Name: Harris Christopher Hollis
Michael Lodge
Chris Blake
Phillip Gipson
Joe Petrizzi

Georgia Southern University
Department of Computer Sciences

CSCI5530 Software Engineering
Spring 2008

Notebook of Documentation for Virtual Bead Loom

Date: 1-May-08

People who provided assistance/affiliation: **Dr. Kera Bell-Watkins, Ron Eglash**, **Group Members**

References:

Virtual Bead Loom software, SmartDraw,
http://java.sun.com/docs/books/tutorial/uiswing/components/icon.html,

Team Structure

Member	Title	Responsibilities
Chris Hollis	Chief Programmer / Team Lead	Designs, implements critical parts

	/ Software Engineer	and makes major decisions / Designer/ Analyst/ Tester/ Trainer
Chris Blake	Understudy/ Administration/ Tester/ Programmer	Substitutes for chief programmer/ Record executive meetings / Designer/ Analyst/ Tester/ Trainer/
Joe Petrizzi	Librarian / Administration/ Tester	Maintains all project documentation/ Recorder/ Analyst/ Configuration management/ Tester
Michael Lodge	Programmer	Requirements Analyst/ Programmer/ Tester
Phillip Gipson	Programmer	Requirements Analyst/ Programmer/ Tester

Table of Contents

Project Legacy

1. Introduction

The Virtual Bead Loom is software designed to help accustom students of grades 6 through 12 to the concepts of computer programming. This is the main goal of the collection of programs of which the Virtual Bead Loom is a part, called Culturally-Situated Design Tools (CSDTs). The

program accomplishes its goal by allowing the students to interact with a digital bead loom, representing the real looms used by Native Americans and other cultures to create designs. The current program encourages the students to replicate on the bead grid any image (usually one given to them by a teacher), and as they invoke methods to create and manipulate the beads, pseudo-code is displayed in a separate window that, in later versions of this software, can be used in place of graphical UI tools to perform the same tasks.

2. Initial Expectations

The initial goal of the project was to assess previous versions of the Virtual Bead Loom done in Flash code and replicate the same functionality in a Java Applet. The clients for this project also suggested additional requirements, including the ability to save and load bead looms as XML files, and the ability to e-mail files across a LAN (typically a classroom setting). It was expected to be able to finish the project within the constraints of the Spring semester, with the deadline for presentation being the fourth week of April, 2008. Each group member expected to spend approximately fifty to one-hundred hours of work each on the project from beginning to completion. There were no monetary costs to developing the software. In addition, the level of quality in the finished product was expected to be as close to professional-grade as possible within the time constraints. It was understood that another development team would be assigned the maintenance and continued development of this project after our goals were completed.

The team's first assessment of the project was to acquire all requirements over the period of two weeks and develop a plan for designing software. The team began by designing each individual feature to be included in the finished product. Following this, a graphical UI was created based on these designs. This process spanned the length of two weeks.

After all requirements were solidified and there was a shell of a program to be worked on, design was turned into implementation and the coding phase began. Coding lasted approximately six to seven weeks, with some testing done along the way. The formal testing phase began immediately after coding was complete, occupying approximately two weeks.

3. Current Status of the Project

The project's major goals of replicating the previous Flash version of software and implementing XML file compatibility were completed, however because of differences between Flash and Java, some parts of the UI had to be redesigned. The e-mail requirement was deemed infeasible due to

the length of time spent coding the completed goals. Furthermore, there were added sections of the UI and code to assist in the transition from this team to another in the future, such as the Code Output window.

There have been several minor bugs found in testing, a complete list of which can be found in the Appendices. These bugs do not have an impact on critical systems on the whole, however some of them prevent some users from achieving full functionality of the program. This has been noted for future development.

The schedule laid out at the beginning of the project was able to be realized, keeping to each set deadline from the requirements phase up until coding. Coding and testing both ran over schedule as problems were found and needed correction. This unforeseen circumstance required more time from each group member than previously expected. The full amount of work can be seen in the Appendices under Group Logs, Individual Logs, and Schedule.

4. Remaining Areas of Concern

The remaining goal that was not accomplished is the ability to e-mail bead loom files over a LAN. This goal has been left for future development teams. In addition, graphical display issues involving the Mac OS were discovered; there will be an attempt by this team to fix these problems, but if it is infeasible it will be left to future teams. This risk has been analyzed by the team and should not present a major problem, as a workaround was found for each major problem that may

occur as a result of this graphical issue. These workarounds are listed in the Appendices.

Additional functionality to be added by future teams definitely will include development of a standardized pseudo-code for display as actions are performed within the bead loom, and development of a parser for this pseudo-code so users can produce their own code to create and manipulate beads. Another functionality in the movement of beads may be changed in the future by the clients to remove the ability to shift beads with the buttons and instead implement a method of relocating objects by directly inputting coordinates.

Remaining on the schedule for the team is a preliminary submission of all documentation and presentation, revision of documentation and final testing of the software, and the final presentation of the software on May 6th, 2008.

5. Activities/Time Logs

The logs and diagrams of activity on the project are referenced in the Appendices under Group Log, Individual Logs, and Schedule.

6. Technical Lessons Learned

The team became further familiar with the creation and manipulation of UI objects within Java and their intricate interactions when installed in an embedded applet. In addition, the use of XML within this project introduced the process of reading and writing XML file streams and the security features necessary to read information from a local disk to an HTML-embedded applet.

Also, the team became familiar with the documents necessary in software engineering, specifically (as technical lessons) the requirements, design, and testing documents and their usage in creating a finished product. Verification and Validation documents were additionally introduced to the team and implemented during the course of the project.

7. Managerial Lessons Learned

Team meetings were planned to be once per week throughout the course of the project. However, during the middle of the testing phase, it became clear that coding was more efficient on the project if all team members were present. Because of this, during the coding and testing phases, team meetings increased to three times per week. During the final week of development, team meetings

were called every day until all necessary goals were achieved. From this, there was a greater amount of effort put forth by each team member than previously expected.

The source code itself was expected to be approximately 2000 lines long. Due to directly implementing Flash code and having to work around issues, it is closer to 3500 lines long. A large chunk of this is the creation and updating of the graphical UI.

Estimated time to be spent of this project was approximately fifty to one-hundred hours per team member. This estimate seems correct, however it was not evenly distributed throughout the duration of the project. The bulk of the hours logged by each member came in the coding and testing phases, as the system was being brought together and problems occurred more frequently.

CSDT Project Proposal (Abstract)

Introduction

Culturally-situated design tools (CSDTs) are small-scale programs designed with progamming students in mind. They have a unique quality about them, tying an introduction to programming languages and design with specific cultural backgrounds and activities in an attempt to make the learning curve for the transition into programming easier for youth.

Example programs already in use are a "Graffiti Grapher," which creates graphic designs, "Break Dancer," which can mimic dance movements, and "Virtual Bead Loom," which can create various patterns and shapes. All of these use simple functions and variables to illustrate in real-time how code can impact what occurs in the program in ways that are familiar to the user. The target user group for CSDTs is middle and high

school students, ranging from grades 7 to 12.

Currently, there exist programs of this nature, but they are in need of upgrades and additional flexibility. One of the upgrades will be to convert from the programs' current language of Flash into a more universal, friendly, and flexible environment, Java. In essence, this will be a ground-up "redesign" of what exists now.

There will need to be a friendly interface presented in the programs, consisting of an output screen where the user can see results directly from the code, a code screen where all of the code will be displayed, and a development screen where the user can directly add, remove, or otherwise modify the code using drop-down menus. That is to say that from merely looking at the user interface that although interactive, the CSDTs do not have one-hundred percent freedom of coding. There will be static options as far as what functions can be implemented, what objects can be created, but there will be great maneuverability even within these constraints.

All of that said, this is not a direct jump into a programming language. For this reason, a pseudocode more closely resembling English must be created as part of this project. This pseudocode will be what the user is able to tinker with in the development windows, through the use of drop-down menus.

Proposal

Development will begin on the conversion of CSDTs from Flash to Java with the adaptation of the Virtual Bead Loom. Several other CSDTs function using similar algorithms and control schema, so this will act not only as the first conversion, but the backbone for further conversions into Java.

Analysis has been performed on the Virtual Bead Loom and it has been found to have six key components that must be developed and integrated. These components are shown below in the Architectural Context Diagram.

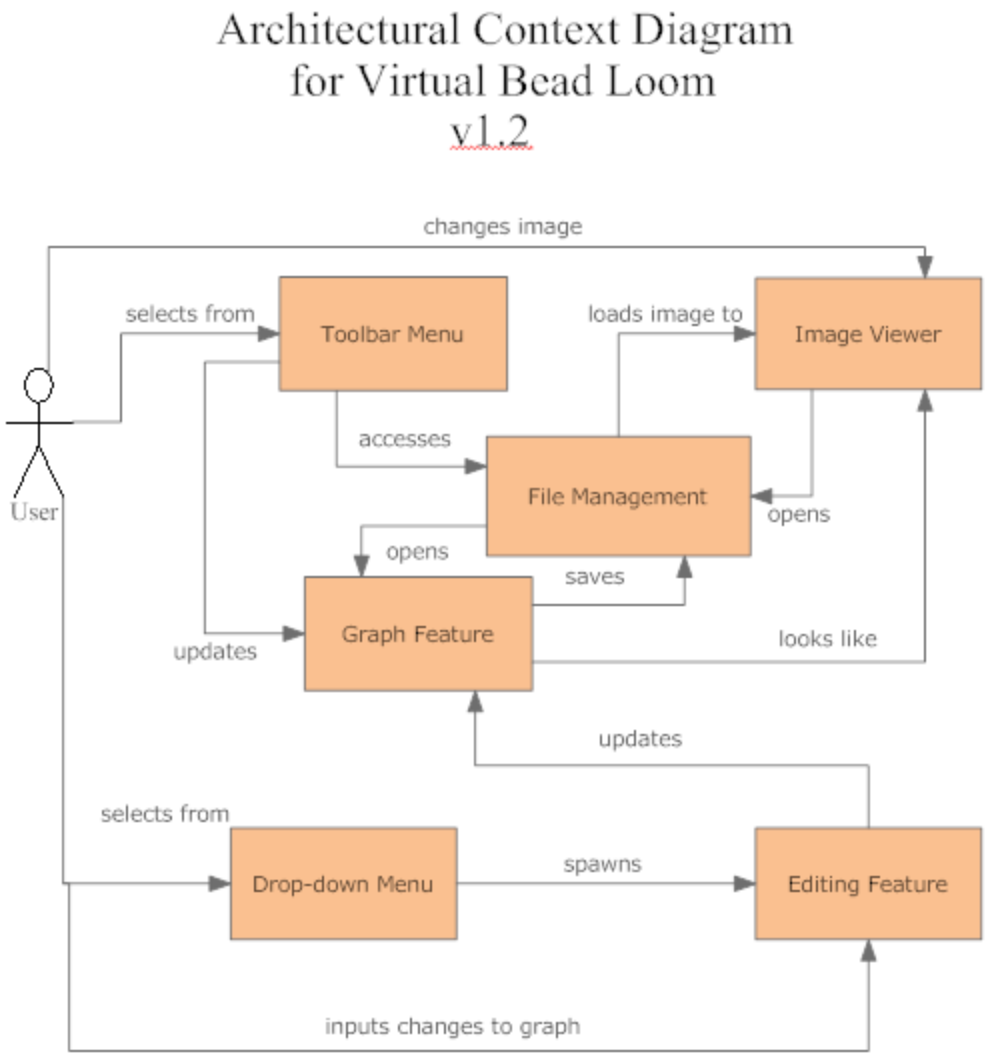


Figure 1. Architectural Context Diagram for Virtual Bead Loom CSDT

In the previous diagram, the components of the software are: the graph screen, the drop-down menus, the editing feature, the toolbar menu, the image viewer, and file management. The user cannot directly interact with the graph screen, but instead must do it through use of the menus and editing feature. Access to the editing feature is granted once a choice is selected from the drop-down menu. Without a selection, the editing feature does not know which specific options to display for the user to edit, and will show nothing (this may later be set to a default single "bead"). The user can then input the insertion points in x-y coordinate form for the "beads" to be placed upon the graph. The toolbar menu can be used to open and save files, clear the graph of all "beads", and change the graph's layout. The image viewer does not directly act with the graph, but the

image in the viewer is the target goal to be reproduced by the user. The file management interacts with the graph screen when files are being opened or saved.

Operation of the system will proceed in the manner given above, as a general rule of thumb. There is further clarification upcoming in the use case scenarios that will be produced, as the systems become more concrete.

Risk and time estimate reports for the graph viewer program segment are on the following pages. They were formulated following COCOMO II using Costar 7.0, demo version. For the purpose of these reports, the estimated lines of code is 700 lines for the graph feature. This figure may significantly increase or decrease as requirements are met. As it stands, this feature should be easily completed by the assigned deadline of May 6, 2008.

Virtual Bead Loom Graph - Archive Report

Costar 7.0 Demo 02/02/2008 23:03:40 Page: 1

Estimate Name:	Virtual Bead Loom Graph	Estimate ID:	
Model Name:	COCOMO II 2000	Model ID:	2000
Process Model:	COCOMO II Model	Phases:	Waterfall

File:
Description:
Model File: Built in.

Hours per Person-Month: 152

COCOMO Estimating Equations		
Effort	= 2.9400 * EAF * (KSLOC)$^{1.0536}$	= Effort in Person-Months
Schedule	= 3.6700 * (Effort)$^{0.3087}$	= Duration in Months
Maintenance Effort	= 2.9400 * EAF * (KSLOC)$^{1.0536}$	= Effort (per year) in Person-Months

EAF = 0.6099

COCOMO II Scale Factors	
Precedentedness:	Somewhat Unprecedented
Development Flexibility:	General Conformity
Architecture / Risk Resolution:	Generally (75%)
Team Cohesion:	Highly Cooperative
Process Maturity:	SEI CMM Level 2

Results	PD+DD+CT+IT
EAF	0.6099
Nominal PM	2.0
Actual PM	1.2
Cost (K$)	0.0
Developed Size	700
Productivity (Lines / PM)	568.4
Unit Cost ($ / Line)	0.00

Software System Specification

I. Introduction:

The purpose of this project is to implement the needed requirements for enhancing the CSDT (Culturally-Situated Design Tools) for middle- and high-school students to engage them into experimenting with the simplicity of an English-like pseudo code, which will ultimately make it easier for transition to a higher programming language. To further expectations of reaching a new and younger generation, actions must be taken to insure that the software can reach as many people as possible. The system specifications must allow a large majority of layman to understand and use the system. To ease further development, module usage and system interconnectivity must be thoroughly documented.

II. Scope and purpose of document:

The purpose of this document is to give the customer a record of the software's system requirements, performances, and constraints by providing detailed descriptions and diagrams

III. Overview:

1. Objectives
    1. Document all system requirements
        1. Focus of work will be adaptation of the Virtual Bead Loom (VBL) project from Flash into Java
        2. If VBL is completed and time permits, continue with breakdancing CSDT
    2. Illustrate all available aspects
    3. Develop or collaborate in annotation of pseudocode language for use in teaching students through use of CSDTs.
2. Constraints
    1. Fully describe all applied restrictions in detail
    2. Software should be able to work on all computers with the Java Virtual Machine installed

IV. Functional and data description:

1. <u>System architecture:</u>
    1. ACD description
        1. The software, development name "Rosa" is a self-contained system. Rosa will not interact with browsers, authenticators, databases, or mailing systems.

V. Subsystem description:

1. <u>System module narrative:</u>

1. JVM subsystem - no optimizations can be made directly to target machine's installed Java Virtual Machine
    1. Performance issues
        1. Handling the graphics in a certain way may hinder the software's efficiency
    2. Design constraints
        1. System must run in Java. Thusly, no runtime optimizations through assembly code are allowed

2. System modeling and simulation results:

1. System model used for simulation
    1. Minimum system specs allowed in public schools will be adequate
2. Simulation results
    1. None as of now
3. Special performance issues
    1. None as of now

VI. Projected Issues:

1. Projected development costs:
    1. This project will require many man-hours of concerted effort from our team members. From a monetary standpoint, little capital is lost to development. The only true cost that is of any significance is weighing the opportunity cost of working on one aspect of the system over another.
    2. Man-hours may be increased on the project if additional development work needs to be done on the pseudo-code language or the goal of adding programmable CSDT (pCSDT) functionality must be completed.
    3. Man-hours may be decreased if Flash source code is provided with ability to step-through. A preliminary version of the user-interface has been provided, streamlining this section of development.

2. Projected schedule:
    1. Development will complete by May 6, 2008
    2. A preliminary presentation of the near complete product will be presented within two-three weeks of May 6 deadline.

Software Feasibility Study

I. Introduction:

1. Statement of the problem:

    1. This project must be completed by May 6, 2008 and may be too much to complete.
    2. All efforts will be focused on Virtual Bead Loom so that future developers/maintainers

have a solid stepping platform to begin on future projects (breakdancer, programmable CSDTs, etc).

3. Client seems to expect more out of the project than Dr. Bell does, such as a programmable interface, etc. Bell said this would be saved for another class during another semester, but if we wanted to go ahead and do as much of that portion as possible, we would be much more likely to be compensated for our work at the end of the semester.

2. Implementation environment:

1. Windows XP Machines and other systems capable of running standard JVM.
2. Classroom setting computer systems with relatively low system specs.

3. Constraints:

1. Software must be usable by children to young adults.

1. Streamlined, straightforward UI must be implemented.
2. Must effectively portray programming in a positive light.

2. Windows focused compatibility
3. Doing our graphics programming with Java may be a bit more difficult.

II. Management summary and recommendations:

1. Important findings:

1. The client would like a few extra things beyond our class requirements.

2. Comments:

1. Adding a protected option to make Java visible would be a plus.

3. Recommendations (includes solution/decision justification):

1. Try to find time to add a programmable interface.

4. Impact (on cost, schedule, quality, functionality, etc.):

1. Quality of software may decrease based on the amount of additional requirements necessary. This would also directly lead to more man-hours being added to the project, primarily during the coding phase.

5. Alternatives

1. Alternative system configurations:

1. Drop goal of project to create pCSDTs (unless we have time near the end).
2. Possibly allow user to step through psuedo code.

2. Criteria used in selecting final approach:

1. Feasibility of completing all code and documentation by the given date.
2. GUI should be based on prototype on the subversion website.
3. Pseudo code should be based on SCRATCH.

III. System description:

1. Abbreviated statement of scope:

    1. This program should be able to teach middle school students a simple, but powerful pseudo code language. The pseudo code we choose follows a fine line: It should be as easy to understand as possible, but as powerful as possible. The interface will implement a visual representation of what the pseudo code is doing, with a step-through option so students can see what is happening, step-by-step, during run time. In our earlier prototypes, the program's addition/removal of pseudo code will be limited to drop-down menus, etc. If time remains, we may add a programmable interface, as desired by the client, but not required by the instructor.

2. Feasibility of allocated elements:

    1. The requirements set by the instructor can easily be accomplished. Any additional requirements added in this document were those that we feel are desired by the client, beyond our obligation to the project. Although not required, we will strive to accomplish these goals along with the ones assigned to us by Dr. Bell.

IV. Cost - benefit analysis:

1. The monetary costs of this project to the developers is negligible ($2,500 - $5,000 [if both groups are compensated])
2. Opportunity costs will be measured to determine what aspects of the system are most crucial
3. The benefits of this project are substantial to the developers in financial and educational aspects

V. Evaluation of technical risk:

1. Georgia Southern receives $2 million per year from the NSF. They are expected to produce results with this funding. As software engineering students, we have an obligation to make this project a quality product, or we could make the computer science department reflect poorly.

VI. Legal ramifications:

1. Flash code is being legally provided to developers. No license purchases required. Development of system is within lawful bounds.

VII. Style/Format:

1. The interface and pseudo code should be simple enough for a young student to understand, but complex enough to hold their attention long enough to master it and be able to move on to a higher programming language.

VIII. Other project specific topics:

1. The client has stated that they do not want the Java code to be made visible. They feel that seeing a complex programming language will intimidate younger students and deter them from learning the pseudo code. Perhaps we could add an option, changeable only by the teacher, which would make Java code hidden/visible. With this extra option, the same program could be used the following year with the same students that have learned the pseudo code to teach them Java as well.

Software Project Plan

I. **Introduction:**

The purpose of this project is to implement the needed requirements for enhancing the CSDT (Culturally-situated Design Tools) for middle and high-school students to engage them into experimenting with the simplicity of an English-like pseudo code, which will ultimately make it easier for transition to a higher programming language.

Middle and high school students use the regular CSDTs, and then gradually move towards the "modified" (mCSDTs) then to "programmable" (pCSDTs) which will all be a collection for the client. These modified and programmable tools will be designed under the software engineering team that will allow the client to increase the degrees of programming operations. These tools will be moved to a more

"user-friendly environment" which will be a Java Platform.

II. Scope and Purpose:

The purpose of this document is to provide the customer an estimate of how much the project will cost and how long the project will take.

III. Project Objectives:

1. Objectives:
    1. Gather, analyze, and assess requirements
    2. Keep detailed documentation on progress and goals
    3. Translate Flash programming into Java (mCSDTs)
    4. Increase functionality of mCSDTs to pCSDTs

2. Major Functions:
    1. Source code window displays code created by interface manipulation.
    2. Development windows permit users to manipulate the display window to create objects.
    3. Display output window shows all manipulations done by the user.

3. Performance Issues:
    1. Target system for use is not definitively known. Program is being developed in Java with platform independence.
    2. Assumed use of legacy hardware components

4. Management and technical constraints:
    1. Hardware of target systems and software (operating system, Java version) are constraints for the system. Specifically, although the software should be OS-independent, it is recommended for Windows XP. Additionally, the software require Java version (newest).

IV. Project Estimates:

1. Historical data used for estimates
    1. Previously utilized source code from earlier versions of the software have been used to assist in estimations for lines of code necessary.

2. Estimation techniques
    1. COCOMO II Model using Costar 7.0 Demo
    2. Instructor specified

3. Estimates
    1. For our team, project estimation complete date is April 22, 2008 for most features, with final modifications made no later than May 4, 2008.

V. Project risks:

1. Risk Analysis:

    1. Identification:

        1. Time constraints are the largest risk factor involved in development. Due to insufficient time, some features may not be able to be completed by the deadline.
        2. Compounded complexity of code may occur due to redevelopment of the system from Flash to Java. This may lead to certain features being infeasible or not able

to be finished by the deadline.

2. Time constraints:

    1. Preliminary system must be completed by April 20th and ready for presentation on April 22nd.
    2. Final system and documentation notebook must be presented on May 6th.

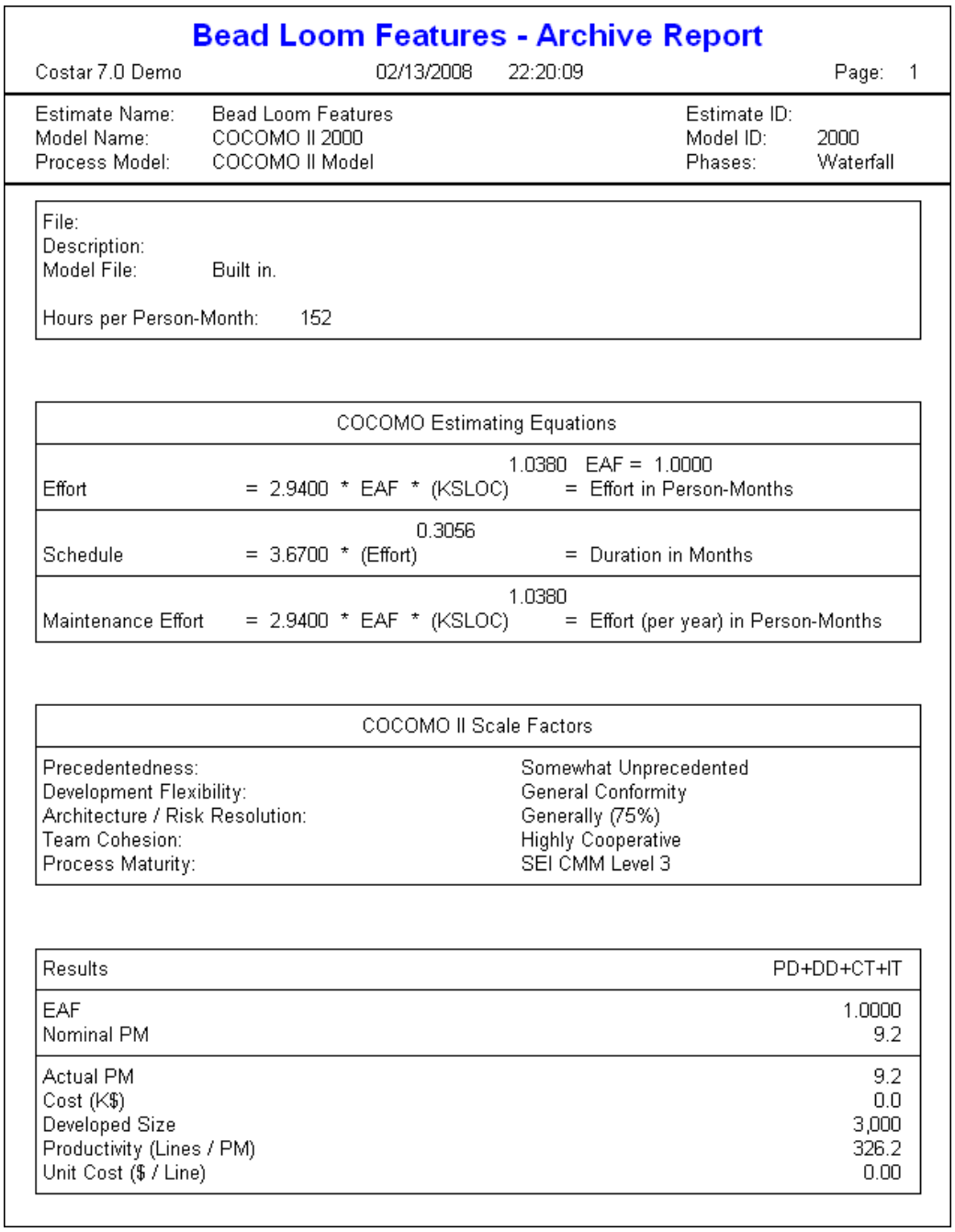
3. Risk Estimation:

    1. Based on our analysis of features for the program, we can estimate the lines of code for each feature. Some will be tackled first and foremost and others will be handled at a later time.

    2. Graph Feature – 700 lines
       Image Feature – 500 lines
       Co-ordinate System – 700 lines
       Drop-Down Menu – 600 lines
       Top-Menu – 500 lines

       Estimates are from assessing previous Java source code and previous experience in similar programming assignments.

    3. Total estimate for all features – **3000** lines of code



Bead Loom Features - Archive Report

Costar 7.0 Demo 02/13/2008 22:20:09 Page: 1

Estimate Name:	Bead Loom Features	Estimate ID:	
Model Name:	COCOMO II 2000	Model ID:	2000
Process Model:	COCOMO II Model	Phases:	Waterfall

File:
Description:
Model File: Built in.

Hours per Person-Month: 152

COCOMO Estimating Equations

Effort	= 2.9400 * EAF * (KSLOC)$^{1.0380}$ EAF = 1.0000	= Effort in Person-Months
Schedule	= 3.6700 * (Effort)$^{0.3056}$	= Duration in Months
Maintenance Effort	= 2.9400 * EAF * (KSLOC)$^{1.0380}$	= Effort (per year) in Person-Months

COCOMO II Scale Factors

Precedentedness:	Somewhat Unprecedented
Development Flexibility:	General Conformity
Architecture / Risk Resolution:	Generally (75%)
Team Cohesion:	Highly Cooperative
Process Maturity:	SEI CMM Level 3

Results	PD+DD+CT+IT
EAF	1.0000
Nominal PM	9.2
Actual PM	9.2
Cost (K$)	0.0
Developed Size	3,000
Productivity (Lines / PM)	326.2
Unit Cost ($ / Line)	0.00

Figure 1. COCOMO II Archive Report. Estimation tool report of risk, feasibility, size, and time for the Virtual Bead Loom computed on team ability, cooperation and difficulty of project.

4. Evaluation:

1. Currently, we will handle the Graph feature first because this is the core of our program and this uses all the entities for the features. The Drop-Down menu will be next. The other features will be handled in the next version. This is just an estimation of the features and NOT the overall project. Other features outside of the ones describe here may arise during the coding phase.

2. We will definitely need backups for everything.

2. Risk Management:

    1. Risk aversion options:
        1. Risk analysis will be performed throughout the semester.

    2. Risk monitoring procedures:
        1. Procedures will be monitored by mandatory documentation.
        2. Document comparisons will be performed through each phase to ensure that all design and implementation is according to proper requirements and specifications.

VI. Schedule:

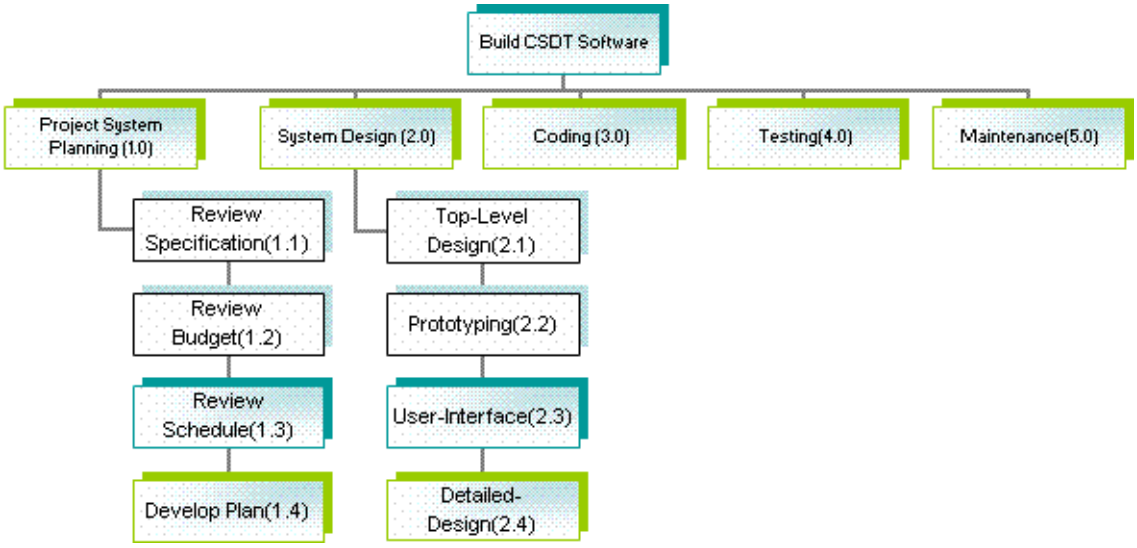


Figure 2. Scheduling Plan. Shows the phases planned to be performed in order.

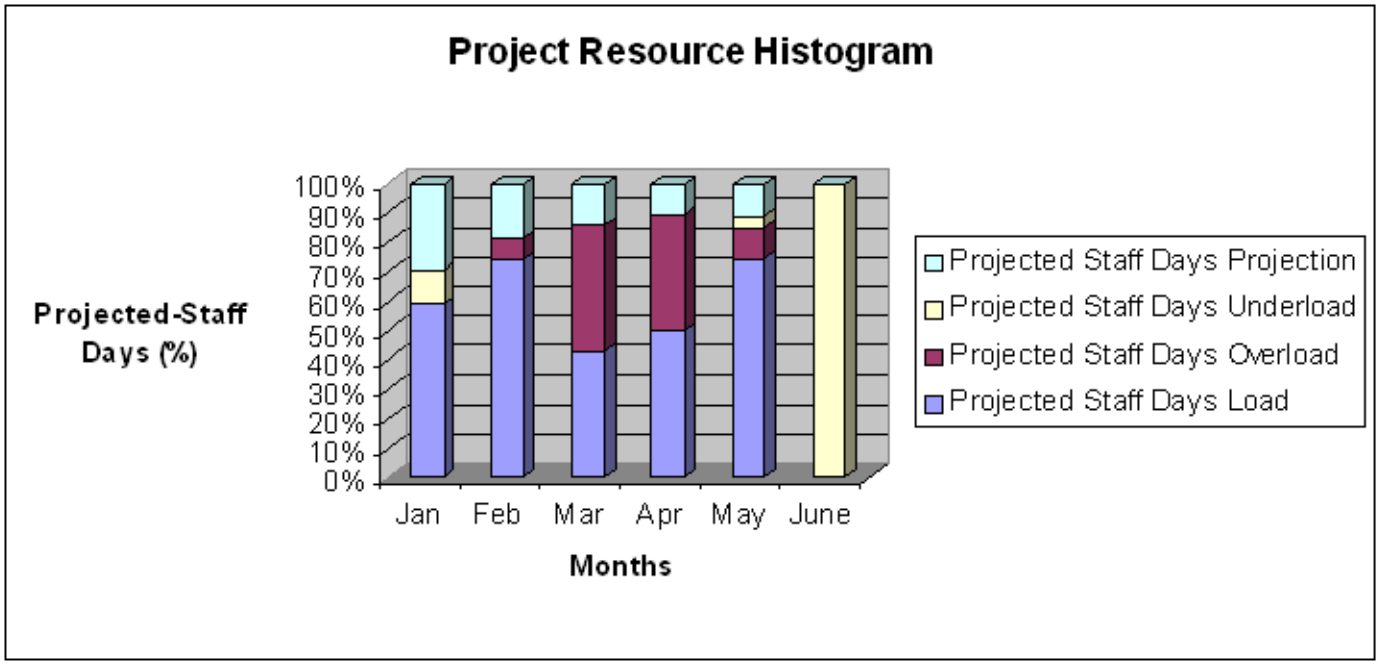


Figure 3. Project Resource Histogram. Shows per-month how much work will be done by the group members sorted by load.

