A Simulation-Based Task Analysis using Agent-Based, Discrete Event and System Dynamics simulation

2015

Anastasia Angelopoulou

University of Central Florida

Find similar works at: https://stars.library.ucf.edu/etd

University of Central Florida Libraries http://library.ucf.edu

STARS Citation

https://stars.library.ucf.edu/etd/5146

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of STARS. For more information, please contact lee.dotson@ucf.edu.
A SIMULATION-BASED TASK ANALYSIS USING AGENT-BASED, DISCRETE EVENT AND SYSTEM DYNAMICS SIMULATION

by

ANASTASIA ANGELOPOULOU
BSc Electrical Computer Engineering, 2011
MSc Modeling and Simulation, 2013

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Modeling and Simulation in the College of Engineering and Computer Science at the University of Central Florida Orlando, Florida

Fall Term
2015

Major Professor: Waldemar Karwowski
© 2015 Anastasia Angelopoulou
ABSTRACT

Recent advances in technology have increased the need for using simulation models to analyze tasks and obtain human performance data. A variety of task analysis approaches and tools have been proposed and developed over the years. Over 100 task analysis methods have been reported in the literature. However, most of the developed methods and tools allow for representation of the static aspects of the tasks performed by expert system-driven human operators, neglecting aspects of the work environment, i.e. physical layout, and dynamic aspects of the task. The use of simulation can help face the new challenges in the field of task analysis as it allows for simulation of the dynamic aspects of the tasks, the humans performing them, and their locations in the environment.

Modeling and/or simulation task analysis tools and techniques have been proven to be effective in task analysis, workload, and human reliability assessment. However, most of the existing task analysis simulation models and tools lack features that allow for consideration of errors, workload, level of operator's expertise and skills, among others. In addition, the current task analysis simulation tools require basic training on the tool to allow for modeling the flow of task analysis process and/or error and workload assessment. The modeling process is usually achieved using drag and drop functionality and, in some cases, programming skills.

This research focuses on automating the modeling process and simulating individuals (or groups of individuals) performing tasks in a dynamic work environment in any domain. The main objective of this research is to develop a universal tool that allows for modeling and simulation of task analysis models in a short amount of time with limited need for training or
knowledge of modeling and simulation theory. A Universal Task Analysis Simulation Modeling (UTASiMo) tool can be used for automatically generating simulation models that analyze the tasks performed by human operators.

UTASiMo is a multi-method modeling and simulation tool developed as a combination of agent-based, discrete event, and system dynamics simulation models. A generic multi-method modeling and simulation framework, named 3M&S Framework, as well as the Unified Modeling Language have been used for the design of the conceptual model and the implementation of the simulation tool. UTASiMo-generated models are dynamically created during run-time based on user inputs. The simulation results include estimations of operator workload, task completion time, and probability of human errors based on human operator variability and task structure.
This dissertation is dedicated to my family.
ACKNOWLEDGMENTS

Though this dissertation is an individual work, I could not have achieved to manage challenging moments without the support and guidance of so many people. First of all, I would like to thank my advisor and chairman of my committee, Dr. Karwowski, for his support and demand for excellence. His encouragement and inspiration assisted me in achieving my goals. I would also like to thank the members of my committee, Dr. Hancock, Dr. Xanthopoulos and Dr. Kincaid and Dr. Shumaker for their time, encouragement, and expertise throughout this project. I would like to acknowledge that this dissertation was supported in part by Grant No. N00014-14-1-0777, Modeling Human Performance in STAMPS, from the Office of Naval Research.

I am most grateful to my wonderful parents, Konstantinos and Helen, and my brothers, Athanasios and Pantelis, who supported and encouraged me during this process and over the years! Thank you to my family for always believing in me, encouraging me and supporting me throughout this journey. I love each and every one of you! Special thanks to Dr. Konstantinos Mykoniatis for his support and encouragement during this process and over the years. I couldn’t have done it without you! Finally, I would like to thank all of my friends at University of Central Florida and Greece for their encouragement!
# TABLE OF CONTENTS

LIST OF FIGURES .............................................................................................................. xi

LIST OF TABLES ................................................................................................................ xii

LIST OF MEDIA/ABBREVIATIONS/NOMENCLATURE/ACRONYMS ................................. xiii

CHAPTER 1: INTRODUCTION ............................................................................................ 1

Research Problem ........................................................................................................... 3

Limitations of current task analysis methods ................................................................. 3

Limitations of current simulation tools ........................................................................... 4

Simulation models for analysis of tasks .......................................................................... 5

Discrete Event Simulation Models ................................................................................... 5

Agent-based Simulation Models ....................................................................................... 6

System Dynamics Simulation Models ............................................................................. 7

Multi-method Simulation Models ................................................................................... 7

Research Objectives ....................................................................................................... 8

Research Methodology ................................................................................................... 9

Expected Results ........................................................................................................... 9

Overview of this research ............................................................................................. 10

CHAPTER 2: REVIEW OF TASK ANALYSIS LITERATURE ........................................... 12
Brief History of Task Analysis .................................................................12
Hierarchical Task Analysis .......................................................................14
Cognitive Task Analysis ...........................................................................15
Spatial Operational Sequence Diagrams. ..................................................16
Timeline Analysis ..................................................................................16
Review of Existing Task Analysis Software Tools.......................................17
TaskArchitect ..........................................................................................17
MicroSaint ..............................................................................................18
Work Models that Compute ......................................................................18
Enhanced Operator Function Model. ..........................................................18
MIDAS. ....................................................................................................18
Universal Task Analysis Methodology .......................................................21
Components of task analysis considered in UTA and UTASiMo design ..........24
Tasks .......................................................................................................25
Work Environment ..................................................................................26
Human Operator ......................................................................................26
Workload Estimation ..............................................................................27
Human Error ............................................................................................28

CHAPTER 3: METHODOLOGY AND CONCEPTUAL MODEL DESIGN ............30
Conceptual Model ...........................................................................................................................................30
Unified Modeling Language (UML). .................................................................................................................31
Universal Task Analysis (UTA). .........................................................................................................................31
Discrete Event Module: Task Structure and Environment ..................................................................................32
Agent-Based Module: Agent Structure and Workload Estimation .................................................................34
Agent Structure ..................................................................................................................................................35
Workload Estimation .........................................................................................................................................38
System Dynamics Module: Human Error Probability Estimation .................................................................38
Human Error Estimation .....................................................................................................................................40
CHAPTER 4: UTASiMo DEVELOPMENT ........................................................................................................42
UTASiMo Overview ..........................................................................................................................................42
Simulation Application Interface. .......................................................................................................................44
The Automatically Generated UTASiMo Models. .............................................................................................47
CHAPTER 5: CASE STUDIES ..........................................................................................................................49
Case Study 1: Plant Simulation ............................................................................................................................49
Case study 1 simulated with UTASiMo. ..............................................................................................................52
Case Study 1 simulated with MicroSaint ..........................................................................................................56
Case Study 2 .....................................................................................................................................................59
Simulation of naval task. ....................................................................................................................................60
Validation of naval task. .................................................................63

CHAPTER 6: DISCUSSION AND FUTURE WORK ............................................70

Research Summary ..................................................................................70

Research Contribution ...........................................................................71

Limitations and Future Work ..................................................................72

APPENDIX: LIST OF SUBMITTED AND PUBLISHED PAPERS ..................74

REFERENCES ..........................................................................................76
LIST OF FIGURES

Figure 1. Agent's internal state .................................................................27
Figure 2. Use Case Diagram illustrating the overall UTA process ..................32
Figure 3. System dynamics simulation model.............................................40
Figure 4. UML Activity Diagram describing the main functions of UTASiMo and the flow of the multi-method simulation process .......................................................................43
Figure 5. UTASiMo's initial page created in AnyLogic.................................45
Figure 6. Animation view of the automatically generated simulation model ..........46
Figure 7. Results produced by the simulation model......................................47
Figure 8. Overall Task Flow ........................................................................50
Figure 9. UTASiMo-generated model when one human operator performs the tasks ........53
Figure 10. (a) UTASiMo-generated model's simulation results for a single run with one operator (b) UTASiMo-generated model's simulation results for 122 replications with one operator ......54
Figure 11. Simulation model created with MicroSaint Simulation software ............57
Figure 12. Overall naval task flow.................................................................61
Figure 13. Automatically generated model for naval task ..................................62
Figure 14. Simulation results for naval task ..................................................63
Figure 15. Workload comparison between real-world and simulated system...........64
Figure 16. Average workload per operator....................................................65
Figure 17. Average Time per Task per operator.............................................65
LIST OF TABLES

Table 1. Comparison among Task Analysis Tools .................................................................20
Table 2. List of UTA Concepts ...............................................................................................22
Table 3. Summary of elements ...............................................................................................25
Table 4. Task Structure ..........................................................................................................34
Table 5. Agent Structure ......................................................................................................37
Table 6. Partial List of Input to Excel Spreadsheet .................................................................51
Table 7. Summary of Results for scenario simulated with UTASiMo .................................56
Table 8. Summary of Results for case study 1 simulated with different tools .......................59
Table 9. Comparison of real-world and simulated system for navel task .........................64
Table 10. Comparison of average time per task between the real and the simulated system ...67
Table 11. Comparison of average workload between the real and the simulated system for each operator ........................................................................................................68
Table 12. Comparison of errors between the real and the simulated system for each task ......69
LIST OF MEDIA/ABBREVIATIONS/NOMENCLATURE/ACRONYMS

3M&S     Multi-Method Modeling and Simulation
AB       Agent Based
CTA      Cognitive Task Analysis
DES      Discrete Event Simulation
HF       Human Factors
HSI      Human System Integration
HTA      Hierarchical Task Analysis
SD       System Dynamics
UML      Unified Modeling Language
UTA      Universal Task Analysis
UTASiMo  Universal Task Analysis Simulation Modeling
CHAPTER 1:
INTRODUCTION

Task Analysis is a methodology that studies the activities an operator (or a group of operators) needs to perform to accomplish a goal. The primary purpose of task analysis is the identification of the tasks the operator needs to perform, their decomposition into primary task steps, and the comparison of the task demands to the operator's capabilities in order to describe human performance in a particular task or scenario. This process helps the analyst to better understand the human element in the system and reduce errors.

