Spring 2012

Exploring the Components of Dynamic Modeling Techniques

Charles Daniel Turnitsa
Old Dominion University

EXPLORING THE COMPONENTS OF DYNAMIC MODELING TECHNIQUES

by

Charles Daniel Turnitsa
B.S. December 1991, Christopher Newport University
M.S. May 2006, Old Dominion University

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirement for the Degree of

DOCTOR OF PHILOSOPHY

MODELING AND SIMULATION

OLD DOMINION UNIVERSITY
May 2012

Approved by:

Andreas Tolk (Director)

Frederic D. McKenzie (Member)

Patrick T. Hester (Member)

Robert H. Kewley, Jr. (Member)

ABSTRACT

EXPLORING THE ELEMENTS OF DYNAMIC MODELING TECHNIQUES

Charles Daniel Turnitsa
Old Dominion University, 2012
Director: Dr. Andreas Tolk

Upon defining the terms modeling and simulation, it becomes apparent that there is a wide variety of different models, using different techniques, appropriate for different levels of representation for any one system to be modeled. Selecting an appropriate conceptual modeling technique from those available is an open question for the practitioner. Existing methods for describing techniques do not capture enough information about the expressive potential of those techniques such that an appropriate selection decision can be made. A formal method to describe conceptual modeling techniques that captures enough about the technique to distinguish it from others is identified as a way to address this gap in the body of knowledge. Such a formal method is derived, and is given additional expressive strength in the special area of dynamic components of conceptual modeling techniques. Application of the formal method to actual conceptual modeling techniques is exhibited, and the capacity of the method to also identify the potential for extension of an existing method is also exhibited. Measures of merit, designed to evaluate the derived method, are tested and shown to be satisfied.

Keywords: conceptual modeling, dynamic models, formal methods

DEDICATION

As ever, this work is dedicated, with love and gratitude, to Anita and Heidi. One the most lovely, the other the most precocious, sometimes the difference is hard to measure. I can never express enough thanks to both of them for all of the support and longsuffering and inspiration throughout the process that led to this dissertation. With their help, and God's grace, the results are found herein.

ACKNOWLEDGEMENTS

The education and inspiration to continue on my own was granted me, in this discipline, by the Modeling, Simulation and Visualization Engineering Department in general, and my advisor (and friend) Dr. Andreas Tolk in particular. In addition, throughout most of the process, I was a contributor to research performed at the Virginia Modeling, Analysis and Simulation Center. I acknowledge that time as being beneficial to my ability to perform doctoral level research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

1. INTRODUCTION

This is the written dissertation prepared in pursuit of a PhD in Modeling and Simulation, as issued by the Modeling, Simulation, and Visualization Engineering department of Old Dominion University. As such, it represents a course of research undertaken, at the doctoral student level, in the discipline of Modeling and Simulation. This dissertation is offered up, in partial satisfaction of the requirements for earning the degree from Old Dominion University's Department of Modeling and Simulation in the Batten College of Engineering and Technology.

1.1 TOPIC

The topic of this dissertation is the evaluation of conceptual modeling techniques. Through a study of the literature of modeling, and the types and approaches to applying conceptual modeling techniques, it is apparent that there are many different types of modeling, done with a wide variety of different modeling techniques. In order to apply some rigor to the identification and comparison of these techniques, a study of them has been completed, with what the body of knowledge expresses concerning modeling and models. The identified gap in the body of knowledge that is identified in chapter two of this dissertation shows that although there exist methods to categorize and describe the representational capabilities of a modeling technique, those methods are individually incomplete to address the full spectrum, leaving the modeling and simulation professional without the tools required to properly identify and evaluate those techniques. The research method followed in preparation of this dissertation that led to answering that gap

is described in the third chapter. Also in that chapter, measures of merit are presented that can address the specific issues that were identified. The chief outcome of the research followed for this dissertation is presented in the fourth chapter. There, an evaluation of what the literature states in order to specifically address the requirements from Chapters Two and Three is made. From this second, focused, literature review a theory describing what the different parts of a model are, how they inter-relate to each other, and how together they provide the capacity for a modeling technique to represent a system. The findings of the research reveal that the representation of the dynamic aspects of a model are currently limited, within the modeling techniques evaluated for the dissertation, the method for describing models that is presented is given particular detail for describing the parts of a model has specific detail applied in the area of what it is that makes a model capable of representing dynamic aspects of a system that is being modeled. This investigation into, and resulting ability to describe (as a derived formal method), dynamic aspects of a model are the main contribution of the resulting system. Having discovered what these components of a model are, they are then each examined in great detail, and a theory is derived describing what the ideal role the components would play within a modeling. Against this derived formal method, a modeling technique can be evaluated, in order to see the representation of each part is within that technique. Employing the theory to describe a number of different modeling techniques is also presented in Chapter Four, to illustrate the applicability of the formal method. The application of the method uncovers an interesting feature common to all of the considered conceptual modeling techniques, and shows how it can be useful to uncover similar features by applying it elsewhere. In Chapter Five, the measures of merit that are

described for judging the successful completion of the dissertation (by answering the research question, either yes or no) are shown, and the success of the dissertation is established against these. Finally, a summary of what has been done, tying the derived formal method, and how it is intended to be employed, to the original gap and research question. This summary, along with recommendations for future research are presented in Chapter Six. An enumerated preview of these chapters is presented later on in this chapter; they are only briefly addressed here to let the reader expect what is to follow.

As mentioned the dissertation has as its main contribution, the investigation of how modeling techniques handle the dynamic nature of the systems they are representing. In pursuing research in this area, it became clear that in order to fully define how the parts of a model (processes) that describe the dynamic activity can fully describe what changes, it became necessary to describe all of the parts of the model to the depth necessary to be aware of what can change about them. If the possibility of change is accepted (and to have a dynamic model, there must be allowance for change), and that possibility is not to be constrained, then the possibility for anything about the model to change must be accepted. This is why exploration of all parts of the model became necessary during the research, and why all possible functional parts of a model are presented within the theory in chapter four. For a model to be able to express change to any part of itself, all parts must be at least identified and that definition must be parameterized at least to the extent of understanding what dynamic change could possibly mean.

1.2 MOTIVATION

There are many possible reasons for modeling. Equally important, there are many possible techniques for modeling. All of these reasons and techniques are prejudged to be valid, for some specific reasons, but assumed that none are valid for all possible reasons. In order to select a model, some formal method should be available to the practitioner so that specific distinctions between models can be made. While this may be possible with currently available methods for static elements of a model (for example, data modeling), it is assumed that comparing the dynamic elements of models remains a worthy area of study, if only because dynamics can be described in so many different ways. None of this as taken as a given within the dissertation, but it is all shown as the result of literature review, and within a soundly derived formal method. These terms (model, simulation, and conceptual model) are all defined and explored within the dissertation itself (first literature review, in Chapter two), but the understanding of what models can represent about the dynamic nature of a system was the motivation for the student in undertaking this topic.

1.3 CONTENTS OF THE DISSERTATION

The Dissertation follows a six chapter outline. It begins with a literature review, and then follows a sequence of steps designed to result in new theory, that has been

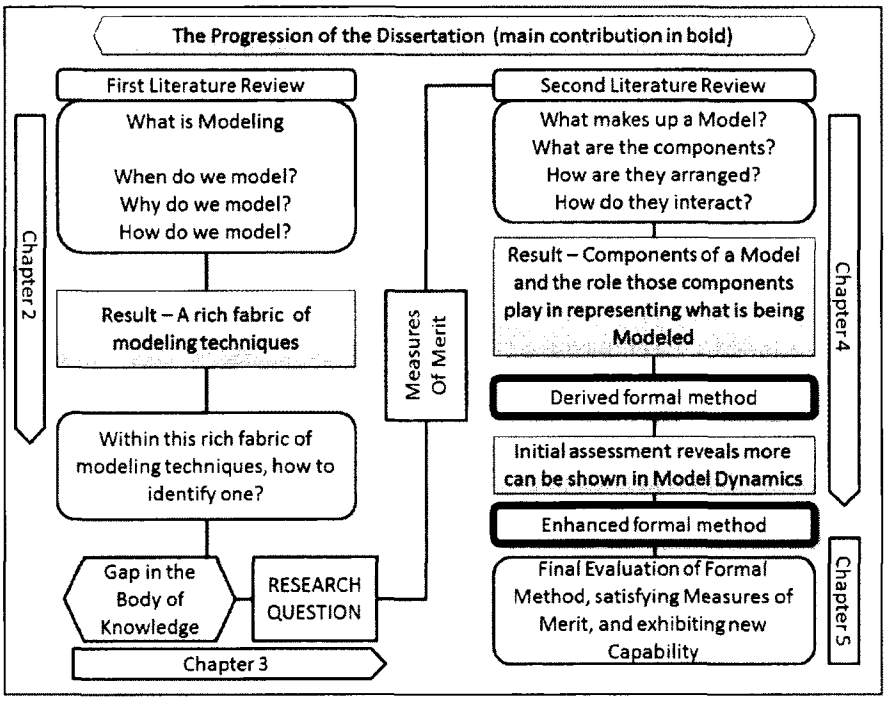

Figure 1. **Progression of the Dissertation**

evaluated, shown to answer the research question, and adds new capability to the body of knowledge. A diagram is presented in Figure 1 giving the general flow of the chapters, and what they contain.

The contents of the chapters are as follows:

1. This chapter presents the introduction to the dissertation, including this overview of its contents.
2. The second chapter presents a literature review leading to an identified gap in the current body of knowledge. This is the first literature review (of two) within the

dissertation, and results in several pieces of information that are vital to the dissertation. First are a number of definitions, based on those found in the literature, for a number of terms. Second are a number of different elements from the literature that are relied on to distinguish the different kinds of conceptual modeling techniques that exist. In examining this, it is seen that there are, for a single modeling exercise, many different stages at answering a question about a system as to when modeling can be applied (describing when we model); there are many possible viewpoints that answer different types of information about the system in question (describing why we model); and many different models (describing how we model).This results in a very broad spectrum of possible conceptual modeling techniques and approaches to select from. To serve as the basis for studies made within the dissertation, a brief review of several techniques from the literature is presented, with the distinctions provided by the distinguishing elements from the state of the art, to categorize those techniques. From this the gap in the current state of the art is identified.

3. The third chapter takes the gap identified in the second, and then develops it into a specific research question, highlighting the details that need to be answered in order to close the gap. This is then supported by a research methodology that describes the steps taken to answer that question, and shows a roadmap of the remainder of the dissertation. Measures of merit, derived for the purpose of the dissertation, are presented that can be used to determine whether or not the research gap is addressed by the dissertation. The purpose for measures of merit is to give an independent reviewer of the research the specifics by which the

resulting formal method was evaluated in order to judge whether it is successful at answering the research question (and, hence, closing at least part of the identified gap in the body of knowledge).These measures are addressed again, in chapter five, to show that they have been satisfied during the evaluation of the formal method.

4. The fourth chapter takes on the job of providing the answer to the research question. It does this in three parts. The first is to identify, through a second literature review, what the literature states about the components of a conceptual model – what they are, what function they perform, and how they work together. This first part synthesizes the results of that review, and develops a rational construct resulting in a formal method to describe the components of a model, and what role each should perform, based on the results of that literature review. The second part of this chapter is perform an evaluation of the resulting formal method from the first part. This is done by applying the formal method to a cross section of different conceptual modeling techniques (a subset of those presented in the second chapter), in order to see the usefulness of the formal method in covering all portions of the techniques selected, and to see if the formal method provides that answers required to answer the specifics of the research question. Finally, the third part of this chapter is to enumerate what the implications are of the formal method, especially concerning processes within conceptual modeling techniques. This will add to the state of the art concerning a descriptive language about processes within modeling techniques, and within models, and further supports the answering of the research question.

5. The measures of merit from the third chapter are examined again, in light of the formal method that was developed and presented in Chapter Four. It is shown that the evaluation of the formal method answered the questions that are set up in the measures of merit. In addition, in showing how the third of three measures of merit are answered, this chapter presents the application of the formal method from the preceding chapter, in a new way other than how it was applied in that chapter. This new aspect of the research question not addressed in the previous chapter is the use of the formal method to evaluate a single technique in order to motivate extensions of that technique. The formal method is applied in this chapter for specifically that reason – applying it to a modeling technique not yet addressed in the dissertation, and identifying and describing specific extensions to that technique that will result in a modeling technique that can represent information about a system more efficiently, and more clearly than without the extension. This is a new capability, now available to the body of knowledge, which the formal method delivers.
6. The final chapter draws to a conclusion the findings presented in chapters four and five, and specifically addresses how they answered the enumerated points of the research question presented in chapter three. In addition, the potential for future work is presented in this chapter.

1.4 PURPOSE STATEMENT

The purpose of this doctoral dissertation is to make a study of the representation of processes within modeling and simulation, specifically how they are represented within conceptual modeling techniques. From a literature review of the subject,

something about the nature of processes within the body of conceptual modeling techniques will be revealed, and this will be addressed by development of a theory of what components a model can consist of, how one of these is a process component, and the specific requirements for fully describing a process. A gap in the body of knowledge is identified, and an associated research question, where specific details that can be used to describe the processes in a modeling technique will be uniquely suited to answer the question and close the gap.

CHAPTER 2

STATE OF THE ART

This chapter begins with a literature review of the terms that are appropriate to the dissertation. These terms include "modeling", "simulation", "conceptual modeling" and "systems". Following that it begins a survey of different modeling techniques, shows how they model the system they are representing, and what specifically is in that representation. This survey then results in a second more specific look at the dynamic nature of systems that the modeling techniques can represent, and out of that a gap in the current state of the art is identified.

The literature review (p. 12) covers a wide variety of terms in three parts, but there is a reason (plan) for the terms that are covered, and the order in which they are covered. First the literature sources that give definitions for the terms 'modeling' and 'simulation' are given, and this is to show that they are very separate terms, representing separate pursuits. However, even though the topic of the dissertation is the investigation of modeling techniques, because modeling is often (not always) related to simulation, some definition of the latter is required. After settling the definitions for these, the literature on why modeling (and simulation) is done is addressed. This gives the general case for why models are developed, and more specifically, why those models are implemented as simulations. The literature next covered is to show how there are many different types of models, done for many different reasons, even within the limited case of using models to answer a question about a system, and even more limited to models that will eventually lead to a simulation. It is illustrated that within even so narrow of a consideration area, there are already a wide number of cases of different stages of

modeling, to satisfy different uses. This then leads to the literature on conceptual modeling (p. 29), and exhibits what the state of the art in that area is. Specifically covered are the different uses for, and different reasons for, conceptual models. This further adds to the number of different approaches, even within one project, that can be taken for modeling. Finally, it is shown (p. 29) that the literature presents the view that models are captures of information about whatever system they are representing. As such, the literature is relied on to show that many perspectives of that system can be the basis for a model, further illustrating the rich fabric of possible models and modeling approaches that are possible. Having established that the literature presents a wide variety of different approaches to

- How to model
- Why to model
- When to model

These questions all answered in response to the model (or models) being used to answer some question about a system, where information about the system must be known in order to successfully derive that answer. The third and last section of the literature review (p. 42) shows what the literature reveals about the capture and representation of information about a system, and how a model must describe the composition and behavior (activities) of the system in question. It shows how the literature for several modeling traditions (systems models, conceptual models, and ontological models) reveals that this composition of a system can be represented with a commonality of components.

Following the literature review, a survey of existing modeling techniques (p. 60) is presented. In order to categorize and understand the possible commonality or differences among them, first a reason for why the particular techniques are selected is given (p. 60) and then an overview of methods for categorizing the possible characteristics of those techniques (p. 63) is given. Finally, the specifics of the survey and some observations about the chosen techniques are given (p. 66).

What it is about the chosen techniques that allow them to represent the dynamic behavior of systems is presented in (p. 88). This section contains two parts, the first of which (p. 89) showing what the literature has to say about what a model can address concerning dynamic behavior, and the second part (p. 96) showing what the literature has to say about specific typological characteristics of models.

Given the first three sub-chapters of this chapter - the literature review, survey of techniques, and overview of dynamic elements of models – the final sub-chapter (p. 98) can identify and summarize a gap in the current body of knowledge (state of the art) concerning models and the ability to identify them.

2.1 LITERATURE REVIEW

The discipline of modeling and simulation is simultaneously as old as human language (modeling) and as new as digital automata (simulation, at least on a computer).The literature relating to such a discipline is varied and rich while also being quite diverse in the sources it comes from. Beginning with an overview of how the literature defines the two constituent parts of the discipline (modeling and simulation), the review will end with specific topics of interest to this dissertation.

Modeling and Simulation. The name of the degree conferring department at Old Dominion University that grants degrees in modeling and simulation has a name that is indicative of the common understanding of this domain within the community. The department's proper name is the Department of Modeling, Simulation and Visualization Engineering (Found online at http://eng.odu.edu/msve/ last checked October, 2011). The degrees offered at the Bachelor's, Master's, and Doctoral level all have the name of a degree in "Modeling and Simulation Engineering". In all these cases, the terms modeling and simulation are used together. This is not out of step with the remainder of the community; in observation, the common abbreviation used is M&S – modeling and simulation, together. Together the two have been identified (Padilla, et al., 2011), recently to constitute a discipline of three different hues. Those three, science engineering, and application are identified as follows:

M&S Science contributes to the theory of M&S, defining the academic foundations of the discipline.

M&S Engineering is rooted in theory but looks for applicable solution patterns. The focus is general methods that can be applied in various problem domains.

M&S Applications solve real world problems by focusing on solutions using M&S. Often, the solution results from applying a method, but many solutions are very problem domain specific and are derived from problem domain expertise and not from any general M&S theory or method.

The question as to whether or not there is an identifiable theory of M&S within the literature is not the point being made here, just that there has been an observation (Padilla, et al., 2011) of the differences between the science, engineering and application

of modeling and simulation. Together the two terms, without breaking them into separate pursuits (as above), is defined by (DoD, 2007) as "one of the key usages of architecture data to enable evaluation of the logical, behavioral, resource and performance characteristics of systems". As will be shown later in (p. 42) there is indeed a direct link between a model and the system it represents. There is also a link between a simulation and the model it implements (p. 11).

However, each of these terms has its own particular meaning, and although they complement each other in some pursuits, understanding the individual definitions is useful.

Modeling. The English term *model* is at least several centuries old. It came into the English language in the 16th century either from the French *Modelle*, or from the Italian *Modello*. At that time, it was used in reference to the plans referred to by groups of craftsmen and artisans working on a building, in both cases the original word is the Latin *Modulus*, and appears to also be related to the mathematical idea of the same name (OED, 1989a). In more modern usage, it is used in a number of disciplines that contribute to that of modeling and simulation, yet in almost all cases there are some dissimilarities in the definitions relied on. A lay definition includes "a usually miniature representation of something" (Merriam-Webster, 2011) and, related but closer to our discipline, "a system of postulates, data, and inferences presented as a mathematical description of an entity or state of affairs" (Merriam-Webster, 2011). Moving to an academic, yet still general, source (Schichl, 2004) can be cited as defining a model as "an abstraction of reality". This is a useful definition when considering the history of models, and how they came out of the use of abstract numbers and language. Numbers led to

mathematics, and language led to early (classical) philosophy. Both are attempts to abstract the world into something that can be systematized.

Moving towards modern definitions, it will be seen that the idea of model representing a system has arisen. It can be seen (Broy, 2003 p. 208), "a model is a representation of a system". This view also held by Banks (1998), is the prevalent definition in the much more general, "a model is a representation of a system" (p.5). For definition of a system, the IEEE has an authoritative definition (IEEE, 1990):

"A collection of components organized to accomplish a specific function or set of functions" (p. 14).