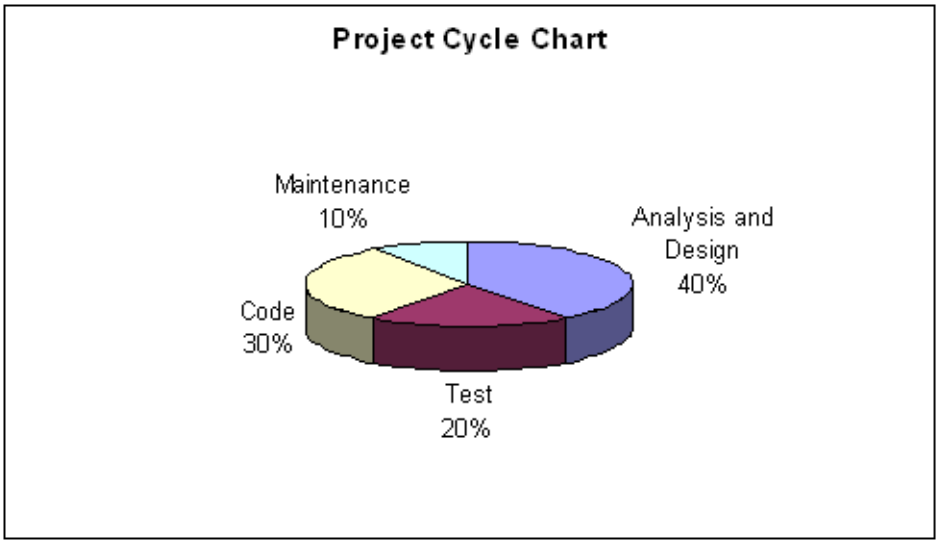


Figure 4. Project Cycle Chart. Shows a percentage of estimated total project time that will be spent on each phase.

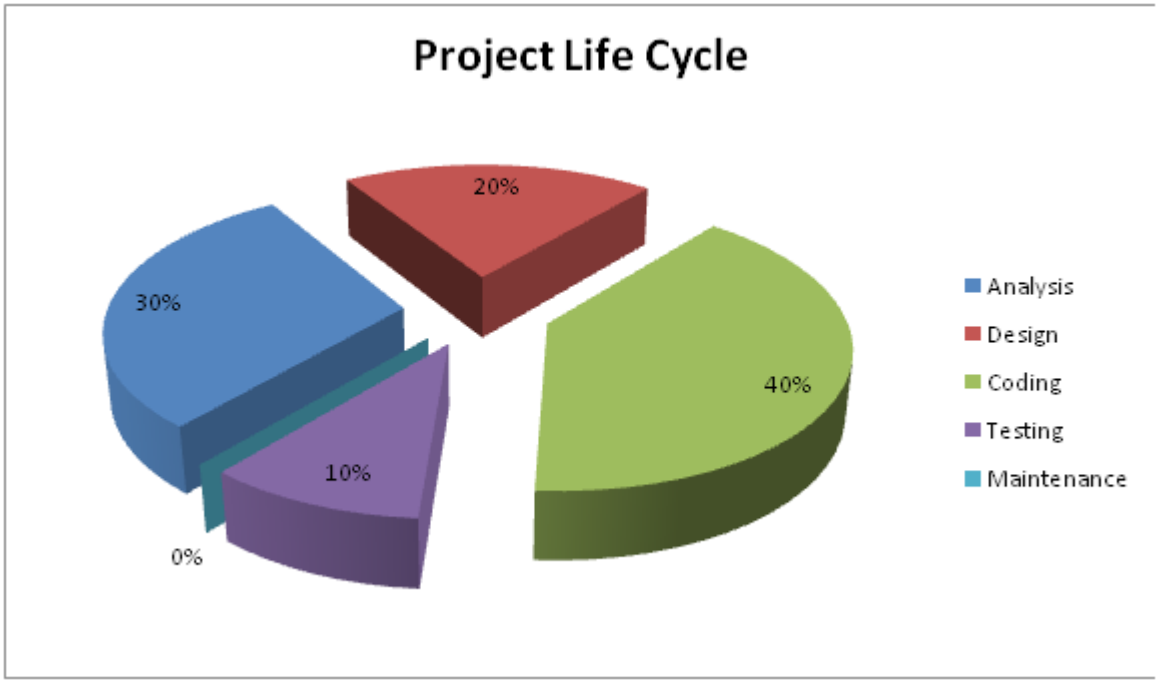


Figure 5. Updated Project Cycle Chart. A finalized version of the chart shown previously in Figure 4. Updated on 29-Apr-08.

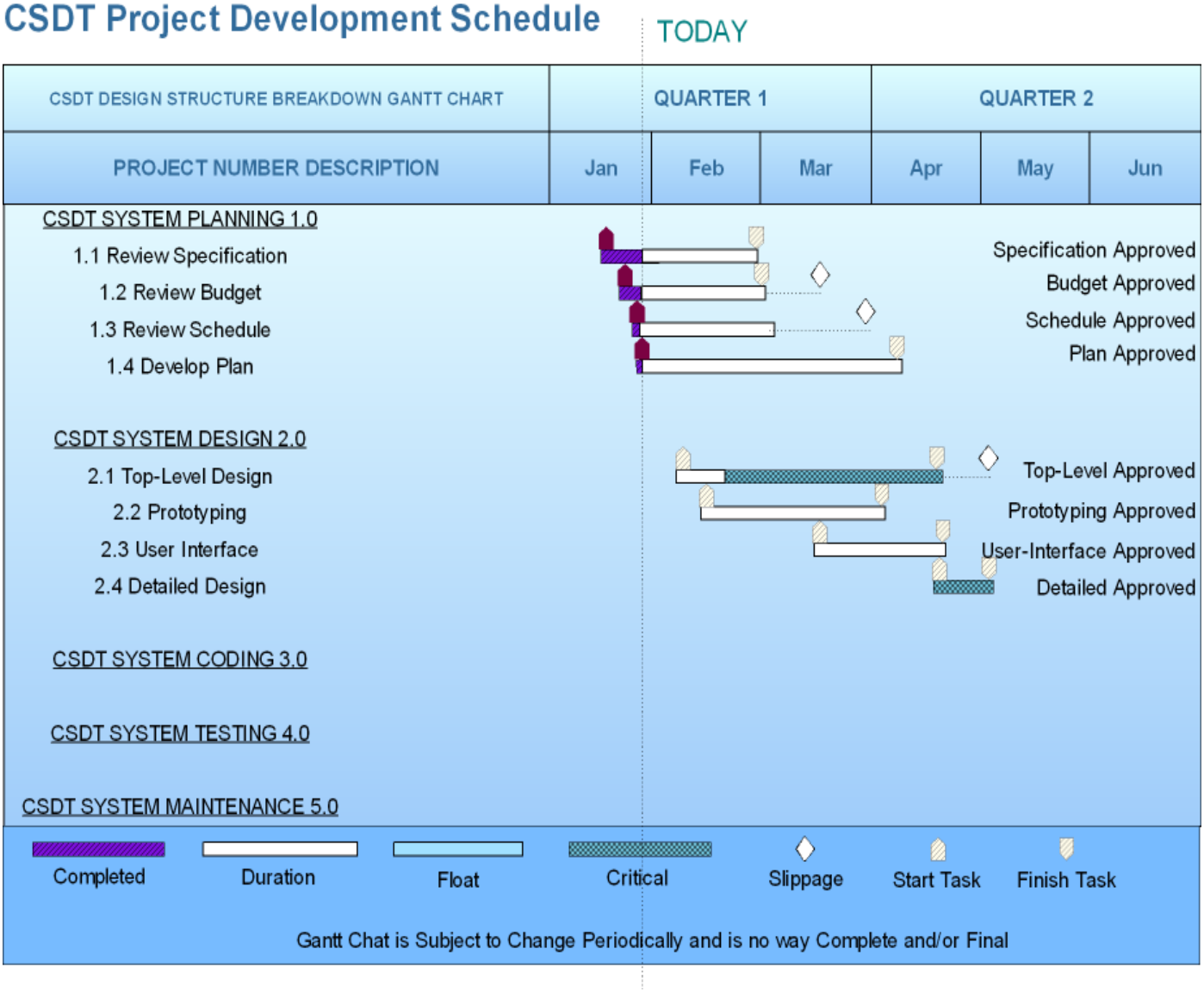


Figure 6. Gantt Chart I. Shows initial projections for timetables within each phase.

Produced 29-Jan-08.

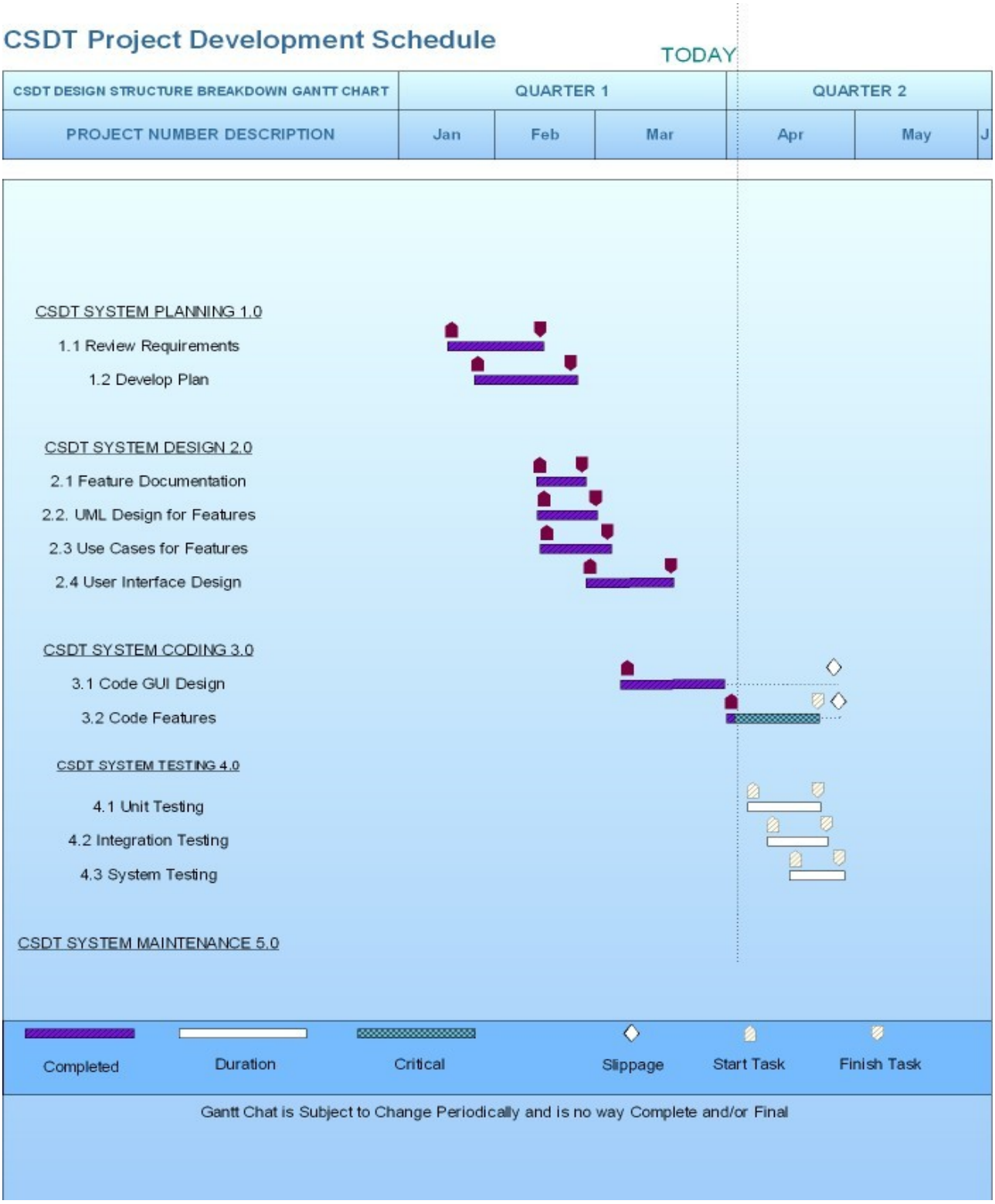


Figure 7. Gantt Chart II. Shows update for timetables within each phase.

Updated 4-Apr-08.

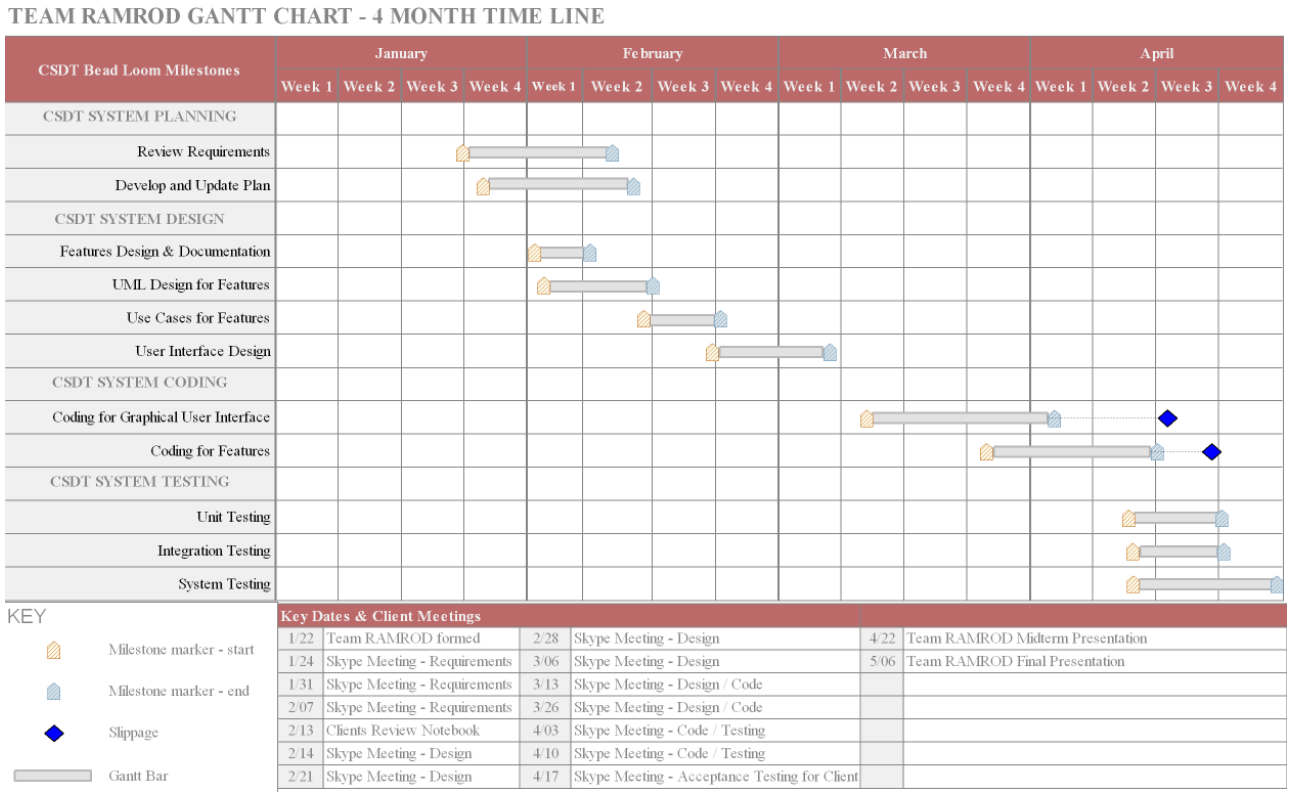


Figure 8. Gantt Chart III. Shows final update for timetables within each phase.

Updated 29-Apr-08.

VI. Project Resources:

1. Currently directed to middle and high school students
2. Windows XP environments.

VII. Staff Organization:

1. Harris Christopher Hollis(Team Lead / Chief Programmer), Chris Blake(Understudy), Joseph Petrizzi(Librarian), Michael Lodge(Programmer), Phillip Gipson(Programmer).
2. Every group member is doing their tasks proficiently.

Requirements Specification

I. Introduction:

1. Overall Description

The Java Virtual Bead Loom CSDT is a tool for middle and high-school students to engage them into experimenting with the simplicity of an English-like pseudo code, which will ultimately make it easier for transition to a higher programming language. The purpose of the document is to provide user scenarios for the features, show diagrams of information, a functional description, and any other requirements related to the design. The program allows students to re-create/draw "beads" onto the screen while using extra functions that will show them pseudo code of what is taking place.

2. Project Constraints

One main overall constraint concerning the software engineering team will be the time needed to complete the project. This project will require many man-hours of concerted effort from our team members. From a monetary standpoint, little capital is lost to development. The only true cost that is of any significance is weighing the opportunity cost of working on one aspect of the system over another. Everyone in the team has other commitments to adhere to outside of class which will hinder the project development. Because of the time allotted to complete this project, some functions may not be 100% reliable and possible become underdeveloped. Another constraint is the complexity of code that is needed to extend the previous prototype project. Although this is a concern, the software engineering team will tackle these issues to the best of their ability.

II. Information Flows

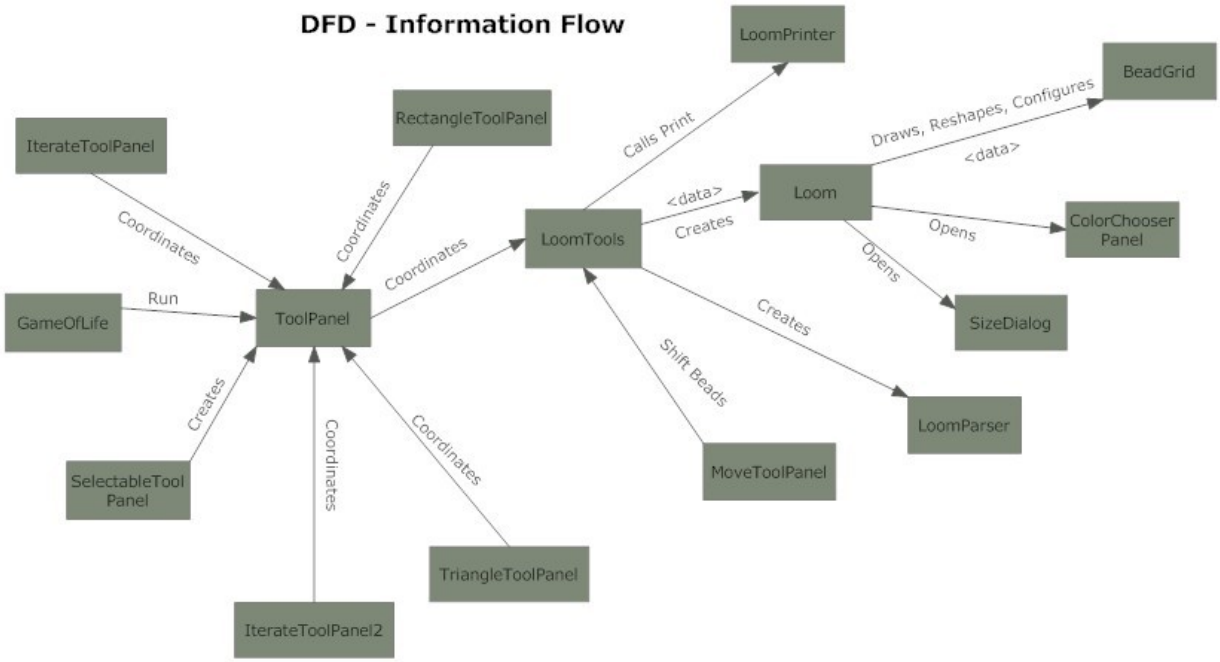


Figure 1. Data Flow Diagram. Shows the relationships between the classes involved in creation of the Virtual Bead Loom software. Relationships are expressed in "English" functionality.

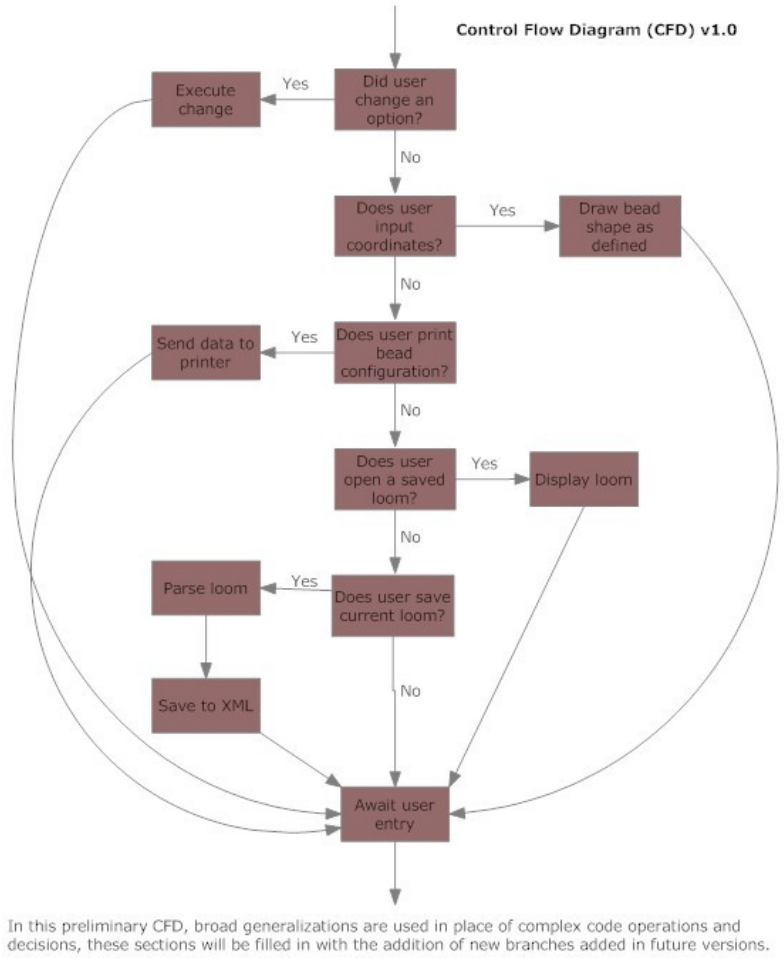


Figure 2. Control Flow Diagram. In this preliminary CFD, broad generalizations are used in place of complex code operations and decisions. These sections will be filled in with the addition of new branches in future versions.

III. Use Case Scenarios for Features

Use Case UC1: Java Virtual Bead Loom **Images** Feature

Scope: Image Selector (Beads)
Level: User-goal
Primary Actor: User (a person who wants to have a different image to select for bead viewing).
Stakeholders and Interests: User can select one image for beads.
Preconditions: The user must have images (JPEG format) stored on machine.
Success Guarantee (Post-Conditions): An image is selected and displayed for the user. The image will remain in the left-hand corner for the user to reproduce in the "graph" feature.

Main Success Scenario (Basic Flow):

1. Program is executed for the user and a Graphical User Interface is displayed.
2. User is displayed a default image in the upper left-hand corner in the “Images” box.
3. Above the Image box, Caption will read “Click for Goal Images”.

Extensions (Alternate Flows 1):

1. Program is executed for the user and a Graphical User Interface is displayed.
2. User is displayed a default image in the upper left-hand corner in the "Images" box.
3. User clicks on the upper left-hand corner (Images) box – above the image goal caption.
4. A dialog box will pop up and will display other images on the user's machine for user's selection.
5. User clicks on same or newly specified image.
6. Same or newly selected image will display in the upper left-hand corner for the user.

Extensions (Alternate Flows 2):

1. Program is executed for the user and a Graphical User Interface is displayed.
2. User is displayed a default image in the upper left-hand corner in the "Images" box.
3. User clicks on the upper left-hand corner (Images) box.
4. A dialog box will pop up and will display other images stored on the user's machine for user selection.
5. User selects an XML file to open.
6. New selected XML will display the image of that XML file in the upper left-hand corner for the user.

Special Requirements:

1. The images interface must be user-friendly.
2. Program will only require images to be in certain format.
3. Images selected must be with in a certain range. (0 KB < Image ≤ 500 KB).
4. Images must be in image format; if the image that is being selected is not in this format, a system dialog box will pop up or a system error message will be displayed. If this occurs, the user will select a different image that meets the images requirements.
5. Image loading time must be preformed in a short amount of time (< 2 seconds)

Frequency of Occurrence:

1. This will depend on the user's special needs, such as selecting a different image.

Open Issues:

1. Can other image file extensions be supported? (e.g. GIF, PNG, etc…)
2. Can the user select many images at once to reproduce multiple images?
3. Can the images feature support higher sizes?

Use Case UC2: Java Virtual Bead Loom **Menu Bar** Feature

Scope: Menu Bar

Level: User-goal

Primary Actor: User (a person who wants to select an option from the menu bar).

Stakeholders and Interests: The user wants to be assured of having access to all possible options within the software

Preconditions: The user must have opened the software.

Success Guarantee (Post-Conditions): The user has selected an option and it was completed successfully

Main Success Scenario (Basic Flow):

1. Clear menu item is selected.
2. User clicks OK in dialog box.
3. The grid is clear of all beads.

Extensions (Alternate Flows 1):

1. Open menu item is selected.
2. User selects a file to open or delete.
3. Desired action is executed.

Extensions (Alternate Flows 2):

1. Save menu item is selected.
2. User names the file and clicks save.
3. File is saved.

Extensions (Alternate Flows 3):

1. Undo menu item is selected.
2. Last action is revoked.

Extensions (Alternate Flows 4):

1. Redo menu item is selected.
2. Stored undone action is reinvoked.

Extensions (Alternate Flows 5):

1. Print menu item is selected.
2. Graph display is printed.

Extensions (Alternate Flows 6):

1. Options menu item is selected.
2. A list of options is presented to the user.
3. Toggle option to have X,Y coordinates follow mouse or stay in top right corner.
4. Resize the area of the grid.
5. Choose display image and size.
6. Create a title for current work.
7. Write notes for current work.
8. Hide/show example works.
9. Close the Options menu.

Special Requirements:

1. Menu bar is always available if the program is running.

Frequency of Occurrence:

1. As often as the user needs to access the various options offered by the menu bar

Open Issues:

1. The save option save the data in an unknown area of the user's computer
2. Switching of the types of beads does not seem implemented
3. If title and notes options are enabled, they cover the section used to manipulate the beads

Use Case UC3: Java Virtual Bead Loom **Graph** Feature

Scope: Bead Graph

Level: User-goal

Primary Actor: User (a person who will use the software to load, produce, or save a created design).

Stakeholders and Interests: User can use the graph to create and modify bead patterns in tandem with the drop down menus on the right hand side of the program.

Preconditions: The CSDT must have already been loaded.

Success Guarantee (Post-Conditions): The user accomplishes the goal of creating an image or recreating an image presented in the upper right image viewer.

Main Success Scenario (Basic Flow):

1. Program is executed for the user and a Graphical User Interface is displayed.
2. User is displayed a blank graph on the interface.
3. User may interact with drop-down and lower right menus.
4. Submission of conditions in drop-down and lower right menus results in proper display of desired beads.

- *User may repeat steps 3 and 4 until satisfied.*

Extensions (Alternate Flows 1):

1. Program is executed for the user and a Graphical User Interface is displayed.
2. User is displayed a blank graph on the interface.
3. User may interact with tool bar menu above the graph.
4. Submission of conditions in tool bar menu results in requested modifications to the graph.

Extensions (Alternate Flows 2):

1. Program is executed for the user and a Graphical User Interface is displayed.
2. User is displayed a blank graph on the interface.
3. User may load a file from the tool bar menu.
4. The graph will load and display the saved file's stored graph schema.

Special Requirements:

1. The graph interface must display a real-time coordinate tracker which will track where the mouse pointer is along the physical graph. It can be toggled to be above the graph or follow the mouse pointer (toggle controls are in above tool bar).
2. The graph must be able to display "beads" of different types (the user may select "bead" types from another menu).
3. The graph must be able to be resized by the user (controlled in another menu).
4. The graph alone will perform no tasks by itself. All tasks must be initiated through other menus or interfaces (tool bar menu, drop-down menu, side menu).
5. Graph interaction should appear instantaneous.

Frequency of Occurrence:

1. This will be used every time the user makes a modification to the graph via a menu.

Open Issues:

1. Should there be permitted direct interactions with the graph in later versions (modified or programmable CSDTs)?

Use Case UC4: Java Virtual Bead Loom **Drop-Down Menu** Feature

Scope: Drop Down Menu

Level: User-goal

Primary Actor: User (a person who wants to change drawing tools)

Stakeholders and Interests: The user wants to be assured of having access to all possible options within the software

Preconditions: The user must have opened the software.

Success Guarantee (Post-Conditions): The user has selected an option and it was completed successfully

Main Success Scenario (Basic Flow):

1. User wishes to place a single bead on the grid.
    1. User selects drop down arrow underneath the Goal Image.
    2. User selects the option titled "Point".

Extensions (Alternate Flows 1):

1. User wishes to place a line of beads across the grid.
2. User selects drop down arrow underneath the Goal Image.
3. User selects the option titled "Line".

Extensions (Alternate Flows 2):

1. User wishes to place a rectangular shape across the grid.
4. User selects drop down arrow underneath the Goal Image.
5. User selects the option titled "Rectangle".

Extensions (Alternate Flows 3):

1. User wishes to place a triangular shape across the grid.
    1. User selects drop down arrow underneath the Goal Image.
    2. User selects the option titled "Triangle".

Extensions (Alternate Flows 4):

1. User wishes to place a custom linear shape across the grid.
    1. User selects drop down arrow underneath the Goal Image.
    2. User selects the option titled "Linear Iteration".

Extensions (Alternate Flows 5):

1. User wishes to place a custom triangular shape across the grid.
    1. User selects drop down arrow underneath the Goal Image.
    2. User selects the option titled "Triangle Iteration".

Special Requirements:

1. Drop down menu is always available if the program is running.

Frequency of Occurrence:

1. As often as the user needs to change drawing shapes.

Open Issues:

1. The possibility of adding more default shapes and to create custom shapes.
2. The example window may not be helpful, so it needs a default option to hide this image.

Use Case UC5: Virtual Bead Loom **Co-ordinate System** Feature

Scope: Co-ordinate system for VBL

Level: User-goal

Primary Actor: User (a person who wants to graph functions onto the grid).

Stakeholders and Interests: User can create different functions and patterns on the grid by inputting values and clicking the "create" button.

Preconditions: The user must select the appropriate function or pattern from the drop-down menu or input their own function (possible addition requested by the client).

Success Guarantee (Post-Conditions): The pattern and/or function will be graphed onto the co-ordinate grid after pressing the "create" button. This function will then be stored into a stack so that the "undo" button can function properly.

Main Success Scenario (Basic Flow):

1. Program is executed for the user and a Graphical User Interface is displayed.
2. User selects a function or pattern from the drop-down menu.
3. User inputs values into the input boxes.
4. User clicks the "create" button.
5. The pattern or function is graphed onto the co-ordinate system with bead images using the values inputted into the input boxes.

Extensions (Alternate Flows 1):

1. Program is executed for the user and a Graphical User Interface is displayed.
2. User inputs his or her own function to be graphed.
3. User clicks the "create" button.
4. The program computes the y-value of the function for each possible x-value in range of the co-ordinate grid and places a bead in the appropriate location on the graph, forming the user's inputted function.

Special Requirements:

1. The user's computer must be capable of running Java code.
2. The input values must be between the minimum and maximum values set for the grid window.
3. If the user chooses to input his/her own function instead of using the drop-down menu for pre-defined patterns, it must be a legal function.

