives mouse events from the User Interaction module and passes them along to their intended module. This module sends and receives coordinates to and from the Grid and Play Area in order to handle tile movements between those two modules. The Data Module provides this module with set rules, tile information, and pattern information. The Tile Arrangement module also sends coordinates back and forth from the grid to the minimap. During recording, this module sends every tile movement to the Record module to store moves. During playback, this module receives tile moves from the Playback module.

User Interface Module

This module sends mouse events created by user mouse input to the Tile Arrangement module.

Grid Module

This module informs the Tile Arrangement module of every move it makes so that the moves may be recorded and the mini-map gets updated. It also passes coordinates along to the Tile Arrangement module when moving tiles to the play area as well as receives coordinates when moving tiles from the play area to the grid.

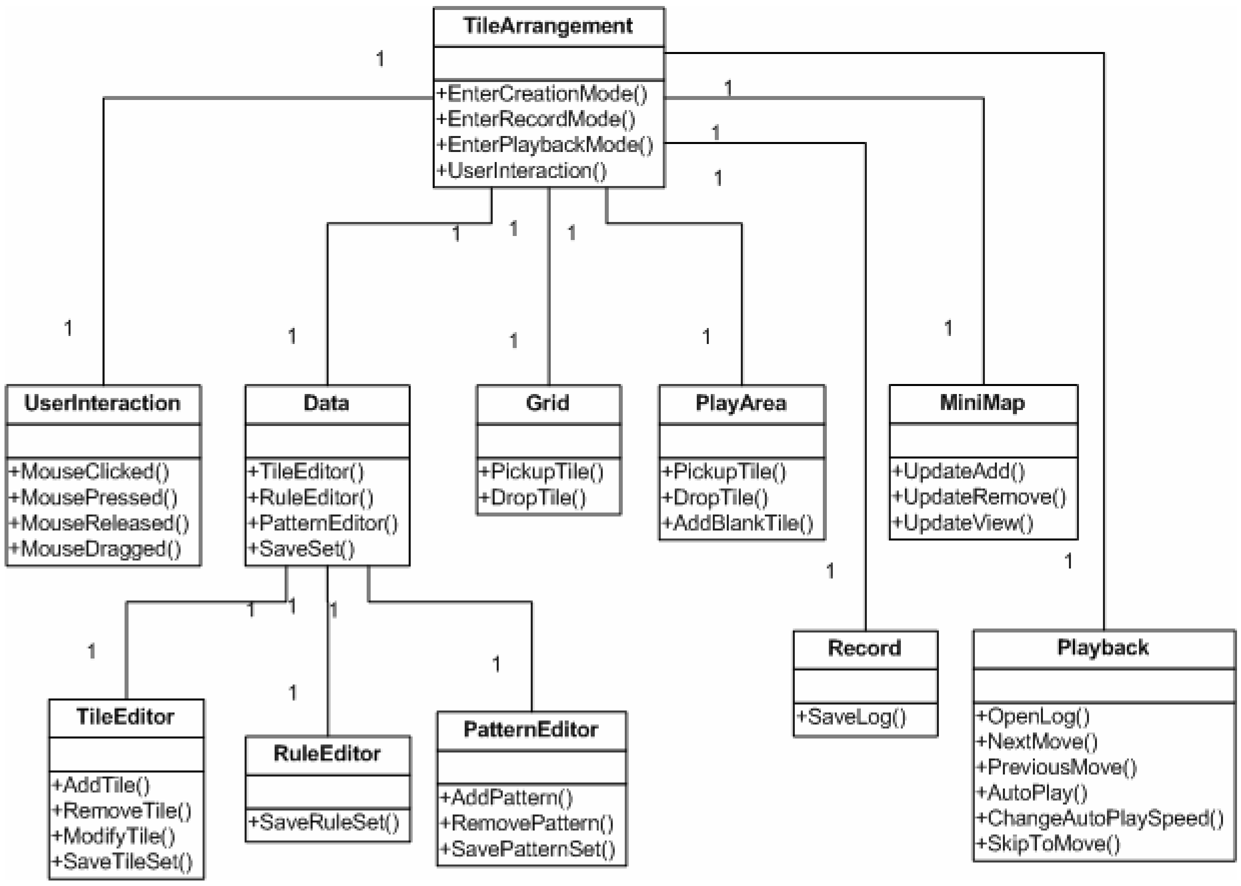
Playback Module

This module informs the Tile Arrangement module of every move it makes so that the moves may be recorded. It also passes coordinates along to the Tile Arrangement module when moving tiles to the grid and it receives coordinates when moving tiles from the grid back to the play area.

Data module

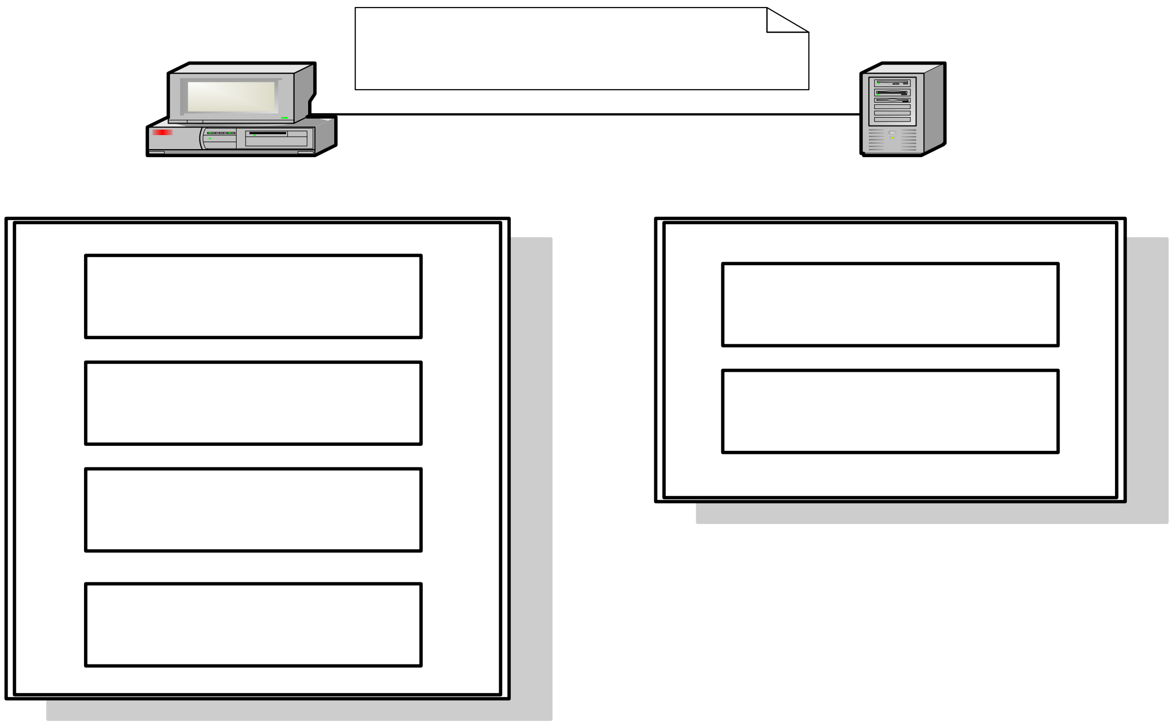
This module provides the Tile Arrangement module with information related to the current set. This information includes the set rules, the tile information, and the pattern information.

6.2.3 Class Diagram



6.2.4 Hardware/Software Mapping

The CLIP system is online and requires that the user have a Java technology-enabled browser. All interfaces for the system will be stored on the server, with the system's code being executed by the browser's Java Virtual Machine. The system will have access to the hard drive of the user's PC so that two files may be stored in secondary memory.



6.2.5 Persistent Data Management

The Clip system will utilize the following files:

<u>Log file</u>

The log file will store the list of moves made by the user. During playback, this file will be read by the system in order to play back the user's moves.

<u>Rule Set file</u>

The rule set file will store rules and options that define a tile set. Information to be stored consists of set name, set description, tile size, ability to add blank tiles, and if pattern checking is enabled.

<u>Tile file</u>

The tile file will store a list of tiles, their unique integer id, and the path to the tile image.