Over the last decades, there is a growing interest in analyzing human activity and human operator behavior to arrive at better system designs. As technology advances and tasks become more demanding, human work changes increasing the need to create new methods and tools for task analysis to face the new challenges. A variety of task analysis approaches and tools have been proposed and developed over the years. According to Diaper and Stanton [1], over 100 task analysis methods have been reported in the literature. Each method is selected based on different criteria, such as the purpose and scope of the analysis, as well as task, cost, and time factors, among others [2], [3]. However, most of the developed methods and tools allow for representation of the static aspects of the tasks performed by expert system-driven human operators, neglecting aspects of the work environment, i.e. physical layout and dynamic aspects of the task. The use of simulation can help face the new challenges in the field of task analysis as it allows for simulation of the dynamic aspects of the tasks, the humans performing them, and their locations in the environment.
The current research provides a review of the most commonly applied task analysis methods and their current limitations: Hierarchical Task Analysis (HTA), Cognitive Task Analysis (CTA), Spatial Operational Sequence Diagrams (SOSD), and Timeline Analysis (TA). It also proposes a novel task analysis simulation tool which integrates basic concepts from HTA, CTA, SOSD, TA, human reliability assessment, and workload estimation to provide a more comprehensive approach that fills the gaps in existing task analysis methods and can be universally applicable.

More specifically, this research focuses on automating the modeling and simulation of individuals (or groups of individuals) performing tasks with a stochastic structure in any domain while incorporating workload prediction and human error estimation. A simulation tool, named Universal Task Analysis Simulation Modeling (UTASiMo), has been developed for automatically constructing task models and analyzing the tasks to estimate workload and human error. UTASiMo [85-87] is a multi-method modeling and simulation application developed in AnyLogic and it is a combination of agent-based (AB), discrete event (DES), and system dynamics (SD) simulation models. UTASiMo dynamically generates models that estimate operator workload and probability of human errors, based on human operator variability and task structure. The simulation model for a specific system is automatically constructed based on a spreadsheet in which the user can specify task and human locations, required resources, available time to execute each task, task frequency, and other information related to humans, tasks, and the system. The user can then experiment with the generated model. One of the major advantages of automating the modeling process is that it allows the user to build a simulation model for a real
world system in a short amount of time without being proficient in simulation as required by current simulation packages.

**Research Problem**

This research addresses the problem of how to create and simulate universal task analysis models without the need for training, programming skills, or modeling and simulation theory to produce results for error estimations and human operator workload, while considering human operator variability and dynamic task structure. In the following subsections, we examine the limitations of current task analysis methods and tools to elaborate further on the research problem.

**Limitations of current task analysis methods.** Task analysis methods are widely used in a range of domains, including healthcare, manufacturing industry, and military. The popularity of task analysis methods is a product of their effectiveness and flexibility. However, they are not flawless. The focus of the current methods is usually on a limited subset of interactions, i.e. study only physical actions or only cognitive processes. For example, HTA does not include the analysis of cognitive processes, while CTA models do not always capture physical capabilities. Moreover, the kind of analysis provided by the current techniques varies among the methods. HTA is mainly a descriptive method rather than analytical, while TA requires the system under analysis to be relatively predictable and repeatable. Finally, current task analysis techniques work under the assumption that the human operator is an expert and knows how to perform a task [13], [14]. In general, task analysis cannot simultaneously represent multiple human
operators with varying levels of skills and expertise [16]. This limitation can be overcome using agent-based simulation, which allows for simultaneous representation of multiple users with diverse characteristics.

**Limitations of current simulation tools.** Task analysis is still a "static" process. It is usually conducted using pen and paper or using audio and video recording devices. Although various tools have been created in the field of task analysis for data collection and organization, few simulation tools have been developed for task analysis. These tools are either directly related to task analysis or are generic-purpose tools that may be used for task analysis [5], [6].

TaskArchitect [7] is one tool that allows for data collection, analysis, and evaluation, while supporting the export function to modeling and simulation tools. TaskArchitect also provides templates for Human Error Analysis (SHERPA). MicroSaint [10] is a general purpose discrete event simulation tool that has been used for task modeling and simulation of human performance. MicroSaint has been used in military and other commercial companies to model tasks and answer human performance-related questions. However, it requires elementary programming skills [12]. It also lacks the capability to model the heterogeneity of agents across a population, which is one of the distinguishing features of AB simulation compared to DES and SD. Other task analysis tools include Work Models that Compute (WMC) [17], the extension of the Operator Function Model (EOFM) [18], and MIDAS [19].

The tools listed above allow for documentation, modeling, and/or simulation of the task analysis process, but they require training and, in some cases, usage of programming languages, thereby requiring time and effort to produce results. Moreover, not all of the tools model the heterogeneity of agents across a population or provide built-in estimators for human error and
workload. Therefore, more adequate and widely accessible simulation tools need to be developed that will be able to perform a more comprehensive task analysis [20].

**Simulation models for analysis of tasks.** Simulation models are important decision making tools within organizations [4]. In the context of Human Factors (HF) and Human System Integration (HSI), simulations are usually human behavior representations of a task sequence, facility, environmental, and other parameters, aiming to resolve issues relevant to the human element and its integration with other system components. Recent advances in technology have increased the need for using simulation models to analyze tasks and obtain human performance data. Limited research has studied the development of DES, AB, and SD simulation models for task analysis.

**Discrete Event Simulation Models.** Discrete Event Simulation (DES) is an M&S approach that represents a system over time by modeling the system as a sequence of processes performed across entities at discrete points in time [72]. DES was first established as a simulation method at IBM in the 1960s. This method is characterized by the graphical representation of the model as a flowchart, where blocks represent the processes of how entities move through the system [73], [74]. Although the representation is static, the model can be randomized by using probability distributions to the model inputs.

A DES model is a logical representation of a real-world queuing system. This representation may not be an exact replication of the actual system, but a simplified and reasonable representation of the most relevant system components. Relevant system components refer to the elements that, if altered, can measurably affect the performance measures of the system. The design of a DES model requires the identification of the key model components and
the flow of entities through the system. It also requires information about the number and capacity of queues and servers, the behavior of servers, and the time it takes for service providers to perform particular tasks. DES is a process of multiple simulation runs, meaning that the analyst may determine, within a confidence interval, how the model performs under certain conditions or alternative configurations of the system.

Applications cover a variety of domains, including manufacturing, logistics, and business process modeling [73]. DES is particularly useful as a performance analysis tool for identifying bottlenecks and collecting statistics on process performance [74] due to its flowchart-like design. The main disadvantage is that the entities simply follow a process with no autonomy, which results in limited capabilities to adapt and learn [73], [74], in contrast to ABS. Thus, a combination of the two methods provides more capabilities.

The most commonly used tool for discrete event task analysis simulation models found in the literature is MicroSaint. The MicroSaint models utilize task analysis to study human factors problems. Examples include the use of the tool to estimate crew workload, fatigue, and overall system performance [91], as well as to build a task network for the evaluation of crew size on safety in a nuclear power plant [92].

**Agent-based Simulation Models.** Agent-based (AB) simulation is a relatively new modeling approach that focuses on agents and the analysis of the actions and interaction of the agents over time. Agents are autonomous, intelligent constructs that have their own behavior, defined by a set of rules. Agents may also interact with other agents in the environment. AB simulation allows for representing agent heterogeneity through individual modeling of each agent. By individually modeling agents, the diversity that exists among agents can be observed.
The first step in the development of the AB model is to define the basic behavior of the agent. This is usually done by defining a simple set of rules that the agent must follow. These internal rules may be very general and represent the goals that the agent must accomplish.

Applications of AB simulation cover a wide range of domains. Applications may include modeling agent's behavior related to supply chains, understanding consumer purchasing behavior, predicting the spread of epidemics, understanding the fall of ancient civilizations, modeling the engagement of forces on the battlefield or at sea, and many others. Regarding task analysis and human performance, AB simulation has been mostly used for modeling the information processing modules. Examples include AB models for simulating emergent behavior in air transportation [93], predicting the impact of revolutionary changes to an air transportation system [94], and simulating multiple operators performing repair actions [95].

**System Dynamics Simulation Models.** System dynamics (SD) approach, developed by Jay Forrester at MIT in the late 1950s, is used to understand how things change through time. SD approach refers to macro-level representations of a system and focuses on the objects moving in a system and the interconnections among them. SD modeling consists of three components: (1) a causal loop diagram, which describes the relationships among system variables; (2) stocks, which represent the items moving into the system; and (3) flows, which are inputs and outputs of stocks and refer to the rate of change of the stocks. SD has been mostly used for modeling and simulation of workload and fatigue [96], stress [97], or human error [98].

**Multi-method Simulation Models.** To the best of our knowledge, there is no simulation tool or study that simulates task analysis models by combining multiple simulation methods, such as AB, DES and SD. Combining more than one method is expected to provide a more
accurate representation of the system and to capture specific characteristics of particular structures.

This research seeks to address these gaps and challenges through the objectives that are presented in the next sections.

**Research Objectives**

The main objective of this research is to implement a tool that automates the modeling process and simulates tasks performed by an individual (or group of individuals) in any context, while accounting for operators' heterogeneity, workload estimation, and error prediction. The resulting simulation application is called Universal Task Analysis Simulation Modeling (UTASiMo) and it is a combination of AB, DES, and SD simulation models. AB captures the operators' behaviors, DES the task order execution and the resources needed to execute the tasks, and SD the error estimation. The UTASiMo-generated models are dynamically created during run-time based on user inputs.

Based on the current issues and challenges, the following research objectives are presented:

1. To develop a universal task analysis simulation application that produces models in order to analyze sequences of tasks and estimate human error probability and human operator workload.

   1.1. To analyze sequences of tasks based on task priority, operator characteristics, and events in order to estimate task completion times.
1.2. To create a human operator model that indicates how the operators with diverse characteristics perform the tasks and to estimate human operator workload

1.3. To estimate human error probabilities influenced by the dynamics of the tasks and the properties of the human operators over time.

2. To develop a novel task analysis simulation application that automates the modeling process and generates task analysis simulation models in any domain, without the need for effort, training, programming skills, or knowledge of modeling and simulation theory.

3. To develop a task analysis methodology that supports the simulation application.

**Research Methodology**

A generic Multi-Method Modeling & Simulation (3M&S) framework [80] has been followed for the implementation of UTASiMo. The framework consists of four main phases:

1. Conceptual Model Development, including information and data collection, as well as assumption making

2. Model Implementation

3. Model Verification and Validation

4. Documentation of results

**Expected Results**

Expected research results include:
1. A universal task analysis simulation application that produces models to analyze sequences of tasks and to estimate human error probability and human operator workload.