Frequency of Occurrence:

1. A set of beads will be placed on the graph once for each time the "create" button is clicked with values in the input boxes.

Open Issues:

1. We will have to decide on a set of legal functions for the user to be able to input in addition to the drop-down menu options.
2. The "remove" button will be excluded from the program and the user will rely on the "undo" button at the top of the CSDT flash.

IV. Functional Description

1. Functional Partitioning

Image Feature: This is the Virtual Bead Loom Images feature that is located on the upper right-hand portion of the flash CSDT.

Top-Menu Bar: A JMenuBar will be used to create the menu in Java and add the menu items to it.

Graph: This is the Virtual Bead Loom's main graph interface which is located on the left side of the graphical user interface of the flash CSDT.

Drop Down Menu: The Virtual Beam Loom software requires the user to select a geometrical shape to apply a bead pattern to allow the user to explore the various geometric shapes and how to build them using mathematics. The purpose of the drop down menu located in the center region of the toolbar portion of the application is to change the nature of the bead entry options located underneath the menu. In addition, the drop down menu also adds an input example in the form of an image for the new options made available by changing selections.

Co-ordinate System: The co-ordinate system will feature several input boxes and a "create" button, which will graph beads onto the co-ordinate grid. The functions regarding which patterns are created with these beads will depend on which option is selected from the drop-down menu as well as the integer values taken from the input boxes.

2. Functional Description

1. Image Feature:

1. *System Module Narrative*

   The image feature handles the interaction between the user and the system for a desired image.

2. *Performance Issues*

   The speed it will take to load images into the image box and handle the required image.

3. *Design Constraints*

   This must be a user-friendly feature and easy to use.

2. Top Menu Bar Feature:

1. *System Module Narrative*

   Performs the necessary menu functions such as open/close/delete, save to xml, etc…
   This feature will do tasks for the graph feature.

2. *Performance Issues*

   None specified

.

3. *Design Constraints*

   Must be simple to use functions

3. Graph Feature:

1. *System Module Narrative*

   Displays the chosen image onto the screen, no tasks are performed here

2. *Performance Issues*

   None specified

3. *Design Constraints*

   Graph must be able to resize and display the appropriate image.

4. Drop Down Menu Feature:

1. *System Module Narrative*

   The purpose of the drop down menu located in the center region of the toolbar portion of the application is to change the nature of the bead entry options located underneath the menu. In addition, the drop down menu also adds an input example in the form of an image for the new options made available by changing selections.

2. *Performance Issues*

   None specified

3. *Design Constraints*

   This must be easy to navigate for the user.

5. Co-ordinate System Feature:

1. System Module Narrative

The co-ordinate system will feature several input boxes and a "create" button, which will graph beads onto the co-ordinate grid. The functions regarding which patterns are created with these beads will depend on which option is selected from the drop-down menu as well as the integer values taken from the input boxes.

2. Performance Issues

None specified

3. Design Constraints

This feature must be easy to use and must be easy visible.

V. Prototype

This is a rough sketch of the overall prototype. This is the very beginning stages and it only shows the Graphical User-Interface. Code will be provided when thorough analysis is performed on previous version of the bead loom project. The GUI will greatly be enhanced when functions, methods, and testing are performed.

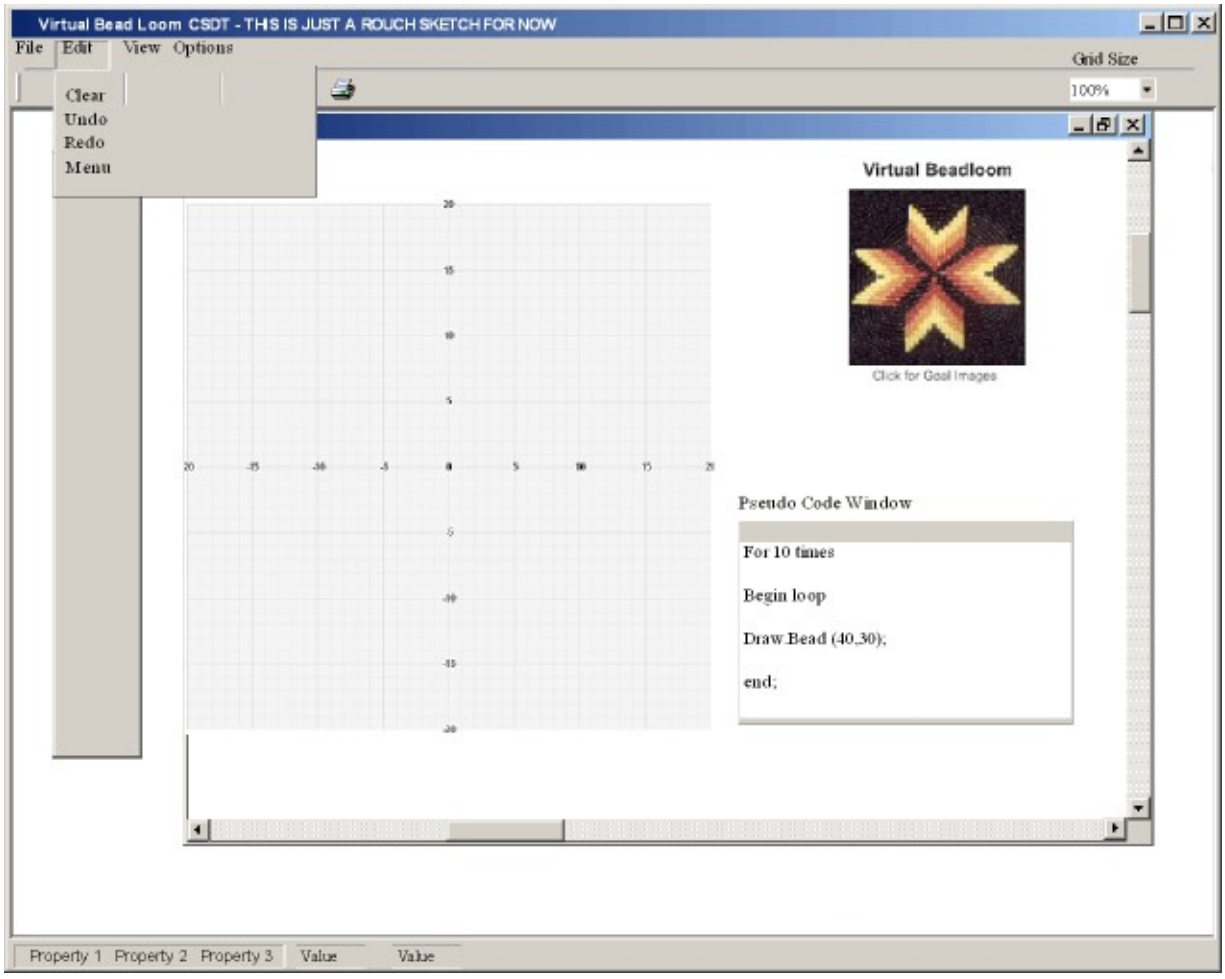


Figure 3. Screenshot of Virtual Bead Loom Interface. Shows all features being utilized within the framework of the GUI.

VI. Validation Criteria

This section describes the characteristics of a successful implementation. This will be useful for qualifying an implementation to guarantee that it conforms to the specifications.

1. Performance Bounds:

1. Java Requirements (Methods/Tools etc…):

Assessing the Java requirements for the images feature will be an important piece for development. Some Java classes and methods will have to be implanted in order to produce this feature. One example would be the "JFileChooser" class at which this will create a file chooser for selecting a file or directory to open or save. A picture of this is below.

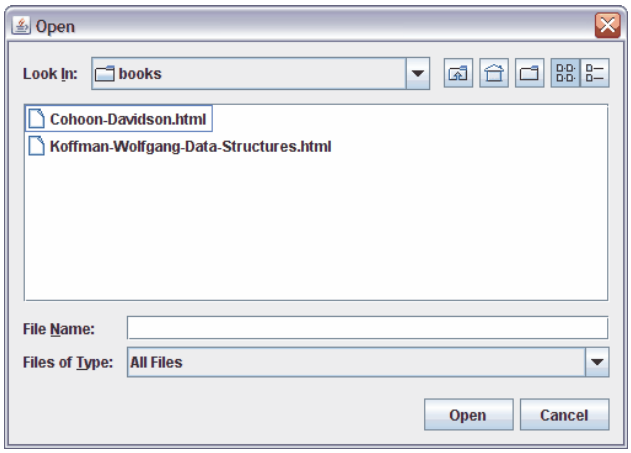


Figure 4. OpenFileDialog. An implementation of file management within the Virtual Bead Loom system. XML will be the chosen file format compatible for the bead files, and the Images feature will open JPEG format images.

For the "File of Type" this will only open files with .JPEG format for now and this is specified in the user scenario requirements. To do this, we will have to import the "FileFilter" class that will allow us to do this.

Some techniques to mimic the flash would be to use some tools from the Java swing library. One for example is the "SwingWorker" class which does the background image loading. Advantages for using this is for loading each image and compute its thumbnail in a background thread. This will speed up the image loading significantly. An example of using Java to mimic the flash images window would be something like this (although interface would be changed)…

Figure 5. A probable example of the Image Viewer within the Virtual Bead Loom user interface.

For the Co-ordinate feature, some methods will be used; for example, when the "create" button is clicked, it will call a create() method with the input values. The create() method will check the value of the option selected in the drop-down menu. The create() method will then call a graph method with the selected values from the input boxes. One of these methods will be able to parse, calculate, and graph primitive trigonometric functions onto the bead grid as per a request from the client.

Design Specification

I. Introduction:

1. Scope and Purpose:

1. System Objectives:

The main objective is to create a design for the system that correlates with the requirements given by the clients. This will being by providing a Java user interface for the flash version of the Virtual Bead Loom. The GUI will provide the user with tools necessary to learn a customized pseudo code to introduce them to programming. This will include drop-down menus and buttons capable of editing the Java code in the background and will show the user the changed pseudo code so that they will understand what it is doing. We also plan to implement a step-through feature so the user can see what is going on at run time, such as during loop iterations.

2. Major Software Functions:

1. Provide a Visual Interface

The user will be able to see a visual representation of what their code is going via the virtual bead loom interface. The GUI will show the beads placed in grid locations based on selections the user has chosen on the drop-down menu or what trigonometric functions they have inputted on their own.

2. Show User Modified Pseudo code

Upon graphing a function, the user will see a customized pseudo code agreed upon by ourselves and the client that will be simple enough for middle school students to understand, but as powerful as possible. The pseudo code will not be directly editable by the user, but will change when drop-down menu selections are changed and when the visual graph of the virtual bead loom changes.

3. Show Code at Runtime

The user will be able to see how their code changes at run time, such as during iterations of for loops. The user will be able to clock a button to step through for each iteration.

4. Save Files in an XML Format

Users will be able to save their work in an XML format for later user by themselves and by their instructor.

3. Major Design Constraints:

It may be difficult to agree upon a pseudo code simple enough for middle school students to understand; yet powerful to perform the operations we need it to do. We will have to make our program capable of verifying, graphing, and showing pseudo code for trigonometric functions inputted by the user, not just those available from the drop-down menus.

2. Reference Documents:

CSDT: Virtual Bead Loom: http://www.ccd.rpi.edu/Eglash/csdt/na/loom/index.html

3. Design Description

1. Data Description

Our application will take data from the GUI's drop-down menus and from simple trigonometric functions inputted by the user to be graphed onto the VBL.

4. File Structure and Global Data

We are still in the process of designing our class structure for the program. An in-development look at the class hierarchy is below:

GUIGoalImages (extends JInternalFrame, implements java.awt.event.ActionListener)

Responsible for handling images for the Image Viewer.

GUIOutputWindow
(extends JInternalFrame)

Creates and handles an internal panel for source code and pseudo-code to be displayed (GUI shell, implemented by a later team).

BeadPanel
(extends JPanel)

Creates a panel to place beads on the grid.

GridPanel (extends Jpanel)

Creates a panel upon which the bead grid is displayed.

GUIMovePanel
(extends JPanel)

Creates a panel with tools to manipulate beads.

BeadLoom (extends JApplet, implements java.awt.event.ActionListener, java.awt.event.MouseListener, java.awt.event.MouseMotionListener, java.awt.print.Printable)

Main class, assembles GUI, controller.

GUIInputTools
(extends JApplet)

Creates panels to create individual or patterns of beads.

CoordList

Contains arraylists which control mapping of beads to grid.

GUIMenuBar

Creates a menu bar for the GUI and maintains its functions.

Layer

Maintains layers of coordinate lists to enable undo functions and individual layer operations.

SaveLayer

Saves arraylists into XML and reads from XML into arraylists.

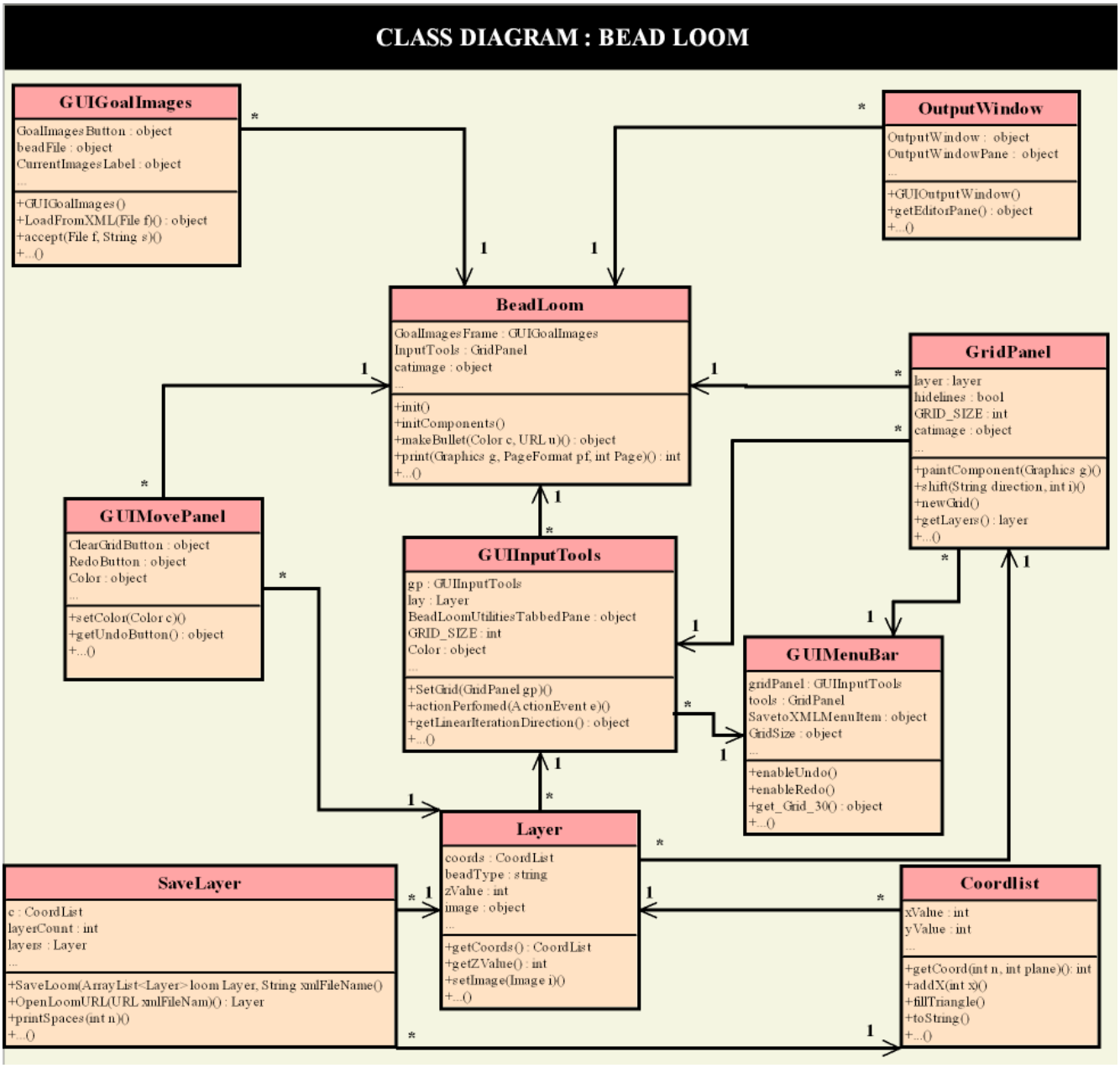


Figure 1. UML Class Diagram. The UML Class Diagram shows the main classes used in the creation of the Virtual Bead Loom and its important attributes and functions.

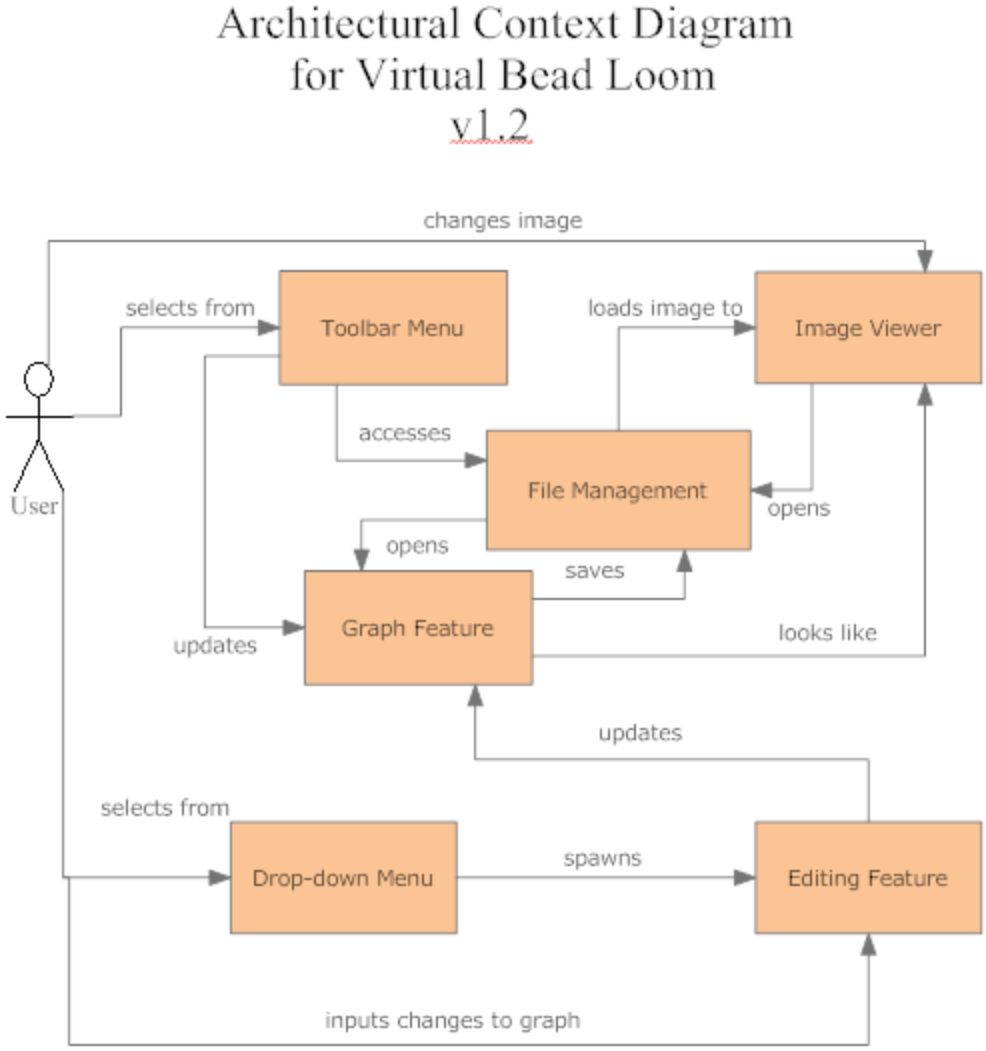


Figure 2. Architectural Context Diagram. Shows the relationships between features in the VBL architecture.

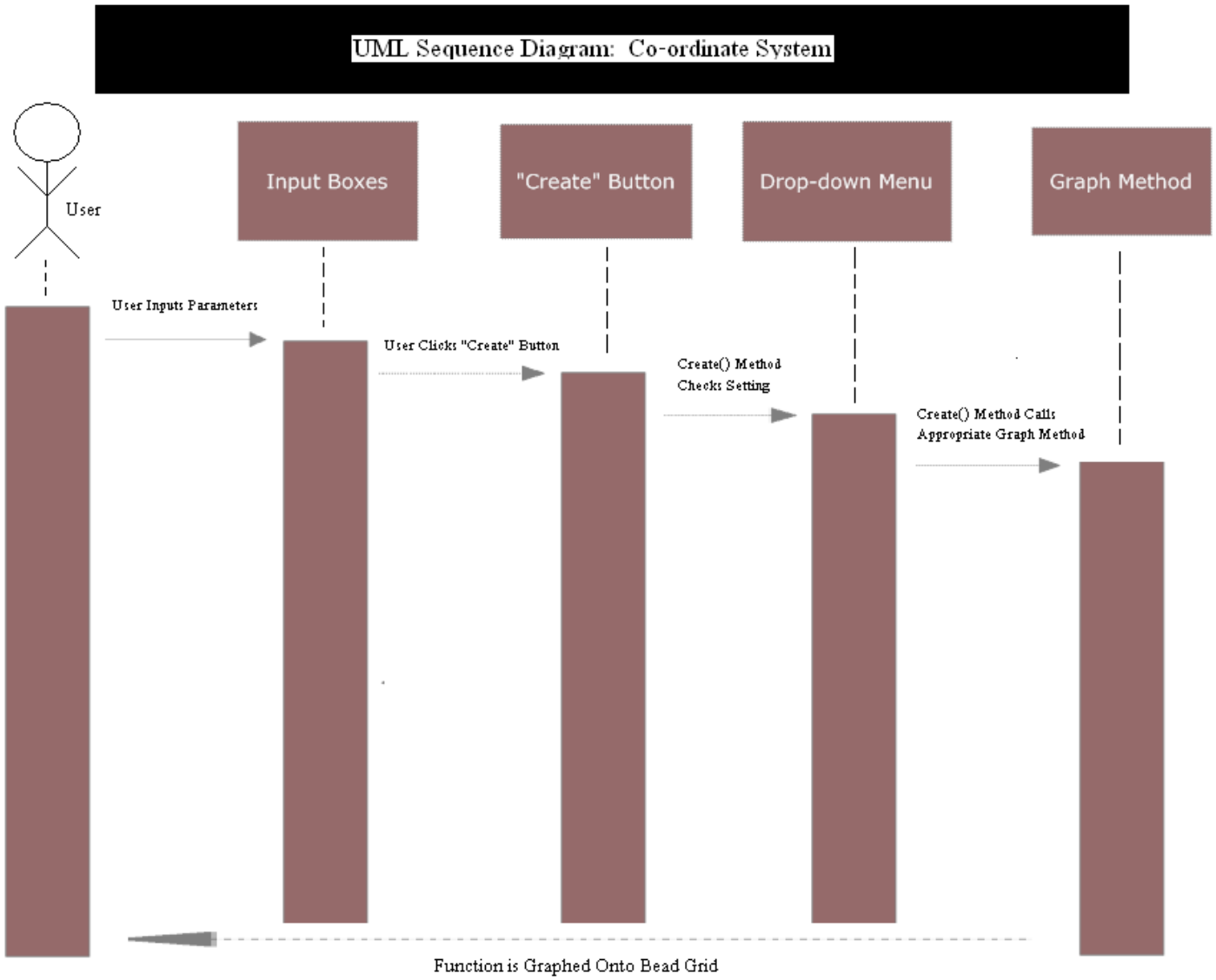


Figure 3. Sequence Diagram: Coordinate System. Shows the sequence operations design for what happens when a user creates a new bead or bead pattern.

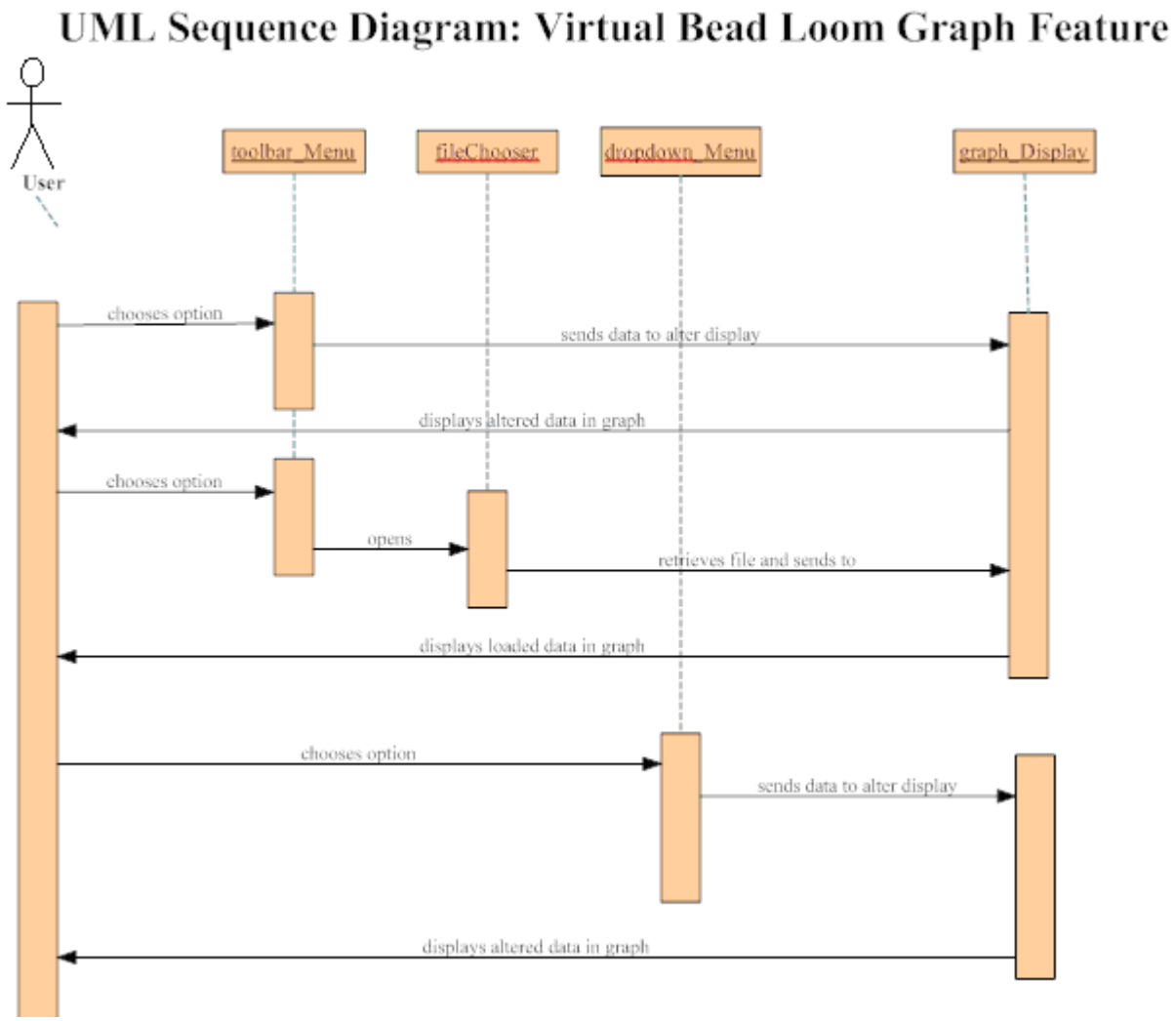


Figure 4. Sequence Diagram: Graph Feature. Shows sequences of interactions between the graph display and other segments of the VBL system.

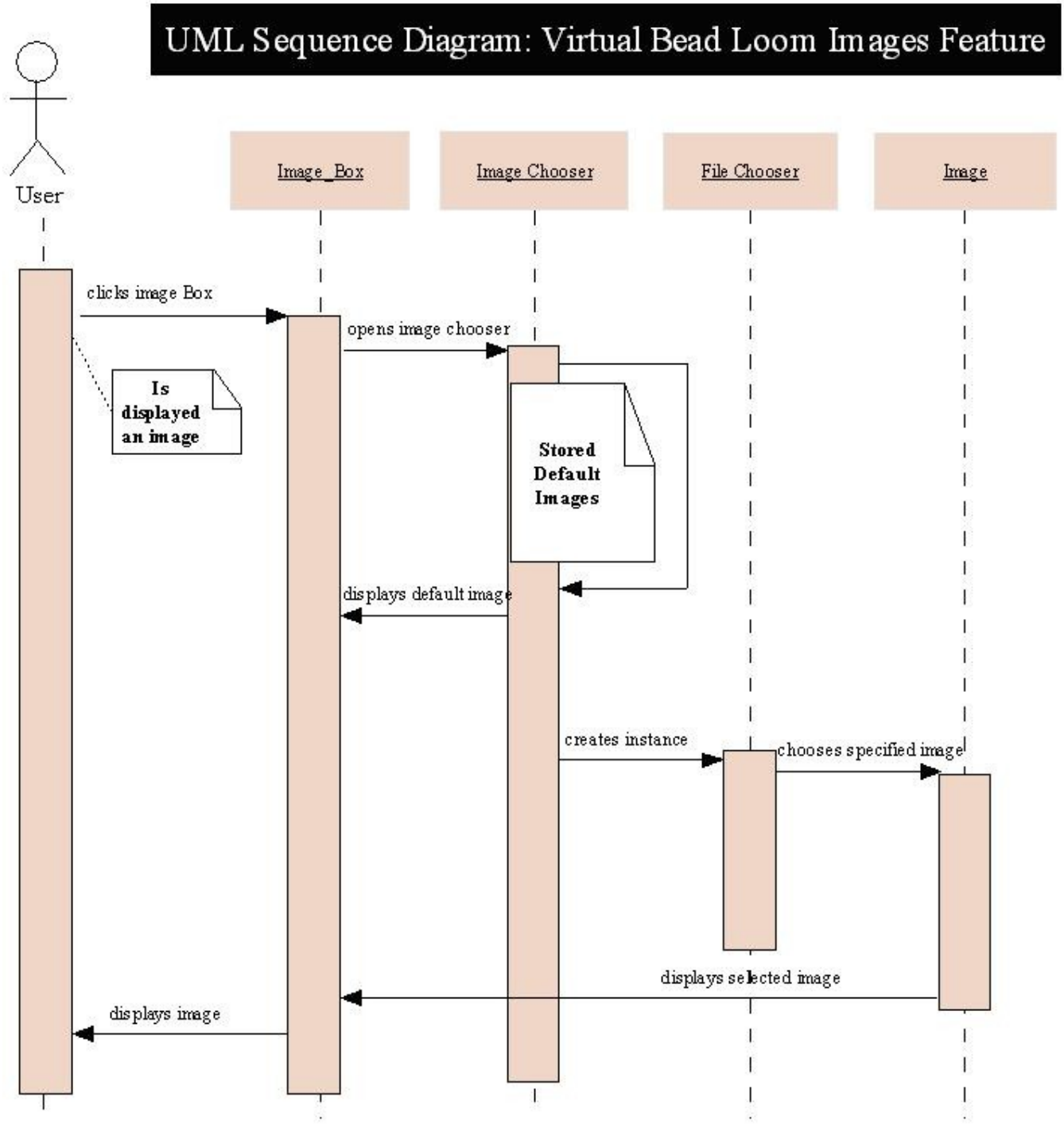


Figure 5. Sequence Diagram: Image Viewer. Shows sequence of interaction with the selection of images in Image Viewer.