<u>Image files</u>

The image files are the GIF and JPG images used to represent the tiles.

<u>Pattern file</u>

The pattern file will store all possible solutions that have been added to the tile set.

6.2.6 Access Control and Security

CLIP is an online system that is accessible to anyone with an internet connection and java enabled browser. There will be no restriction the systems functionality regardless of the user. Security is only necessary on the server-side where the applet is signed indicating that any read/write action taken by the applet is appropriate.

6.2.7 Global Software Control

The system is event driven. The actions taken by the users either by dragging/dropping a tile, pressing a command button, or selecting menu option, will determine which sub system is activated. If there is no action taken by the user then the system is idle.

6.2.8 Boundary Conditions

Initialization

The system will be a java applet. It will be hosted at a location of our client's choice and accessible to users via a java enabled web browser. It will allow the user to save their work as individual files to their own computer and also allow users to view the playback given a saved file.

Termination

The system will be shut down by exiting the web-browser.

Failure

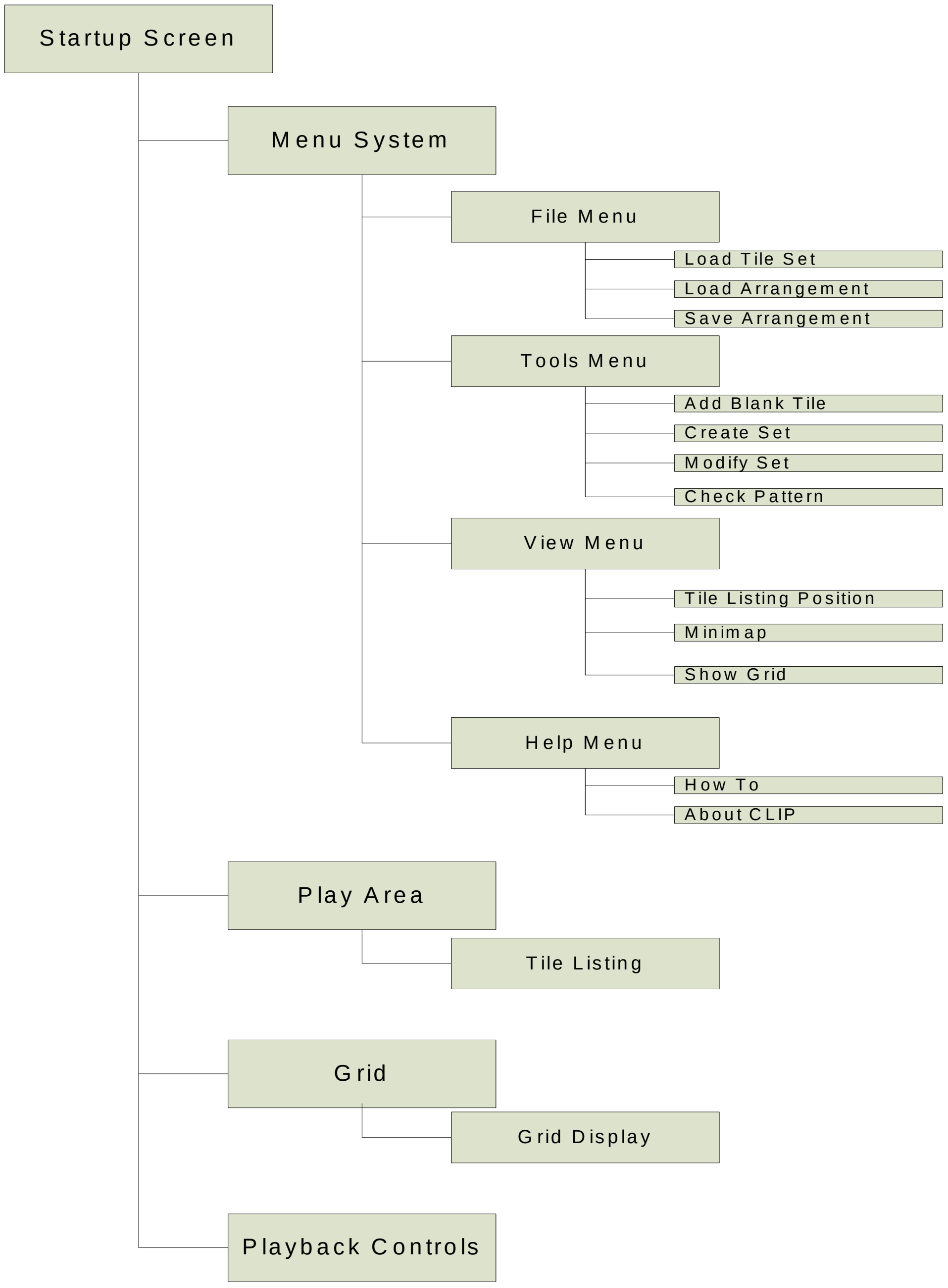
After an unexpected client side failure, the system will display a brief, descriptive error message to the user before termination.

6.3 User Interface

The User Interface of the C.L.I.P system is important in recreating the look and feel of current system's physical environment. Furthermore, the our system also focuses on providing ways to easily create/modify tile sets, load existing tile sets up for use in tile arrangement, and allow for easy play back of previously done tile arrangements.

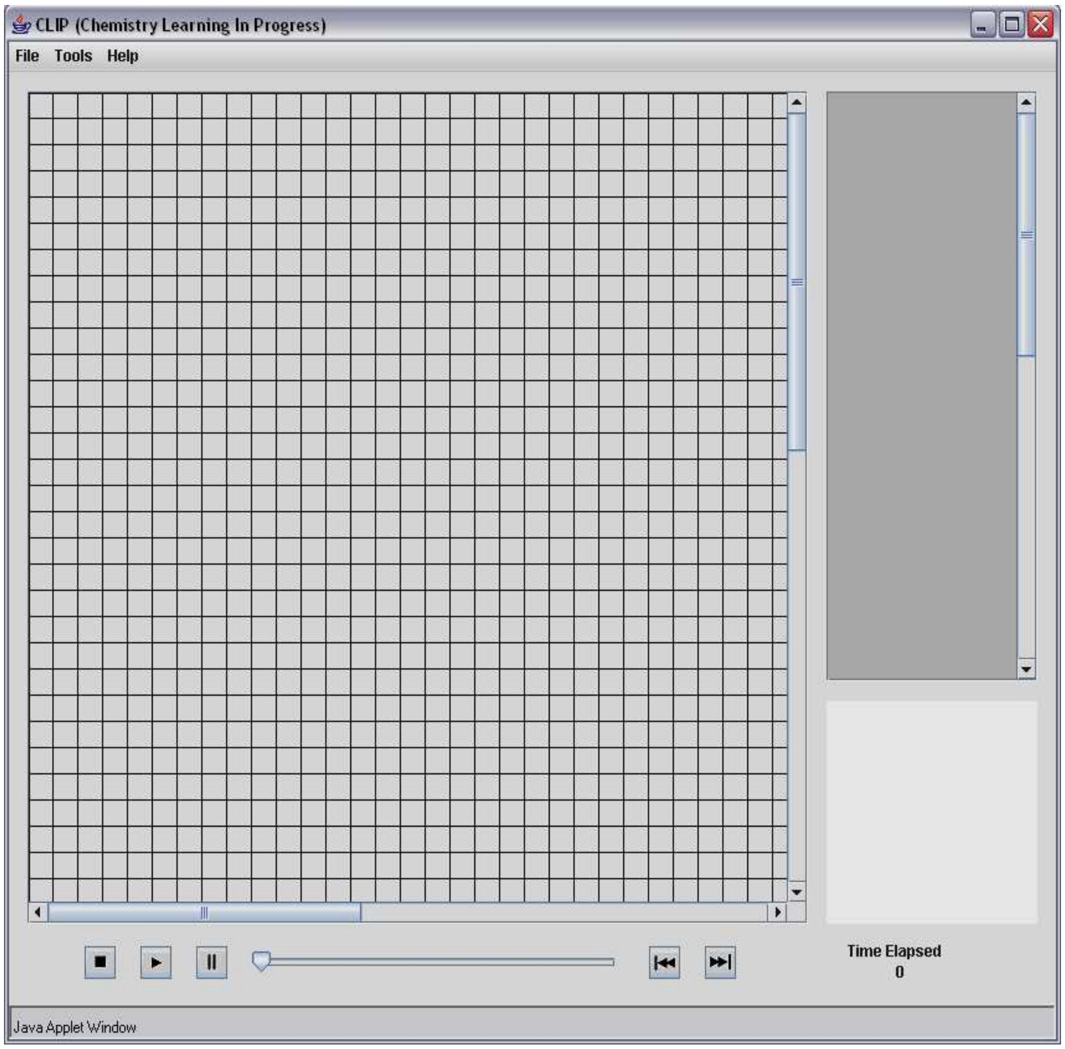
Hierarchical View

The hierarchical view shows the hierarchy of controls for the user interface where at the top level is the startup (main) screen. Directly attached to it [startup screen] is the menu system, play area, grid, and playback controls. From there, each of those components consist of sub components that carry out the larger component's purpose.