2. A novel task analysis application that automates the modeling process and generates task analysis simulation models in any domain, without the need for effort, training, programming skills, or knowledge of modeling and simulation theory.

3. The development of a methodology that supports the task analysis simulation application.

**Overview of this research**

The key contributions of this dissertation include: (1) a simulation-based application for automating the model building process, which drastically reduces the time and effort of the analysts; (2) a methodology that supports the simulation application; (3) UTASiMo-generated models which combine three simulation methods: agent-based, discrete event, and system dynamics; and (4) UTASiMo-generated task analysis models that estimate workload and human error probability in any domain.

This research includes the following chapters. Chapter One contains the introduction of this research. Chapter Two provides a literature review on task analysis and task analysis methods and tools and proposes a simulation-based task analysis methodology and tool. Chapter Three contains the modeling details and development-related decisions, while Chapter Four provides an overview of the simulation application. Chapter Five presents two case studies for
the purpose of validating and evaluating the results produced by the simulation application.

Finally, Chapter Six concludes with a discussion and future work.
CHAPTER 2:
REVIEW OF TASK ANALYSIS LITERATURE

The main objective of this chapter is to provide a review on the most commonly used task analysis methods and task analysis-related research. More specifically, the chapter consists of an overview on HTA, CTA, SOSD, and TA. Moreover, the chapter describes the gaps and limitations of the current task analysis methods and tools, and proposes a new methodology and tool to fill these gaps.

Brief History of Task Analysis

Task analysis is a methodology that studies the activities that an operator (or a group of operators) needs to perform to accomplish a goal. The primary purpose of task analysis is to identify the task demands on the operator, compare them with the operator's capabilities and, if required, modify them to reduce error and achieve successful performance. Task analysis is supported by a set of specific techniques that help the analysts collect and analyze information in order to make design decisions [21]. Task analysis is widely used not only in systems design [7], [2], but also in a wide range of industries [22]. Applications of task analysis include human error prediction, aid in designing a new system, investigation of accidents during task performance, identification of the cause of an accident, interface design, and training needs analysis and design, among others.

Task analysis has its roots in the work of Taylor [23] and Gilbreth [24] that introduced techniques for identifying, evaluating, and organizing task components of manual work. The
tendency of complex systems to fail due to human errors led to the development of concepts of limits of human performance and methods of analysis [25-27]. As tasks became more demanding, attention was focused on the components of human performance in relation to training requirements during the 1950s and 1960s [28], [29]. During this period, Annett et al. [30] introduced the principle of hierarchical decomposition (HTA). Attention was shifted to the identification of the skills and abilities needed to successfully perform a task. A focus of interest in decisions and knowledge regarding human performance started to form, leading to the introduction of Cognitive Task Analysis (CTA). The aforementioned methods were adopted in the field of ergonomics to investigate technical challenges, as well as system and human performance issues [31-34].

A large amount of research has been conducted on task analysis and various task analysis techniques/methods have been developed. Kirwan and Ainsworth [2] defined task analysis as the study of what actions and/or cognitive processes an operator is required to perform to achieve a goal. Sheridan [35] defined task analysis as the decomposition of a task into elements and the relations among those elements in space and time.

Task analysis is also a fundamental key in the field of ergonomics linking theories of human performance and methods for analyzing human activity. Annett and Stanton [20] state that task analysis is "the collective noun used in the field of ergonomics, which includes HCI, for all the methods of collecting, classifying and interpreting data on human performance". From a systems perspective, task analysis is defined as [36]: 
"...the collective noun used in the field of ergonomics, which includes HCI, for all the methods of collecting, classifying, and interpreting data on the performance of systems that include at least one person as a system component."
Some of the benefits of task analysis include: ensuring compatibility between system goals and human capabilities in order to achieve a goal; optimizing human elements; minimizing potential errors; and leading to more efficient and effective integration of the human element into system design.

A wide variety of different task analysis methods and definitions exist in the literature. Each method is selected based on the purpose and scope of the analysis, the task, cost and time factors, expertise, and the underlying design model (e.g., [2], [3]). The next sections provide a review of the most commonly used task analysis methods and their characteristics. These methods are Hierarchical Task Analysis (HTA), Cognitive Task Analysis (CTA), Spatial Operational Sequence Diagrams (SOSD) and Timeline Analysis (TA).

Hierarchical Task Analysis. Hierarchical Task Analysis is the most frequently used method primary because it is generic in nature and can be applied in any domain. HTA is a top-down approach developed by Annett and Duncan in the late 1960s [37]. HTA is a hierarchical decomposition technique [30] that has two key features: (1) the functional feature, which includes top-level goals of the task rather than actions; and (2) the hierarchical feature, which involves the decomposition of goals to a hierarchy of sub-goals in order to analyze complex tasks, such as planning and decision making [20]. The benefit of analyzing a task in this macro level is that the analysis is independent of the technology used to achieve the goals and, therefore, can be reused across contexts.

Although HTA is effective for decomposing complex tasks, it is usually used in combination with other task analysis methods to increase its effectiveness due to the narrow view of the task [13]. HTA is not a predictive tool and focuses on existing tasks. Moreover, HTA is
mainly a descriptive method, rather than analytical, and does not include the analysis of cognitive processes [14]. Finally, HTA considers the operator’s cognitive processes as a black box and does not support the components that are necessary for the analysis of the cognitive human activity.

**Cognitive Task Analysis.** Cognitive task analysis may be defined as a set of methods for determining the mental demands and cognitive skills required to perform a task proficiently [38]. CTA is more appropriate for cognitively complex tasks that require extensive knowledge and judgment and take place in a ‘complex, dynamic, uncertain, real-time environment’ [40]. CTA is described as the decomposition of cognitive tasks into their elements and the relations among these elements related to the human cognitive ability requirements [21]. The term cognitive task refers to the group of related mental activities required to achieve a goal. Klein and Militello [39] define CTA as the set of the cognitive skills or mental demands needed to perform a task.

The most commonly used CTA techniques include cognitive interviewing, protocol analysis, neural network modeling, and computer simulations [41]. A wide range of CTA methods have been developed over the last years [42], [43], including a cognitive simulation to investigate cognitive activities involved in fault management with nuclear power plant operators [44] and CTA of nuclear power plant control room operations [45].

However, CTA is a time-consuming process and the results of a CTA can be sometimes misleading, since it considers analysis of the cognitive processes, while assuming expert performers [14]. Finally, CTA models do not always capture physical capabilities and variability in operator's skills and experience.
**Spatial Operational Sequence Diagrams.** The Spatial Operational Sequence Diagrams [2], [12] is a technique for representing the spatial links among operations. More specifically, a form of map diagram is produced by plotting the sequence of the task steps of the human operator's interaction with equipment.

The advantages of the SOSD technique include the clear representation of task steps and the data visualization that make it more easily understood to users compared to other techniques' abstract representations. This process highlights the parts or sequences of tasks that require excessive movement between locations and assists in taking appropriate actions to optimize the task. However, there are disadvantages that cannot be ignored. The pictorial representation of the task sequences can become complex, as it depends on task size, task dependencies, and type of layout. It is also difficult to show the available time of associated tasks and actions, as well as the sequences of tasks, in the same diagram. Thus, SOSD need to be used in conjunction with other task analysis techniques, such as TA or decomposition methods, before making any inferences about changes that a task may demand [2].

**Timeline Analysis.** Timeline Analysis is an analytical technique that determines the functional and temporal requirements for any given task combination [2]. Functional requirements refer to the steps and actions that the human operator needs to perform to complete a task, while temporal requirements refer to the time required to complete the task. TA is a simple, pen and paper technique that maps tasks along time, taking into consideration frequency and duration of tasks, as well as interactions with other tasks and human operators. TA includes clear allocation of all operations in the task performed in a fixed order, measurable amount of required time, and representation of information in graphical form.
Since TA is a representational technique, it relies heavily upon other methods for gathering basic information. When combined with workload analysis, it can be used for estimating staffing requirements and for providing information about task steps that need to be redesigned due to high workload. It can also be used in human reliability analysis to determine whether the operator is likely to complete a task within a particular amount of time. In addition, it would be more useful if HTA is conducted prior to TA [2].

However, the system under analysis needs to be relative predictable and have repeatable performance. The technique presents difficulties when it needs to represent different operators with different roles, while considering unpredictable factors. Moreover, although TA can be used to predict completion times for tasks and allocation of tasks between different operators, it may ignore important influences on task allocation. This may occur because timeline analysis looks only at temporal relationships and does not take into account operator's skills and spatial characteristics, i.e. locations [2].

**Review of Existing Task Analysis Software Tools**

The next sections present and compare the various task analysis software tools that have been developed for both commercial and non-commercial purposes.

**TaskArchitect.** TaskArchitect [7] is a commercially available tool to perform HTA, and the most comprehensive tool among the various commercially available task analysis tools in the market, as of today. It is most commonly used by for the system design phase, but it can occasionally serve trade studies that require detailed analysis of human tasks. It can support
design of complex interactive systems and facilitate discussion among the design stakeholders, thus enabling evaluation and documentation of the design product. TaskArchitect allows for information to be recorded and illustrated in a link diagram. Such information includes locations of tasks carried out in the workplace and movements of human operators between tasks. The spatial information helps the analyst understand the tasks and the environment, along with the HTA data. Finally, TaskArchitect allows the analysts to export data to modeling and simulation tools.

**MicroSaint.** MicroSaint is a DES task analysis tool that allows for specifying the descriptions of the task, designing the task flow, and analyzing the task. MicroSaint supports HTA and procedural task analysis. HTA is developed from general to specific, starting from the most complex task and breaking down the task to less complex tasks. A procedural task analysis is a sequential, step-by-step analysis.

**Work Models that Compute.** Work Models that Compute (WMC) [17] is a work modeling and simulation framework that describes complex behaviors emerging from the interaction of multiple agents.

**Enhanced Operator Function Model.** The Enhanced Operator Function Model (EOFM) [13], which is an extension of the Operator Function Model, represents human behavior as an input-output system using an XML notation. An instantiated EOFM describes how a human operator acts as part of tasks, which are hierarchical representations of goal-driven activities.

**MIDAS.** The Man Machine Integrated Design and Analysis System (MIDAS) [19] has been designed for evaluating proposed human-machine system designs and testing a variety of
behavioral models. MIDAS consists of a user interface, an anthropometric model of the human operator, symbolic operator models, and an environmental model.