Following that definition, it can be seen that the model itself is a system, however it must be remembered that the model (even if a system) is intended to represent another – which brings out the idea of simplification or abstraction (or else the original would be more useful to refer to). Following (Banks, 2009) we see this definition of model that includes in the definition that the model can be of an actual referent, or of a contrived referent:

"A model is a representation of an event and/or things that is real (a case study) or contrived (a use-case). It can be a representation of an actual system. It can be something used in lieu of the real thing to better understand a certain aspect about that thing" (p. 5).

A utilitarian definition that brings in the aspect of a model being some simplifying representation is found in (Zeigler, et al., 2000 p.29), "a model is a simplifying abstraction of reality." From Banks (1998) it is recommended that the abstraction should be simplified as much as possible, only complex enough to provide answers to questions

the model is intended to address. Bringing together these ideas of representation and abstraction results in the definition found in (Tolk, et al., 2009) of modeling as "the purposeful abstraction of reality and capturing of assumptions and constraints". A second view on the abstracting relationship between the model and its referent is seen in (Banks, 2009):

To produce a model you must abstract from reality a description of a vibrant system. The model can depict the system at some point of abstraction or at multiple levels of the abstraction with the goal of representing the system in a mathematically reliable fashion. (p. 5)

The inclusion of this definition is to show that a model, although an abstraction, is purposeful. It is developed with a purpose, which means from a perspective, with intentionality, to address a particular need. Bringing the idea of purpose in with representation and abstraction, the result from Hester and Tolk (2010) applies, "modeling is the process of abstracting, theorizing, and capturing the resulting concepts and relations in a conceptual model" (p. 3). In this definition, the idea of representation is described as "theorizing", which shows how a model is a theory for the system it is representing. While it brings together these several identified elements of modeling (purpose, representation, abstraction), this definition also introduces the term "conceptual model" as the resulting artifact that is derived from the process of modeling. This is the definition for modeling that will be relied on in this dissertation.

Definition 1: Modeling is the purposeful process of abstracting and theorizing about a system, and capturing the resulting concepts and relations in a conceptual model.

As will be shown later (p. 27) on defining what conceptual modeling is, the community views the use of models in several ways. Two of these ways will be described, based on references from the literature; however this definition does not imply either view to the exclusion of the other. Before discussion simulation, then, it might do well to address that the definition of model that has been distilled from the literature does not necessary imply that there will be a simulation that results from that model. There may be such a relationship, however it is not presumed by the definition.

Simulation. The term simulation is often equated with model. As with model, simulation, or simulate, has been in the English language for centuries. From the Latin word Similis, the word came into English in the middle of the 17th century, and is related to the idea of 'to be like', or similar (OED 1989b). Lay definitions (Merriam-Webster, 2011) often refer to the words model and simulation as synonyms. Before progressing, it should be pointed out that where the term "simulation" is used herein it is referring to a simulation system, or the use of such a system (typically a computer simulation, or its use). Where the meaning differs from this, it will be mentioned in the text. A common definition of simulation might include the idea of enabling or implementing a model. From the literature it is clear that this is held to be true. It can be seen in Banks (1998) that simulation is "the imitation of the operation of a system over time" (p. 3). The idea that the identity and execution of the simulation come from the model is reinforced by Bratley, et al. (1987), "Simulation means driving a model of a system with suitable inputs and observing the corresponding outputs" (p. 2).

A model, as the chosen definition Hester and Tolk (2010) states, is "the process of abstracting, theorizing, and capturing the resulting concepts and relations in a conceptual

model" (p. 2); that source then goes on to define simulation as "the process of specifying, implementing, and executing the model" (p. 2). Following Banks (2009) with the definition of model found there (see above), that source also presents a definition of the term simulation that is illustrative.

A simulation is an applied methodology that can describe the behavior of that system using either a mathematical model or a symbolic model. Simply, simulation is the imitation of the operation of a real-world process or system over a period of time. (p. 3)

Combining these views of simulation, found in the literature, and considered as the means for actualizing a model in order to represent a system, it can be seen that a valid synthesis can be rendered as:

Definition 2: Simulation is the process of specifying, implementing and executing a model.

While Definition 1 did not imply that a simulation would necessarily result from a model, Definition 2 explicitly states that all simulations are implementations of a model. Definition 1 defines an activity that ends with a captured conceptual model. As Definition 2 defines an activity that is based on having access to a model, then when both activities are combined, it is possible that a single human agent will conflate these two processes – deriving a conceptual model as a product of the mind, and committing it to a simulation as a single procedure. While possible, not having the model in form that can be later referenced limits many of the activities that rely on such a model, as will be seen from literature referenced in the following sections.

Why Modeling and Simulation. Reasons for modeling, historically, are to analyze (usually mathematically, as has been shown, but perhaps otherwise) or subject

the representation to some investigation that would be problematic with the referent it represents. This is shown by Law and Kelton (2000) as the sequence of decisions to be made, in order to determine whether simulation is the correct approach to study a system. From Law and Kelton (2000), it may be possible to experiment with the actual system; if this is not possible then it becomes desirable to experiment with a model of the system. When experimenting with a model, it may be possible to experiment with a physical model, if not then it becomes desirable to experiment with a mathematical model. If it is possible that a mathematical model can result in a closed form solution, then doing an analytical solution is possible, if not, then it is finally desirable to implement the model as a simulation (*Figure 2.* Reasons for modeling and simulation).

Once the decision to undertake modeling and simulation has been made, as guided by Law and Kelton (2000), then it should be established what the process of applying modeling and simulation should accomplish. In Banks (1998) it is stated that the model should "complex enough to answer the questions raised, but not too complex" (p. 6). Taking a closer look at what another source identifies, the US Department of Defense Modeling and Simulation Coordinating Office (MSCO) (2006) has presented guidance as to what the modeling and simulation exercise should accomplish. It is to answer the following questions:

- Which particular aspects of the problem will be addressed by the model or simulation (i.e., what is the specific application)?
- What requirements need to be met to find a solution? What aspects of the problem domain need to be addressed? What characteristics of the user domain need to be included?

- What capabilities does the model or simulation need in order to address these issues?
- What decisions will be made on the basis of M&S results?
- What are the ramifications of improper modeling? What risks are involved if erroneous results are accepted?

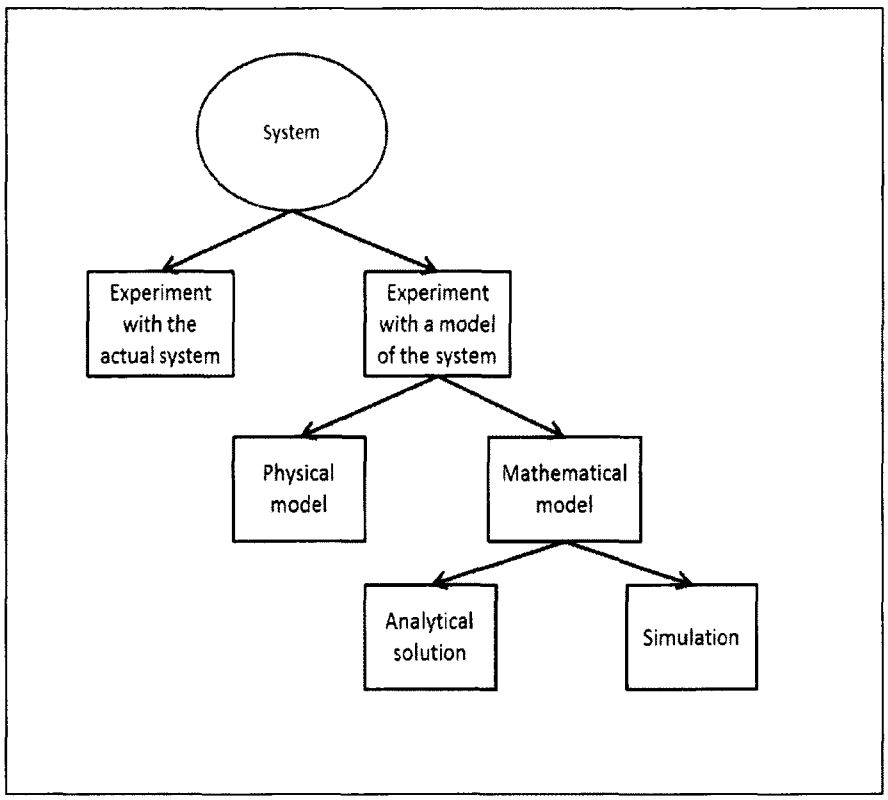

Figure 2. **Reasons for modeling and simulation**

- What acceptability criteria are used to determine when success has been achieved?

That modeling and simulation is the appropriate technique, illustrates how a simulation can allow for other effects on the model than are traditionally available to the

mathematician – the simulation approach is useful when a closed-form analytical solution is not appropriate. First, implementing the model as a simulation can allow for the interaction of agents outside of the model (human actors, other models) that can introduce effects based on the particular implementation of the model. Second, simulations can interact with real life systems, when a model cannot. This can be accomplished by a simulation system implementing a model to produce generated behaviors that are then given to some real life system or equipment for some purpose. The model, while it could be analyzed to see what the generated behaviors would be, could not stimulate the real world system unless that model is implemented as a simulation. From the literature, therefore, it can be seen that simulations are distinct from models. Models are capable of being analyzed and mathematically enacted apart from being used within a simulation; however there are additional possible benefits from simulating. Equally, while a simulation requires a model for what it enacts, the simulation can potentially perform beyond what the model could outside of a simulation. The clearest combination of the two terms, however, is derived from Banks (1998). In that source it is stated that a model is "a representation of an actual system" (p. 5). It is also stated, in the same source, that a simulation is "the imitation of the operation of a real-world process or system over time" (p. 6). If the simulation is an imitation, yet it operates over time, it must have its operations given definition, structure, and capacity by something element. That element is a representation of the real-world process or system, which according to Banks (1998) is a model.

It has been shown by Hester and Tolk (2010), that a model is "the process of abstracting, theorizing, and capturing the resulting concepts and relations in a conceptual

model" (p. 75). And from the same source, it has been shown that a simulation is "the process of specifying, implementing, and executing the model" (p. 76). These two can be combined to show that a simulation is the implementation of, and execution of a model over time. This combination is in alignment, and contributes to the distillation that resulted in Definitions 1 and 2.

Multiple Views for Multiple Models. The purpose element of the definition of a model indicates, as stated above, that the developer of the model has a particular purpose in mind. Shown in Sargent (2005), it is stated that "a model should be developed for a specific purpose (or application) and its validity determined with respect to that purpose" (p. 2). As we have seen that a simulation is developed from a model – it is an interpretation of that model, again with a particular purpose in mind. Models may serve other purposes, however a simulation requires a model. The simulation, then, is an executable means of stating what the model is interpreting of the referent system.

The relationship between the object of the simulation (referent), the mental understanding of how the referent functions (model) and the implementation of that model (simulation) can be viewed in light of how semiotics tells us that informational signals (words, symbols, signs) are formed in a similar process. This can be shown by Ogden's diagram describing semiotics (Ogden & Richards, 1923). In its basic form, the semiotic triangle (*Figure 3.* Semiotic Triangle, expressing referent as symbol) illustrates how a communicating agent (a speaker) has the intention of expressing some symbol that stands for some referent. First, the speaker forms a mental conceptualization of that referent, and then expresses that conceptualization in the form of a symbol – words, or an image.

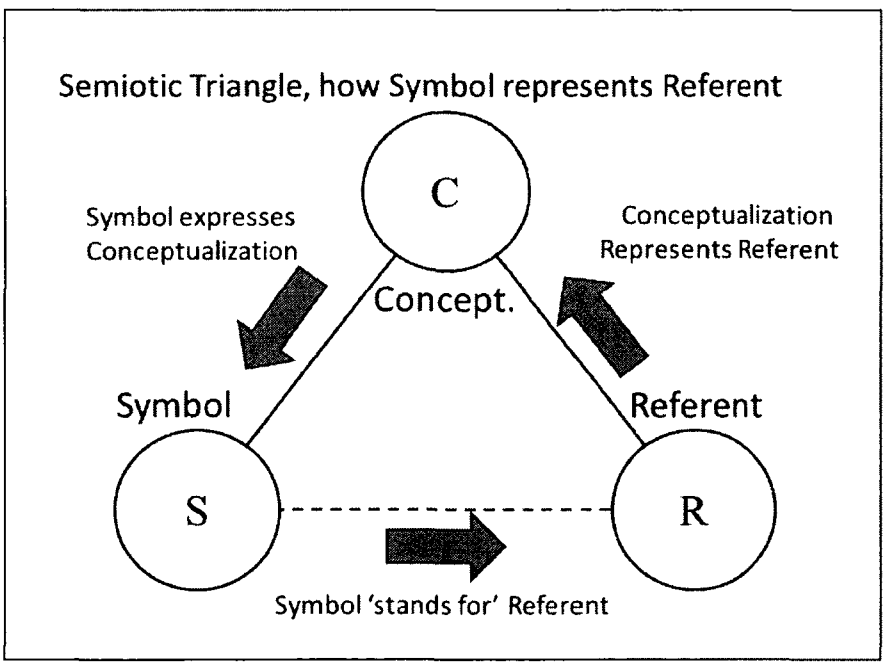

Figure 3. **Semiotic Triangle, expressing referent as symbol**

By seeing that referent can be replaced with referent system, and mental conceptualization with conceptual model, and realize that the expressed symbol is the resulting simulation, then the triangle explains the relationship, in M&S, between the referent, the conceptual model, and the simulation (*Figure 4.* M&S knowledge of a referent, after the Semiotic Triangle). Aligning this explanation with the definitions chosen for both model and simulation show that the purpose element of modeling can have any number of models, each depending on a single purpose that may serve as the conceptualization of the referent. Equally, as no single method for implementation is inherent in the explanation given for a simulation, then there must be any number of possible simulations that may express any of the possible models of the referent.

View this in light of what is presented in Yilmaz (2004). That author shows how for an approach at constructing a simulation of a system, that there are at least two

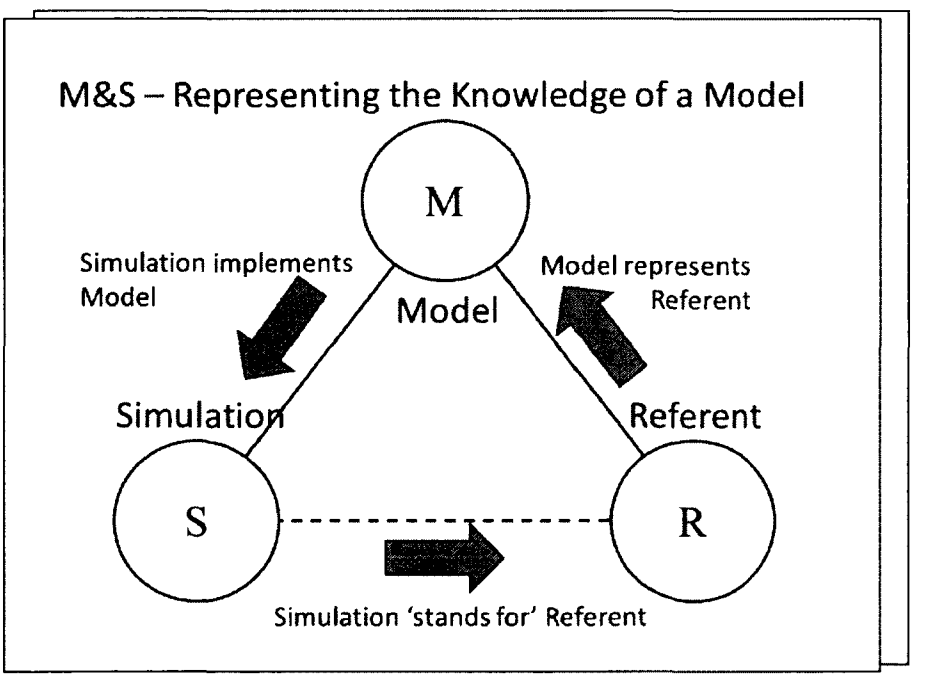

***Figure 4.* M&S knowledge of a referent, after the Semiotic Triangle**

different categories of models. The first is referred to as the conceptual model, which is a capture of the simulation practitioner's idea (conceptualization) of what the system is, and what its essential parts consist of. The second is referred to as the "simulation model", which is a capture of the conceptual model, but expressed in terms that may be implemented, by a computer. It is clear that not only are there any number of possible models that may serve to capture the conceptualization of the referent, but that in a single exercise of producing a simulation to stand for a referent, there are likely to be more than one model developed. Yilmaz shows how a conceptual model can be developed in order to capture the conceptualization that the modeler has of a referent, he also shows a 'simulation model' which is an interpretation of the conceptual model, but its form and structure are aligned to show the conceptualization in terms that may be implemented – it is using the purpose of abstracting and theorizing about the conceptualization, such that those concepts and relations can be expressed in a way making the most sense for implementation. The separation of the semiotic triangle, applied to the M&S concept of representation of a system, now has the model element split, into at least two different elements – a conceptual model and a simulation model. This is to continue to borrow the terms of Yilmaz (2004).

This shows that, at least, there are two possible different stages of developing models – of the same referent, for the same simulation – during the modeling stage. Taking Yilmaz (2004) as motivation, it is not clear how this should be limited to just two, for as an abstraction may be needed to interpret the "conceptual" model for the "simulation" model, then an abstraction may further be needed at any point in between those two, resulting in possibly more models.

The end is that a whole range of models may be possible, although taking the observations shown in Yilmaz (2004), anywhere along this range; a model is likely to be more towards the paradigm of capturing the conceptual idea of the modeler, or more towards interpreting the conceptualizations in terms appropriate for the implementation. These paradigmatic poles in this continuum of possible models can be called the "conceptual" paradigm, or the "simulation" paradigm, following the lead of Yilmaz (2004), but also with acknowledgement of the identification of three distinct models, all used to express the referent, from the doctoral dissertation of Brade (2004). He shows how there is a conceptual model, which is the abstracted and idealized representation of the referent; there is the formal model, which is a formalized version of the conceptual model, following a well-defined modeling formalism; and finally there is the executable model, which describes the formal model in a form that can be implemented (referred to, from here, as the implementable model). Brade's (2004) taxonomy is illustrated in (*Figure 6.* Brade exhibits multiple model views). This last describes the relationships and activities of the formal model in the language of the implementation method – identifying, for instance, procedure calls, data manipulation and so on.

In considering the relationship between the referent and the model, and then the relationship between the model and simulation, in light of Definitions 1 and 2 from earlier, we can see that each of these relationships may result in any number of ends that all properly satisfy the identified definitions. To see this better, consider from Turnitsa, et al. (2010) the following excerpt:

Modeling and Simulation comprises two parts: while modeling resides on the abstraction level, simulation resides on the implementation level. In modeling, we answer

the question what we model, in simulation we answer the question how we model. In order to be able to model, we furthermore need to know why we model (modelers intent), which is defined by requirements derived from the experimental frame or the context of the model, or in academic research the research question. In M&S as in other disciplines, ontologies need to capture reality in a form that is computer understandable. However, in M&S this reality is contingent on the modeler and on a research question. In other words: the focus on this contribution lies on ontological support on formulating the research questions and the modeling, not focusing on simulation.

