

# NAVAL POSTGRADUATE SCHOOL

## Monterey, California



## THESIS

**ANALYSIS, DESIGN, IMPLEMENTATION AND  
EVALUATION OF GRAPHICAL DESIGN TOOL TO  
DEVELOP DISCRETE EVENT SIMULATION MODELS  
USING EVENT GRAPHS AND SIMKIT**

by

Angel San Jose

September 2001

Thesis Advisor:

Arnold Buss

Thesis Co-Advisor:

Nita Miller

Second Reader:

Gordon Bradley

**Approved for public release; distribution is unlimited**



<b>REPORT DOCUMENTATION PAGE</b>			<i>Form Approved OMB No. 0704-0188</i>
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
<b>1. AGENCY USE ONLY</b>	<b>2. REPORT DATE</b> September 2001	<b>3. REPORT TYPE AND DATES COVERED</b> Master's Thesis	
<b>4. TITLE AND SUBTITLE:</b> Analysis, Design, Implementation And Evaluation Of Graphical Design Tool To Develop Discrete Event Simulation Models Using Event Graphs And Simkit			<b>5. FUNDING NUMBERS</b>
<b>6. AUTHOR(S)</b> Angel E. San Jose			<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b> Naval Postgraduate School Monterey, CA 93943-5000			<b>10. SPONSORING / MONITORING AGENCY REPORT NUMBER</b>
<b>9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b> N/A			<b>11. SUPPLEMENTARY NOTES</b> The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.
<b>12a. DISTRIBUTION / AVAILABILITY STATEMENT</b> Approved for public release; distribution is unlimited			<b>12b. DISTRIBUTION CODE</b>
<b>13. ABSTRACT</b> Discrete Event Simulation (DES) is one of the most widely used methodologies for Operations Research (OR) modeling and analysis. However, designing and implementing DES can be a time-consuming and error-prone task. This thesis designed, implemented and evaluated a tool, the Event Graph Graphical Design Tool (EGGDT), to help OR analysts in the design, implementation, and maintenance of DES reducing the development and debugging times. The Unified Modeling Language was used to document the development of the EGGDT, which was programmed in Java using J2D and Swing. Human Factors techniques were employed to help in the design process and to evaluate the final prototype of the EGGDT. During the design process, two formative experiments were performed to evaluate the Graphical User Interface design decisions. A final summative experiment was done to test if the potential users consider the tool a useful means to develop OR simulations. Participants of the experiments agreed that tools like the EGGDT are an essential instrument when developing simulations.			
<b>14. SUBJECT TERMS</b> Modeling and Simulation, Human Factors, Discrete Event Simulation, Event Graph, CASE Tools, UML, Graphical Interface Design, Simkit, Java, J2D, Swing.			<b>15. NUMBER OF PAGES</b> 136
			<b>16. PRICE CODE</b>
<b>17. SECURITY CLASSIFICATION OF REPORT</b> Unclassified	<b>18. SECURITY CLASSIFICATION OF THIS PAGE</b> Unclassified	<b>19. SECURITY CLASSIFICATION OF ABSTRACT</b> Unclassified	<b>20. LIMITATION OF ABSTRACT</b> UL

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

**ANALYSIS, DESIGN, IMPLEMENTATION AND EVALUATION OF  
GRAPHICAL DESIGN TOOL TO DEVELOP DISCRETE EVENT SIMULATION  
MODELS USING EVENT GRAPHS AND SIMKIT**

Angel E. San Jose  
Lieutenant Commander, Spanish Navy  
B.S., Spanish Naval Academy, 1986

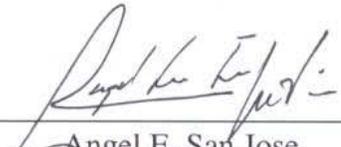
Submitted in partial fulfillment of the  
requirements for the degree of

**MASTER OF SCIENCE IN OPERATIONS RESEARCH**

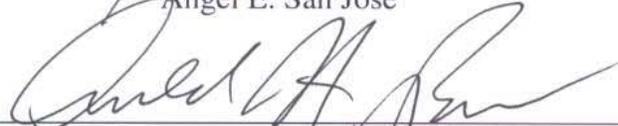
from the

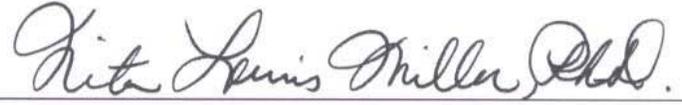
**NAVAL POSTGRADUATE SCHOOL  
September 2001**

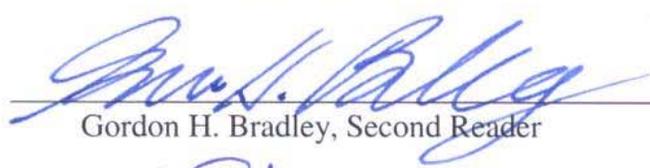
Author:

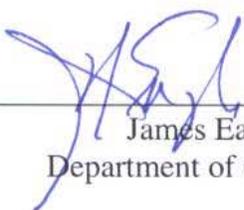
  
Angel E. San Jose

Approved by:

  
Arnold H. Buss, Thesis Advisor

  
Nita Miller, Co-Advisor

  
Gordon H. Bradley, Second Reader

  
James Eagle, Chairman  
Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

## **ABSTRACT**

Discrete Event Simulation (DES) is one of the most widely used methodologies for Operations Research (OR) modeling and analysis. However, designing and implementing DES can be a time-consuming and error-prone task. This thesis designed, implemented and evaluated a tool, the Event Graph Graphical Design Tool (EGGDT), to help OR analysts in the design, implementation, and maintenance of DES reducing the development and debugging times.

The Unified Modeling Language was used to document the development of the EGGDT, which was programmed in Java using J2D and Swing. Human Factors techniques were employed to help in the design process and to evaluate the final prototype of the EGGDT.

During the design process, two formative experiments were performed to evaluate the Graphical User Interface design decisions. A final summative experiment was done to test if the potential users consider the tool a useful means to develop OR simulations. Participants of the experiments agreed that tools like the EGGDT are an essential instrument when developing simulations.

THIS PAGE INTENTIONALLY LEFT BLANK

## **DISCLAIMER**

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at risk of the planner.

THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS

<b>I.</b>	<b>INTRODUCTION.....</b>	<b>1</b>
<b>A.</b>	<b>AREA OF RESEARCH .....</b>	<b>1</b>
<b>B.</b>	<b>BACKGROUND .....</b>	<b>1</b>
	<b>1. Simulation Development .....</b>	<b>1</b>
	<b>a. Discrete Event Simulation and Event Graphs .....</b>	<b>1</b>
	<b>b. Simkit.....</b>	<b>2</b>
	<b>2. Human Factors Techniques in Software Development .....</b>	<b>3</b>
	<b>a. Usability.....</b>	<b>3</b>
	<b>b. User-Centered Development.....</b>	<b>4</b>
	<b>c. Prototyping .....</b>	<b>4</b>
	<b>d. Formative and Summative Usability Experiments .....</b>	<b>4</b>
	<b>e. Parallel Development (Application vs. GUI) .....</b>	<b>5</b>
<b>C.</b>	<b>OBJECTIVES AND RESEARCH QUESTIONS .....</b>	<b>5</b>
<b>D.</b>	<b>SCOPE OF THE THESIS AND METHODOLOGY .....</b>	<b>6</b>
	<b>1. UML .....</b>	<b>6</b>
	<b>a. Analysis.....</b>	<b>7</b>
	<b>b. Design.....</b>	<b>7</b>
	<b>2. Iterative Process .....</b>	<b>7</b>
	<b>3. Software Usability .....</b>	<b>8</b>
	<b>4. Component-Based and Pattern Modeling .....</b>	<b>8</b>
<b>E.</b>	<b>CONCLUSION .....</b>	<b>9</b>
<b>II.</b>	<b>EVENT GRAPH EXAMPLE: RELIABILITY SYSTEM.....</b>	<b>11</b>
<b>A.</b>	<b>INTRODUCTION.....</b>	<b>11</b>
<b>B.</b>	<b>DEFINITION OF THE PROBLEM .....</b>	<b>11</b>
<b>C.</b>	<b>VERSION 1: REPAIR SERVICES ARE NOT CONSIDERED.....</b>	<b>12</b>
<b>D.</b>	<b>VERSION 2: REPAIR SERVICES ARE CONSIDERED.....</b>	<b>15</b>
<b>E.</b>	<b>EVENT-LIST .....</b>	<b>16</b>
<b>F.</b>	<b>JAVA SOURCE CODE.....</b>	<b>17</b>
<b>G.</b>	<b>CONCLUSION .....</b>	<b>18</b>
<b>III.</b>	<b>EGGDT SOFTWARE REQUIREMENTS, ANALYSIS AND DESIGN .....</b>	<b>19</b>
<b>A.</b>	<b>INTRODUCTION.....</b>	<b>19</b>
<b>B.</b>	<b>EGGDT REQUIREMENTS .....</b>	<b>19</b>
	<b>1. Overview Statement.....</b>	<b>19</b>
	<b>2. Goals.....</b>	<b>20</b>
	<b>3. Implementation .....</b>	<b>20</b>
	<b>4. Functionality Requirements.....</b>	<b>20</b>
	<b>5. GUI Requirements .....</b>	<b>21</b>
<b>C.</b>	<b>EGGDT ANALYSIS.....</b>	<b>21</b>
	<b>1. Functionality Analysis .....</b>	<b>21</b>
	<b>a. Use-Cases .....</b>	<b>21</b>
	<b>b. Conceptual Model .....</b>	<b>24</b>
	<b>2. Task Analysis.....</b>	<b>24</b>

	a.	<i>Use-Cases</i> .....	24
	b.	<i>Conceptual Model</i> .....	26
D.		EGGDT DESIGN.....	27
	1.	System Method “Create Edge” .....	27
	2.	System Method “Create Edge Figure” .....	28
	3.	Mouse Control Behavior .....	29
E.		CONCLUSION .....	30
IV.		<b>FIRST EXPERIMENT: USER RESPONSE TO THE PROPOSED TOOL</b>	
		<b>GRAPHICAL USER INTERFACE (GUI) DESIGN</b> .....	33
A.		INTRODUCTION.....	33
B.		PURPOSE OF THE EXPERIMENT .....	33
C.		DESIGN OF THE EXPERIMENT .....	33
	1.	Participants.....	34
	2.	Tasks to Perform.....	34
D.		DESIGN OF THE PROTOTYPE .....	34
E.		EXPERIMENTAL PROTOCOL .....	36
	1.	Pilot Testing.....	36
	2.	Evaluation Sessions.....	36
F.		STATISTICAL ANALYSIS .....	37
	1.	Usability Attribute “Learnability” .....	38
	2.	Usability Attribute “First Impression” .....	39
	3.	Qualitative Data .....	42
	a.	<i>Buttons and Menus</i> .....	42
	b.	<i>Modality</i> .....	42
	c.	<i>Edges</i> .....	42
	d.	<i>Text Editing</i> .....	43
G.		LESSONS LEARNED .....	43
H.		CONCLUSION .....	43
V.		<b>SECOND EXPERIMENT: ARC CONSTRUCTION</b> .....	45
A.		INTRODUCTION.....	45
B.		PURPOSE OF THE EXPERIMENT .....	45
C.		DESIGN OF THE EXPERIMENT .....	45
	1.	Internal Validity .....	46
	a.	<i>History</i> .....	46
	b.	<i>Selection</i> .....	46
	2.	External Validity .....	46
	3.	Conclusion Validity.....	46
	4.	Statistical Validity .....	47
D.		DESIGN OF THE PROTOTYPE .....	47
E.		EXPERIMENTAL PROTOCOL .....	48
F.		STATISTICAL ANALYSIS .....	51
	1.	Difference in Performance Time .....	51
	2.	Linear Model to Explain Time Difference.....	52
	3.	User Preference .....	53

4.	Relationship Between Preference, Performance Time and Treatment Order.....	54
G.	CONCLUSION .....	55
VI.	FINAL EXPERIMENT: USERS' JUDGMENTS TOWARD GRAPHICAL DESIGN TOOLS .....	57
A.	INTRODUCTION.....	57
B.	PURPOSE OF THE EXPERIMENT .....	57
C.	DESIGN OF THE EXPERIMENT .....	57
1.	Explanatory Variables.....	57
2.	Participants.....	58
D.	PROTOTYPE DESIGN .....	58
E.	EXPERIMENTAL PROTOCOL .....	62
F.	STATISTICAL ANALYSIS .....	63
1.	Computer Proficiency Questions.....	65
2.	Demographical Questions .....	66
G.	CONCLUSION .....	67
VII.	CONCLUSIONS AND RECOMMENDATIONS.....	69
A.	CONCLUSIONS .....	69
B.	RECOMMENDATIONS.....	71
	APPENDIX A. EG ELEMENTS .....	73
A.	EG ELEMENTS.....	73
B.	EG EXAMPLE: QUEUING SYSTEM.....	75
	APPENDIX B. EGGDT REQUIREMENTS.....	77
A.	FUNCTIONALITY REQUIREMENTS.....	77
B.	GUI REQUIREMENTS (TASK ANALYSIS) .....	81
	APPENDIX C. FIRST EXPERIMENT FORMULARIES AND DATA .....	85
A.	TASK LIST (PARTICIPANT'S VERSION) .....	85
B.	TASK LIST (EVALUATOR'S VERSION).....	86
C.	INFORMED CONSENT FORM.....	87
D.	SESSION INSTRUCTIONS .....	87
E.	QUESTIONNAIRE FOR THE USER INTERFACE SATISFACTION.....	88
1.	Overall Reaction to the Event Graph Design Tool .....	89
2.	Screen .....	89
3.	Terminology and System Information .....	89
4.	Learning.....	89
5.	System Capabilities.....	89
F.	USABILITY SPECIFICATION TABLE .....	91
	APPENDIX D. SECOND EXPERIMENT FORMS AND DATA.....	93
A.	PARTICIPANT CONSENT FORM .....	93
B.	MINIMAL RISK CONSENT STATEMENT .....	94
C.	PRIVACY ACT STATEMENT.....	95
D.	COMPUTER PROFICIENCY SURVEY.....	96

<b>E.</b>	<b>DATA .....</b>	<b>98</b>
<b>APPENDIX E.</b>	<b>FINAL EXPERIMENT FORMS AND DATA .....</b>	<b>101</b>
<b>A.</b>	<b>PARTICIPANT CONSENT FORM .....</b>	<b>101</b>
<b>B.</b>	<b>MINIMAL RISK CONSENT STATEMENT .....</b>	<b>102</b>
<b>C.</b>	<b>PRIVACY ACT STATEMENT.....</b>	<b>103</b>
<b>D.</b>	<b>COMPUTER PROFICIENCY SURVEY.....</b>	<b>104</b>
<b>E.</b>	<b>DATA .....</b>	<b>106</b>
<b>F.</b>	<b>FIELDS EXPLANATION .....</b>	<b>107</b>
<b>LIST OF REFERENCES.....</b>		<b>109</b>
<b>INITIAL DISTRIBUTION LIST .....</b>		<b>111</b>

## LIST OF FIGURES

Figure I-1	Parallel System Development.....	5
Figure I-2	UML Collaboration Diagram. Create Edge.....	9
Figure II-1	System A.....	11
Figure II-2	Block Diagram of System A.....	12
Figure II-3	EG for System A (Version 1).....	12
Figure II-4	Plot of the Exact Survival Function vs. the Simulation Output Values for System A (Version 1).....	14
Figure II-5	EG for System A (Version 2:Repair services considered).....	15
Figure III-1	UML-UCD. Basic EGGDT Functionality.....	22
Figure III-2	UML-CM Diagram. Basic EGGDT Model.....	24
Figure III-3	UML-UCD. EGGDT GUI Functionality.....	25
Figure III-4	UML-CM Diagram. EGGDT Interface Model.....	27
Figure III-5	UML-Cold. Create Edge.....	28
Figure III-6	UML-Cold. Create Edge Figure.....	29
Figure III-7	UML-SMD. Mouse Controller.....	30
Figure IV-1	Snapshot of the EGGDT Prototype.....	35
Figure IV-2	Box-plot of the Times for Benchmark Task.....	38
Figure IV-3	Differences in Performing Both Tasks of Creating Edges.....	39
Figure IV-4	Participants’ Responses for Overall Reaction Questions.....	41
Figure IV-5	Participants’ Responses for “Experience Taken into Consideration”.....	42
Figure V-1	The Direct or Free Method (F Method).....	47
Figure V-3	Welcome Window.....	49
Figure V-4	Training Window.....	49
Figure V-5	Task Window.....	50
Figure V-6	Result window.....	51
Figure V-7	Relationship Between Test Times per Participant.....	52
Figure V-8	Preference of the Participants (VH or F).....	53
Figure V-9	Individual Performance Time Differences Conditioned by the Preference (VH or F).....	54
Figure VI-1	Initial Choice Dialog of the EGGDT.....	58
Figure VI-2	Properties Window.....	59
Figure VI-3	Snapshot of the EG for the “Arrival Model”.....	59
Figure VI-4	Snapshot of the Java Class Generated Code Window for the “Arrival Model”.....	60
Figure VI-5	Snapshot of EG for the “CPU Model”.....	62
Figure VI-6	Participants’ Responses to Questions II.C and II.D.....	63
Figure VI-7	Participants’ responses to questions I.C and II.D.....	66
Figure VI-8	Participants’ Responses to Question “Usefulness of the Tools” Conditioned by Military Branch.....	67

THIS PAGE INTENTIONALLY LEFT BLANK

## LIST OF TABLES

Table I-1	Relationship Between EGs and Simkit.....	3
Table II-1	Example of a Possible Event-List for System A.....	16
Table II-2	Example of a Possible Event-List for System A (After time 10) .....	17
Table III-1	Extended UC Create EG .....	23
Table III-2	Extended UC Create EG Element.....	23
Table III-3	Extended UC Initiate EGGDT .....	26
Table IV-1	Benchmark Times in Seconds.....	37
Table IV-2	Questionnaire Responses .....	40
Table V-1	ANOVA Results of Time to Perform Tasks with F Method vs. Time with VH Method .....	52
Table V-2	Linear Model Summary. Difference of Time Performance Against Order, Graphical Experience And Computer level .....	53
Table V-3	Logistic Regression Summary. Preference Against Order and Performance Times .....	55
Table VI-1	Java Class Generated from the EG for the “Arrivals Model”.....	61
Table VI-2	Summary Statistics for Questions II.C and II.D. ....	65

THIS PAGE INTENTIONALLY LEFT BLANK

## ACKNOWLEDGMENTS

To my wife Maria and my daughter Marta, may you forgive my absences.  
Os eché tanto de menos!

I want to thank my professors Arnie Buss, Gordon Bradley and Nita Miller for their help in writing this paper, I hope our partnership continue in the future.

Many people have been involved in this little project and have helped me in one way or another. My classmates who worked with me in several class projects related with my thesis: Gokhan Ozkan, Mark Harrington, Raj Mohan, James Campbell, Parke Paulson, and Erik Hovda. My Portuguese brother Paulo Silva who helped me just being my friend and my Brazilian brother Ze Paes who stood with me these two years.

While I was finishing this thesis, the tragic events of the World Trade Center and the Pentagon occurred. They may remind us, technical people, how fragile our knowledge is in this bizarre world. I want to dedicate this little piece of my life to those who lost their lives, their families, their friends, or their hope in the tragic morning of September 11, 2001.

THIS PAGE INTENTIONALLY LEFT BLANK

## EXECUTIVE SUMMARY

In a constantly changing world, decision-makers face problems with high levels of uncertainty. Operations Research (OR) analysts help by scientifically studying the different alternatives for each problem and proposing solutions. Among the many techniques used by OR analysts, simulation is one of the most important.

Easy-to-use applications to design simulations models, like Arena, can be found. These applications claim they provide a visual environment to model simulation and a press-a-button analysis of the simulation outcomes. However, these applications cannot solve all kinds of problems and, even worse, they are not scalable.

A more flexible technique to implement simulation models is to use Event Graphs to describe the Discrete Event behavior of the systems and to generate a computer program based on this model. This approach has the advantage of being both flexible and scalable. Many systems may be modeled using discrete-event simulation, including production systems, transport systems, weapons systems, or military operations.

During this thesis research an Event Graph Graphical Design Tool (EGGDT) was developed to help OR analysts in designing EGs. The EGGDT provides a computer environment to draw EG elements, to define simulation variables and to generate the skeleton of the Java source code of the simulation using Simkit. The EGGDT can reduce simulation development and debugging times.

For the Analysis and Design (A&D) phases of the EGGDT, the Unified Modeling Language (UML) was used. The UML allowed the depiction of the A&D decisions and was used to document the application.

Since the EGGDT is a graphical application to be used by OR analysts, it was necessary to consider the Human Factors involved in its development. The User-Centered approach, used to develop the EGGDT, is when the final user is considered and consulted for every major decision in every phase of the development of the system. The final users of the EGGDT are OR students at the Naval Postgraduate School (NPS). Three

experiments were performed as a part of the effort to involve the user in the development of the Graphical User Interface (GUI) of the EGGDT.

The first experiment tested the users' opinion about the initial design of the GUI of the EGGDT. The prototype used for the experiment implemented all the interface services required to the EGGDT but did not provided any functionality. The reaction of the participants in the experiment was positive, so the general layout of the GUI was considered adequate and used in following prototypes and in the final version of the EGGDT. However, the participants expressed difficulties in creating the edges of the EGs; in an EG, circles represent events, while arcs represent edges.

The second experiment focused on the edge construction. Two methods were proposed and considered by the users. Participants in the experiment expressed their preference for the "Free Method" that allowed the drawing of edges by specifying the end events and any number of middle points. The experiment also showed that participants did their tasks faster with the "Free Method" than they did with the other one. The "Free Method" was then selected to implement the EGGDT.

To test the claim that simulation-design tools help OR analysts who are developing simulation models, a final experiment was performed. Participants in the experiment agreed that tools like the EGGDT could improve their satisfaction in developing simulations; they also unanimously stated their preference for using these tools over manual methods. In addition, they expressed their opinion that these tools could be useful in a wide range of OR applications. Finally, they stated that tools like the EGGDT encouraged them to have more confidence when facing simulation projects.

In conclusion, this investigation shows that OR students at the NPS consider the EGGDT and similar tools an essential instrument when developing simulations. The principal investigator also believes that this statement can be generalized to all OR analysts and to others involved in modeling and simulation projects and studies.

## LIST OF ACRONYMS

A&D.....	Analysis and Design
CASE .....	Computer Aided Software Engineering
DES .....	Discrete Event Simulation
EG .....	Event Graph
EGGDT .....	Event Graph Graphical Design Tool
GUI .....	Graphical User Interface
HTML .....	Hyper Text Markup Language
IDE .....	Integrated Development Environment
JWARS .....	Joint Warfare System
MTTF .....	Mean Time To Failure
MOP .....	Measurement Of Performance
NPS .....	Naval Postgraduate School
OR.....	Operations Research
Simkit.....	Simulation Kit
UC.....	Use-Case
UML.....	Unified Modeling Language
UML-CD.....	UML Class Diagram
UML-CM .....	UML Conceptual Model
UML-CoID.....	UML Collaboration Diagram
UML-SD .....	UML Sequence Diagram
UML-SMD.....	UML State Machine Diagram
UML-UC.....	UML Use-Case Diagram
XML.....	Extensible Markup Language

THIS PAGE INTENTIONALLY LEFT BLANK

# I. INTRODUCTION

## A. AREA OF RESEARCH

One of the major tools for Operations Research (OR) is simulation. This technique is used in many applications, such as reliability, acquisition planning, transportation, and system assessment. The value of simulation is evident by the amount of money the US Department of Defense is expending on the Joint Warfare System (JWARS) project. JWARS will be a campaign-level simulation model providing a simulation of joint warfare that will support “operational planning and execution, force assessment studies, system trade analyses, and concept and doctrine development” [MAX00].

Analysts in OR implement simulations on computers using computer software applications or programming languages. Consequently, the quality of the simulation – and of the whole OR project as well – depends on the quality of its software. Computer Aided Software Engineering (CASE) tools can help software engineers to develop computer applications, they can also help OR analysts to model simulations. In designing CASE tools, the area of Human Factors must be considered. These tools are intended for use by OR analysts; therefore, the productivity achieved depends on their usability and on the level at which they are accepted by the user. This thesis designs and implements one type of CASE tool and evaluates the human factors acceptability of this tool.

## B. BACKGROUND

### 1. Simulation Development

#### *a. Discrete Event Simulation and Event Graphs*

The Discrete-Event-Simulation (DES) paradigm is the preferred framework for OR simulations. Many systems in OR studies can be modeled as discrete-event systems. For example, DES can be used to model production systems, transport systems, weapons systems, or military operations. Discrete-event systems are those

whose state changes over time. Such systems are defined by piecewise constant state trajectories.

Event Graphs (EG), introduced by Schruben [SCH83], are practical means of representing DES models. These minimalist graphs allow depiction of the behavior of the system. Only four elements exist in an EG (see [BUS01]):

- Simulation parameters represent the characteristics of the system, for example, the random variable “Arrival Time” or the system’s constant “Maximum Number of Servers”.
- State variables convey the state of the system. DES systems are studied by tracking the changes of the values of these variables. Some examples of state variables are “Number of Available Servers” over time or the “Total Number of Customer Served”.
- Events represent a particular state transition in the system. When an event is fired by the simulation’s controller, its actions are executed and the events specified by its outgoing edges can be scheduled (an event can only be scheduled by another event). Examples of events are the “Arrival” event or the “Start Service” event.
- Scheduling Edges represent the scheduling of one event to distinguish it from another. This scheduling can be restricted by a delay time and a condition. For example, the event “Arrival” schedules the event “Start Service” if the state variable “Number of Available Servers” is greater than zero, that is, if a server is available.

The simulation time and event flow are governed by the event list. Using this list, the simulation’s controller determines which event to fire.

### ***b. Simkit***

Simkit was first implemented in a master’s thesis by LT Kirk Stork, US Navy [STO96] and has subsequently been reviewed and extended by Professor Arnold Buss. Since 1997, Simkit has been used to teach DES in the OR Department at NPS. Simkit is a simulation engine implemented as a Java library supporting the realization of DES models. Simkit provides methods to run iterations, control parameters of experiments, and gather output data.

EGs and Simkit are utilized to graphically design and implement DES models. They have a straightforward correspondence; that is, for every element in an EG there is specific Java code that Simkit interprets.

Elements of an EG and Simkit associated code are given in Table I-1. See Appendix A for details and a simple example

<b>EG</b>	<b>Simkit</b>
State Variables	Class instance variable
Simulation Parameters	Class instance variable
Events	Class “do” method
Event Parameters	Parameter of a “do” method
Event Actions	Code lines of a “do” method
Scheduling Edges	“waitDelay” call
Canceling edges	“interrupt” call
Edge Delay Times	Time argument of “waitDelay” call
Edge Conditions	“if condition block”, wrapping a “waitDelay” or “interrupt” call.
Edge Arguments.	“waitDelay” or “interrupt” call arguments

Table I-1 Relationship Between EGs and Simkit

## 2. Human Factors Techniques in Software Development

### a. Usability

Usability is defined as the property of an item of being suitable and convenient to use. In [SHN92], Shneiderman defines five components of computer usability as

- Ease of learning
- High speed of user task performance
- Low user error rate
- Subjective user satisfaction
- User retention over time

For the users of an interactive system, “the interface is the application”. For that reason, developing good interfaces to improve user acceptance of the product is

vital. Clearly, usability does not mean just a window interface since many windows applications exhibit a very low level of usability.