Table 1 summarizes the comparison of the tool with respect to (1) programming skills, training, and knowledge of simulation theory; (2) simulation capabilities; and (3) M&S methods. The comparison reveals that more adequate and widely accessible simulation tools need to be developed that will be able to perform a more comprehensive task analysis [20].
Table 1. Comparison among Task Analysis Tools

<table>
<thead>
<tr>
<th>Simulation Tool / Application Name</th>
<th>No need for programming skills, training or knowledge of simulation theory</th>
<th>Simulation Capabilities</th>
<th>M&amp;S Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Supports simulation</td>
<td>Automated model construction</td>
<td>Dynamic aspects</td>
</tr>
<tr>
<td>WMC</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MicroSaint</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>TaskArchitect</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EOFM</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>MIDAS</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>UTASiMo</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Universal Task Analysis Methodology

Current task analysis methods and tools usually omit human operators or assume perfect human behavior. Representation of human activity that includes simultaneous representation of multiple users with variability in the level of skills and abilities (i.e. from novice to expert human operators) considering both cognitive and physical demands is also missing. Therefore, a task analysis methodology, named Universal Task Analysis (UTA), is proposed for filling these gaps and applying simulation to perform task analysis.

The UTA methodology considers basic concepts and elements of multiple task analysis methods, including HTA, TA, and SOSD. More specifically, UTA adopts the hierarchical decomposition approach of a task into subtasks while incorporating cognitive processes, i.e. monitoring a panel or reacting to an event. UTA also includes the estimation of the execution time for each task step and provides a graphical representation of a flow diagram linking the task steps in the order they are performed. In addition, UTA allows for defining human operator profiles, as well as events that may happen during the task execution. Table 2 illustrates a list of the basic UTA concepts and elements adapted from current task analysis techniques.
<table>
<thead>
<tr>
<th>No.</th>
<th>Concept</th>
<th>HTA</th>
<th>SOSD</th>
<th>task analysis</th>
<th>CTA</th>
<th>UTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Identify task steps</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>Events and timings</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td>Task times assessment</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>4</td>
<td>Graphical flow diagram linking task steps</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td>Workload estimation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>6</td>
<td>Error prediction</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

*HTA = Hierarchical Task Analysis; SOSD = Spatial Operational Sequence Diagram; TA = Timeline Analysis; CTA = Cognitive Task Analysis; UTA = Universal Task Analysis*
UTA also supports the simulation application and attempts to fill the gaps in the current task analysis techniques. UTA is divided into 4 main steps:

_Step 1:_ Identification of the overall task under analysis.

_Step 2:_ Decomposition of overall task into subtasks. The tasks and subtasks should be broken down into further subtasks until no further breakdown is necessary. Each subtask should be performed by a single human operator.

_Step 3:_ Data collection input process, which requires the user to fill in a spreadsheet template with the collected data. Data include execution times for cognitive and physical tasks, task locations, human operator roles, and other environmental and human-related factors.

_Step 4:_ Simulation analysis. Once all the sub-tasks have been fully described and the data have been collected, the simulation can start. The user uploads the spreadsheet on UTASiMo and runs the simulation. The models are automatically created based on the spreadsheet inputs. The UTASiMo-generated models simulate and animate the system and the operator behavior and produce the results.

Finally, UTA considers the roles of agents, the types of tasks, and the work structure, among others. The next section describes the basic elements that are included in the UTASiMo simulation model: (1) task characteristics; (2) human operator (agent) characteristics; and (3) environmental conditions and events. These elements are used for determining the completion time of the task, the operator workload and the human error probability.
Components of task analysis considered in UTA and UTASiMo design

Over the last decades, there is a growing interest in modeling and simulation of human performance in various fields, such as training and risk assessment of complex systems. A prerequisite to the measurement of human performance is task analysis. Modeling and simulation of task analysis should consider individual characteristics and technology related to tasks. Kirwan and Ainsworth [2] provide a taxonomy of descriptive task decomposition categories that have been used in various studies. A subset of these categories' elements has been chosen to be included into the development of the UTASiMo. These elements are common to most task analysis studies and methods and are used as input to most of the existing task analysis simulation tools. The number of elements was also chosen to be as low as possible in order for UTASiMo to be usable, widely applicable, and capable of fully automating the process so that anyone without specialized knowledge to be capable of using it. The elements are summarized in Table 3.
Table 3. Summary of elements

<table>
<thead>
<tr>
<th>Category</th>
<th>Example of Factors</th>
<th>Literature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>Task/action, Subtasks, Complexity, Skills, Time Taken, Errors</td>
<td>Kirwan and Ainsworth [2], Clark et al. [100],</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Laughery[15], Van der Veer et al. [101]</td>
</tr>
<tr>
<td>Human operator</td>
<td>Skills, Required Speed, Workload, Errors</td>
<td>Kirwan and Ainsworth [2], Laughery[15], Van der Veer et al. [101]</td>
</tr>
<tr>
<td>Work environment and other requirements for</td>
<td>Hardware features, Initiating events, Personnel requirements, Events or criteria</td>
<td>Kirwan and Ainsworth [2], Clark et al. [100],</td>
</tr>
<tr>
<td>undertaking the tasks</td>
<td>used to decide among alternative tasks</td>
<td>Laughery[15], Van der Veer et al. [101]</td>
</tr>
</tbody>
</table>

Tasks. Tasks are key components in task analysis. Although task-related research has broadly emerged in the social and behavioral science, there is limited agreement on the understanding of a task and its characteristics in the literature [47], [48]. In this section, different conceptual bases for defining the tasks are presented.

Miller [49] defined a task as "any set of activities, occurring at the same time, sharing some common purpose that is recognized by the task performer". Wheaton [50] described a task as a set of responses to a specific stimulus intended to achieve a goal. Similar definitions were proposed by Hackman [51] and McCormick [52]. Fikes [53] stated that tasks are the combination
of a set of goals to be achieved along with a set of requirements and constraints to devise plans. Carroll [54] defined a task as an activity performed to achieve a specified objective. Preece et al. [55] described tasks as consisting of a set of activities "required, used, or believed to be necessary to achieve a goal using a particular device". Fleishman et al. [56] described the tasks in terms of the abilities required to perform them.

In this research, a task is defined as a set of discrete activities performed in a sequence to achieve a desired goal, which has a beginning and end. The activities may be a combination of decisions, perceptions, and/or physical activities. Tasks performed by one agent are called primitive tasks or activities. By performing a sequence of activities, a complex task can be performed. A detailed representation of the task structure is provided in Chapter 3.

**Work Environment.** Except from the tasks that take place in a work environment, the work environment itself is also important. Most of the task analysis methods are focused on modeling one user and that user's static tasks, neglecting parts of the environment, i.e. locations and physical layout, and dynamic aspects of the task.

Moreover, human operators have different skills and expertise levels, and are better placed in certain jobs rather than others. Therefore, organizations need to divide up the work and allocate it in order to accomplish complex tasks. The division of work includes the breakup of the task into subtasks that can be performed by a single person and the allocation of individuals to tasks.

**Human Operator.** Task analysis effectively integrates the human element into the system design. One of the highly significant areas of interest in task analysis is the simultaneous representation of human operators with varying characteristics. In this research, a human
operator is defined as an agent in order to observe the diversity that exists among the different agents in their attributes and behaviors.

An agent also has an internal state consisting of three components (Figure 1): ability to "Perceive" the current state of environment, ability to "Adapt" and renew its own representation of the environment, and ability to "Act" based on its new internal state and its current perception. A detailed implementation of the agent and its internal state is provided in Chapter 3.

Cognitive and physical demands may be placed on the human operator (workload) and errors may occur while the operator is performing the task. Each operator’s workload is also one of the factors that affect human error.

**Workload Estimation.** There is no universally acceptable definition of workload in the literature [60]. Stein et al. [61] define workload as the amount of physical and mental effort in
accordance with the task demands and operator’s internal performance. Workload may result to problems when it is too high (overload) or too low (underload). High or extreme workload may result in poor performance and stress, while too low workload may cause boredom. Both may lead to human error and failure to perform a task correctly. Factors that affect workload may include time spent on tasks, number of tasks, operator experience and state, lack of skill, environmental factors, and task complexity, among others [60]. According to MIL-HDBK-46855A [62], the following rates for workload estimations are provided: Low (< 60%); Medium (60-75%); High (75-90%); and Extreme (> 90%).

A variety of subjective and analytic workload techniques exist, including uni- and multidimensional rating scales, NASA-T LX, questionnaires and interviews, and timeline analysis. The concept of utilization as a physical and mental workload measure has been used in numerous studies [63-65]. However, the operator utilization, as a workload measure, doesn't take into account agent's individual characteristics, such as skills that the agent possesses to execute the task, task frequency, or other factors. For example, task duration can be modified by the agent's skills influence on task performance. Therefore, an adjusted operator utilization is used here as a measure of each human operator's workload, which is described in more detail in Chapter 3. The workload analysis is followed by a dynamic analysis, which estimates potential human unreliability during the simulated scenarios.

**Human Error.** Human error is a key factor associated with accidents that may have consequences to people, systems, and the environment. Human Reliability Analysis (HRA) is a common methodology used to analyze human error. HRA can be defined as a set of methods that assess the impact of human errors on system performance. HRA involves the use of qualitative
and quantitative methods to assess the human contribution to risk [68] and to provide a probabilistic quantification of undesired system consequences.

A variety of HRA approaches have been proposed in the literature, including HEART (Human Error Assessment and Reduction Technique), THERP (Technique for Human Error Rate Prediction), and APJ (Absolute Probability Judgment), among others. A typical feature of all approaches is a set of factors “which influence the likelihood of an error occurring” [66]. These factors include influences related to the individual, system, task, or environment. Examples of factors include experience, task complexity, workload, working conditions, and system quality, among others.

In this work, we utilize the SPAR-H HRA method to build a SD model for estimating human error probabilities of the simulated system. The SPAR-H method [67], [69] has been developed for assessing human error probabilities in the nuclear industry. However, the method shows promise for wider application in other domains as the human error probability (HEP) data can be applicable to other domains. Moreover, the SPAR-H method has its base in task analysis and it can be easily adapted to incorporate factors that affect error in multidisciplinary domains [68], [70]. A more detailed description of the incorporation of the SPAR-H HRA method into the development of the SD model is provided in Chapter 3.