To envision this, see Figure 5 and consider, as stated earlier, that a model could be implemented at any stage in a continuum across the top edge of the figure, not only as the identified "conceptual model" and "simulation model". To see that this is possible, consider that a model, by Definition 1, is a capture of the results of purposefully abstracting and theorizing concerning the referent. As it is with purpose, the application of each purpose may result in any number of models of the same referent. Purpose is the modeler's world view interpreting the referent for some specific reason – to provide an answer to some question about the referent. There is any number of questions that can be asked of a referent, and there is any number of world views that provide a framework for understanding what the referent is, and what about it should be represented in order to answer the question.

In looking at Definition 2, the requirement we have is that a simulation is an implementation of a model. Implementation, by using a computer programming

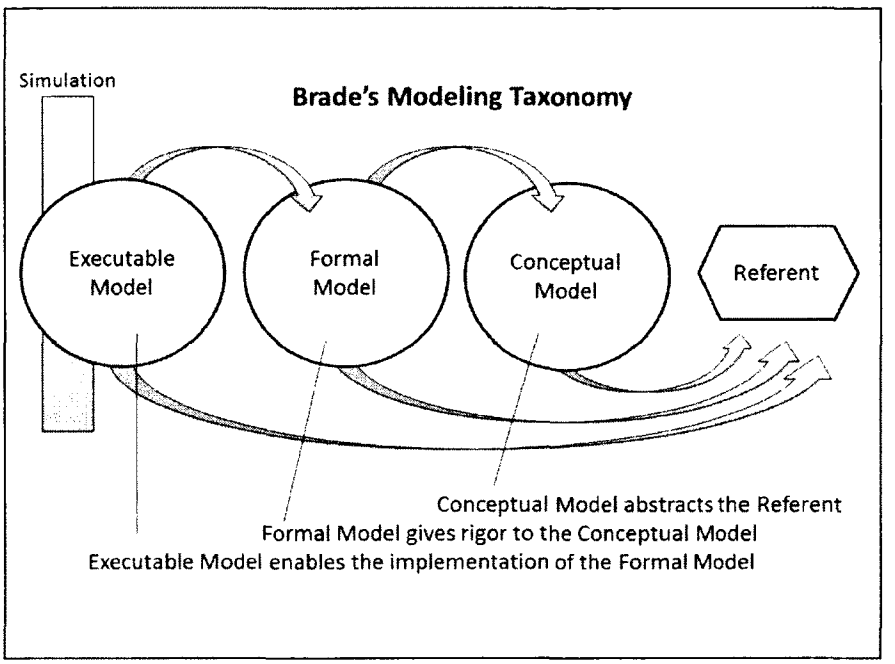

Figure 6. **Brade exhibits multiple model views**

paradigm, can be done many different ways. This is seen from the literature on modeling by looking at what UML (OMG, 2002) has to say that there may be platform independent models, as well as any number of associated platform specific models. This is further corroborated by Yilmaz (2004) and Brade (2004), as will be demonstrated in (p. 30). In order for there to be a platform independent model, also in corroboration with West (1996) that a conceptual model not be restricted to one implementation technique, then the model is free to be the motivator for any number of implementations.

To summarize this (p. 11), it can be seen that a lot comes from the definition just in defining these terms. Models are artifacts that result from the process of attempting capture knowledge that describes a particular perspective on the world, or more precisely, the part of the world (a system) that the model captures knowledge about. Modeling is done for a wide variety of reasons, even in the case explored here of the special relationship between a simulation and the model it is based on. It has been seen that the literature is clear on the point that for any one exercise of going through the process of modeling a system, and then developing a simulation of that system, that there are any number of different perspectives about the relationship, and therefore any number of models each capturing a specific perspective. Finally, we have seen that in order to simulate, we enable an imitation of the real system to be implemented and give some account of the behavior of the system. The imitation of a real system that the simulation is implementing is a model.

Conceptual Modeling. When models are considered in relation to simulations, they are frequently referred to as conceptual models. In this relationship, the literature shows that within the community there are at least two views; the first view is that the conceptual model is considered to be the guiding framework for what the simulation is doing. The second view is that the conceptual model has, apart from its relationship to the simulation, information concerning the intentioned approach to solving a problem. What these two views have to say concerning conceptual modeling is presented here.

Conceptual Model for Simulation Design. The first view, as mentioned, is from a group within the community who consider the Conceptual Model as a way of expressing the intentioned design, for the purposes of guiding the development of, and

describing the requirements for, the resulting simulation. This is in agreement with the traditional view of conceptual models from the US Department of Defense history with distributed simulation systems. This is shown in Pace (2000) by pointing out that this is seen in the view of the Distributed Interactive Simulation (DIS) community view that the conceptual model is "between the simulation developer and the user about what the simulation will do". Pace demonstrates that there is more that can be gotten from the conceptual model, than what the DIS community allows for. He demonstrates that there are two specific things which the conceptual model grants to the design of a simulation. These are (i) that the conceptual model allows the simulation to be evaluated for purposes of testing validity – it is a guide to what the simulation will be doing. And (ii) the conceptual model, by granting the motivation (and upon review, the description) of the behavior of the simulation, gives the simulation engineer the information needed in order to bring the simulation together within a campaign of activity. The first of these two things granted by the conceptual model is endorsed within MSCO (2006), as the conceptual model and its understanding are both key to proper verification and validation methods. The second of these two things is endorsed within Robinson (2008), as it is summarized that interoperability between simulation systems must be solved by aligning the conceptual models, before the technical chores of aligning the inputs and outputs of the simulations themselves. That the conceptual model has, as its intent, the capture of knowledge about the system, is corroborated by Tolk, et al. (2010). That source states:

The goal of conceptual modeling in Modeling and Simulation (M&S) is not focusing on describing an abstract view of the implementation, but to capture a model of the referent, which is the thing which is modeled, representing a sufficient simplification

for the purpose of a given study serving as a *common conceptualization of the referent and its context* within the study. (Tolk, et al, 2010)

The conceptual model provides for the simulation in two broad categories, according to Pace (2000).

- In the first category, it grants context, by codifying the perspective of the modeler who devised the conceptual model including the subject matter, the constraints on that matter, and the assumptions of what properly constitutes the matter.
- In the second category, it motivates the design of the simulation by granting the captured conceptualization.

The second category is done through description of the objects, processes, and relations that the simulation will base its operation on. Together the two, making up the conceptual model, are driven by interpretation of the purpose for the model, and drive the requirements for a resulting simulation. They drive those requirements by defining the potentials for representation and behavior that the simulation can expect to exhibit. From the literature sources presented here (Robinson, 2008; Tolk, et al., 2010; Pace, 2000), it can be seen that this first view of the conceptual model of a simulation system is the motivation for the design of that system, and is also a descriptor of what the system is, and what it is capable of doing.

Conceptual Model aside from Simulation Design. The second view from the literature, on the role of conceptual models, grants more of a unique identity of usefulness to the conceptual model, apart from serving as just a motivator for the simulation, and its development. It is shown (Sargent, 2005) how understanding a simulation system, and evaluating it in a validation and verification process, that understanding and evaluating

the conceptual model is key to several steps in the validation and verification (V&V) process.

Some of the specific techniques in a modeling exercise are described by the literature and procedures found in MSCO (2006). These are grouped into different categories, and are listed here, along with descriptions following those within the original source.

Informal Techniques – these are some basic methods for inspection of a conceptual model, and the model's *facia* (or apparent) correctness. These techniques include:

- Audit – this is an audit of the techniques and processes followed by the staff during development of and implementation of the conceptual model.
- Face validation – this is a technique where a subject matter expert (SME) is asked about the behavior and output of the conceptual model, with regards to the real life system it is emulating.
- Review – similar to an audit, but done after the process of either development or implementation (depending on which stage in the life cycle the review is implemented) of the conceptual model. This technique involves confirmation from the involved staff that the required steps of model development were correctly followed, and at least an affirmation from staff as to model consistency and model completeness.

Static Techniques – these are techniques that involve inspection of and evaluation of the conceptual model as a derived and developed artifact. There are a wide variety of techniques available here, and an overview of some of these techniques is provided

below, as a means for exhibiting some of the information that is expected to be found within a conceptual model. In the pursuit of what it means to model, and the possible uses a conceptual model can be put to, these are the focus of this dive into what the practice of V&V has to say concerning conceptual models.

- Cause-effect graphing – addresses the question of what causes what in the conceptual model. Causes and effects are first identified in the system being modeled and then their representations are examined in the conceptual model.
- Control analysis – techniques include calling structure analysis, concurrent process analysis, control flow analysis, and state transition analysis (Fujimoto, 1993). Includes specifics such as calling structure analysis (which internal processes in the model are called by which others, when, why); concurrent process analysis (important for conceptual models that feature concurrent processes in their representation, such as parallel or distributed models); control flow analysis (how the focus on which processes are currently in control of the activity within the conceptual model is represented and controlled); state transition analysis (what the identifiable states of the conceptual model are, and how they transition from one to another) (Page & Nance, 1994).
- Semantic analysis – is performed on conceptual models that are already closer to what Yilmaz (2004) identifies as a "simulation model". It serves to determine the modeler's intent as reflected in the conceptual model. Evaluation of the representations and symbols used within the model make up this analysis (Whitner & Balci, 1989).

- Symbolic evaluation – is a technique that is especially valuable when working with the conceptual model, rather than results that a simulator based on that model may produce (Adrion, et al., 1982). With symbolic evaluation, analysis is made of the symbols used within the model – checking them for consistency and to ensure that correct use of appropriate symbols is made throughout Dillon (1990).
- Syntactic analysis – is a technique that is simply followed to ensure that the formal structure and practices of the conceptual modeling technique have been followed throughout.

Dynamic Techniques – these techniques are applied to the resulting implemented system (a simulator, capable of performing simulations) that results from the conceptual model. As such, they are of more interest to a deep study into simulators and simulations, rather than into models, but there is enough relation to the conceptual model, and how it influenced the resulting simulator, that a few techniques warrant mention here. Again, these descriptions are taken from MSCO (2006) and modified to illustrate the terms used herein.

- Assertion checking – is a technique whereby a statement made about the results of the conceptual model is evaluated from the results of a simulation based on that model, and then tested to see if the statement remains true throughout.
- Bottom-up testing - Bottom-up testing is used with bottom-up conceptual model development. Many well-structured models consist of a hierarchy of submodels. In bottom-up development, simulation model construction (and the resulting implementation) starts with the simulation's routines at the base level, i.e., the ones that cannot be decomposed further, and culminates with the submodels at the

highest level. As each routine is completed, it is tested thoroughly. When routines with the same parent, or submodel, have been developed and tested, the routines are integrated and their integration is tested. This process is repeated until all submodels and the conceptual model as a whole have been integrated and tested. The integration of completed submodels need not wait for all submodels at the same level to be completed. Submodel integration and testing can be, and often is, performed incrementally (Sommerville, 1996).

- Sensitivity analysis - Sensitivity analysis is performed by systematically changing the values of conceptual model input variables and parameters over some range of interest and observing the effect upon model behavior. While this must be in agreement with what the conceptual model predicts to indicate compliance, it is tested upon the implementation that results from the simulation model (Shannon, 1975).

Formal Techniques – finally, there are formal mathematical proofs of correctness that can be applied to the model, in order to evaluate whether the claims it makes are valid. Several of these are mentioned below. It should be noted that while these are a way of expressing the activity within a conceptual model, they typically prove to be too formal to capture conceptualizations – but are very appropriate for being able to capture logical and relations and transformations that take place within the model.

- Inductive Assertions – assesses model correctness in a manner similar to the proof of correctness method (below), but is deemed to fall short due to its reliance on induction. It evaluates three different categories of information concerning the conceptual model, thereby assuming that a conceptual model suitable for this type

of formal evaluation must be able to represent information required for these three evaluations (although they are related to each other, providing for the overall inductive assertion technique). They are:

- Input to output transformation validation for all variables within the conceptual model.
- An assertion statement for each of these transformations that can be evaluated – does the path of the variable as it progresses through the steps required to exhibit the transformation exist, and is it followed in a valid fashion.
- Overall verification of the conceptual model is established by evaluating all of the assertion statements for all variables and variable transformations within the conceptual model.

If these three things can be shown, and if the model itself can be shown to terminate, and these can be shown by induction, then the model is said to have been proven by inductive assertion (Manna, et al., 1973; Reynolds & Yeh, 1976).

- Lambda Calculus – is applied to model V&V by treating the overall conceptual model as a string. Lambda calculus techniques are then followed in order to exhibit all of the possible transformations (in order) that the string (conceptual model) goes through in order to be transformed from the beginning state to the ending state. As a formal method to express and evaluate, formally, transformations within the model this is an accepted technique (Barendregt, 1981).

- Predicate Calculus – is a method for representing and evaluating predicates. Predicates, when consider for formal techniques such as this, are relations between states. Predicate calculus formally captures those states, including the transformation, and the correctness with which the conceptual model represents these is evaluated (Backhouse, 1986).
- Proof of correctness – is an ideal technique that is not available today (MSCO, 2006) but is recognized that if and when it becomes available, it will revolutionize the approach of V&V. The technique, as proscribed, will describe all aspects of the conceptual model formally, and provide techniques in which to evaluate each of those aspects formally (Schach, 1996).

This contribution of conceptual modeling (to the process of V&V) is a different use than using the conceptual model as a motivator and guide for development. Using the conceptual model for V&V is to put the conceptualization of the referent system to the test and it is also to make clear the intentioned workings of the simulation system. As such, it may be used at different times during the lifecycle of the simulation system the conceptual model is related to, but identified applications are different from the first view (different from Pace (2000) and Robinson (2008), and also different application from MSCO, (2006) as to how the conceptual model assist with V&V). The different techniques enumerated above give an overview of the information that a conceptual model is expected to convey concerning the system it represents, in order to fulfill what V&V expects from the conceptual model – but when those techniques are applied is at different stages throughout the lifecycle (and not just for motivating the construction of the simulator).This second view, of treating the model separate from considerations of

how it relates to the simulation is also seen in Balci (1998), where Balci speaks of the validation of the model apart from the partnered simulation system (if there is one). Note that this differs from how Pace describes the role of the conceptual model in V&V. In that (which is part of the first view, presented earlier) role, the conceptual model is useful for understanding what the simulation should be doing. The goal, given in the preceding subsection, concerning what Tolk, et al. (2010) have to say concerning what a conceptual model has to present concerning the system it represents, certainly supports both of these roles (the first and the second). In this second role, identified chiefly by the contributions of Balci, Sargent and others, in the area of V&V, the role of the conceptual model is useful as an artifact for examining the functionality of the simulation, but it is equally useful for examining the modeler's rational understanding of the problem and the systems involved.

In order to rationalize together the multiple uses for models presented here, with the multiple stages within the process of progressing from model to simulation as forwarded by Yilmaz (2004) and Brade (2004) and others, it is useful to look at what is implied by the many uses for a conceptual model that the V&V community implies (for an example). The following table illustrates the techniques of evaluation, the type of technique (informal, static, dynamic, formal) and then evaluates which part of the Yilmaz taxonomy of models and which part of the Brade taxonomy of models (covered in greater detail in the following subsection) would be most appropriate to answer the questions asked by the technique.

Table 1

Modeling questions and the continuum of models

V&V Technique	Type of Technique	Yilmaz Taxonomy	Brade Taxonomy
Audit	Informal	Conceptual model	Formal model
Face Validation	Informal	Simulation model	Implementable model
Review	Informal	Both	All
Cause-Effect Graphing	Static	Conceptual model	Conceptual model
Control Analysis	Static	Simulation model	Formal model
Semantic Analysis	Static	Simulation model	Implementable model
Symbolic Evaluation	Static	Conceptual model	Formal model
Syntactic Analysis	Static	Both	Formal, Implementable
Assertion Checking	Dynamic	Simulation model	Implementable model
Bottom-Up Testing	Dynamic	Both	All
Sensitivity Analysis	Dynamic	Simulation model	Implementable model
Inductive Assertion	Formal	Simulation model	Formal model
Lambda Calculus	Formal	Both	Formal model
Predicate Calculus	Formal	Conceptual model	Formal model
Proof of Correctness	Formal	Unknown	Unknown

What Table 1 illustrates is that, at least for the V&V community, there are a wide variety of different questions that a conceptual model can be expected to answer. Yilmaz and Brade have been shown to present a taxonomical structure of different model types, having different focuses and perspectives on what the system they are representing. The different questions that the V&V techniques pose are each looking for a different focus of information, as represented by the taxonomies of Yilmaz (2004) and Brade (2004). However, as has been shown, although these taxonomies are referred to, it is not implied that there are only two (Yilmaz) or three (Brade) perspective points for developing conceptual models, but a whole continuum of possible perspective points. That the various techniques in (Table 1

Modeling questions and the continuum of models) frequently have different combinations of views from Yilmaz and Brade show that such a finite taxonomy is insufficient but that the continuum is more likely.

Conceptual Model as Description. Taking these two views into account, the first that the conceptual model is useful insofar that it motivates and can be accepted as a stand-in for the simulation system, and the second that the conceptual model has useful application as its own artifact, and should be treated separately from the simulation, it can still be investigated what the literature has to say about what the conceptual model should include in its description of a referent. This view is still in alignment by what has been shown so far from Tolk, et al. (2010). It has been stated Mylopoulos (1992) that conceptual modeling is "the activity of describing some aspects of the physical and social world around us for the purposes of understanding and communication." In addition, it is stated in Robinson (2008) that conceptual modeling, as an activity, "is the process of abstracting a model from a real or proposed system." Robinson (2008) goes on to state

that it is certainly the most important part of the modeling and simulation activity. In addressing information that is useful to both of the above views, that the description of the motivation for the design (giving the perspective that the model is derived from, as well as the understood assumptions and constraints on the design), and also serving as a useful artifact for reasons aside from simulation design, there are some additional requirements on conceptual models introduced by West. In West (1996) he states that to be useful to the enterprise employing it, a conceptual model should exhibit two characteristics. These are (1) that the conceptual model is robust, and (2) that it is implementation neutral. In calling for the robustness, West is referring to the desired quality of a conceptual model that it be complete and exhaustive in its depiction of the meaning of the components of the system it represents, which will be addressed below. In calling for the conceptual model to be implementation neutral, West (1996) shows that a model may (and likely will) be used for any number of evaluations for different resulting systems or uses. Because of this the conceptual model, and the method followed for capturing it, should not limit future use by expressing itself in the terminology of one particular implementation style or another. Robinson backs up this second claim of West, stating Robinson (2006) that the conceptual model will not be in the language of the implementation; "a non-software specific description of the simulation model that is to be developed." From Pace (2000), a different list of criteria for what constitutes an acceptable conceptual model is enumerated as follows:

- Completeness – all elements of the problem domain that the resulting simulation will address are included – objects, processes, and elements of control and structure.

- Consistency – all elements are in alignment with each other, with regards to structure, scope and resolution
- Coherence – all elements have function and potential
- Correctness – all elements of the model are valid representations of the referent

Definitions from the literature as to what the elements are will be described in the following paragraphs. Before moving to that discussion, however, the requirement for identifying those elements is clearly seen in Pace (2000) by following Pace's four steps towards development of a conceptual model:

- Collect authoritative context information
- Identify objects and processes
- Develop simulation environment
- Address relations among elements

To attain a robust (robust from West, but defined in detail by Pace) and meaningful description of system the model is representing, we have to identify context (constraints, assumptions, perspective), identify the system elements (objects and processes), identify the environment in which those will exist with each other, and identify the relations amongst the elements. In opposition to the view by Pace (2000), however, that a model should include all of these things, Brade (2004) has identified that several of these elements properly belong in different models, of which he describes three – the conceptual model, the formal model, and the implementable model. Brade shows how each model is responsible for different representation of knowledge, and different types of knowledge, about the referent, so that comparison of each to external

specification and evaluation will result in, for instance, different credibility of each. If the relation between the various models – conceptual, formal, and implementation – is strong, then that credibility should be complimentary, but the demonstrated implication is that in order to achieve either the criteria of Pace, or the robustness of West, it may be possible that multiple models are required. Exactly that view is found in the literature of the Unified Modeling Language, in Booch, et al. (1999), where the four basic principles of modeling are identified as:

1. The choice of models by the modeler will have a profound influence on how the problem is attacked and how a solution is shaped Every model may be expressed at different levels of precision, or resolution
2. The best models are connected to the real system.
3. No single model is sufficient.