The following points describe some techniques that can be used to improve usability of software applications.

***b. User-Centered Development***

User-Centered seeks involvement of the “final user” in the development process from the first stages of the conception of the system. Many systems have been developed without getting inputs from the “final user”; these systems usually have very poor usability. A “final user” is understood to be the one who eventually utilizes the application.

Two errors are possible when developing applications involving humans. The first occurs when the developer is actually one of the potential users. The developer should not be considered a “final user”; therefore, the application’s usability must not be assessed by the developer, unless the developer is the only potential user.

The second problem appears when the user selected to assess usability does belong to the organization but is not one of those who will eventually use the tool. As an illustration, consider an application that deals with bank accounts: a final user could be a bank teller but not a bank executive, even though the executive could be the client representative.

***c. Prototyping***

Prototyping is the best solution to build a rapid model of the Graphical User Interface (GUI). Prototypes do not provide functionality but show how this functionality is made accessible. Prototypes are usually done in specialized languages or tools. The final product uses the lessons learned from the experiments.

***d. Formative and Summative Usability Experiments***

Formative usability experiments are employed to get inputs from the user during the development. These experiments are performed using prototypes of the entire application or other prototypes built to address particular issues.

Summative experiments are performed with the final application, or an intermediate deliverable product. The intent of these experiments is to refine the final product and evaluate its usability level.

*e. Parallel Development (Application vs. GUI)*

The separation of functionality and interface development provides a way to obtain a more workable system. Implementing this approach, the interface decisions are separated from the functionality decisions, leading to a more decoupled system. A better interface can be achieved since it is not as tightly constrained by functionality design decisions.

Figure I-1 is based on [HIX93, p. 115]. This graph depicts the dual process of building the application software in parallel with the interface software.

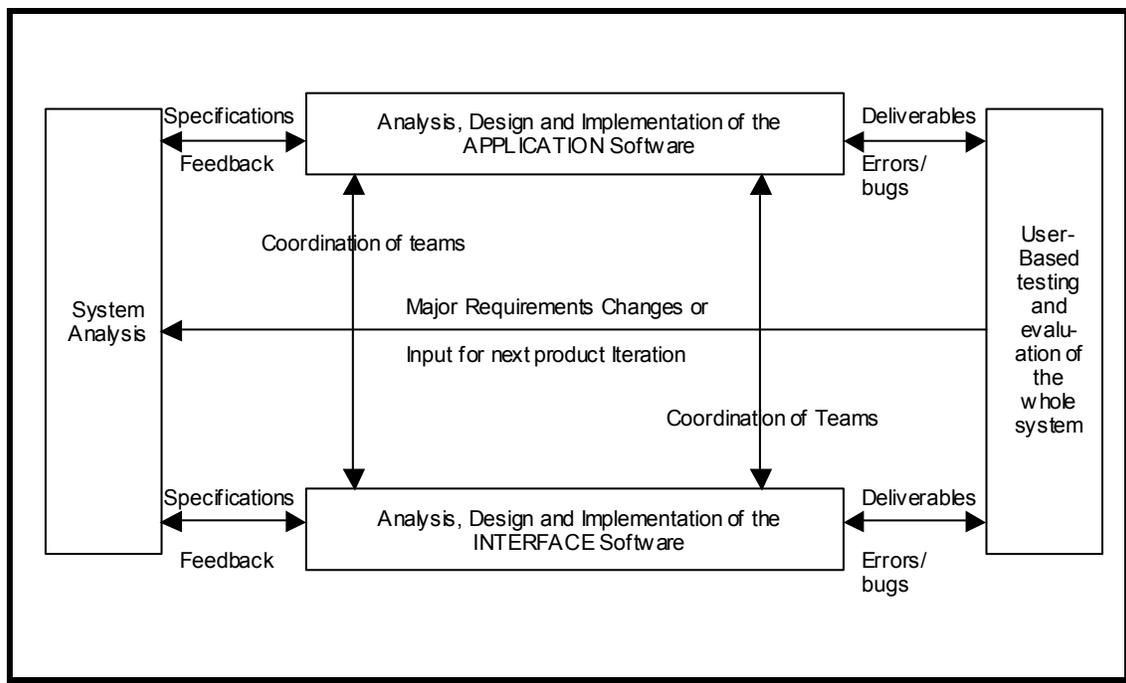


Figure I-1 Parallel System Development

**C. OBJECTIVES AND RESEARCH QUESTIONS**

The whole process of designing and translating EGs into code is currently performed manually. This procedure produces high error rates, long development times, difficult to maintain software, and user dissatisfaction. An Integrated Development

Environment (IDE) for simulation projects can automate this process. An IDE tool is a suite of applications whose purpose is to analyze, design, implement and debug simulation models, plan and execute experiments, and finally, to study the results.

An important component of an IDE for simulation projects is a graphical design tool. The Event Graph Graphical Design Tool (EGGDT) developed here allows the user to construct EGs and to automatically generate code skeletons in Java using Simkit. The decision to build the EGGDT was undertaken because no tool for designing EGs and generating Java code using Simkit was found. The objective of this research is to use Human Factors techniques and guidelines to model and build an EGGDT and to measure its effectiveness.

The hypothesis stated here is that the use of these kind of tools, when they are properly designed by means of user-centered approaches, improves OR analyst satisfaction and helps to enhance OR productivity. To test this hypothesis, reactions of the potential users in front of one of these tools were observed and measured.

#### **D. SCOPE OF THE THESIS AND METHODOLOGY**

The methodology used to develop the EGGDT was based on the following principles:

- Use the Unified Modeling Language (UML) as the Analysis and Design (A&D) graphical modeling language.
- Implement an iterative process based on Use-Cases (UC).
- Plan the development pursuing software usability.
- Design the application as a component-based model found by utilizing design patterns.

##### **1. UML**

The A&D graphical tool, UML<sup>1</sup>, is used to develop software systems. This instrument communicates and depicts the application's structure helping to set architectural alternatives and to justify decisions of A&D. Finally, UML improves

---

<sup>1</sup> For more information about UML see references [LAR98], [FOW00], [RUM99]

maintainability by providing a means to generate the necessary documentation. Document generation is the most time-consuming task in software development.

To conduct this project certain parts were analyzed and designed using UML. Since, in this project, the EGGDT represents the work of only one developer, documentation in this paper is not comprehensive.

The UML artifacts used to perform the A&D of the EGGDT were

*a. Analysis*

- Use-Case Diagrams (UML-UC) depicting decompositions of the functionalities required in the system. The external entities playing a role in the system can also be included.
- Conceptual Model Diagrams (UML-CM) showing the high-level problem-domain object-decomposition of the system.
- Sequence Diagrams (UML-SD) illustrating the high-level operation calls between the external entities and the system.

*b. Design*

- Class Diagrams (UML-CD) explaining the design-level class-decomposition of the system.
- Collaboration Diagrams (UML-CollD) describing the details of every system's method.
- State Diagrams (UML-StD) showing the behavior of control classes, such as mouse controller classes.

**2. Iterative Process**

The purpose of adopting a software development process is to establish a workflow that specifies a series of steps to follow. Following a development process guarantees a certain discipline and order. A good example of a development process is the iterative process<sup>2</sup>; this is a good choice for this particular project (and for many others). In an iterative process, the overall project is built in successive iterations. From each iteration a product is released, with subsequent stages built upon the ancestor products.

During this research only one iteration of the EGGDT was performed. A Use-Case approach was used to determine what to develop in this iteration. The more critical

---

<sup>2</sup> For more information about software development process see for example [FOW00], [COK97] or [JAB99].

and risky UCs were chosen to ensure that the core functionality of the EGGDT and those components that compromised the whole feasibility of the product were developed first.

### **3. Software Usability**

Usability was pursued by incorporating several techniques. One technique, user-centered development, in particular, was achieved via a two-flow development, one for the application software and another for the interface software. Formative experiments were performed to assess the intermediate products and compare design alternatives. A summative experiment was performed to determine the opinions of the potential users about these kinds of tools.

### **4. Component-Based and Pattern Modeling**

The use of patterns is essential at the design level. Design patterns were introduced by [GAM95]; however, the version used during this research is in [LAR98]. Patterns provide known structures and lessons learned, that is, they are the means by which developers can take advantage of the wisdom and experience of other developers. During the software design, patterns such as “Controller”, “Expert” or “Creator” were used.

Figure I-2 shows the UML collaboration diagram (UML-Cold) used to design the functionality of creating edges with the EGGDT. First, consider that this UML-CD only deals with the problem of creating an edge at the functionality level not at the interface level. This model makes no reference to arcs or mouse or any other graphical element.

The rectangles with a bent corner such as those used in Figure I-2 refer to the patterns that assign the operations to the different classes represented by shaded rectangles. The operations names are written over the lines that join the classes; the arrows indicate the direction of the call. For example, `addOutgoingEdge()` is a method of the class `Event` called from a object of the class `GUIController`. This method has been assigned to `Event` because `Event` is the only one that knows about its own outgoing edges.

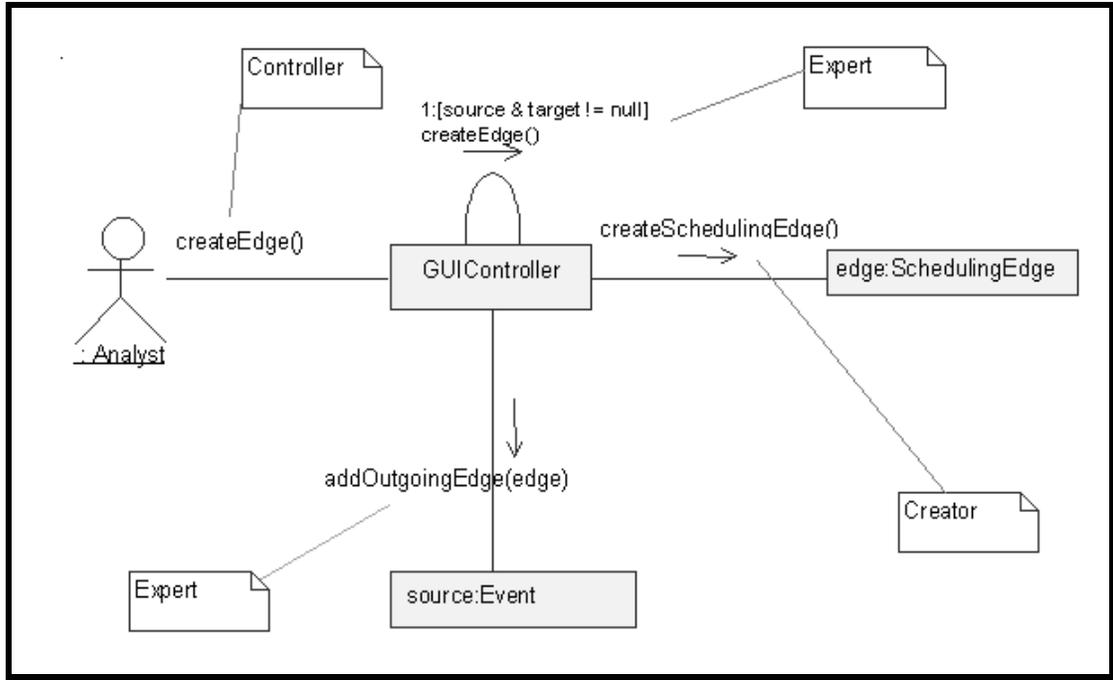


Figure I-2 UML Collaboration Diagram. Create Edge

In [BUS00] a description of Component-based modeling is found. To model the EGGDT this technology was used as well as the traditional Object Oriented (OO) approach. In a component-based system the architecture is based in interfaces and the use of the listener-pattern [BUS00, par. 5]; in contrast to this, in classical OO design the architecture is based on inheritance.

## E. CONCLUSION

Simulation is a major tool to analyze OR problems. Simulations are software applications that can be developed with the help of CASE tools. The use of these tools is intended to improve user satisfaction when creating simulation models. However, since no simulation-modeling tool based on EGs and Simkit was commercially available, the EGGDT was developed.

The EGGDT is an example of a CASE tool that allows designing simulations with EGs and facilitates their implementation in Java using Simkit. To develop the EGGDT, human factors have been considered; user-centered techniques and formative experiments have been used to ensure usability.

Chapter II presents an example of a DES model of a system depicted with EGs and codified in Java using Simkit. Chapter III details the EGGDT requirements and the A&D decisions. The next two Chapters, IV and V, describe the formative experiments performed to ensure usability during development and the summative experiment carried out to evaluate the subjective increase in user satisfaction using these kinds of tools. Chapter VI describes the summative experiment carried out to test whether the simulation-design tools improve OR analyst satisfaction when developing simulations.

## II. EVENT GRAPH EXAMPLE: RELIABILITY SYSTEM

### A. INTRODUCTION

This chapter presents an Operations Research (OR) problem that can be modeled as a Discrete Event Simulation (DES) using Event Graphs (EG). This example shows the utility of DES models and EGs to solve OR problems. The EGs in this chapter were created with the Event Graph Graphical Design Tool (EGGDT) Version 0.3. The skeleton of the Java code was generated by the EGGDT from the EG model.

### B. DEFINITION OF THE PROBLEM

Consider the system in Figure II-1. Computer1 sends information to Computer2 across a network. The link between Computer1 and Computer2 depends on three nodes inside the network (Nodes 1, 2 and 3) that have specific failure time distributions. These distributions can be empirical but are known.

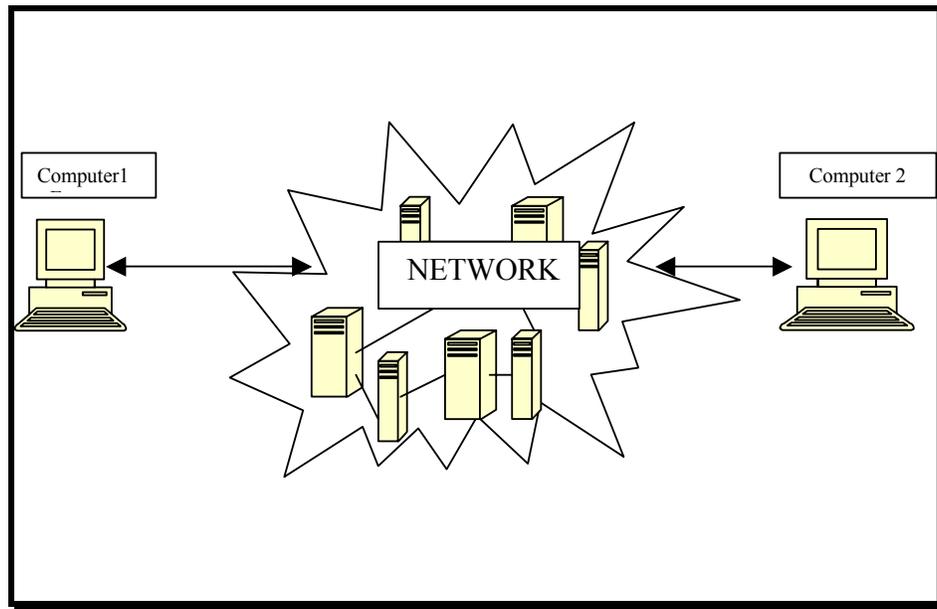


Figure II-1 System A

Figure II-2 shows the block diagram of these three nodes considering only the connections that affect the link between Computer1 and Computer2. This block diagram approach can be used to study reliability issues. For this example, the Measurement Of

Performance (MOP) is the Mean Time To Failure (MTTF) of the link between Computer1 and Computer2.

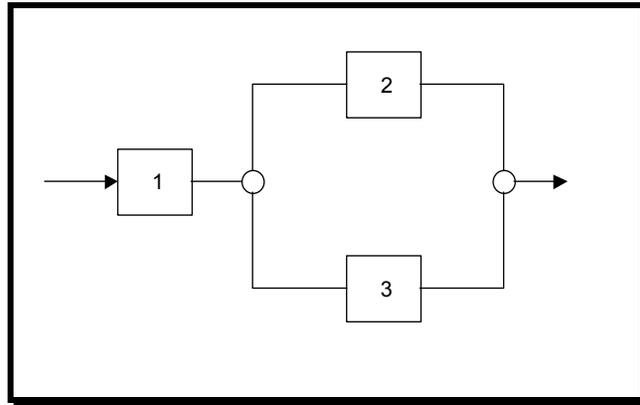


Figure II-2 Block Diagram of System A

**C. VERSION 1: REPAIR SERVICES ARE NOT CONSIDERED**

A DES model of System A is shown in the EG of Figure II-3. This model does not consider repair services of the nodes, so when one component fails, it is not put into service again.

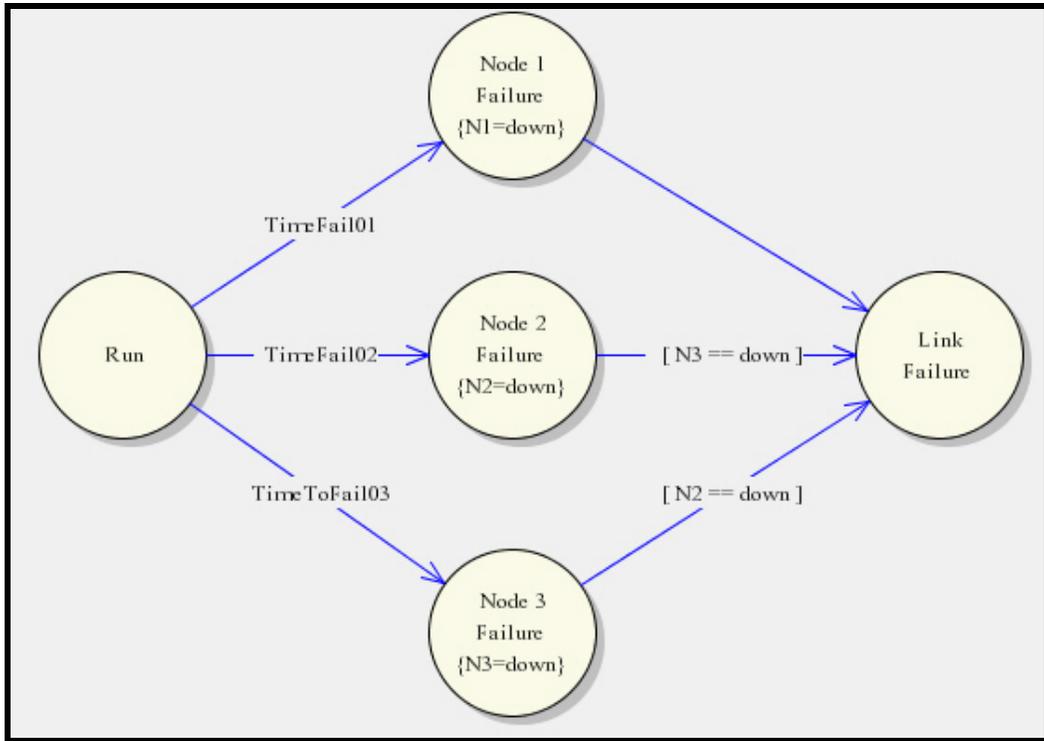


Figure II-3 EG for System A (Version 1)

The EG in Figure II-3 starts with the “Run” event that then schedules the three “Nodes Failure” events. The “Node 1 Failure” event sets the state variable “N1” to “down” to indicate that the node is not working. According to the configuration depicted in the block diagram of Figure II-2, a failure in “Node 1” causes the link to fail; consequently, “Node 1 Failure” event schedules a “Link Failure” event to be scheduled with zero delay time.

The behaviors of the events “Node 2 Failure” and “Node 3 Failure” are similar. Each sets its state variable to “down” and schedules a “Link Failure” event if the other component in parallel (N2 or N3) has already failed.

To calculate the MTTF, a number of replications have to be performed recording the time the “Link Failure” event occur for every replication. The mean of these times can be used as an estimator to approximate the MTTF of the link if the number of replications is large. Similarly, a histogram of the times when the “Link Failure” event occurred for every replication can be an estimate of the distribution of the link failures times.

This simple problem can solved using the mathematics provided by the reliability literature. In particular, the structure function of System A is given in Equation–II-1

$$\text{Equation–II-1} \quad \Phi(X) = \min [ X_1, \max \{ X_2, X_3 \} ]$$

$X_1$ ,  $X_2$  and  $X_3$  are binary variables that equal one if the component is working and zero if the component fails.  $\Phi(X)$  has two possible values 1 or 0 depending on whether the link is up or not. Based on Equation–II-1, the survival function ( $S(t)$ ) for this system is expressed in Equation–II-2.

$$\text{Equation–II-2} \quad S(t) = S_1(t) [ 1 - \{ 1 - S_2(t) \} \{ 1 - S_3(t) \} ]$$

Where  $S(t)$  is the probability that the link is up in time  $t$ ; the terms  $S_1(t)$ ,  $S_2(t)$  and  $S_3(t)$  are the particular survival functions of the components. To get the MTTF the integral of  $S(t)$  from 0 to infinity is used. However, the integral can be difficult or impossible to evaluate analytically, so numerical methods have to be used. In addition, this approach does not consider repair services.

As an illustration of the validity of the simulation approach, let consider that the survival functions  $S_1(t)$ ,  $S_2(t)$  and  $S_3(t)$  present exponential distributions with mean time between failures of 2.5 ,2.0 and 2.2 months. Figure II-4 shows a probability plot of the exact survival function  $S(t)$  and the values from the simulation. The simulation was program in Java using Simkit based in the EG of Figure II-3. The model was run 500 times to get the plot in Figure II-4. The departure of the simulation curve from the exact curve in Figure II-4 is very small.

The values obtained were

Exact MTTF = 1.54

Simulation MTTF = 1.57

95% CI if the simulation MTTF = [1.45, 1.69]

The exact MTTF is included in the 95% CI for the simulation MTTF. As the number of iterations increases, the simulation MTTF approaches the exact MTTF. For example at 10,000 iterations, the simulation MTTF is 1.56.

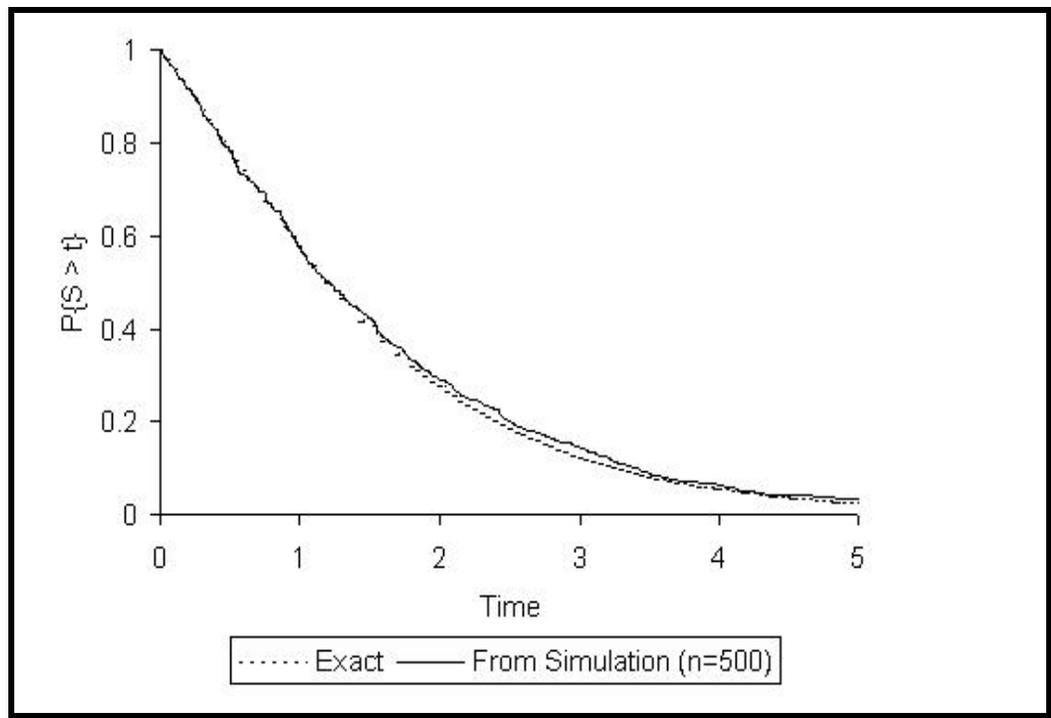


Figure II-4 Plot of the Exact Survival Function vs. the Simulation Output Values for System A (Version 1)

**D. VERSION 2: REPAIR SERVICES ARE CONSIDERED**

When nodes can be repaired and put back to service, the mathematical model of the previous section is inadequate to capture the system behavior. However, the simulation model can be easily adjusted to accommodate this situation.

The assumptions for this model are:

- When one node fails, it is repaired in a random time whose probability distribution is known.
- The nodes continue working even if the link fails; this is a reasonable assumption because these nodes are not only serving this link so they are kept working even if the link is broken.
- Repair facilities exist to serve every node independently, i.e., repair queues do not exist. The distributions of the repair times are known.

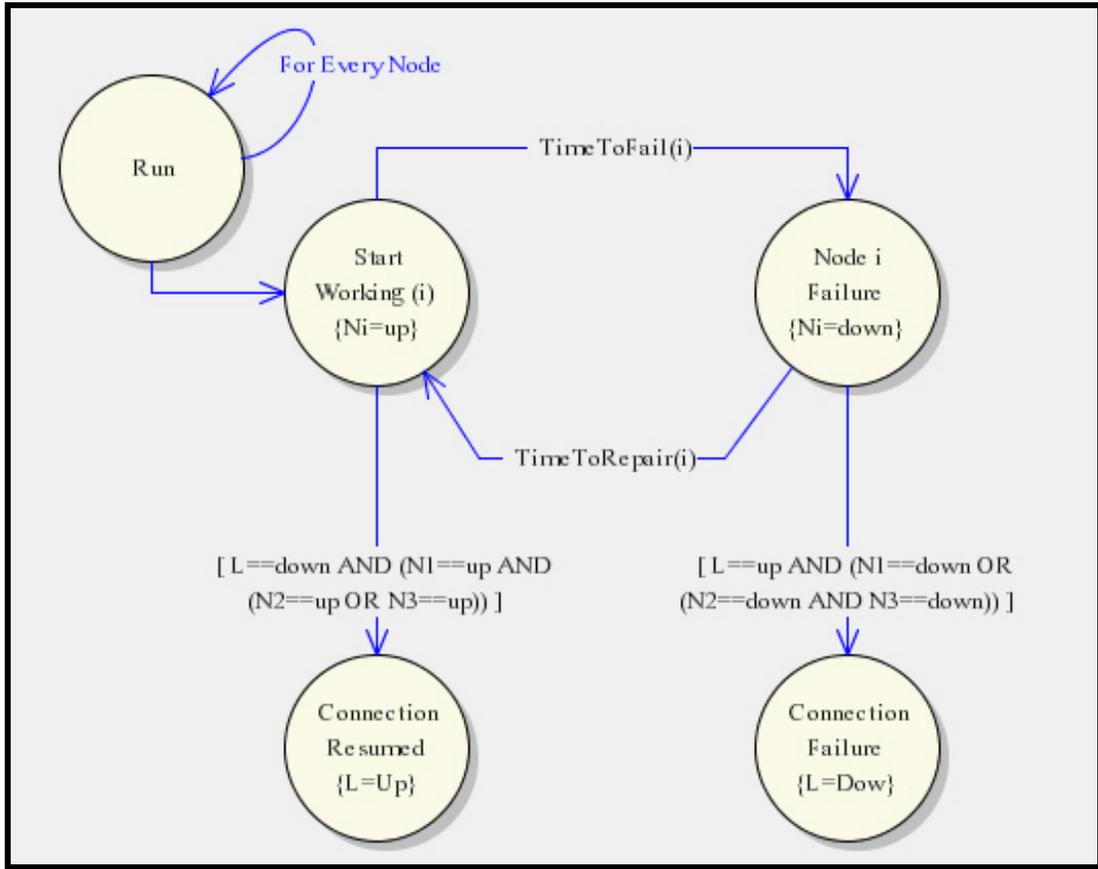


Figure II-5 EG for System A (Version 2:Repair services considered)

The EG of Figure II-5 commences when the “Run” event schedules all the “Start Working” events. These events set the state variables “N1”, “N2” and “N3” to “up” to indicate that the nodes are working. When the necessary nodes are up(i.e., “Node 1” and “Node 2” or “Node 3”), the “Connection Resumed” event is fired; this event sets the state variable L, which represents the state of the link, to “up”.