The next chapter describes the conceptual design of the simulation application.
CHAPTER 3:

METHODOLOGY AND CONCEPTUAL MODEL DESIGN

"Parts of this chapter have been submitted for review or have been published in [85-87]"

This chapter describes the methodology followed for the conceptual model design. A generic multi-method modeling and simulation (3M&S) framework [80] has been followed for the identification of relationships between the basic components of task analysis and the conceptual model creation of the UTASiMo simulation application. UTASiMo [85-87] is a simulation application implemented in Anylogic™ as a multi-method simulation application which has an integrated deployment of AB, DES and SD simulation models. The UTASiMo-generated models are multi-method stochastic models capable of simulating tasks performed by an individual (or group of individuals) in any context and of estimating human error probabilities, operator workload and task completion times based on agent and task structures. The UTASiMo models do not pre-exist, but they are dynamically created during run-time based on user inputs. Both the tool and the generated models have the same architecture. Therefore, the conceptual model described in the next sections represents the structure of both UTASiMo and UTASiMo-generated models.

Conceptual Model

The first steps of the conceptual modeling phase of the 3M&S framework [80] include the problem formulation and decomposition of objectives into smaller objectives, which are all
described and defined in Chapter 1 of this dissertation. The next steps include the identification of constraints and assumptions, the suggestion of M&S methods to model each objective which leads to the formulation of conceptual models for each selected method, and the definition of inputs and outputs of the resulting conceptual models. After selecting from a list of generic criteria and assigning weights to each criterion, the framework suggested to implement the task sequences using DES, the human operator model using AB, and the human error estimation using SD. The conceptual models for each selected methods, the assumptions, inputs, and outputs of each model are described in the next sections.

**Unified Modeling Language (UML).** The unified modeling language (UML) [76], [77] was used to identify the general concepts and constructs and to investigate the relationships that are essential for defining these constructs. UML provides a set of modeling concepts and constructs. UML is a modeling language mainly used to describe and represent the steps and actions that a user performs to achieve a goal while interacting with the system [78]. UML diagrams were used to represent the concepts needed in the development of the UTASiMo as they help describe these relationships and capture both static and dynamic aspects of a system.

**Universal Task Analysis (UTA).** The overall UTA process is outlined in Figure 3 using a UML diagram, which describes the user's interaction with UTASiMo when performing simulation-based task analysis. The user collects and organizes the task-, operator-, and environment-related data required for the analysis of the overall task and fills in an Excel template (database) with the data. The user then selects to simulate the scenario. UTASiMo accesses the database to retrieve the information required for the automated generation and initialization of the model. When the simulation of the model is over, the results are presented to
the user. The outcomes include task completion times, workload estimation and human error probability estimation.

Figure 2. Use Case Diagram illustrating the overall UTA process

**Discrete Event Module: Task Structure and Environment.** As already mentioned, a task is defined as a sequence of discrete activities performed in a sequence to achieve an overall goal, which has a beginning and end. Activities, also called primary tasks, are tasks that do not require further decomposition.
DES is used for modeling a task as a series of discrete activities, such as decisions, perceptions, and/or physical activities. The activities are represented as nodes in a network and are connected with dependencies between them, while the task is a network of nodes. The task structure properties represented in the model include the number of subtasks, task complexity, required skills, priority, duration, and critical time. Table 3 shows the task structure attributes which are used as inputs in the DES model or as points of interaction with the other models.

Task complexity is defined in terms of single demands (mental or temporal) that the task places on the operator performing it. Task complexity can be measured subjectively [81], thus a scale from 1 (simplest task) to 5 (most complex task) is used to define each task's complexity [88]. Task complexity is also one of the factors that affect human errors while operators perform their tasks.

Task duration refers to the estimated execution time for an average operator to perform a task. Task duration is assumed stochastic following a triangular probability distribution. The probability distribution accounts for the variability in task execution times, since it is unlikely for two different operators to perform the same task in exactly the same time on two different occasions. The triangular distribution is chosen as (1) it is the most commonly used distribution for modeling expert opinion [82] and (2) it can also be used to model the time required to perform a task in the absence of real-world data [83].

Each task can also be completed efficiently when the human operator possesses a certain set of skills. Therefore, we assume that the mismatch between the operator's skills and task-related skills impacts the duration of the task and the probability of human error during task execution. Table 4 shows the task structure in more detail.
Table 4. Task Structure

<table>
<thead>
<tr>
<th>Properties</th>
<th>Definition</th>
<th>Model Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task ID</td>
<td>Unique task identifier</td>
<td><em>tid</em> (integer)</td>
</tr>
<tr>
<td>Task Name</td>
<td>Name of the task</td>
<td><em>tname</em> (string)</td>
</tr>
<tr>
<td>Task location</td>
<td>Node location in two dimensional continuous space</td>
<td><em>tlocation {x, y}</em></td>
</tr>
<tr>
<td>Number of Subtasks</td>
<td>Sequence of connected activities</td>
<td><em>tnumber</em> (int)</td>
</tr>
<tr>
<td>Task Complexity</td>
<td>Cognitive demand the task places on the operator performing it</td>
<td><em>complexity</em> (int)</td>
</tr>
<tr>
<td>Skills</td>
<td>Skills and knowledge required by each agent to perform the task.</td>
<td><em>tskills</em> (double)</td>
</tr>
<tr>
<td>Priority</td>
<td>Recommended order to perform the tasks</td>
<td><em>tpriority</em> (int)</td>
</tr>
<tr>
<td>Critical Time</td>
<td>The time in which the task needs to be performed</td>
<td><em>tcrit</em> (time)</td>
</tr>
<tr>
<td>Duration</td>
<td>The actual time it takes for the task to be performed</td>
<td><em>duration</em> (time)</td>
</tr>
<tr>
<td>Task Frequency</td>
<td>Number of times the task is performed</td>
<td><em>tfrequency</em> (int)</td>
</tr>
</tbody>
</table>

**Agent-Based Module: Agent Structure and Workload Estimation.** Task analysis effectively integrates the human element into the system design. One of the highly significant areas of interest in task analysis is the simultaneous representation of human operators with varying characteristics. By modeling human operators individually using AB simulation, the
diversity that exists among them in their attributes and behaviors can be observed. A typical structure of an AB model consists of [31]: (1) a set of agents which have attributes and behaviors; (2) a set of agent rules defining agent behavior and/or interaction among agents; and (3) the agents’ environment. Agents may interact with their environment in addition to other agents.

**Agent Structure.** The human operator is represented as an agent having one or more of the following characteristics [57]: (1) identifiable and discrete individual with a set of rules directing its behaviors; (2) autonomous agent that can act independently in an environment and have control over its actions and its internal state; (3) situated agent that works in an environment and interacts with it; and (4) flexible agent that adapts its behaviors to be better suited to its environment.

Regarding the first characteristic, a unique identifier is assigned to each simulated agent upon creation. Since task execution depends on the operator's capabilities, each agent is assigned with a skill factor. The skill factor accounts for operator variability and is used to calculate the completion time of the task when multiple operators perform the same task under similar conditions. For example, an average operator with skill factor 1.0 would require time T to complete a task, while an expert operator with skill factor 0.9 would require time 0.9T, which is equal to 90% of the time taken by an average operator to complete the same task. Each operator can support one skill factor per task. The skill factor is used to denote the level of performance attained by an operator working under customary conditions.

Regarding the other three characteristic, an agent has a spatial property and a behavior modeled by a statechart. The operators change states based on their task execution strategy: how
to choose the next task position and when to stop. Thus, an agent has an internal state consisting of three components to control its actions: ability to "Perceive" the current state of environment, ability to "Adapt" and renew its own representation of the environment, and ability to "Act" based on its new internal state and its current perception. The agent's perception is related to information that the agent receives. An agent may receive information regarding: (1) whether or not it is in physical condition to carry out the task (2) its physical location, (3) if there are still tasks to execute, and (4) locations of the tasks.

The internal state is modeled as a statechart with three states: Perceive, Adapt, and Act. The Perceive state is an event handler triggered every millisecond to check the environment for any event. The event is either user-defined (i.e. alarm, stairs) or can be retrieved from the model (i.e. number of tasks to be executed, next task location, etc.). The Adapt state includes a list of actions linked to each event. Currently, each event may be linked to one or more actions. The agent selects the appropriate action based on triggering events and information gathered from the environment. In case of user-defined events, the action is selected based on equal probabilities. The Act state is the execution of a task or an event-triggered action.

For example, the agent checks the environment and receives information that there are still tasks to be executed. The agent then checks the list with the assigned tasks and moves to either the next in the list if there is a task priority/prerequisite or to the closest one. The model also takes into account travel times from one task location to another. The travel time is calculated based on equation 3.1. The distance is calculated based on the location coordinates between two tasks.

\[
\text{Travel Time} = \frac{\text{Distance}}{\text{Agent's Speed}} \quad (3.1)
\]

36
A detailed list of the agent structure is presented in Table 5. The first part of the Model Variable indicates the name of the variable in the model and the second part in the parenthesis declares the type of the variable.