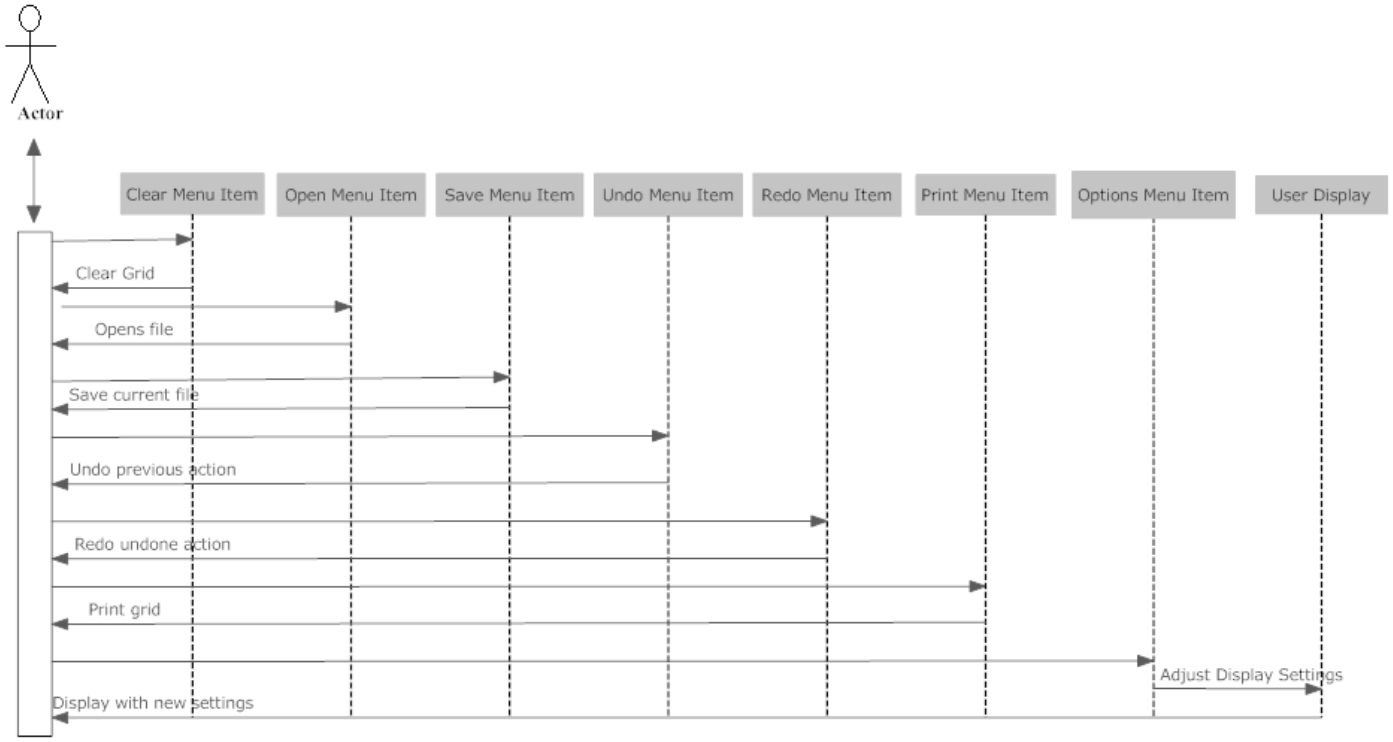


Figure 6. Sequence Diagram: Menu Bar. Shows sequence of events when various menu items are selected from the toolbar.

Appendix:

Through the coding phase, it was noted that a drop-down menu was unnecessary as a redesign of the user interface allowed for the division of panes into tabs. Using this, there was a tabbed controller panel made for the creation of objects. The "removal" of this drop-down feature is shown in the following update of the Images sequence diagram. Instead, additional XML functionality is shown. In the previous diagrams, substituting the drop-down menu with the tabbed pane is suggested.

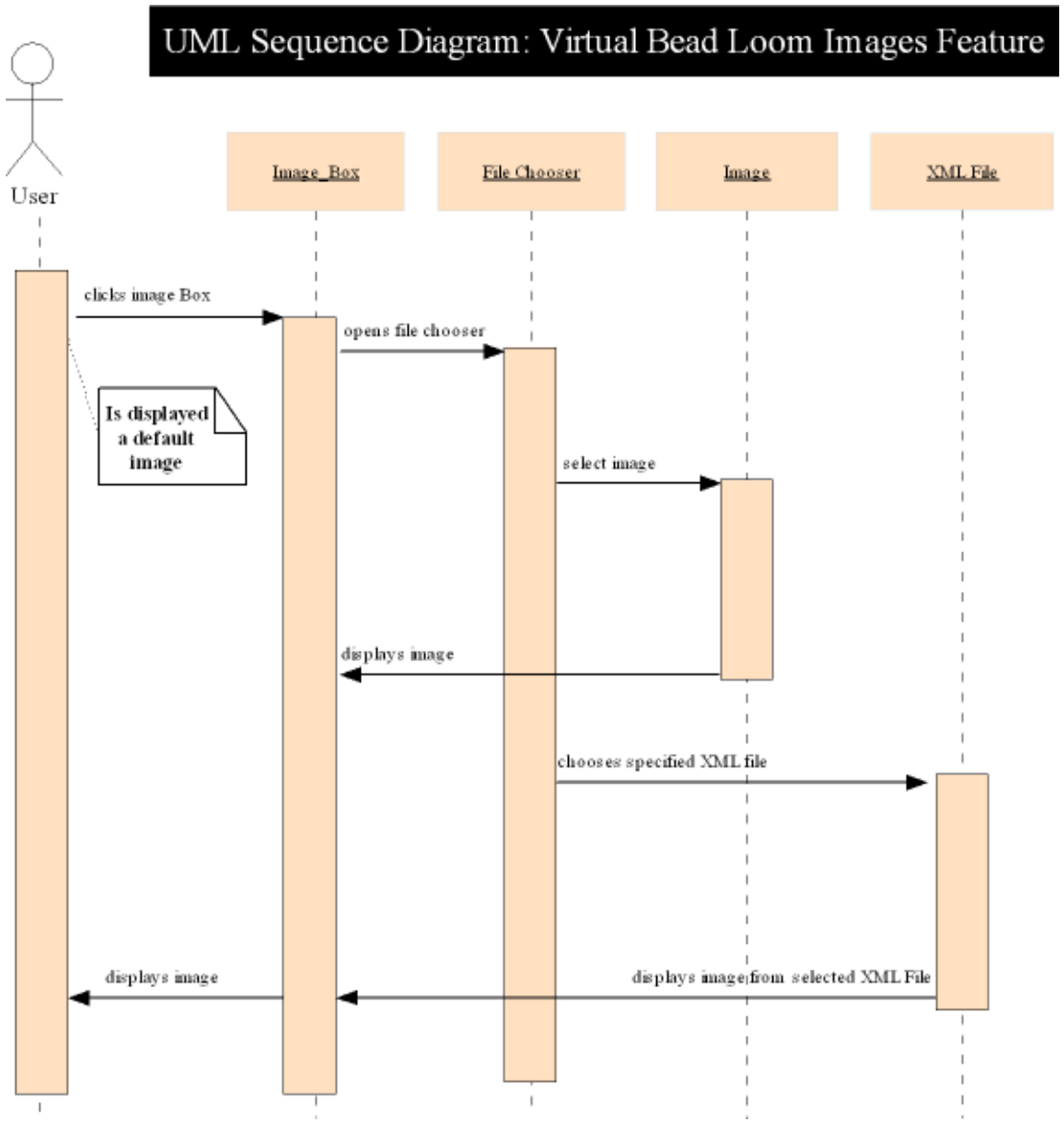


Figure 7. Updated Sequence Diagram: Images Feature. The drop-down menu feature was eliminated from the program, and additional functionality for XML is shown instead.

Software Verification and Validation

I. Purpose of Software Verification and Validation Plan:

1. **Specific purpose and scope**

The purpose of this document is to set out with a plan to begin verification and validation tasks for the Virtual Bead Loom CSDT. Within this document are clear plans for the project's organization, communication and documentation, schedule, resources (including manpower, tools, and costs), and a summary of the plans for all verification and validation tasks to be performed.

2. Waivers from standard

The Virtual Bead Loom project is not a massive system, nor is it entirely system-critical. However, it is treated as a priority system and thusly, as much of the standard for the SVVP will be completed as needed. Unnecessary or inapplicable portions of this standard will not be utilized in this plan.

3. Identification of software project and products

The Virtual Bead Loom project is part of the CSDT program, which will help teach and introduce to students in grades K-12 computer programming. This specific project allows students to interact with a pseudo-code language to make and mimic designs based on algorithmic interaction.

1. Goals

The goals for this V&V Plan are to take steps to ensure that the enacting of this project keeps within the overall goals for the project set out by the clients, and to begin formulating tests to make sure that all portions of the software project do indeed work as necessary. Additionally, V&V will be performed to ensure that as the project passes through life-cycle phases, all necessary documents and tasks are achieved properly.

Validate requirements with design documents
Verify source code is done according to standards
Thorough testing according to documentation

II. References and Definitions:

1. **Definitions**:

1. Acceptance Testing - *Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system*

2. Anomaly – *Anything observed in the documentation or operation of software that deviates from expectations based on previously verified software products or reference documents.*

3. Component Testing - *Testing conducted to verify the implementation of the design for one software element (unit, module) or a collection of software elements.*

4. Concept Phase – *The initial phase of a software development project, in which the user needs are described and evaluated through documentation.*

5. Critical Software – *Software whose failure could have an impact on safety, or could cause large financial or social loss.*

6. CV - *Team/Person that checks the verification and validation (Joe Petrizzi)*

7. Design Phase – *The period of time in the software life-cycle during which the designs for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements.*

8. DN – *Team/Person(s) responsible for driver and navigator. (Michael Lodge, Phillip Gipson)*

9. Implementation Plan – *The period of time in the software life cycle during which a software product is created from design documentation and debugged.*

10. Installation and Checkout Phase - *The period of time in the software life cycle during which a software product is integrated into its operational environment and tested in this environment to ensure that it performs as required.*

11. Integration Testing - *An orderly progression of testing in which various software elements and/or hardware elements are integrated together and tested. This testing proceeds until the entire system has been integrated.*

12. Life-Cycle Phase – *A period of time during software development or operation that may be characterized by a primary type of activity that is being conducted. These phases may overlap on another; for V&V purposes, no phase is concluded until its development products are fully verified.*

13. LE - *Lead Software Engineer.*

14. Minimum Task – *Those V&V tasks applicable to all projects.*

15. Operation and Maintenance Phase - *The period of time in the software life cycle during which a software product is employed in its operational environment monitored for satisfactory performance, and modified as necessary to correct problems or to respond to changing environments.*

16. Optional Tasks – *Those V&V tasks that are applicable to some, but not all software, or that may require the use of specific tools or techniques.*

17. Required Inputs – *The set of items necessary to perform the minimum V&V tasks mandated within any life-cycle phase.*

18. Required Outputs – *The set of items produced as a result of performing the minimum V&V tasks mandated within any life-cycle phase.*

19. Requirements Phase - *The period of time in the software life cycle during which the requirements, such as functional and performance capabilities for a software product, are defined and documented*

20. Software Design Description – *A representation of software created to facilitate analysis, planning, implementation, and decision making. The software design description is used as a medium for communicating software design information, and may be though as of a blueprint or model of the system.*

21. Software Verification and Verification Plan - *A plan for the conduct of software verification and validation.*

22. Software Verification and Validation Report - *Documentation of the V & V results and appropriate software quality assurance results.*

23. System Testing - *The process of testing an integrated hardware and software system to verify that the system meets its specified requirements.*

24. Test Case – *Documentation specifying inputs, predicted results, and a set of execution conditions for a test item.*

25. Test Design – *Documentation specifying the details of the test approach for a software feature or combination of software features and identifying the associated tests.*

26. Test Phase – *The period of time in the software life-cycle in which the components of a software product are evaluated and integrated, and the software product is evaluated to determine whether or not requirements have been satisfied.*

27. Test Plan – *Documentation specifying the scope, approach, resources, and schedule of intended testing activities.*

28. Test Procedure – *Documentation specifying a sequence of actions for the execution of a test*

29. US - *Understudy for the project (Chris Blake)*

30. Validation - *The process of evaluating software at the end of the software development process to ensure compliance with software requirements.*

31. VBL – *Abbreviation for Virtual Bead Loom, the project being verified and validated.*

32. Verification - *The process of determining whether or not the products of a given phase of the software development cycle fulfill the requirements established during the previous phase.*

III. References:

1. The Institute of Electronic Engineers. (1986). *IEEE Standard for Software Verification and Validation Plans*. New York

2. CriTech Research. (1999). *Verification and Validation*. http://www.critech.com/w.htm

IV. **Verification and Validation Overview**

1. Organization:

This shows the organization of all tasks for this document.

The CV Team checks the verification and validation of the activities accordance to the project. The DN Team performs V&V as they perform as driver and navigator. Both SS and TT Teams create the anomaly reports as needed.

The US reports problems to the LE. It is the responsibility of the LE that these problems are solved.

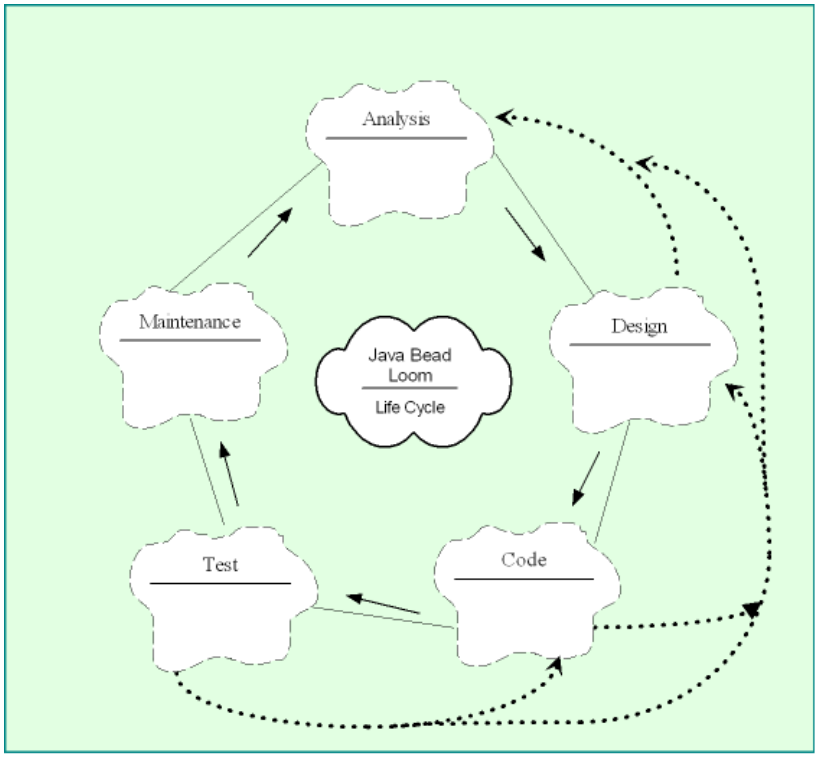
Person or Group	Team Members
LE	Harris Christopher Hollis
US	Chris Blake
CV	Joe Petrizzi
DN	Michael Lodge And Phillip Gipson

All lines of communication through the V&V process will be via emails within the team's five members, the professor, and the clients, in addition to already running twice(or more)-weekly meetings. Further meetings will be made amongst group members as needed as the deadline for software development nears.

The authority for resolving issues and approving V&V tasks and deliverables will be LE. All tasks will be delegated by him, and reports will be turned into him. After these changes or modifications are made, if any, the V&V deliverables will be added to the group notebook in final version and sent on to the clients.

2. Master Schedule:

1. Project Life Cycle, Milestones, and completion dates. *(subject to change)*



Milestone	Deadline
Validate requirements with design documents	4/20/08
Verify source code is done according to standards	4/20/08
Make sure changes in interface information is done correctly	4/20/08
Thorough testing according to documentation	4/20/08

3. Feedback and Development

    1. The client and end user will be giving feedback for the project. This will ensure that the development will adhere to the client's requirements. Notification to any changes will be presented from the end user and client.

3. Summary of Resources

    1. The LE of the project is Harris Christopher Hollis. Our team collaborated on the design and development of the project and used the Eclipse IDE platform software. Eclipse is used for our debugging and testing for complex features. We also used an Object Oriented Metrics program to test the project code thoroughly. Our time put forth into this project was the only demanding resource we used. Some special needs were required and they are listed below.

1. Security: Access rights to documentation will be limited by staff via Google Docs.
2. Access Rights: All staff members will have full access to the project. The professor and client will be updated on progress periodically.
3. Documentation Control: Client and professor will be given access to completed documents only. Staff will be aloud access to all complete and incomplete documents.

4. Responsibilities

    1. Responsibilities per team member are divided up.

        1. **Harris Christopher Hollis** – Team Lead, Administrator, Designer, Tester, Junior Programmer, Analysis Phase, Coding Phase, Testing Phase.

        2. **Chris Blake** – Understudy, Designer, Team Recorder, Tester, Senior Programmer, Design Phase, Coding Phase, Testing Phase.

        3. **Joe Petrizzi** – Team Librarian, Tester, Junior Programmer, Requirements Phase, Testing Phase, Maintenance Phase, Coding Phase.

        4. **Phillip Gipson** – Senior Programmer, Tester, Coding Phase, Testing Phase, Maintenance Phase

        5. **Michael Lodge** – Senior Programmer, Tester, Coding Phase, Testing Phase, Maintenance Phase

5. Tools, Techniques, and Resources

    1. Software Tools, Techniques, and Resources:

    Eclipse IDE software is the main tool that we used for this project. Unit Testing Tool – (Junit) is used for testing features. Other tools for black box testing were used by IBM Rational Testing Suite.
    Some techniques we used were for graphing purposes to show analysis and design – the software programs that we used for this is SmartDraw (for graphing) and Costar 7.0 Demo Version (for COCOMOII model).

    Various internet based web sites for programming code were used as reference and resources – these websites are cited when they are used. Financial resources included are Georgia Southern University and the National Science Foundation.

    A Java applet of a prototype version of our software has been supplied, so to view it we are going to use an Internet browser, currently Firefox Mozilla. Firefox is also being used to observe the Flash version of the CSDT as well.

    The entire Eagle Lab in Georgia Southern University's College of Information Technology is available for use. Each computer in the Eagle Lab is installed with the same software packages, so no differences can be accounted for. No other resources were used except for the time spent from each member.

    2. Acquisition, Training, and Support:

    All materials (computers, documentation, etc) are being directly supplied to the development

team by executive members and customers, supported by their respective organizations.

The clients are met with weekly to provide ideas and feedback. Also, appointed customers work with a testing group for determining positive aspects of the program in respect to accessibility.

V. Life Cycle Verification and Validation

This is a detailed plan for each phase of the V&V tasks in management.

Methods of criteria must meet predetermined boundaries for correctness, accuracy, testability objectives, criticality, and performance. The boundaries included communication amongst team members, and the documentation must meet expectations.

According the schedule and resources, the deadlines are set for April 20th and must be completed by this time. Milestones for the project concerning the V&V tasks were completed by the set date.

1. Concept Document Evaluation:

With referring to the Feasibility Documentation, the predictions made were changed throughout the semester. The overall program inherits all other previous functions from the previous java program with the exceptions of adding some features. The programmable features were not feasible and therefore will have to be completed by another team. The GUI was re-done for design purposes and is more user-friendly than before. The GUI has an Output Window Frame for future implementation– this window will be the programmable features when completed.

VI. Requirements Phase

The V&V tasks for this phase includes: Traceability Analysis, Evaluation, Interface Analysis, System Test Plan Generation, and Acceptance Test Plan Generation.

1. Software Requirements Traceability Analysis

Upon comparing the SRS to our project plan and our concept documents, we have concluded that most of the information remains the same except some minor changes with the GUI interface. Listed below is a table representing the percentage of acceptability for the users to trace the required documentation.

Requirement	Trace
Accept if 100% of customers analyze software requirements to determine if they are consistent with, and within the scope of, system requirements.	Requirements can be seen in document "Requirements Specification v1.0.1" Page 2
Accept if 100% of customers that the requirements completely satisfy the capabilities specified in the concept documents for the Project Plan	Refer to Requirements Specification v1.0.1 and the Concept Phase in the Software Verification and Validation Plan v1.0

2. Software Requirement Evaluation

The requirements have not changed with concerning the SRS document, all others remain the same. Only changes are within the interface itself.

75% of users must agree that the features are accurate.
75% of the users must agree with the prioritization of the features.
75% of the users must agree with the criticality of the system.

All features of the VBL are testable:

1. Image Feature
2. Top Menu Bar
3. Graph
4. Drop Down Menu
5. Co-ordinate System

3. Software Requirements Interface Analysis

Currently, there are no known interface bugs with the flash bead loom already in place. The coordinate input boxes are the only user input boxes which are constrained to double integer inputs with only numerical characters with the exception of the period (.) to denote a decimal point. The current interface will be improved by adding layer functionality and removing the "Remove" button from the bead input section to avoid confusion with the students.

The input boxes were tested by entering all different invalid types of input.

4. Software Test Plan Generation

These are tests that are designed for the performance and functionality of the system. The testing will be done by each member of the team and therefore will enhance efforts for generating tests.

5. Acceptance Test Plan Generation

This standard only applies if Department X is playing a role in the development of the Acceptance Test Plan. If the customer is assuming full responsibility for developing the Acceptance Test Plan, then this standard does not apply.

VII. References

CM051 *Documentation Format Guideline*

SD060 *Acceptance Testing Procedure*

http://www.tbs-sct.gc.ca/emf-cag/acceptance/outline/atpo-vper_e.asp

VIII. Approvals

Development Manager.

IX. Responsibilities

The Project Manager (see *Acceptance Testing Procedure*) is responsible for ensuring that an Acceptance Test Plan and Test Cases are produced.

The Project Manager is responsible for ensuring that the Acceptance Test Plan and Test Cases are updated when requirements change.

Test Case 1: Accept the software if 75% of users are satisfied with its performance

Requirements Traceability:

1. Cross references with minimal satisfactory users requirement in Software Requirements Specifications ver. **1.0** Project Constraints

Description of Test:

1. A random selection of users will be given the opportunity to test the current version of the software. They will rate their satisfaction with the product based on a set of predetermined questions using a rating scale of 1-10, with 10 being extremely satisfied and 1 being extremely dissatisfied. An average rating of 6 has been determined to be the threshold of satisfaction.

Expected Results:

1. The development team predicts that at least 80% of the users will be satisfied with the current version of the software. The users' complaints/comments will be evaluated towards improving the product before final release.

Test Case 2: Accept the software if 75% of user test runs are relatively bug/error-free

Requirements Traceability:

1. Cross references with minimal satisfactory users requirement in Software Requirements Specifications ver. **1.0** Use Case Special Requirements

Description of Test:

1. A random selection of users will be given the opportunity to test the current version of the software multiple times. They will record any errors they had with a particular test run. If less than 75% of

the test runs are error free, then the software will be reviewed by the development team. If more than 75%, software will be considered relatively error-free.

Expected Results:

1. The development team predicts that at least 85-90% of the test runs will end error-free. Any documented errors will be resolved.

Test Case 3: Accept the software if 100% of customers would potentially consider purchasing the software.

Requirements Traceability:

1. Cross references with minimal satisfactory users requirement in Software Requirements Specifications ver. **1.0** Project Constraints

Description of Test:

1. All customers will be given the opportunity to test the current version of the software. Each customer will either pass or fail the software based on their criteria. If all deem the software ready, it will head into distribution

Expected Results:

1. The development team predicts that all of the customers will feel that their requirements have been met. Post-distribution maintenance will be arranged.

X. Implementation Phase Verification and Validation

The V&V tasks for this phase includes: Source Code Traceability Analysis, Source Code Evaluation, Source Code Interface Analysis, Source Code Documentation Analysis, Test Case Generation, Test Procedure Generation, and Component (Unit) Test Execution.

***Important note: The Implementation Verification and Validation document will only contain analysis and references to the GridPanel class, which the bulk of work was done in pair programming.

1. Source Code Traceability Analysis

The following table will analyze the source code for direct references from the code itself to items or designs previously discussed or mentioned in any of the previous design documentation.

Source Code Module	Design Document Reference(s)	Accuracy Comparison of Code to Reference(s)
private void initCells()	USRv1.1.1s4.1, USRv1.1.1s5.3.Fig1	Creation of grid conforms to reference.
protected void paintComponent(Graphics g)	RQSv1.1s4.3, RQSv1.1s5.2.3	Currently being worked on to conform to drawing/showing of the grid.
public JLabel findMousePosition(MouseEvent e)	RQSv1.1s2.3.1	Coordinates display conforms to reference

XI. Source Code Evaluation

Source Code Segments	Adherence to Code Standards
private JLabel MoveLastBeadsLabel; private JDesktopPane BeadLoomDesktopPane; private GUIGoalImages GoalImagesFrame;	names of variables are capitalized according to the words that it is comprised of

if (ok) {
try {
job.print();
}
catch (PrinterException ex) {
/* The job did not
successfully complete */
}
}
	when a bracket is used for class declarations, methods, and loops/statements, a new line will be used. The beginning of the new line will be indented one tab worth of white space

Although the source code has been completed, testing time only allowed for test suites to be performed for the GridPanel class (unit testing), rudimentary integration and system testing, and acceptance testing.

XII. Source Code Documentation Analysis

While the development team is lightly commenting their code for brief memory refreshers, these comments have already been deemed insufficient for a final product. There are plans to go back through the code once the project has reached a stable build. During the code review, the team has thoroughly documented all aspects of code so that any maintaining teams that come afterwards will be able to acclimate quickly to the system.

XIII. Test Case and Procedure Generation

For the unit testing, a report will be generated for each type of tests. The reason for these reports is to maintain a report history of testing procedures for the components and to verify the results again for testing at later dates. Each type of tests has fields that must be filled out for that component to be tested. The form is specified below.

Tester Name: *(This is the person's name that is testing the program)*	**Date:** *(This is the current date when test is performed)*
Time Began: *(This is the actual time for start of tests)*	**Time Ended:** *(This is the actual end time for ending the tests)*
Instructor: *(This field can be left blank if there is none)*	**Program ID:** *(This is the Testing unit's ID field for the type of test given)*
Program Test Name: *(This is the Test name from the Program ID field)*	
Objective: *(Specifies what the tests should perform)*	
Description: *(Specifies a valid description of the type of tests to perform)*	
Conditions: *(A set of conditions for the tests to perform under)*	

Expected Results: *(A set of expected results that the testing should return)*
Actual Results: *(A set of actual results that the tests returns)*

XIV. Component (Unit) Testing

The testing consisted of using the GUIInputTools class. DrawPoint, DrawLine, DrawRectangle, and DrawTrigFunctions were used in these tests. Unit testing of these features were done by a test suite class (UnitTest.java) and the controller class (BeadLoom.java) to call the methods to test. Code Inputs are shown in 6a and Code Outputs are shown in 6b.

Draw Point X Test Reports

Tester Name: Chris Hollis	**Date:** 4/12/08
Time Began: 17:30	**Time Ended:** 17:40
Instructor: Dr. Bell-Watkins	**Program ID:** DPT1A
Program Test Name: Draw Point Test 1a	
Objective: To verify that specified inputs show on the grid accordingly	
Description: Preset inputs are used for X value. (Y-value preset to 0)	
Conditions: Inputs are preset to integers and non-integers.	
Expected Results: Output will state "Layer added" if correct input is valid. Output will throw and state "NumberFormatException" if input is invalid.	

Actual Results:
 	
`Testx1: java.lang.NumberFormatException: For input string: "A"`
`Testx2: Layer added.`
`Testx3: java.lang.NumberFormatException: For input string: "5.5"`	

```
Testx4: java.lang.NumberFormatException: For input string: " "
Testx5: Layer added.
```

Tester Name: Chris Hollis	**Date:** 4/12/08
Time Began: 17:45	**Time Ended:** 17:50
Instructor: Dr. Bell-Watkins	**Program ID:** DPT1B
Program Test Name: Draw Point Test 1b	
Objective: To verify that specified inputs show on the grid accordingly	
Description: Preset inputs are used for the Y value. (X-value preset to 0)	
Conditions: Inputs are preset to integers and non-integers. Code Inputs are shown in Section 6a.	
Expected Results: Output will state "Layer added" if correct input is valid. Output will throw and state "NumberFormatException" if input is invalid.	

Actual Results: Throws a NumberFormatException for a non-integer value.
 	
`Testy1: java.lang.NumberFormatException: For input string: "B"`
`Testy2: Layer added.`
`Testy3: java.lang.NumberFormatException: For input string: "2.3"`
`Testy4: java.lang.NumberFormatException: For input string: " "`
`Testy5: Layer added.`	

Draw Point Y Test Results

Draw Line Test Results

Tester Name: Joe Petrizzi	**Date:** 4/12/08
Time Began: 17:55	**Time Ended:** 18:05
Instructor: Dr. Bell-Watkins	**Program ID:** DPT1B
Program Test Name: Draw Line Test	
Objective: To verify that specified inputs show on the grid accordingly	
Description: Preset inputs are used for the X1, Y1, X2, and Y2 values.	
Conditions: Inputs are preset to integers and non-integer types. Four sub tests are performed for inputs. Code Inputs are shown in section 6a.	
Expected Results: Output will state "Layer added" if correct input is valid. Output will throw and state "NumberFormatException" if input is invalid.	

Actual Results: Throws a NumberFormatException for a non-integer value.
 	
`TestLineXY: Layer added.`
`TestLineXY1: Layer added.`
`TestLineXY2: Layer added.`
`TestLineXY3: java.lang.NumberFormatException: For input string: "$"`	

Draw Vertical Line Test Results

Tester Name: Chris Hollis	**Date:** 4/12/08
Time Began: 18:15	**Time Ended:** 18:20
Instructor: Dr. Bell-Watkins	**Program ID:** DVLT
Program Test Name: Draw Vertical Line Test	
Objective: To verify that specified inputs show on the grid accordingly	
Description: Preset inputs are used for the X1, Y1, X2, and Y2 values. Inputs for X1 are equal to X2 and Y2 has a lesser value than Y1. (Vertical Line, undefined slope.)	
Conditions: Inputs are preset to integers and non-integer types. Three sub tests are performed for inputs. Code Inputs are shown in section 6a.	
Expected Results: Output will state "Layer added" if correct input is valid. Output will throw and state "NumberFormatException" if input is invalid.	
Actual Results: `TestVLineXY1: java.lang.NumberFormatException: For input string: " "`	

```
TestVLineXY: Layer added.
TestVLineXY2: Layer added.
```

Draw Horizontal Line Test Results

Tester Name: Chris Hollis	**Date:** 4/12/08
Time Began: 18:25	**Time Ended:** 18:30
Instructor: Dr. Bell-Watkins	**Program ID:** DHLT
Program Test Name: Draw Horizontal Line Test	
Objective: To verify that specified inputs show on the grid accordingly	
Description: Preset inputs are used for the X1, Y1, X2, and Y2 values. Inputs for Y1 are equal to Y2 and X2 has a lesser value than X1. (Horizontal Line, slope = 0.)	
Conditions: Inputs are preset to integers and non-integer types. Two sub tests are performed for inputs. Code Inputs are shown in section 6a.	
Expected Results: Output will state "Layer added" if correct input is valid. Output will throw and state "NumberFormatException" if input is invalid.	

Actual Results:
`TestHLineXY: Layer added.`
`TestHLineXY1: java.lang.NumberFormatException: For input string: "-"`	

Draw Rectangle Test Results

Tester Name: Chris Hollis	**Date:** 4/12/08
Time Began: 18:35	**Time Ended:** 18:45
Instructor: Dr. Bell-Watkins	**Program ID:** DRT
Program Test Name: Draw Rectangle Test	
Objective: To verify that specified inputs show on the grid accordingly	
Description: Preset inputs are used for the X1, Y1, X2, and Y2 values.	
Conditions: Inputs are preset to integers and non-integer types. Four sub tests are performed for inputs. Code Inputs are shown in section 6a.	
Expected Results: Output will state "Layer added" if correct input is valid. Output will throw and state "NumberFormatException" if input is invalid.	