Each “Start Working” event also schedules its own “Node Failure” event with delays determined by the corresponding “Time to Fail” random variable. “Node Failure” events schedule their own “Start Working” events using their “Time to Repair” random variables. “Connection Failure” is scheduled by one of the “Node Failure” events when the link is up and “Node 1” is down or “Node 2” and “Node 3” are both down.

This cycle continues until the simulation is stopped based on time or number of failures. These details depend on the design of the experiments to be performed. To obtain the MTTF the same method explained for Version 1 can be used. For this particular problem, the analytical solution is difficult or impossible because of the distributions of the times to failure and repair.

#### E. EVENT-LIST

To illustrate how the event-list works using the DES model of Version 2 Figure II-5, suppose the “Node 1” and “Node 3” are currently working and “Node 2” is being repaired. A possible content for the event-list is detailed in Table II-1.

Time	Event	Parameters
10	Node 1 Failure	i = 1
13	Node 3 Failure	i = 3
16	Start Working 2	i = 2

Table II-1 Example of a Possible Event-List for System A

Table II-2 shows the event-list in time 10 after being modified by execution of the “Node 1 Failure” event.

Time	Event	Parameters
10	Connection Failure	
13	Node 3 Failure	i = 3
14	Start Working 1	i = 1
16	Start Working 2	i = 2

Table II-2 Example of a Possible Event-List for System A (After time 10)

## F. JAVA SOURCE CODE

For the sake of clarity of the example, the EG in Figure II-5 was depicted in a high level manner. The Java code described below is more detailed but is based on this EG. The Java source code from this model is encapsulated in a class that extends the Simulation Kit (Simkit) simulation entity abstract class.

```
import simkit.*;
public class SimpleReliabilityModel02 extends SimEntityBase {
```

The main code of this class is explained below. The state variables and simulation parameters are declared as private instance variables of the class

```
    // State Variables
    private boolean L = false ; // true = up & false = down
    private Boolean[] N = new Boolean[3]; // set all to false

    // Simulation Variables
    private randomVariate[] timeToFail = new randomVariate[3];
```

For every event, a “do” method is implemented.

```
    /**
     */
    public void doStartWorking(int i) {
        N[i]= true;
        if(L== false && (N1==true &&(N2==true || N3==true)) ) {
            waitDelay("ConnectionResumed", 0.0);
        }
        waitDelay("NodeFailure" , TimeToFail[i].generate(), i);
    }
    /**
     */
    public void doNodeFailure (int i) {
        N[i]= false;
        waitDelay("StartWorking" , TimeToRepair[i].generate(),i );
        if ( L== true && (N1==false || (N2==false && N3==false))) {
            waitDelay("ConnectionFailure" , 0.0 );
        }
    }
```

```

}
/**
 */
public void doConnectionResumed ( ) {
    L=true;
}
/**
 */
public void doConnectionFailure ( ) {
    L=false;
    // Insert the necessary code to record the time the link
    // has been up.
}

```

The “waitDelay” calls implement the outgoing edges of the event. The “if statements” are a consequence of the edge conditions.

## G. CONCLUSION

This example shows that systems can be modeled using the DES paradigm. DES models can be represented as EGs and from them executable simulations can be obtained. The analytical solutions for a wide range of problems are not available. For example, systems whose behavior involves stochastic processes are very difficult or impossible to abstract as mathematical models; simulation is the only resource in these kinds of situations. DES simulations are also easily expandable and accept a broad diversity of input parameters. In other words, DES simulations are a flexible tool to study the behavior of systems. The next chapter details the requirements of the EGGDT.

### **III. EGGDT SOFTWARE REQUIREMENTS, ANALYSIS AND DESIGN**

#### **A. INTRODUCTION**

The purpose of this chapter is to detail the requirements for the Event Graph Graphical Design Tool (EGGDT) and the more important Analysis and Design (A&D) decisions<sup>3</sup>. The goal of the EGGDT is to allow depiction of Event Graphs (EG). For more information about EGs see [BUS01], [BUS96], [SCH83] or [SCH95].

#### **B. EGGDT REQUIREMENTS**

##### **1. Overview Statement**

The focus of this particular research is on the analysis, design and implementation of a computer tool aimed at supporting simulation design and implementation using Event Graphs (EGs), Java and the Simulation Kit (Simkit).

The intent of the EGGDT tool is to help Operations Research (OR) analysts in the design, implementation, and debugging of simulation software. The system is envisioned as a graphical tool to draw EGs for eventual translation into code thereby reducing errors and obtaining documentation from the models.

An EG and a Simkit application have a straightforward correspondence. For every element in an EG specific Java code exists in the Simkit program. This correspondence between an EG and Simkit code is shown in Table I-1. See Appendix A for details and a simple example.

The functional and interface requirements for the EGGDT have been considered separately. Separation between Graphical User Interface (GUI) and functionality allows for a user-centered development approach. As discussed in the introductory chapter, the user-centered method is essential to meet user expectations.

---

<sup>3</sup> The study contained in this chapter started in two group projects for the IS-3020 and MV-4202 courses. The rest of participants for the IS-3020 project were: LtJG Gokhan Ozkan, Maj Mark Harrington, Maj Raj Mohan. LCdr Paulo Silva was the other participant in the MV-4202 project.

Initially, the EGGDT is aimed at OR students and educators of the Naval Postgraduate School (NPS). Eventually, other users will be OR analysts or those involved with modeling and simulation studies and practices. The users of the EGGDT are assumed to have a background in and be familiar with OR methods and simulation design, particularly EG modeling.

## **2. Goals**

The goal of the EGGDT project is to:

- Provide a friendly GUI to develop EGs.
- Automate the process of generating simulation code from a known EG.
- Provide a tool to document the simulation details.

## **3. Implementation**

The EGGDT was implemented in Java using the Swing and Java 2D (J2D) libraries because Java is a modern, platform-independent object oriented programming language. Additionally, the Swing and J2D libraries offer a wide variety of functionalities for GUIs. As an extra benefit, Java is the Internet programming language; therefore, even though the first version of the EGGDT is a stand-alone computer application, implementing it as an Internet application is a straightforward extension.

Another reason to adopt Java is that Simkit is written in Java, so the communication between the tool and Simkit is smooth. Java virtual machines are available for all major platforms, including MS-Windows, Linux, Free BSD, Mac OS X and commercial Unix. Java virtual machines are also available in handheld devices, such as Palm Pilot. It is projected that by 2002 there will be more Java than C++ developers.

## **4. Functionality Requirements**

Functionality requirements refer to the desired internal behavior of the system with no reference to the way these functionalities are provided to the user. The external behavior is detailed in the interface requirements.

The functionality requirements are broken down in Appendix B.

The most important functionality requisites are:

- Allow the user to create, delete and modify EGs and all its elements (events, edges and simulation variables).
- Generate the skeleton of Java class from the EG.

## 5. GUI Requirements

GUI requirements refer only to the external appearance and behavior of the EGGDT. Clarifying the distinction between functionality and interface entails realizing that even though an EG contains events, edges and simulation variables, its representation can take many forms. For example, an EG can be represented as a picture with circles and arcs, a Java class, a list of the elements in an XML (Extensible Markup Language) file, a text file, an HTML file or in any other form storing the information contained in the EG.

The EGGDT requirements of the GUI specify that the approach chosen represents EGs as pictures containing circles and arcs for the events and edges, and some kind of tabular structure for the simulation variables.

Appendix B specifies the interface functionalities that the EGGDT has to provide.

## C. EGGDT ANALYSIS

During this research only one iteration of the EGGDT project was performed. A Use-Case (UC) approach was followed to select those functionalities that should be implemented; the most essential or risky UCs were selected.

### 1. Functionality Analysis

The functionality analysis describes how the basic functions of the EGGDT have been broken down; no reference to graphical elements is made in this kind of study.

#### a. *Use-Cases*

As discussed in Chapter I, UML UCs have been used to document the analysis of the EGGDT. The basic functionality of the EGGDT is covered in the UC diagram (UML-UCD) of Figure III-1.

The following are the UCs contained in this diagram:

- “Create EG” (see Table III-1).
- “Create EG element”. EG elements are events, edges and simulation variables (see Table III-2).

- “Modify EG elements” content.
- “Delete EG Element”.
- “Generate Java code” using the correspondence articulated Table I-1.

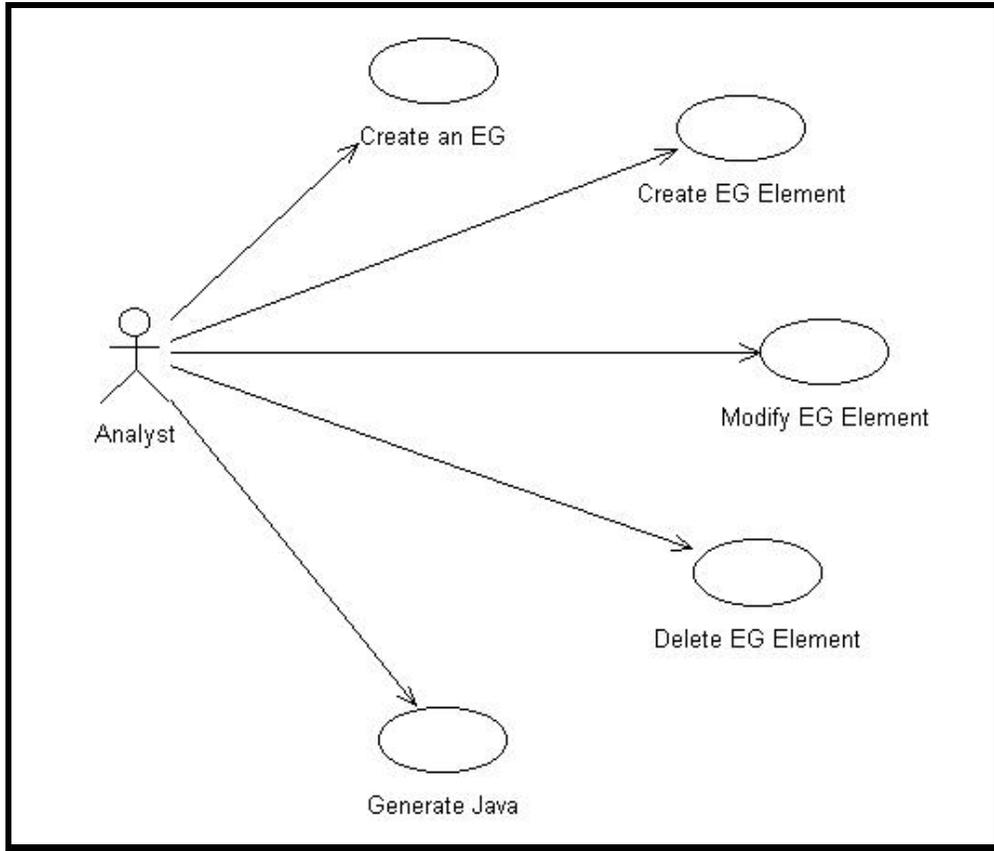


Figure III-1 UML-UCD. Basic EGGDT Functionality

These UCs were considered to cover the core requirements of the EGGDT. The functionality requirements were enumerated in paragraph II-B-0.

Once UCs were identified they were detailed in the extended UCs. As an illustration to explain UCs, the extended UCs for “Create EG” and “Create EG Element” are presented in Table III-1 and Table III-2.

Use Case:	Create EG.
Actors:	Analyst
Purpose:	Create a new EG. Includes the EG properties like name, project, package and so on.
<b>Typical Course of Events</b>	
Actor Action	System Response
1. The Analyst initiates an EG entering all necessary data.	2. Creates a new EG.
<b>Alternative Courses.</b>	
Line 2. If the name of the EG is not unique in the package, the system prompts an error msg.	

Table III-1 Extended UC Create EG

Use Case:	Create EG element
Actors:	Analyst
Purpose:	Create a new EG element. Allows the user to attach element's properties. The system must update the list of elements
<b>Typical Course of Events</b>	
Actor Action	System Response
1. The Analyst initiates the creation of an event, edge or simulation variable	
<b>Case "Create Event"</b>	
	2. Creates the new event instance.
	3. Updates the list of events.
<b>Case "Create Edge"</b>	
2. The Analyst specifies the source and target event of the edge.	3. Creates the new edge instance.
	4 Updates the outgoing edges list of the source event.
	5 Includes a reference of the target event in the edge.
	6. Updates the list of edges.
<b>Case "Create Simulation Variable"</b>	
2. The Analyst specifies if it is a State Variable or Simulation Parameter.	3. Creates the new simulation variable instance.
	6. Updates the list of simulation variables.
<b>Alternative Courses.</b>	
Line 2. If the name of the event or the simulation variable is not unique in the EG, the systems prompts an error msg.	

Table III-2 Extended UC Create EG Element

**b. Conceptual Model**

The Conceptual Model (UML-CM) in Figure III-2 summarizes and identifies the primary classes related with core functionality and their associations. The EG is composed of events, edges and simulation variables. Events have outgoing edges which point to target events. Simulation variables can be state variables or simulation parameters. Events actions modify state variables and edges conditions consult their value.

**2. Task Analysis**

**a. Use-Cases**

Task analysis addresses the problem of defining the external appearance and behavior of the EGGDT. Figure III-3 shows the UML-UCD corresponding with the interface functionality required.

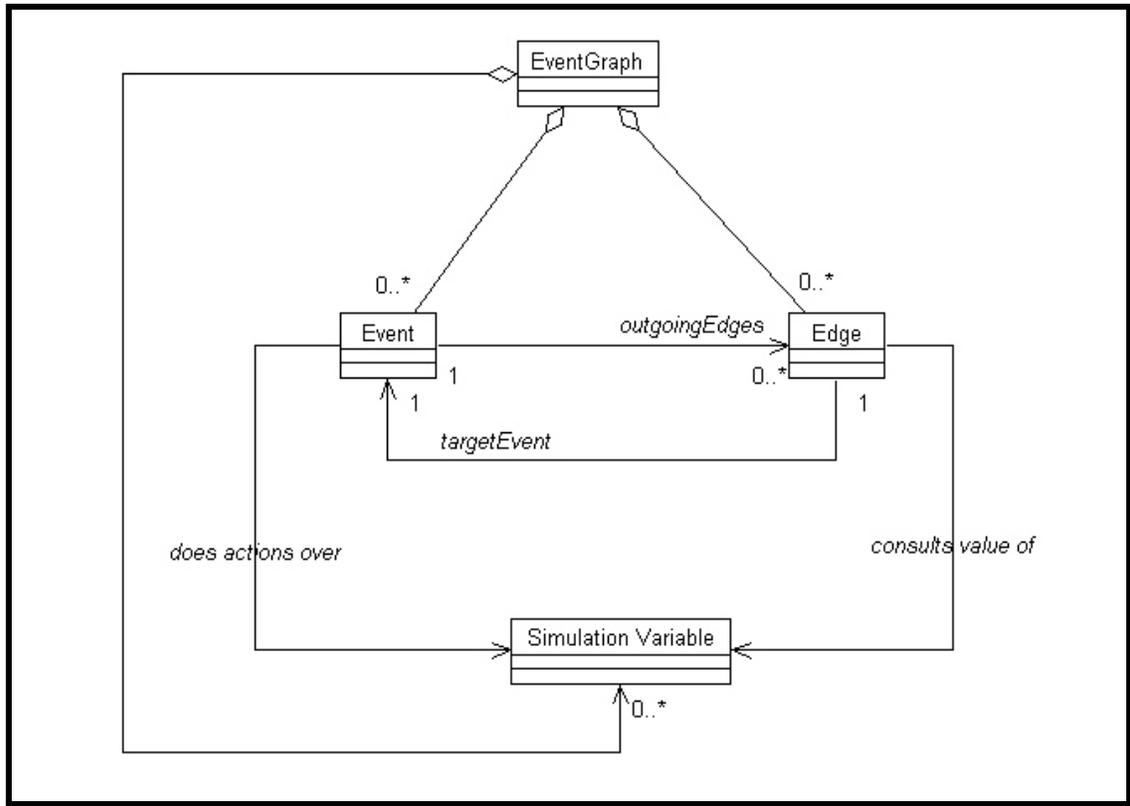


Figure III-2 UML-CM Diagram. Basic EGGDT Model

The UCs identified are:

- “Initiate the EGGDT” (see Table III-3).
- “Manipulate Graphical Element” includes the creation and modification of these elements. Graphical elements are circles and arcs representing the events and edges respectively.
- “Manipulate simulation variables” that are included in the EGGDT for appropriate representation, for example, a table or tree.
- “Manipulate EG files” implementing the “Open”, “Save”, “Rename” and “Saved As” services.

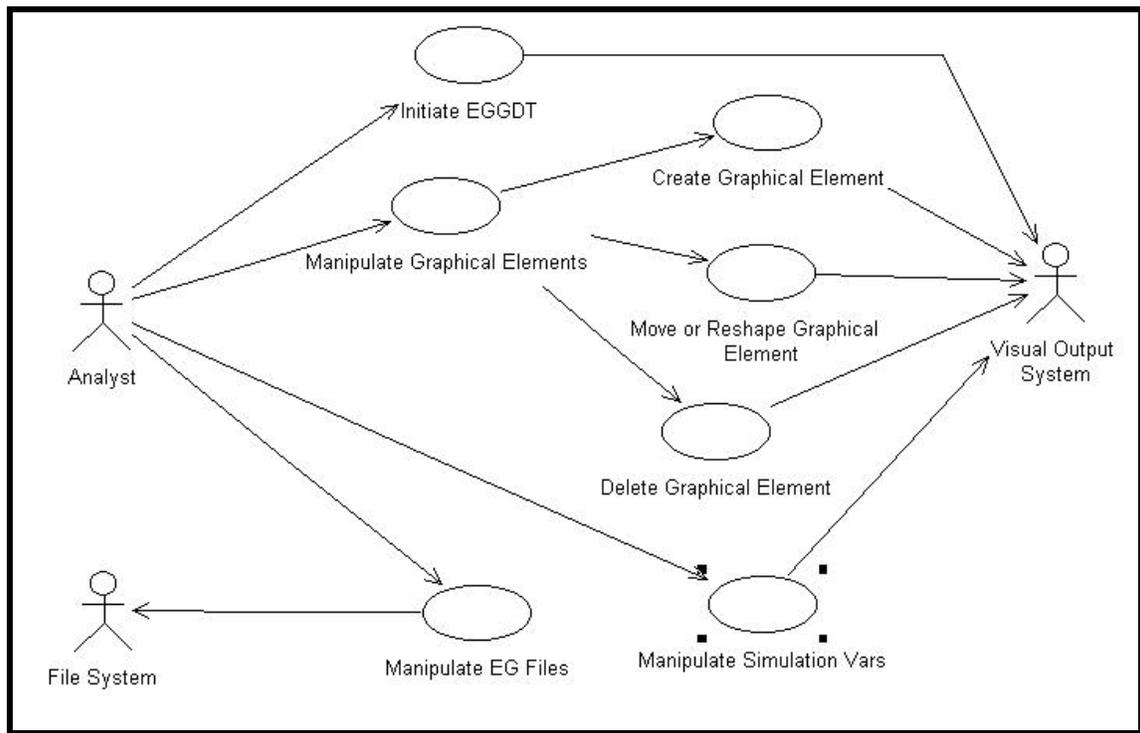


Figure III-3 UML-UCD. EGGDT GUI Functionality

As an example, the extended UC for Initiate EGGDT is provided.

Use Case:	Initiate EGGDT.	
Actors:	Analyst, Visual Output System	
Purpose:	Launch the Graphical Interface Tool. Initialize all the variables. Load the options for the session and display the GUI with a start up window.	
<b>Typical Course of Events</b>		
Actor Action		System Response
1. The Analyst starts the EGGDT program.		2. Initiates the graphical tool. 3. Prompts the user to create a New EG or Open an existing EG.
4. The Analyst enters the selected option.		
<b>Case “Create New EG”</b>		
		5. Displays the EG properties window.
6. The Analyst modifies the properties of the EG.		7. Opens a new blank EG in the Visual Output System.
<b>Case “Open EG”</b>		
		5. Displays an open file window.
6. The Analyst selects the EG file to open.		7. Loads the selected EG elements and shows them in the Visual Output System.
<b>Alternative Courses.</b>		
Line 6. The system prompts an error message, if any EG property is unacceptable or the selected EG file to open is not valid.		

Table III-3 Extended UC Initiate EGGDT

***b. Conceptual Model***

The UML-CM in Figure III-4 summarizes and identifies the primary classes related with the interface. The GUI controller manipulates two different interface areas. The “Drawing Area” displays the representations of edges and event– arcs and circles. The “Area of Variables” provides a means to manipulate simulation parameters and state variables.

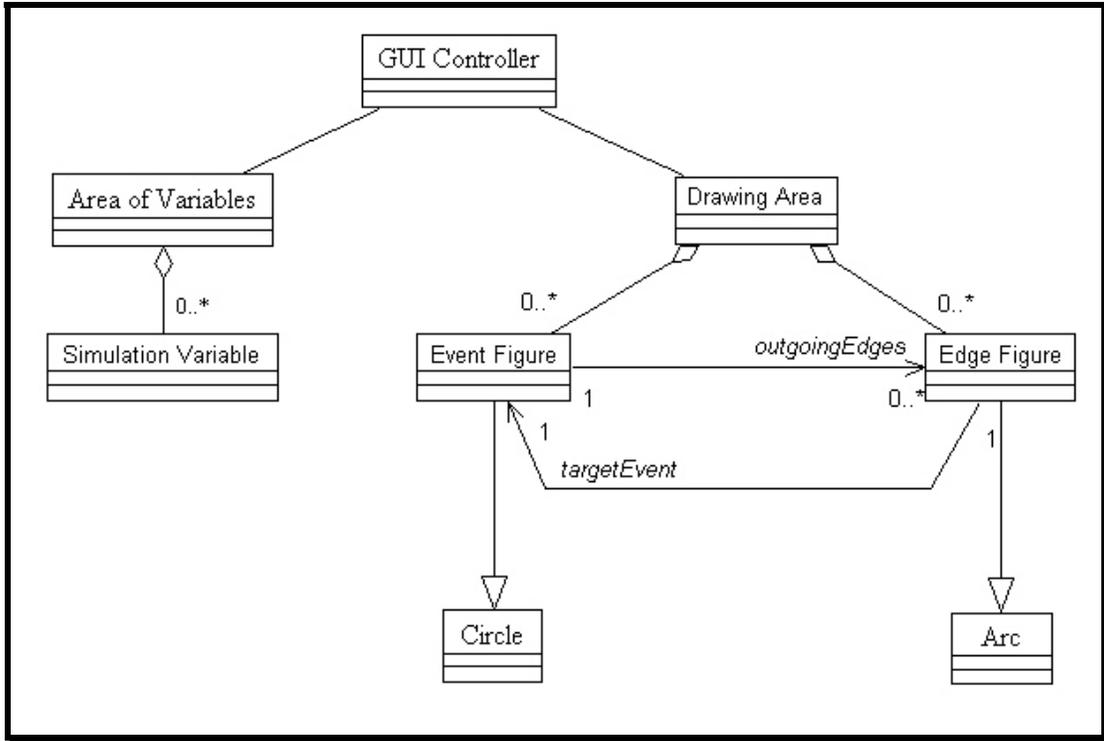


Figure III-4 UML-CM Diagram. EGGDT Interface Model

#### D. EGGDT DESIGN

The design phase pursues the building of the software architecture. Many tools are available to help perform this task; Collaboration Diagrams (UML-Cold) and State Machine Diagrams (UML-SM) were used in the development of the EGGDT. UML-ColdDs allow easy access in applying patterns, as discussed in Chapter I.

Following are some examples showing the approach used to design the EGGDT. To demonstrate the different approaches used to design the application and interface, two parallel methods, one from each domain, have been chosen.

##### 1. System Method “Create Edge”

“Create Edge” is a method in the domain of the application; no interface considerations are contemplated. Figure III-5 shows the UML-Cold of this method.

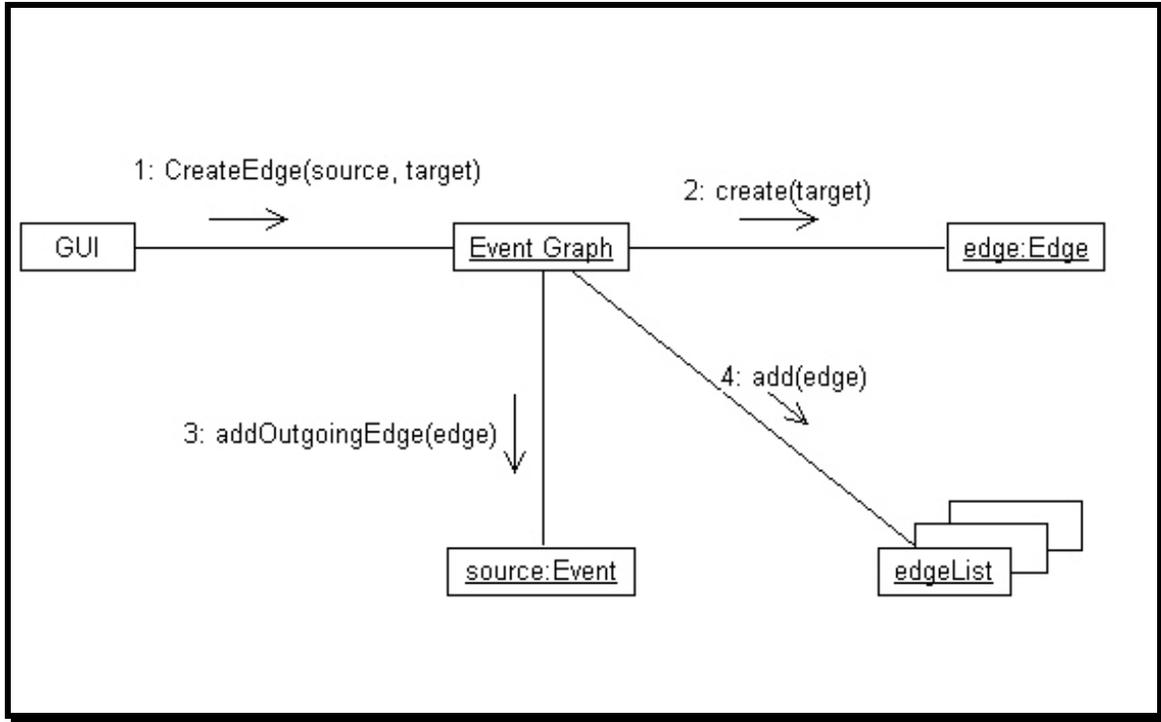


Figure III-5 UML-Cold. Create Edge

The GUI initiates a request to create an edge. The event graph object receives the source and target event for the new edge. It creates an edge passing a reference of the target event. Finally, the event graph object passes the new edge reference to be included in the list of edges and in the list of outgoing edges of the source event.