These four principles agree with the other findings from the literature, concerning purpose, the existence of multiple models for every referent, and that the model should be descriptive of the referent.

The literature also suggests what the conceptual model should entail; what it should include. The literature reveals a more detailed requirement of what West calls for in saying that a conceptual model should be robust. From Robinson (2008) can be seen the first requirement – that a conceptual model should include the "objectives, inputs, outputs, content, assumptions and simplifications." This is corroborative with what is seen in Pace (2000), that the conceptual model should provide subject matter of what the model is doing, and also the motivation (including the simplifications that derive from perspective and abstraction, as well as limits from constraints and assumptions). Work

showing how the combination of constraints and assumptions is a key in both limiting and defining the final simulation product deriving from the model is found in (King & Turnitsa, 2008), and also in Tolk, et al. (2009). As was pointed earlier (p. 11), the application of Definitions 1 and 2 to a referent, in order to derive first a model of the referent, and then implement a simulation of that referent, lead to a number of purposeful decisions to be made about the modeler's (and simulation professional's) interpretation of the referent. In following the view of Balci (1998) and Sargent (2005), that a model can be considered aside from as a motivator for a simulation, this still remains clear. These decisions include interpreting the use to which the model, will be applied, as well as the world view the modeler interprets that the referent within. These result in the modeling assumptions (about the world view) and constraints (how much of the model should be shown, in order to provide an answer for the identified use). Assumptions and constraints affecting the resulting simulation that is implemented from the model include which implementation paradigm, and the interpretive questions about the model that the paradigm asks, as well as how they are answered. That a wide variety of different types of assumptions exist, and each affects the perspective of the modeling question, and the resulting developed conceptual model, has been presented in King & Turnitsa (2008), and the effects that both assumptions and constraints have on developed models has been shown within (King, 2009).

Additional prominent views from the literature include (1) the experimental frame that is a key to the DEVS formal system of Discrete Event Specification following Zeigler, et al. (2000), and (2) the idea of the conceptual model and simulation model being separate, according to Yilmaz (2004).

First, the experimental frame, from DEVS, gives the general concept and context in which the discrete event system that the DEVS formal tuple describes. The experimental frame describes the world that the discrete event system is operating in, and gives the context for the system that the discrete event system is simulating. From this description, it can be seen that the idea of the experimental frame as conceptual model falls into the idea of the conceptual model as existing to motivate the development of the simulation. It also serves to capture some of the assumption and constraint information that is hinted at in both King and Turnitsa (2008) and Tolk, et al. (2009).

Second is the view by Yilmaz (2004), extending the idea of the conceptual frame, that an understanding of the referent system is best handled as a conceptual model, but then when it is desirable to move from that model to an implementation, a second model is useful that can following the modeling definition, 'the process of abstracting, theorizing, and capturing the resulting concepts and relations in a conceptual model'", and express an abstraction of the conceptual model in terms suitable for implementation. This second model is called the simulation model (Yilmaz 2004).

Models of Systems. In order to examine what the literature states that a system model should be comprised of; one can begin with Robinson's (2008) statement concerning the elements of a model, and determine what a model of a system should contain. The enumerated elements of a model that are found in Robinson, again, are content, input, output, objectives, assumptions and simplifications. First, by comparing this with the stated definition from (p. 11), we see that Robinson includes assumptions and simplification (abstraction, by the definition). Robinson also includes content, input and output – which together are the theorizing from the definition – that is, they are the

conceptualization about the referent. Finally, that Robinson mentions objectives shows that, as per the definition given earlier (p. 12), the model is purposeful. In this section, we are concerned about the elements that make up the model, that is, what are the parts of the conceptualization that can be identified. For this, from Robinson, we are left with input, output and content. As the first two of these are concerned, Input is different than Output, there must be some way of describing how the model would describe that process of a system transforming the input into the output. This capability for transformation is what the content of the model describes. It is a conceptualization of the content of the system – a description of what the system does. If a model is to be useful for engineering purposes – evaluation, etc – then it must be sufficiently expressive in order to be evaluated. As Law and Kelton (2000) have shown, a model is consulted when it is not possible to consult the actual system it represents. Combining these two ideals – the usefulness of a model, and the fact that the model abstracts away whatever it is about the system that makes it unavailable for experimentation – we can see that a model of a system should be expressive enough to be representative of the referent system, yet be abstract enough to be useful in situations where the actual system is not available. Representing the contents of a system in an abstract way is the way to satisfy these two criteria.

Contents of a System. The goal established for a conceptual model to exhibit what a system's contents are has been shown by the expectations of the V&V community for what a conceptual model has in order to answer specific question. However, it still remains to be shown from the literature what those content elements are. This can be seen from the literature for several domains. From modeling and simulation, it can be

seen in conceptual models. From systems engineering, it can be seen in techniques such as SysML and Object Process Methodology (OPM). From software engineering, it can be seen from UML. And from knowledge engineering, it can be seen from ontological modeling. These are all presented in the following section, from the attendant literature.

From a perspective intended to show the functional decomposition of a system and represent it in a model, as shown in Dori (2002), it is stated that systems can be modeled by objects and processes. Objects are the elements of the system that will remain the same unless operated on by some other part of the system, by his definitions. Processes are the elements of the system that perform those operations. Dori indicates that his method expands on most system modeling techniques, which rely on the capture of objects and the traditional system science technique of representing processes as state changes between states of objects, to show that processes are more complex and deserve additional consideration. This has additionally been shown to be the case within Tolk, et al. (2009), where alignment and composability of models is shown to require consideration of objects, processes, and constraints. From Schlezinger, et al. (2006) it is seen that both elements are required – objects grant the model understanding of the structure of the system, and processes grant the model understanding of the dynamic behavior of the system. This is shown in Schlezinger, et al. (2006) while discussing the presence of both in the conceptual modeling technique 'Object-Process Methodology', which is the topic of study in Chapter 5 of this dissertation. The statement made is as follows:

The two equally important class types in OPM, objects and processes, differ in the values of their perseverance attribute: the perseverance value of object classes is static,

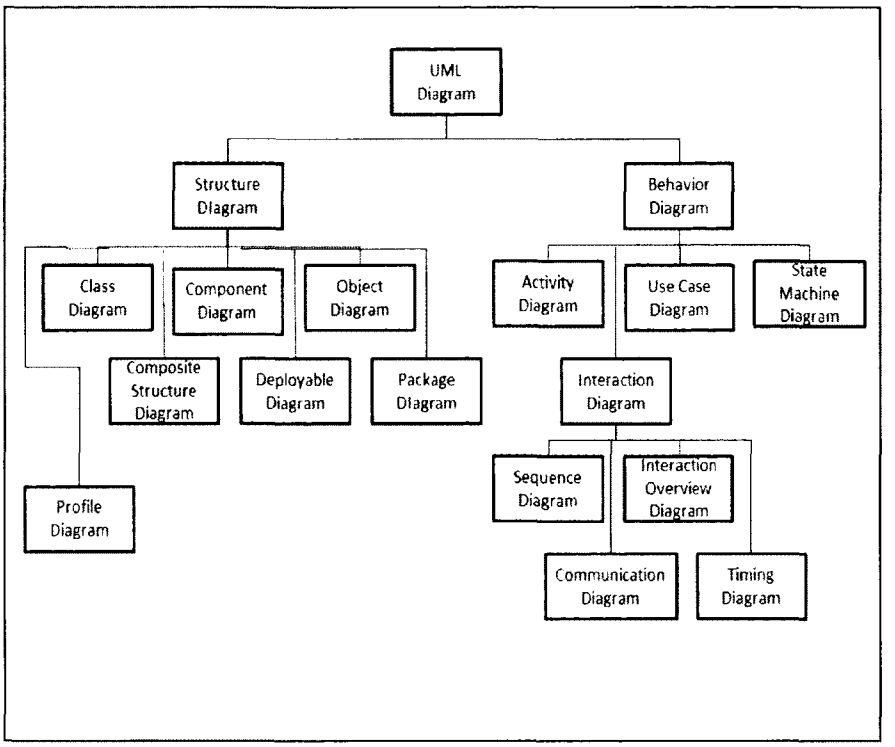

Figure 7. **UML Suite of Modeling (Diagramming) techniques.**

while the perseverance value of process classes is dynamic. OPM's combination of objects and processes in the same single diagram type is intended to clarify the two most important aspects that any system features: structure and behavior (Schlezinger, et al, 2006).

Bringing this into alignment with Robinson (2008), the behavior of the system is described by the model as its contents, and it relates how the model transforms input into output. If the model is considered from the resolution of considering that behavior as a single transformation (input into output) the model is a function, accepting input to the model as its domain, and producing output from the model as its range. Internal to the

system, unless it is trivial in composition, there will be any number of different sub-stages of this transformation, where parts of the model are being transformed to be used by other parts of the model, as per Dori (2002). The Unified Modeling Language, or UML (OMG, 2002) assembles a number of different modeling techniques to describe a

system (*Figure 7.* UML Suite of Modeling (Diagramming) techniques.). These techniques are split into two categories, agreeing with Shlezinger, et al. (2006), in that they are described as the **structural diagrams** (giving the structure of the system) and **behavioral diagrams** (giving information about the functioning of the internal processes that make up the overall system transformation). Considering the Systems Modeling

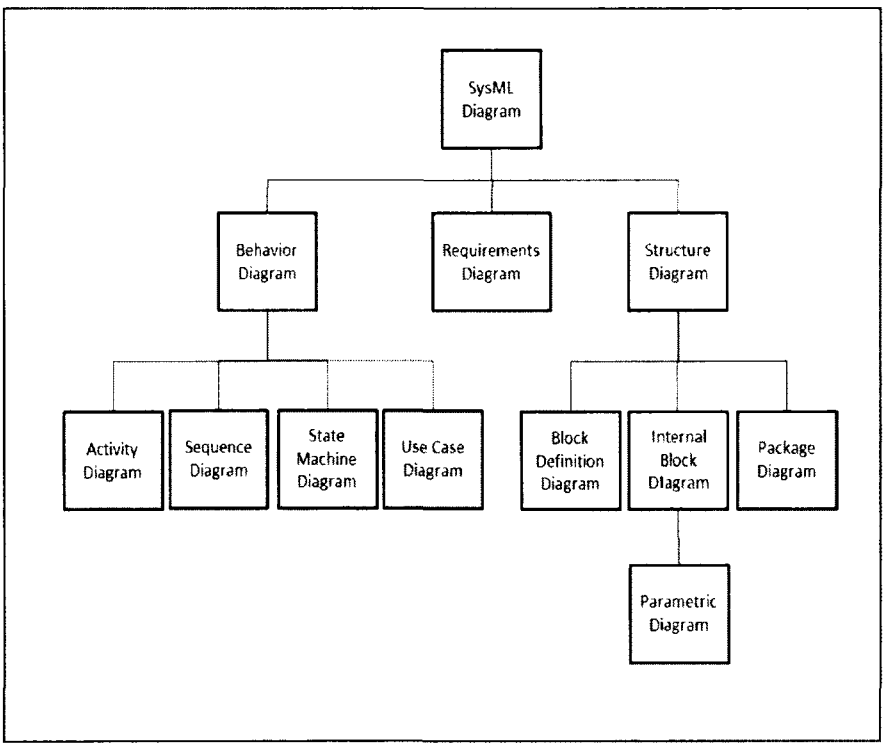

Figure 8. **SysML Suite of Modeling (Diagramming) Techniques**

Language, or SysML,it is seen (OMG 2010) that these two categories are also present, but SysML shown in Figure 8, also includes a new modeling technique, or diagram as they are called in SysML and UML, called the "requirements diagram", which is additional to, and separate from, the UML categories of behavioral and structural diagrams.

In addition to what UML (from software engineering) and SysML (from systems engineering) might have to state concerning the elements of a conceptual model, from the discipline of M&S, we can also see that for the purpose of building a model based system (where the resulting implementation system – a simulator – has as its goal the enabling of a conceptual model) the model itself takes on a special value. With software engineering, or systems engineering, there is a real world referent that the conceptual model refers to. Once there is an implementation, such as a software system, or an engineering system, that is developed, it can always be evaluated against that real world referent, and its observed behaviors and composition. This is possible once the system is implemented, and also during the implementation itself. However, when the system being developed is itself intended to implement a model – that systems referent is the model. The model itself now takes on special value as being the referent for the resulting system, and other models of the model will serve towards development of that system, and so on, but the original model itself that the system is implementing (as a simulator implements a conceptual model) is the original referent. In this case, it can be shown that model accuracy and proper encapsulation of knowledge within the referent-model is crucial. This special relation is presented within Tolk, et al. (2011).

In understanding what software engineering (UML) and systems engineering (SysML, OPM) declare that a model should consist of, we have seen so far the specific mention of objects and processes, a view from the literature that is corroborated by Tolk, et al. (2009). As has been shown, it is expected by the users of models (for instance, the V&V community) that the model will serve as some knowledge representation of the system it represents. This is also in alignment with the idea of using modeling and

simulation to represent the semiotic triangle (Ogden and Richards 1923). There are other modeling communities, however, that are also concerned with knowledge representation. One of those worth considering is the knowledge engineering community. This community, which is concerned with knowledge representation and knowledge modeling, is defined (in terms of modeling) as, "Knowledge engineering is the application of logic and ontology to the task of building computable models of some domain for some purpose (Sowa, 2000, p 2)".

As a pursuit, there are several characteristics of knowledge engineering, especially with relations to modeling, which are worth considering. They are listed here, in order to provide a background for seeing what the composition of knowledge engineering models consist of. These are taken from Davis, et al. (1993).

1. A knowledge representation is a surrogate. What is meant by this characteristic of knowledge engineering is that a knowledge representation is some symbolic representation for a referent. This is clearly in line with Definition 1, establishing that a knowledge representation is a model.
2. A knowledge representation is a set of ontological commitments. An ontological commitment is to state that something is a true representation of something else (in the epistemological sense). As this dissertation does not go into epistemology, or model theory (the means of evaluating the ontological commitments of formal models), it is not covered here, but it does show that a knowledge representation is a model that is intended to be a true representation of its referent.
3. A knowledge representation is a fragmentary theory of intelligent reasoning. Again, intelligent reasoning is not a topic explored in the dissertation, but for the

purposes here it is enough to state that a knowledge representation can be relied on to render answers about the referent it describes.

4. A knowledge representation is a medium for efficient computation. This is to suggest, in the taxonomical structures of Yilmaz (2004) and Brade (2004), that a knowledge representation is a model that is ordered and logical enough to assist with formulation of the simulation model (after Yilmaz) or the implementation model (after Brade).
5. A knowledge representation is a medium of human expression. As all models that can be shared are – as they serve as the means for expressing a conceptualization (Ogden and Richards, 1923).

As the foundational method for describing knowledge, the philosophical pursuit of ontological representation will be considered to see if knowledge representation models contain the same contents as the other modeling domains considered (software engineering, system engineering and modeling and simulation). In the literature of philosophy, and more recently in the domains of systems engineering based on knowledge representation, and the same for information systems, there are explanations of this sort. From the ontology literature it is clear that the presence of objects and processes, and their relationship, is such a clear part of the fabric of understanding, that it engenders two different viewpoints on how a system is understood. The first is the object-centric view, referred to in Grenon and Smith (2004) as the 3-D, or three dimensional, viewpoint. It is so named because it assumes that objects are (in a Newtonian sense) in existence and inviolate in their definition, unless acted upon by some outside force – a process. A modern description of this metaphysical commitment

is presented in Lowe (1998). The second view on reality is the process-centric view, referred to in Grenon and Smith (2004) as a fourth dimensional view. This second metaphysical view is much more in alignment with some of the ideas of modern physics, in that what we perceive of as objects are just signifiers of ongoing processes – everything is always in some state of flux. This second view is presented in Sider (2001). A source that unifies these two viewpoints is Grenon and Smith (2004), and a conclusion made in that source is that regardless of the perspective (either object-centric or process-centric) the reality is defined by objects and processes and their relations with each other.

Association by Relations. Not mentioned explicitly so far, but always there in all of the literature sources mentioned so far, is the implication that these content defining elements – the objects and processes – can be associated with each other. This is present, certainly, in data modeling with its view (Chen 1976) of the relational model. It is also there, in most graphical modeling techniques as the way of connecting the elements of the model – some boxes are related to others. Considering the diagrams from either UML or SysML this can readily be seen, as well as the conceptual models described by Robinson (2008) where there is definite and committed association between objects such as input and output and internal model components, and the transformational processes that affect them. Without the ability to indicate association of the object and process elements by these relations, the conceptualization of the system would not be possible. Taken together, then, the elements that Robinson (2008) claims make up a conceptual model are what (Dori 2002) claims are part of a systems model, and what Grenon and Smith (2004) claim are part of an ontological model – objects, processes and the relations between them. This is seen in (**Error! Reference source not found.**) and illustrates that

for the domains of systems modeling (both for systems engineering and software engineering), modeling and simulation (conceptual modeling) and knowledge representation (ontological modeling) that the components that models are comprised of are the same. Those components are objects, processes and relations.