Actual Results:
`TestRectangle1: Layer added.`
`TestRectangle2: Layer added.`
`TestRectangle3: Layer added.`
`TestRectangle4: java.lang.NumberFormatException: For input string: "UGD@$%"`	

Draw Trigonometric Function Test

Tester Name: Joe Petrizzi	**Date:** 4/12/08
Time Began: 18:50	**Time Ended:** 18:55
Instructor: Dr. Bell-Watkins	**Program ID:** DTFT
Program Test Name: Draw Trigonometric Function Test	
Objective: To verify that specified inputs show on the grid accordingly	
Description: Preset inputs are used for the multiplier and argument fields.	
Conditions: Inputs are preset to integers and non-integer types. One sub test is used performed for inputs. Code Inputs are shown in section 6a.	
Expected Results: Output will state "Layer added" if correct input is valid. Output will throw and state "NumberFormatException" if input is invalid.	
Actual Results: `TestTrig: java.lang.NumberFormatException`	

XV. (a) Actual Code Input Values

```
public class UnitTest {

	UnitTest() { }


/** Reset Coordinates **/
	public void reset(GUIInputTools InputTools){
		InputTools.getDrawPointXTextField().setText("0");
		InputTools.getDrawPointYTextField().setText("0");
	}

/** Test 1a Draw Point X Functions **/
	public void Testx1(GUIInputTools InputTools){
		System.out.print("Testx1: ");
		InputTools.getDrawPointXTextField().setText("A");
		InputTools.getDrawPointButton().doClick();
	}
	public void Testx2(GUIInputTools InputTools){
		System.out.print("Testx2: ");
		InputTools.getDrawPointXTextField().setText("5");
		InputTools.getDrawPointButton().doClick();
	}
	public void Testx3(GUIInputTools InputTools){
		System.out.print("Testx3: ");
		InputTools.getDrawPointXTextField().setText("5.5");
		InputTools.getDrawPointButton().doClick();
	}
	public void Testx4(GUIInputTools InputTools){
		System.out.print("Testx4: ");
		InputTools.getDrawPointXTextField().setText(" ");
		InputTools.getDrawPointButton().doClick();
	}
	public void Testx5(GUIInputTools InputTools){
		System.out.print("Testx5: ");
		InputTools.getDrawPointXTextField().setText("51");
		InputTools.getDrawPointButton().doClick();
	}

/** Test 1b Draw Point Y Functions **/
	public void Testy1(GUIInputTools InputTools){
		System.out.print("Testy1: ");
		InputTools.getDrawPointYTextField().setText("B");
		InputTools.getDrawPointButton().doClick();
	}
	public void Testy2(GUIInputTools InputTools){
		System.out.print("Testy2: ");
		InputTools.getDrawPointYTextField().setText("6");
		InputTools.getDrawPointButton().doClick();
	}
	public void Testy3(GUIInputTools InputTools){
		System.out.print("Testy3: ");
		InputTools.getDrawPointYTextField().setText("2.3");
		InputTools.getDrawPointButton().doClick();
	}
	public void Testy4(GUIInputTools InputTools){
		System.out.print("Testy4: ");
		InputTools.getDrawPointYTextField().setText(" ");
		InputTools.getDrawPointButton().doClick();
	}
	public void Testy5(GUIInputTools InputTools){
		System.out.print("Testy5: ");
		InputTools.getDrawPointYTextField().setText("61");
```

```
            InputTools.getDrawPointButton().doClick();
      }


/** Test 2 Draw Line Functions **/
      public void TestLineXY(GUIInputTools InputTools){
            System.out.print("TestLineXY: ");
            InputTools.getDrawLineX1TextField().setText("2");
            InputTools.getDrawLineX2TextField().setText("2");
            InputTools.getDrawLineY1TextField().setText("2");
            InputTools.getDrawLineY2TextField().setText("2");
            InputTools.getDrawLineButton().doClick();
      }
      public void TestLineXY1(GUIInputTools InputTools){
            System.out.print("TestLineXY1: ");
            InputTools.getDrawLineX1TextField().setText("-1");
            InputTools.getDrawLineX2TextField().setText("8");
            InputTools.getDrawLineY1TextField().setText("4");
            InputTools.getDrawLineY2TextField().setText("50");
            InputTools.getDrawLineButton().doClick();
      }
      public void TestLineXY2(GUIInputTools InputTools){
            System.out.print("TestLineXY2: ");
            InputTools.getDrawLineX1TextField().setText("0");
            InputTools.getDrawLineX2TextField().setText("-31");
            InputTools.getDrawLineY1TextField().setText("7");
            InputTools.getDrawLineY2TextField().setText("45");
            InputTools.getDrawLineButton().doClick();
      }
      public void TestLineXY3(GUIInputTools InputTools){
            System.out.print("TestLineXY3: ");
            InputTools.getDrawLineX1TextField().setText("-0");
            InputTools.getDrawLineX2TextField().setText("$");
            InputTools.getDrawLineY1TextField().setText("60");
            InputTools.getDrawLineY2TextField().setText("0");
            InputTools.getDrawLineButton().doClick();
      }


/** Test 3 Draw Vertical and Horizontal Line Functions **/

// Vertical Line
      public void TestVLineXY(GUIInputTools InputTools){
            System.out.print("TestVLineXY: ");
            InputTools.getDrawLineX1TextField().setText("2");
            InputTools.getDrawLineX2TextField().setText("2");
            InputTools.getDrawLineY1TextField().setText("2");
            InputTools.getDrawLineY2TextField().setText("1");
            InputTools.getDrawLineButton().doClick();
      }
      public void TestVLineXY1(GUIInputTools InputTools){
            System.out.print("TestVLineXY1: ");
            InputTools.getDrawLineX1TextField().setText("6");
            InputTools.getDrawLineX2TextField().setText(" ");
            InputTools.getDrawLineY1TextField().setText("8");
            InputTools.getDrawLineY2TextField().setText("-53");
            InputTools.getDrawLineButton().doClick();
      }
      public void TestVLineXY2(GUIInputTools InputTools){
            System.out.print("TestVLineXY2: ");
            InputTools.getDrawLineX1TextField().setText("33");
            InputTools.getDrawLineX2TextField().setText("1");
            InputTools.getDrawLineY1TextField().setText("-76");
```

```
            InputTools.getDrawLineY2TextField().setText("0");
            InputTools.getDrawLineButton().doClick();
      }

// Horizontal Line
      public void TestHLineXY(GUIInputTools InputTools){
            System.out.print("TestHLineXY: ");
            InputTools.getDrawLineX1TextField().setText("2");
            InputTools.getDrawLineX2TextField().setText("1");
            InputTools.getDrawLineY1TextField().setText("2");
            InputTools.getDrawLineY2TextField().setText("2");
            InputTools.getDrawLineButton().doClick();
      }
      public void TestHLineXY1(GUIInputTools InputTools){
            System.out.print("TestHLineXY1: ");
            InputTools.getDrawLineX1TextField().setText("-");
            InputTools.getDrawLineX2TextField().setText("+");
            InputTools.getDrawLineY1TextField().setText("@");
            InputTools.getDrawLineY2TextField().setText("7");
            InputTools.getDrawLineButton().doClick();
      }

/** Test 4 Draw Rectangle Functions **/
      public void TestRectangle1(GUIInputTools InputTools){
            System.out.print("TestRectangle1: ");
            InputTools.getDrawRectangleX1TextField().setText("10");
            InputTools.getDrawRectangleX2TextField().setText("0");
            InputTools.getDrawRectangleY1TextField().setText("0");
            InputTools.getDrawRectangleY2TextField().setText("10");
            InputTools.getDrawRectangleButton().doClick();
      }
      public void TestRectangle2(GUIInputTools InputTools){
            System.out.print("TestRectangle2: ");
            InputTools.getDrawRectangleX1TextField().setText("0");
            InputTools.getDrawRectangleX2TextField().setText("10");
            InputTools.getDrawRectangleY1TextField().setText("10");
            InputTools.getDrawRectangleY2TextField().setText("0");
            InputTools.getDrawRectangleButton().doClick();
      }
      public void TestRectangle3(GUIInputTools InputTools){
            System.out.print("TestRectangle3: ");
            InputTools.getDrawRectangleX1TextField().setText("0");
            InputTools.getDrawRectangleX2TextField().setText("-10");
            InputTools.getDrawRectangleY1TextField().setText("0");
            InputTools.getDrawRectangleY2TextField().setText("10");
            InputTools.getDrawRectangleButton().doClick();
      }
      public void TestRectangle4(GUIInputTools InputTools){
            System.out.print("TestRectangle4: ");
            InputTools.getDrawRectangleX1TextField().setText("-0");
            InputTools.getDrawRectangleX2TextField().setText("-13");
            InputTools.getDrawRectangleY1TextField().setText("UGD@$%");
            InputTools.getDrawRectangleY2TextField().setText("0000000000000");
            InputTools.getDrawRectangleButton().doClick();
      }

/** Test 5 Draw Trig Functions **/
      public void TestTrigFunctions(GUIInputTools InputTools){
            System.out.print("TestTrig: ");
            InputTools.getTrigFunctionArgumentTextField().setText("8");
            InputTools.getTrigMultiplierTextField().setText("5");
            InputTools.getGraphTrigFunctionButton().doClick();
      }}
```

XV. (b) Actual Code Output from Tests:

```
Testx1: java.lang.NumberFormatException: For input string: "A"
Testx2: Layer added.
Testx3: java.lang.NumberFormatException: For input string: "5.5"
Testx4: java.lang.NumberFormatException: For input string: " "
Testx5: Layer added.
Testy1: java.lang.NumberFormatException: For input string: "B"
Testy2: Layer added.
Testy3: java.lang.NumberFormatException: For input string: "2.3"
Testy4: java.lang.NumberFormatException: For input string: " "
Testy5: Layer added.
TestLineXY: Layer added.
TestLineXY1: Layer added.
TestLineXY2: Layer added.
TestLineXY3: java.lang.NumberFormatException: For input string: "$"
TestVLineXY1: java.lang.NumberFormatException: For input string: " "
TestVLineXY: Layer added.
TestVLineXY2: Layer added.
TestHLineXY: Layer added.
TestHLineXY1: java.lang.NumberFormatException: For input string: "-"
TestRectangle1: Layer added.
TestRectangle2: Layer added.
TestRectangle3: Layer added.
TestRectangle4: java.lang.NumberFormatException: For input string: "UGD@$%"
TestTrig: java.lang.NumberFormatException
```

XV. (c) Methods Called for Testing Inputs

These methods were called in the controller (BeadLoom.java)

```
UT = new UnitTest();

//Draw Point X Testing
UT.Testx1(InputTools);
UT.Testx2(InputTools);
UT.Testx3(InputTools);
UT.Testx4(InputTools);
UT.Testx5(InputTools);

UT.reset(InputTools);

//Draw Point Y  Testing
UT.Testy1(InputTools);
UT.Testy2(InputTools);
UT.Testy3(InputTools);
UT.Testy4(InputTools);
UT.Testy5(InputTools);

UT.reset(InputTools);

//Draw Line Testing
UT.TestLineXY(InputTools);
UT.TestLineXY1(InputTools);
UT.TestLineXY2(InputTools);
UT.TestLineXY3(InputTools);
```

```
//Draw Vertical Line Testing
UT.TestVLineXY1(InputTools);
UT.TestVLineXY(InputTools);
UT.TestVLineXY2(InputTools);

//Draw Horizontal Line Testing
UT.TestHLineXY(InputTools);
UT.TestHLineXY1(InputTools);

//Draw Rectangle Testing
UT.TestRectangle1(InputTools);
UT.TestRectangle2(InputTools);
UT.TestRectangle3(InputTools);
UT.TestRectangle4(InputTools);


//Draw Trig Functions
UT.TestTrigFunctions(InputTools);
```

XVI. System Test Suite

Due to time constraints, the following system test suite is designed only to test the functionality of the Draw Point feature contained in GUIInputTools.

Assumptions for all tests:

2. Program is running.
3. Grid is cleared of all beads and layers.
4. Default grid size when executing program is 30x30.
5. Text fields are designed to handle zeroes, positive integers, and negative integers.
6. Maximum grid size is 100x100. Due to this constraint, the maximum numerical input for the text fields is 50, and the minimum numerical input is -50.

Test Group DP01:

These tests are for designed use of the Draw Point feature where one or more of the coordinates in the point is zero. The corresponding X and Y values are to be entered into the Draw Point text fields, and the Draw Point button is then pressed. Repeat this process for each coordinate pair.

Let T# be the number of tests performed for this type of test.
Let X be the X coordinate to be plotted.
Let Y be the Y coordinate to be plotted.
Pass: Expected outputs show on grid at the specified coordinates with respect to the coordinate axes.
Fail: X and Y values do not show correctly on grid coordinates.

Input	Expected Outcome
T1: X = 0, Y = 0	A bead would be plotted on coordinates (0,0).
T2: X = 5, Y = 0	A bead would be plotted on coordinates (5,0).
T3: X = -5, Y = 0	A bead would be plotted on coordinates (-5,0).
T4: X = 0, Y = 5	A bead would be plotted on coordinates (0,5).
T5: X = 0, Y = -5	A bead would be plotted on coordinates (0,-5).

Test Group DP02:

These tests are for designed use of the Draw Point feature where neither of the coordinates in the point is zero. This test group is also designed to test the accuracy of the quadrant system. The corresponding X and Y values are to be entered into the Draw Point text fields, and the Draw Point button is then pressed. Repeat this process for each coordinate pair.

Let T# be the number of tests performed for this type of test.
Let X be the X coordinate to be plotted.
Let Y be the Y coordinate to be plotted.
Pass: Expected outputs show on grid at the specified coordinates in each quadrant.
Fail: X and Y values do not show correctly on grid coordinates.

Input	**Expected Outcome**
T1: X = 5, Y = 5	A bead would be plotted on coordinates (5,5).
T2: X = -5, Y = 5	A bead would be plotted on coordinates (-5,5).
T3: X = -5, Y = 5	A bead would be plotted on coordinates (-5,5).
T4: X = -5, Y = -5	A bead would be plotted on coordinates (-5,-5).

Test Group DP03:

These tests are for designed use of the Draw Point feature where one or more of the coordinates in the point is outside the visible bounds of the default grid, but still should be plotted. The corresponding X and Y values are to be entered into the Draw Point text fields, and the Draw Point button is then pressed. At the end of each test, resize the grid to maximum (100x100) to see if the point has been correctly plotted, then return the grid size to default (30x30). Repeat this process for each coordinate pair.

Let T# be the number of tests performed for this type of test.
Lets X be the X coordinate to be plotted, Let Y be the Y coordinate to be plotted.
Pass: Expected outputs show on grid if grid size is higher than default grid size.
Fail: X and Y values do not show on grid.

Input	**Expected Outcome**

T1: X = -31, Y = 0	A bead would be plotted on coordinates (-31,0).
T2: X = 0, Y = 31	A bead would be plotted on coordinates (0,-31).
T3: X = -31, Y = -31	A bead would be plotted on coordinates (-31,-31).
T4: X = 31, Y = 31	A bead would be plotted on coordinates (-5,-5).
T5: X = -50, Y = 50	A bead would be plotted on coordinate (-50,50).
T6: X = 50, Y = -50	A bead would be plotted on coordinate (50,-50).
T7: X = 50, Y = 50	A bead would be plotted on coordinate (50,50).
T8: X = -50, Y = -50	A bead would be plotted on coordinate (-50,-50).

Test Group DP04:

These tests are for testing the Draw Point function for input values that are outside of bounds. The corresponding X and Y values are to be entered into the Draw Point text fields, and the Draw Point button is then pressed. Repeat this process for each coordinate pair.

Input	**Expected Outcome**
T1: X = 51; Y = 0	Program throws an exception that the coordinate exceeds the grid boundary.
T2: X = -51; Y = 0	Program throws an exception that the coordinate exceeds the grid boundary.
T3: X = 0; Y = 51	Program throws an exception that the coordinate exceeds the grid boundary.
T4: X = 0; Y = -51	Program throws an exception that the coordinate exceeds the grid boundary.
T5: X = 51; Y = 51	Program throws an exception that the coordinate exceeds the grid boundary.
T6: X = -51; Y = -51	Program throws an exception that the coordinate exceeds the grid boundary.
T7: X = 51; Y = -51	Program throws an exception that the coordinate exceeds the grid boundary.
T8: X = -51; Y = 51	Program throws an exception that the coordinate exceeds the grid boundary.

Test Group DP05:

These tests are for testing the Draw Point function for input values that are non-integers. The corresponding X and Y values are to be entered into the Draw Point text fields, and the Draw Point button is then pressed. Repeat this process for each coordinate pair.

T1: X = -0; Y = 7	Program operates normally, plots point at coordinate (0,7) and treats "-0" as "0".
T2: X = 7; Y = -0	Program operates normally, plots point at coordinate (0,7) and treats "-0" as "0".
T3: X = -0; Y = -0	Program operates normally, plots point at coordinate (0,7) and treats "-0" as "0".
T4: X = a; Y = 0	Program throws an exception that the coordinate must be an integer.
T5: X = *; Y = 0	Program throws an exception that the coordinate must be an integer.

T6: X is empty ; Y = 0	Program throws an exception that the coordinate must be an integer.
T7: X = 0; Y = 3.4	Program throws an exception that the coordinate must be an integer.
T8: X = 2; Y = 00000	Program operates normally, plots point at coordinate (2,0) and treats “00000” as “0”.
T9: X = (space)3; Y = 3	Program throws an exception that the coordinate must be an integer.
T10: X = 3; Y = 3 5	Program throws an exception that the coordinate must be an integer.

Name: Harris Christopher Hollis
Michael Lodge
Chris Blake
Phillip Gipson
Joe Petrizzi

Georgia Southern University
Department of Computer Sciences

CSCI5530 Software Engineering
Spring 2008

Assignment: User's Manual for Virtual Bead Loom v2.5

Date: 16-Apr-08

People who provided assistance/affiliation: **Dr. Kera Bell-Watkins, Ron Englash, Group Members**

References:
Virtual Bead Loom software, Microsoft Paint

Table of Contents

6. Using Color
    1. How to change the color of beads

7. Moving Beads and the Layers Tool
    1. Creating a layer
    2. Selecting a layer
    3. Moving bead objects
    4. Deleting a layer
    5. Clearing the grid

8. The Menu Toolbar
    1. File Menu
        1. Print
    2. Edit Menu
        1. Undo
        2. Redo
    3. Options Menu
        1. Grid Option Menu
            1. Grid Size
            2. XY Follow Mouse
            3. Hide Grid
        2. Examples Menu
    4. Windows Menu
        1. Grid
        2. Goal Images
        3. Output Window
        4. Move Beads Window
        5. Bead Utilities Window
    5. Help Menu
        1. About

9. The Code Output Window
    1. Code output menu toolbar
    2. Code window

Appendices:

Appendix A: Known bugs and workarounds

Appendix B: Incomplete features/Features to be completed in later versions

1. Introduction to Virtual Bead Loom

Thank you for using the Virtual Bead Loom software. This is the first of a series of programs collectively grouped as Culturally-Situated Design Tools (CSDTs). This software is designed for students of grades 6 through 12 to introduce concepts of graphing, user interfaces, and programming. Using this software, the user can load images and replicate them upon the digital bead loom, or grid. This manual will give step-by-step instructions on how to use the software and an idea of what is still to come in further development.

1. System Requirements

The Virtual Bead Loom is designed to be run as an embedded Java Applet within a standard web page. To run the Virtual Bead Loom, the user's computer must have:

A Java enabled browser using Java 5.0 or higher.

Windows XP (Service Pack 2 or better)/Windows Vista/Mac OS X 10.3 or better

The user's computer is suggested to use Windows XP (Service Pack 2) or Windows Vista. There may be minor bugs in the program when run in the Mac environment (see Appendix A for a listing of these bugs).

2. The Virtual Bead Loom Interface

The Virtual Bead Loom Interface can be broken down into five distinct parts: The Grid window, the Goal Images window, the Bead Utilities window, the Adjust Beads window, and the Code Output window. In the screen shot below, these windows are numbered one through five, corresponding with the section in this chapter.

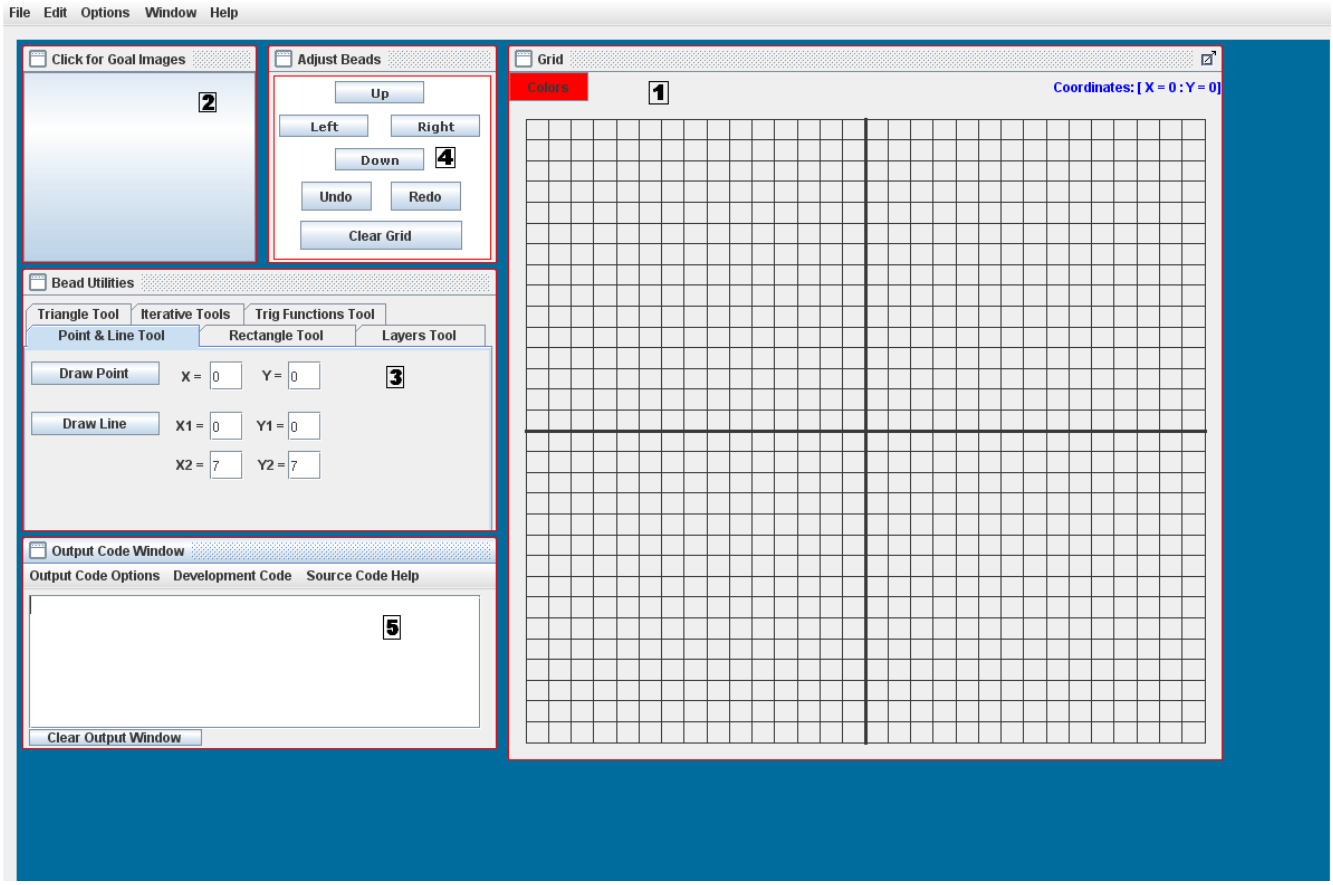


Figure 1. The Virtual Bead Loom Interface. The entire user interface with all windows visible at startup.

1. The Grid Window

The Grid window is located on the right hand side of the screen, and it is where all of the beads created in the Virtual Bead Loom software will be displayed. The grid itself is divided into four quadrants, with the x- and y-axes drawn as heavier lines to distinguish themselves. These quadrants reflect the Cartesian plane used in mathematics. The lines of the grid themselves may be turned on or off (see Chapter Eight, Using the Menu Toolbar). The axes and grid coordinates along the side should be enabled by default, but these features have not been programmed into the Virtual Bead Loom yet. The colors button for changing the colors of beads to be drawn is located in the top left corner of this window. Hovering the mouse over a point on the grid will show the coordinates above the grid.

2. The Goal Images Window

The Goal Images window is located in the upper left-hand corner of the screen. This window is used to show a goal image of what is to be recreated on the grid window.

3. The Bead Utilities Window

The Bead Utilities window is located in the center of the left-side side of the screen, beneath the Goal Images window and above the Code Output window. It is composed of several tabbed panels which can be used to draw beads on the grid and help to cycle through layers of created beads.

4. The Adjust Beads Window

The Adjust Beads window is located at the center of the top of the screen, to the left of the Goal Images Window and to the right of the Grid Window. This window allows the movement of currently selected bead objects, clearing of the grid, and the ability to undo or redo actions.

5. The Code Output Window

The Code Output window is located in the bottom left-hand corner of the screen. This window displays pseudo-code that is generated by the Virtual Bead Loom for specific actions within the program. This window is also not fully functional and is included primarily as room for expansion within the Virtual Bead Loom program for future developers.

3. Getting Started

This chapter is designed to help the user become accustomed to the Virtual Bead Loom system, including creating new files, opening saved files, saving files, and exiting the program.

1. Creating a new bead loom

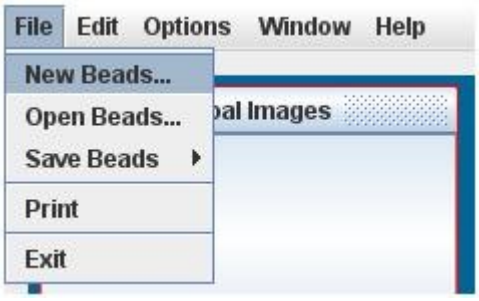


Figure 2. Creating a new bead loom.

The Virtual Bead Loom program opens with a new bead loom ready. However, it is possible to create a new bead loom file if so desired. In the menu bar at the top of the screen, left-click File -> New Beads. A dialog box will appear, asking if the user wishes to save the current bead loom. [screenshot of dialog box] If the user wishes to save, select yes and follow the directions as listed in

section three of this chapter, Saving a Bead Loom. Otherwise, select no. A new, clear bead loom will appear.

2. Opening a saved bead loom

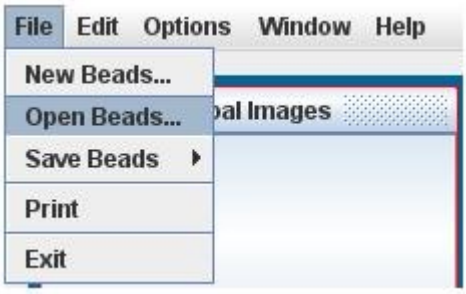


Figure 3. Opening a saved bead loom.

To open a previously saved bead loom file and load it into the Virtual Bead Loom interface, select File -> Open Beads from the menu bar at the top of the screen. A file selection box will pop up on the screen.

Figure 4. The Open Beads dialog box.

Choose the bead loom file to be loaded (note: it must be in .xml format, as this is the only file type that will load into the Virtual Bead Loom), and select OK. The bead loom that was selected will appear in the interface.

3. Saving a bead loom

Figure 5. Saving a bead loom.

To save the current bead loom for later use, select File -> Save Beads. At this point, there will be two options visible. The first, Save to XML, will save the current bead loom in .xml form to be loaded into the Virtual Bead Loom at a later time. Selecting this option will pop up a file saving dialog box.

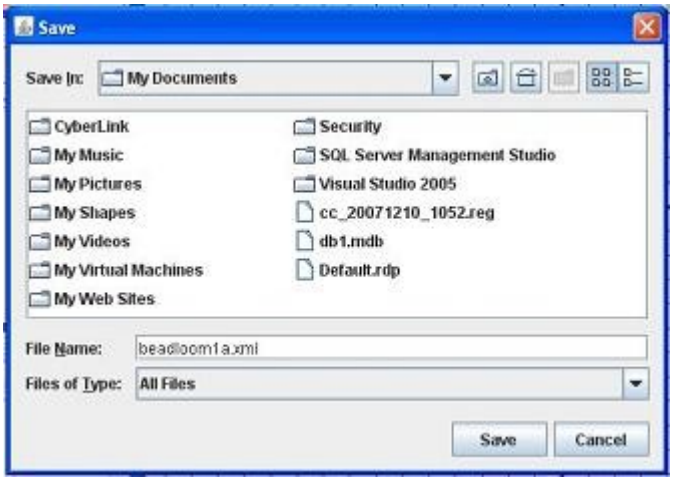


Figure 6. Saving to file in XML format.

Enter the name for the file, choose the folder to be saved into, and select Save. For example, typing "beadloom1a" will save the current bead loom as beadloom1a.xml.

The second option presented in the save menu is Save to JPEG. This option will take a snapshot of the current Grid Window and save it as an image file in .jpeg format. Selecting this option will pop up a file saving dialog box.

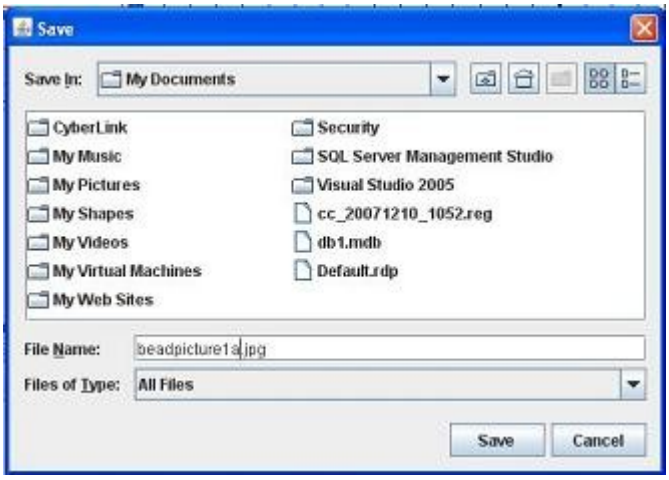


Figure 7. Saving to file in JPEG format.

Enter the name for the file, choose the folder to be saved into, and select Save. For example, typing "beadpicture1a" will save the current bead loom as beadpicture1a.jpeg. Pictures of the Grid Window saved in this manner can not be opened by the Virtual Bead Loom program. If the user wishes to do further work on the saved project, remember to use the Save to XML option.

4. Exiting the program

Figure 8. Exiting the Virtual Bead Loom.

To exit the Virtual Bead Loom program, select File -> Exit. This will close the program. Note: In the current version, this option may cause your browser to unexpectedly quit. It is safer to exit the program by using the browser's exit command.

1. Loading an image

To load an image into the Goal Images Window, click on the center of the window. The entire window acts as a load button for this purpose. A file selection box similar to the Open Beads dialog will pop up on the screen. The file types that may be loaded with this feature are .xml, .jpg, and .bmp. Navigate to the folder where the image to be loaded is stored, select the file, and click OK. The image chosen will appear in the Goal Images Window.

Figure 9. A loaded goal image in the goal images window.

5. Drawing Beads on the Grid

1. Drawing a point

To draw a point on the grid, select the Point and Line Tool tab in the Bead Utilities Window.

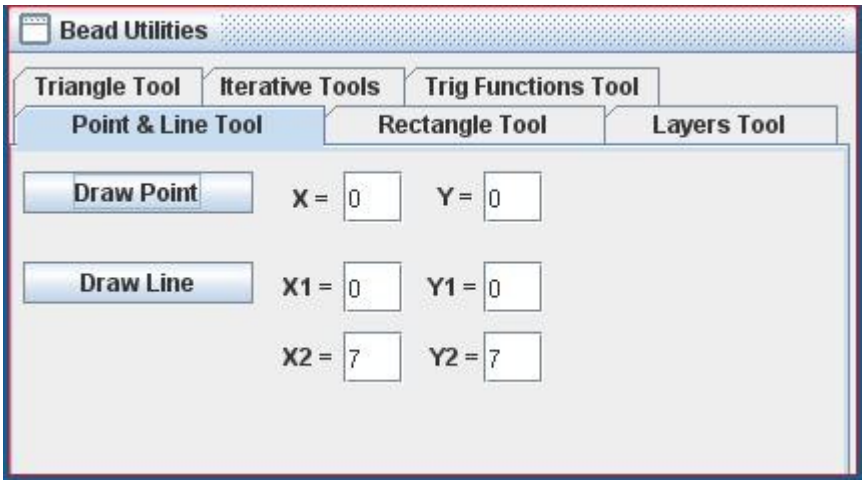


Figure 10. The Point and Line Tool.

Type the x- and y-coordinates of the point to be drawn, and click the Draw Point button. A single

bead will be drawn at the specified coordinate.

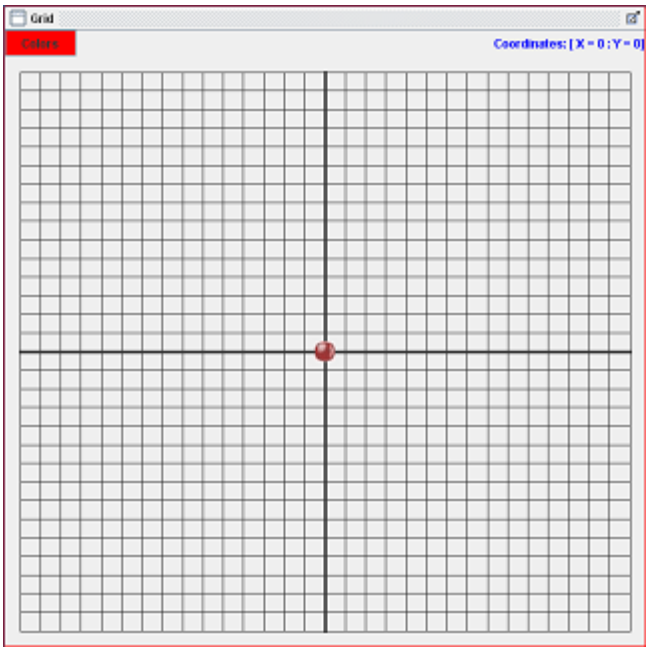


Figure 11. Drawing a point on the grid at (0,0).