## 2. System Method “Create Edge Figure”

The method “Create Edge Figure” is part of the interface domain; therefore, the output and input system are considered. Figure III-6 contains the UML-Cold for this method. This method is called by the Analyst, who passes the source and target events and the path of the arc. The GUI controller object creates an edge figure instance providing the target event and path. The edge is included in the list of edge figures and in the list of outgoing edges of the source event. Finally, the output system is requested to repaint.

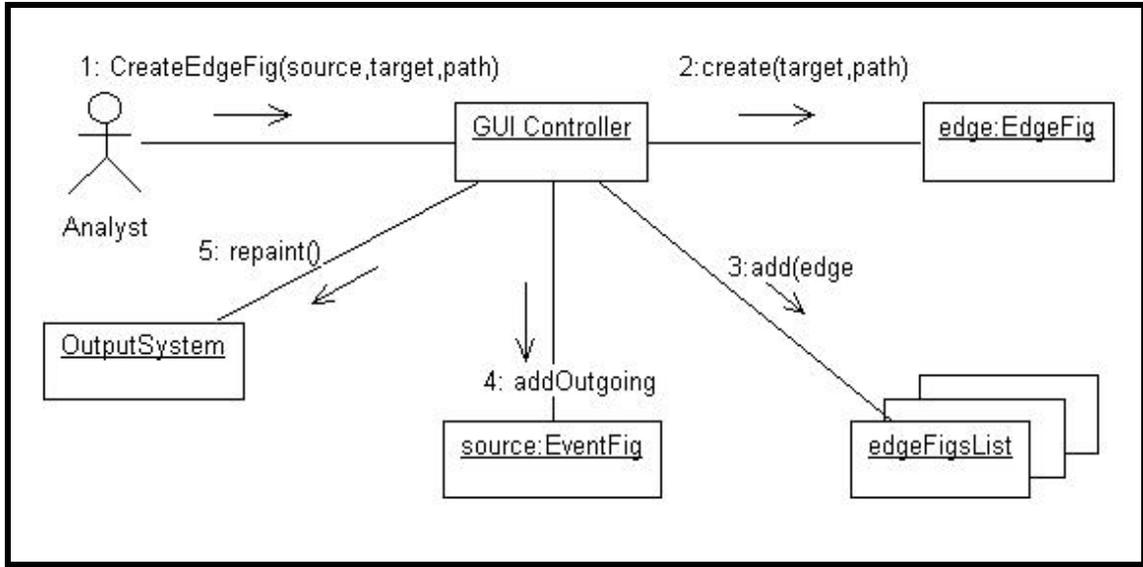


Figure III-6 UML-Cold. Create Edge Figure

### 3. Mouse Control Behavior

State machines were found to be very useful in determining and depicting the behavior of control elements. UML State Machines Diagrams (UML-SMD) were used.

The mouse control behavior is depicted in the UML-SMD of Figure III-7. The states of the mouse controller are represented by rounded rectangles. The transitions between states are represented by arcs. The events that trigger these transitions are expressed as labels in the arcs. These events are:

- mD.- dragging the left button on the mouse
- mP.- pressing the left button on the mouse
- mR.- releasing a button on the mouse
- mDC.-double clicking the left button on the mouse
- mCD.- dragging the central button on the mouse
- mM.- moving the mouse

As an example of mouse control modeling, consider the behavior of the controller when creating an edge. The controller starts in an “idle” state; when the mouse’s central button is dragged over an event or the mouse’s left button is dragged over an event while pressing the keyboard’s control-key, the system transits to the “creating edge” state. Each time a button on the mouse is released over an empty area, a new inflection point is created, but the controller does not change its state.

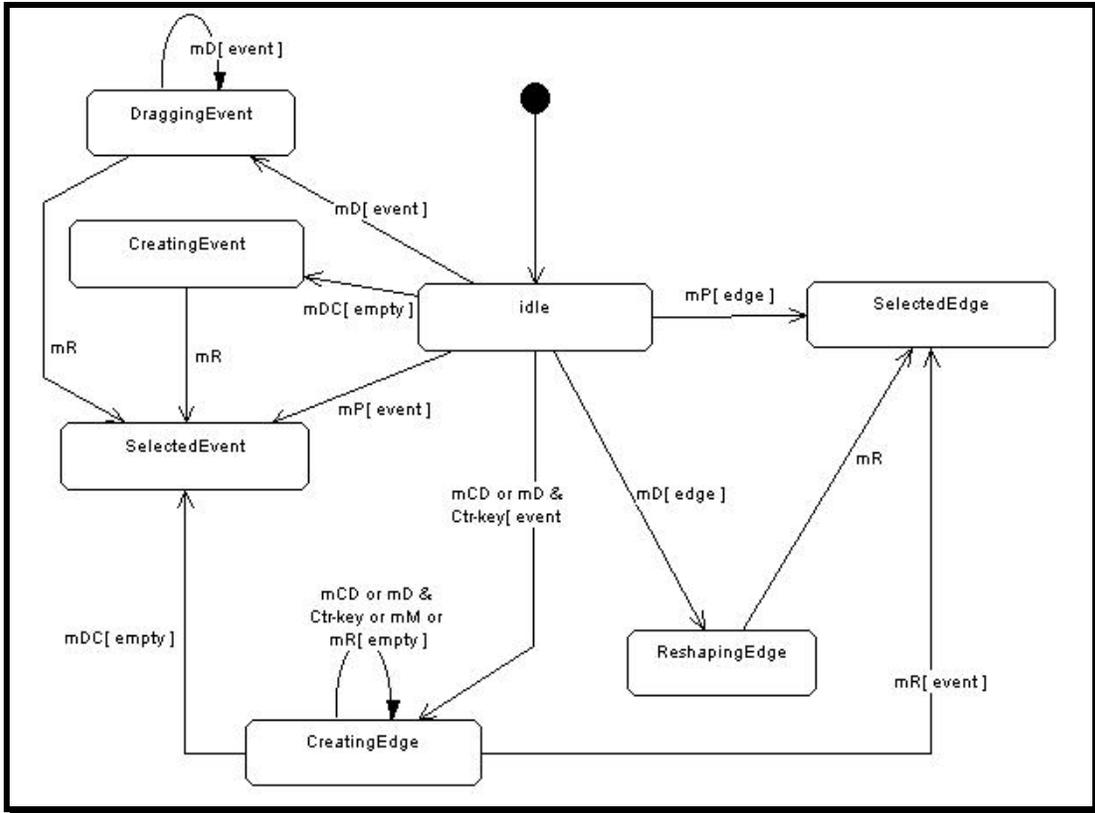


Figure III-7 UML-SMD. Mouse Controller

If the mouse is released over an event, an edge is created and the controller transits to “selected edge” state. Alternatively, if the mouse is double-clicked over an empty area, the operation will be cancelled and the controller transits to the “selected event” state.

## E. CONCLUSION

This chapter provided an overview of the development of the first iteration of the EGGDT project. The requirements were broken down into application and interface. During the analysis phase, a UC approach was used to determine which requirements to develop in this iteration. Extended UCs describe the details of every functionality required. UML-CMs identify the primary objects in the application and interface domain.

Finally, for the design phase UML-CoIDs and UML-SMDs were used. UML-CoIDs provided a means of getting inside the details of every system method and an easy

way to implement patterns. UML-SMDs were found beneficial in representing the behavior of control classes, such as the mouse controller.

By applying these techniques, separating application and interface domains at all levels of development and using UML artifacts to depict the A&D decisions, a more decoupled and maintainable software architecture has been achieved. The next chapter explains how usability was pursued through formative experiments during the development.

THIS PAGE INTENTIONALLY LEFT BLANK

## **IV. FIRST EXPERIMENT: USER RESPONSE TO THE PROPOSED TOOL GRAPHICAL USER INTERFACE (GUI) DESIGN**

### **A. INTRODUCTION**

Two formative experiments were conducted during the design of the Event Graph Graphical Design Tool (EGGDT) to ensure that the final product would fulfill user expectations.

The goal of the first experiment<sup>4</sup> was to evaluate the quality of the initial design of the EGGDT. The rest of this chapter describes this experiment discussing its influence on the EGGDT final design. The second experiment is covered in Chapter V.

### **B. PURPOSE OF THE EXPERIMENT**

This experiment did not attempt to be a comprehensive test of the tool, but a device to obtain feedback from the potential users about the approach chosen for the GUI of the EGGDT. Additionally, the experiment provided an initial opportunity to use the techniques of Human Factors necessary to accomplish usability. These techniques include design of experiments, task analysis, design of experimental protocol, development of prototypes and analysis of the data from the experiments.

When this experiment was conducted, the GUI was still in the initial phase, therefore formative evaluation was used. Since this was the first usability experiment, this evaluation was expected to introduce important design changes. The following sections of this chapter will summarize the evaluation work.

### **C. DESIGN OF THE EXPERIMENT**

Usability goals were set prior to appraising the GUI. These goals served as a reference point for the evaluation of the GUI. In addition, they acted as a tangible measurement of the usability success level for the interface design.

---

<sup>4</sup> This experiment was part of the final project for MV-4203 (Prof. Rudy Darken) in team with LCdr. Paulo Silva (Portuguese Navy).

Hix suggests in [HIX93] that first time developers should not start making a large usability specification table. Therefore, only the following three usability attributes for assessment were included: “initial performance”, “learnability” and “first impression”. The complete Usability Specification Table is presented in Appendix C.

### **1. Participants**

The word “participant” is used to refer to the subjects who collaborated in the evaluation experiment. Five participants involved in five evaluation sessions were representative of the users group for the EGGDT as defined in Chapter III.

### **2. Tasks to Perform**

A set of tasks was selected for the purpose of the interface evaluation. These tasks are a subset of those identified in the task analysis phase (Appendix B), and represent typical tasks users performance. Appendix C lists all tasks that the participants were required to perform.

Most tasks were benchmark tasks, which means they were used to obtain some quantitative measurement of usability performance of a given interface attribute. Other tasks were not quantitative but were included to complement the sequence of events in the experiment.

## **D. DESIGN OF THE PROTOTYPE**

The prototype presented the following capabilities:

- The prototype showed full button and menu bars; however, only the following buttons were enabled:
  - “Select tool” 
  - “Create Event” 
  - “Create Edge” 
- The intent of the prototype was to test the GUI approach even though the necessary functionalities were not yet implemented.

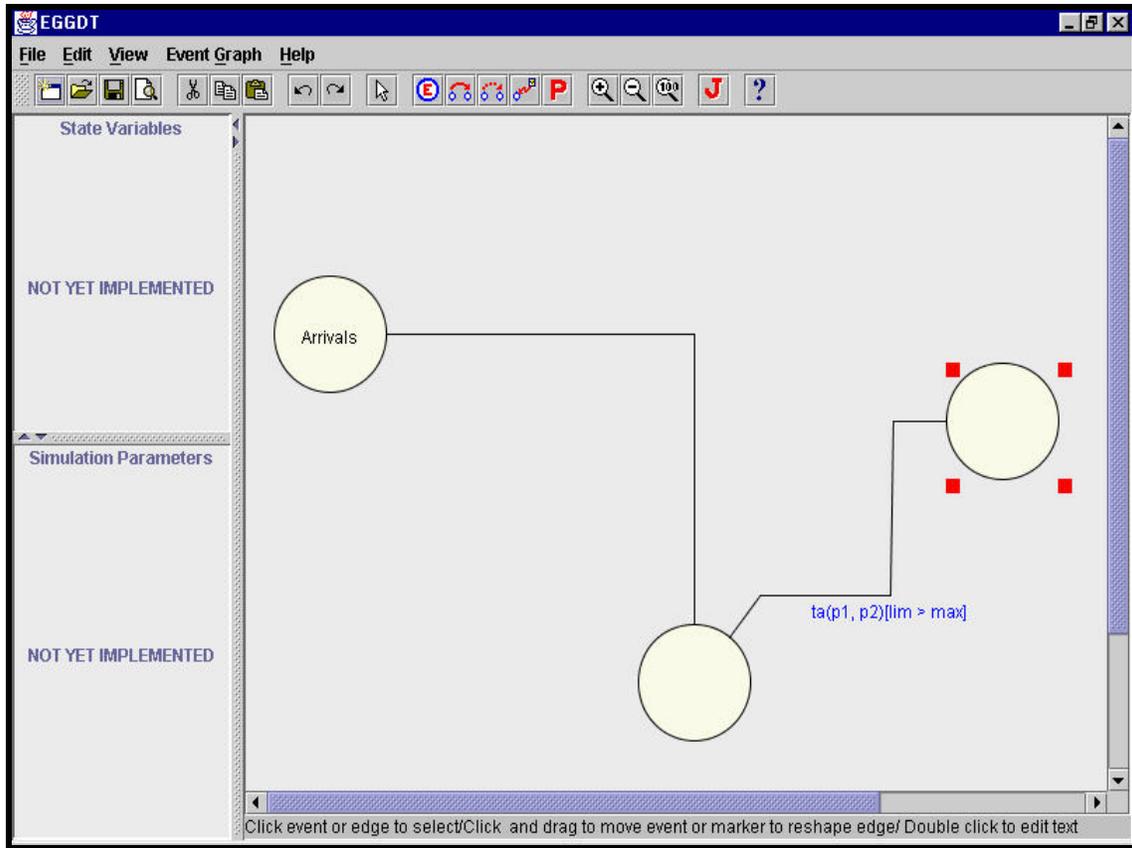


Figure IV-1 Snapshot of the EGGDT Prototype

- Modes:
  - Events could be created by selecting “Create Event” and clicking in the drawing area.
  - Edges could be drawn by selecting “Create Edge” and clicking first in the source event and then in the target event. Middle points could not be defined. Three edge interior points were provided for reshaping, but they could not be deleted.
  - The “select tool” had to be activated to move events and reshape edges.
  - The prototype used the permanent modality method, i.e., when the “Create Event” button was pressed, it stayed pressed, so multiple events could be created. In a temporary modality method, the button would have to be pressed to create each event.
- The status bar offered help depending on the active mode.

No further functionalities were included in this first prototype. The prototype’s layout is shown in Figure IV-1.

## **E. EXPERIMENTAL PROTOCOL**

The standard protocol for performing evaluation with humans requires a consent form to be signed by the participant. Appendix C shows the consent that was conveyed to each participant to read and sign.

A sheet with introductory instructional remarks was prepared, given to each participant, and read aloud by the evaluator. Any additional verbal explanation about the interface was given in such a way that all participants obtained the same information, thus ensuring consistency among participants. Appendix C includes a sample of the instructions.

### **1. Pilot Testing**

When all pieces of the setup were together, a pilot test was performed clarifying the challenge of a realistic observation of errors. Since the experiment's environment setup was simplistic, without videotaping, audiotaping or interface interaction recording capability embodied in the prototype, the evaluation was simplified. The performance data focused primarily on the required time to complete tasks. Nevertheless, the evaluator also recorded qualitative information about the types of errors observed.

Once the necessary modifications on the task list and data collection forms were introduced, a second pilot test was performed. A new problem was observed when users responded to the questionnaire. Hix stated in [HIX93] that participants should clearly understand that the evaluation is not proposed to assess them, therefore they should not fear any consequences of a "bad" or "good" performance. In this academic environment, where all selected participants were students (sometimes classmates of the principal investigator), participants may have been tempted to provide higher scores on the questionnaire. As a result, the following message was introduced at the beginning of the questionnaire: REMEMBER, YOU ARE NOT EVALUATING US BUT THE TOOL. SINCERITY IS APPRECIATED.

### **2. Evaluation Sessions**

Having completed two pilot sessions and introduced some modifications to the required material, the evaluation sessions were performed. Five participants took part in

separate sessions conducted in the OR simulation laboratory in Glasgow Hall at the Naval Postgraduate School (NPS). Participants were first briefed about the purpose of the experiment, and then given a copy of the Session Instructions (Appendix C), which the experiment's controller read aloud. Participants also read and signed the consent sheet (Appendix C).

## F. STATISTICAL ANALYSIS

The following table summarizes quantitative data extracted from the Data Collection Forms. For each of these tasks a statistical analysis is presented, including mean, standard deviation and 95% confidence interval (CI).

	<b>Benchmark Tasks</b>					
<b>Participant</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>
<b>P1</b>	80.0	113.0	60.0	10.0	14.0	37.0
<b>P2</b>	57.0	120.0	20.0	9.0	20.0	21.0
<b>P3</b>	49.0	120.0	99.0	2.0	22.0	37.0
<b>P4</b>	43.0	47.0	56.0	4.0	11.0	28.0
<b>P5</b>	47.0	17.0	50.0	45.0	14.0	33.0
<b>Mean</b>	55.2	83.4	57.0	14.0	16.2	31.2
<b>Stand Deviation</b>	14.8	48.2	28.3	7.7	4.6	6.8
<b>95% CI</b>	36.9	23.6	21.9	0.0	10.5	22.8
	73.5	143.2	92.1	35.9	21.9	39.6

Table IV-1 Benchmark Times in Seconds

In Table IV-1 :

- The participants' responses are assumed to be independent and identically distributed (iid).
- CIs are calculated using t-student distribution with 4 degrees of freedom.
- This small sample results in large variances and wide confidence intervals.

Figure IV-2 shows a box-plot of the benchmark tasks' times. Every participant performed similarly in all tasks except for task 2 showing a great variance (83.4 sec). This task was “Create an Edge between events ‘Run’ and ‘Arrival’ with delay time ‘ta’ ”. As discussed later, the edge creation method presented difficulties; it was not intuitive, so some participants were able to create edges very fast, whereas others required help to complete the task.

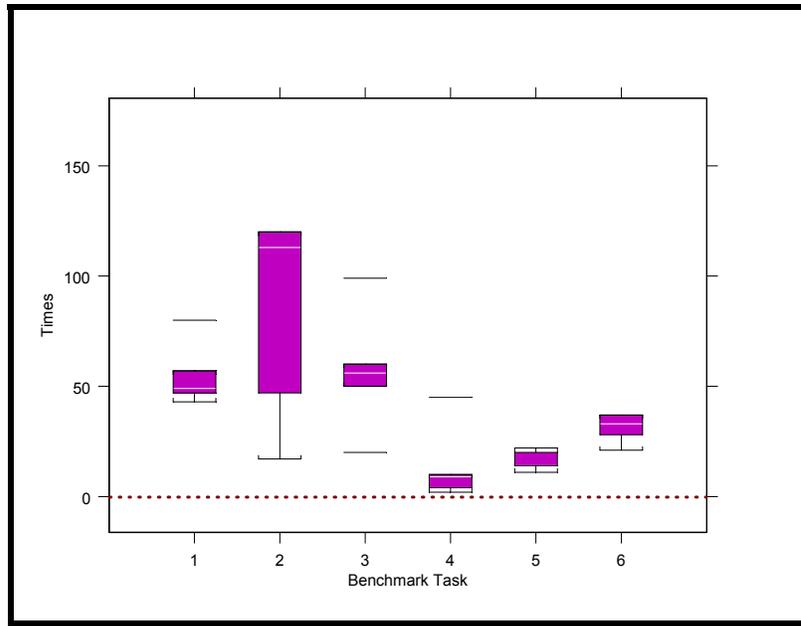


Figure IV-2 Box-plot of the Times for Benchmark Task

### 1. Usability Attribute “Learnability”

Learnability was measured through the comparison between benchmark tasks 2 and 6 (create an edge). Both were very similar tasks, one performed at the beginning of the experiment and the other at the end.

To test the null hypothesis that both means are the same, a paired t-test was performed. This resulted in a p-value of 0.0743 and a 95% CI of [-8.18, 112.58]. Even though the CI includes 0 for a 0.5 level of confidence, the difference between means is considered significant.

In conclusion, participants in the experiment generally learned the tool easily (the experiment only lasted 20 minutes) and performed tasks faster at the end of the session.

Figure IV-3 illustrates the difference in performing both similar tasks. With one exception, all participants performed better in the second task.

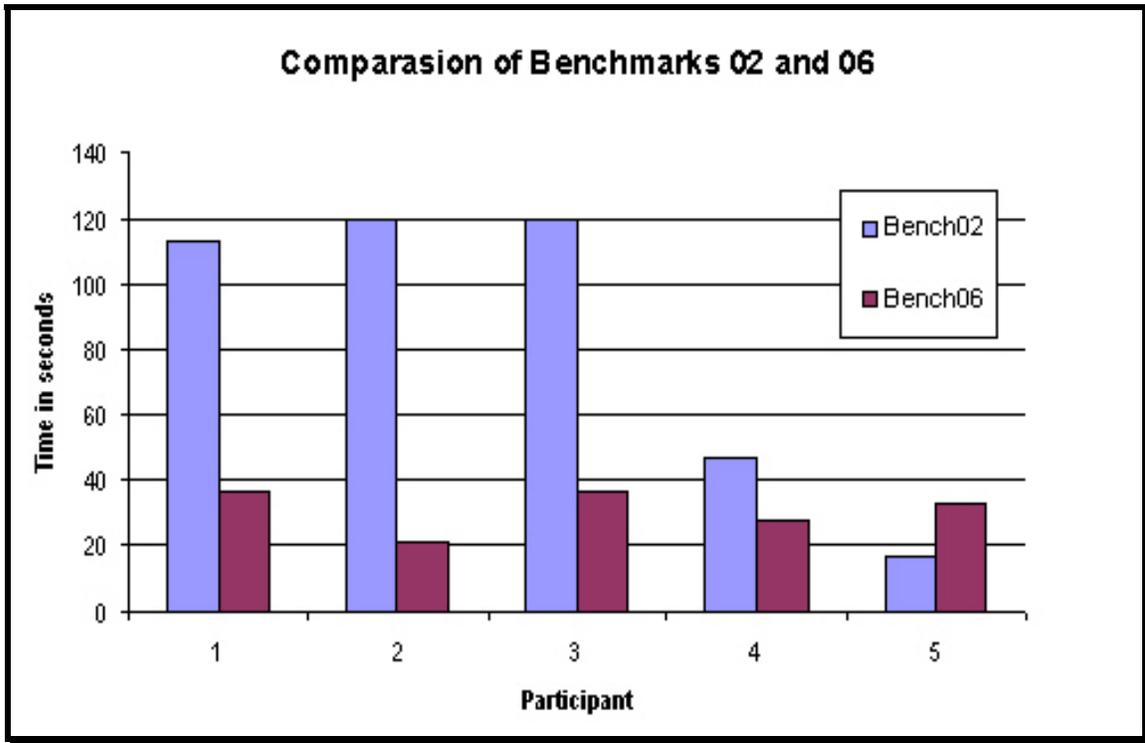


Figure IV-3 Differences in Performing Both Tasks of Creating Edges

## 2. Usability Attribute “First Impression”

The “First Impression” usability attribute measurement was observed through the questionnaire for “User Interface Satisfaction” (Appendix C). This questionnaire provided subjective but quantitative data, which are summarized in Table IV-2.

	Question																	
Participant	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P1	6.0	5.0	6.0	7.0	7.0	8.0	8.0	7.0	7.0	6.0	7.0	7.0	8.0	8.0	9.0	8.0	6.0	6.0
P2	7.0	8.0	7.0	7.0	5.0	6.0	6.0	7.0	7.0	7.0	5.0	6.0	7.0	7.0	9.0	9.0	9.0	5.0
P3	9.0	7.0	7.0	8.0	7.0	9.0	9.0	7.0	9.0	7.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	7.0
P4	7.0	7.0	7.0	7.0	7.0	8.0	8.0	8.0	8.0	7.0	7.0	7.0	8.0	8.0	8.0	8.0	7.0	8.0
P5	9.0	8.0	8.0	5.0	7.0	9.0	9.0	7.0	7.0	8.0	9.0	9.0	9.0	7.0	9.0	9.0	9.0	7.0
<b>Mean</b>	7.6	7.0	7.0	6.8	6.6	8.0	8.0	7.2	7.6	7.0	7.4	7.6	8.2	7.8	8.8	8.6	8.0	6.6
<b>Std Dev.</b>	1.3	1.2	0.7	1.1	0.9	1.2	1.2	0.4	0.9	0.7	1.7	1.3	0.8	0.8	0.4	0.5	1.4	1.1
<b>95% CI</b>	5.9 9.2	5.5 8.5	6.1 7.9	5.4 8.1	5.5 7.7	6.5 9.5	6.5 9.5	6.6 7.7	6.5 8.7	6.1 7.9	5.3 9.5	5.9 9.2	7.1 9.2	6.8 8.8	8.2 9.3	7.9 9.2	6.2 9.8	5.2 8.0

Table IV-2 Questionnaire Responses

Table IV-2 notes:

- Scores go from 0 to 9 -- from worst to best.
- Questions (see Appendix C for a detailed description):
  - Overall reaction to the Event Graph Design Tool (questions 1 to 5)
  - Screen:
    - Characters on the computer screen (question 6)
    - Tool bar with buttons (question 7)
    - Organization of information on screen (question 8)
    - Terminology and system information:
      - Computer terminology is related to the task being done (question 9)
      - Menu Items and Tool bar with buttons function identification (question 10)
  - Instruction
    - Operation of the system (question 11)
    - New features explored by trial and error (question 12)
    - Names and use of commands are learned (question 13)
    - Tasks are performed in a straightforward manner (question 14)
  - System capabilities
    - System speed (question 15)
    - System reliability (question 16)
    - Correction of mistakes (question 17)
    - Experienced and inexperienced users' needs are taken into consideration (question 18)

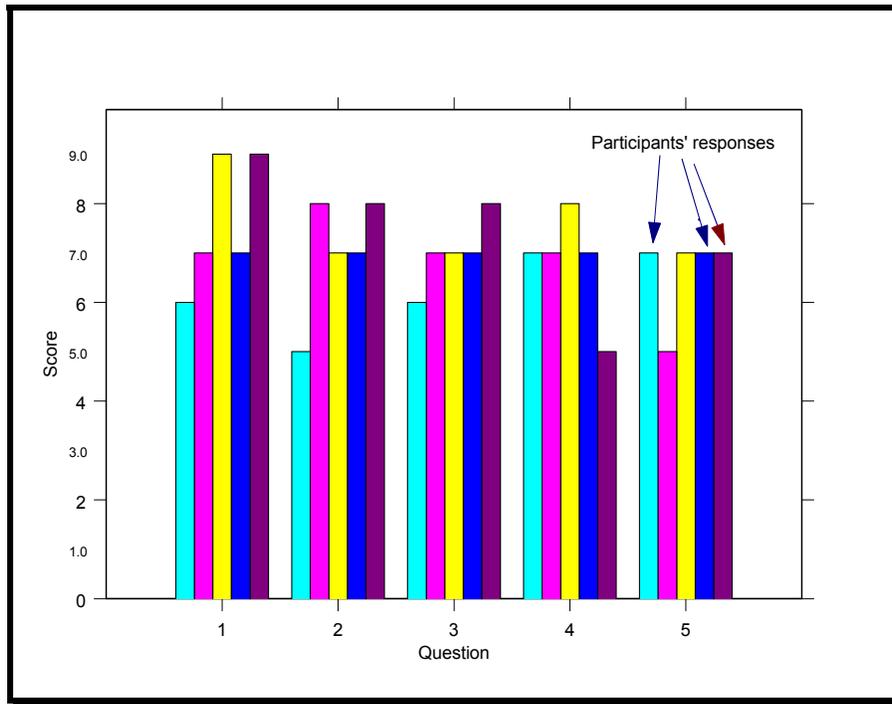


Figure IV-4 Participants' Responses for Overall Reaction Questions

Figure IV-4 shows the participants opinions of the tool (questions 1 to 5). The questions related to the participants overall reaction to the EGGDT. The questionnaire used quantitative scales from 0 to 9 labeled from terrible to wonderful; difficult to easy; frustrating to satisfying; dull to stimulating; rigid to flexible. The values assigned by the participants are evaluated as “good” considering the immature state of the tool. However, participants did not consider the tool flexible. It was assumed that this immaturity could be eliminated in successive versions of the product.