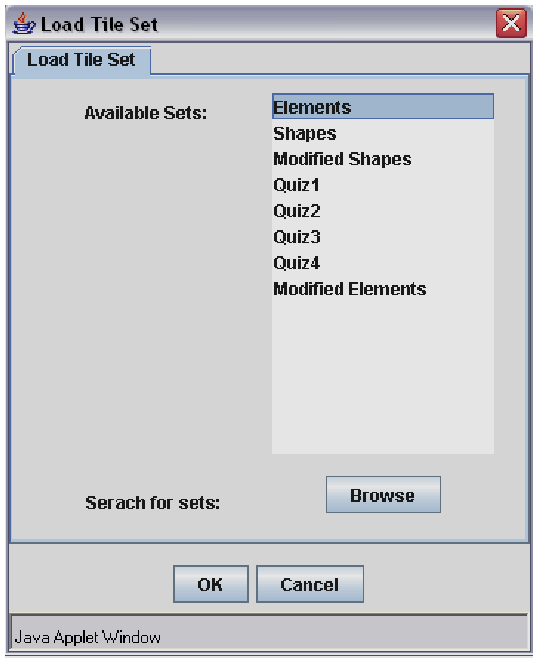
Main Screen – Initial Look

After the web page containing the applet has successfully loaded, the main screen will appear in a frame outside the web browser. This screen will contain a menubar, an empty grid, the playback controls, and an empty tile listing.



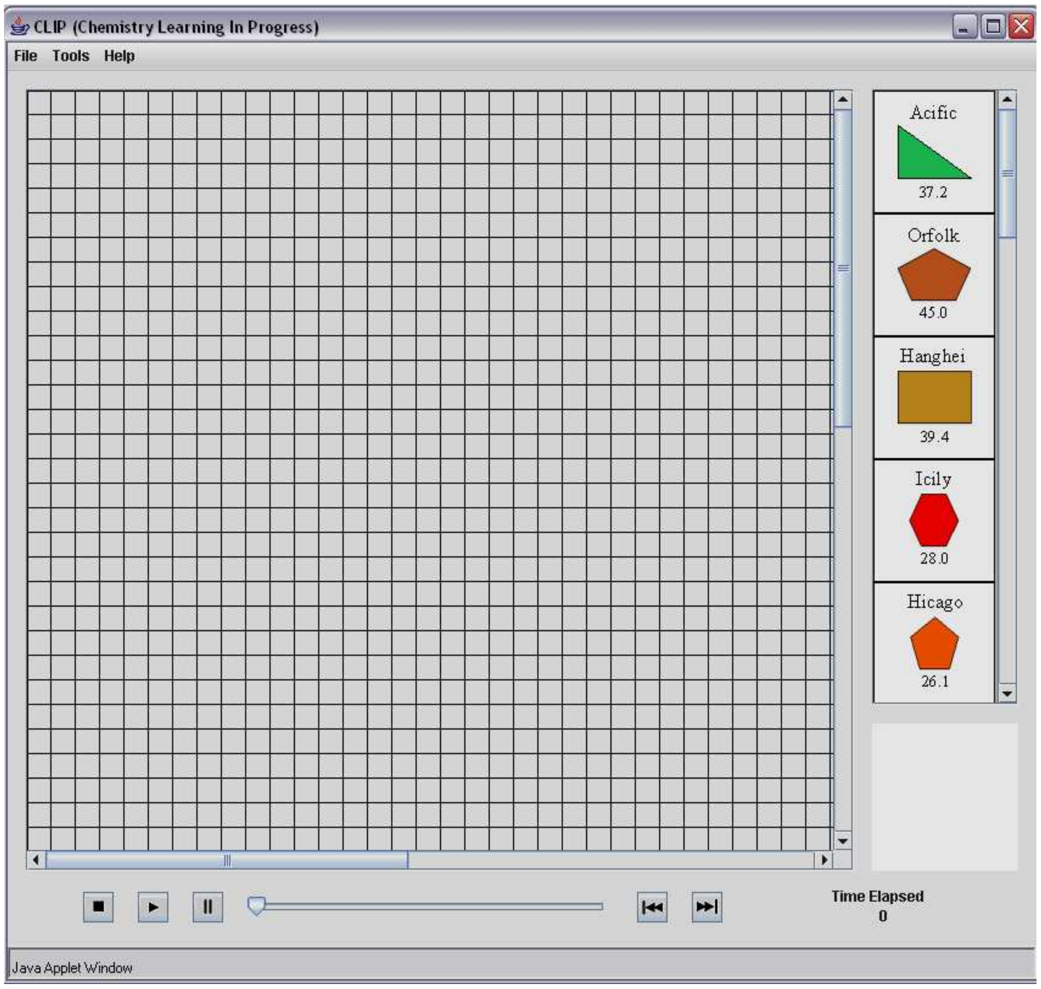
Load Tile Set Dialog

After a user has selected load tile set – under file menu – a load tile set dialog will appear. The dialog will display a listing of preloaded sets. These are the sets that are available from the web site hosting the applet. The user also has the option to browse for additional sets that may be located elsewhere.