Table 5. Agent Structure

<table>
<thead>
<tr>
<th>Properties</th>
<th>Definition</th>
<th>Model Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent ID</td>
<td>Unique agent identifier</td>
<td>aid (integer)</td>
</tr>
<tr>
<td>Agent Role</td>
<td>Name or role of the operator</td>
<td>name(string)</td>
</tr>
<tr>
<td>Agent location</td>
<td>Agent location in two dimensional continuous space.</td>
<td>location {x, y}</td>
</tr>
<tr>
<td>Speed</td>
<td>Agent walking speed. The default walking speed 1.5 m/sec [89-90].</td>
<td>speed (double)</td>
</tr>
<tr>
<td>Skills</td>
<td>Skills and knowledge possessed by each team member.</td>
<td>skills (double)</td>
</tr>
<tr>
<td>Task execution time</td>
<td>The actual time taken to execute the task</td>
<td>execution_time (double)</td>
</tr>
<tr>
<td>State</td>
<td>Each agent has a state that declares when he is available to be assigned a task.</td>
<td>state (Perceive, Adapt, Act)</td>
</tr>
<tr>
<td>Assigned tasks list</td>
<td>A list with all the tasks assigned to the operator</td>
<td>assigned_tasks ( ArrayList&lt;Task&gt; )</td>
</tr>
<tr>
<td>Workload</td>
<td>The mental and physical demands on the human operator</td>
<td>workload (double)</td>
</tr>
<tr>
<td>Error</td>
<td>Estimation of probability of error while performing tasks</td>
<td>error(double)</td>
</tr>
</tbody>
</table>
Workload Estimation. The concept of utilization as a physical and mental workload measure has been used in numerous studies [63-65]. The operator utilization as a workload measure does not take into account the agent's individual characteristics, such as skills that the agent possesses to execute the task, task frequency, or other factors. For example, task duration can be modified by the agent's skills influence on task performance. Therefore, an adjusted operator utilization is used here as a measure of each human operator's workload, given by equation 3.2.

\[
Workload = \frac{\text{Time Busy}}{\text{Total Time}} \times 100 
\]  

(3.2)

where

\[
\text{Time Busy} = \text{OperatorSkillFactor} \times \text{TaskDuration} \times \text{TaskFrequency}
\]

\[
\text{TotalTime} = \text{maximum Task Time} \quad \text{(Available task time)}
\]

The adjusted workload assessment multiplies estimated task duration by the simulated agent's skills and the task frequency, sums the results of all tasks, as in equation 3.3, and compares them with the agent’s overall available time. This enables a workload analysis which checks whether human agents are capable of performing the assigned tasks within the available time.

\[
\text{Total Workload} = \sum_{i=1}^{n} Workload_i
\]

(3.3)

System Dynamics Module: Human Error Probability Estimation. System dynamics is a modeling and simulation method that enables the investigation of broader system behaviour. Systems modeled with SD contain elements that dynamically change based on various
influences. For example, human error, which is an attribute of human agents in such systems, is influenced by the changing system's environment. Therefore, the most appropriate method to model the dynamic nature of such systems is SD. Moreover, SD is more appropriate for modeling the effect of human error in the reliability of the system instead of focusing on the human error probability [98].

The SD model is incorporated into humans, which are implemented as agents using a statechart. The statechart is responsible for the higher level controller of the human's behavior and actions during the task execution process. The SD model (Figure 3) contains causal loops that show interrelations among system parameters and expose feedback loops within the system. The main feedback loop incorporates workload and quality of service. The loop is initiated by changes in agent's available time to perform a task, which, subsequently, influences workload. More specifically, a reduction in available time increases workload. Workload has a negative effect on quality of service which increases human error occurrence and constitutes a reinforcing loop. Quality of service (i.e. medical care) is also affected by working conditions and system/interface design quality [99].

These influences in combination with properties of the task, the environment, and the agent give rise to human error estimates for each task-step. The model uses these estimates to classify the task as of high, medium, or low criticality.
**Human Error Estimation.** In this research, the SPAR-H method is used for estimating the simulated system's human error probability (SimHEP). The following six factors affecting error (FAE) have been identified: available time to complete task, workload, skills and experience, task complexity, ergonomics, the quality of any procedures in use, and working conditions. Probabilistic modifiers can be assigned to each FAE to estimate SimHEP.

The SimHEP is calculated based on the decomposition of tasks into subtasks. A HEP for each subtask is calculated in terms of FAE multipliers and base error rate, as in equation 3.4. FAE multipliers are obtained from the system dynamics model. The base or default error rate is called nominal human error probability (NHEP). The SPAR-H method defines NHEP to be equal to 0.001 for action tasks and 0.01 for diagnosis tasks.

\[
HEP_j = (\prod_{i=1}^{6} FAE_i) \times NHEP
\]  

(3.4)
FAEs may increase, decrease, or have no effect on HEP. If the effect of multiple FAEs increases HEP to a value greater than 1, a correction process of SPAR-H is applied, as in equation 3.5.

\[
FAE_{\text{corrected}} = \begin{cases} 
(\prod_{i=1}^{6} FAE_i - 1) \times NHEP + 1, & \text{if } \prod_{i=1}^{6} FAE_i > 50 \\
(\prod_{i=1}^{6} FAE_i), & \text{otherwise}
\end{cases}
\] (3.5)

The overall SimHEP is calculated as the mean error probability of all subtasks based on the number N of tasks that contribute to total error, as in equation 3.6.

\[
\text{SimHEP} = \frac{\sum_{j=0}^{N} HEP_j}{N}
\] (3.6)

The SimHEP provides a quantitative basis to the simulated system's evaluation. Based on the SimHEP, each task is classified as of high/critical (0.01-0.1), medium (0.001-0.01), or low (<0.001) risk to aid analysts in determining which areas of the system may need redesign.

The next Chapter describes the UTASiMo development, the user interface, as well as the capabilities of the UTASiMo-generated simulation models.
CHAPTER 4:  
UTASiMo DEVELOPMENT

This section describes the development and user interface of UTASiMo. Since AnyLogic™ [15] is the only simulation platform that supports discrete event, agent-based, and system dynamics simulation, we decided to use AnyLogic to create a simulation application that could automatically generate multi-method simulation models from a spreadsheet. The next sections describe the basic elements and interface of the simulation application.

UTASiMo Overview

The UTASiMo simulation application provides a facility whereby a simulation model for a particular task can be automatically constructed from a spreadsheet without programming or modeling effort. The ability to automatically generate a simulation model can significantly decrease the cost and time of developing a simulated system.

Figure 4 provides an overview of the main functions of UTASiMo and the flow of the multi-method M&S process. As soon as the user uploads the Excel file on UTASiMo, the simulation application generates a simulation model and initializes each of the agents, tasks, and resources. After the model initialization, the tool associates each task with an agent and resource as defined by the user in the Excel file. For each agent, task, and resource (ATR) sequence, the agents move to the assigned tasks and utilize the associated resources to execute the tasks. After all ATR sequences have been executed, the model produces results for task completion times, workload estimation, and human error probabilities.
Figure 4. UML Activity Diagram describing the main functions of UTASiMo and the flow of the multi-method simulation process.
Simulation Application Interface. UTASiMo is comprised of two basic elements: the spreadsheet template (Excel file) and the interface of the simulation application (Anylogic-dependent). The spreadsheet contains the following components needed for the model construction: task ID, task name, expected task durations, minimum and maximum task duration, parent tasks, task complexity, task frequency, task locations in the form of (x, y) coordinates, equipment, human operators, and errors associated with each task. Each component represents a parameter or variable in the simulation model and specifies how the system behaves during analysis. Each component comes with a default value in the spreadsheet. To change this, the user just types the new value in the corresponding cell. To ensure the values will be in a format accepted by the simulation model, a dialog box is displayed in case of a wrong entry to specify the correct format to the user.

The second component is the AnyLogic-dependent interface. Here, the user is able to run the generated model and obtain the simulation results. The interface contains three pages. The first page of the interface, which is depicted in Figure 5, prompts the user to upload or specify the following information: the spreadsheet, the system's layout (optional), and the overall critical task time.
The user is then transferred to the second page, which simulates and animates the generated model. A generated model of a movie theater is illustrated in Figure 6. The purpose of the system's layout is to illustrate where tasks, agents, and objects are located in relation to each other. Tasks are represented as nodes and the lines connecting the nodes represent the tasks that are connected and the sequence in which the tasks are performed. Human operators are represented by an "agent" icon, which changes color based on the agent's state, i.e. black for walking and blue for being busy.
Finally, Figure 7 shows the third page, which contains the simulation results. The simulation results include estimation of execution times for human tasks, human operator workload estimation, and prediction of human error probability. The execution time for each task is the time spent on a task and is affected by the operator's skills. The workload for each simulated operator is calculated in terms of time spent on a task, available task time, and operator's skills.
The Automatically Generated UTASiMo Models. Once the spreadsheet template has been completed and uploaded onto the tool, a model can be generated and the simulation analysis can be initiated by pressing the "Run" button in the initial page (Figure 4). This will start the model building process. The spreadsheet information is used to initialize the parameters of the basic modeling constructs of the UTASiMo-generated models, which include the human agents, the resource agents (equipment, servers, technology), and the tasks. More specifically, the information is used to associate agents and resources with each task. This task, agent, and resource (ATR) sequence corresponds to pathways through the system model. Then, UTASiMo automatically generates a model that presents and simulates a sequence of tasks performed by agents using a range of resources.
As mentioned earlier, the generated models combine three simulation methods: AB, DES, and SD. The overall task in the system is modeled as a sequence of discrete activities, whereas each human operator is modeled as a combination of AB and SD models. The UTASiMo-generated models have an animation capability and allow for a diagram of the work environment layout to be included as a background scene. The animation capability of the generated models provides an animated view of the system as the models are executing. Users can observe the agents that flow through the system to perform the tasks or wait in a queue to be served. This can provide useful information as the user can visually identify any bottlenecks in the system.

Another capability that UTASiMo provides is that it allows the user to observe data and charts produced by the UTASiMo-generated models both at runtime and at the end of the simulation. Users can observe the results on screen (Figure 7) and save them in an Excel file at the end of the simulation.

Finally, the user can reconfigure the parameters of the UTASiMo-generated models by changing the spreadsheet's values and experiment with alternative scenarios. UTASiMo allows also for replicated scenario runs. After experimentation and based on the simulation results, the user can find the desired solution for the system and identify the system design with the lowest task execution times, human error probabilities, and workload.

The next chapter presents two case studies used for validating UTASiMo-generated models.
CHAPTER 5:
CASE STUDIES

This chapter presents two case studies to evaluate and validate the results provided by the simulation application. The first case study concerns the simulation of 11 subtasks performed by a human operator that needs to provide back-up emergency cooling in a plant in the event of a switch room fire. The simulation results are analyzed and compared with the results produced by a model created using the MicroSaint software for validation purposes. The second case study includes the simulation of 104 subtasks that take place in a command and control room during a navy task mission. A hypothetical scenario is created and the simulation results are compared with real-world workload and error data to evaluate the predictive ability of the model.