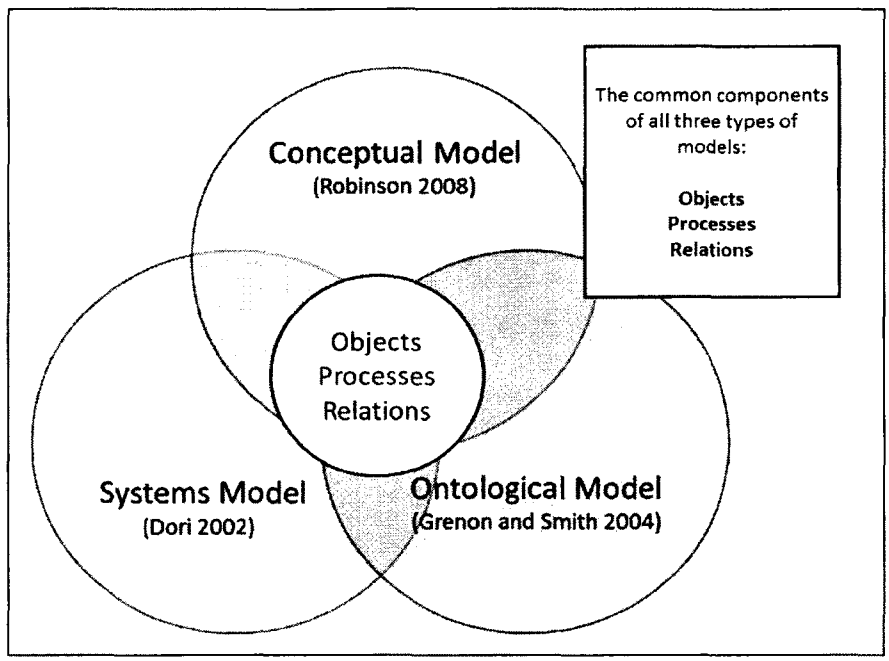

Figure 9. **Common elements to modeling approaches.**

Conceptualizing the System. The tradition within computer science for depicting the components of a system (simulation or otherwise) begins in 1947 with Goldstine and Von Neumann, as flow charts (Von Neumann 1963). At the time, the goal was to represent, conceptually, what the automata were doing. The charts include boxes representing the computational events at a conceptual time during the execution of some set of instructions. The boxes are connected, showing the flow of the instructions, showing the procession of one computational event after another, and when a decision is made, it is

represented as a diamond that can branch in the relations between the states. This is a very object-centric view, as all of the processes are transitions from one computational event to another. It is also very implementation specific, tending towards what Yilmaz (2004) would identify as a "simulation model", or what Brade (2004) would identify as an "implementation model". The state of modeling progressed from that early point, to include the idea of a finite state machine (FSM). In an FSM, the various states of the system are captured, and the relations between the states are transitions. It also allows for branching, as a flow chart. A finite state machine is defined as:

A model of computation consisting of a set of states, a start state, an input alphabet, and a transition function that maps input symbols and current states to a next state. Computation begins in the start state with an input string. It changes to new states depending on the transition function. There are many variants, for instance, machines having actions (outputs) associated with transitions (Mealy machine) or states (Moore machine), multiple start states, transitions conditioned on no input symbol (a null) or more than one transition for a given symbol and state (nondeterministic finite state machine), one or more states designated as accepting states (recognizer), etc. (Black, 2008)

There are two types of finite state machine diagramming techniques, showing historically for computer systems modeling, the general dichotomy seen in the ontology literature as distinguishing a basic understanding of a system from an object-centric view or a process-centric view based on their designed capability to produce output. The two FSM types that exhibit output are the Moore type and the Mealy type (Anderson, 2006). The first (Moore, 1956) is considered to be more object-centric (*Figure 10.* Moore

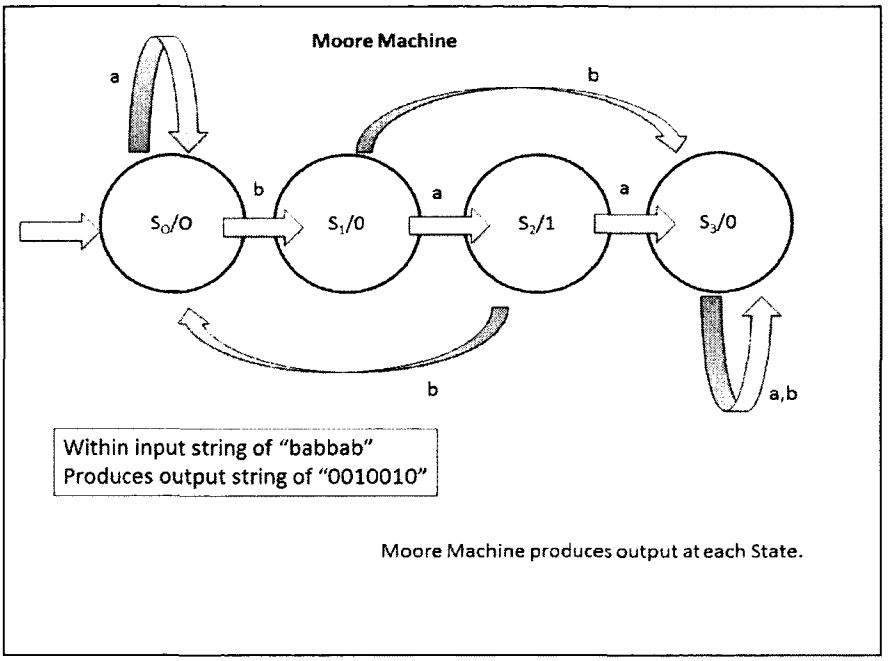

Figure 10. **Moore Machine**

Machine). The output in a Moore machine occurs at each state, or object, including the initial state. So, even prior to any transitions occurring, a Moore machine produces output.

In the second (Mealy, 1955) output producing FSM type, each transition produces output as well as determining what the next state of the machine will be, and the output of the transition is recorded, and then transition to the next state occurs (*Figure 11.* Mealy Machine). In agreement with the much later definition of an object, from Dori (2002), an FSM considers that an object at rest within a state will remain as is, until it is affected by transition to some new state.

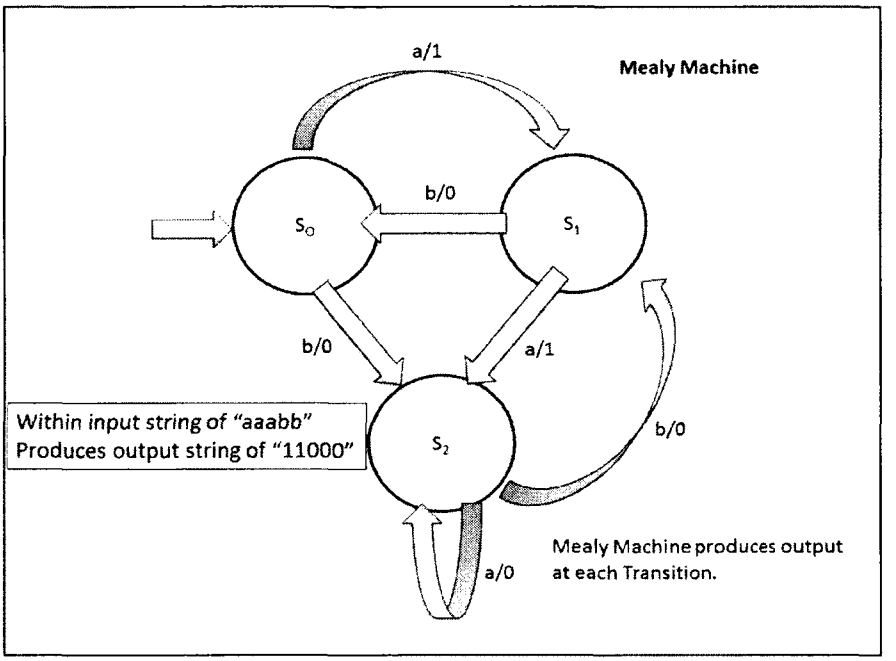

Figure 11. **Mealy Machine**

To consider further improvement on this method of representation, one can look to (Sowa, 2000), which shows the next progression in this capability to diagram, or model, a system. Sowa shows how additional understanding in this progression can be seen in the method of representing functions, states, transitions and relations among them – the Petri network diagram, or Petri net (Petri, 1966). In the Petri net, the identity of each state is captured. It is conceptually possible for objects to move among those states, and retain an identity separate from the states themselves. Also, of importance in this overall modeling evolution is the fact that with Petri nets each transition is also given identity. Sowa exhibits that more is needed than just identifying states of objects or of the system, and then representing processes as the connective transitions (as is the focus of FSM techniques). Consider in (*Figure 12.* Three techniques for modeling a dynamic

system) that the history of representation of states and state transitions is shown by Sowa (2000). First, there were flow charts showing how the boxes represent computational events, and the diamond represents a decision point, where the next computational event is selected by some criteria.

The representation improved, in the second diagram in (*Figure 12.* Three techniques for modeling a dynamic system), by using FSMs to represent the states involved. The state of the system becomes the focus. This is further improved by the third diagram, which is of a Petri net, showing both the states, as well as the transitions between them (the computational events). In Sowa (2000), the author calls for a complete view of a system (simulation or otherwise) by modeling, equally, the objects and processes. Again, the relations that provide association among these elements is understood, and even present in the examples and diagrams from Sowa (2000), but they are not explicitly called for. What is clear, again from (*Figure 12.* Three techniques for modeling a dynamic system) is that there are relations between each state (the system at rest, represented by p, q, r, s, t) and the processes, or computational events (that represent the transitions between those states as a,b,c,d,e).

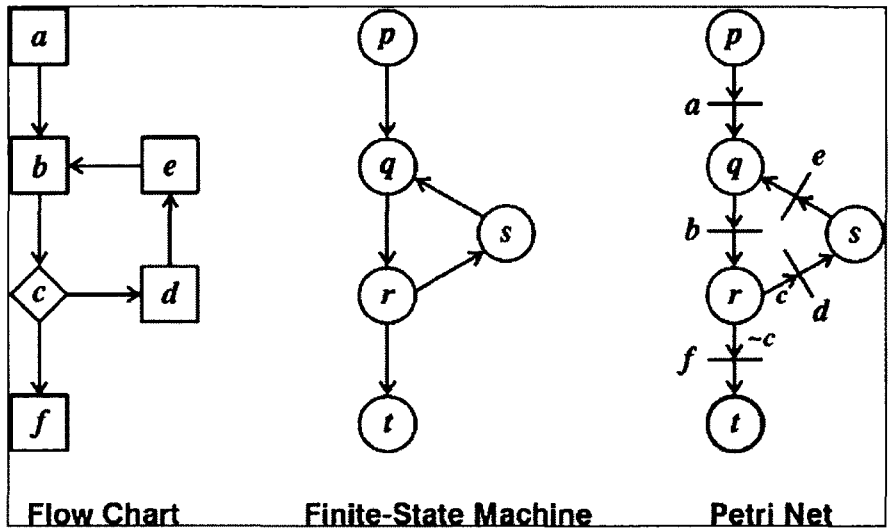

Figure 12. **Three techniques for modeling a dynamic system**

Models as Theory. It has been shown (Sowa, 2000) that there are many possible ways, given a model, to theorize about the system (referent) that the particular model represents. While each one of these theories, with relationship to the model it is based on, may be evaluated as either true or not, the theories themselves are from different perspectives (as are the models), which then leads to possibly contradicting claims concerning the referent. Sowa refers to this as knowledge soup. He goes on to state (Sowa, 2000):

The problems of knowledge soup result from the difficulty of matching abstract theories to the physical world. The techniques of fuzziness, probability, defaults, revisions, and relevance are different ways of measuring, evaluating, or accommodating the inevitable mismatch. Each technique is a metalevel approach to the task of finding or

constructing a theory and determining how well it approximates reality. To bridge the gap between theories and the world, see that (*Figure 13* Relationship of Model to Referent (world) and Theory) shows models as Janus-like structures, with an engineering side facing the world and an abstract side facing the theories. (p. 383)

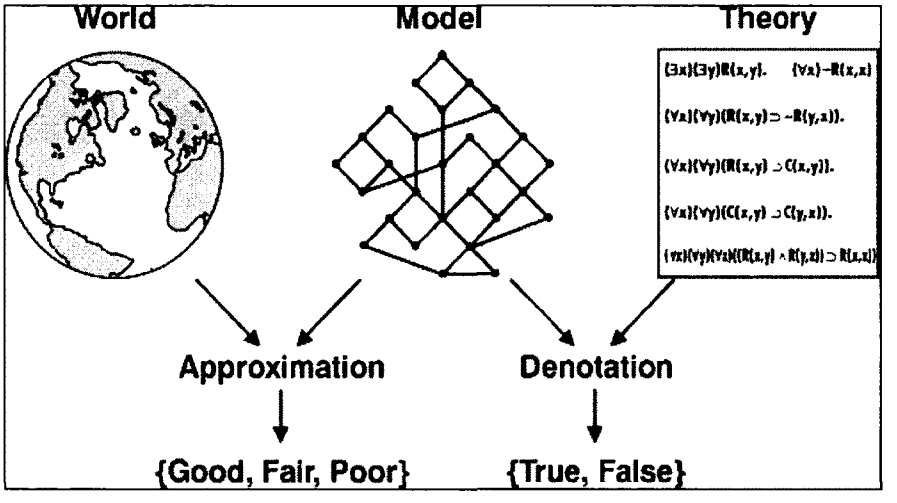

Figure 13 **Relationship of Model to Referent (world) and Theory**

As has been shown here, there are modeling techniques (conceptual modeling, systems modeling, ontological modeling) that exist to support a variety of different domains (modeling and simulation, computer science, systems engineering, knowledge engineering). It has been shown, for the M&S community, that as all of these domains involve modeling, as an effort to interpret the world, so that theories (or systems) can be developed from those models, that each of these become the purview of modeling and simulation – at least insofar as modeling is used within them. This is shown by the M&S Body of Knowledge (BoK) work being developed and maintained by Tuncer Ören and

Bill Waite (for the index of the body of knowledge, which is heavily cross-linked, see the main site maintained at http://www.site.uottawa.ca/~oren/MSBOK/MSBOK-index.htm). As an example of the breadth of what has been identified, there are over 1,100 terms in the BoK relating to different types of modeling, all seen as properly part of the field of practice. The developed areas here, of computer science, systems engineering, knowledge engineering, and conceptual modeling sit firmly in the acknowledged areas of practice, and the areas of theory contribution, as per the BoK.

Summary of Literature Review. From the literature it has been seen that while modeling and simulation is often described together as a single pursuit, each of the component terms results in a distinct definition. For modeling, from the literature we distill the following:

Definition 1: Modeling is the purposeful process of abstracting and theorizing about a system, and capturing the resulting concepts and relations in a conceptual model.

While it may be possible that a conceptual model is used to inspire the implementation of a simulation, the conceptual model has value and use beyond that role. In considering the simulation separate from the model, from the literature we distill this:

Definition 2: Simulation is the process of specifying, implementing and executing a model.

The literature has also been show to identify that the components of a conceptual model can be seen as being of three types. These first two types identified are the object and the process, and these are shown to be relying on a third type for association, the relation. These have been seen to be present, in a number of different modeling communities, which themselves have all been shown to be part of the body of knowledge

of modeling and simulation. Those areas include computer science, systems engineering, knowledge engineering, and modeling and simulation itself.

From this, identified are what a model is, what a simulation is, what a conceptual model is, and what it is composed of. It is identified that there may be any number of models from a referent; that there may be any number of simulations from a model. In this, now defined, area of the science, it will be useful to consider a number of conceptual modeling techniques, and as the dissertation is concerned with the study of processes within conceptual modeling techniques, a brief survey identifying what it is about the described system each of these techniques can exhibit as being transformed within the model.

2.2 SURVEY OF CONCEPTUAL MODELING TECHNIQUES

With the definitions from the literature established for this dissertation, examining a sample of conceptual modeling techniques (CMTs) is warranted. This is for the dual purposes of (1) establishing that the decided upon definitions are applicable to the subject area of the dissertation, and (2) exploring the state of the art to find a gap in current knowledge or technique that the dissertation can address. The overall interest motivating this dissertation (from chapter 1) is an exploration of processes within modeling techniques, so particular attention in these descriptions is paid to how change is captured within the CMT. A subset of these techniques is explored further in chapter 4 of the dissertation, and they are given a more rigorous treatment than the overview presented here. The method for formally identifying the components, and role of those components, from within a technique that is presented in chapter 4 of the dissertation is the method enabling that rigorous treatment. The following sections are meant as a

sample of some of the techniques available to the modeler, and the description here is only to answer the two purposes identified – comparing the techniques' adherence to Definition 1, and examining this representation from the body of knowledge to expose a gap in current state of the art.

Selection of Techniques. Doing a survey of techniques from the field is useful in examining the state of the art; however it becomes necessary to describe which techniques to choose, and why they are chosen. In the preceding portion of this chapter, in examining what the literature has to say concerning definitional elements of what models are, what simulations are, what conceptual models are, a number of different modeling techniques were mentioned. From among these, the techniques of this examination will be chosen, however those that are among the more popular (in terms of frequency of appearance in the literature) will be given precedence. This process of consideration results in techniques that were selected by two criteria – those that were mentioned by sources considered in the literature review, and those that are currently widely used and popular techniques. In addition, there have been several techniques that were identified as possible candidates, by the criteria given here, but which were eventually rejected. A list of those rejected techniques, and the reason for excluding them from analysis, is in (appendix 1). Several individual techniques have been selected, because of their reference above, and also a number of techniques that together support each other and provide for the multiple model approach discussed earlier, as has been shown to be addressed in the literature by Pace (2000) and Booch, et al. (1999). The technique that combines several different techniques that is chosen is UML, because of its widespread use and popularity, and simultaneously because it has also been the source

of many references to the literature in the preceding sections. The list of those techniques selected, and the criteria for their selection, is presented here:

Table 2

Motivation for Conceptual Modeling Technique Selection

Conceptual Modeling Technique	Motivation for Selection
Activity diagram	Part of the UML/SysML suite of modeling techniques to describe behavior
Communications diagram	Part of the UML/SysML suite of modeling techniques to describe behavior
Discrete Event System Specification	Theory of modeling and simulation, provides the idea of the experimental frame
Flow chart	Identified as one of the early system modeling techniques from Computer Science
Petri nets	Identified as the technique that first grants identity to processes
Sequence diagram	Part of the UML/SysML suite of modeling techniques to describe behavior
State diagram	Identified as one of the early system modeling techniques; key to theoretical computer science view of finite state machines

Table 2 Continued	
Statechart diagram	Part of the UML/SysML suite of modeling techniques to describe behavior
Use-case diagram	Part of the UML/SysML suite of modeling techniques to describe behavior

As can be seen from the table, five techniques have been selected from the UML modeling suite of techniques. These are also found in SysML, and they include the activity diagram, the communications diagram, the sequence diagram, the statechart diagram, and the use-case diagram. The reason for their inclusion is because of the popularity of those two methods for modeling among the (UML) software engineering and (SysML) systems engineering communities. Both of those approaches, as per Booch, et al., rely on having a combination of different techniques to show a modeled system, and the five techniques chosen are part of the behavioral diagrams portion of those approaches (Booch, et al. 1999). As the dissertation topic is in the area of processes for conceptual models, the behavioral diagrams are appropriate. In addition, there are a number of historical diagramming techniques that are part of the argued case (presented earlier) in Sowa for stronger representation of processes in system descriptions (Sowa, 2000). These include the state diagram, the flow chart, and the Petri net diagramming technique. The final technique chosen is from the literature of modeling and simulation theory. It is the technique for modeling the transformations that take place in a discrete

system, from Zeigler, et al. (2000), known as the Discrete Event System Specification (DEVS).

Survey Characteristics. Hester and Tolk (2010) point out that there are several dimensions of taxonomical differences among conceptual models. It is pointed out that this list is not exhaustive. The dimensions given include the following:

- Static vs. Dynamic – The distinction given here is that a static model is a picture of a system at a single point of time, and that a dynamic model exhibits how change occurs within the system over time. This accords with the definition and findings from literature given earlier that a model's contents describe how input is transformed, over time, into output. As this dissertation is concerned with understanding processes of change within models, dynamic models are of interest here.
- Discrete vs. Continuous – The distinction given here is that a discrete model describes the changes that take place within it as being instantaneous and a continuous model describes changes that take place over time. Understanding this in light of Definition 1 requires that if the details of the change are part of the abstraction in the model, then this distinction may not be clear. Understanding this in light of Yilmaz (2004) would require, for his "simulation model" that the distinction be clear, but not necessarily so for his "conceptual model". This is also so in that the distinction need not be made in the "conceptual model" of Brade (2004), vs. his idea of either the "formal model" or the "implementation model", both of which would require understanding of the distinction. For the topic of this dissertation, as both of these types (discrete and continuous) represents modes of process, they are both of interest.