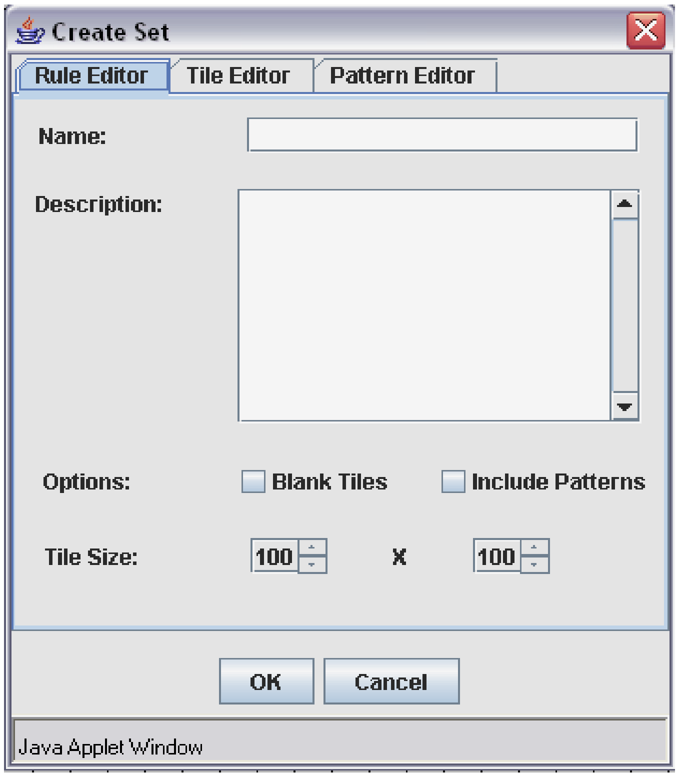
<u>Main Screen – Tile Set Loaded</u>

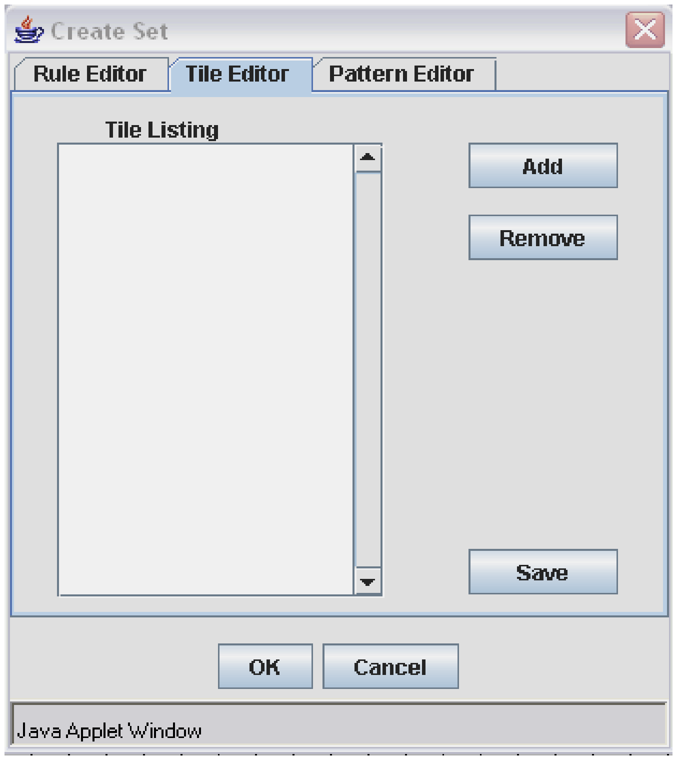
After the user has selected a tile set, its contents will be loaded into the tile listing to the right of the grid.

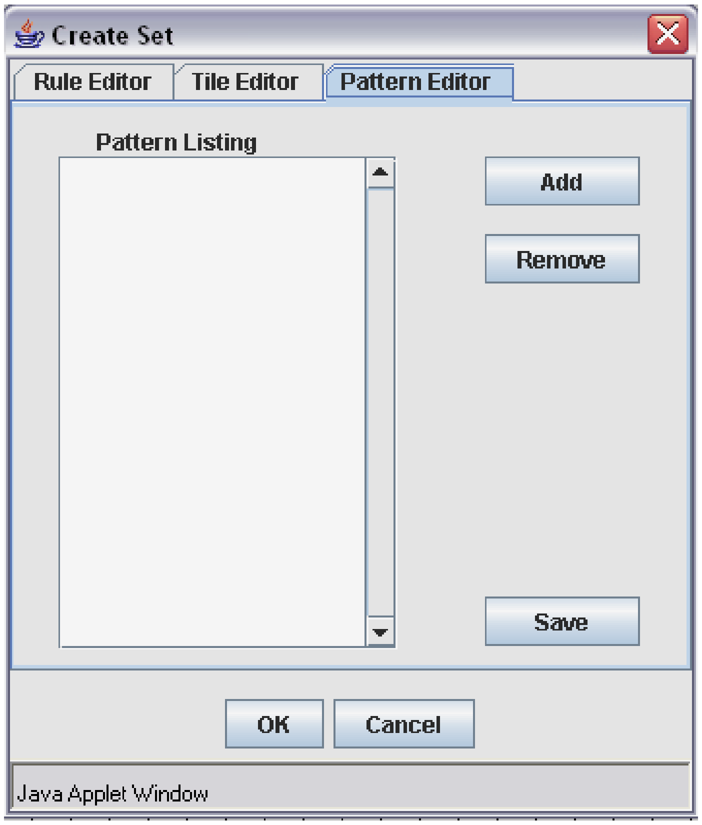


<u>Create Set – Rule Editor, Tile Editor, Pattern Editor</u>

When a user wishes to create a new set, they can select from the menu bar, Tools -> Create Set. A Create Set dialog will appear with 3 tabs. The first two tabs – Rule Editor & Tile Editor – are required, while the third, Pattern Editor, is not necessary. The rule editor outlines all the characteristics that define a certain set. The tile editor tab allows a user to start adding tile images to the set they defined under the rule editor. The last tab called pattern editor is an option for the user to decide what solution(s) will be included with the set so that pattern checking can be used.

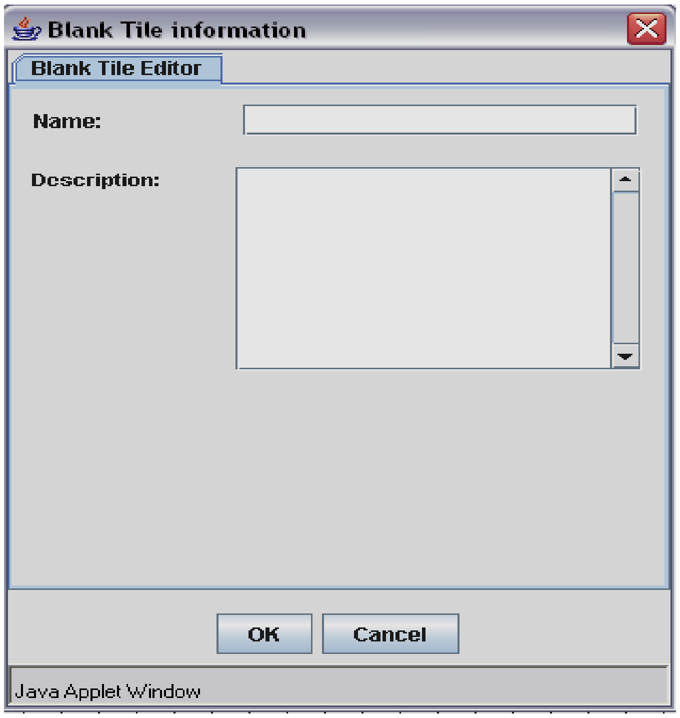

Create Set
Rule Editor
Tile Editor
Pattern Editor
Name:
Description:
Options:
Blank Tiles
Include Patterns
Tile Size:
100
X
100
OK
Cancel
Java Applet Window


Create Set
Rule Editor
Tile Editor
Pattern Editor
Tile Listing
Add
Remove
Save
OK
Cancel
Java Applet Window



<u>Insert Blank Tile</u>

For some sets it is allowable to add blank tiles that take the place of tiles that are not in the set, but that the user believes need to be in order to fill in the gaps of a pattern. To add a blank tile, a user will select from the menu bar Tools-> Add Blank Tile and a dialog with appear. In this dialog, the user will describe the characteristics of the missing tile.



6.4 Packages and File Organization

Tile Files

The tile files are sets of picture files that are to be loaded by a user to be arranged on the grid. The system will begin with several different sets of tiles for use by the users. To allow the expansion of the program the system can have additional tile sets created. The tile file will store a list of tiles, their unique integer id, and the path to the tile image. This file will be created from the create set item in the tools menu.

Log Files

The log file stores information about all of the moves that were made including all of the blank tiles that were added while the user was using the program. This file will be created when the user is in creation mode and selects the "Save Recording" tab in the file menu. The menu will pop up a save file dialog box and allow the user to save the file to the local hard drive. This file will also be used in the Playback mode for file reading. The file will be accessed when the user selects the "Open Recording" tab in the file menu. The program will read all of the moves that are saved on the file and display them one at a time. All entries will be time-stamped with the time (hours:minutes:seconds) since the last action.

The file will have 4 different moves: (gg) when a tile moves from the grid area to a different position in the grid area, (gp) when a tile moves from the grid area to play area, (pg) when a tile moves from the play area to the grid area, (pp) when a tile moves from the play area to a different place in the play area. These actions will all have the same format:

Move type, tileID, initial position, final position, timestamp;

The program also needs to save other types of actions that take place such as add blank tile (ab) which will have the description that is entered by the user and this event will have a format of:

Move type, tileID, initial position, name, description;

Another move type is when the user enters a reason for leaving out a tile from a set. This move type (lo) has the format:

Move type, tileID, Reason, timestamp;

The last type of action that needs to have a move type is when the user is done with all moves and finishes saving the file (ed). All we do is make a timestamp of the event with format of:

Move type, timestamp;

Example of a Log File:

```
pg  4  0,3  40,60  0:1:12
pg  6  0,4  200,300  0:0:44
gg  4  40,60  800, 400  0:1:01
gp  6  200,300  0,0  0:0:18
pp  6  0,0  0,7  0:0: 33
ab  23  0,22  blanktileName  "description"
….
lo  18  "this tile didn't belong"  0:1:05
ed  0:0:05
```

Rule Set Files

The rule set file has all of the information about a specific set. It has all of the options that are enabled or disabled in the set, such as the ability to leave out tiles, add blank tiles, and ability to check patterns. This file will also store the information about tile size, the name of the set, and the description and rules associated with the set. This file will be made when a user creates a new set. This is done through the create set tab in the tools menu. The user selects the option that should be enabled in the new set and the name and description of the set. This file can be edited by the modify set tab in the tools menu. It has a format similar to:

```
TileSetID
setName
setDescription
[width]x[height]
Blanktiles=[true:false]
Patterns=[true:false]
LeaveTilesOut=[true:false]
```

Image Files

The image files are the *.jpg or *.gif files that have the image of tiles on them. These are what will be used to display the information for the tile to the user. The user must create or find their own files for this purpose.