2. Drawing a line

To draw a line on the grid, select the Point and Line Tool tab in the Bead Utilities Window. Type the x- and y-coordinates of the starting point in the x1 and y1 boxes, then type the coordinates of the ending point in the x2 and y2 boxes and click the Draw Line button. A line of beads will be drawn from the first point indicated to the last point.

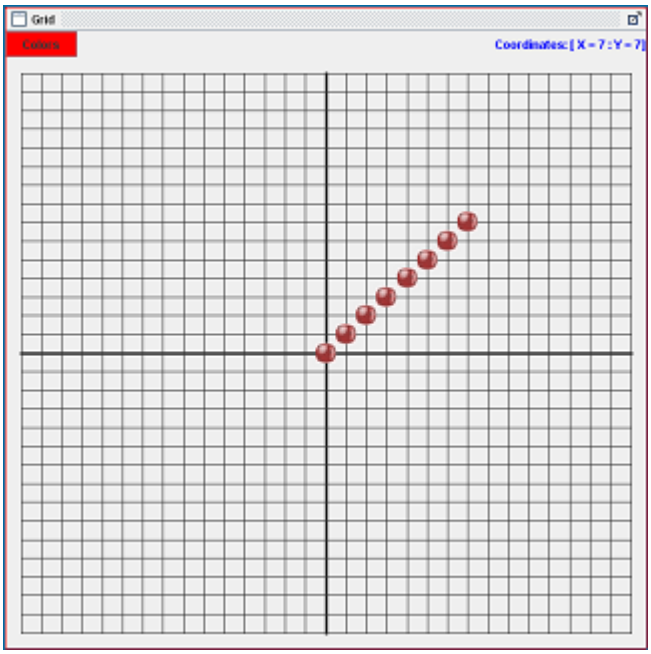


Figure 12. Drawing a line on the grid with endpoints (0,0) and (7,7).

3. Drawing a rectangle

To draw a rectangle on the grid, select the Rectangle Tool tab in the Bead Utilities Window.

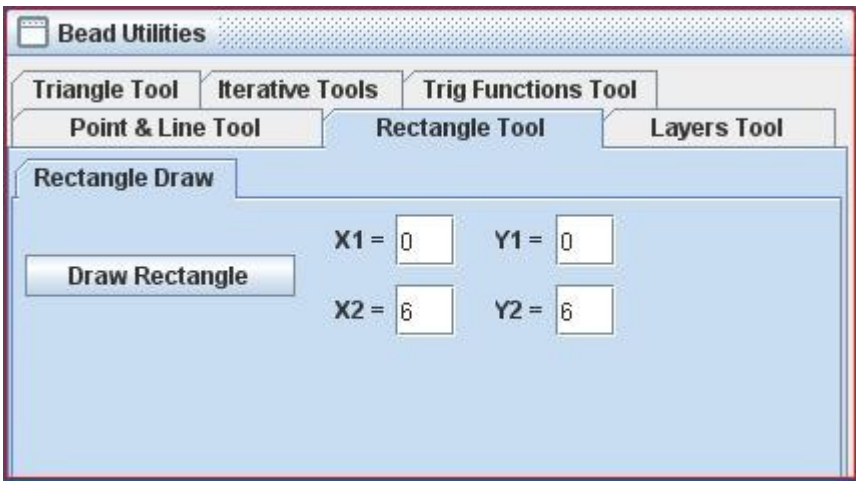


Figure 13. The Rectangle Tool.

Type the x- and y-coordinates of the first corner point in the x1 and y1 boxes, then type the coordinates of the opposite corner in the x2 and y2 boxes and click the Draw Rectangle button. A rectangle of beads will be drawn using the two points given as the endpoints for the diagonal of the rectangle.

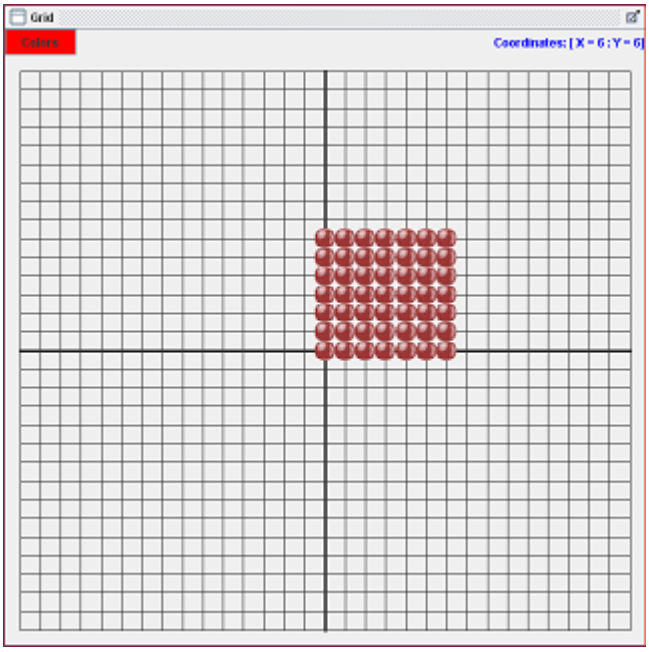


Figure 14. Drawing a rectangle with diagonal endpoints at (0,0) and (6,6).

4. Drawing a triangle

To draw a triangle on the grid, select the Triangle Tool tab in the Bead Utilities Window.

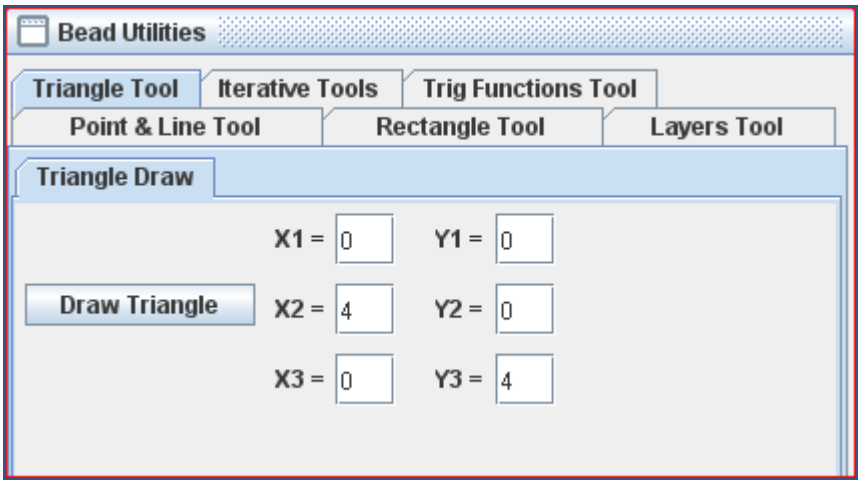


Figure 15. The Triangle Tool.

Type the x- and y-coordinates of the first point of the triangle in the x1 and y1 boxes, then the coordinates of the second point in the x2 and y2 boxes, the coordinates of the third point in the x3 and y3 boxes and click the Draw Triangle button. A triangle of beads will be drawn between the entered points and filled.

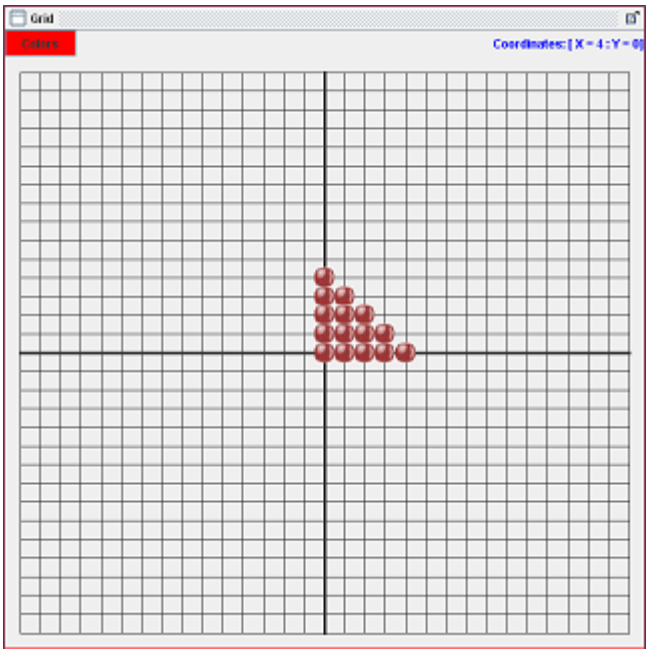


Figure 16. Drawing a triangle with vertices (0,0), (0,4), and (4,0).

5. Drawing a series of lines as a single object

To draw a series of lines as a single object on the grid, select the Iteration Tool tab in the Bead Utilities Window, then select the Line Iteration tab.

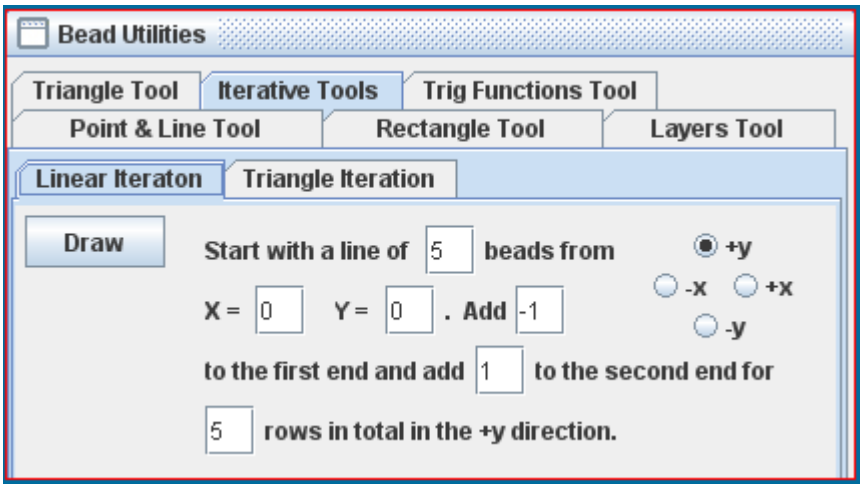


Figure 17. The Linear Iteration Tool.

There are several fields to fill in during the use of this tool. The first field, following "Start with a line...", indicates the length of the initial line. The next two fields, following "beads from", indicates the starting point of this line. The next field, following "Add", indicates the addition (or in the case of a negative number, subtraction) of beads to the starting side of the line to be iterated. The next field, following "to the first end and...", indicates the addition (or in the case of a negative number, subtraction) of beads to the ending side of the line to be iterated. The final field, following "to the second end", indicates the number of iterations the line is to go through. Finally, there is a series of four selection buttons to the right of the window, labeled "+x", "-x", "+y", and "-y". The selected button indicates the direction of iteration. Pressing the Draw button will create your line iteration. Three examples follow to help understand the complexity of the iteration tool.

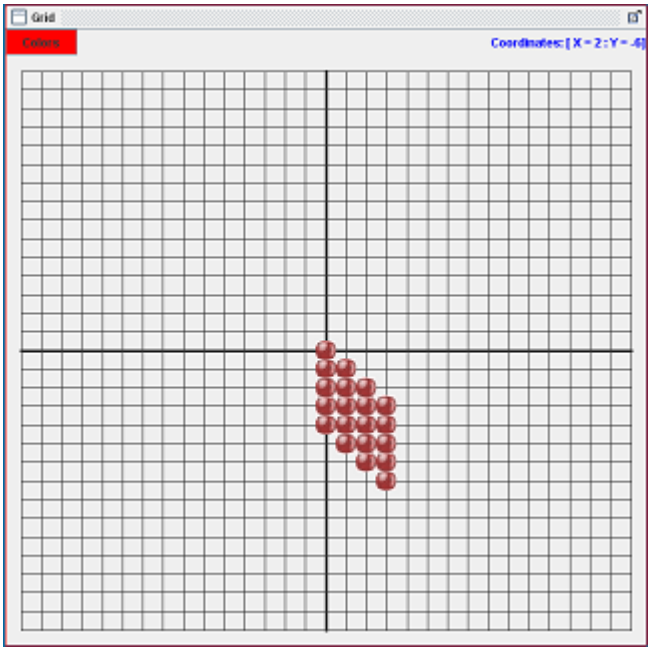


Figure 18. A linear iteration with 5 beads, starting at (0,0), -1 added to front end, +1 added to back end, through 4 iterations, drawn in the +x direction.

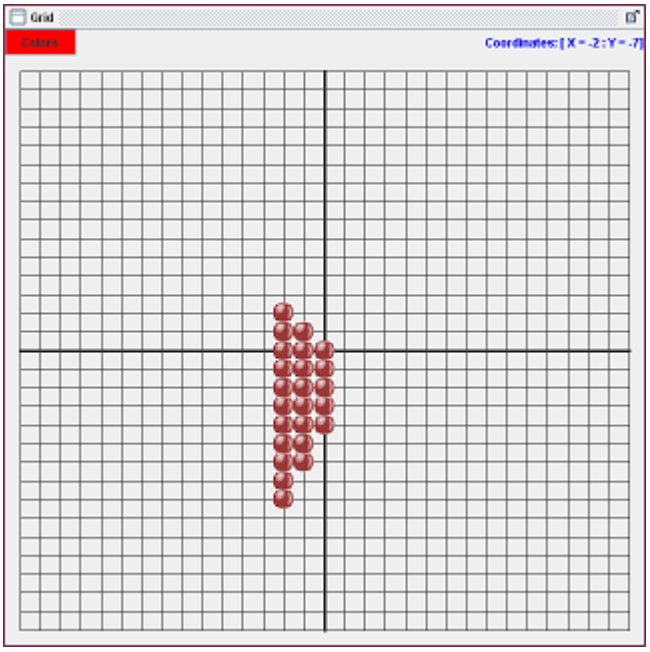


Figure 19. A linear iteration with 5 beads, starting at (0,0), +1 added to front end, +2 added to back end, through 3 iterations, drawn in the –x direction.

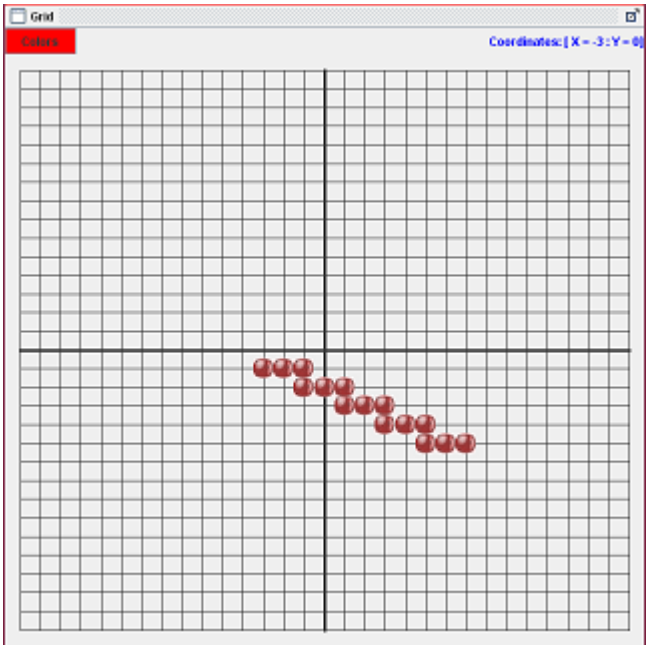


Figure 20. A linear iteration with 3 beads, starting at (5,-5), +2 added to front end, -2 added to back end, through 5 iterations, drawn in the +y direction.

6. Drawing a staggered triangle

To draw a staggered triangle, select the Iteration Tool in the Bead Utilities Window, then select the Triangle Iteration tab.

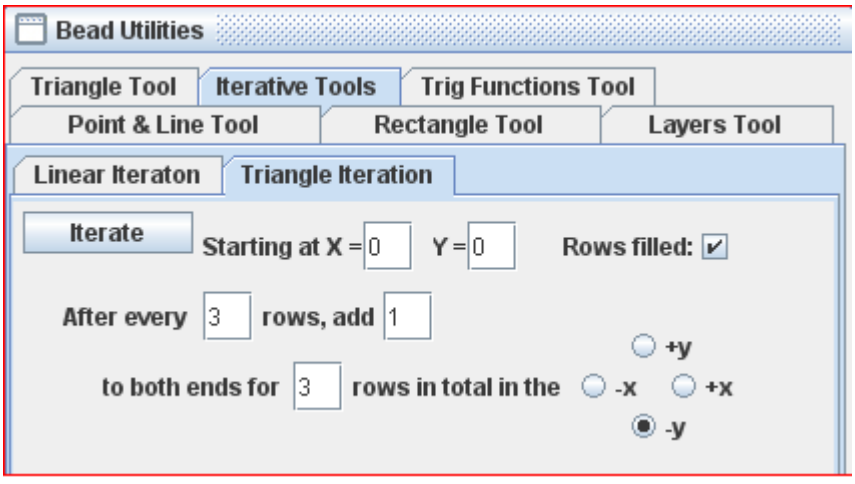


Figure 21. The Triangle Iteration Tool.

As in the case with line iterations above, the triangle iteration tool also has several fields to enter. The first two fields “Start X” and “Start Y”, indicate the starting coordinate. The rows filled check box allows the triangle to be spaced apart when unchecked and filled in when checked. The next field sets the number of rows of the triangle at the same length. The “add” field sets how many

beads are added on each side to widen the triangle at each change in length. The final field sets how many length increases of the triangle there are. Finally, as in linear iteration, there are four directions the triangle can be drawn in. Two examples to help guide the user:

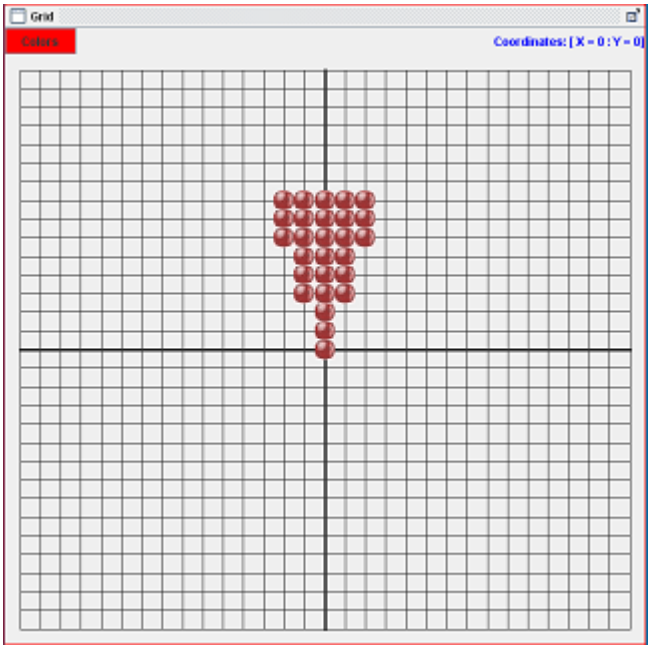


Figure 22. A triangle iteration with starting point (0,0), after every 3 rows, adding 1 to each side, iterating 3 times in the +y direction, filled box checked.

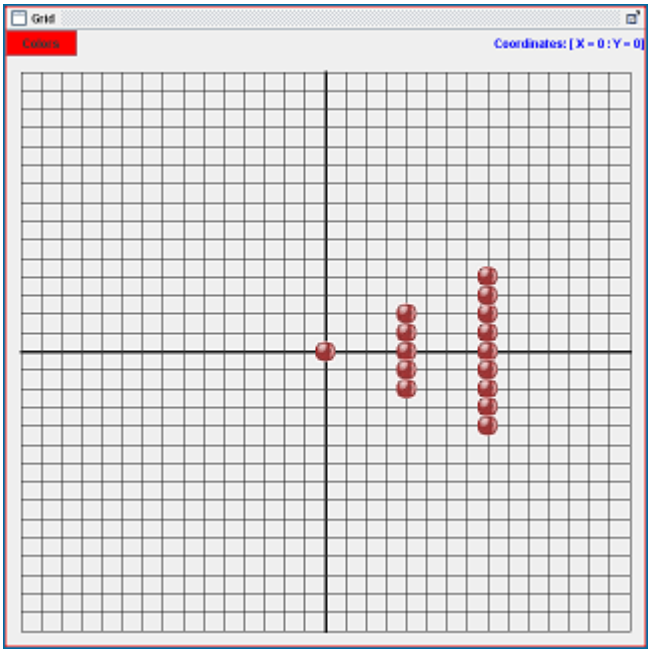


Figure 23. A triangle iteration with starting point (0,0), after every 4 rows, adding 2 to each side, iterating 3 times in the +x direction, filled box unchecked.

7. Drawing a trigonometric function

To draw a trigonometric function, select the Trig Function Tool in the Bead Utilities Window.

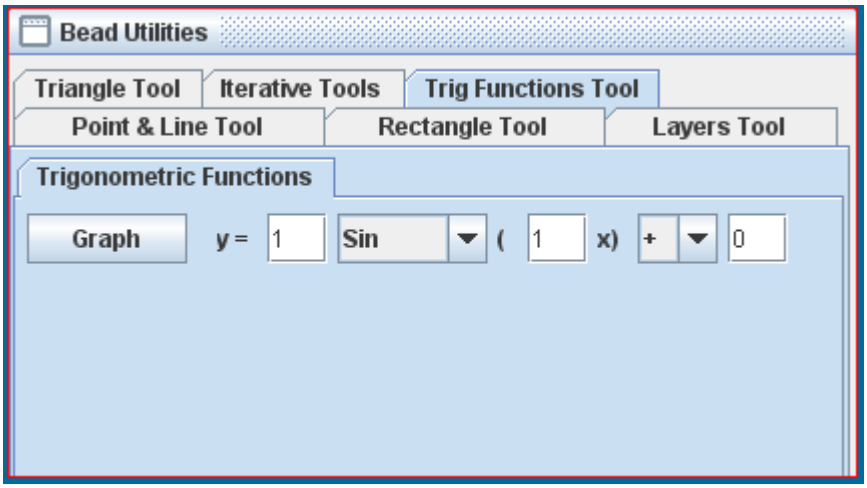


Figure 24. The Trigonometric Functions Tool.

Trig functions are dealt with in the Virtual Bead Loom as normal mathematical functions and are expressed in the form "y=Csin(Dx) +/- E". The fields in this tool directly correspond to this formula. The first field allows the user to set the value for C in the previous expression. The next field is a drop-down menu, allowing the user to select sine, cosine, tangent, arc sine, arc cosine, or arc tangent. The next field allows the user to set the value for D in the previous expression. The next field is a drop-down menu, allowing the user to select plus or minus. The final field allows the user to set the value for E in the previous expression. Finally, hitting the Graph button will draw the function to the grid.

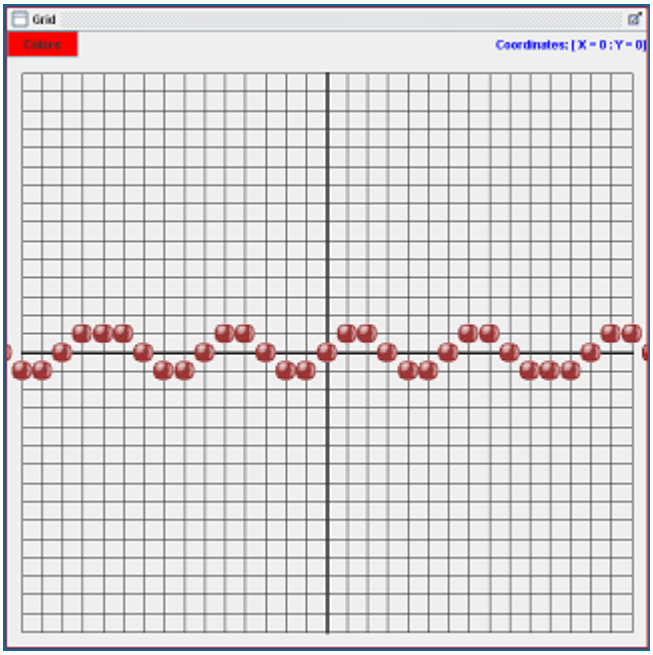


Figure 25. A trig function drawn: y=1sin(1x)+0.

6. Using Color

1. How to change the color of beads

The button to change the color of beads is located at the top of the Grid window and is labeled "Colors". Currently, the user can only change the color of beads that are about to be drawn onto the loom, and can not change the color of already drawn beads. Pressing the Colors button will bring up a color selection box.

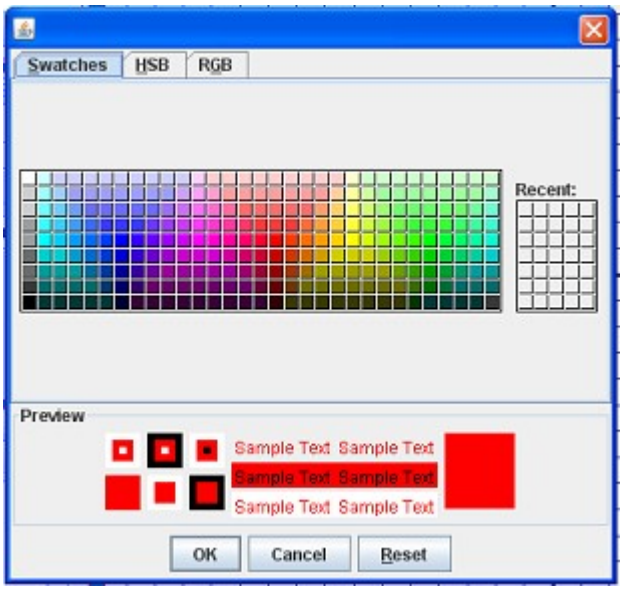


Figure 26. The color selection dialog box.

Select the desired color and click OK. The next beads drawn to the grid will show up in that color.

7. Moving Beads and the Layers Tool

Layers are an integral part of the Virtual Bead Loom program. Each separately drawn bead object is placed on a separate layer. This layer system allows the user to manipulate each object individually, without having to undo all progress up to that point if a mistake has been made, etc. This chapter is designed to help the user understand the layer system within the program.

1. Creating a layer

The user does not have to manually create layers each time he or she wishes to create a new bead object. Every time a new bead object is drawn to the grid, it is automatically placed on a new layer by the system. This layer will appear in the Layers Tool tab in the drop-down menu.

2. Selecting a layer

To select a layer, select the Layers Tool tab in the Bead Utilities Window.

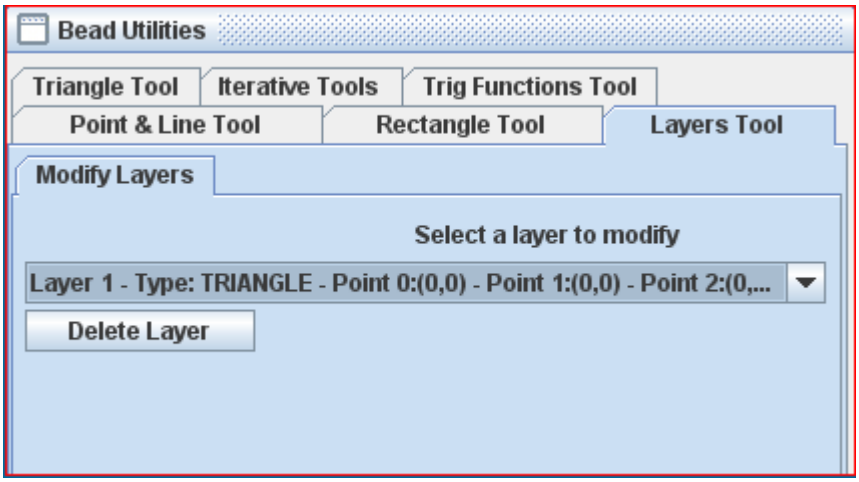


Figure 27. The Layers Tool.

In the window, there is a drop-down menu. Clicking it will reveal a list of all the layers representing bead objects on the grid. Deleted or undone layers will not appear in this list. Click the layer that is to be modified, and it will be selected. The details of the layer's information will be displayed in the field inside the drop-down menu.

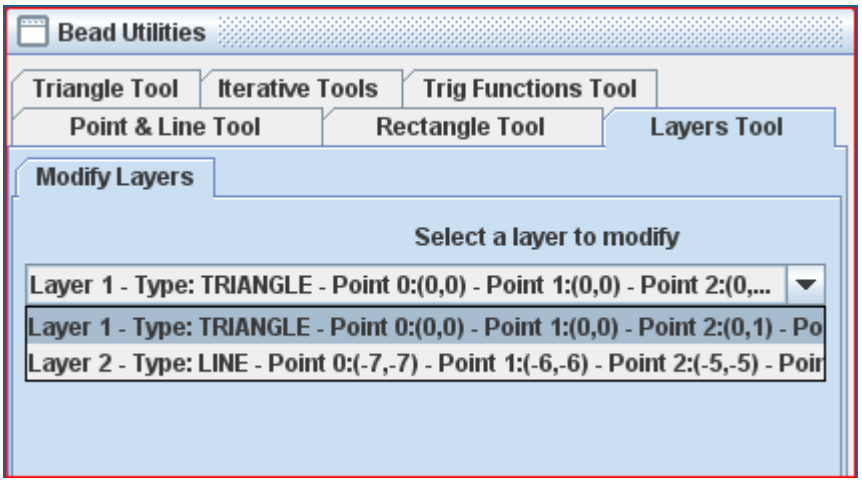


Figure 28. Selecting a layer from the drop down menu.

3. Moving bead objects

To move a bead object, first select the layer that it is contained within. Once it has been selected, it can be moved up, down, left, or right by the appropriate buttons contained within the Adjust Beads Window.

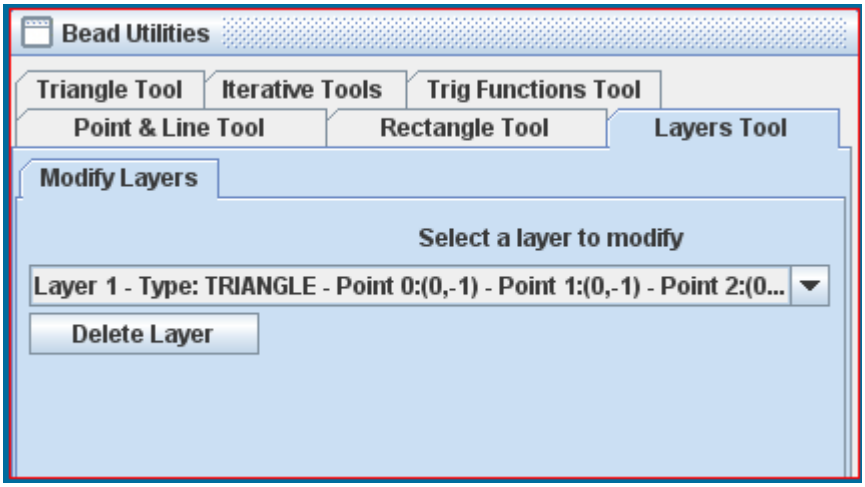


Figure 29. Moving a bead object. The layer information shows that the beads have been moved 1 unit down.

4. Deleting a layer

To delete a single bead object, the layer it is on must be deleted. To delete the layer, first select the layer to be deleted from the drop-down menu. Then, click the delete layer button. The selected layer will be removed from the drop-down menu and the bead object it contained will be removed from the grid.