The mean for question 18 (“Experienced and inexperienced users' needs are taken into consideration”) in Table IV-2 is the lowest among the means. Figure IV-5 shows the individual scores. As a result of these scores more emphasis has to be made in designing the tool so that it takes experience into consideration. A means of achieving this goal is through help topics and a self-study courses. However, these improvements were not planned for the product's iteration covered by this thesis research effort.

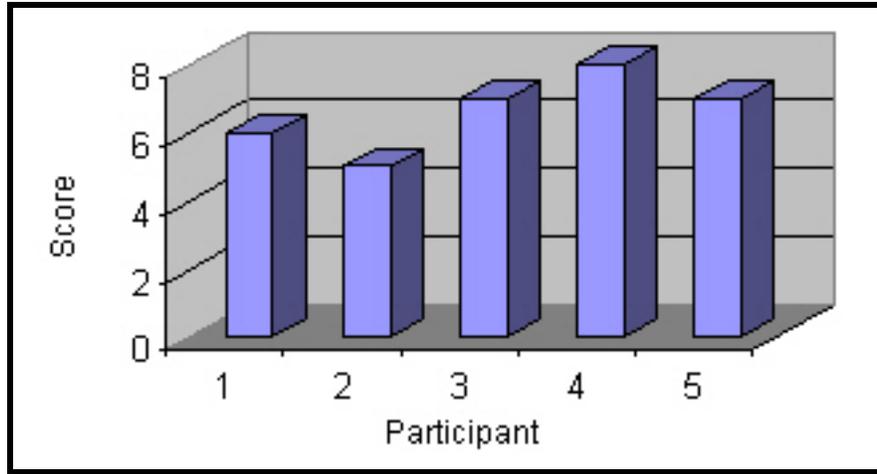


Figure IV-5 Participants' Responses for "Experience Taken into Consideration"

In conclusion, the users' first impression was good. Participants seemed to be concerned about adaptability to inexperienced and experienced users and flexibility of the tool.

### 3. Qualitative Data

Throughout each session, notes were taken by the evaluator. This section presents a summary of that qualitative data.

#### a. Buttons and Menus

- The buttons "print", "attach note" "generate Java" and "properties" were difficult to identify.
- Menu item "Message Bar" should be called "Status Bar".
- Most participants first used the menu items to do the tasks, instead of buttons.

#### b. Modality

- At the beginning of the experiment, participants had problems understanding the permanent modality.
- Most participants did not identify the option "Select Element", which had to be selected to move events and reshape edges.
- Most participants did not realize that changes in the cursor were associated with mode changes.

#### c. Edges

- All participants had problems initially creating edges. When they noticed the status bar provided help, most of them were able to solve the problem. Later in the session, they had no problems creating edges.

- When edges overlapped, the participants had difficulties in selecting and editing them.
- Some participants complained that the inflection points could not be deleted.
- Participants had some problems moving selection markers of the inflection points of the edges, especially if they were in a sharp corner (programming problem).

*d. Text Editing*

- Participants had no problems editing event names; however, they had problems editing edge properties when edges superimpose one another.
- Some participants tried to exit editing using the Enter key. To exit editing, it was necessary to click outside the editor area.

## **G. LESSONS LEARNED**

The gathering of data (times and errors) by the experiment's controller was a difficult, error-prone task. When possible, a computer application must manage the collecting of data.

The questionnaire was too dense. Questions have to be selected carefully so as not to confuse participants. For example, the five questions about the overall opinions of the tool were overwhelming and unnecessary.

Pilot experiments are the key to discovering flaws in the design of experiments and protocols. The two pilot experiments discovered important defects in the proposed design, and as so, they helped to obtain more accurate experimental data for the evaluation.

## **H. CONCLUSION**

A formative experiment was planned in the early stages of EGGDT development to test the usability of the approach chosen for the GUI. The usability attributes tested were “initial performance”, “learnability” and “first impression”.

A basic prototype, incorporating all interface options but constraining its functionality, was built. After two pilot experiments that uncovered major flaws in the experiment's design, five participants performed the experiment in different sessions. The

experiment consisted of a battery of tasks perform with the prototype. Each participant's times and errors were recorded along with their responses to the questionnaire.

Major problems were related to the way edges were created. In response, a new formative experiment was implemented to compare different edge construction methods. This experiment is discussed in the next chapter.

## V. SECOND EXPERIMENT: ARC CONSTRUCTION

### A. INTRODUCTION

During the design of the Event Graph Graphical Design Tool (EGGDT) two formative experiments were conducted to ensure that the final product would fulfill user expectancies. The first formative experiment was described in Chapter IV. The second experiment<sup>5</sup>, which had particular concern with the way the edges are built, is discussed in this chapter.

### B. PURPOSE OF THE EXPERIMENT

After the first experiment, acceptance of the EGGDT was clearly predicated on the technique employed to draw edges. The first prototype provided a very rudimentary tool to create and manipulate edges. Two alternative techniques were proposed: the direct or free draw method (F method) and the right angle or vertical horizontal method (VH method). This experiment tried to determine which technique was more appropriate to be employed in the EGGDT.

### C. DESIGN OF THE EXPERIMENT

A crossover experiment design was selected with the following arrangement

	<b>R<sub>1</sub></b>	<b>X<sub>1</sub></b>	<b>O<sub>11</sub></b>	<b>X<sub>2</sub></b>	<b>O<sub>12</sub></b>
<b>N</b>	<b>R<sub>2</sub></b>	<b>X<sub>2</sub></b>	<b>O<sub>21</sub></b>	<b>X<sub>1</sub></b>	<b>O<sub>22</sub></b>

Selecting experiment participants was not random (N), because they had to be students of the Naval Postgraduate School (NPS). However, from the initial sample, two groups were randomly selected (R<sub>1</sub> and R<sub>2</sub>); one of these groups used the VH method first (X<sub>1</sub>) and then the F method (X<sub>2</sub>), while the others performed the experiment in the opposite order (F and then VH). Observations of both treatments were taken (O<sub>1</sub> and O<sub>2</sub>).

---

<sup>5</sup> This the material in this chapter was partially developed as a final project for the Human Factors Course OA3401 (Prof. Nita Miller) and in group with James Campbell, Parke Paulson, and Erik Hovda.

The two explanatory variables chosen were the difference in time required to perform the related tasks with each method and the participants' preference between the two methods.

To design the experiment the following validity threats were considered.

## **1. Internal Validity**

### ***a. History***

The threat to validity posed by history (commonly referred to as the learning effect) was addressed by a crossover experimental design. This threat is a consequence of the multiple treatment or exposures to the prototype. For example, the first time the participants see the prototype, they are naive and do not know exactly what to expect. The second time the participants see the prototype, they have expectations that they acquired during the first treatments. Their performance could thereby be affected by the order in which the exposures to the prototype occur. Alternating the order of the treatments for each participant mitigates this problem.

### ***b. Selection***

Participant population was not completely random; however, to help minimize this potential threat, the order in which each participant did the experiment tasks was random.

## **2. External Validity**

This threat was minimal because the population of future users was defined as students within the OR curriculum. The sample was drawn from this same group.

## **3. Conclusion Validity**

Selected explanatory variables correspond to the question that was being asked. The participant preference and the time to complete the experiment were believed to be valid variables in determining which technique was better.

#### 4. Statistical Validity

ANOVA was determined to be the best choice to analyze all of the gathered data. Logistic regression was considered as a statistical tool that could provide further insight into the data.

#### D. DESIGN OF THE PROTOTYPE

The prototype implemented the two techniques to draw edges. The F method allowed drawing straight arrows to connect circles. Dragging in any internal point reshaped arcs; arcs could take any shape desired with any number of inflection points. A possible layout with this method is shown in Figure V-1.

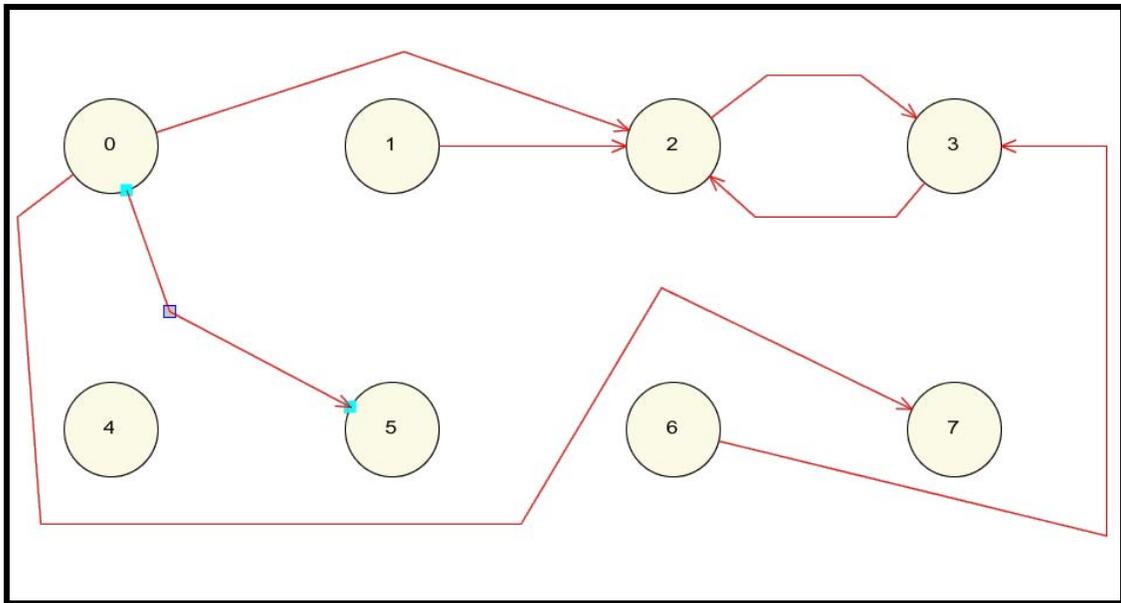


Figure V-1 The Direct or Free Method (F Method)

The VH method (Figure V-2) created arcs with one horizontal and two vertical segments that could be moved up or down and left or right. Two end segments (tail and head) connected to the circles and could be rotated at any angle around them. The prototype also provided two selectable points in the junction between the end segments and the vertical segments that could be moved any place.

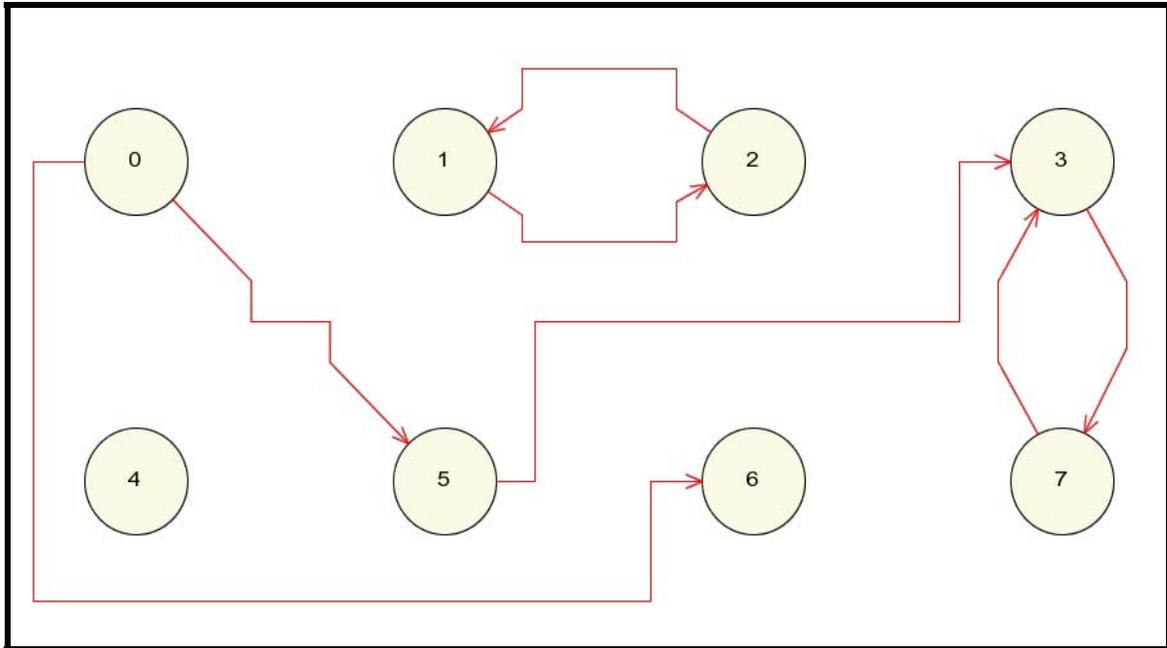


Figure V-2 The Right Angle or Vertical Horizontal Method (VH Method)

An application was developed to hold the prototype and to take control of the experiment's flow. This program also collected the data from each experiment.

#### E. EXPERIMENTAL PROTOCOL

Eighteen participants were recruited to be subjects in the experiment that were under the control of three different controllers. The Loosely Coupled Components (LCC) laboratory and Human Factors Human Systems Integration Laboratory (HSIL) in Glasgow Hall at NPS were used to set up the experiments.

Experiments were controlled by the computer program referred to above. Experiment controllers had to explain the details of the experiment and the use of each method, while documenting the participant's times for each task and number of errors calculated by the application. This kept the influence of the controllers at a minimum.

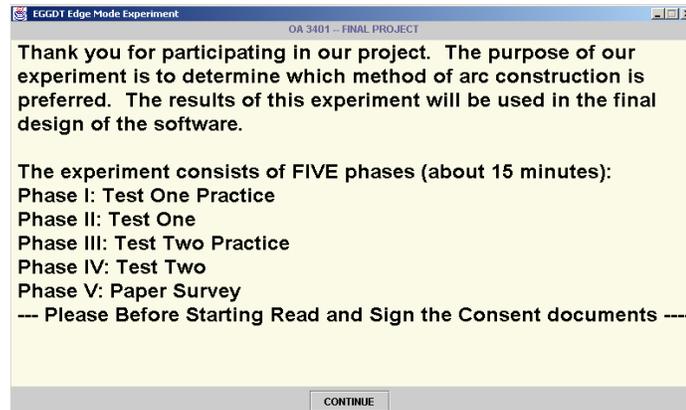


Figure V-3 Welcome Window

The Figure V-3 shows the welcome window.

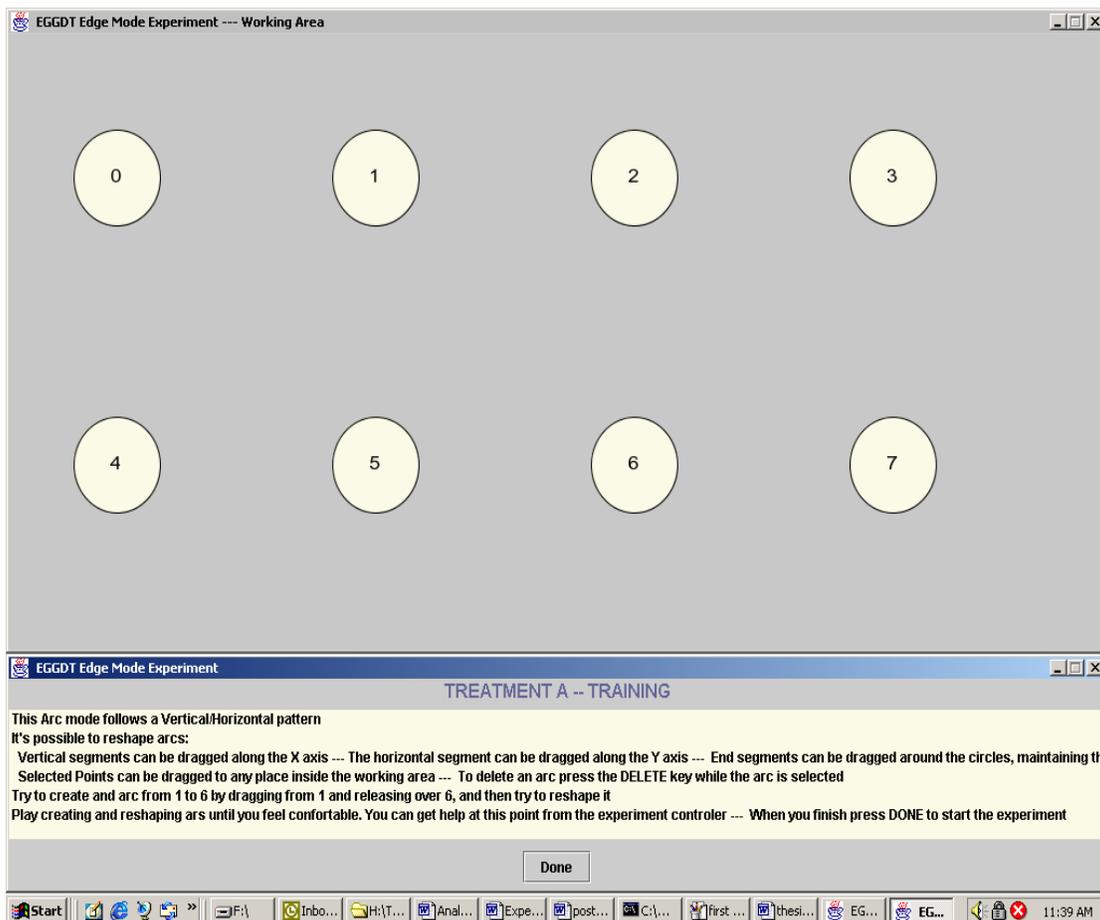


Figure V-4 Training Window

Experiments were divided into four phases. In the first phase participants learned how to use one of the methods. As discussed above, the order in which the participant was given the methods was random. Figure V-4 and Figure V-5 show the VH arc construction method's training and task window. Participants had the opportunity to play with this technique in the training window. In the next step, they had to complete all tasks in the task window and press the done button. The system displayed a window with the results (Figure V-6).

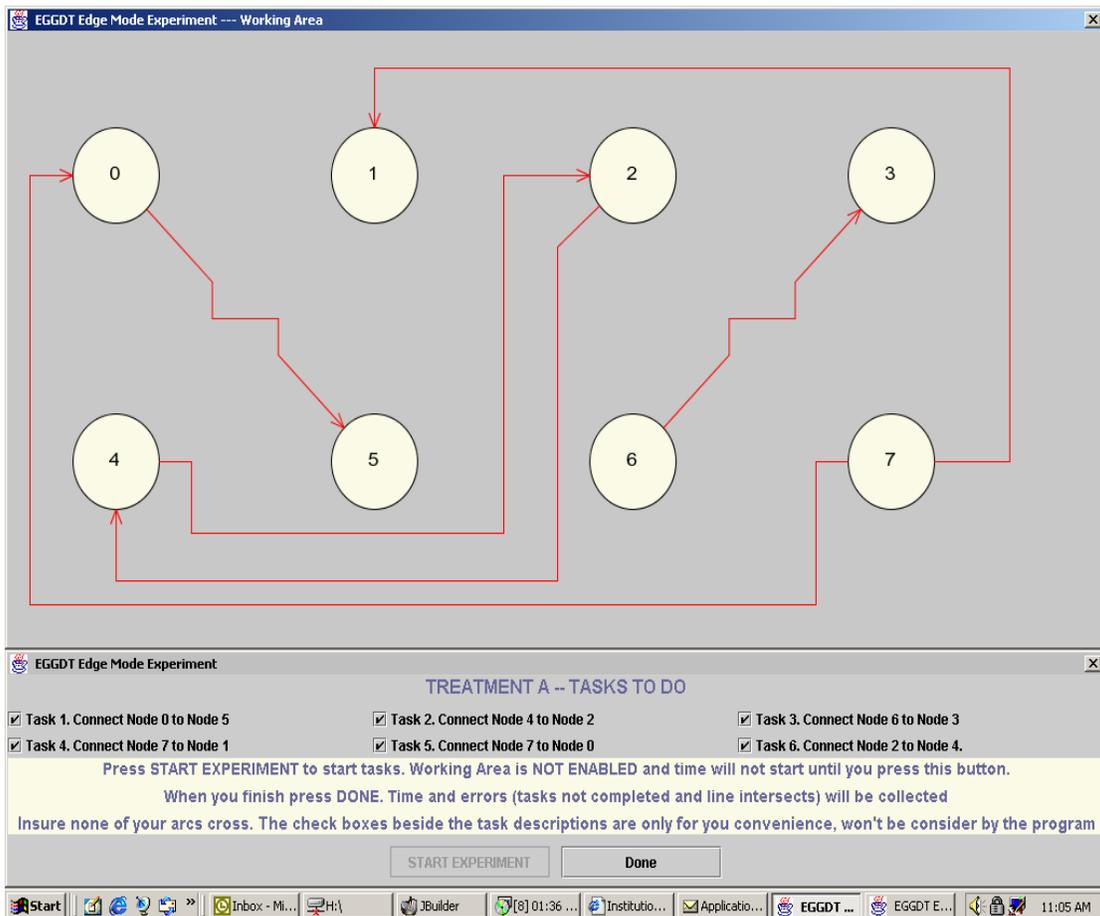


Figure V-5 Task Window

The procedure to perform the experiment with the other treatment was the same. The tasks to perform were mirrored to the ones done first, for example if one task was to create an edge between events 0 and 7 and other between events 5 and 2; the mirror tasks were create edges between 3 and 4 and between 6 and 5.

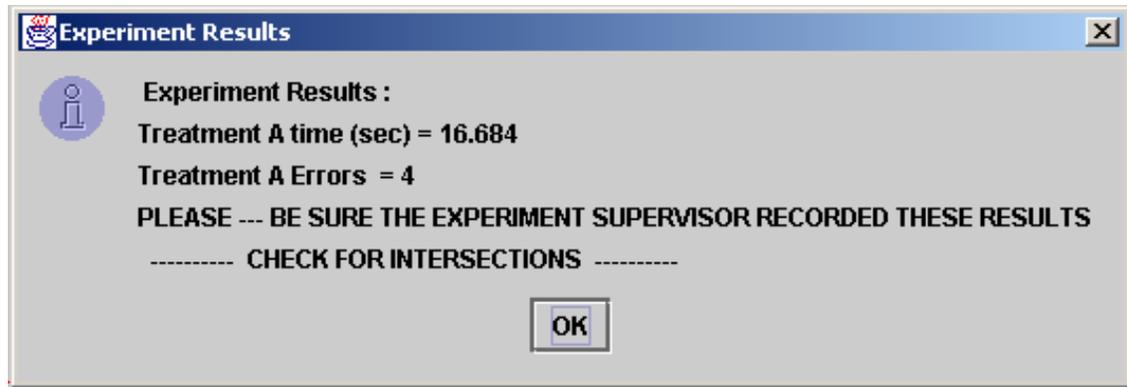


Figure V-6 Result window

Appendix D inventories the data collected.

## F. STATISTICAL ANALYSIS

As stated above, the objective of the experiment was to determine which alternative arc technique was preferred. Two factors influenced the selection: the difference in time to perform the tasks with the two different methods and personal preference. This statistical study demonstrates that the F method was preferred over the VH method.

### 1. Difference in Performance Time

The scatter plot in Figure V-7 shows the relationship between test times for each individual participant. Points plotted above the line show better time performance on the VH method of arc construction. Evidence demonstrated that all but four participants scored better in the F method.

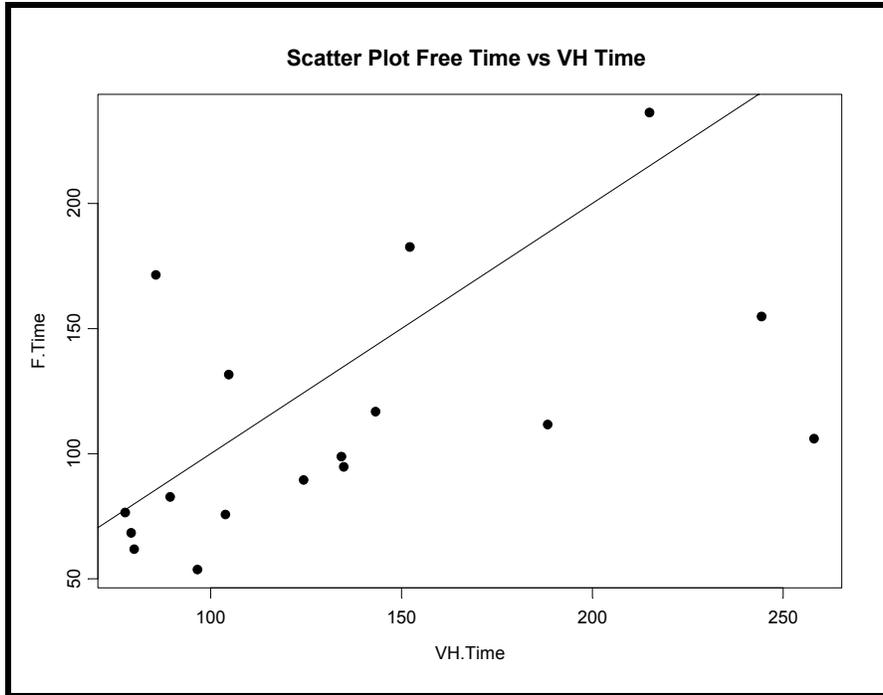


Figure V-7 Relationship Between Test Times per Participant

The following ANOVA (Table V-1) shows a significant difference between the two times to complete the tasks (p-value of .032).

	Df	Sum of Sq	Mean Sq	F Value	Pr(F)
VH Time	16	10446.29	10446.29	5.59	0.032

Table V-1 ANOVA Results of Time to Perform Tasks with F Method vs. Time with VH Method

This above analysis indicates that the F Method provides a faster way to draw arcs. However, a linear model was built to test if other factors influenced this difference in performance.

## 2. Linear Model to Explain Time Difference

The only linear combination that could be fitted to explain time difference was “Order” (VH/F or F/VH) plus “self-expressed experience with graphical environments” (like PowerPoint) plus “self-expressed level in computers use”.