- Deterministic vs. Stochastic – The distinction given for this dimension are that with a deterministic model, the output that results from the model's contents will be the same every time, given the same input. In this regard, a deterministic model does approximate a mathematical function – a single answer from among the range, given a specific input from among the domain. The stochastic distinction is identified as applying to models that have internal processes that are based on some non-deterministic – stochastic – range of possible outcomes. The observations from Hester and Tolk (2010) that the choice to model in a deterministic manner is one of abstraction tends to support, however, that similar modeling techniques (again, at Brade's conceptual level, versus at the formal or implementation level) may support either of these distinctions. For the topic of this dissertation, both types are of interest; again as both types represent different modes of process.
- Sequential vs. Parallel – The distinction given for this dimension, by Hester and Tolk (2010), is that a sequential model is one that treats its internal processes within a single queue, yet a parallel model is one that allows for multiple concurrent processes. The identified results of making this distinction are that sequential models require problems of synchronization to be handled, where parallel models require that problems of coordination be handled. The difference between a sequential system and a parallel system again could be blurred by abstraction, but this is such a driving design element, that the ability to abstract away this (as with the other dimensions) would only be at Brade's (2004) conceptual level and not at the formal or implementation levels (Brade, 2004). This distinction is again one where both types

are of interest to the dissertation, because they each represent modes of process execution.

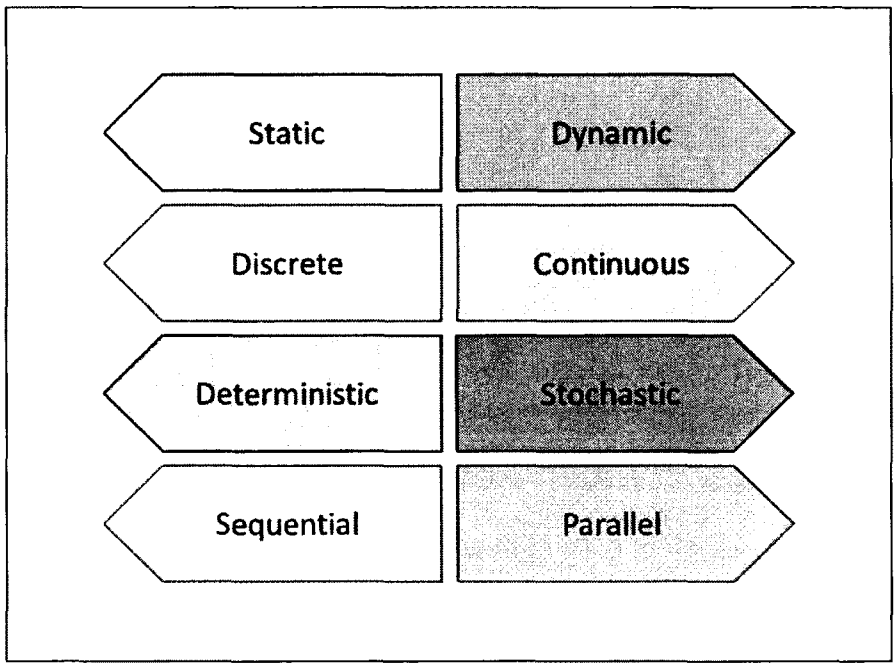

Figure 14. **The typological distinctions of models**

The typology given gives a number of distinctions (*Figure 14.* The typological distinctions of models), which may give a method for distinguishing models in terms of what they can represent, and specifically which dynamic elements than can represent. This collection of distinctions is in agreement with that shown in Law and Kelton (2000), and covers the standard methods for a practitioner of M&S to be able to draw distinctions between different modeling techniques. Other distinction lists are possible, such as the distinction between live, virtual and constructive, however this enumerated list (*Figure 14.* The typological distinctions of models) provides some information pertaining to the dynamic nature of the model involved. As such, it is the state of the art for typologically

identifying a modeling technique, according to the dynamic characteristics that the technique exhibit. It will be relied on here to present a high level identification of the dynamic characteristics of the modeling techniques surveyed. Because it is possible that abstraction can blur these distinctions at the conceptual level suggests that the same modeling technique could describe Brade's (2004) "conceptual model" of a system, without being affected by the distinctions, however if the model is to be at either of the "formal model" or "implementation model" type, then the distinction comes into sharper focus. Where applicable, these distinctions will be identified as part of the characteristics of each selected modeling technique. Identification of what the modeling technique is intended to show, as well as if it strays from being a technique that could support Definition 1.

Survey Results. The techniques chosen for this survey were selected because of the survey motivation reasons, given above (p. 60), and presented (concisely) in (Table 2 Motivation for Conceptual Modeling Technique Selection). Additional techniques were explored and surveyed during the preparation of the dissertation, by were not included in this survey for a variety of reasons. Those that were investigated, but are not part of this survey, are enumerated in Appendix 1.

Activity Diagram – Activity diagrams are part of the UML/SysML suite of modeling techniques that serve to model a system OMG (2002). An activity diagram represents the system's workflow (Booch, et al., 1999). System workflow is the sequence of occurrence of activities within the system. Activity diagrams exhibit a stepwise progression of activities within a system, but do not give details as to the specific temporal attributes (discrete or continuous, start time, stop time) of the activity

addressed; however relational information to other activities is the focus of this technique. Some of the details concerning the ordering of activities that can be captured are sequence, concurrency, and details about iteration of the particular activity. The activity diagram is presented as an example in Chapter 4 of this dissertation. For a simple example see (*Figure 15.* Elements of an Activity Diagram).

The activity diagram of UML 2 is to show the workflow relationship (sequential ordering) of the various parts of the system being modeled. It does this, in diagrammatic form, by showing the various functional elements of the system, and modeling the sequences that lead from one to the next, etc. It is very similar to the State diagram from UML 1, and replaces that earlier effort. In the SysML extension of UML 2 (OMG, 2010), the activity diagram also captures information on the passing of resources between the various activities.

The principle components of the activity diagram are a series of shapes, representing various elements of the system being modeled, and connectivity between those shapes by arrows, representing the flow of activity from one to another. Together these make up a diagram that is very similar to a state chart, hence the relation to the UML 1 State diagram. First the shapes are described here and then description of the arrows and their characteristics. It should be noted that the terms activity and Process are both used within the definitional materials of activity diagram components. Here an activity represents some unspecified grouping of individual actions, whereas when identified separately, it is possible to also represent a single, specific action (as in the case of signal generation).

Activities, the main component, are represented by rounded rectangles. These are intended to capture some operation that is part of the system, meaning that certain processes are occurring in order to accomplish a particular activity. Internally, there are no details concerning which system objects are involved in the activity, or what changes to those objects are taking place. It is possible to identify a sub-activity by including a "rake" symbol in the rounded rectangle. This represents an activity that the modeler deems to be of a lesser degree than the surrounding activities in the workflow being modeled. There are no specifics given (Booch, 1999) as to when an activity should be represented as a sub-activity.

Workflow terminators are represented by circles. There are two different types the first being a filled black circle represents the beginning of workflow, which is a connected sequence of activities. The second type is a filled black circle with a second circle around it, representing the end of a workflow.

Signals are represented by a rectangle with a triangular arrow side. An out-pointing (convex) triangle means a generating signal, which represents that a single action (rather than a whole activity) results in the production of some signal that must be completed before continuing. It is interesting that in knowledge system representation literature (Sowa, 1984) these signals are also called events. An in-pointing (concave) triangle means an accepting signal, which represents a single action that awaits some signal produced by another activity before it can begin. In both cases, signals are used to represent specific processes that must either accept input or produce output (or both) and that those (output and input) are necessary to the workflow.

Timing signals are a separate special case of a generating symbol. They are represented by an upright bowtie symbol. They model the requirement to have some period of time expire before an activity proceeds.

As mentioned, some of the sub-languages of the UML family may include the modeling of physical resources as part of the sequencing of workflow. This is particularly, currently, of SysML, but possibly also others, especially once the 2.4 specification is released (OMG, 2010). This modeling is called object Flow, as it is intended to show the movement of objects (resources) from activity to activity, following the workflow. Within an activity diagram, these are represented as squares connected to the edge of an activity block, at the point of a workflow arrow either entering or exiting the block. The block is labeled as to what it represents.

The arrows connecting the shape components of an activity diagram show the sequencing of the various activities and signals (processes). While they diagrammatically only indicate sequence, the specification allows for accompanying text to describe conditions that either are dependent on the sequence proceeding, or are required in order for the sequence to proceed. There is no distinction between these two types of notes.

Arrows can pass through a thick line drawn perpendicularly to the arrows. This indicates that a concurrency of different activities is about to take place, and usually the arrow will split (at the thick line) into several arrows each pointing to different activity blocks.

Arrows can also pass through diamonds, which represent decision points. Again, text can be employed to describe the decision being made, and the criteria being evaluated.

The activity diagram modeling technique represents information about a dynamic

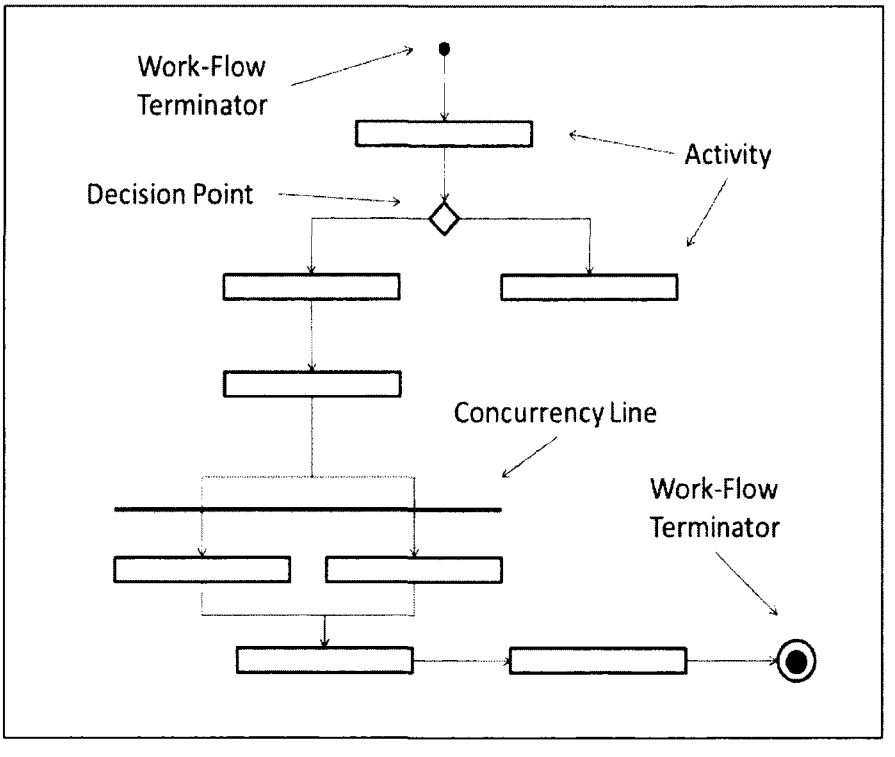

Figure 15. **Elements of an Activity Diagram**

system, it does not distinguish between continuous or discrete, the sequencing of activity is deterministic, and it can capture sequential as well as parallel activities. As can be seen in diagram (*Figure 15.* Elements of an Activity Diagram) there are some specific

graphical symbols for the components of the activity diagram. The work-flow terminator indicates either the beginning or end of the overall work-flow (the collection of all activity shown within the diagram). There are boxes representing activities – in an actual example (see Chapter 4) these would contain details describing the specific activity. A decision point is marked, if there needs to be choice between the next steps in possible activities. When a concurrency line is encountered, it indicates that the following activities occur concurrently.

Communications Diagram – Communication diagrams are part of the UML/SysML suite of modeling techniques (OMG, 2002). The communication diagram represents information about a system, showing which elements are communicating with which other elements, in order to exchange information or control messages (Booch, et al., 1999). The temporal sequencing of those interactive communications is also shown within the diagramming technique, but that information is secondary to the identity of element that is performing the communication. The information shown in a communications diagram is very similar to a sequence diagram, but where the primary focus with a communications diagram is on the communicating element, and the sequence is secondary, with the sequence diagram, the primary focus is on the temporal sequencing of the communicating interactions. An example of a communication diagram is presented in chapter 4 of this dissertation. The communication diagram modeling technique models dynamic systems, it is capable of showing both discrete and continuous interactions, represents information that is deterministic, and is capable of showing sequential as well as parallel activity.

The fourth diagram type to be examined in this chapter is the communication diagram, which is another type of behavioral diagram, as all of those shown here, which captures information very similar to that shown in a sequence diagram. The difference between the communication diagram and the sequence diagram is one of focus. The sequence diagram captures the sequencing of interactions very well, while the communication diagram captures the identity of interaction sources very well. Two different models of the same aspect of the same referent system, presented from different perspectives. The communication diagram captures the source and target of interactions – the same messages and responses as shown in the sequence diagram – with more detail on the specifics of the source and target identity. Components include the object/class/instance block that can serve as a source or target; also the presence of interaction is modeled as a link between blocks.

The blocks of a communication diagram can, at their base form, stand for an object in the system, such as the object and its resulting life line in the sequence diagram. In addition, the block can also have the class that it belongs to indicated, and also identity as to whether it is an instance, or even a particular instance. It is represented in the diagram as a rectangle, with the naming information inside of it.

The links between the blocks of the communication diagram, which represent messages and responses, are drawn as lines connecting blocks. There is an associated arrow showing in which way the communication travels (from origination to target), and also some information naming the interaction. There is also a scheme of labeling that allows the sequencing of the interactions within the diagram to be determined, and a nesting of that scheme to allow for the representation of layered groups of interactions –

as can be shown with the interaction frames of the sequence diagram. The links can be used to show self-referential exchanges of information, by a looping line returning to the same block it comes from. In that case, of course, there is no need for an arrow – the source and target is the same block.

As can be seen within (*Figure 16.* Elements of a Communications Diagram), there are some specific graphical representations of the elements commonly found within a communications diagram. These include the communication link, which links the two communicating objects – the producer of the communication and the receiver of the communication – and indicates the direction of communication with an arrow. There are also the communicating objects themselves, represented as boxes. In an actual communications diagram example (see chapter 4) there would be more details for both of these elements.

Discrete Event Systems Specification – The discrete event systems specification, or DEVS, has been presented as part of the theory of modeling and simulation, through Zeigler's treatment (Zeigler, et al., 2000). It is a method for representing, formally, the specifics of information concerning the discrete transformations that take place within a discrete event system. It does this by representing a seven-tuple of information about the event. The structure of the seven-tuple is M=(X, S, Y, δ_{int},δ_{out},λ,ta). The first element, X is the set of input values. The second element, S is the set of states. The third element, Y is the set of output values. The next two elements are each a transition function – the first, δ_{int} is the internal transition function, and determines when one state from S transitions to another state. The second transition function, δ_{out} determines how the overall state set of the system outside of this event transitions. The transformation of input to output is described by λ ,called the output function. And finally, the time

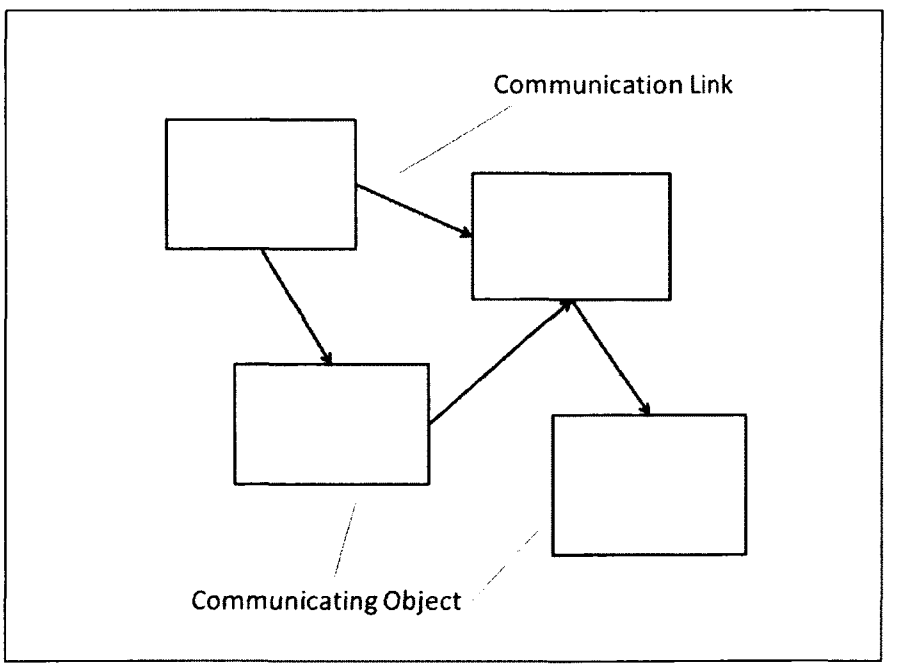

Figure 16. **Elements of a Communications Diagram**

advance of states is the last element, called *ta*. Together these represent the information about a discrete event within a system. What has been presented here is known as classic DEVS, and many variations and iterations of it have been developed and are now part of the body of knowledge. This discrete event specification is part of Zeigler's (2000) theory of modeling and simulation which includes not only the variations on classic DEVS as mentioned, but also the idea of the framework for modeling and simulation, which addresses many of the same issues covered earlier in this chapter in discussing the relationship between the conceptual model and the referent, and between the conceptual model and the simulation. Zeigler (2000) does well to identify that the resulting construct that performs simulation is a simulator, and that simulation is the act of what a simulator does, by implementing a model. This will be further addressed in Chapter 4 of this dissertation. The classic DEVS modeling technique represents information about a dynamic system, it represents information concerning a discrete system, the sequencing of activity is deterministic although the output function may have a stochastic element in its definition, and classic DEVS captures only sequential activities. In addition to what has been discussed here, which is called alternatively classic DEVS, or atomic DEVS, there is also the concept of distributed DEVS, which introduces an additional element, δ_{ext}, allowing for several DEVS specifications (or individual discrete event models) to be brought together, as a system of systems, allowing for each sub component (each of them a DEVS atomic component) to act, produce output, and also alert each other as to state changes and progression, so that where they are dependent on the knowledge of the states of other atomic DEVS components, that knowledge is available. This is described well in Wainer (2009).

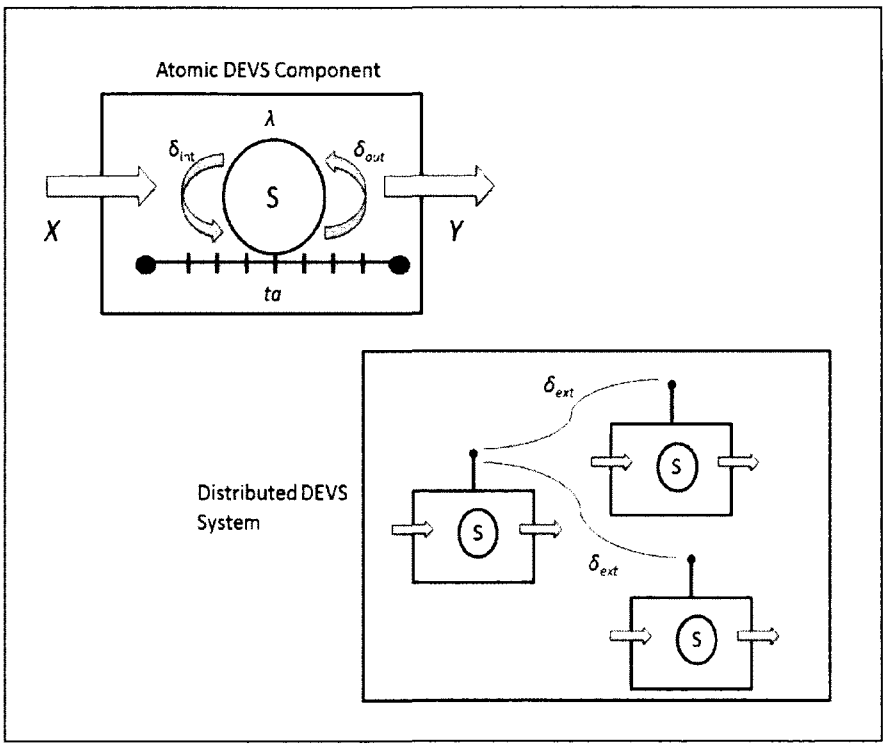

Figure 17. **DEVS Atomic and Distributed Systems**

The graphical depiction of DEVS, which is not typical, but is used sometimes for illustration and to clarify a relationship among distributions of atomic DEVS components, is shown as an example in (*Figure 17.* DEVS Atomic and Distributed).