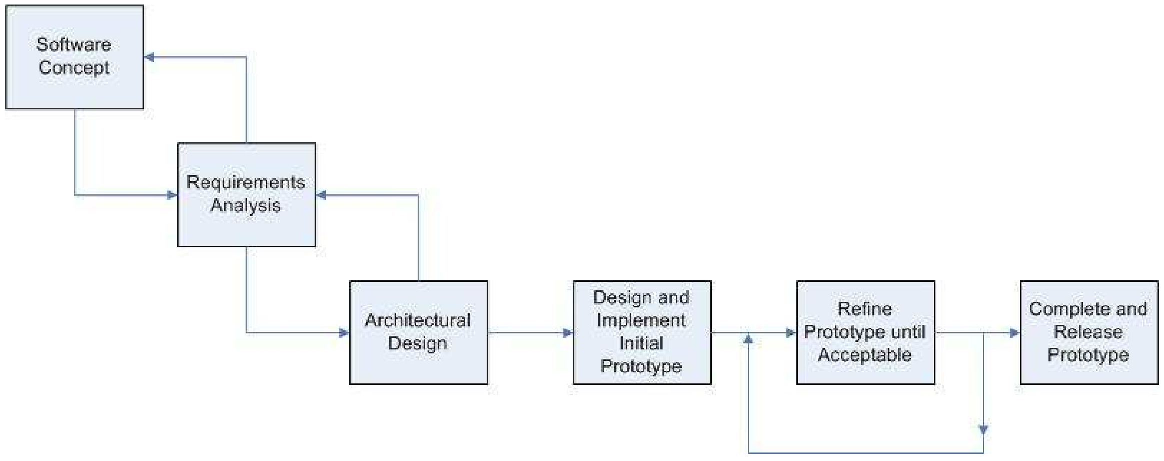
Pattern Files

The pattern files are the files that store information about the final arrangement of the tiles for a specific set. This file can be used to check to see if the solution the user has is one of the best solutions if the set allows it. This file will be created in the create set tab in the tools menu. The user will be allowed to set up certain final arrangements that would be considered correct. It would have a similar format to:

```
TileSetID
[numPatterns]
PatternName
PatternArray
```

7. Development Process

7.1 Process Model


Software Concept
Requirements Analysis
Architectural Design
Design and Implement Initial Prototype
Refine Prototype until Acceptable
Complete and Release Prototype

The lifecycle model that we will use for the entirety of project will be the waterfall model. However, within our coding and testing phases, we will gear our model to a more evolutionary prototyping approach.

Given the small team size and project milestone deadlines, the waterfall model provides a solid lifecycle that we can follow without needing to make compromises to our goals or deadlines. During the coding and testing phases of the lifecycle, our project will take an evolutionary prototyping approach. It is very important that the system be easily understood and used by the users. This will require a strong focus on providing a good user interface for users to interact with the system.

Therefore, the evolutionary prototyping will allow us to gather important feedback on each iteration of the prototype so that we may provide the best possible interface for the users.

7.2 Software Documentation and Review

We have several important milestones for the project as outlined below:

Requirements Analysis Document	09/26/2005
Requirements Analysis Presentation	10/05/2005
System Design Document	10/10/2005
System Design Presentation	10/12/2005
Contract First Draft	10/24/2005
Project Plan Document	10/26/2005
Project Plan Presentation	11/02/2005
Final Contract	11/09/2005
Prototype Plan	11/14/2005
Final Report	12/12/2005

7.3 Methodologies, Policies, Standards, Tools

Methodologies

The system will be developed in an object oriented, modular design as laid out in our Design Document. By keeping the system object oriented and modular, it will help in translating the current system to the new system as well as easily allow team members to work on different aspects of the system in parallel and seamlessly integrate them with one another.

Policies

Most of the code written for the system will be made by team members working on their own or in small groups within the group. Every member is expected to thoroughly comment and test their code on their own. After the code writer believes the code is ready, then it will be reviewed and tested by the team at a team meeting so that everyone understands the code and finds no problems or performance issues with it.

Standards

The coding standards and conventions that we will use for the project can be found at http://java.sun.com/docs/codeconv/html/CodeConvTOC.doc.html. These standards have been examined and we decided that they will work well for our project. Any modifications or additions to these standards that may arise are made by the Lead Programmer and followed by the rest of the team members.

Tools

Many tools have been and will be used for the project as outlined below.

Microsoft Word – Microsoft Word is the primary word processor the project will use for all of its documents. In addition, it will be used to help us design the digital cards for the application.

Microsoft Visio – Microsoft Visio is the primary tool we use for diagrams that are included in our documents.

Microsoft PowerPoint – Microsoft PowerPoint is the primary presentation application that we use to develop our document presentations.

Microsoft Paint – Microsoft Paint is a simple graphics program that we will use to take our card designs that are made in Word and make them into images that we can manipulate.

Eclipse– Eclipse is an open-source development tool that will allow us to create our Java Applet and its supporting classes.

WinSCP – WinSCP is a secure file transfer application that we use to upload/download all our files to/from our Project Web Site and the Applet Website.

GIMP – GIMP is an open-source graphics program that we may rely upon for developing a more aesthically pleasing applet environment.

7.4 Configuration Management

Due to the small team size and modular design of the system, we feel that we do not need any specialized software for version control. Using specialized software would likely complicate the issue more than necessary. Instead, the Lead Programmer will handle the version control. Team members will be able to work on assigned modules and send them to the Lead Programmer when completed. The modules will then be implemented by the Lead Programmer generally during a scheduled team meeting.

The Lead Programmer will be responsible for clearly labeling each version with a version number and uploading the latest version to a secure online location accessible to all team members. All previous versions of the software will also be stored online in an organized manner.

8. Test Plan

Module Testing

As detailed by the lifecycle model our coding and testing phases will follow an evolutionary prototyping approach. Therefore code will be written onto the latest approved version of the prototype. This eliminates the need for imitation drivers or stubs as the modules completed will be tested using the latest accepted version of the prototype. An example of this would be the testing of the playback module. Since the grid and playarea modules are already part of the prototype by the time the playback module is scheduled to be completed, we can test the module by loading a log file that contains tile moves from the play area to the grid. Then the test will be to see if these moves are visually displayed.

Integration Testing

Once members have finished coding and testing the individual modules that they are working on, the code will be written into the main prototype. Even though members will be using their own copy of latest version of the prototype while writing their code, it is not unreasonable to expect that different modules will be coded by different team members concurrently. Therefore this testing is to assure that the code being written into the main prototype is compatible with another member's code and previously coded modules. An example of this would be testing the integration between the grid, tile arrangement, and playarea modules by moving a tile from the play area and placing it in the grid. The result would be immediately visible on the screen. The result can also be checked by looking at the log file created if the save arrangement button is clicked after the move.

System Testing

In our evolutionary prototyping development the system test will be a complete test over the systems expected functionality with the modules it contains at the time of testing. Since module and integration testing will be performed first, system testing is the last test to be performed before the prototype is advanced to the next acceptable version. This will be the version that the members will use as the driver to their new modules. The actual testing will consist of a repetition of all the module testing and integration testing that has taken place between version upgrades. System testing I will contain the tile arrangement, grid, and playback modules. System testing II will contain the previous modules plus the data module.