Coefficients	Value	Std. Error	t value	Pr(> t )
Intercept	120.03	47.56	2.52	0.020
Order	36.33	9.84	3.69	0.002
Graphical	63.76	16.43	3.88	0.002
Level	-93.42	22.37	-4.18	0.001
<b>Residual stand. Error</b>	36.03 on 14 degrees of freedom			
<b>Multiple R-Squared</b>	0.60			
<b>F-statistic</b>	7.17 on 3 and 14 df, p-value is 0.004			

Table V-2 Linear Model Summary. Difference of Time Performance Against Order, Graphical Experience And Computer level

Table V-2 summarizes this linear model, which presents several problems:

- The “Level’s” coefficient value is negative but the “Graphical” experience is positive. This sign difference is confusing, since these two coefficients had been considered positively correlated.
- The R squared is 0.60, therefore, much of the variability in time difference is not explained with this model. It is assumed that this variability is due to the different characteristics of the two methods.
- The large residual standard error 36.03 is an indication that this is not a good model.

In conclusion, although it is possible to fit a linear model (a bad one) to explain the time performance difference, the difference in time is explained by the contrast between techniques and not by other factors.

### 3. User Preference

The pie chart in Figure V-8 gives a summary of the participants’ preferences.

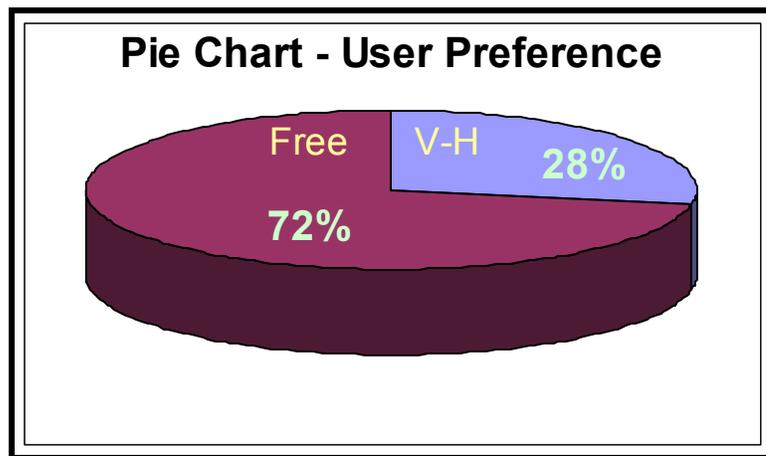


Figure V-8 Preference of the Participants (VH or F)

Clearly the participants expressed a preference for the F Method. However to test if other factors influenced this preference, the following analysis was conducted.

**4. Relationship Between Preference, Performance Time and Treatment Order.**

Figure V-9 displays the individual performance time differences conditioned by the expressed preference (VH or F). Only one person who preferred the VH method actually scored better in the vertical-horizontal time. On the other hand, as many as four participants scoring better in VH preferred the F method. This indicates that test times were not a factor in determining a participant’s preference.

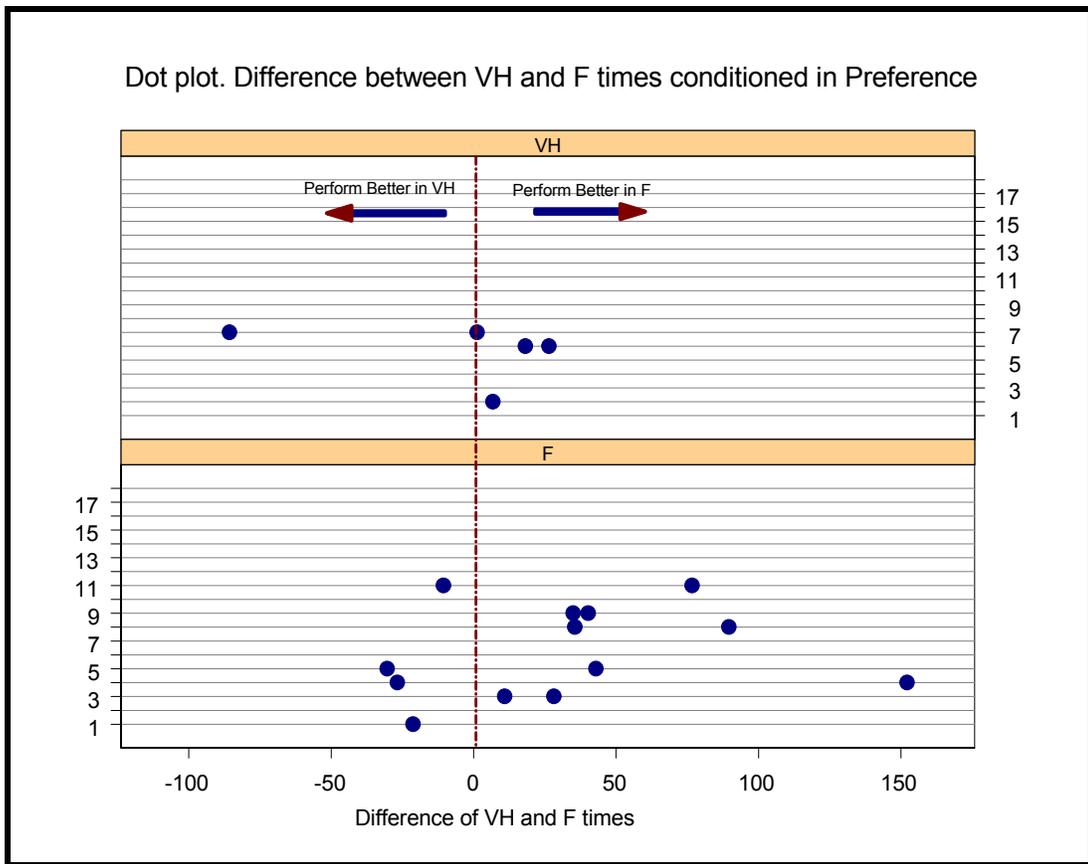


Figure V-9 Individual Performance Time Differences Conditioned by the Preference (VH or F)

Logistic Regression was performed to test if the preference had any relationship with the treatment order or performance times. Table V-3 presents the summary of the

regression. The p-values show that the order of the treatments (VH/F or F/VH) and the performance times were not significant to the expressed preference.

Coefficients	Value	Std. Error	T value	p-values
Intercept	3.90	2.90	1.30	
Order	-2.35	1.57	-1.49	0.13
VH time	-0.05	0.03	-1.59	0.11
F time	0.02	0.03	0.77	0.29

Table V-3 Logistic Regression Summary. Preference Against Order and Performance Times

In conclusion, the participants' preference did not seem to be affected by any external factor. Extrapolating to the entire EGGDT population, this result indicates that the F method is considered better by the potential users of the tool.

## G. CONCLUSION

A second formative experiment was performed to test the best method to draw edges in the EGGDT. A crossover experimental design helped to relieve the history threat that was identified for this experiment. The explanatory variables chosen were the difference in time to perform the related tasks with each method and the individual preference.

The prototype implemented the VH and F method. An application contained the prototype and provided accounting of tasks times and errors. This application also provided help for both methods. The participants had the opportunity to practice each method in a specific training window before performing the actual tasks of the experiment. Nineteen participants collaborated in the experiment in separate sessions that were supervised by three experiment controllers.

Based on the responses from the preference survey, as well as the lower times for the free draw arc construction method, and the conclusions of the statistical study above, the Free Draw Arc Method (F Method) was concluded to be the best choice for continued use and development of the EGGDT.

THIS PAGE INTENTIONALLY LEFT BLANK

## VI. FINAL EXPERIMENT: USERS' JUDGMENTS TOWARD GRAPHICAL DESIGN TOOLS

### A. INTRODUCTION

As stated in Chapter I, this thesis evaluates whether the use of graphical design tools improves the satisfaction of OR analysts developing simulation models. To test this hypothesis, a final experiment was performed using Version 0.3 of the Event Graph Graphical Design Tool (EGGDT). This chapter summarizes the experiment and analyzes the results. A description of the capabilities of the EGGDT Version 0.3 is also provided in paragraph D.

### B. PURPOSE OF THE EXPERIMENT

The purpose of this experiment was to determine if the use of the EGGDT, or any other graphical design tool, increases OR analyst satisfaction when developing simulation applications.

### C. DESIGN OF THE EXPERIMENT

The experiment was designed as follows:

N	R <sub>1</sub>	X <sub>1</sub>	O <sub>01...9</sub>
	R <sub>2</sub>	X <sub>2</sub>	O <sub>10...19</sub>

Participants were recruited from the population of NPS students (see Chapter III) so the initial selection was not random (N). Two random samples of the selected users (R<sub>1</sub> and R<sub>2</sub>) were obtained to perform two sessions (X<sub>1</sub> and X<sub>2</sub>). Finally, all participants filled out an individual questionnaire (O<sub>i</sub>).

#### 1. Explanatory Variables

Initially the explanatory variables hypotheses tested were:

- Participants' prior opinion of these kinds of tools is related to user satisfaction.
- Participants' preference for using manual versus computer-aided tool is related to user satisfaction with the tool.

The possible answers to these questions were "yes", "no" or "no opinion". After a pilot experiment, it became clear that answers to these two questions could not account

for differences since participants' responses were all the same. For this reason, it was decided to include two more explanatory variables in the questionnaire:

- Participants' perception of how the use of these tools can influence the OR field.
- Participants' feelings toward simulation after the session.

The choices in these questions were arranged from 1 to 5 using traditional Likert scaling. Likert scaling of responses allows for use of parametric statistics in the analysis.

## 2. Participants

The cooperation of nineteen participants in two evaluation sessions was obtained. These participants were representative of the group of users of the EGGDT as defined in Chapter III. The volunteers were OR students at the Naval Postgraduate School (NPS) in either their fourth or eighth quarter of school.

## D. PROTOTYPE DESIGN

Version 0.3 of the EGGDT was the final prototype developed for this thesis research, and the experiments covered in this chapter were performed with this program. The prototype implements the following features:

- Allow the creation of EGs and all their components (event, edges and simulation variables).
- Allow the drawing of EGs as graphical pictures as stated in Chapter III.
- Allow the deletion and modification of elements of EGs.
- Provide the functions "Save", "Save As", "Open" and "Rename".
- Generate the Java class matching the EG.

When the prototype starts, it shows a choice dialog in Figure VI-1. The user can choose to create a new EG or open an existing one.



Figure VI-1 Initial Choice Dialog of the EGGDT

If the “Create New EG” option is selected, the EGGDT shows the EG properties window in Figure VI-2. This version of the EGGDT only implemented the “Name” and “Directory” properties; the rest of the fields are considered informative. For example, if the user provides a project name, the EGGDT does not create a project.



Figure VI-2 Properties Window

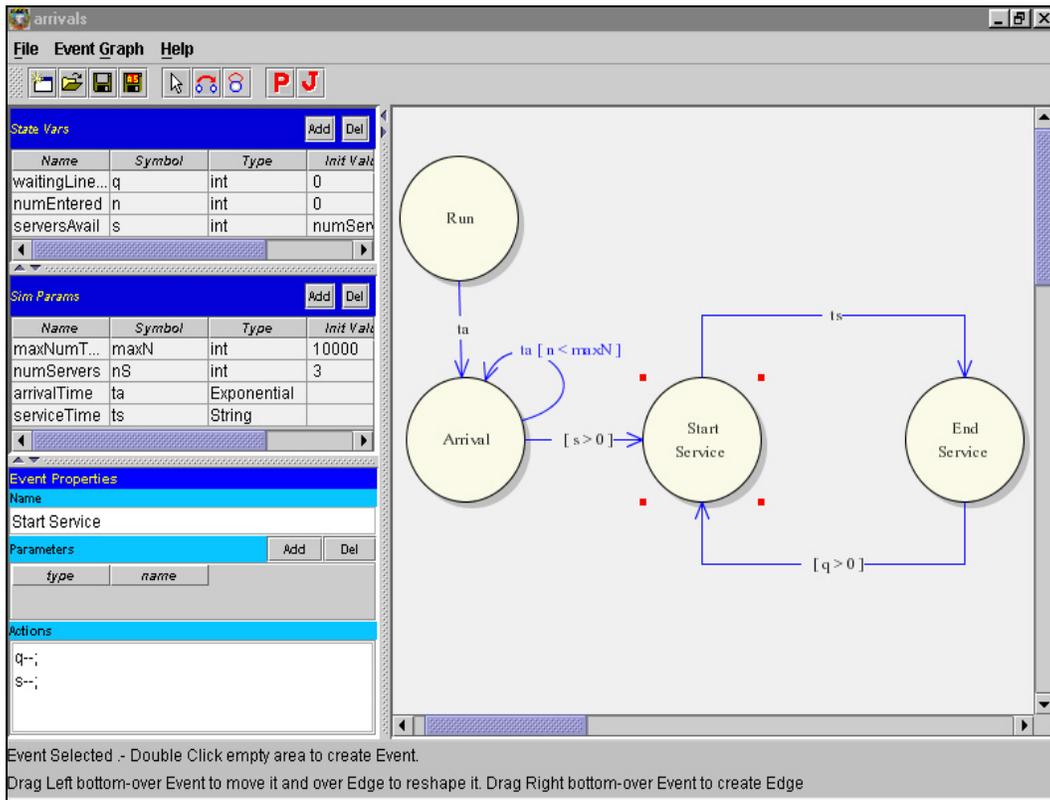


Figure VI-3 Snapshot of the EG for the “Arrival Model”

Elements of EGs (events, edges and simulation variables) can be created with the tool. An example of a “Arrival Model” is in Figure VI-3. The right side of the EGGDT is used to draw the graphical elements that represent events and edges.

The upper-left side contained the state variables and simulation parameters. The lower-left side showed the properties of the selected element. Properties of events included name, actions and input parameters. Properties of edges convey delay time, condition and arguments matching the input parameters of the target event.

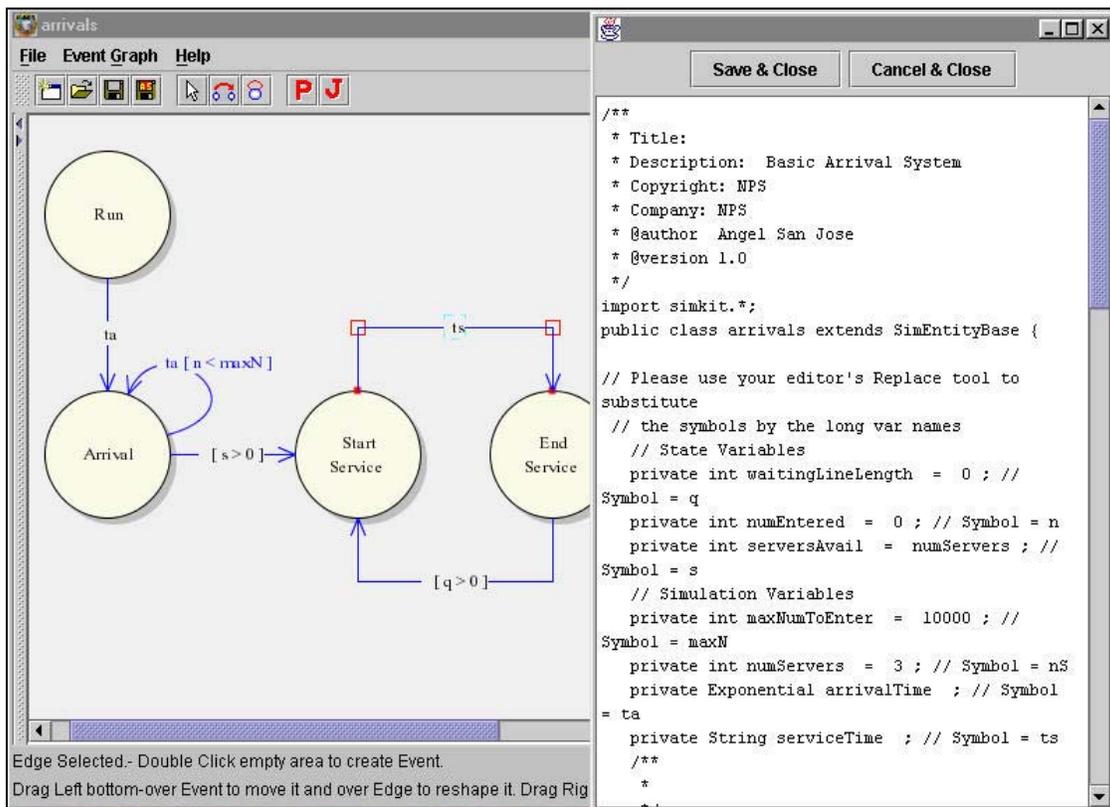


Figure VI-4 Snapshot of the Java Class Generated Code Window for the “Arrival Model”

The EGGDT generates the Java class source code on demand by pressing the  button. The source code window is shown in Figure VI-4. The source code window is not editable, but it can be saved, edited, compiled and run outside the EGGDT program.

The generated source code is not actually ready to be compiled, but it constitutes a skeleton of a Java program. It also provides the “get method” for state variables and

simulation parameters and “set method” for simulation parameters. Table VI-1 shows the complete Java class generated from the “Arrivals” model (header comments have been suppressed).

<pre> import simkit.*; public class arrivals extends SimEntityBase { // <b>State Variables</b> private int waitingLineLength = 0 ; // Sym = q private int numEntered = 0 ; // Sym = n private int serversAvail = numServers ; // Symbol = s // Simulation Variables private int maxNumToEnter = 10000 ; // Symbol = maxN private int numServers = 3 ; // Symbol = nS private Exponential arrivalTime ; // Symbol = ta private String serviceTime ; // Symbol = ts /** */ public void doRun ( ) { waitDelay("Arrival" , ta); } /** */ public void doArrival ( ) { n++; q++; if ( n &lt; maxN ) { waitDelay("Arrival" , ta); } if ( s &gt; 0 ) { waitDelay("StartService" , 0.0 ); } } /** */ public void doStartService ( ) { q--; s--; waitDelay("EndService" , ts); } /** */ public void doEndService ( ) { s++; if ( q &gt; 0 ) { waitDelay("StartService" , 0.0 ); } } } </pre>	<pre> // <b>*** SETTERS &amp; GETTERS ***</b> public int getWaitingLineLength ( ) { return waitingLineLength ; } public int getNumEntered ( ) { return numEntered ; } public int getServersAvail ( ) { return serversAvail ; } public int getMaxNumToEnter ( ) { return maxNumToEnter ; } public int getNumServers ( ) { return numServers ; } public Exponential getArrivalTime ( ) { return arrivalTime ; } public String getServiceTime ( ) { return serviceTime ; } public void setMaxNumToEnter ( int maxNumToEnter ) { this.maxNumToEnter = maxNumToEnter ; } public void setNumServers ( int numServers ) { this.numServers = numServers ; } public void setArrivalTime ( Exponential arrivalTime ) { this.arrivalTime = arrivalTime ; } public void setServiceTime ( String serviceTime ) { this.serviceTime = serviceTime ; } } </pre>
--	---

Table VI-1 Java Class Generated from the EG for the “Arrivals Model”

## E. EXPERIMENTAL PROTOCOL

The participants were gathered in the student laboratory, GL-318, in Glasgow Hall at the Naval Postgraduate School (NPS) on August 24 and 28, 2001. The principal investigator explained the purpose of the experiment and then passed out the informed consent form in Appendix E for the participants to read and sign.

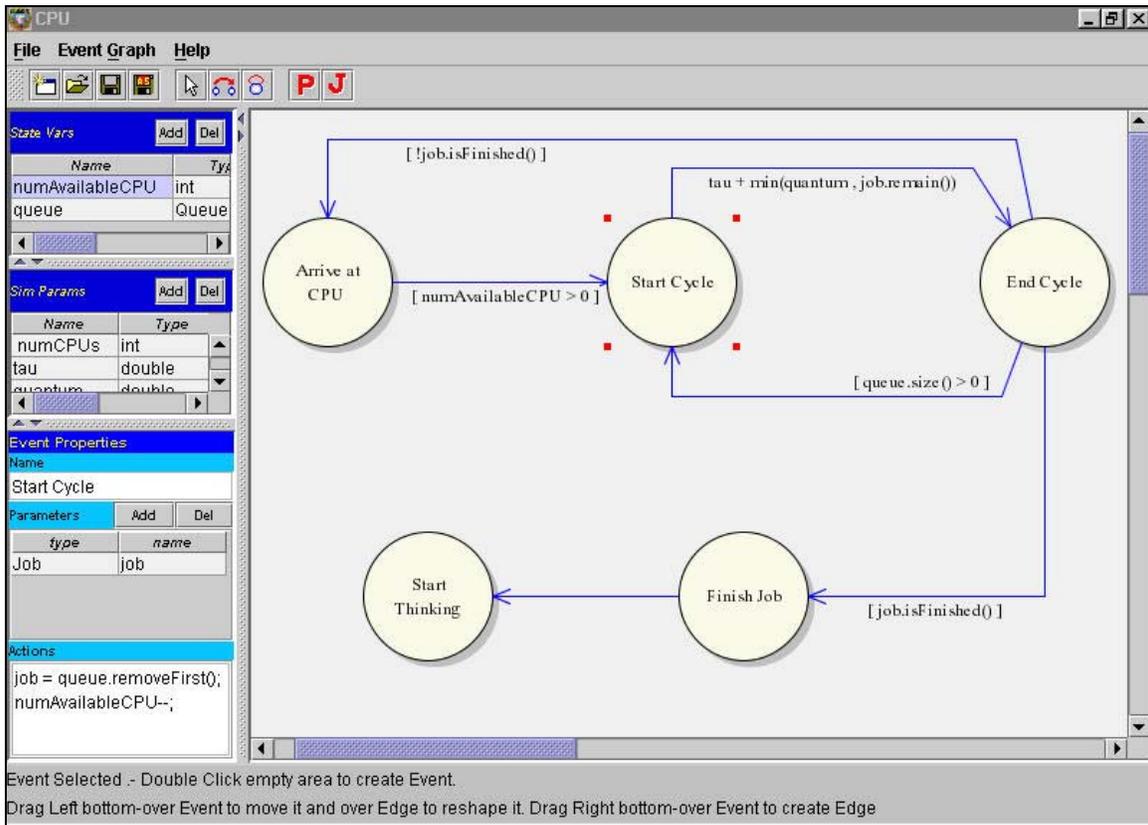


Figure VI-5 Snapshot of EG for the “CPU Model”

A quick tour through the tool was performed in a lecture fashion. The principal investigator explained how to use the EGGDT and the participants could follow the tasks on their computers. A snapshot of the example used to illustrate the capabilities of the EGGDT is in Figure VI-5. Participants were also allowed to ask questions about the tool. The principal investigator explained the possibilities of future versions of the EGGDT. Finally, participants filled out the questionnaires.

## F. STATISTICAL ANALYSIS

Analysis of the questionnaire data should that all participants felt that the EGGDT and similar tools could improve their satisfaction when developing simulation applications. Participants also expressed their unanimous preference for using these tools rather than manual methods to develop simulation models.

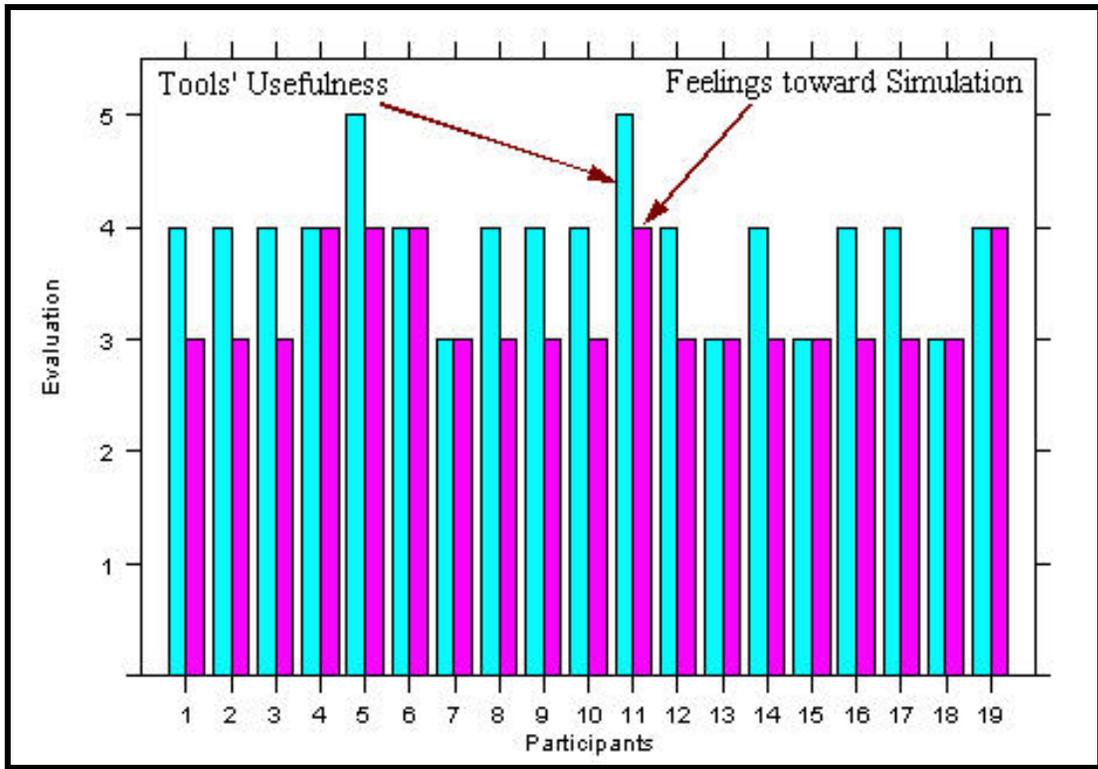


Figure VI-6 Participants' Responses to Questions II.C and II.D

Figure VI-6 shows participants' responses to questions II.C and II.D (see questionnaire in Appendix E)

- II.C. - After viewing this presentation, how useful do you think these tools will be in the OR field? (briefly, "Usefulness of the Tools").
- II.D. - Assuming you were skeptical about using simulation in your future OR projects, how would you rate your feelings about simulation after having seen this experiment? (briefly, "Feelings toward Simulation after".)

The choices for the question "Usefulness of the Tools" were:

1. These tools are not needed.

2. These tools are valid but not needed.
3. These tools could have limited utility in specific applications.
4. These tools have utility in a wide range of applications.
5. These tools are a major step forward in the OR field.

Thirteen out of nineteen participants thought the tool has utility in a wide range of applications; four participants consider the utility limited to specific applications, whereas, two of them expressed their belief that this tool is a major step forward in the OR field. These responses corroborate that participants were excited about these kinds of tools.

The choices for the question “Feelings toward Simulation After” were:

1. My feelings are unchanged; I probably would not use simulation.
2. These tools seem useful but I would only use them as a last resort.
3. With a tool like this, I think I could design some useful simulations with confidence.
4. I am very confidence that these tools will cause me to favor simulation in most OR projects.
5. I now believe that simulation should be my primary OR tool.

Fourteen out of nineteen participants chose answer number three. The remaining five participants chose answer number four. Participants were realistic about the possibilities of these tools; they appreciated the usefulness of these tools but were aware of their limitations.

The two questions received uniformly “good” grades. Figure VI-6 clearly shows that the variables were positive correlated. Each participant assigned the same or better grade to the question “Usefulness of the Tools” than to the question “Feelings toward Simulation After”. The actual correlation value is 0.547.

Table VI-2 summarizes the statistics for these two questions. The mean for “Usefulness of the Tools” (3.9) is slightly better than mean for “Feelings toward

Simulation After” (3.2). The standard deviations were very small (0.57 and 0.45), that indicates what participants agreed with these questions.

No external factors seem to influence the participants’ responses. The following two paragraphs discuss the participants’ responses to the computer proficiency and demographic questions.