Flow Charts – The flow chart considered as a modeling technique has been included in the survey because of their historical significance to computer science, and because they are mentioned in Sowa (2000) as an object focused method of representing information about a system. For a simple example see (*Figure 18.* Elements of a Flow Chart). The classic graphical method for showing the steps that a process or algorithm goes through. A flow chart does not describe the specific effects or timing details of the process steps, but represents the topology (initialization, halting, and post-process effects, each in part) of the various elements of the algorithm. Originally, flow charts were

known as Gilbreth Process Charts, and were first presented in Gilbreth (1922). Initially flow charts were employed to show the sequencing of industrial activities. The flow chart as a modeling technique represents information concerning the sequence of steps within an overall system, and also the decisions that can lead to different paths of steps within the system. While different branching and decision points in the flow chart can be represented, there is not a mechanism of representing changes to the structure of the sequence. Flow charts have mostly been replaced in modern approaches such as UML and SysML by the activity diagram (also surveyed here) (OMG, 2002). A flow chart represents information about a system that is dynamic, it represents discrete transitions between states, it is deterministic, and represents sequential activity.

Examples of the elements of a flow chart are shown in (*Figure 18*. Elements of a Flow Chart). These are the common graphical representation of the common elements. There are specific communities that have, for themselves, additional elements, or elements with special meanings (Sterneckert, 2003). Rounded rectangles (or sometimes

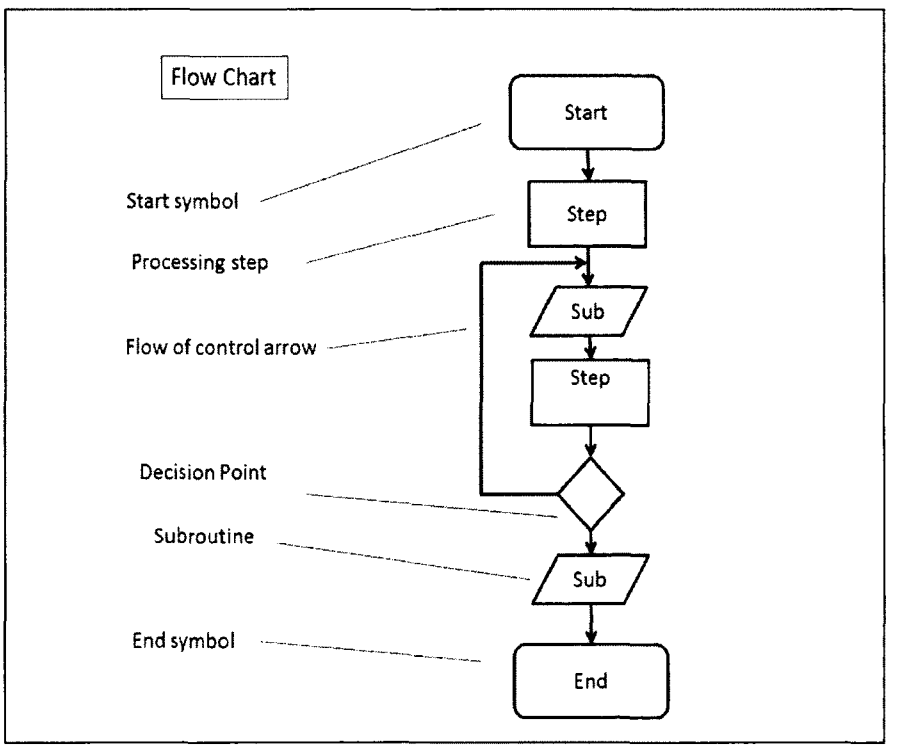

***Figure 18*. Elements of a Flow Chart**

ovals or circles) are used to mark both the beginning and end of the flow chart. Square-edged rectangles are used to represent the processing steps of the flow chart. If this were specifically for a program, then the processing steps would represent computational events. The angled rectangles are subroutines, which have a definite start and stop point, and can be represented separately as their own model. The diamonds are decision points,

where flow of control (represented by arrows) can change and branch to different "next-steps" in the flow chart.

Petri Nets – Also called Petri network graphs, this is a method for showing conceptual places (states) and transitions between them (Reisig, 1992). It is a modeling method intended to represent a complex topology of different states, and how objects (the "tokens" of a Petri net) move from state to state (Petri, 1966).For a simple example see (*Figure 19*. Elements of a Petri Net). The mechanism used in the modeling method to indicate movement from state to state is called a transition, and the possible route for a token through the topology of states is indicated in the model by arrows, indicating the "route of transition". One of the strongest contributions to modeling that Petri provided was his concept of concurrency of activity. With a Petri net, the advance of a tick of conceptual time is universal throughout the network, such that if the transition of one or more tokens from some state to another state is due to occur at time X, when time X is reached all of the transitions keyed to that time (or system condition) will occur simultaneously. The focus of the method is to show the possible transition routes between states; to describe the transitions of tokens; and to enable the visualization of where a token is at any time step taken in sequence by the model representing the operations of the referent system it represents. The Petri net modeling technique represents information about a dynamic system, it is capable of representing discrete actions, it can represent transition conditions that are either deterministic or stochastic, and the technique is capable of representing sequential and parallel activity (Peterson, 1981).

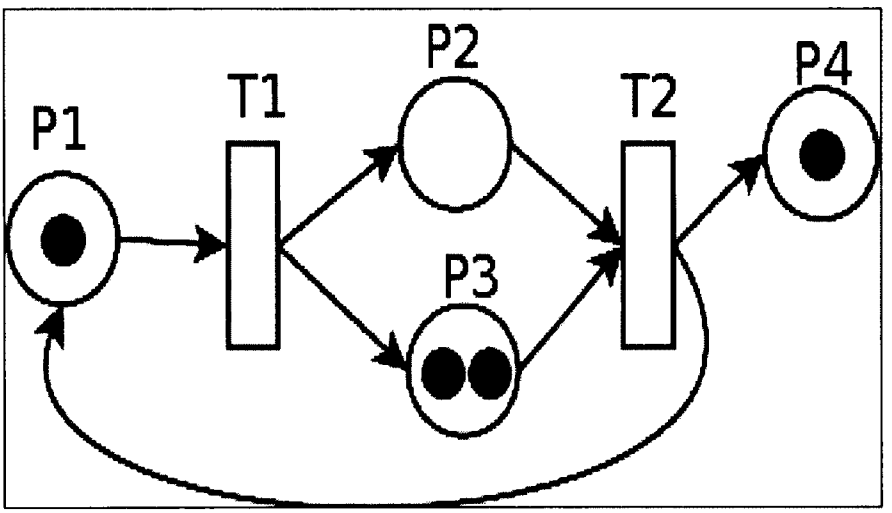

Figure 19. **Elements of a Petri Net**

An example of the graphical elements of a petri net is given in (*Figure 19.* Elements of a Petri Net). The round (circle) areas are the places of the petri net. The separators between places (vertical rectangles) are the transitions. The dots on the places are the tokens. The arrows are the control flow – indicating which new place a transition will allow a token to travel to, once it leaves an existing place.

Not pictured is the possibility of a colored petri net. In this situation, each of the tokens is given some characteristic (a "color"), and each transition is not just defined in terms of which new place a token goes to, but it may vary for different colored tokens (Jensen, 1987).

Sequence Diagram – Sequence diagrams are part of the UML/SysML family of behavioral diagrams (OMG, 2002). They are also part of the sub group called the interaction diagrams, which include the sequence diagram and the interaction diagram (which is not included in this survey). The sequence diagram is a method for showing

the sequence of interactions between objects within the system (Booch, et al., 1999). For a simple example of the elements, see (*Figure 20.* Elements of a Sequence Diagram).

From the perspective of the sequence diagram, objects are defined by the object oriented view of the software design community, which is distinguished here from the definitions of an object within a model found earlier in this chapter. The information that a sequence diagram represents is very similar to the information about the system that a communication diagram represents, except the focus is different. With a communication diagram, the focus is on the object's methods that are interacting, and the sequence is captured secondarily. With the sequence diagram, the interaction sequence is the primary focus of the representation, although the identity of the interacting component is also shown. The sequence diagram is presented as an example in chapter 4 of this dissertation. The sequence diagram modeling technique represents information about dynamic systems, is capable of representing both discrete and continuous interactions, represents information that is deterministic, and is capable of showing sequential as well as parallel activity. The sequence diagram has some graphical components that are common to its use as a modeling technique, and these are presented in (*Figure 20.* Elements of a Sequence Diagram).

Sequence diagrams are one of the two diagram types from UML 2 that are part of the interaction diagram sub category of the behavioral diagrams. Sequence diagrams are intended to show the sequential interaction of objects within the model. The implication here is that interactions from one object to another are what happens within an object oriented worldview, which is what UML is designed to support. There are two

approaches to modeling the dynamic sequencing of the system with a sequence diagram, the first (basic form) is without activations. The second, which is with activations, employs a technique (described below) that makes clear the intentional sequential-nesting of multiple interactions between objects. The modeling elements that make up a sequence diagram are first the object. In addition to the objects, messages and returns from one object to another are modeled. A horizontal interaction from one object can be used to indicate the creation of another object. Complex messaging can be modeled using techniques for *Branches* and *Loops*. Further *Clarity* can be gained in the model by collecting a number of different messages and/or returns together with an interaction Frame. In a complex sequence diagram, where the sequencing of interactions between different objects allows them to overlap each other, it is possible to map which object is active, and also possible to model self-referencing interactions, such as call-backs and recursion. These are all described, in the following paragraphs.

Objects in the sequence diagram (not the same as objects in the OPR methodology see) represent the objects in the system from an object oriented design perspective. This means they are defined by a class, including both the objects and a number of methods affiliated with that object. As they appear in the sequence diagram, there is supposed to be an understood set of methods affiliated with the object (these methods are explicitly identified in a class diagram, one of the structural diagrams from UML 2) that can support the messages and returns that are modeled as part of the sequence diagram. The objects are represented as a vertical line in the sequence diagram, called the object's life line. It is possible to depict a whole class, borrowing a truncated class depiction technique that is used in some of the structural diagrams of UML 2, but it

is advised to do this only when knowledge about the class is required to render the sequence diagram useful.

Messages and *returns* in the sequence diagram are representative of the passing of information back and forth between objects. There are some different types available. The first distinction is for messages; there are synchronous messages and asynchronous messages available as diagram items. The first appear as a solid line with a solid arrow head, the line coming from the object initiating the message, and the arrow pointing to the object's life line that the message is addressed to. It is customary in the diagram to label the message for the method that initiates the message. The distinction between synchronous messages and asynchronous messages is one for software systems (primarily). A synchronous message is one where the sending object goes into a wait mode after sending the message, until the receiving object completes whatever request the message may initiate. An asynchronous message is one where the sending object sends the message, and then the portion of the system that the object represents goes on with whatever other activity it would perform next, without waiting for the process (or processes) that the message may initiate to complete.

Returns are similar to messages, except they are always sent back to the object that initiated a message, and in response to a message. They are less commonly labeled than a message, but it is not undone. When a return is labeled, it is often labeled with the name of whatever data element is returned in response to the original message. Returns are depicted in the sequence diagram as a dotted line. Both message and returns are typically depicted as horizontal lines between the life lines of the objects in question, however when the sending of a message or a return itself is expected to take a non-trivial

amount of time, it is often modeled as an angled line, showing that it takes up vertical space in the "timing" scale of the sequence diagram.

Activation Frames are used to show when a sequential combinations of messages and returns - which may include nested sequences - are desired to be shown in the diagram. They are represented by narrow vertical boxes drawn over the life lines of the objects. When there may be nested sequences, each nested sequence is represented by another vertical box. These can add clarity to the diagram, by showing which responses are paired with which messages, and also which series' of messages belong together.

Branches and loops can be shown by including one or more messages going to the target object's life line, but coming from the same origination point on the sender object's life line. In addition, if several branches are required to be shown, it is typical to depict the condition in the description of the message, so that it is clear in the diagram which branch is taken, and why. Loops are depicted by placing all of the messages and responses that are part of the loop inside of an interaction frame, and typically by labeling the frame as a loop with a name and also a conditional that determines when the loop iterates, and why.

Interaction Frames are blocks drawn around a group of sequence diagram elements - usually a collection of messages and responses - and then labeled with some name that describes why they are included together. One of the reasons to do this, as mentioned above, is to show a loop, but there are other conceptual reasons why the diagram will want to depict a group of sequential interactions as a named item. When a conditional exists on an interaction frame, it is also represented whatever alternative sequence of interactions will take place if the conditional does not hold.

Call-backs are depicted in a sequence diagram when an object has a message or response that originates and points back to itself as the target of the interaction. This could be for a number of reasons, but recursion is one possible case for doing this.

The graphical elements that make up the sequence diagram are the objects, rectangles representing the parts of the model that are the focus of the activation the diagram represents. Each of these has a lifeline, representing the relative passage of time (so indicating the ordering of activations of the objects). Messages between the lifelines of the objects involved are represented as solid arrows, with the direction of the message (sender to receiver) indicated by the arrow. Returns of focus back to the sender are indicated with a dotted line. When an object is in focus of activity for a period of time, this is indicated on the lifeline with a vertical rectangle, called an activation frame.

State Diagrams – The state diagram considered as a modeling technique has been included in the survey because of their historical significance, and because they are mentioned in Sowa as an object focused method of representing information about a finite state machines (Sowa, 2000). This technique represents information about a system, through a state-based (data or object oriented) paradigm of viewing a system. As mentioned earlier, there are two main approaches to considering a finite state machine (FSM), the Moore diagram and the Mealy diagram. In an FSM, the various states of the system are captured, and the relations between the states are transitions. FSMs are capable of producing output, and in the case where the state produces the output, it is a Moore machine (Moore, 1956); in cases where the transition produces the output, it is a Mealy machine (Mealy, 1955). It also allows for branching, as a flow chart. There are two types of finite state machine diagramming techniques, showing historically for

computer systems modeling, the general dichotomy seen in the ontology literature as distinguishing a basic understanding of a system from an object-centric view or a process-centric view. The two FSM types, as mentioned, are the Moore diagram and the Mealy diagram. The Moore (1956) is considered to be more object-centric (where by objects, the states of objects are recorded). In the Moore FSM type inputs to transitions are recorded as part of the original state before the transition, and the output of the transition is recorded as the next state. The second type of FSM is the Mealy diagram, where the transition itself has the characteristics of describing the input and output, separate from the states connected (Mealy, 1955). An additional diagramming technique for describing a state machine is the Harel statechart (Harel, 1987), but these are not considered here, belonging more properly to the history of development of the statechart for UML and SysML (see below). A state diagram represents information about a system that is dynamic, it represents discrete transitions between states, it is deterministic, and represents sequential activity. For graphical elements of a state machine diagram, see the examples earlier (*Figure 10.* Moore Machine) and (*Figure 11.* Mealy Machine) The circles represent states, the arrows represent transitions.

Statechart Diagram – Statechart diagrams are part of the UML/SysML family of behavioral diagrams (OMG, 2002). Historically, state charts were a part of core computer science, an overview is found in (Harel, 1987). A state chart diagram is a visual approach that shows the changes between system states. Statechart diagrams are very much rooted in the traditional view of a system as only a number of finite states connected by state transitions. Statecharts differ from state diagrams historically by having a different diagrammatic approach, although the information represented is the same qualitatively, the representation (and abstraction) is clearer. The chief difference resides when there are recurrences of the same state within a system. In a state diagram each time a state is entered, it is properly represented as a new node in the graph, but with

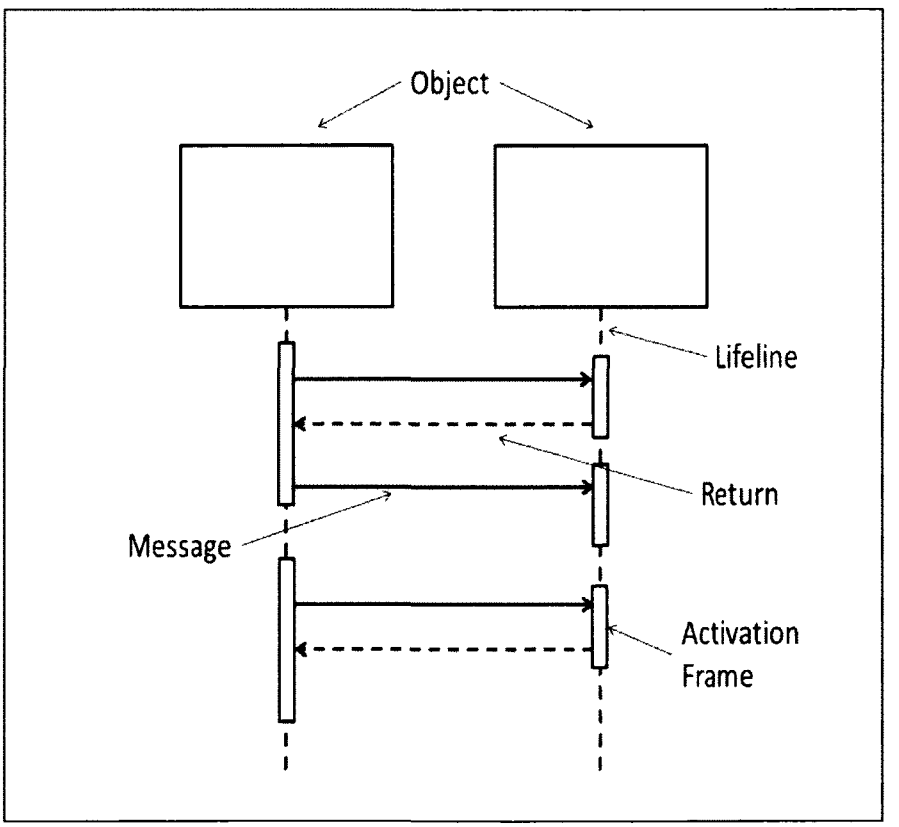

Figure 20. **Elements of a Sequence Diagram**

a statechart revisiting a state can be represented without diagramming a new node. The expression of understanding about the system is the same, yet the abstraction may be clearer with the statechart because of making revisiting states clearer. As a modeling technique, the statechart represents information about a dynamic system, represents discrete transitions between states, it is deterministic, and represents sequential activity.

Graphically, a statechart has much in common with a state machine. As a statechart is only responsible for exhibiting states, and state transitions, those are the only proper graphical parts of the diagramming technique, other than terminal points indicating start and stop. There are variations on the basic idea, with nested states (having a state element contain several sub elements connected by their own network of transitions) being one of them. These graphical elements can be seen in (*Figure 21.* Elements of a Statechart).