Case Study 1: Plant Simulation

For the purpose of evaluating the UTA framework and UTASiMo, a real world example was adapted and modified [102] to match the spreadsheet template that the user needs to fill in. The example concerns the need to provide back-up emergency cooling in a plant in the event of a switch room fire. Plant operators need to complete the task within two hours of the initial failure of the primary means of cooling provided by the plant. The UTA framework starts with a hierarchical decomposition, similar to the HTA method. The overall task "Provide Emergency Cooling" is divided into 11 subtasks. Figure 8 illustrates the flow of the overall task execution. Then, the user fills in the spreadsheet template, as depicted in Table 6.
Figure 8. Overall Task Flow
Table 6. Partial List of Input to Excel Spreadsheet

<table>
<thead>
<tr>
<th>Task ID</th>
<th>Task Name</th>
<th>Parent Task</th>
<th>Initial Location</th>
<th>Mean Time (minutes)</th>
<th>Operator ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Provide emergency cooling-</td>
<td>0</td>
<td>(120, 320)</td>
<td>120</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Detect Alarm</td>
<td>0</td>
<td>(150, 350)</td>
<td>0.01</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Take Readings</td>
<td>1</td>
<td>(280, 380)</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Collect Hoses</td>
<td>2</td>
<td>(120, 191)</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Carry Hoses</td>
<td>3</td>
<td>(610, 220)</td>
<td>20</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>Collect Pump</td>
<td>2</td>
<td>(340, 151)</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Take Pump</td>
<td>5</td>
<td>(610, 220)</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>Connect Hose to pump</td>
<td>6</td>
<td>(680, 151)</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>Connect to Cooling</td>
<td>7</td>
<td>(610, 220)</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>Start Generator</td>
<td>8</td>
<td>(580, 151)</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>Start Pump</td>
<td>9</td>
<td>(581, 152)</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>Monitor</td>
<td>10</td>
<td>(580, 151)</td>
<td>10</td>
<td>1</td>
</tr>
</tbody>
</table>
The human operator in all scenarios is assumed of average experience and assigned with a skill factor of 1.0 and initial walking speed of 90m/min. The critical time for the overall task "Provide Emergency Cooling" is 120 min. Each subtask has a specified location, a mean execution time and it may be composed of both physical and cognitive components. The cognitive task components are incorporated into the event spreadsheet. Events E1 and E2 may occur during the normal task execution of tasks T1 and T4. The events E1 and E2 trigger the execution of task T2 and change agent's speed to 120m/min, accordingly.

Finally, the user uploads the spreadsheet on UTASiMo and runs the simulation. The model is automatically created in a very short time based on the spreadsheet inputs. The generated model simulates and animates the system and operator behavior and produces the results.

The execution times provided in the adapted example are rounded up to the nearest five minutes to account for possible problems that may occur during the task execution [102]. The layout and coordinates are not to scale. In particular, the scenario has been studied with the physical system and equipment performing optimally without failures or malfunctions during the event of the switch room fire. The scenario describes the expected behavior of the human operator in the plant after the initiating event of the switch room fire. The scenario is simulated with UTASiMo and MicroSaint in order to compare the results. In both cases, the physical system and equipment are assumed to perform optimally without failures or malfunctions.

**Case study 1 simulated with UTASiMo.** In this scenario simulated with UTASiMo, no human error or plant failure occurs following the initiating event of the fire. Table 6 summarizes the parameters assumed as the input data of UTASiMo. The spreadsheet and layout image are
uploaded to UTASiMo and the simulation starts. Figure 9 depicts the generated work environment and the animation view of the UTASiMo-generated model, while Figure 10(a) illustrates the output of the UTASiMo-generated model for one simulation run.

Figure 9. UTASiMo-generated model when one human operator performs the tasks
Figure 10. (a) UTASiMo-generated model's simulation results for a single run with one operator

(b) UTASiMo-generated model's simulation results for 122 replications with one operator
Various techniques have been used for verification and validation of the UTASiMo-generated model's simulation results. First, the model has been successfully tested for one operator in order to verify the total task execution time (Figure 10(a)). The model has been verified and validated by observing the animation of the simulation output. Validation also includes comparison of the simulated system behavior with the behavior of the real-world system [102]. Quantitative measures, such as the total task time, are examined for validity. The number of replications is calculated at a 95% confidence interval, based on equation 5.1.

\[ n = n_0 \frac{h_0^2}{h^2} \]  

where \( h_0 \) is the half-width of the “initial” number of \( n_0 \) replications and \( h \) the desired level of precision. The recommended number of replications is 122. The simulation results after 122 replications are illustrated in Figure 10(b) and Table 7.

The simulation outputs show that the corresponding actions are executed according to the regular procedure. However, the total simulated time is close enough to the critical time (120 min) and it may also exceed the two hour limit (see histogram in Figure 10(b)). Moreover, the workload of the human operator ranges in high levels. Since, the simulated results approach the results of the task analysis conducted on paper, the model is considered valid.
Table 7. Summary of Results for scenario simulated with UTASiMo

<table>
<thead>
<tr>
<th>Measure Name</th>
<th>Observed Value (minutes)</th>
<th>Simulated Average (minutes)</th>
<th>Simulated Standard Deviation</th>
<th>Simulated Confidence Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Time</td>
<td>110</td>
<td>113.912</td>
<td>10.679</td>
<td>[112.017, 115.807]</td>
</tr>
<tr>
<td>WL</td>
<td>&gt;75%</td>
<td>85.301</td>
<td>7.013</td>
<td>[84.056, 86.546]</td>
</tr>
</tbody>
</table>

**Case Study 1 simulated with MicroSaint.** The scenario is also simulated with MicroSaint as an additional technique to compare the simulation results and validate the model. MicroSaint allows user to construct DES models using a series of modeling components and the Microsoft C# programming language.

The task analysis data (Table 6) is entered into MicroSaint as a series of individual tasks, as illustrated in Figure 11, in order to develop the simulation model. The interrelationships among the tasks, i.e. task sequences and triggering events, are also defined in the form of a task network. For each task, the following attributes are defined: task name, operator assigned to perform the task, location of the task, and task duration.
In contrast to UTASiMo, the Microsaint software does not provide a built-in estimator for workload. Thus, we implemented a function "CALCULATE_WORKLOAD" in C#, which is called in each task's beginning effect field and calculates the current workload for the operator performing the task. The function uses the resource utilization and, more specifically, the equations 3.2 and 3.3, as a measure for the mental and physical workload. Finally, we run the simulation model for 122 replications using the same input data and assumption as in the case of the UTASiMo. The results obtained from MicroSaint are then compared with the UTASiMo output.

Two-sample t-tests are used to determine if there is statistical difference between the MicroSaint and the UTASiMo simulation outputs. The level of significance is set to $a = 0.05$. More specifically, we set up the following hypothesis for the mean workload (WL) obtained by the different simulation tools:
H₀: There is no difference between the mean workload obtained by the different simulation tools ($μ_{WL_{UTASiMo}} - μ_{WL_{MicroSaint}} = 0$)

H₁: There is difference between the mean workload obtained by the different simulation tools ($μ_{WL_{UTASiMo}} - μ_{WL_{MicroSaint}} ≠ 0$)

We set up the following hypothesis for the mean total time (TT) obtained by the different simulation tools:

H₀: There is no difference between the mean workload obtained by the different simulation tools ($μ_{TT_{UTASiMo}} - μ_{TT_{MicroSaint}} = 0$)

H₁: There is difference between the mean workload obtained by the different simulation tools ($μ_{TT_{UTASiMo}} - μ_{TT_{MicroSaint}} ≠ 0$)

Table 8 summarizes the results of the two-sample t-tests and comparison between the MicroSaint and UTASiMo simulation outputs in terms of total time and workload estimation. Since there is no built-in error assessment method in MicroSaint, error estimations are not taken into account, in this case.
Table 8. Summary of Results for case study 1 simulated with different tools

<table>
<thead>
<tr>
<th>Measure Name</th>
<th>UTASiMo Simulation Output</th>
<th>MicroSaint Simulation Output</th>
<th>t-Test results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean (minutes)</td>
<td>Standard Deviation</td>
<td>Mean (minutes)</td>
</tr>
<tr>
<td>Average Workload</td>
<td>85.301</td>
<td>7.013</td>
<td>84.349</td>
</tr>
<tr>
<td>Total Time</td>
<td>113.912</td>
<td>10.679</td>
<td>114.019</td>
</tr>
</tbody>
</table>

Since the p-value for both tests is greater that the level of significance (a=0.05), the hypothesis cannot be rejected and there is no significant difference between the two simulation outputs. Therefore, the UTASiMo-generated model was considered valid.

The comparison also revealed that building simulation models with UTASiMo requires less time and effort, since the process is automated. UTASiMo also provides a built-in workload and error estimator in contrast to Micro Saint. The latter requires programming in order to define functions that calculate workload and error.

**Case Study 2**

The second case study demonstrates the use of the tool to evaluate a naval task performed by a group of individual watchstanders. We used real task data to construct a hypothetical task scenario and produce a model using UTASiMo. The error and workload simulation outputs are
compared with real data to evaluate the predictive ability of the UTASiMo-generated models. The model building process and validation are described in the next sections.

**Simulation of naval task.** In this case study, we constructed a hypothetical naval task scenario using real data to demonstrate the use of the tool for modeling and simulating a scenario and evaluating the simulation results. First, we need to prepare the Excel spreadsheet that will be used as input to UTASiMo in order to automatically generate the simulation model for the scenario. The UTA methodology is applied for preparing the spreadsheet template. The first steps include the identification and decomposition of the overall task into individual subtasks until they cannot be further decomposed and can be performed by a single human operator. In this case, the overall task is broken down into 104 subtasks, operated by a group of 7 individual watchstanders. The subtasks include communication, monitoring, assessment, identification, planning, and execution. The flow of the overall task is illustrated in Figure 12.
After the task decomposition, we need to provide data such as execution times, task locations, and human operator roles, among others, for each of the defined subtasks. Therefore, estimated timing information, locations, complexity, frequency, equipment, and human operators associated with each subtask are entered in the spreadsheet. This allows for the model building process to occur within Excel, which is a tool that most people are familiar with and they can easily modify it. The next step is to upload the Excel spreadsheet and the control room layout into UTASiMo and run the simulation. The model is then automatically generated, as depicted in Figure 13.
Figure 13. Automatically generated model for naval task

The UTASiMo-generated model represents tasks performed on watch in support of mission accomplishment and estimates the timing, workload, and human error for each task. Crew shift changes are considered in the predictions but they are not modeled. Each operator's state is tracked and the accumulated workload for each operator over the course of the scenario is also estimated. Figure 14 shows the simulation results.
Validation of naval task. The verification and validation of the generated model includes comparisons with the real system data. Animation of the simulation output is used for verification of the model, while examination of quantitative measures takes places for model validation. More specifically, task times, workload, and human error obtained from the simulation model are compared with the real system measures, as described in Table 9.
Table 9. Comparison of real-world and simulated system for navel task