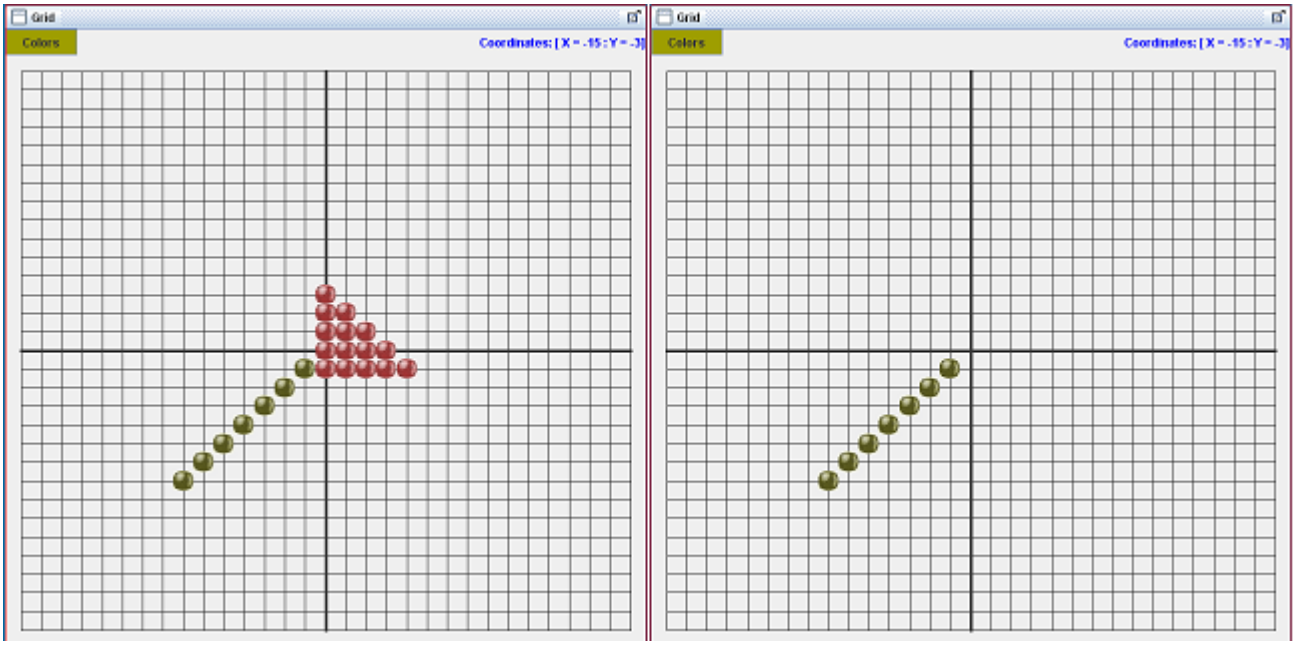


Figure 30. The bead grid, before and after a layer is deleted.

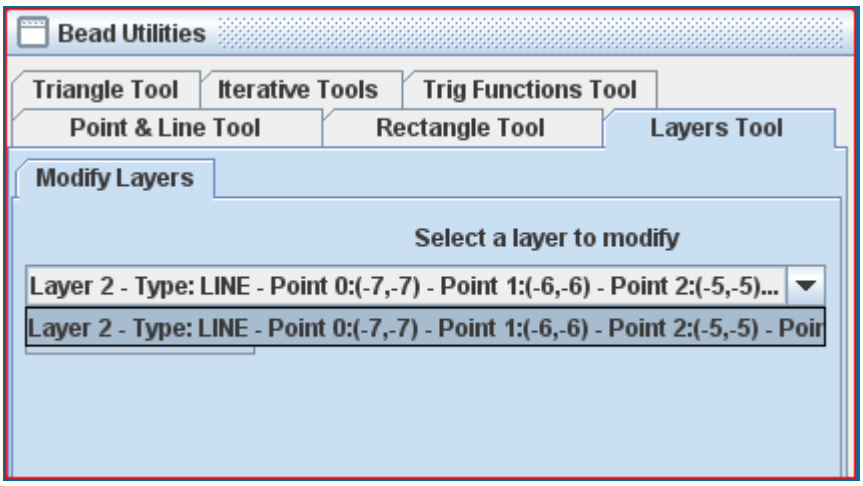


Figure 31. The layer drop down menu after layer 1 has been deleted.

5. Clearing the grid

To clear the grid of all layers and bead objects, click the button named "Clear Grid", located in the Adjust Beads Window. A dialog box will pop up to confirm. This action will remove all objects from the grid and all layers from the layer list.

8. The Menu Toolbar

1. File Menu

The File menu contains the following options: New Bead Loom, Open Bead Loom, Save Bead Loom, Print, and Exit. All options in this menu except for Print have already been described in Chapter Three, Getting Started.

1. Print

To print a screen shot of the grid, select File -> Print.

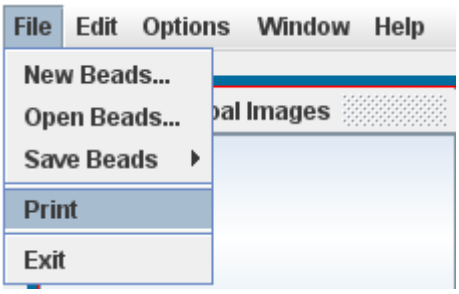


Figure 32. The Print feature.

A printer dialog box will pop up. Select OK to print the grid.

2. Edit Menu

The Edit menu contains the commands to undo and redo. It is located in the picture below.

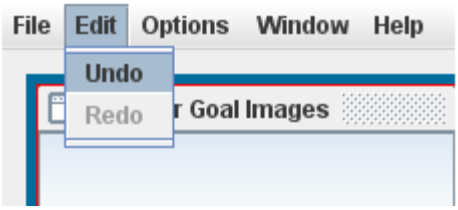


Figure 33. The Edit Menu.

1. Undo

The undo feature allows the user to undo the last creation or deletion of beads to the grid. This feature does not allow the undoing of Adjust Beads; any bead movement between the last action and the action previous to it will be undone along with the last action. To use the undo feature, select Edit -> Undo from the menu bar or choose the undo button from the Adjust Beads window. The undo feature is not selectable if there have been no actions performed on the grid (i.e. A new grid is created).

2. Redo

The redo feature allows the user to redo the last action that was previously undone. To use the redo feature, select Edit -> Redo from the menu bar or choose the redo button from the Adjust Beads window. The redo feature is not selectable if nothing has been undone, or if there has been a creation or deletion action performed since the last undo.

3. <u>Options Menu</u>

1. Grid Options Menu

There are a number of options that are listed in the Grid Options Menu. Only three of them are currently enabled in this version of the Virtual Bead Loom: Grid Size, XY Follow Mouse, and Hide Grid. Their functions are described below.

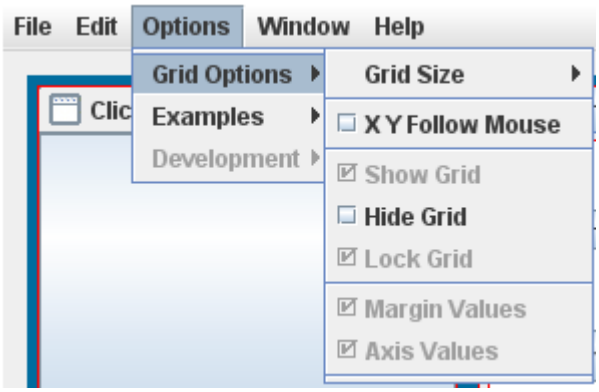


Figure 34. The Grid Options Menu.

1. Grid Size

To change the size of the grid, select Options -> Grid Options -> Grid Size from the menu bar. There will be a list of grid sizes in this menu that can be chosen, from 30x30 up to 100x100.

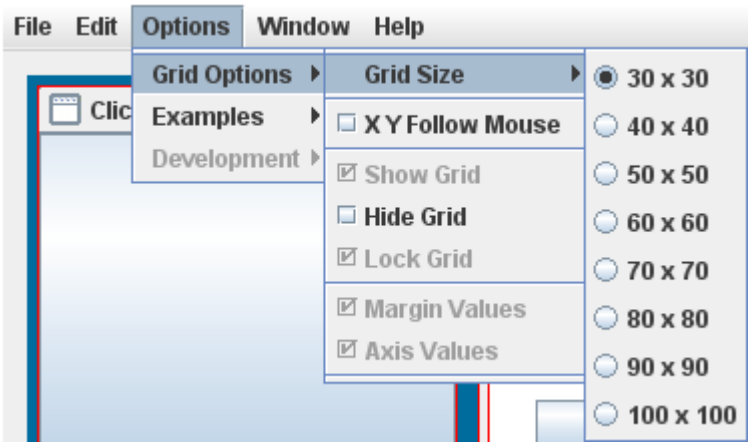


Figure 35. The Grid Size Menu.

Selecting a new grid size will change the grid's dimensions and resize the beads that both are currently on the grid and any beads that will be later drawn to the grid.

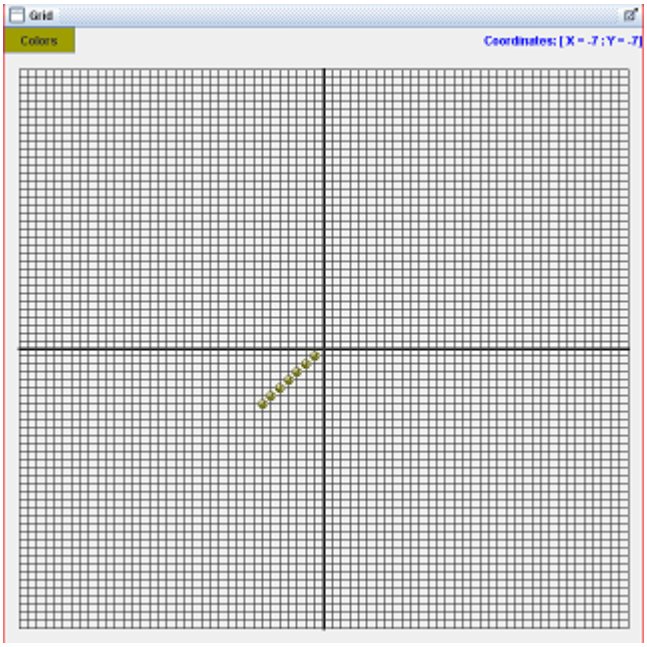


Figure 36. The grid resized to 70x70.

2. XY Follow Mouse

The XY Follow Mouse feature enables the ability to have the coordinates of the grid follow the mouse pointer in addition to being displayed in its normal fashion at the top of the grid. To enable this feature, select Options -> Grid Options -> XY Follow Mouse from the tool bar. To disable this feature, repeat this action.

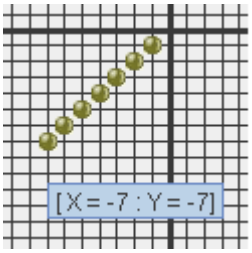


Figure 37. XY Follow Mouse feature enabled.

3. Hide Grid

The Hide Grid feature will remove the coordinate grid within the grid window while keeping the beads in the same locations. This feature is ideal for a clearer image to save as a .jpeg file or for printing. To hide the grid, select Options -> Grid Options -> Hide Grid. To disable this feature, repeat this process.

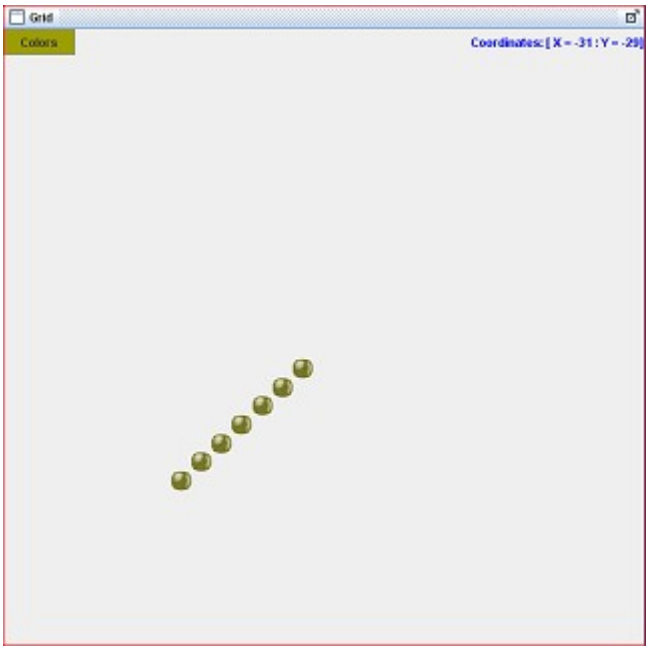


Figure 38. The grid after Hide Grid is enabled.

2. Examples Menu

The examples menu will contain sample bead looms that correspond with creation features within the program or sample goal images. These examples have not yet been added to the Virtual Bead Loom.

4. Window Menu

The Windows menu allows the user to hide or show each window within the Virtual Bead Loom.

1. Grid

To toggle the visibility of the grid window, select Window -> Grid from the menu bar.

2. Goal Images

To toggle the visibility of the Goal Images window, select Window -> Goal Images from the menu bar.

3. Output Window

To toggle the visibility of the output window window, select Window -> Output Window from the menu bar.

4. Move Beads Window

To toggle the visibility of the Adjust Beads window, select Window -> Move Beads from the menu bar.

5. Bead Utilities Window

To toggle the visibility of the bead utilities window, select Window -> Bead Utilities from the menu bar.

5. Help Menu

1. About

To see information about the Virtual Bead Loom program, select Help -> About from the menu bar.

9. The Code Output Window

The Code Output window has been included in this version of the Virtual Bead Loom program to be used as a base starting point for future development of the program. It is not currently enabled to be interfaced with by the user interactively. This screen will not be visible on program start.

1. Code output menu tool bar

The menu tool bar in the Code Output window has many suggested options for future development. However, none of these options are enabled to do anything yet.

2. Code window

The code window within the Code Output window is currently enabled to show predetermined pseudo-code as bead objects are created. This pseudo-code is non-functional and cannot be edited by the user. This code is shown to help the user understand what functions may be called by the Virtual Bead Loom program. This is provided to help future developers enable programmable functionality.

Appendix A: Known Bugs and Workarounds

In the Mac environment:

The colors button functions normally, but the button itself cannot change colors. The background of the button is where the chosen color is displayed; this may be partially obscured by the button itself.

The Iteration Tools may have fields that are cut off by the size of the window. This has been attempted to be worked around by making the window resizable, and also to rearrange the fields within these tools to be more visible without having to change the size of the window.

When the grid is resized, the beads may not scale accordingly. There is no current workaround for this problem.

The XY Follow Mouse feature may appear lagged on some browsers. There is no current workaround for this problem.

General Bugs:

When the colors dialog box is closed, the beads already on the grid that were covered by the display may shift from their grid positions slightly, but will revert to their proper positions when another action is performed. Suggested workaround for this issue is to move the grid window slightly, or simply click on a tab.

The colors button is not capable of changing the color of existing layers. This is an issue to be fixed in later development. Suggested workaround for this issue is to create a new layer with the desired color, then delete the old layer.

Appendix B: Incomplete Features/ Features to be Completed in Later Versions

The colors feature will be changed to permit the changing of an existing layer's color.

The bead movement feature may be changed from shifting directions to entering a coordinate to move the bead object to.

There will be an option to use a non-negative grid for younger users.

There will be more examples added in the Options menu.

The Code Output window will display a normalized pseudo-code and will eventually permit the user to create pseudo-code to manipulate the Virtual Bead Loom rather than use the interface.

References

1. Instructor supplied documentation, WebCT via https://my.georgiasouthern.edu 2008.

2. Java API (Java 2 Platform SE 5.0), Sun Microsystems, http://java.sun.com/j2se/1.5.0/docs/api/ 2008.

3. Software Engineering: Theory and Practice 3rd Edition, Pfleeger, Shari Lawrence, et al.

4. The Institute of Electronic Engineers. (1986). *IEEE Standard for Software Verification and Validation Plans*. New York

5. CriTech Research. (1999). *Verification and Validation.* http://www.critech.com/vv.htm

Appendices

Appendix A: Defect Logs

Defect Recording Log
Students Harris Christopher Hollis, Chris Blake, Joe Petrizzi, Michael Lodge, Phillip Gipson Date 2/20/2008

Instructor Dr. Kera Bell-Watkins Program # 0

Date	Number	Type	Inject	Student	Fix Time
2/2/2008	1	10	Project Plan	Harris Christopher Hollis	1

Description: This was an update to the Project Plan Document which is now V1.2 - The project risks and analysis was added; The COCOMO II models were added.

Date	Number	Type	Inject	Student	Fix Time
2/20/2008	2	10	Defect Log	Harris Christopher Hollis	0.5

Description: Update the Defects Log for each member to post. Version is now 1.0.1

Date	Number	Type	Inject	Student	Fix Time
2/6/2008	3	10	CSDT Abstract	Joe Petrizzi	3

Description: After Skype meeting with clients, focus of the CSDT project was narrowed from framework for all programs to just the Virtual Bead Loom.
The abstract was modified to eliminate any requirements outside the new scope of the project.

Date	Number	Type	Inject	Student	Fix Time
3/6/2008	4	10	V&V Plan	Chris Blake	0.5

Description: Resource Summary portion of Verification and Validation Outline was updating to include all relevant information.

Date	Number	Type	Inject	Student	Fix Time
4/1/2008	5	50	Program	Harris Christopher Hollis	3

Description: Updated current version of the Graphical User Interface - current version is now 0.1.1

Date	Number	Type	Inject	Student	Fix Time
4/6/2008	6	80	Program	Phillip Gipson	1

Description: Print function was not working properly - it had an infinity loop that printed infinity pages. Looping as been fixed

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/6/2008	7	50	Program	Harris Christopher Hollis	2

Description: Print function did not display right page size when printing. This has been resolved.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/6/2008	8	10	V&V	Harris Christopher Hollis	3

Description: Updated and redone the V&V document - current version is now 1.1

Date	Number	Type	Inject	Student	Fix Time
4/7/2008	9	50	Program	Phillip Gipson	1

Beads were shifting backwards (Up shifts Down and Left shifts Right). Shifting if the beads have been fixed.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/7/2008	10	80	Program	Michael Lodge	0.5

Description: Goal images did not rescale to goal image button. This feature has been fixed.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/7/2008	11	80	Program	Phillip Gipson	1

Description: Shifting beads did not shift according to the coordinates. Shifting fixed.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/12/2008	12	80	Program	Michael Lodge	1

Description: Shifting of beads only moves the top layer and not a selected layer.

User can now select the layers in the layers tab and move a selected layer
(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/12/2008	13	80	Program	Michael Lodge	2

Description: Undo and Redo did not work correctly when bead is drawn. This issue has been corrected this using a clone layers ArrayList
(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/12/2008	14	80	Program	Joe Petrizzi	1.5

Description: Grid coordinates do not show correctly. Grid coordinates show correctly (formula implemented)
(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/13/2008	15	80	Program	Phillip Gipson	1

Description: JApplet, JFileChooser does not work in web browser. Applet must be signed in order for it to work

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/14/2008	16	80	Program	Chris Blake	0.5

Description: When beads draw on top of each other, black box forms. Used temp image instead of original image

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/16/2008	17	80	Program	Harris Christopher Hollis	0.5

Description: XY FollowMouse did not work correctly. Correctly works when selected

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/16/2008	18	80	Program	Michael Lodge	1

Description: Bead did not redraw when undo/redo called. Clear resets the z-value in class GridPanel. Issue resolved.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/16/2008	19	80	Program	Michael Lodge	1

Description: When "clear" is pressed and then "undo" is called, and new bead is drawn - selecting of beads do not work correctly for that given bead.
This issues has been fixed - The Z-Value does not reset anymore.
(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time

4/16/2008 20 80 Program Chris Blake 1.5
Description: Bead shading did not work correctly - some colors do not show. Used a temp image instead of original image

(Please refer to Anomaly Reports section 3 for more details)

Date Number Type Inject Student Fix Time
4/16/2008 21 80 Program Chris Blake 0.5
Description: XML opening did not work to show beads on screen. Current image became a temp image for fix.

(Please refer to Anomaly Reports section 3 for more details)

Date Number Type Inject Student Fix Time
4/16/2008 22 80 Program Phillip Gipson 0.75
Description: Save to JPEG did not show JDialog to save. Save to JPEG works with JFileChooser

(Please refer to Anomaly Reports section 3 for more details)

Date Number Type Inject Student Fix Time
4/17/2008 23 80 Program Chris Blake 0.5
Description: Drawing of beads do not check for bounds for max grid size. Bounds have been fixed to max grid size for input values.

(Please refer to Anomaly Reports section 3 for more details)

Date Number Type Inject Student Fix Time
4/17/2008 24 80 Program Harris Christopher Hollis 0.5
Description: Program didn't open XML beads on JApplet browser - Access Denied for File. Signed XOM-1-1.jar file

(Please refer to Anomaly Reports section 3 for more details)

Date Number Type Inject Student Fix Time
4/17/2008 25 80 Program Phillip Gipson 1
Description: Beads once again do not re-size to grid. This happen during clients using prototype version. Bead sizing now works.

Date Number Type Inject Student Fix Time
4/17/2008 26 80 Program Chris Blake 1
Description: program didn't open XML beads - Access Denied for Files.

(Please refer to Anomaly Reports section 3 for more details)

Date Number Type Inject Student Fix Time
4/18/2008 27 80 Program Phillip Gipson 2
Description: Move beads panel has been changed to "Adjust Beads". Adjust Beads have buttons spaced out. Undo and Redo buttons have been added to it. Removed the Colors button.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/18/2008	28	80	Program	Phillip Gipson, Chris Hollis	0.5

Description: Grid Panel now has the Colors button and Coordinates Label added to it. Removed the Bottom Menu Bar and moved Coordinates Label to grid panel

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/18/2008	29	80	Program	Michael Lodge	1

Description: Layers Panel has the Stating Z values first to see which layer is selected. Layers Panel added a Delete Layers Button.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/18/2008	30	80	Program	Michael Lodge	0.05

Description: Output Window is set visible false for future implementation.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/18/2008	31	80	Program	Phillip Gipson	0.75

Description: Triangle Iteration did not match the flash version.

(Please refer to Anomaly Reports section 3 for more details)

Date	Number	Type	Inject	Student	Fix Time
4/18/2008	32	80	Program	Phillip Gipson	0.75

Description: Default Colors changed to Red. Default values are changed.

(Please refer to Anomaly Reports section 3 for more details)

Appendix B: Anomaly Reports

Anomaly Report

Defect Number: 6

Date: Apr 6, 2008

Defect Type: 80

Inject: Main Program

Description: Upon coding the print function, a group member accidentally recalled the function recursively. This recalling lead to an infinite loop in which the function would constantly try to print the screen.

Solution: The program was set into Debug mode and stepped through the Print method within Beadloom.java. The recursive statement was discovered and eliminated

Anomaly Report

Defect Number: 7

Date: Apr 6, 2008

Defect Type: 50

Inject: Main Program

Description: Upon completion of the main the print function, a group member tested the function and retrieved the paper printout. It was discovered that the program was only capturing a portion of the user's screen and printing that section for the length of the paper.

Solution: A team member developed a formula that scaled down the GUI window to fit withing that portion that was being captured by the program.

Anomaly Report

Defect Number: 9

Date: Apr 7, 2008

Defect Type: 50

Inject: Main Program

Description: During development of the shift functions for the beads, it was discovered that the beads would move in the opposite direction as was desired by the user.

Solution: A team member discovered that while the beads were being shifted according to it's layer's offset value. The layer offset were being adjust as though the screen was working on a four-quadrant scheme, with coordinate (0,0) being in the middle of the screen. However, the screen works as though (0,0) is at the top-left position of the screen, increasing as you move right or down. The offsets were changed to accommodate this fact.

Appendix C: Executive Meetings

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Sunday, January 20, 2008 3:00 pm – 3:30 pm (IT-Building)

**We were not informed that the IT-building would be closed for Jan 18-21, thus we met outside the building and collaborated through e-mail discussions for this day.*

Present:

Chris Hollis

Chris Blake

Joe Petrizzi

Michael Lodge

Phillip Gipson

Absent:

Agenda:

To discuss overall project and make sure everyone knows each others' views on the project.
Discuss project requirements and gather proposals and analysis from each member.
Assign tasks for each member such as web-logging, spreadsheet, documents, group homework deadlines, and future meeting times.

Goals this week:

Continue assigning roles and relationships to each member and gather project details.
Discuss the documentation for the system specification, feasibility study outline and software project plan.
Adhere to the scheduled project deadlines.
Meet again in one week.

Status of last week's goals:

Goal	**Status**	**Notes**
Assign Team Group Roles	COMPLETE	Each member has their respective roles assigned.

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, January 22, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:

Chris Hollis

Chris Blake

Joe Petrizzi

Michael Lodge

Phillip Gipson

Absent:

Agenda:

Meet with the professor to discuss more project details
Ask questions that may or may not have impact on project.
Discuss requirements and analysis with project and come up with proposed questions to ask client for upcoming voice/video meeting.

Goals this week:

Discuss project planning and design ideas.
Assign tasks for each member concerning documentation below.
Have documentation for the system specification, feasibility study outline and software project plan completed.

Status of last week's goals:

Goal	Status	Notes
Establish a web-based log	50%	Currently working on getting server host to record data.
Assign this week's roles to team members	COMPLETE	Assigned spreadsheet, requirements and analysis proposals from each member
Met deadline Requirements and Analysis for project	COMPLETE	Each members input for the project's requirements and analysis, diagrams, pictures, etc…
Meet again in one week	COMPLETE	A meeting was held January 20th from 3:00 pm to 3:30 pm. Plus email collaboration.

Group Executive Meeting: 01/22/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, January 22, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:
Chris Hollis
Chris Blake
Joe Petrizzi
Michael Lodge
Phillip Gipson

Absent:
None

Agenda:

Meet with the professor to discuss more project details

Ask questions that may or may not have impact on project.

Discuss requirements and analysis with project and come up with proposed questions to ask client for upcoming voice/video meeting.

Goals this week:

Discuss project planning and design ideas.

Assign tasks for each member concerning documentation below.

Have documentation for the system specification, feasibility study outline and software project plan completed.

Status of last week's goals:

Goal	Status	Notes
Establish a web-based log	50%	Currently working on getting server host to record data.
Assign this week's roles to team members	COMPLETE	Assigned spreadsheet, requirements and analysis proposals from each member
Met deadline Requirements and Analysis for project	COMPLETE	Each members input for the project's requirements and analysis, diagrams, pictures, etc…
Meet again in one week	COMPLETE	A meeting was held January 20th from 3:00 pm to 3:30 pm. Plus email collaboration.

Notes:

1. Goals for week: Assign tasks for 3 documents due.

2. Email necessary pictures, documents, etc. to client 24 hours before Skype meeting.

3. Determine what sub-goals we have for the project.

4. Make sure Chris and Mike give us all necessary information about the wiki they fucked up last semester.

5. Ask the faculty what things are important to them about the wiki.

Questions for the client:

1. Should Java code should be made available for students in addition to the pseudocode?

2. Do you want to be able to upload your own gif's?

3. Is there any particular standard for the pseudocode that you want us to use?

4. To show the user what happens at runtime - can a step through option be implemented? For example: showing progression through for-loops.

5. Interface consists of 3 main development windows (Should each window be its own tab? Or have all windows constrained to one frame?

6. Do we allow the user to put in own parameters? Or choose from a set list? Ie: An object created using a method w/ parameters: John = new danceDesign (male, brown, body3);

**Bell said not to ask about openGL, but directX was ok to ask about. Then she mentioned to ask them what platforms they wanted the software to be able to run on. I don't think she understands the platform limitations of directX. Maybe we should stress to her tomorrow that openGL will be able to run on any platform that Java can, but directX cannot.

Group Executive Meeting: 01/29/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, January 29, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:
Chris Hollis
Chris Blake
Joe Petrizzi
Phillip Gipson
Michael Lodge (late)

Absent:

Agenda:

- Meet with the professor to discuss the Software Specification, Feasibility Study, and Software Project Plan submitted on Saturday.
- Further discuss the wiki project and what needs to be done by this group.
- Confirm what needs to be done on the current CSDT project.
- Submit previous week's Executive Minutes and Group Activity Log for review.

Goals this week:

Update group members on the online Wiki.
Assign tasks for each member concerning documentation below.

Compile list of questions for CS staff to answer about what they want in the online wiki.
Update acronyms list and limit them to 3 characters.

Update group log. All entries should be limited to 3 character acronyms and each goal should be incremented in sequence. Also, date and time formats need to be updated.

Make sure old versions of documents are kept on record.

Status of goal completion:

Goal	Status	Notes
Teach group members about online Wiki.	COMPLETE	Group members not involved in the Wiki project last semester have been caught up.
Assign this week's roles to team members	COMPLETE	Assigned features of VBL to each person.
Met deadline for VBL modeling for project	COMPLETE	Each person created use cases, UML diagrams, and written paragraphs for their assigned features.
Create questionnaire to give staff members about the Wiki.	COMPLETE	Questionnaire for staff members completed. Any additions to this document by group members can be added via Google Docs online.
Make sure old versions of documents are saved after	COMPLETE	Joe now keeps records of all document versions, while the most recent revisions are available to all group members on Google Docs.

revisions.		
Update acronyms list and group log.	75% COMPLETE	Acronyms are now limited to 3 characters and the group log is updated respectively. Group log still needs to have the date/time format updated.
Meet again in one week	COMPLETE	A meeting was held February 5th from 5:00 pm to 6:00 pm. Plus email collaboration.

Notes:

1. Update Group Activity Log with both current, short-term goals (documentation due, etc) and long-term goals (presentation date due, etc).

2. Group Activity Log acronyms should have cap of three letters and numbers should be incremented by each goal in sequence.

3. Format Group Activity Log dates to be in DD-MON-YY format instead of MM/DD/YYYY format.

4. Keep a repository of all old versions of documentation, but only submit the latest versions for assignments.

5. Upcoming due for this week: updated requirements, preliminary ACD diagram for design. Assign goals for these!

6. Hopefully receiving more stable list of requirements by Thursday, source code availability following shortly.

Group Executive Meeting: 02/05/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, February 5, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:
All

Absent:
None

Agenda:

Meet with professor to discuss VBL modeling completed from last group homework.
Present list of questions we wish to ask the staff about the online wiki.
Submit previous week's Executive Meeting Document and Group Activity Log.

Status of last week's goals:

Goal	Status	Notes
Teach group members about online Wiki.	COMPLETE	Group members not involved in the Wiki project last semester have been caught up.
Assign this week's roles to team members	COMPLETE	Assigned features of VBL to each person.
Met deadline for VBL modeling for project	COMPLETE	Each person created use cases, UML diagrams, and written paragraphs for their assigned features.
Create questionnaire to give staff members about the Wiki.	COMPLETE	Questionnaire for staff members completed. Any additions to this document by group members can be added via Google Docs online.
Make sure old versions	COMPLETE	Joe now keeps records of all document versions, while the most recent revisions are available to all

of documents are saved after revisions.		group members on Google Docs.
Update acronyms list and group log.	75% COMPLETE	Acronyms are now limited to 3 characters and the group log is updated respectively. Group log still needs to have the date/time format updated/.
Meet again in one week	COMPLETE	A meeting was held February 5th from 5:00 pm to 6:00 pm. Plus email collaboration.

Group Executive Meeting: 02/12/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, February 12, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:
Chris Blake
Chris Hollis
Michael
Phillip

Absent:
Joe

Agenda:

Discuss prototype for requirements specification'
Clarify what pseudocode model we will be using
Make sure there are no problems with Wiki access
Clarify questions about defect log

Goals this week:

Control flow diagrams for outlines.

More data flow diagrams
Create defect log
Update previous requirements and outlines

Status of goal completion:

Goal	Status	Notes
Requirements Specification	100% Complete	Project Requirements Specification completed for individual homework 2c Chris Hollis.
Design Specification	100% Complete	Project Design Specification completed for individual homework 2c by Chris Blake.
User's Manual	100% Complete	Project Preliminary User's Manual completed for individual homework 2c by Phillip.
Cost/Risk Analysis	100% Complete	Cost/Risk analysis for individual portions combined for the whole project.
Prototype	100% Complete	Prototype created.
Defect Log		

Notes:

1. Create prototype.
2. Create defect log.

Group Executive Meeting: 02/19/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, February 19, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:

Joe
Mike
Phillip
Chris Hollis
Chris Blake

Absent:

None

Agenda:

Decide who will be asking questions for next week's Skype meeting

Goals this week:

Commenting Java files for VBL

Discuss what is feasible for 4 weeks of work on the Wiki

Status of goal completion:

Goal	Status	Notes
Create Defect	100%	Defect log created for VBL and recent revisions added to the document.
Feasibility Study	100%	Feasibility outline created for the wiki for 150 hours.
V&V	5%	Potions of V&V assigned and 5% completed in class by each group member.
Break up Coding	0%	

Notes:

We need to make a defect log for the VBL, but not for the Wiki.
Still keep track of document versions for the Wiki.
We can begin a defect log of the VBL from this point forward, but if we can remember any recent changes, we should note them.

Make a feasibility study for the Wiki (150 hours [30 hours/person in 4 weeks]).
Get started on V&V (we will be assigned something for it to be due next Tuesday).
Break up sections for V&V and tell her who is responsible for what.
She will post a more formal outline for the V&V for the homework and will break up portions of it so that we will have two weeks to fully complete it.
We can begin coding now, based off of our design. We need to decide who will be coding what and notify Bell of how we divided it up.
Our group will be going first for Skype questions (Mike).
You should say the following when addressing the client:

Who you are
What we have done (% complete)
What will be done for the following week
How far we plan to progress on those goals
Our next scheduled milestone (what and when it is)
Then, ask our group questions

Group Executive Meeting: 02/26/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, February 26, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:

Joe
Mike
Phillip
Chris Hollis
Chris Blake

Absent:

None

Agenda:

Discuss V&V deadlines (when are they?)