	<b>Usefulness of the Tools</b>	<b>Feelings toward Simulation After</b>
Min	3	3
1 <sup>st</sup> Quartile	4	3
Mean	3.9	3.2
Median	4	3
3 <sup>rd</sup> Quartile	4	3.5
Max	5	4
Total #	19	19
Std. Dev.	0.57	0.45

Table VI-2 Summary Statistics for Questions II.C and II.D.

### 1. Computer Proficiency Questions

Participants were questioned about their opinions in simulation before and after the experiment. Figure VI-7 show the participants’ responses to question II.D “Feelings toward Simulation After” versus participant’s responses to question I.C “Feelings toward Simulation Before” the experiment. The scales in this graph have to be considered with caution because the multiple-choices used for both questions were different. The options for “Feelings toward Simulation After” were described above; the options for “Feelings toward Simulation Before” were:

1. I should have enrolled in the IT curriculum.
2. I could take them or leave them. I won’t use it again.
3. I think it can help to do some pretty cool stuff.
4. I like simulation
5. I think I going to focus my professional future in this area.

Most participants liked simulation before the experiment and they thought that tools like the EGGDT gave them more confidence to develop simulation. In conclusion, the future OR analysts are willing to use simulation in their projects and think that these kinds of tools can help them to do their jobs.

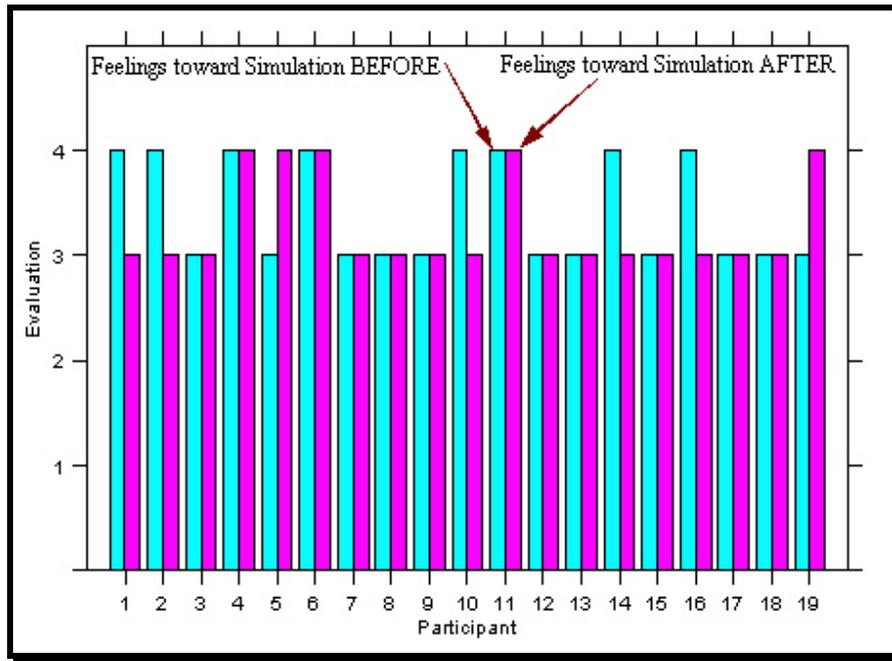


Figure VI-7 Participants' responses to questions I.C and II.D.

## 2. Demographical Questions

The data gathered from the demographical questions of the questionnaire (questions part III Appendix E) did not contribute to explaining the study results.

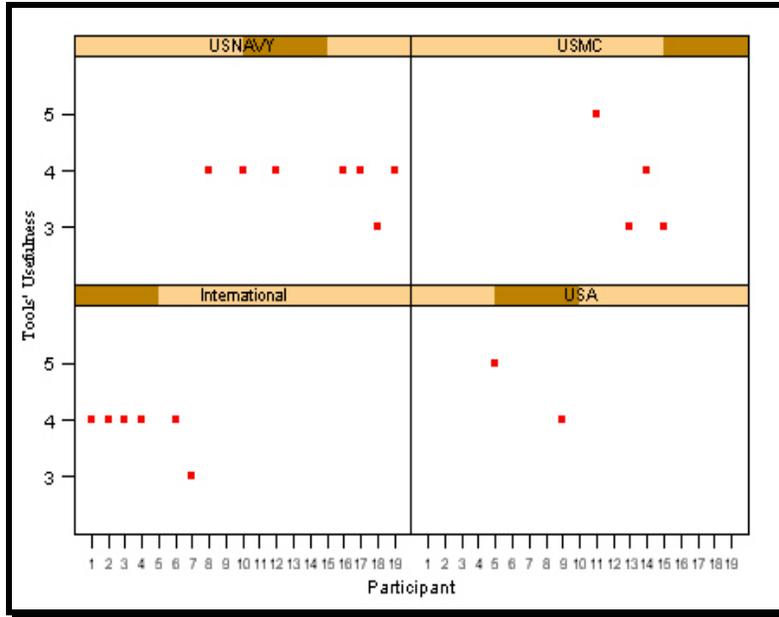


Figure VI-8 Participants' Responses to Question "Usefulness of the Tools" Conditioned by Military Branch

As an illustration, consider the graph in Figure VI-8 that shows the responses to "Usefulness of the Tools" question conditioned by military branch; no pattern of relationship between military branches and "Usefulness of the Tools" was identified.

## G. CONCLUSION

The thesis stated in Chapter I was that the use of simulation-design tools can improve OR analyst satisfaction when developing simulation models. A summative experiment was performed to prove this claim. For this experiment, Version 0.3 of EGGDT was used to illustrate what the simulation-design tools could do for OR analysts.

The EGGDT Version 0.3 implemented the most important requirements stated in Chapter III and Appendix B. It allowed the user to see a depiction of EGs with circles and arcs used to represent events and edges. It generated Java code from EG models. Finally, it offered the typical file management options, "Save", "Save As", "New", "Open" and "Rename".

The experiment was performed in two sessions. The experiment controller briefed the tool for 45 minutes and participants filled out a questionnaire. The data gathered from

these questionnaires clearly shows that participants believed that these kinds of tools can improve user satisfaction and they expressed their preference for using these tools. Participants stated that these kinds of tools are very useful in a wide range of OR applications and that these tools increase their confidence to develop simulation models.

In conclusion, from the data collected from the experiments, OR students believe that simulation-design tools improve OR analyst satisfaction when developing simulations.

## VII. CONCLUSIONS AND RECOMMENDATIONS

### A. CONCLUSIONS

This thesis demonstrated that the use of simulation-design tools improved the satisfaction of the Operation Research (OR) analysts while developing simulations to model OR problems. To test this statement it was necessary to provide a simulation-design tool to be used in the evaluative experiments. The Event Graph Graphical Design Tool (EGGDT) was developed because no other suitable tool was available. The EGGDT is a Computer Aided Software Design (CASE) program to develop Discrete Event Simulation (DES) models using Event Graphs (EG) and to generate the Java source code. The EGGDT depicts EGs using circles, arcs and tables for representation of events, edges and simulation variables.

To document the Analysis and Design (A&D) of the EGGDT, the Unified Modeling Language (UML) was used. UML provided a way to communicate A&D decisions. The UML artifacts utilized were Use-Case Diagrams, Conceptual Model Diagrams, Class Diagrams, Collaboration Diagrams and State Machines. Java was chosen to implement the EGGDT because Java has the required graphical capabilities and is a modern object-oriented multi-platform language.

Human Factors techniques were used to evaluate the EGGDT. The User-Centered approach was considered an appropriate paradigm. Two parallel software development flows were used, one for the application software and the other for the Graphical User Interface (GUI) software, allowed the separation between application domain and interface domain. This way decisions related to the GUI were not affected by application details leading to a more decoupled interface design. Using prototypical versions of the EGGDT, two formative experiments were conducted to obtain feedback from users during the EGGDT implementation.

The first formative experiment evaluated the initial approach chosen for the GUI of the EGGDT. The prototype implemented the more important GUI services, but it did not provide the related functionalities. Five students from the OR curriculum were

selected to perform a series of benchmark tasks with the prototype and to answer a questionnaire. The participants expressed their positive expectations about the tool. The benchmark tasks statistically demonstrated that the GUI of the prototype had a satisfactory learnability level. The initial design of the GUI of the EGGDT was considered valid; however, since participants encountered problems with the arc construction method, a second formative experiment was planned to test this particular issue.

The second experiment focused on the arc construction. Two methods were proposed and considered by the users. The experiment consisted of two similar batteries of tasks. Participants had to perform a given task battery with each method. Nineteen participants were recruited for the individual sessions. Times and errors were recorded and user preference were determined. Participants expressed their preference for the “Free Method” that allowed the drawing of arcs by specifying the end circles and any number of middle points. The benchmark times showed that participants did their tasks faster with the “Free Method” than they did with the other one. The “Free Method” was then selected to implement the EGGDT version 0.3 used for the summative experiment.

To test the claim that simulation-design tools help OR analysts who are developing simulation models, a summative experiment was performed using Version 0.3 of the EGGDT (which was the last version implemented during this thesis research). The experiment was based on a questionnaire that was filled out by participants after the principal investigator briefed the EGGDT as an example of a simulation-design tool. Two sessions were completed in one of the computer laboratories in Glasgow Hall at the Naval Postgraduate School (NPS). The nineteen participants agreed that these kinds of tools could improve their satisfaction in developing simulations; they also unanimously stated their preference for using these tools instead of manual methods. In addition, they expressed their opinion that these tools could be useful in a wide range of OR applications. Finally, they stated that tools like the EGGDT encouraged them to have more confidence when facing simulation projects.

In conclusion, this investigation shows that OR students at the NPS consider the EGGDT and similar tools an essential instrument when developing simulations. The

principal investigator also believes that this statement can be generalized to all OR analysts and to others involved in modeling and simulation projects and studies.

## **B. RECOMMENDATIONS**

The EGGDT is still in a very immature state; further development is necessary. The next iteration of the EGGDT development can include some the following:

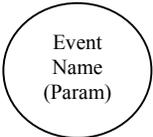
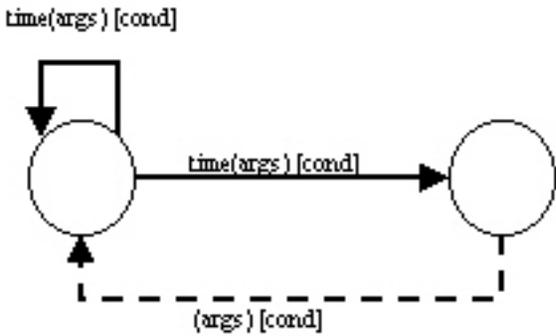
- Refactor of the EGGDT design, code and documentation.
- Implement canceling edges.
- Include the “Undo”, Redo”, “Copy” and “Paste” services.
- Implement a service to save EG models in XML format (version 0.3 of the EGGDT saves EG model in binary format).
- Improve the Java code generator.
- Design a project control approach that allows the creation of multiple related EGs.
- Define a complete methodology that includes the necessary phases to develop a simulation model, beginning with the requirements specification. The A&D phases could be driven by UML artifacts. The detail design of simulation classes has to involve building their EGs.
- Implement a reengineering tool that allows the extraction of EG models from source code of simulation Java classes implementing the Simkit interfaces.

The use of the EGGDT by the OR students at the NPS is an essential means of getting input from real users. The program can be run from the common drive of the OR department network and it is ready for downloading from the web side of NPS Loosely Coupled Components group (<http://diana.gl.nps.navy.mil/LCC/>) which is under the supervision of Professor Arnold Buss.

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX A. EG ELEMENTS

### A. EG ELEMENTS

Element	Representation/ Example	Definition/ Observations
State Variables	Alphanumeric String. delayTimeInQueue, numberInQueue1	Snapshot of the state of the simulation.
Events		Events represent a particular state transition in the system. [Buss96]
Edges		<p>Represent the scheduling of an event from other event. Two types:</p> <ul style="list-style-type: none"> <li>- Scheduling Edges;</li> <li>- Canceling edges, cancels the next occurrence of the event. (Dashed line)</li> </ul> <p>Attributes (all optional):</p> <ul style="list-style-type: none"> <li>- Delay time to schedule the target event. If Delay time is not present it is considered zero. It does not apply to Canceling Edges.</li> <li>- Arguments for the target event.</li> <li>- Condition to schedule the target event.</li> </ul>
Simulation Parameters	Alphanumeric String. numOfServers,	Simulation Parameters are constant during a simulation run.
Event Actions	Expressions { Q++ ; queue.add(customer) }	Sequence of operations (command) that are executed when the event is activated

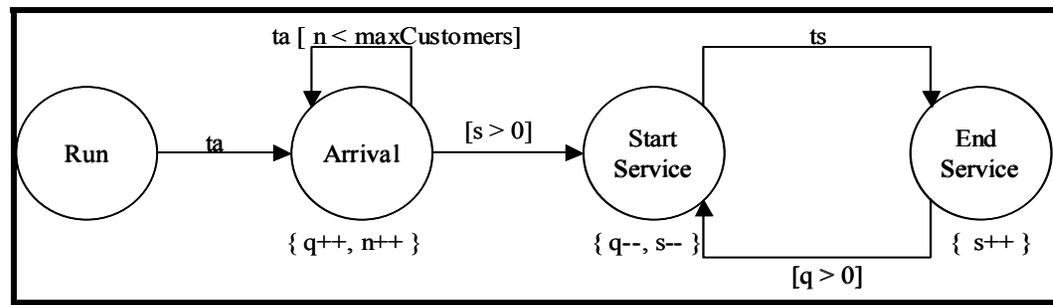
Delay Times	A number or acronym (t plus initial of the event name) ta, ts, 1.5	Expressed in time units. The units have to be uniform across the simulation.
Edge Conditions	A condition expression in square brackets. [ S > 0 ]	The event is scheduled if the condition is true at the time the source event is activated.
Event Parameters	A list of data types or classes' name in parenthesis ( int, boolean)	Input parameters types for the execution of the event action.
Edge Arguments	List of argument's names in parenthesis (n , is Ready)	Arguments for the target event. Must match event parameters.

## B. EG EXAMPLE: QUEUING SYSTEM

The model in the figure simulates a queuing process, such as supermarket cashier line or bank teller line, in which an “arrival” occurs every certain time “ta” (a value draw from a particular distribution); there are “S” servers that can only attend one arrival at a time. The service times of the servers are “ts” (values draw from other particular distribution). When the servers are busy (s=0), the pending arrival has to wait in a FIFO queue.

Parameters:	State Variables	Events / Actions
{ta} .- inter arrival time (sequence of random vars)	q .- number in the queue (init val = 0)	Run / nothing
{ts} .- inter service time (sequence of random vars )	s .- number of available servers.(init val = S)	Arrival / increment q and n by one
maxCustomers .- max number of customers.	n .- total number of arrivals (init val = 0)	Start Service / decrement q and s by one
S = total number of servers.		End Service / increment s by one

The simulation starts with the event "Run" that schedules an Arrival event in ta units of time (u.t). At time ta the event Arrival is activated; q++ and n++ are execute; if n is less than maxCutomers a new Arrival event is schedule in ta u.t.; if s is greater of zero, that is, at least one server is idle, a Start Service event is schedule with a zero delay. When Start Service event is activated, it executes its actions and schedules an End Service event in ts u.t. End Service increments number of servers idle and schedules a new Start Service event if the queue is not empty.



EG of the Arrival Model

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX B. EGGDT REQUIREMENTS

### A. FUNCTIONALITY REQUIREMENTS

The functionality requirements are broken down in the list below by reference number and function.

<b>RF.01<sup>6</sup></b>	<b>EG Graphic Tool</b>
<i>RF.01.01</i>	<i>Basic Functionality</i>
RF.01.01.01	Allow the user to start the EGGDT.
RF.01.01.02	Allow the user to Undo and Redo Actions.
RF.01.01.03	Allow the user to Copy and Paste.
<i>RF.01.02</i>	<i>EG Manipulation</i>
RF.01.02.01	Allow the user to Create a New EG
RF.01.02.02	Allow the user to Retrieve an EG
RF.01.02.03	Allow the user to Save an EG
RF.01.02.04	Allow the user to Print an EG
<i>RF.01.03</i>	<i>Event Manipulation</i>
RF.01.03.01	Allow the user to Create an Event
RF.01.03.02	Allow the user to Delete an Event
RF.01.03.03	Delete related Scheduling edges when Event Source or Target is deleted
RF.01.03.04	Allow the user to Specify Events Names
RF.01.03.05	Check Events Names are different
RF.01.03.06	Allow the user to Specify Event Input Parameters
RF.01.03.07	Allow the user to Specify Event Actions
RF.01.03.08	Check Event Actions refer only to State Variables
RF.01.03.09	Allow the user to Modify Events Names

---

<sup>6</sup> RF stands for Functionality Requirement

RF.01.03.11	Allow the user to Modify Event Input Parameters
RF.01.03.12	Allow the user to Modify Event Actions
RF.01.03.13	Check completeness of events after modification
<i>RF.01.04</i>	<i>Edge Manipulation</i>
RF.01.04.01	Allow the user to Create Scheduling Edges
RF.01.04.02	Allow the user to Create Canceling Edges
RF.01.04.03	Check Edges have a Source Event and a Target Event (they can refer to the same Event for Scheduling Edges, but not for Canceling Edges)
RF.01.04.04	Allow the user to Delete Scheduling Edges
RF.01.04.05	Allow the user to Delete Canceling Edges
RF.01.04.06	Allow the user to Specify the Scheduling Delay Time. Default delay time is zero.
RF.01.04.07	Allow the user to Specify Edge Arguments.
RF.01.04.08	Check Edge Arguments match Target Event Input Parameters
RF.01.04.09	Allow the user to Specify Edge Conditions
RF.01.04.10	Allow the user to Modify the Scheduling Delay Time
RF.01.04.11	Allow the user to Modify Edge Arguments.
RF.01.04.12	Allow the user to Modify Edge Conditions
RF.01.04.13	Check Edge completeness after modification
<i>RF.01.05</i>	<i>Simulation Entity Parameters and Entity State Variables Manipulation</i>
RF.01.05.01	Allow the user to Define Simulation Parameters; A name has to be provided
RF.01.05.02	Allow the user to Define State Variables; A name has to be provided
RF.01.05.03	Check Names and Acronyms of Simulation Parameters and State Variables are all different
RF.01.05.04	Allow the user to Delete Simulation Parameters
RF.01.05.05	Allow the user to Delete State Variables

RF.01.05.06	Report any expression (Edge Condition, Edge Argument or Event Action) that is invalidated when a Simulation Parameters or State Variables is deleted.
RF.01.05.07	Allow the user to specify Simulation Parameters and State Variables Acronyms
RF.01.05.08	Allow the user to specify Simulation Parameters and State Variables Types
RF.01.05.09	Allow the user to specify Simulation Parameters and State Variables Initial Values
RF.01.05.10	Allow the user to delete Simulation Parameters and State Variables Acronyms
RF.01.05.11	Allow the user to delete Simulation Parameters and State Variables Types
RF.01.05.12	Allow the user to delete Simulation Parameters and State Variables Initial Values
RF.01.05.13	Report any expression (Edge Condition, Edge Argument or Event Action) that is modified when a Simulation Parameters or State Variables property is modified or deleted.
<b>RF.02</b>	<b>EG design Acceptance</b>
<i>RF.02.01</i>	<i>Configuration Management</i>
RF.02.01.01	Mark EG's as Configuration Items
RF.02.01.02	Keep track of EG versions
<b>RF.03</b>	<b>Code Generator</b>
<i>RF.03.01</i>	<i>Manual Code Insertion</i>
RF.03.01.01	Allow the user to write the Java code of the Events ("do" methods)
<i>RF.03.02</i>	<i>Automatic Code Generation</i>
RF.03.02.01	Generate class skeleton for the EG
RF.03.02.02	Generate "get" and "set" methods for the simulation parameters
RF.03.02.03	Generate "get" (no "set") methods for the simulation State Variables

RF.03.02.04	Generate “do” methods skeleton from Events
RF.03.02.05	Generate “waitDelay” instructions inside “do” methods of Source Events from Scheduling Edges Delay Times
RF.03.02.06	Generate “if blocks” inside “do” methods of Source Events from Edges Conditions
RF.03.02.07	Generate “interrupt” instructions (with argument the Target Event) inside Source Events from Canceling Edges.
RF.03.02.08	Paste the code of the class – including the manually introduced -- and generate an ASCII file with name <EGname.java>
<b>RF.04</b>	<b>Code Acceptance</b>
<i>RF.04.01</i>	<i>Configuration Management</i>
RF.04.01	Keep track of Code versions related to the EG versions

## B. GUI REQUIREMENTS (TASK ANALYSIS)

The following table specifies the interface tasks that the user has to be provided by the tool.

<b>RI.01<sup>7</sup></b>	<b>EG Graphic Tool</b>
<i>RI.01.01</i>	<i>Basic Tasks</i>
RI.01.01.01	Provide the user with an interface to “Start” the tool
RI.01.01.02	Provide the user with an interface to “Exit” the tool
RI.01.01.03	Provide the user with an interface to “Undo” and “Redo” actions
RI.01.01.04	Provide the user with an interface to “Copy” and “Paste”
RI.01.01.05	Provide the user with configurable “Drawing Grid”
RI.01.01.06	Provide the user with an interface for “Configuration Management”
<i>RI.01.02</i>	<i>EG Manipulation Tasks</i>
RI.01.02.01	Provide the user with an interface to “Create New EG”
RI.01.02.02	Provide the user with an interface to “Retrieve EG”
RI.01.02.03	Provide the user with an interface to “Save EG”
RI.01.02.04	Provide the user with an interface to “Print EG”
RI.01.02.05	Provide the user with a resizable “Drawing Area”
RI.01.02.06	Provide the user with an interface to “Layout the EG” that optimizes the layout of Events and Edges figures
<i>RI.01.03</i>	<i>Event Manipulation Tasks</i>
RI.01.03.01	Provide the user with an interface to “Create Event”
RI.01.03.02	Draw events in the specified screen’s position
RI.01.03.03	Provide the user with an interface to “Delete Event”
RI.01.03.04	Delete form screen related Scheduling Edges when their Source or Target Event is deleted
RI.01.03.05	Provide the user with an interface to “Modify Event Position”

<sup>7</sup> RI stands for Interface Requirement

RI.01.03.06	Provide the user with an interface to “Name Event”
RI.01.03.07	Provide the user with an interface to “Define Event Input Parameters”
RI.01.03.08	Provide the user with an interface to “Define Event Actions”
RI.01.03.09	Provide the user with an interface to “Define priority of Events”
<i>RI.01.04</i>	<i>Edge Manipulation Tasks</i>
RI.01.04.01	Provide the user with an interface to “Create Scheduling Edge”
RI.01.04.02	Provide the user with an interface to “Create Canceling Edge”
RI.01.04.03	Draw Edges on the screen connecting them to their corresponding Source and Target Events.
RI.01.04.04	Provide the user with an interface to “Modify Edge Shape”
RI.01.04.05	Provide the user with an interface to “Modify Edge’s Source or Target Event”
RI.01.04.06	Provide the user with an interface to “Specify Scheduling Edge Delay Time”
RI.01.04.07	Provide the user with an interface to “Specify Edge Arguments”
RI.01.04.08	Provide the user with an interface to “Specify Edge Conditions”
RI.01.04.09	Provide the user with an interface to “Delete Edge”
<i>RI.01.05</i>	<i>Simulation Entity Parameters and Entity State Variables Manipulation Tasks</i>
RI.01.05.01	Provide the user with an interface to “Define Simulation Entity Parameter”
RI.01.05.02	Provide the user with an interface to “Define Simulation Entity State Variable”
RI.01.05.03	Provide the user with an interface to “Delete parameter or State Variable”
RI.01.05.04	Provide the user with an interface to specify Acronyms, Types, and Initial Values for Parameters and State Variables.
RI.01.05.05	Show Parameters and State Variables in screen in an organized way
RI.01.05.06	Provide the user with an interface to inspect Parameters and State Variables cross references with Events and Edges

<b>RI.02</b>	<b>Code Generator Tasks</b>
RI.02.01	Provide the user with an interface to “Generate Java Code”
RI.02.02	Provide the user with an interface to “Edit Code”

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX C. FIRST EXPERIMENT FORMULARIES AND DATA

### A. TASK LIST (PARTICIPANT'S VERSION)

- A. Describe the menus and buttons.
- B. Create events : “Run”, “Arrival” and “Start”.
- B1. Delete event “Start”.
- C. Create an Edge between events “Run” and “Arrival” with delay time “ta”.
- D. Create an Event named “Leave”.
- E. Create an edge between events “Arrival” and “Leave”, with delay time “tl” and condition “finishes == true”.
- F. Modify the Event Graph layout by aligning all events horizontally, with edges as straight lines and in the following order from left: Run-Arrival-Leave.
- G. Rename event “Run” to “Start”.
- H. Delete event “Leave”.
- I. Create an edge between “Arrival” and “Start” with condition “IamFedUp == true” ( now there is an edge from “Start” to “Arrival” and other form “Arrival” to “Start”).
- J. Modify position of Edge from “Start” to “Arrival” with delay time “ta” so that it connects to events vertically on top and then extends horizontally between them. (Edge to look like “”).
- K. Modify position of Edge from “Start” to “Arrival” with delay time “tl” so that it connects to events vertically on bottom and then extends horizontally between them. (edge to look like “”).
- L. Delete Event “Arrival”.
- M. Create an Event named “AlmostDone”.
- N. Create an edge between ”AlmostDone”. and “Start” with condition “IamHavingFun == true”.

## B. TASK LIST (EVALUATOR'S VERSION)

**Benchmark 1** (measure task performance):

- A. Describe the menus and buttons.
- B. Create events : “Run” and “Arrival”.

*Intervening nonbenchmark tasks:*

- B1. Delete event “Start”.

**Benchmark 2** (measure task performance time):

- C. Create an Edge between events “Run” and “Arrival” with delay time “ta”.

*Intervening nonbenchmark tasks:*

- D. Create an Event named “Leave”.
- E. Create an edge between events “Arrival” and “Leave”, with delay time “tl” and condition “finishes == true”.

**Benchmark 3** (measure task performance time)

- F. Modify the Event Graph layout by aligning all events horizontally, with edges as straight lines and in the following order from left: Run-Arrival-Leave.

**Benchmark 4** (measure task performance time)

- G. Rename event “Run” to “Start”.

*Intervening nonbenchmark tasks:*

- H. Delete event “Leave”.
- I. Create an edge between “Arrival” and “Start” with condition “!amFedUp == true”. ( now there is an edge from “Start” to “Arrival” and other form “Arrival” to “Start”).

**Benchmark 5** (measure task performance time)

- J. Modify position of Edge from “Start” to “Arrival” with delay time “ta” so that it connects to events vertically on top and then extends horizontally between them. (Edge to look like “”).

*Intervening nonbenchmark tasks:*

- K. Modify position of Edge from “Start” to “Arrival” with delay time “tl” so that it connects to events vertically on bottom and then extends horizontally between them. (edge to look like “”).
- L. Delete Event “Arrival”.
- M. Create an Event named “AlmostDone”.

**Benchmark 6 – final task** - (measure task performance time)

- N. Create an edge between “AlmostDone” and “Start” with condition “IamHavingFun == true”.

**C. INFORMED CONSENT FORM**

You are going to become a research participant for our evaluation of the design for a new interactive computer system. This evaluation is being conducted by Angel San Jose (OR) and Paulo Silva (CS01) and is part of a project for the MV4203 - Interactive Computation Systems and the thesis of Angel San Jose. Angel SanJose, who will be glad to answer you any questions about this evaluation session, will run your evaluation session. As a participant, you have some rights, which are listed bellow.