<table>
<thead>
<tr>
<th>Measure Name</th>
<th>Real Data</th>
<th>UTASiMo Simulation Output</th>
<th>t-Test results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean (minutes)</td>
<td>Standard Deviation</td>
<td>Mean (minutes)</td>
</tr>
<tr>
<td>Average Workload</td>
<td>59.6</td>
<td>22.8</td>
<td>55.7</td>
</tr>
<tr>
<td>Time per task</td>
<td>1.07</td>
<td>2.06</td>
<td>1.15</td>
</tr>
<tr>
<td>Errors</td>
<td>Low</td>
<td>-</td>
<td>Low</td>
</tr>
</tbody>
</table>

Two basic types of data are derived from the model and compared with the real data; workload data and task time data for each watch position. Workload data are presented at four levels: second-by-second workload per operator graph (Figure 14); average workload per task (Figure 15); average workload per operator (Figure 16); and estimation of the average workload for the overall task (Figure 14). Time data are presented at three levels: average time per task and percentage of time allocated to each task (Figure 14); average time spent on a task for each operator (Figure 17); and estimation of the average time for the overall task (Figure 14).
Figure 16. Average workload per operator

Figure 17. Average Time per Task per operator
In the second-by-second workload graph (Figure 14), workload over 90% indicates operators becoming overloaded and may begin to omit tasks [104]. Figure 15 indicates that simulated operators performing the tasks with IDs T4, T54, T55, T59 may face extreme workload conditions. According to the simulation results, average total workload is in low levels, which means that errors may occur due to underload. This may also indicate an opportunity to reduce crew size by reassigning some tasks to other operators.

We set-up two-sample t-tests to determine if there is statistical difference between the actual data and the simulation output for each of the aforementioned cases. The level of significance is \( a = 0.05 \). More specifically, we set up the following hypothesis for the mean workload (WL) obtained by UTASiMo and the real system:

\[
H_0: \text{There is no difference between the actual workload and the mean workload obtained by UTASiMo}\ (\mu_\text{WL}_{\text{UTASiMo}} - \mu_\text{WL}_{\text{Real System}} = 0)
\]

\[
H_1: \text{There is difference between the actual workload and the mean workload obtained by UTASiMo}\ (\mu_\text{WL}_{\text{UTASiMo}} - \mu_\text{WL}_{\text{Real System}} \neq 0)
\]

We set up the following hypothesis for the mean time per task (T) obtained by UTASiMo and the real system:

\[
H_0: \text{There is no difference between the actual time per task and the mean time per task obtained by UTASiMo}\ (\mu_\text{T}_{\text{UTASiMo}} - \mu_\text{T}_{\text{Real System}} = 0)
\]

\[
H_1: \text{There is difference between the actual time per task and the mean time per task obtained by UTASiMo}\ (\mu_\text{T}_{\text{UTASiMo}} - \mu_\text{T}_{\text{Real System}} \neq 0)
\]
Tables 10, 11, and 12 summarize the results of the two-sample t-tests and the comparison between the real-world system data and the simulation output of the UTASiMo-generated model for each operator.

Table 10. Comparison of average time per task between the real and the simulated system

<table>
<thead>
<tr>
<th>Measure Name</th>
<th>Real Data</th>
<th>UTASiMo Simulation Output</th>
<th>t-Test results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Standard Deviation</td>
<td>Mean</td>
</tr>
<tr>
<td>Operator 1</td>
<td>0.83</td>
<td>1.04</td>
<td>0.85</td>
</tr>
<tr>
<td>Operator 2</td>
<td>0.47</td>
<td>0.78</td>
<td>0.48</td>
</tr>
<tr>
<td>Operator 3</td>
<td>0.93</td>
<td>0.85</td>
<td>1.16</td>
</tr>
<tr>
<td>Operator 4</td>
<td>0.44</td>
<td>0.82</td>
<td>0.48</td>
</tr>
<tr>
<td>Operator 5</td>
<td>0.67</td>
<td>1.18</td>
<td>0.69</td>
</tr>
<tr>
<td>Operator 6</td>
<td>1.98</td>
<td>2.83</td>
<td>1.90</td>
</tr>
<tr>
<td>Operator 7</td>
<td>1.74</td>
<td>3.72</td>
<td>2.06</td>
</tr>
</tbody>
</table>
Table 11. Comparison of average workload between the real and the simulated system for each operator

<table>
<thead>
<tr>
<th>Measure Name</th>
<th>Real Data</th>
<th>UTASiMo Simulation Output</th>
<th>t-Test results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean (minutes)</td>
<td>Standard Deviation</td>
<td>Mean (minutes)</td>
</tr>
<tr>
<td>Operator 1</td>
<td>62.05</td>
<td>15.35</td>
<td>55.34</td>
</tr>
<tr>
<td>Operator 2</td>
<td>50.42</td>
<td>18.88</td>
<td>53.77</td>
</tr>
<tr>
<td>Operator 3</td>
<td>56.30</td>
<td>23.98</td>
<td>56.43</td>
</tr>
<tr>
<td>Operator 4</td>
<td>56.66</td>
<td>21.84</td>
<td>69.68</td>
</tr>
<tr>
<td>Operator 5</td>
<td>58.84</td>
<td>21.62</td>
<td>60.78</td>
</tr>
<tr>
<td>Operator 6</td>
<td>48.44</td>
<td>22.79</td>
<td>59.32</td>
</tr>
<tr>
<td>Operator 7</td>
<td>57.40</td>
<td>11.61</td>
<td>64.47</td>
</tr>
</tbody>
</table>
Table 12. Comparison of errors between the real and the simulated system for each task

<table>
<thead>
<tr>
<th>Error per Task</th>
<th>Real Data</th>
<th>UTASiMo Simulation Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium</td>
<td>T20, T60, T85, T94, T101</td>
<td>T20, T60</td>
</tr>
<tr>
<td>Low</td>
<td>All others</td>
<td>All others</td>
</tr>
<tr>
<td>High</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Since the p-value for all tests is greater that the level of significance, we cannot reject the null hypothesis and there is no significant difference between the observed data and the simulation output for the generated model. Therefore, the simulated model was considered valid.

Table 12 summarizes a comparison of qualitative data about task criticality of the real system with the simulated error probability to show the predictive ability of the model. The UTASiMo-generated model was able to predict all but three errors. However, the low error probability does not always indicate the criticality of the tasks. Therefore, more work needs to be done to improve the overall reliability and to overcome the current limitations of the human error estimation. Overall, a simulation model is usually a simplification and approximation of the real system. "There is no such thing as absolute model validity, nor is it even desired."[105]. Therefore, the simulated model was also considered valid in terms of error estimation.
CHAPTER 6:  
DISCUSSION AND FUTURE WORK

This chapter summarizes the research contribution and provides directions to continue this work. The chapter starts with a research overview, followed by a summary of the research contributions. Finally, the chapter addresses research limitations and future research directions.

Research Summary

This research led to the implementation of UTASiMo (Universal Task Analysis Simulation Modeling) [85-87], a multi-method simulation tool that automatically generates simulation models for task analysis. UTASiMo provides users with the capability to construct task analysis simulation models in a very short time by simply uploading a spreadsheet (i.e. Excel) onto the tool. The UTASiMo-generated models can then be used to experiment with human-, task-, and system-related variables for the purpose of optimizing the system design. By performing the experimentation on the computer, changes can be tested in the simulation model prior to implementation and without modifying the real system or disrupting operations. UTASiMo may also be used to evaluate various system and staffing configurations to identify the system design with the lowest error probabilities, workload, and task execution times.

The first reason that motivated this implementation is that the majority of task analysis simulation tools require well-developed computer simulation skills. Moreover, they lack features that can provide a more holistic representation of a complex system. This is achieved through the integration of three simulation methods, AB, DES, and SD. The second reason is that simulation
studies can be time-consuming and costly. Therefore, the tool was designed to drastically reduce the time required for constructing a simulation model by automating the modeling process.

The model conceptualization was the first step toward the development of the multi-method simulation model. A generic 3M&S framework was followed to help in the conceptualization process.

This research also presented two case studies to verify and validate UTASiMo. In the first scenario, the simulation results produced by UTASiMo were compared with those produced by MicroSaint for validation purposes. In the second scenario, the UTASiMo simulation output was compared to real-world data to show the predictive ability of the tool.

**Research Contribution**

One of the key contributions of this dissertation includes the development of a simulation-based application for automating the model building process, which drastically reduces the time and effort of the analysts. The capability of automatically generating task analysis simulation models in a short amount of time, without the need of programming or simulation theory knowledge, is one of the major unique advantages of UTASiMo. This allows for flexible and easily accessible simulation of models. More specifically, UTASiMo automatically generates simulation models for analysis of human tasks based on a spreadsheet template. The generated models integrate DES, AB, and SD simulation methods and produce simulation results that can assist in selecting among different alternative scenarios.
The dissertation also proposed a methodology that supports the simulation application. The UTA methodology combines different concepts and aspects from multiple task analysis methods, including Hierarchical Task Analysis, Timeline Analysis and Spatial Operational Sequence Diagrams, to analyze tasks in various scenarios. For example, UTA adopts the hierarchical decomposition approach of a task into subtasks, includes the estimation of the execution time of each task step, and provides a graphical representation of a flow diagram linking the task steps in the order they are performed. Integrating the strong points of each task analysis methods in one methodology allows for making more clear inferences about changes that a task may demand [2].

**Limitations and Future Work**

The UTASiMo-generated models are high level representation of the real systems as they were designed to include common elements applicable to any domain. If a more detailed representation of the system is needed, UTASiMo needs to be customized to generate models that include elements relevant to the specific system. Example of such elements include a more extensive list of FAEs for a specific scenario, modules for estimating fatigue, more states for human agents (sleeping, fatigue, eating, etc), among others.

Our future goal is to work on overcoming some limitations of the tool. Currently, UTASiMo has the ability to generate and analyze models that involve sequential task execution. UTASiMo needs to be able to simulate operators working on more than one task at the same time. Therefore, a parallel task execution module will be developed and integrated into
UTASiMo in a later stage as it involves special considerations. The tool also lacks a communication module to allow for agent collaboration. Finally, the tool currently uses triangular distribution for task estimation. In the future, more distribution types will be incorporated in the tool.
APPENDIX:

LIST OF SUBMITTED AND PUBLISHED PAPERS


REFERENCES