Acceptance Testing

After all of the functionality has been added to the system we will begin acceptance testing. During this testing the team will observe selected chemistry students using the system in the HCI lab. For this testing we are gong to enlist the help of ten students and six teachers that will be categorized into two separate groups and tested using two different methodologies.

Task Oriented

During these tests the user will be seated next to a team member in front of a computer running the system. The other team members will be in the other room of the HCI lab operating the cameras, VCR, and observing the test. The team member leading the test and next to user will be asking the user to perform a series of specific tasks on the system

Acceptance Testing I
User: Chemistry Student
Number of Users to be tested: 5

These tasks to be performed are:
- open the applet
- load the first tile set into the playarea

- reorganize the play area by taking the third tile and placing it in the first spot
- move all of the tiles to any locations on the grid
- arrange the tiles on the grid in the most logical arrangement according to the student
- move a specific tile from the grid to the playarea
- add a blank tile to the set of tiles
- save the final arrangement of the grid
- fill in the pop up form asking why they left the tile placed back into the playarea out of the final arrangement

These sessions will be focusing primarily on the use of the system through the perspective of the student. Therefore, tasks that relate to the teachers role such as the playback of arrangements and tile set modifications will not be included. The team will record any mistakes the user makes, as well as their response as to why they made the mistake, while performing the tasks. Examples of mistakes may include clicking on the wrong menus or moving the tiles to wrong locations. We will also be keeping track of the time it takes the student to complete the task. From these tests we hope to discover any problems with interface that prevents the students from accessing the functionality of the system.

Acceptance Testing II
User: Chemistry Professor
Number of Users to be tested: 3

The tasks to be performed are:
- the same tasks as students
- open the recording the user submitted earlier
- move ten steps forward
- move a step backward
- auto play the rest of the moves till the last recorded move
- modify a tile set by adding another tile
- modify the same tile set by removing a different tile from the tile set
- create a new tile set from images preloaded onto the computer
- create a new best arrangement pattern for the new tile set
- adjust the rules of the new tile set to not allow students to input blank tiles

- adjust the rules of the new tile set to not allow students to add new best arrangement patters
- turn off the gridlines

These sessions will primarily focus on the use of the system through the role of the teacher. However, they will need to complete the tasks the students have because their understanding of the system is more complex. The teachers will be evaluated and observed the same way as the students. During these tests we hope to locate any misunderstandings the teachers may have due to the interface organization.

Open Environment

These tests will be conducted similar to tests above. However the user will not be asked to perform or be guided through specific tasks. They will be given a brief description on the purpose of the system and be asked to simply use it. While they are guiding their way through the system the lead tester will be asking them questions about the language of the menu options, size of the forms, difficulty of scrolling, difficulty of using the minimap, colors of the forms, visibility of the information on the tiles, organization of the menus, and overall comfort with using the system. From these tests we hope to discover anything in the interface that is unappealing to the users.

Acceptance Testing III
User: Chemistry Student
Number of Users to be tested: 5

Acceptance Testing IV
User: Teacher
Number of Users to be tested: 5

9. Deliverables

These documents are online and are available for the client to view at any time. They serve as the project documentation and an evolutionary understanding of the product between the development team and the client.

Requirements and Analysis Document

This document describes the functionality of the system required by the client.

Design Document

This document defines the architecture of the system and decomposes the system into smaller subsystems and objects. It also describes the methods of design and testing that will be used and illustrates the user interfaces.

Client Prototype

This prototype will be located on a website that will later evolve into the permanent site for our client. It will only be accessible to our client and specific chemistry students at this time. The prototype will be used for demonstrational purposes and to test the user interface and system functionality. It will be available to our client after 12/12/2005.

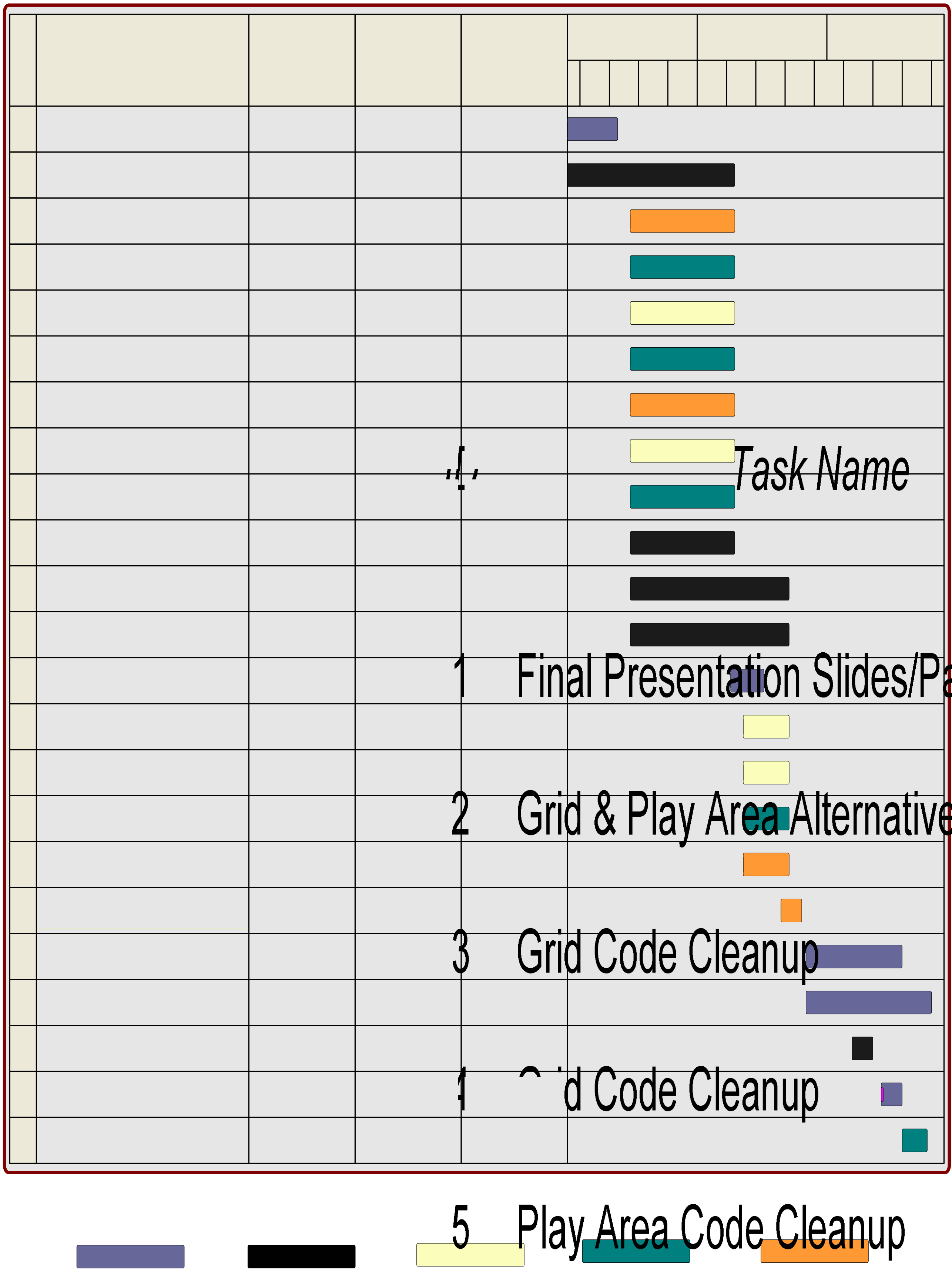
User-Manual

This manual will be presented with the software and will describe the functionality of the system in detail. It will explain how to utilize the system from the teacher and student perspectives. It will also include a developers section that will contain the results form the testing, developing, and coding stages of this project.

Clip System

This is the final product. The CLIP system will replace the current system and be online and accessible to anyone after 5/06/2005.