You will be asked to sit in from of a Lab PC and perform a number of tasks associated with the systems. We are evaluating the system, to make it as effective and usable as possible. We are not in any way evaluating you. We expect the session to last about 30 minutes. The evaluator will be monitoring you throughout the session to obtained the required feed back from the session. Your name will not be associated with any data collected. There are no known risks associated with this evaluation. You will be given a written tasks list and a small questionnaire at the end for you to express you opinion.

Your rights as a participant are as follows:

1. You have the right to withdraw from the session at any time.
2. At the end of you session, you may see your data, if you so desire. If you wish to withdraw your data at that point, inform the evaluator. Otherwise, it might be impossible because of our efforts to keep anonymity.
3. Ensure you do not comment this session with any other participants that have not yet performed their session.

Finally, we greatly appreciate your time and effort for participating and helping us in this evaluation. Remember, this is not a pass-fail assessment; instead it is a useful contribution to the development of this computer system. Your signature bellow indicates that you have read and understand the contents of this consent form and that you entirely agree to participate.

Do you have any questions?

NAME \_\_\_\_\_

SIGNATURE \_\_\_\_\_

DATE \_\_\_\_\_

**D. SESSION INSTRUCTIONS**

We thank you very much for your time and participation in this experiment. You help will enable us to evaluate this novel tool that will help you creating Event Graphs (EG).

As you can observe, this a simple drawing tool that will be used to draw an EG and will be able to run in any platform which has a Java Run time environment.

You have been designing simulations models using EGs which you draw by hand, and translated to Java using Simkit also manually. The propose of this tool is to allow you to

create an EG and to automatically generate the corresponding Java Class using the Simkit library.

The prototype only permits to draw, modify and delete events and edges --scheduling but not canceling nor self edges. The rest of the functions of the final tool are not yet implemented. Particularly you can: draw and name an event, modify its position in the screen, delete an event, create an edge between to events – arrow heads are not yet implemented, introduce the edge properties (delay times, arguments and conditions), modify the shape of an edge and delete edges.

A different notation, with respect to the one learned in the Simulation class, has been introduced. The edges are not curves but segments of straight lines and the properties of the edge (delay time, arguments and conditions) have to be written in the same line with the format: *delay time (argument list) [condition list]*.

During this evaluation, you will be asked to perform a number of specific tasks using this system. Then we will give you some free time to play around with the tool, exploring it the way you want. We then will ask you to perform a few more specific tasks and finally.

The list of tasks will be given to you in written format and we ask you to read each task aloud and to make sure you understand it before you begin. In order to obtain the best possible feedback from your participation, we will probably be timing how well the system is helping you on those tasks. We therefore ask you to work through each individual task without stopping. When you are done with one task you can then relax before going into the next one.

While you are doing your tasks, we ask you to think aloud. By doing that we can therefore observe if the system is helping you the way you expect or if it is lacking some behavior or causing any difficulties to your actions. Feel completely free to comment all bad and good aspects of the interface as you go along. The more comments you tell us the more helpful will your participation be.

This is by no means an evaluation of your self. Rather, you are only helping us evaluating this tool so that we can improve it to meet your best expectations for a tool like this.

When you are done with the task list, we will ask you to complete a short questionnaire to rate the system and to express your overall opinion about it.

We expect this session to last 30 minutes in total.

Before you start, do you have any question?

## **E. QUESTIONNAIRE FOR THE USER INTERFACE SATISFACTION**

For each of the following questions, fill in 0-9 or leave blank if question is not applicable)  
Skip question if not applicable

**REMEMBER, YOU ARE NOT EVALUATING US BUT THE TOOL.**

**SINCERITY IS APPRECIATED**

**1. Overall Reaction to the Event Graph Design Tool**

<i>terrible</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>wonderful</i>
<i>difficult</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>easy</i>
<i>frustrating</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>satisfying</i>
<i>dull</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>stimulating</i>
<i>rigid</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>flexible</i>

**2. Screen**

Characters on the computer screen

<i>hard to read</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>easy to read</i>
---------------------	----------------------------	---------------------

Tool bar with buttons

<i>hard to read</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>easy to read</i>
---------------------	----------------------------	---------------------

Organization of information on screen

<i>confusing</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>very clear</i>
------------------	----------------------------	-------------------

**3. Terminology and System Information**

Computer terminology is related to the task you are doing

<i>never</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>always</i>
--------------	----------------------------	---------------

Menu Items and Tool bar with buttons function identification

<i>difficult to identify</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>easy to read</i>
------------------------------	----------------------------	---------------------

**4. Learning**

Learning to operate the system

<i>difficult</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>easy</i>
------------------	----------------------------	-------------

Exploring new features by trial and error

<i>difficult</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>easy</i>
------------------	----------------------------	-------------

Remembering names and use of commands

<i>difficult</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>easy</i>
------------------	----------------------------	-------------

Tasks can be performed in a straightforward manner

<i>never</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>always</i>
--------------	----------------------------	---------------

**5. System Capabilities**

System speed

<i>too slow</i>	<i>0 1 2 3 4 5 6 7 8 9</i>	<i>fast enough</i>
-----------------	----------------------------	--------------------

System reliability

*unreliable*    0 1 2 3 4 5 6 7 8 9    *reliable*

Correcting your mistakes

*difficult*    0 1 2 3 4 5 6 7 8 9    *easy*

Experienced and inexperienced users' needs are taken into consideration

*never*    0 1 2 3 4 5 6 7 8 9    *always*

**COMMENTS**

**F. USABILITY SPECIFICATION TABLE**

<b>Usability Attribute</b>	<b>Measuring Instrument</b>	<b>Value to be Measured</b>	<b>Current Level (Secs)</b>	<b>Worst Acceptable Level(Secs)</b>	<b>Planned Target Level (Secs)</b>	<b>Best Possible Level (Secs)</b>	<b>Observed Results (Secs)</b>
Initial performance	“Create an Event” task per Benchmark 1	Length of time to successfully create an event	60 (using standard graphical tool) 5 (by hand)	60	50	20	P1..5 = 80 / 57 / 49 / 43 / 47 Mean = 55.2 Stdv = 14.77
Initial performance	“Create an Edge” task per Benchmark 2	Length of time to successfully create an edge	120 5	120	60	20	P1..5 = 113 / 120 / 120 / 47 / 17 Mean = 83.4 Stdv = 48.19
Initial performance	“Modify EG layout” task per Benchmark 3	Length of time to successfully modify EG layout	60 120	60	30	10	P1..5 = 60 / 20 / 99 / 56 / 50 Mean = 57.0 Stdv = 28.25
Initial performance	“Rename Event” task per Benchmark 4	Length of time to successfully rename event	20	20	10	5	P1..5 = 10 / 9 / 2 / 4 / 45

Usability Attribute	Measuring Instrument	Value to be Measured	Current Level (Secs)	Worst Acceptable Level(Secs)	Planned Target Level (Secs)	Best Possible Level (Secs)	Observed Results (Secs)
performance	task per Benchmark 4	successfully rename and Event	20				Mean = 14 Stdv = 7.65
Initial performance	“Modify Edge” task per Benchmark 5	Length of time to successfully modify Edge	60 60	60	20	40	
Learnability	Repeat “Create an Edge” task at the end of the session per Benchmark 6	Length of time to successfully create an edge	120 5	120	60	20	P1..5 = 37 / 21 / 37 / 28 / 33 Mean = 31.2 Stdv = 6.80
First impression	Participant responds to QUIS Questionnaire	Average score (From 0 to 9)	Not know 5 but low	5	7	9	P1..5 = 7.0 / 6.9/ 8.3 /7.5 /8.1 Grand Mean = 7.54

## APPENDIX D. SECOND EXPERIMENT FORMS AND DATA

### A. PARTICIPANT CONSENT FORM

1. **Introduction.** You are invited to participate in a comparative experiment of arcs modalities to be used in the Event Graph Graphical Design Tool (EGGDT) . With information gathered from you and other participants we will develop a tool for designing Event Graph for simulation models. We ask you to read and sign this form indicating that you agree to be in the study. Please ask any questions you may have before signing.
2. **Background Information.** This experiment is part of the thesis research currently carried out by Angel San Jose (OA curriculum) and the final project of the team A for the OA-3401 course.
3. **Procedures.** If you agree to participate in this study, the researcher will launch a computer application. Every step of the experiment will be explained in detail in the different windows. The researcher will be available to help you in the preliminary phases. You are expected to perform the benchmark tasks by yourself. In any case if you think you cannot complete the task, ask for assistance.
4. **Risks and Benefits.** This research involves no risks or discomforts greater than those encountered in ordinary computer work. The final product will benefit OR analysts.
5. **Compensation.** No tangible reward will be given. You will see your results at the conclusion of the experiment.
6. **Confidentiality.** The records of this study will be kept confidential. No information will be publicly accessible which could identify you as a participant.
7. **Voluntary Nature of the Study.** If you agree to participate, you are free to withdraw from the study at any time without prejudice. You will be provided a copy of this form for your records.
8. **Points of Contact.** If you have any further questions or comments after the completion of the study, you may contact the research supervisor, Dr. Nita Miller (telephone 656-2281) or the Principal Investigator Angel San Jose (aesanhos@nps.navy.mil).
9. **Statement of Consent.** I have read the above information. I have asked all questions and have had my questions answered. I agree to participate in this study.

-----  
Participant's Signature

-----  
Date

-----  
Researcher's Signature

-----  
Date



**C. PRIVACY ACT STATEMENT**

NAVAL POSTGRADUATE SCHOOL, MONTEREY, CA 93943

**PRIVACY ACT STATEMENT**

1. Authority: Naval Instruction
2. Purpose: Comparison between arcs modalities to be used in the Event Graph Graphical Design Tool (EGGDT) .
3. Use: The arcs modality selected will be used in the EGGDT to allow the user to draw edges connecting events.
4. Disclosure/Confidentiality:
  - a. I have been assured that my privacy will be safeguarded. I will be assigned a control or code number, which thereafter will be the only identifying entry on any of the research records. The Principal Investigator will maintain the cross-reference between name and control number. It will be decoded only when beneficial to me or if some circumstances, which is not apparent at this time, would make it clear that decoding would enhance the value of the research data. In all cases, the provisions of the Privacy Act Statement will be honored.
  - b. I understand that a record of the information contained in this Consent Statement or derived from the experiment described herein will be retained permanently at the Naval Postgraduate School or by higher authority. I voluntarily agree to its disclosure to agencies or individuals indicated in paragraph 3 and I have been informed that failure to agree to such disclosure may negate the purpose for which the experiment was conducted.
  - c. I also understand that disclosure of the requested information, including my Social Security Number, is voluntary.

---

Signature of Volunteer Name, Grade/Rank (if applicable) DOB SSN Date

---

Signature of Witness Date

**D. COMPUTER PROFICIENCY SURVEY**

Participant ID Number \_\_\_\_\_ Computer \_\_\_\_\_

Test A Time \_\_\_\_\_ Test A Errors \_\_\_\_\_ First ? \_\_\_\_\_  
Test B Time \_\_\_\_\_ Test B Errors \_\_\_\_\_ First ? \_\_\_\_\_

1. How many years of basic computer use do you have? \_\_\_\_\_
2. What is your perceived computer experience level?
  - a. None
  - b. A little experience
  - c. Moderate
  - d. I can make computers do what a I want
  - e. Expert
3. Do you have computer at home? \_\_\_\_\_
4. Rate your proficiency in graphical or presentation software (i.e. Powerpoint, Harvard Graphics)
  - a. None
  - b. I don't know that much about these software titles
  - c. Moderate proficiency
  - d. I can build a strong presentation
  - e. I can make the screen dance a fine jig
5. Rate your computer mouse agility.
  - a. Mouse? I didn't think that we were testing hygienic products on lab animals!
  - b. Poor
  - c. Average
  - d. Good
  - e. Excellent
6. Which statement best describes your feelings towards computers?
  - a. Someone get me a sledgehammer.
  - b. I could take them or leave them.
  - c. I think they can do some pretty cool stuff.
  - d. I like computers.
  - e. I would marry one.

Part II – Arc Preference

1. Which of the two modes of arc manipulation in the experiment do you prefer?
  - a. Direct draw
  - b. Right Angle
2. Have you participated in an experiment related with this EGGDT program?

Part III – Demographics

1. How old are you? \_\_\_\_\_

2. Which is your dominant hand or are you ambidextrous? \_\_\_\_\_
3. What is your gender? \_\_\_\_\_
4. Do you prefer to use a computer mouse with your right or left hand?  
\_\_\_\_\_
5. You can configure a computer mouse to be used either “left-handed” or “right-handed”. Which configuration do you use? \_\_\_\_\_

Part IV -- Comments

Do you have any recommendations for improving the arc manipulation methods?

---

---

---

## E. DATA

Symbol	Data Table Name	Explanation
#	#	Participant's order
ID	ID	Combination of # and Controller
Controller	Controller	Initial of the experiment's controller
Lab	Lab	Laboratory (Human Factors or Loosely Coupled Components)
VH.Time	VH.Time	Time to perform VH treatment
F.Time	F.Time	Time to perform F treatment
Dif.VH.F	Time.Diff.VH.F	VH.Time - F.Time
E.VH	E.VH	Errors in VH treatment
E.F	E.F	Errors in F treatment
Order	Order	Order of treatments (VH/F or F/VH)
Exp	Exper.years	Computer experience
Level	Level	Self expressed computer level
Home PC	Home.PC	Participant has PC at home
Graph Exper	Graphical	Self expressed experience with graphical environments
Mouse DEX	Mouse.DEX	Self expressed mouse dexterity
PC Emp	PC.Emp	Feelings towards computers
Pref	Pref	Self Expressed preference VH or F

Age	Age	Participant's age
RH domi	RH.Dominant	Right hand dominant
RH Mouse	RH.Mouse.User	Right hand mouse user
Mouseconfi	RH.Mouse.Config	Right hand mouse configuration
Gender	Male.Gender	Participant's gender

#	ID	Controler	Lab	VH.Time	F.Time	Dif. VH.F	E.VH	E.F	Order	Exp	Level	Home PC	Graph Exper	Mouse DEX	PC Emp	Pref	Age	RH domi	RH Mouse	Mouse Config	Gender
1	B-01	B	HSIL1	215.00	236.26	-21.26	0.00	0.00	VH/F	15.00	3.00	1	3.00	4.00	4.00	F	33.00	1	1	1	1
2	P-02	P	HSIL1	89.43	82.66	6.77	0.00	0.00	F/VH	14.00	4.00	0	4.00	4.00	3.00	VH	30.00	1	1	1	1
3	P-03	P	HSIL1	79.22	68.32	10.91	0.00	0.00	VH/F	10.00	3.00	1	3.00	4.00	4.00	F	30.00	0	1	1	1
4	A-01	A	LCC	103.90	75.70	28.20	0.00	0.00	VH/F	12.00	3.00	1	3.00	4.00	3.00	F	35.00	1	1	1	1
5	A-02	A	LCC	258.10	106.00	152.10	0.00	0.00	F/VH	15.00	4.00	1	5.00	4.00	4.00	F	33.00	1	1	1	1
6	A-03	A	LCC	143.25	116.80	26.45	0.00	0.00	VH/F	15.00	3.00	1	3.00	4.00	3.00	VH	34.00	1	1	1	1
7	A-04	A	LCC	80.00	61.80	18.20	0.00	0.00	F/VH	13.00	5.00	1	5.00	5.00	3.00	VH	27.00	1	1	1	1
8	A-05	A	LCC	104.80	131.60	-26.80	0.00	2.00	VH/F	15.00	3.00	1	3.00	3.00	3.00	F	29.00	1	1	1	1
9	P-01	P	HSIL2	152.24	182.59	-30.35	0.00	0.00	VH/F	20.00	4.00	1	4.00	5.00	1.00	F	36.00	1	0	0	1
10	B-02	B	HSIL1	85.65	171.38	-85.72	0.00	0.00	F/VH	18.00	4.00	1	3.00	3.00	2.00	VH	29.00	0	1	1	1
11	P-05	P	HSIL1	96.60	53.68	42.92	0.00	2.00	VH/F	15.00	3.00	1	4.00	4.00	4.00	F	30.00	1	1	1	1
12	P-04	P	HSIL1	244.35	154.75	89.60	0.00	0.00	F/VH	15.00	3.00	1	3.00	3.00	2.00	F	30.00	1	1	1	0
13	A-06	A	LCC	134.30	98.80	35.50	0.00	0.00	F/VH	23.00	4.00	0	3.00	4.00	2.00	F	43.00	0	1	1	1
14	A-07	A	LCC	124.40	89.50	34.90	0.00	0.00	VH/F	20.00	4.00	1	5.00	5.00	4.00	F	38.00	1	1	1	0
15	A-08	A	LCC	134.90	94.70	40.20	0.00	0.00	F/VH	15.00	4.00	1	4.00	4.00	4.00	F	30.00	1	1	1	1
16	A-09	A	LCC	188.30	111.60	76.70	0.00	0.00	F/VH	3.00	2.00	1	2.00	3.00	4.00	F	17.00	1	1	1	1
17	B-03	B	HSIL1	77.65	76.44	1.21	0.00	0.00	F/VH	18.00	4.00	1	4.00	5.00	3.00	VH	35.00	1	1	1	1
18	A-10	A	LCC	89.40	99.96	-10.56	0.00	0.00	VH/F	10.00	4.00	1	5.00	4.00	3.00	F	33.00	1	1	1	0

## APPENDIX E. FINAL EXPERIMENT FORMS AND DATA

### A. PARTICIPANT CONSENT FORM

1. **Introduction.** You are invited to participate in an evaluation experiment of the Event Graph Graphical Design Tool (EGGDT) . With information gathered from you and other participants I will improve my tool for designing Event Graph for simulation models. I ask you to read and sign this form indicating that you agree to be in the study. Please ask any questions you may have before signing.
2. **Background Information.** This experiment is part of the thesis research currently carried out by Angel San Jose (OA curriculum).
3. **Procedures.** If you agree to participate in this study, you will be provided with a computer in which the EGGDT application will be install. I will explain the use and possibilities of the current EGGDT prototype version. You will have the opportunity to follow me in a tour through the application (this step will take a quarter of hour approximately). After this you will be able to play with the tool by yourself. I will be available for answering question in any moment. As a last task I will provide a questionnaire about the application for you to fill.
4. **Risks and Benefits.** This research involves no risks or discomforts greater then those encountered in ordinary computer work. The final product will benefit OR analysts. In the short term it will benefit particularly those of you who are or will be involved in simulation thesis researches.
5. **Compensation.** No tangible reward will be given
6. **Confidentiality.** The records of this study will be kept confidential. No information will be publicly accessible which could identify you as a participant.
7. **Voluntary Nature of the Study.** If you agree to participate, you are free to withdraw from the study at any time without prejudice .
8. **Points of Contact.** If you have any further questions or comments after the completion of the study, you may contact the research supervisor, Dr. Nita Miller (telephone 656-2281) or the Principal Investigator Angel San Jose (aesanjose@nps.navy.mil).
9. **Statement of Consent.** I have read the above information. I have asked all questions and have had my questions answered. I agree to participate in this study.

-----  
Participant's Signature

-----  
Date

-----  
Researcher's Signature

-----  
Date





## D. COMPUTER PROFICIENCY SURVEY

### I. Computer proficiency

- A. What is your perceived COMPUTER level?
1. None.
  2. A little experience
  3. Moderate.
  4. Good
  5. Expert
- B. Which statement best describes your feelings towards COMPUTERS?
1. Someone get me a sledgehammer.
  2. I could take them or leave them
  3. I think they can do some pretty cool stuff.
  4. I like computers
  5. I would marry one.
- C. Which statement best describes your feelings towards SIMULATION?
1. I should have enrolled in the IT curriculum.
  2. I could take them or leave them. I won't use it again.
  3. I think it can help to do some pretty cool stuff.
  4. I like simulation
  5. I think I going to focus my professional future in this area.

### II. Simulation Design tools evaluation.

- A. Do you think this tool, or tools like this, can improve your satisfaction when developing simulation applications?
1. Yes.
  2. No.
  3. No opinion.
- B. To develop simulation models, do you prefer to use the methods you have been using so far or some kind of tool like this?
1. Current tools.
  2. Tools like this.
  3. No opinion.
- C. After viewing this presentation, how useful do you think these tools will be in the OR field.
1. These tools are not needed.
  2. These tools are valid but not needed.
  3. These tools could have limited utility in specific applications.
  4. These tools have utility in a wide range of applications.
  5. These tools are a major step forward in the OR field.
- D. Assuming you were skeptical about using simulation in your future OR projects; how would you rate your feelings about simulation after having seen this experiment?
1. My feelings are unchanged; I probably would not use simulation.
  2. These tools seem useful but I would only use them as a last resort.
  3. With a tool like this, I think I could design some useful simulations with confidence.

4. I am very confidence that these tools will cause me to favor simulation in most OR projects.
5. I now believe that simulation should be my primary OR tool.

**III. Demographics**

- A. How old are you? \_\_\_\_\_
- B. What is your gender? \_\_\_\_\_
- C. Military branch
  1. US Navy.
  2. USMC.
  3. US Army.
  4. USAF.
  5. International.
  6. Other \_\_\_\_\_
- D. Curriculum/quarter \_\_\_\_\_

**IV. Comments**

YOUR SUGGESTIONS ARE VERY IMPORTANT.  
Do you have any recommendations for improving the EGGDT?

**E. DATA**

Num	Session	Level	FeelCom	FeelSim	Satisfaction	Preference	Useful	NewFeelSim	Age	Gender	Branch	Curriculum	Quarter
1	2	4	4	4	1	2	4	3	26	M	5	OR	4
2	2	4	2	4	1	2	4	3	43	M	5	OR	4
3	2	4	3	3	1	2	4	3	35	M	5	OR	4
4	2	4	4	4	1	2	4	4	28	M	5	OR	4
5	2	3	4	3	1	2	5	4	32	M	3	OR	4
6	2	3	4	4	1	2	4	4	26	M	5	OR	4
7	2	4	3	3	1	2	3	3	34	M	3	OR	4
8	2	3	3	3	1	2	4	3	28	M	1	OR	8
9	2	4	3	3	1	2	4	3	30	M	3	OR	4
10	2	4	4	4	1	2	4	3	32	F	1	OR	4
11	1	3	4	4	1	2	5	4	28	M	2	OR	8
12	1	3	4	3	1	2	4	3	30	F	1	OR	8
13	1	3	4	3	1	2	3	3	34	M	2	OR	8
14	1	3	3	4	1	2	4	3	40	M	2	OR	4
15	1	4	3	3	1	2	3	3	30	M	2	OR	4
16	1	4	4	5	1	2	4	3	39	F	1	OR	6
17	1	3	3	3	1	2	4	3	38	M	1	OR	8
18	1	3	3	3	1	2	3	3	32	M	1	OR	4
19	1	3	3	3	1	2	4	4	33	F	1	OR	8

## F. FIELDS EXPLANATION

Num	Participant's order number
Session	Two session took place
Level	Participant's Computer level
FeelCom	Participant's feelings toward computers
FeelSim	Participant's feelings toward simulation
Satisfaction	Participant's opinion if IDE tools improve satisfaction
Preference	Participant's preference between old tools or IDE tools
Useful	Participant's opinion in usefulness of the IDEs
NewFeelSim	Participant's feelings toward Simulation after the session
Age	Participant's age
Gender	Participant's gender
Branch	Participant's military branch
Curriculum	Participant's curriculum
Quarter	Participant's quarter

THIS PAGE INTENTIONALLY LEFT BLANK

## LIST OF REFERENCES

[BUS00] Component-Based Simulation Modeling / Arnold H. Buss / Proceedings of the 2000 Winter Simulation Conference/ 2000.

[BUS01] Basic Event Graph Modeling / Arnold H. Buss / Simulation News Europe – Issue 31/ 2001.

[BUS96] Modeling with Event Graphs / Arnold H. Buss / Proceedings of the 1996 Winter Simulation Conference / 1996.

[COK97] Surviving Object-Oriented Projects / Alistair Cockburn / Addison-Wesley Object Technology Series/ 1997.

[FOW00] UML Distilled / Martin Fowler / Addison-Wesley Object Technology Series/ 2000.

[GAM95] Design Pattern: Elements of Reusable Object-Oriented Software. / Gamma, E., R. Helm, R. Johnson, and J. Vlissides / Addison-Wesley / 1995.

[GEA00] Graphic Java. Mastering the JFC. Volume 2. Swing / David M. Geary / Sun Microsystems Press / 1999.

[GRA98] Patterns in Java: a Catalog of Reusable Design Patterns Illustrated with UML / Mark Grand / Wiley Computer Publishing / 1998.

[HAN98] UML Toolkit. / Hans-Erik Eriksson & Magnus Penker / John Wiley & Sons, Inc / 1998.

[HAR00] Java 2D API Graphics / Vincent J. Hardy / Sun Microsystems Press / 2000.

[HIX93] Developing User Interfaces: Ensuring Usability Thought Product & Process / Devorath Hix et al / John Wiley & Sons, inc / 1993.

[JAB99] Unified Software Development Process / Ivar Jacobson / Addison-Wesley Object Technology Series / 1999.

[KOR97] Modeling and Simulation with UML and Java / Enrique V. Korthright / IEEE Paper / 1997.

[LAR98] Applying UML and Patterns. An Introduction to OO Analysis and Design / Craig Larman / Prentice Hall / 1998.

[MAX00] An Overview of the Joint Warfare System (JWARS) / Daniel T. Maxwell / PHALANX September / 2000.

[RAS00] The Humane Interface: New Directions for Designing Interactive Systems / Jef Raskin / ACM Press / 2000.

[RUM99] The Unified Modeling Language Reference Manual / James Rumbaugh et al / The Addison-Wesley Object Technology Series / 1999.

[SCH83] Simulation Modeling with Event Graphs / Lee W. Schruben / Communications of the ACM, 26, 957-963 / 1983.

[SCH95] Graphical Simulation Modeling and Analysis: Using Sigma for Windows / Lee W Schruben. / The Scientific Press Series / 1995.

[SHN92] Designing the User Interface: Strategies for Effective Human-Computer Interaction / Shneiderman, B. / Addison-Wesley / 1992.

[STO96] Sensors in Object Oriented Discrete Event Simulation / Kirk A. Stork / NPS Thesis (MS. In Operations Research) / 1996.

## INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center  
Ft. Belvoir, VA 22060-6218
2. Dudley Knox Library  
Naval Postgraduate School  
Monterey, CA 93943-5101
3. Chairman, Code OR  
Operations Research Department  
Naval Postgraduate School  
Monterey, California  

---
4. Dr. Arnold Buss  
Operations Research Department  
Naval Postgraduate School  
Monterey, California  

---
5. Dr. Gordon Bradley  
Operations Research Department  
Naval Postgraduate School  
Monterey, California  

---
6. Dr. Nita Miller  
Operations Research Department  
Naval Postgraduate School  
Monterey, California  

---
7. CN Jefe Gabinete de Investigacion Militar Operativa de la Armada.  
Spanish Navy  

---
8. CC. Antonio Gonzalez García  
Spanish Navy  

---
9. CC. Jose Paes.  
Brazilian Navy  

---

10. CC. Paulo R. Silva  
Portuguese Navy

---