Goals this week:

Complete V&V Outline and Conception Phase
Submit Feasibility Study for Wiki
Break up coding and begin programming
Decide on a team name and submit it to Dr. Bell

Notes:

1. No justification for our wiki feasibility requirements -- say what we can't get done and why.
2. Things to include in code comments

    1. programmer vision number

    2. group name
    3. group members and roles
    4. date started
    5. description
    6. inputs from different module(s)
    7. different outputs expected (return statements, things to be displayed)
    8. all of our limitations and assumptions based on inputs and output

3. She is pushing back the homework for V&V from Wednesday (no due date yet).

General Group Notes:

**Please log changes in the defect logs - Phillip, you can add the changes in the acronyms glossary..

Group Executive Meeting: 03/25/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, March 25, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:

Chris Blake

Chris Hollis
Michael
Phillip
Joe

Absent:

None

Goals this week:

Break up segments for coding based on Chris' GUI.

Continue Programming.
Have Mike and Phillip pair program the grid.
Create a plan for completing V&V phase.

Status of goal completion:

Goal	Status	Notes
V&V Documentation	50% Complete	V&V Documentation will continue throughout the end of the semester.
Wiki Feasibility Study	100% Complete	Completed by Chris Hollis.
Team Name	100% Complete	We decided on Team Ramrod.
Coding	5% Complete	We were forced to push back our coding phase.

Notes:

1. Create Milestones through end of semester.

2. Send Dr. Barnes the updated notebook.
3. Joe will ask questions to client this Thursday.
4. Share V&V documentation with client on Google Docs.
5. Teach Chris how to use arrays and methods.
6. Choose two people for pair programming of a module.
7. Have someone else test the code.
8. Mid-term 2: April 22 (presentation).

Group Executive Meeting: 04/01/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, April 1, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:

Mike
Phillip
Chris Hollis
Chris Blake

Absent:

Joe

Agenda:

Confirm schedule with Dr. Bell
Show our code for the grid segment of our project

Goals this week:

Correct V&V homework
Send Dr. Barnes an updated notebook
Share V&V documentation with the client as requested
Finish our coding phase (Mike, Phillip, and Chris Blake)
Revise code for GUI to meet our coding standard

Status of goal completion:

Goal	Status	Notes
Create Finalized Class Structure	100% Complete	Completed by all group members in a meeting.
Pair programming of grid segment	100% Complete	Coded by Mike and Phillip, tested by Chris Blake and Chris Hollis, documented by Joe.
Decide on milestones through end of the semester	50% Complete	Partially completed, due dates are just now being confirmed, so schedule may have to be revised.

Correct V&V documentation (group homework)	0% Complete	Each person will be correcting the portion of the homework that they were originally responsible for.
Create a plan for completing the remaining V&V documentation	50% Complete	Same as milestones schedule above, due dates are just now being confirmed.

Group Executive Meeting: 04/08/08

GSU Software Engineering Group 1

Executive Board Weekly Meeting

Date and Time: Tuesday, April 8, 2008 5:00 pm – 6:00 pm (IT-Building)

Present:

Mike
Phillip
Chris Hollis
Chris Blake

Absent:

Joe

Agenda:

Show our current project to Dr. Bell.
Ask if she knows what the linear iteration does? If not, we'll ask the client on Thursday.
Concerning the redo V&V and Implementation
Use Cases - redoing from previous java bead loom to ours?

Goals this week:

Finish our coding phase.
Send the client an updated notebook.
Complete Implementation and Test Phase of V&V Documentation.
Redo group homework 2b.

Status of goal completion:

Goal	Status	Notes
Homework #7	100% Complete	Done by Phillip and Joe with input from the rest of the group.
Homework 6a Redo	100% Complete	Completed by Joe
Homework 3a Redo	100% Complete	Completed by Chris Hollis.
Revised code for GUI to meet coding standard	80% Complete	GUI code has been revised now nearly meets our coding standard.
Progress into our coding phase.	80% Complete	We are nearing completion of our coding phase, though we may have to push it back until this weekend.

Notes:

1. Place all of our constants and external inputs into a separate class or interface.

Appendix D: Group Time Log

Appendix E: System Metrics

CoordList.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes:	Min	Max	Median	Avg
Attributes per class:	0	0	0	0
Methods per class:	10	10	10	10
Complexity per class:	6	6	6	6
Fan Out per class:	33	33	33	33
Fan In per class:	1	1	1	1
Ancestors per class:	0	0	0	0
Statements per class:	73	73	73	73

****
CoordList.java
Lines of Commentary: 4 Commentary Percentage: 4.52%
*****public class CoordList
Attributes: 0
Methods: 10
Lines of Comments: 3
Complexity: 6
Fan Out: 33
Fan In: 1
Ancestors: 0
Statements: 73
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Methods:

public int getCoord : input parameter types(int, int)
Attributes: 0
Lines of Comments: 1
Complexity: 2
Fan Out: 2
Fan In: 0
Statements: 2
Input Parameters: 2
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void addValues : input parameter types(int, int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 2
Fan In: 0
Statements: 2

Input Parameters: 2
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getSize : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void addX : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void addY : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void sortByX : input parameter types(int, int)
Attributes: 0
Lines of Comments: 0
Complexity: 4
Fan Out: 6
Fan In: 0
Statements: 10

Input Parameters: 2
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void sortByY : input parameter types()
Attributes: 1
Lines of Comments: 0
Complexity: 5
Fan Out: 33
Fan In: 0
Statements: 27
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Attributes:
int temp

public void sort : input parameter types(ArrayList<Integer>,ArrayList<Integer>)
Attributes: 1
Lines of Comments: 0
Complexity: 4
Fan Out: 12
Fan In: 1
Statements: 11
Input Parameters: 2
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Attributes:
int temp

public void fillTriangle : input parameter types()
Attributes: 1
Lines of Comments: 1
Complexity: 6
Fan Out: 13
Fan In: 0
Statements: 7
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Attributes:

int loopCount

public String toString : input parameter types()
Attributes: 1
Lines of Comments: 0
Complexity: 4
Fan Out: 6
Fan In: 0
Statements: 7
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Attributes:
String output

GridPanel.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes:	Min	Max	Median	Avg
Attributes per class:	12	12	12	12
Methods per class:	29	29	29	29

Complexity per class:	10	10	10	10
Fan Out per class:	51	51	51	51
Fan In per class:	2	2	2	2
Ancestors per class:	1	1	1	1
Statements per class:	151	151	151	151

****

GridPanel.java
Lines of Commentary: 10 Commentary Percentage: 12.45%
***** class GridPanel
Ancestors: JPanel
Attributes: 12
Methods: 29
Lines of Comments: 10
Complexity: 10
Fan Out: 51
Fan In: 2
Ancestors: 1
Statements: 151
Attribute Closure: 0.33
Attribute OO-ness: 0.16

Attributes:
private BeadLoom bl
private boolean repaint
private boolean hidelines
private boolean isMainGrid
int GRID_SIZE
int FILL
int PAD
double w
double h
double xInc
double yInc
Image catimage
Methods:

public void setMain : input parameter types(boolean)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setGridSize : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setLines : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 2
Fan Out: 0
Fan In: 0
Statements: 2
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

protected void paintComponent : input parameter types(Graphics)
Attributes: 9
Lines of Comments: 3
Complexity: 10
Fan Out: 51
Fan In: 1
Statements: 39
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.55

Attributes:
JComboBox lb
Layer temp
Graphics2D g2
double x1
BasicStroke stroke
BasicStroke wideStroke
int iconSize
int xcoord
int ycoord

public void setImage : input parameter types(Image)

Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 1
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public Image getImage : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 2
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void mouseClicked : input parameter types(MouseEvent)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void mousePressed : input parameter types(MouseEvent)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void mouseEntered : input parameter types(MouseEvent)

Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void mouseExited : input parameter types(MouseEvent)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void mouseReleased : input parameter types(MouseEvent)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void mouseDragged : input parameter types(MouseEvent)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void mouseMoved : input parameter types(MouseEvent)

Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JLabel findMousePosition : input parameter types(MouseEvent)
Attributes: 3
Lines of Comments: 0
Complexity: 3
Fan Out: 9
Fan In: 0
Statements: 7
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.33

Attributes:
JLabel l
int get_row
int get_col

public void paint : input parameter types(Image, Graphics, int, int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 0
Statements: 1
Input Parameters: 4
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JLabel output : input parameter types(String)
Attributes: 1
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 1
Statements: 2
Input Parameters: 1

Attribute Closure: 0.0
Attribute OO-ness: 1.0

Attributes:
JLabel l

public void setPanelGridSize : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 6
Fan Out: 16
Fan In: 0
Statements: 15
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getGridSize : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 1
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getPad : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public double getXINC : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0

Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public double getYINC : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void refreshGrid : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void addLayer : input parameter types(Layer)
Attributes: 0
Lines of Comments: 1
Complexity: 1
Fan Out: 10
Fan In: 0
Statements: 6
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void undo : input parameter types()
Attributes: 1
Lines of Comments: 0
Complexity: 2
Fan Out: 12

Fan In: 0
Statements: 7
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Attributes:
int undoSize

public void redo : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 2
Fan Out: 13
Fan In: 0
Statements: 6
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void shift : input parameter types(String, int)
Attributes: 0
Lines of Comments: 0
Complexity: 7
Fan Out: 24
Fan In: 0
Statements: 9
Input Parameters: 2
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void clear : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 5
Fan In: 1
Statements: 4
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setLayers : input parameter types(ArrayList<Layer>)
Attributes: 0

Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void newGrid : input parameter types()
Attributes: 1
Lines of Comments: 1
Complexity: 5
Fan Out: 46
Fan In: 0
Statements: 26
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 1.0

Attributes:
Object[] options

GUIGoalImages.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes:	Min	Max	Median	Avg
Attributes per class:	5	5	5	5
Methods per class:	7	7	7	7
Complexity per class:	10	10	10	10
Fan Out per class:	24	24	24	24
Fan In per class:	1	1	1	1
Ancestors per class:	2	2	2	2
Statements per class:	53	53	53	53

****
GUIGoalImages.java
Lines of Commentary: 3 Commentary Percentage: 6.25%
*****public class GUIGoalImages

Ancestors: JInternalFrame ActionListener
Attributes: 5
Methods: 7
Lines of Comments: 3
Complexity: 10
Fan Out: 24
Fan In: 1
Ancestors: 2
Statements: 53
Attribute Closure: 1.0
Attribute OO-ness: 0.8

Attributes:
private JLabel CurrentImagesLabel
private JButton GoalImagesButton
private File beadFile
private boolean isXML
private BeadLoom bl
Methods:

public void setBeadLoom: input parameter types(BeadLoom)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void actionPerfomed: input parameter types(ActionEvent)
Attributes: 3
Lines of Comments: 0
Complexity: 3
Fan Out: 9
Fan In: 0
Statements: 11
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 1.0

Attributes:
File tempFile
FileDialog chooser
ImageIcon ii

public ImageIcon LoadFromXML: input parameter types(File)
Attributes: 4
Lines of Comments: 0
Complexity: 2
Fan Out: 24
Fan In: 1
Statements: 18
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.75

Attributes:
String fileName
GridPanel gp
BufferedImage myImage
Graphics2D g2

public boolean accept : input parameter types(File, String)
Attributes: 1
Lines of Comments: 0
Complexity: 10
Fan Out: 10
Fan In: 1
Statements: 9
Input Parameters: 2
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Attributes:
int count

public JButton getGoalImagesButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JLabel getCurrentImagesLabel : input parameter types()
Attributes: 0

Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public ImageIcon getScaledImage : input parameter types(ImageIcon)
Attributes: 3
Lines of Comments: 0
Complexity: 1
Fan Out: 2
Fan In: 1
Statements: 4
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 1.0

Attributes:
Image myImage
Image scaledImage
ImageIcon m

GUIInputTools.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes:	Min	Max	Median	Avg
Attributes per class:	115	115	115	115
Methods per class:	48	48	48	48
Complexity per class:	153	153	153	153
Fan Out per class:	231	231	231	231
Fan In per class:	7	7	7	7
Ancestors per class:	3	3	3	3
Statements per class:	771	771	771	771

****
GUIInputTools.java
Lines of Commentary: 121 Commentary Percentage: 17.33%
*****public class GUIInputTools
Ancestors: JApplet ActionListener ItemListener
Attributes: 115
Methods: 48

Lines of Comments: 121
Complexity: 153
Fan Out: 231
Fan In: 7
Ancestors: 3
Statements: 771
Attribute Closure: 0.93
Attribute OO-ness: 0.95

Attributes:

```
  private GridPanel grid
  private Layer lay
  private JTabbedPane BeadLoomUtilitiesTabbedPane
  private JPanel BeadLoomUtilitiesPanel
  private JLabel DrawPointXLabel
  private JLabel DrawPointYLabel
  private JLabel DrawLineX1Label
  private JLabel DrawLineY1Label
  private JLabel DrawLineX2Label
  private JLabel DrawLineY2Label
  private JTextField DrawPointXTextField
  private JTextField DrawPointYTextField
  private JTextField DrawLineX1TextField
  private JTextField DrawLineX2TextField
  private JTextField DrawLineY1TextField
  private JTextField DrawLineY2TextField
  private JButton DrawPointButton
  private JButton DrawLineButton
  private JTabbedPane LayersDrawTabbedPane
  private JLayeredPane LayersSelectPane
  private JComboBox LayersBox
  private JButton DeleteLayer
  private JLabel LayersLabel
  private JTabbedPane RectangleDrawTabbedPane
  private JLayeredPane RectangleDrawLayeredPane
  private JButton DrawRectangleButton
  private JLabel DrawRectangleX1Label
  private JLabel DrawRectangleX2Label
  private JLabel DrawRectangleY1Label
  private JLabel DrawRectangleY2Label
  private JTextField DrawRectangleX1TextField
  private JTextField DrawRectangleX2TextField
  private JTextField DrawRectangleY1TextField
  private JTextField DrawRectangleY2TextField
  private JTabbedPane TriangleDrawTabbedPane
  private JPanel TriangleDrawPanel
  private JLabel DrawTriangleX1Label
  private JLabel DrawTriangleX2Label
```

```
private JLabel DrawTriangleX3Label
private JLabel DrawTriangleY1Label
private JLabel DrawTriangleY2Label
private JLabel DrawTriangleY3Label
private JButton DrawTriangleButton
private JTextField DrawTriangleX1TextField
private JTextField DrawTriangleX2TextField
private JTextField DrawTriangleX3TextField
private JTextField DrawTriangleY1TextField
private JTextField DrawTriangleY2TextField
private JTextField DrawTriangleY3TextField
private ButtonGroup IterationRadioButtons
private JTabbedPane LinearIterationTabbedPane
private JPanel LinearIterationPanel
private JButton LinearIterationDrawButton
private JLabel LinearIterationStartLabel
private JTextField LinearIterationStartLengthTextField
private JLabel LinearIterationFromLabel
private JLabel LinearItereationXLabel
private JTextField LinearItereationXTextFiedd
private JLabel LinearItereationYLabel
private JTextField LinearIterationYTextFiedd
private JLabel LinearIterationAddLabel
private JTextField LinearIterationFirstEndTextField
private JLabel LinearIterationFirstEndLabel
private JTextField LinearIterationSecondEndTextField
private JLabel LinearIterationSecondEndLabel
private JTextField LinearIterationTotalRowsTextField
private JLabel LinearIterationTotalRowsLabel
private ButtonGroup LinearIterationDirectionButtonGroup
private JRadioButton LinearIterationPYDirectionButton
private JRadioButton LinearIterationNYDirectionButton
private JRadioButton LinearIterationPXDirectionButton
private JRadioButton LinearIterationNXDirectionButton
private JLabel LinearIterationPYLabel
private JLabel LinearIterationNYLabel
private JLabel LinearIterationPXLabel
private JLabel LinearIterationNXLabel
private JTabbedPane BeadIterationTabbedPane
private JPanel BeadIterationPanel
private JButton IterateButton
private JLabel BeadIterateStartXLabel
private JLabel BeadIterateStepsLabel
private JLabel BeadIterateWidthLabel
private JLabel BeadIterateStartYLabel
private JLabel BeadIterateCyclesLabel
private JLabel BeadIterateFilledLabel
private JLabel BeadDirectionLabel
```

private JTextField BeadIterateStartXTextField
private JTextField BeadIterateStepsTextField
private JTextField BeadIterateWidthTextField
private JTextField BeadIterateCyclesTextField
private JTextField BeadIterateStartYTextField
private JCheckBox BeadIterateFilledCheckBox
private JRadioButton BeadIterateHorizontalButton
private JRadioButton BeadIterateNegHorizontalButton
private JRadioButton BeadIterateVerticalButton
private JRadioButton BeadIterateNegVerticalButton
private JTabbedPane TrigFunctionsTabbedPane
private JLayeredPane TrigFunctionsLayeredPane
private JButton GraphTrigFunctionButton
private JLabel YEqualsLabel
private JTextField TrigMultiplierTextField
private JComboBox TrigFunctionSelector
private JLabel TrigOpenParenthesisLabel
private JTextField TrigFunctionArgumentTextField
private JLabel TrigCloseParenthesisLabel
private JComboBox TrigPlusOrMinusSelector
private JTextField TrigAdditiveTextField
int GRID_SIZE
double xInc
double yInc
Image catimage
double gridX
double gridY
URL u
private Color color

Methods:

public void setGrid : input parameter types(GridPanel)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void drawLine : input parameter types(int, int, int, int, ArrayList<Integer>, ArrayList<Integer>)
Attributes: 11
Lines of Comments: 4
Complexity: 14

Fan Out: 13
Fan In: 1
Statements: 40
Input Parameters: 6
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Attributes:
int temp
int temp
double m
int startX
int endX
int intY
double doubleY
int startY
int endY
int intX
double doubleX

public void itemStateChanged : input parameter types(ItemEvent)
Attributes: 0
Lines of Comments: 0
Complexity: 2
Fan Out: 2
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void actionPerformed : input parameter types(ActionEvent)
Attributes: 62
Lines of Comments: 5
Complexity: 153
Fan Out: 231
Fan In: 0
Statements: 251
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.22

Attributes:
int x
int y

CoordList coords
Layer l
int x1
int x2
int y1
int y2
CoordList coords
Layer l
int x1
int x2
int y1
int y2
int temp
int temp
CoordList coords
Layer l
int startX
int startY
int steps
int width
int cycles
boolean incY
boolean fill
int tempSteps
CoordList coords
Layer l
int x1
int x2
int x3
int y1
int y2
int y3
int start
int stop
CoordList coords
Layer l
int startLength
int x1
int y1
int inc1
int inc2
int rows
int x2
int y2
CoordList coords
Layer l
String multiplierString
String argumentString

float multiplier
int additive
boolean legalMultiplier
int numerator
int denominator
int numerator
int denominator
double functionValue
final int size
int increment
CoordList coords
Layer l

public JTabbedPane getBeadLoomUtilitiesTabbedPane : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setGridSize : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setPad : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setxInc : input parameter types(double)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setyInc : input parameter types(double)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getGridSize : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getPad : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public double getxInc : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public double getyInc : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getDrawPointButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getDrawLineButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getRectangleButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getIterateButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getDrawTriangleButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getGraphTrigFunctionButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawPointXTextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawPointYTextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawLineX1TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawLineX2TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawLineY1TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawLineY2TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawRectangleX1TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawRectangleX2TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawRectangleY1TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawRectangleY2TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getBeadIterateStartXTextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getBeadIterateStartYTextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getBeadIterateStepsTextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getBeadIterateWidthTextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getBeadIterateCyclesTextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawTriangleX1TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawTriangleX2TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawTriangleX3TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawTriangleY1TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawTriangleY2TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextField getDrawTriangleY3TextField : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setImage : input parameter types(Image)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 7
Statements: 2
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getSelectedZ : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JComboBox getLayerList : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 2
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public Layer getLay : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getLinearIterationDrawButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

private synchronized Image makeBullet : input parameter types(Color)
Attributes: 3
Lines of Comments: 0
Complexity: 1
Fan Out: 2
Fan In: 1
Statements: 5
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 1.0

Attributes:
Image bullet
ImageFilter imgf
ImageProducer imgp

public void setColor : input parameter types(Color)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 7

Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getLinearIterationDirection : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 5
Fan Out: 4
Fan In: 1
Statements: 5
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

GUIMenuBar.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes:	Min	Max	Median	Avg
Attributes per class:	52	52	52	52
Methods per class:	38	38	38	38
Complexity per class:	2	2	2	2
Fan Out per class:	3	3	3	3
Fan In per class:	1	1	1	1
Ancestors per class:	0	0	0	0
Statements per class:	209	209	209	209

****
GUIMenuBar.java
Lines of Commentary: 56 Commentary Percentage: 37.24%
*****public class GUIMenuBar
Attributes: 52
Methods: 38
Lines of Comments: 56
Complexity: 2
Fan Out: 3
Fan In: 1
Ancestors: 0
Statements: 209
Attribute Closure: 0.88
Attribute OO-ness: 0.88

Attributes:
private GridPanel gridPanel
private GUIInputTools tools
static final int GRID_40
static final int GRID_50
static final int GRID_60
static final int GRID_70
static final int GRID_90
static final int GRID_100
private JMenuBar TopMenuBar

private JMenu FileMenu
private JMenuItem NewBeadsItemMenu
private JMenuItem OpenBeadsItemMenu
private JMenu SaveBeadsMenu
private JMenuItem SaveToJPEGMenuItem
private JMenuItem SaveToXMLMenuItem
private JMenuItem PrintMenuItem
private JMenuItem ExitMenuItem
private JMenu EditMenu
private JMenuItem UndoMenuItem
private JMenuItem RedoMenuItem
private JMenu OptionsMenu
private JMenu GridOptionsMenu
private JMenu GridSizeMenu
private JCheckBoxMenuItem XYFollowMouseMenuItem
private JCheckBoxMenuItem ShowGridMenuItem
private JCheckBoxMenuItem HideGridMenuItem
private JCheckBoxMenuItem LockGridMenuItem
private JCheckBoxMenuItem AxisValuesMenuItem
private JCheckBoxMenuItem MarginValuesMenuItem
private JMenu ExamplesMenu
private JMenuItem DrawSnakeMenuItem
private JMenuItem DrawBearMenuItem
private JMenuItem DrawSantaMenuItem
private JMenu DevelopmentCodeMenu
private JMenu WindowMenu
private JMenuItem GridMenuItem
private JMenuItem GoalImagesMenuItem
private JMenuItem OutputWindowMenuItem
private JCheckBoxMenuItem MoveBeadsItem
private JCheckBoxMenuItem BeadUtilitiesItem
private JMenu HelpMenu
private JMenuItem PseudoCodeMenuItem
private JMenuItem AboutMenuItem
private JRadioButton Grid_30
private JRadioButton Grid_40
private JRadioButton Grid_50
private JRadioButton Grid_60
private JRadioButton Grid_70
private JRadioButton Grid_80
private JRadioButton Grid_90
private JRadioButton Grid_100
private ButtonGroup GridSize

Methods:

private void Grid_40ActionPerformed : input parameter types(ActionEvent)
Attributes: 0
Lines of Comments: 1

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

private void NewBeadsItemMenuActionPerformed : input parameter types(ActionEvent)
Attributes: 0
Lines of Comments: 1
Complexity: 1
Fan Out: 0
Fan In: 1
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

private void OpenBeadsItemMenuActionPerformed : input parameter types(ActionEvent)
Attributes: 0
Lines of Comments: 1
Complexity: 1
Fan Out: 0
Fan In: 1
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

private void SaveToXMLMenuItemActionPerformed : input parameter types(ActionEvent)
Attributes: 0
Lines of Comments: 1
Complexity: 1
Fan Out: 0
Fan In: 1
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

private void SaveToJPEGMenuItemActionPerformed : input parameter types(ActionEvent)
Attributes: 0
Lines of Comments: 1

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

private void ExitMenuItemActionPerformed : input parameter types(ActionEvent)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 1
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

private void PseudoCodeMenuItemActionPerformed : input parameter types(ActionEvent)
Attributes: 0
Lines of Comments: 1
Complexity: 1
Fan Out: 0
Fan In: 1
Statements: 0
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void enableUndo : input parameter types(boolean)
Attributes: 0
Lines of Comments: 0
Complexity: 2
Fan Out: 3
Fan In: 0
Statements: 3
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void enableRedo : input parameter types(boolean)
Attributes: 0
Lines of Comments: 0

Complexity: 2
Fan Out: 3
Fan In: 0
Statements: 3
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuBar getJMenuBar : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getGridMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getGoalImagesMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getOutputWindowMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getMoveBeadsItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getBeadUtilitiesItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getPrintMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getDrawSnakeMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getDrawBearMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getDrawSantaMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getAboutMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getSaveToJPEGMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getSaveToXMLMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getNewBeadsItem: input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getUndoMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getRedoMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getXYFollowMouseMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getShowGridMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getHideGridMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getLockGridMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JMenuItem getOpenBeadsMenuItem : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getGrid_30 : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getGrid_40 : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getGrid_50 : input parameter types()
Attributes: 0
Lines of Comments: 0

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getGrid_60 : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getGrid_70 : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getGrid_80 : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getGrid_90 : input parameter types()
Attributes: 0
Lines of Comments: 0

Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JRadioButton getGrid_100 : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

GUIMovePanel.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes:	Min	Max	Median	Avg
Attributes per class:	10	10	10	10
Methods per class:	11	11	11	11
Complexity per class:	1	1	1	1
Fan Out per class:	0	0	0	0
Fan In per class:	0	0	0	0
Ancestors per class:	1	1	1	1
Statements per class:	53	53	53	53

****
GUIMovePanel.java
Lines of Commentary: 22 Commentary Percentage: 24.96%
*****public class GUIMovePanel
Ancestors: JPanel
Attributes: 10
Methods: 11
Lines of Comments: 22
Complexity: 1
Fan Out: 0
Fan In: 0
Ancestors: 1
Statements: 53
Attribute Closure: 1.0
Attribute OO-ness: 1.0

 Attributes:
  private JPanel MoveLastDrawnBeadsPanel
  private JButton MoveUpButton
  private JButton MoveLeftButton
  private JButton MoveRightButton
  private JButton MoveDownButton
  private JButton TopColorsButton
  private JButton ClearGridButton
  private JButton UndoButton
  private JButton RedoButton
  private Color color
 Methods:

public Color getColor : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0

Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setColor : input parameter types(Color)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JPanel getMoveLastDrawnBeadsPanel: input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getMoveUpButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getMoveLeftButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0

Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getMoveRightButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getMoveDownButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getTopColorsButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getClearGridButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0

Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getUndoButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getRedoButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

GUIOutputWindow.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes: Min Max Median Avg

Attributes per class: 20 20 20 20
Methods per class: 5 5 5 5
Complexity per class: 1 1 1 1
Fan Out per class: 1 1 1 1
Fan In per class: 1 1 1 1
Ancestors per class: 1 1 1 1
Statements per class: 64 64 64 64
****
GUIOutputWindow.java
Lines of Commentary: 20 Commentary Percentage: 40.0%
*****public class GUIOutputWindow
Ancestors: JInternalFrame
Attributes: 20
Methods: 5
Lines of Comments: 20
Complexity: 1
Fan Out: 1
Fan In: 1
Ancestors: 1
Statements: 64
Attribute Closure: 1.0
Attribute OO-ness: 1.0

Attributes:
private JInternalFrame OutputWindow
private JInternalFrame OutputCodeWindowFrame
private JMenuBar OutputCodeWindowMenuBar
private JMenu OuputCodeOptionsMenu
private JMenuItem OpenLogMenuItem
private JMenuItem StopLogMenuItem
private JMenuItem SaveOutputMenuItem
private JMenu OutputDevelopmentCodeMenu
private JMenu FunctionsMenu
private JMenuItem IfStatementMenuItem
private JMenuItem WhileLoopMenuItem
private JMenuItem DoLoopMenuItem
private JMenuItem ForLoopMenuItem
private JMenuItem DrawTriangleMenuItem
private JMenuItem DrawRectangleMenuItem
private JMenu SourceCodeHelpMenu
private JMenuItem CodeMeaningMenuItem
private JScrollPane OutputWindowScrollPane
private JTextArea OutputWindowPane
private JButton ClearOutputWindowButton
Methods:

public JMenuBar getOutputCodeWindowMenuBar : input parameter types()
Attributes: 0

Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JScrollPane getOutputWindowScrollPane : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JButton getClearOutputWindowButton : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public JTextArea getEditorPane : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void ClearOutputWindowButtonActionPerformed : input parameter types(ActionEvent)
Attributes: 0

Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 1
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Layer.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes:	Min	Max	Median	Avg
Attributes per class:	8	8	8	8
Methods per class:	20	20	20	20
Complexity per class:	2	2	2	2

Fan Out per class: 3 3 3 3
Fan In per class: 2 2 2 2
Ancestors per class: 0 0 0 0
Statements per class: 43 43 43 43
****
Layer.java
Lines of Commentary: 19 Commentary Percentage: 221.19%
*****public class Layer
Attributes: 8
Methods: 20
Lines of Comments: 18
Complexity: 2
Fan Out: 3
Fan In: 2
Ancestors: 0
Statements: 43
Attribute Closure: 1.0
Attribute OO-ness: 0.37

Attributes:
private String type
private Color color
private String beadType
private int xOffset
private int zValue
private Image image
private static int ZZzzzz
private static GUIMovePanel movePanel
Methods:

public String getType : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public CoordList getCoords : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0

Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public Color getColor : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 1
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public String getBeadType : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getXOffset : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getYOffset : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0

Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getSize : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 1
Fan In: 2
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public int getZValue : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public Image getImage : input parameter types()
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setType : input parameter types(String)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0

Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setCoords : input parameter types(CoordList)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setColor : input parameter types(Color)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setBeadType : input parameter types(String)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setXOffset : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0

Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setYOffset : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setZValue : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setImage : input parameter types(Image)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public void setMovePanel : input parameter types(GUIMovePanel)
Attributes: 0
Lines of Comments: 0
Complexity: 1
Fan Out: 0
Fan In: 0

Statements: 1
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

public String toString : input parameter types()
Attributes: 1
Lines of Comments: 0
Complexity: 2
Fan Out: 3
Fan In: 0
Statements: 7
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Attributes:
String output

public static void resetZ : input parameter types()
Attributes: 0
Lines of Comments: 1
Complexity: 1
Fan Out: 0
Fan In: 0
Statements: 1
Input Parameters: 0
Attribute Closure: 0.0
Attribute OO-ness: 0.0

SaveLayer.java Metrics

MetaPackage : C:\Documents and Settings\student\Desktop\project\project

This package's attributes:	Min	Max	Median	Avg
Attributes per class:	2	2	2	2
Methods per class:	3	3	3	3
Complexity per class:	16	16	16	16
Fan Out per class:	58	58	58	58
Fan In per class:	1	1	1	1
Ancestors per class:	0	0	0	0
Statements per class:	90	90	90	90

****
SaveLayer.java

Lines of Commentary: 59 Commentary Percentage: 54.29%
*****public class SaveLayer
Attributes: 2
Methods: 3
Lines of Comments: 20
Complexity: 16
Fan Out: 58
Fan In: 1
Ancestors: 0
Statements: 90
Attribute Closure: 1.0
Attribute OO-ness: 0.5

Attributes:
private static CoordList c
private static int layerCount
Methods:

public static void saveLoom : input parameter types(ArrayList<Layer>, String)
Attributes: 19
Lines of Comments: 9
Complexity: 4
Fan Out: 55
Fan In: 0
Statements: 48
Input Parameters: 2
Attribute Closure: 0.0
Attribute OO-ness: 0.89

Attributes:
Element loom
Element layerElement
Element coords
Element options
Attribute layerType
Attribute color
Attribute xOffset
Attribute yOffset
Attribute beadShape
Attribute ZValue
String x
String y
Document doc
FileOutputStream out
Serializer serializer
File temp
Builder builder

Document doc
Element root

public static void listChildren : input parameter types(Node, int)
Attributes: 5
Lines of Comments: 7
Complexity: 16
Fan Out: 58
Fan In: 1
Statements: 34
Input Parameters: 2
Attribute Closure: 0.0
Attribute OO-ness: 0.4

Attributes:
String data
String data2
Element temp
Attribute attribute
String attValue

private static void printSpaces : input parameter types(int)
Attributes: 0
Lines of Comments: 0
Complexity: 2
Fan Out: 1
Fan In: 1
Statements: 3
Input Parameters: 1
Attribute Closure: 0.0
Attribute OO-ness: 0.0

Appendix F: Source Code