10. Efforts and Schedules



Task Name

1 Final Presentation Slides/Paper

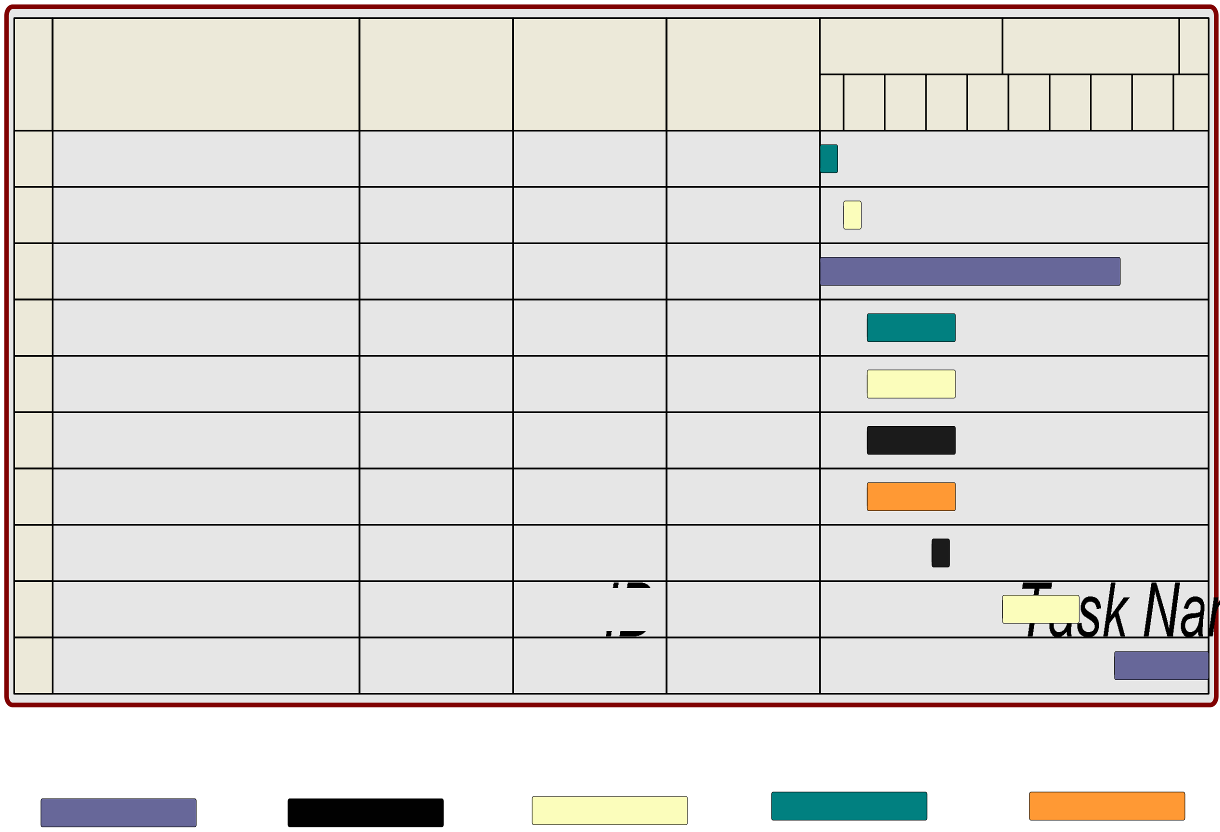
2 Grid & Play Area Alternatives

3 Grid Code Cleanup

4 Grid Code Cleanup

5 Play Area Code Cleanup

6 Play Area Code Cleanup



Effort Measurement

The duration times for each of the tasks mentioned below are only best-guess estimates. The best-guess estimates are based on normal workload expectations where the individual or individuals will do what is required within a reasonable time given the level of priority of a task. There are three levels of priority: high, normal, and low. Most tasks initially fall under normal priority or low priority, but their status changes as they become more important (or even less important, though rare).

High Priority – Needed as soon as possible for other areas of the project where without it, it would be difficult to move forward.
Normal Priority – Needed as soon as possible for other areas of the project but there is enough flexibility to allow for some delay.
Low Priority – Not necessary for other areas of the project right now but would be nice to have done as soon as possible.

Tasks

Below is a description of the tasks our team will undertake to complete the proposed system. The task names correspond to the previous diagrams. There are five attributes for a task: task name, purpose, resources (team member (s) assigned), priority, and dates active (start to finish dates). In addition, the resources named for a task are considered in charge of that tasks and are responsible for completing it. The persons in charge of a task are always free (in fact encouraged) to ask other team members for assistance.

Task Name: Final Presentation Slides/Paper
Purpose: Compilation of information from the RAD, SDD, and Project Plan and development of presentation slides based on the compilation.
Resources: All Team Members
Priority: High
Dates Active: 12-01-05 to 12-12-05

Task Name: Grid Code Cleanup
Purpose: Quality enhancements to the Grid implementation, which will result in a significant improvement in performance and usability.
Resources: Nathan & Neil
Priority: Normal
Dates Active: 12-16-05 to 01-09-06

Task Name: Play Area Code Cleanup
Purpose: Quality enhancements to the Play Area implementation, which will result in a significant improvement in performance and usability.
Resources: Rich & Neil
Priority: Normal
Dates Active: 12-16-05 to 01-09-06

Task Name: User Interaction Code Cleanup
Purpose: Quality enhancements to the User Interaction implementation, which will result in a significant improvement in performance and usability.
Resources: Nathan & Brian
Priority: Normal
Dates Active: 12-16-05 to 01-09-06

Task Name: User Tutorials

Purpose: Development of step-by-step webpage instructions on using the system. These will be what we use when conducting Acceptance Testing sessions.
Resources: Neil & Brian
Priority: Low
Dates Active: 12-16-05 to 01-09-06

Task Name: Rule Editor Implementation
Purpose: Implementing the read and write operations of the rule editor portion of the system.
Resources: Rich
Priority: Normal
Dates Active: 1-12-06 to 1-22-06

Task Name: Tile Editor Implementation
Purpose: Implementing the read and write operations of the tile editor portion of the system.
Resources: Rich
Priority: Normal
Dates Active: 1-12-06 to 1-22-06

Task Name: Additional Tile Sets Designed
Purpose: Creating images for new tile sets.
Resources: Brian
Priority: Normal
Dates Active: 12-16-05 to 1-21-06

Task Name: UI Development
Purpose: Improving the entire look and interaction with the user including the placement and organization of UI components based on team consensus, client progress reviews, and acceptance testing sessions.
Resources: Brian
Priority: Normal
Dates Active: 12-16-05 to 1-21-06

Task Name: Grid/Play Area Alternatives
Purpose: Researching an alternative implementation for the Grid and Play Area that uses a JTable for the Grid and a JList for the Play Area. These components are the involved in the typical approach to drag and drop functionality (therefore cleaner and more robust) but they will only be

pursued if they are found to be capable of doing what the current implementation can do.
Resources: Brian
Priority: Normal
Dates Active: 12-01-05 to 1-09-06

Task Name: Save/Load Implementation
Purpose: Implementing read and write operations that relate to loading tile sets and previous arrangements, and saving current arrangements.
Resources: Neil
Priority: Normal
Dates Active: 1-12-06 to 1-22-06

Task Name: Mini-map Implementation
Purpose: Implementing a miniaturized rendering of the Grid
Resources: Nathan
Priority: Normal
Dates Active: 1-12-06 to 1-22-06

Task Name: Client Progress Review I, II
Purpose: Official demo sessions with our client to report on our progress and get feedback based on our latest prototype. The person indicated in the resources section leads the review meeting.
Resources: Nathan (I), Neil (II)
Priority: Normal
Dates Active: 1-21-05 to 1-23-05 (I), 3-01-06 to 3-03-06 (II)

Task Name: Simple Pattern Implementation
Purpose: Implementing an optional feature of simple pattern comparisons.
Resources: All Team Members
Priority: Normal
Dates Active: 1-27-06 to 2-18-06

Task Name: Acceptance Testing I, II, III, IV
Purpose: Official demo sessions with potential users to get feedback based on the latest prototype. . The person indicated in the resources section leads the acceptance session.
Resources: Brian (I, IV), Neil (II), Rich (III)
Priority: Normal

Dates Active: 2-07-06 to 2-11-06 (I), 2-19-06 to 2-24-06 (II), 3-05-06 to 3-07-06 (III), 3-20-06 to 3-22-06 (IV)

Task Name: System Testing I, II
Purpose: Functional testing completed by the Team to check that all functionality at the time works as intended.
Resources: All Team Members
Priority: Normal
Dates Active: 1-09-06 to 1-12-06 (I), 2-14-06 to 2-28-06 (II)

Task Name: System Refinement
Purpose: Improvements made to the system during and after System Testing that relates to improving the performance and usability of the system.
Resources: All Team Members
Priority: Normal
Dates Active: 1-27-06 to 2-25-06

Task Name: Developer Manual
Purpose: A document based on our RAD, SDD, and internal code documentation that fully explains from a technical standpoint our system.
Resources: Rich & Neil
Priority: Normal
Dates Active: 3-09-06 to 3-23-06

Task Name: User Manual
Purpose: A document based on our RAD, SDD, internal code documentation, and user tutorials that fully explain in non-technical terminology how to use the system.
Resources: Brian & Nathan
Priority: Normal
Dates Active: 3-09-06 to 3-23-06

Task Name: Client Acceptance Review
Purpose: The system is presented to the client for final review. It's a sort of last chance review before we focus the rest of our efforts on documentation – both internal and external – and no more coding will be allowed unless absolutely critical. The person indicated in the resources section leads the review meeting.
Resources: Rich
Priority: Normal

Dates Active: 4-01-06 to 4-13-06

Task Name: Any Additional System Refinement
Purpose: Any final refinements to the system are done now.
Resources: All Team Members
Priority: Normal
Dates Active: 3-01-06 to 4-20-06

Task Name: System Delivery
Purpose: The system is officially complete and is delivered to the client via the web site that we establish. We present the client with a hard copy of both the user and developer manuals and list the soft copies of both on the web site.
Resources: All Team Members
Priority: Normal
Dates Active: 4-20-06 to 5-05-06

11. Measurements

Project progress will be tracked via two methods. The first method will be team log reports and the second method will be manager meetings.

Team Log Reports

Each member of the team will write up a weekly log report. In each member's log, they will report on the work they have done throughout the week. At the end of each week, the team members will send their log reports to the project manager. The project manager will look through the logs of the other members and then write up a weekly overview on the progress of the team. This report will then be sent to the Senior Project Instructor.

Manager Meetings

Every other week, the project manager will meet with the Senior Project Instructor and provide verbal progress reports and team reports. This is an opportunity for the Instructor to catch up on the current progress on the team as well as clear up any issues in which the instructor is unclear on.

12. Risk Management

Our biggest risk is the research we have to do before we even get into the developing our prototype. Since we chose to use Java, most of the team has to read up on it and get familiar with how it works and where it differs from what they know already.

Furthermore, we have encountered a number of user interface difficulties, which revolve around dealing with the limited screen space. A contributing factor to the limited space problems deals with trying to accommodate for cards that have varying amounts of text (namely, the element cards have all types of information that vary in size) and how we can make the system environment more relatable to the real world. This requires us to spend additional time in the design stage while we code, therefore, potentially holding up some of the coding we need to get done but cant. Also, in order to deal with the UI issues we have, we are looking into developing a mini-map for making the system friendlier when arranging the cards, but how to go about doing this is not known to the team at this time. This requires additional research time.

In summary, all our risks revolve around requiring research into areas we have not yet been exposed to. Taking the time to research steals time from other areas that may need more time down the road. However, we have decided to combat this by assigning a team member to be a Risk and Development Officer to coordinate a modularized development schedule that allows us to work on multiple areas at once. In addition, this team member will continually monitor our progress with individual task checklists handed out to the other members and he will lead the integration process during our weekly code reviews.

13. Ethics

The ethical considerations that our group had to consider on this project included keeping the results secure. We did not want everyone to be able to see the results of the other students in the class. This is because a student could get the results from another student in order to copy their solution. The way we are planning to deal with this situation is to have the students e-mail the solution to the professor. This will allow the e-mail service to make sure that the transfer of the file is secure.

Another possible ethical consideration is who we would allow to use the system. The client Susan Wiediger of the Chemistry Department wants other professors to be able to use the system. In order to deal with this we are planning on not requiring any sort of log on or user/passwords in order to access the program. This means that we will allow anyone to use the system, regardless of if they are a member of our client's chemistry class.

Another ethical consideration involving the available users is what platform we will allow this program to run on. If we developed it only for a Windows machine some of our client's co-workers may not be able to use the program. Because of which, she wants the system to be usable on different platforms besides Windows. In order to deal with this we are planning on developing the system with java.

The online nature of the planned java applet may require us to access the user's hard drive. We may need to be able to read and write specific files on the user's computer. These read/write privileges require us to think ethically. We should make sure that the user is aware of what we are doing, and that the program is not trying to do anything that may be detrimental to the user, or the user's computer (such as viruses or spy ware).

14. Glossary

Attribute: An attribute refers to an attribute on a tile or a card. Each set has its own set of attributes and each tile or card within a set has a set of values for those attributes that make that card or tile unique.

Blank Tile: A blank tile is a tile added to a set in Tile Arrangement Mode. A blank tile is not a permanent addition to the set. The user can edit a blank tiles attributes. The purpose of a blank tile is to fill in holes in a user's pattern (similar to undiscovered elements on the periodic table).

Card: A card is the physical version of a tile. A card belongs to a set. Every card in a set has the same list of attributes. The values for the attributes vary from card to card. Multiple cards can have the same values for some attributes but the whole set of values for each card should uniquely define that card for that set.

Comparison Pattern: This is one rule of a set that determines whether or not a pattern will be provided to allow the comparison of a user's tile arrangement against what is considered a 'best' or logical tile arrangement.

Creator Mode: The system provides the ability for sets to be created and modified given the user-chosen rules, tile images, and comparison pattern(s).

Play Area: This is the area to the right of the Grid. It contains a list of tiles, a mini-map, and a series of playback controls. The primary component of the play area is the list of the tiles where the tiles are shown when the system starts up. Tiles in this list can be moved around by the drag and drop action. Using the drag and drop motion, a user is allowed to move tiles to another position in the list of cards or to the Grid. The mini-map component of the Play Area is for showing a miniaturized version of the Grid. Finally, the playback controls allow for the playing of a recorded tile arrangement.

Pattern: A pattern refers to any kind of arrangement of tiles on the grid.

Playback Mode: The program enters this mode when the instructor is viewing a student's results. In this mode, the instructor can move through the steps the student made while forming a pattern. It supports fast forward and reverse functions as well. Its purpose is that of a possible grading technique as well as a research tool to observe the thought processes of students while performing this activity.

Pre-Loaded Set: A pre-loaded set is a set of tiles that has been pre-loaded on the server. This allows users to load up the applet and start using it with the provided data.

Rules: These are attributes that are a part of the set to distinguish them from other sets. Such rules include a set name, description, type (2 types only), and whether or not comparison pattern(s) are present.

Set: A set is a collection of tiles or cards. Each set has a list of attributes that are common to all tiles or cards within it. Furthermore, the system supports the option to create or modify sets.

Tile: A tile is the electronic version of a card. A tile belongs to a set. Every tile in a set has the same list of attributes. The values for the attributes vary from tile to tile. Multiple tiles can have the same values for some attributes but the whole set of values for each tile should uniquely define that tile for that set.

Tile Arrangement Mode: Tile Arrangement Mode is a mode the program enters when a users wants to select a set and try to form a pattern with the tiles from that set. The main function within Tile Arrangement Mode is dragging tiles from the play area and grid to locations on the grid in order to form a pattern.

Version #	Date	Author	Description
0.1	12/1/05	Team	Decided on layout and content of the document.
0.2	12/4/05	Brian Navarro	Revised System Decomposition. Re-wrote User Interface section and Efforts and Schedules section.
	12/4/05	Neil Alfredson	Re-wrote Design Trade-offs section. Re-wrote Packages and File Organization section.
	12/4/05	Rich Carney	Revised Scenarios and Use Cases. Re-wrote Testing section.
	12/04/05	Nathan Mikeska	Revised Subsystem Interfaces and Class Diagram. Insertion of sections into document. Formatting and spelling/grammar fixes.
1.0	12/11/05	Team	Reviewed and accepted as version 1.0.