Use-Case Diagram – Use-case diagrams are part of the UML/SysML family of behavioral diagrams (OMG 2002). The use-case diagram modeling technique represents information about how an agent – either human or some other system or component – makes use of part of the modeled system. It represents the relationship between the using

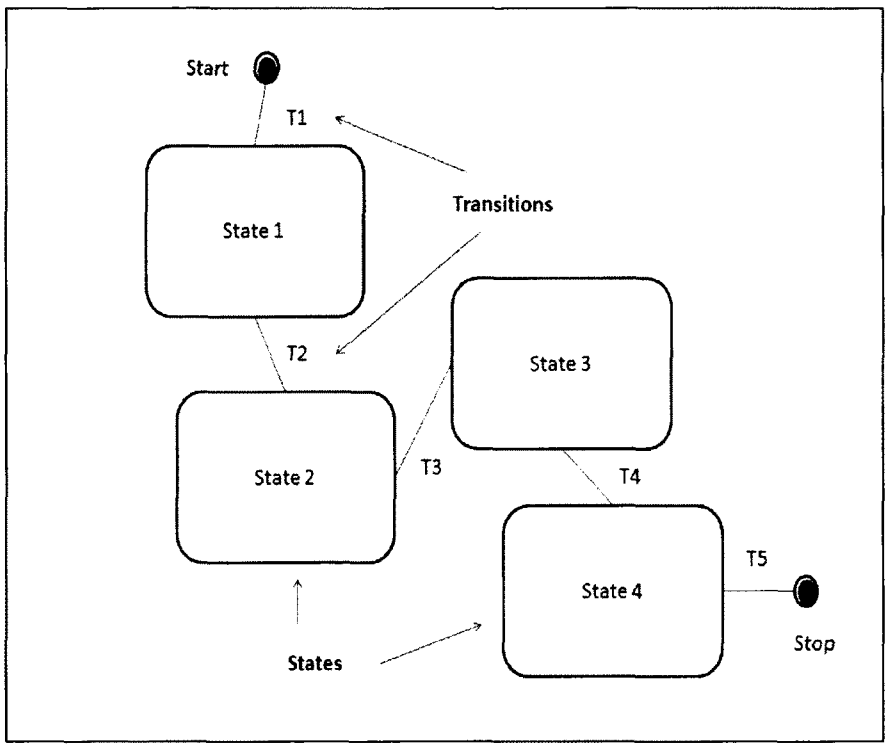

Figure 21. **Elements of a Statechart**

agent, and the identified role the system takes for their particular use (Booch, et al., 1999). There is not provision for specific temporal attribution, just a relational sequencing between use-cases, so that the individual uses might represent either a discrete instance or a continuous employment of part of the system. The use-case diagram is given a rigorous treatment, examining all of the specific elements and their role, in Chapter 4 of this dissertation. The use-case diagram modeling technique

represents information about dynamic systems, does not distinguish between discrete and continuous activity, represents information that is deterministic, and is capable of showing sequential as well as parallel activity. As can be seen in (*Figure 22.* Elements of a Use-Case Diagram) there are some specific graphical elements common to the representation of the elements of a use-case diagram.

First, there are actors, which are represented by stick men. Next there are the connectors indicating the relationship between an actor and a use-case. Finally the use-cases themselves are represented as bubbles, or ovals. In an actual example of a use-case diagram (see chapter 4) there would be details in the diagram giving more information to

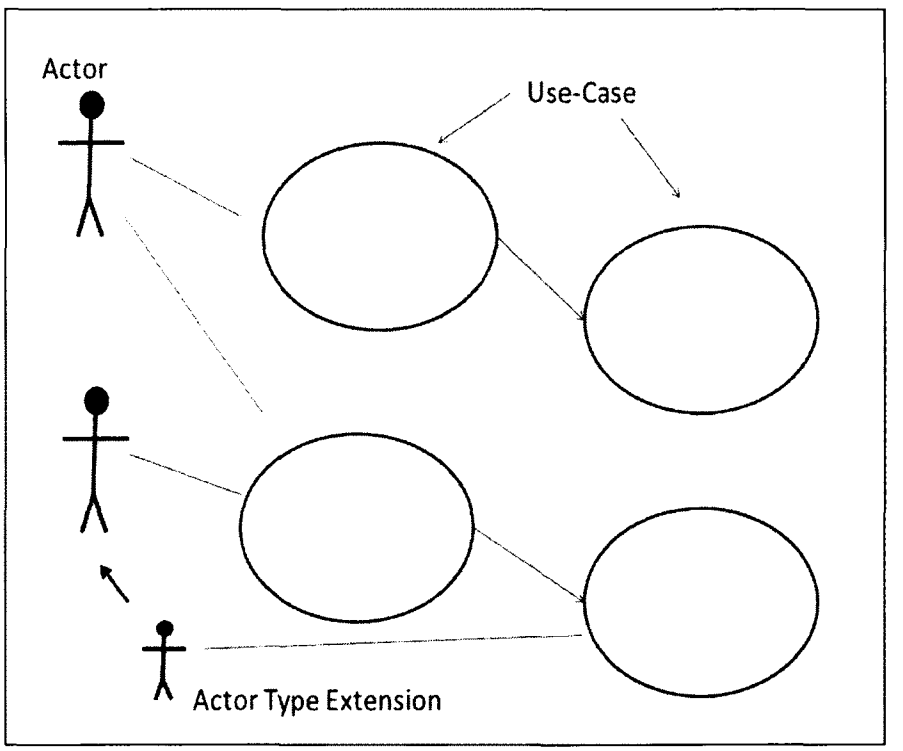

***Figure 22.* Elements of a Use-Case Diagram**

the use-cases themselves, as well as the actors and connectors.

The use-case diagram of UML is intended to show the various use cases that a system can be used for, by modeling the interaction of various actors with the various use-cases that are possible with the system. In this context a use-case (used here, as it is, for software engineering and systems engineering) is a model of the tasks that a user will undertake with a system, in order to reach a certain goal. Use cases are different from activities, in that they capture the intentionality behind a particular grouping of activities. That intentionality, as the name use-case implies, is for the accomplishing of a particular goal.

The elements of a use-case diagram are the actors; the use-cases; and the links and arrows connecting actor-roles to use cases, actors to other actors, and use-cases to other use-cases. The diagram also includes the possibility of determining the boundary of what is in the system, and what is outside of the system. A description of these elements follows.

Actors could be referred to as actor-Roles because they are often different roles that an individual actor will take, in order to accomplish a variety of different goals. They appear in the diagram as a stick figure. While it is usual to think of these as a human interoperating with the system in some way (activating it, entering data, retrieving data, etc.), an actor could also be an external system of some sort.

Use-cases are an element in the model that represents a particular grouped sequence of activities that together are to assist an actor in reaching a desired goal. They are represented in the diagram as an oval with a name for the use-case that they represent.

Links connecting an actor to a use-case are a simple line. They show which actor (or actor-role) the use-case is intended to support. These usually cross the boundary of the system, signifying that the actor is outside the system's proper definition, but the link shows the interaction that exists.

Arrows connect a use-case to other use-cases that are required in order to help the initiating actor satisfy the goal that they desire. When this relationship between use-cases is simply one of sequencing ("first this use-case, and then followed by that use-case"), the arrow is depicted in the model by a solid arrow. When the relationship is more specific - such as one use-case including another as a sub-case, or requiring that another be enacted first, then there is a depiction in the model of a dotted line arrow, and the arrow is labeled to capture the specifics of the relationship.

Finally, when one actor can perform all of the use cases of another actor, rather than re-show those links, the actor can have an extending relationship of the other actor shown in the model. This is depicted as an arrow, with a hollow arrow head, pointing from one actor to another. The extending actor can then have additional use-cases that it is linked to, that do not apply to the original actor that was extended.

From this sampling of different CMTs, there are many things that a model of a dynamic system can represent. Using this information as a basis for our evaluation, the next part of this chapter will identify a gap in the current practice.

2.3 MODELING THE DYNAMICS OF A SYSTEM

Each of the CMTs that were looked into in the preceding section allow for some representation of either a dynamic system, or some dynamic aspects of a system, as identified by the modeling technique categorization of Hester and Tolk (2010). As this

dissertation is interested in investigating processes in modeling and simulation, and a process is the cause of change (dynamic behavior), the alterations and transformations that these CMTs describe are the focus of investigation in the subject. A survey of the identified CMTs has been presented in the preceding section, identifying several descriptive taxonomical attributes of each technique. Before analyzing this survey, a specific look at what the dynamic elements are of each technique is here presented.

Change in Models. Taking the various CMTs that were introduced (p. 30), the elements of change that each can present is now identified. As mentioned, three of the four distinctions identified by Hester and Tolk have to do with the nature of how the modeling technique represents the dynamic nature of the referent system being described. Those three distinctions are (1) whether the described dynamics are discrete or continuous, (2) whether the described dynamics are deterministic or stochastic, and (3) whether the described dynamics take place in sequential series or in parallel. The fourth distinction indicates whether or not the system has any dynamic activity to be represented, or if it is static (all surveyed here are dynamic). The elements of the model that are changing are identified here for the nine candidate techniques. This will identify specifically what makes each technique dynamic. The purpose is not to identify any one technique as more or less capable as another, as all are useful, but rather to identify the different ways in which change is potentially represented by each modeling technique. So what is being presented here is a more in depth description of the dynamic elements in each of these techniques, from the sources identified in the introduction and description of the techniques, given above. Other than identify the dynamic element (what it is that is changing), the characteristics from the typology found in both Hester and Tolk (2010)

and Law and Kelton (2000) are also identified here. In order to be clear, the text for identifying these characteristics is presented here from the Hester and Tolk (2010) source:

- *"Static* versus *Dynamic*: When representing a system at a given point in time, the simulation is static. This is of particular interest when a temporal element is not relevant to the analysis. Typical examples for the static category are Monte-Carlo models. If time is of importance and the evolution of a system over time is needed, dynamic simulation needs to be applied. Examples of problems in which dynamic considerations may be important include degraded performance of components and conflict between opposing forces. As a result, dynamic problems are significantly more cumbersome and require a greater amount of computational effort than static problems. Understandably, dynamic problems are often represented with an equivalent form of a static problem, in order to make solving them easier. As can be expected, the time step and time frame over which the analysis is taking place all contribute to the additional complexity inherent in dynamic problems.
- *Discrete* versus *Continuous*: Within the discrete paradigm, state variables change instantaneously. This is often triggered by an event. If the states change continuously with time, the continuous paradigm is utilized. Mixed forms are possible, and are sometimes referred to as heterogeneous simulation methods (Franck and Zerbe, 2003).
- *Deterministic* versus *Stochastic*: If a simulation does not contain any random parameters and always produces the same output for a given input, it is

deterministic. If probabilistic components are used to represent not only point estimates but to generate variations in the simulation following the laws of statistics, the simulation becomes stochastic. It is of course unrealistic to assume that *any* problem operate sin a completely deterministic domain. A deterministic environment is a simplification, but it is often necessary to reduce computational effort. Additional methods to account for uncertainty include approaches such as designing with a built-in factor of safety and incorporation of redundant system components. While these approaches avoid the computational burden of stochastic analysis, they do not address the underlying issue of the uncertainty present in the system. Further, they may lead to overly conservative engineering solutions.

- *Sequential* versus *Parallel*: If the underlying paradigm of a model is a single-server queue, the simulation necessarily is sequential, as all events and state changes must be modeled as a sequence in the queue. If several events can be simulated simultaneously, this introduces parallel simulation. While scheduling is the main challenge in sequential simulation, synchronization becomes a new challenge in parallel simulation. (Hester & Tolk, 2010, p. 3)

In addition to these characteristics, capturing some general information about what the dynamic behavior representational capability of the technique is also included. In some cases, this is the passage of time, in some cases it is the change in system state. It is possible that it could also be a change in the state of a single object of the model. To distinguish between these two, defining what is referred to here as system state is required. As a practical guide, a state can be thought of as an operational mode that the

system it belongs to may enter. The system may be in that mode for a period of time. While in that mode (the "state") the system is capable of performing certain behaviors. This is seen from the definition of state used for software and system modeling in Weilkins (2006). There it is stated that, "A state represents a set of value combinations for the underlying element. It has a name, and may have an internal behavior that is executed based on defined events" (p. 194).

Each of these states is thought of as being different and identifiable from other states. This is also corroborated from the literature, see Friedenthal, et al. (2009) for the following, "The states in any one region [of the system] are exclusive; that is, when the region is active, exactly one of its states is active" (p. 241).

Thus, from the literature we see that there are a number of different states, each defining a different set of behaviors for the system, and each of these is independent and identifiable from each other. At this level, the dynamic behavior of some modeling techniques would be to show the movement between the states. This is termed state transition, and is defined as, "A transition specifies a state transition. It is a directed relationship between two states, and defines a trigger and a condition that both lead to the state transition, as well as a behavior that is executed during the transition." (Weilkins, 2006, p.195).

As mentioned, this view of states and state transitions is from the literature on systems and software modeling. From Hester and Tolk (2010) , we have already seen that the whole community of models may include those whose behaviors are continuous rather than discrete, so it remains that state transitions in a continuous behavior system

may not have the same level of distinction and separation between states as in a discrete behavior system.

In addition to state transitions, and time change, it is also possible that a single object of the element may change, in which case an identifiable value combination (Weilkins, 2006) will have one or more of its values change, but nothing else about the system will change. If it is in a behavioral mode, that mode remains (i.e. – the state remains) but one of the value elements of the system is altered.

From these definitions, we can see (at least) that the dynamic behavior capable of being represented could be between states (change of state) or of elements within a state (change of object). Each of the surveyed elements below, in addition to having the classification of (Hester and Tolk 2010) applied will also identify if it represents state change or object change as the dynamic behavior it models.

- Activity diagram – What is represented in an activity diagram is the focus on what activity of a system being modeled is being done at the particular time. The diagram does not describe the temporal attributes of the activity, or the specific description of the activity (i.e. what it – the activity – changes within the system), only the sequential ordering, and series relationship of one activity to another. For the model of the referent system, this is a change in state showing which activity is active at any one time. As with the other UML/SysML techniques, it is intended to work together with other modeling techniques to show a more complete view of the referent system than any one of the techniques is capable of singly.

- Communications diagram – Specifically the dynamic element about the referent system that the communication diagram technique exhibits is the exchange of information between object-oriented design elements, represented as communication links, but representing the messaging and invocation that take place by the methods of objects operating on data of their own and of other objects. What is changing is system state representing which element has "control" over the data objects representing that information (change of state), for the period that the object-oriented design element has "ownership" of that data object (transformation of an object).
- Discrete Event System Specification – The atomic DEVS specification is capable of showing several dynamic elements, within a discrete event system. First it can show how the sequence of states changes over time (change of state). Second, it can show how a state produces some output (transformation of an object). Finally, it can show how an "external event" (in DEVS terminology – some event other than the state running out its natural "lifespan") can affect the state of the model.
- Flow chart – A flow chart describes the topology of the different system states, and how they are connected, and the changes of focus from one state to the next.
- Petri Nets – Describes the transition of tokens between different places (change of state) within a system In that a place in a petri net can be responsible for some activity or transformation to a token, this may also be represented (transformation of an object).

- Sequence Diagram – The same information of the communications diagram is re-presented within the sequence diagram, however with a different perspective. Here, rather than indicate the object-centric view of the methods that the objects have invoked (represented as messages and returns), the temporal sequence of the exchanges is the focus of this CMT. Although the focus on sequence of the state change is the focus here, rather than the focus on the individual elements performing in each state, as with the communications diagram, this is still a dynamic change of state that is represented. Details on the transformation of objects are not a focus of the sequence diagram.
- State Diagram – This is similar to a flow-chart, yet without as much emphasis on the actions and decisions that take place between the states, focusing more on capturing the possible states, and their connection to each other.
- Statechart Diagram – The modern UML/SysML re-implementation of the state diagram CMT allows the modeler to represent similar information – the states of a system, and the change of focus in system operation from one state to the next.
- Use-Case Diagram – This CMT represents the dynamics of the referent system that deal with the sequential operations of the different portions of the system that work together towards a stated goal, for a particular user. As the user represents not only the initialization of those processes for some specific goal, but also the introduction of input, and then the reception of output, the change of these is also part of the use-case diagram (transformation of objects). The specifics of this last change (of the input to the use case, into the output from the use case) is not clear, however, without relying on other modeling techniques from the larger

UML/SysML family, so that the use-case diagram on its own is only capable of representing a dynamic change between states of the system, each state identifying what portion of the overall system would be used to satisfy a specific use-case.

These various CMTs and the possible change that they can exhibit can be shown succinctly in a table.

Table 3

Dynamic Elements

Conceptual Modeling Technique	Dynamic Elements
Activity diagram	Change of state.
Communications diagram	Transformation of objects, and a change of state.
Discrete event system specification	Transformation of objects, and a change of state.
Flow chart	Change of state.
Petri nets	Change of state and transformation of objects
Sequence diagram	Change of state.
State diagram	Change of state.
Statechart diagram	Change of state.
Use-case diagram	Change of state.

From this survey of the dynamic elements within the modeling techniques, several things can be seen from analysis. First there are, as to be expected from a variety of

CMTs, very different specific elements that can be the subject of dynamic activity. Each of the techniques, as a modeling technique, shows this dynamic activity, in a way of expressing information about the system that is being modeled in any one particular application of the CMT. Second, however, each of the dynamic elements can be categorized into either state changes (changes of the state of the system) or object transformations (changes to some object within the system). It is possible that other techniques surveyed might show other dynamic elements, but for the group represented in this dissertation these are the two options identified.

Typological Characteristics of Techniques. As shown in the previous section, the dynamic elements of the techniques surveyed here can be shown in two categories. One of the categories is dynamic changes to markers of substance (that is, according to substance theory, the objects and their attributes within a system). The other category is dynamic changes over time of system state. Combining these two possible dynamic methods (state change or object change), along with the other three taxonomical categories presented above, results in the following table (Table 4

System Aspects Represented by Model Techniques). The definitions of the terms were given in the preceding section (p. 99).

Table 4

System Aspects Represented by Model Techniques

CMT	State Change/ Object Change	Discrete/ Continuous	Deterministic/ Stochastic	Sequential/ Parallel
Activity diagram	S	D/C	D	S/P
Communications diagram	S/O	D/C	D	S/P
DEVS	S/O	D	D/S	S
Flow chart	S	D	D	S
Petri nets	S/O	D	D/S	S/P
Sequence diagram	S	D/C	D	S/P
State diagram	S	D	D	S
Statechart diagram	S	D	D	S
Use-case diagram	S	D/C	D	S/P

What is seen from this chart is that using the characteristics from Hester and Tolk (2010), and splitting up the dynamic characteristic into its two possible categories, we begin to have some indicator of the attributes, and representational possibilities of each of the modeling techniques. What emerges from analysis, however, is that there are broad typological similarities. Using the characteristics identified here the activity diagram, sequence diagram, and use-case diagram have, for example, identical features. From the description and from the literature it is clear that they have different uses, represent different information concerning their referent system, and result in different models

visually. Additionally, in each of the four characteristics (state/object change; discrete/continuous; deterministic/stochastic; sequential/parallel) there are some techniques that can show either possible categories, but that none of the techniques is capable of both categories for all four characteristics. What this table shows, and this is highlighted and detailed in the following section, is that the current methods for identifying the characteristics of a technique do provide information about the technique. But it is not enough in order to distinguish what it is capable of modeling, in contract to what other techniques are capable of modeling.

2.4 KNOWLEDGE GAP

Allowing that Hester and Tolk (2010) have taken into account other state of the art attempts to catalog types of conceptual models, it can be seen from the typology of the techniques surveyed that even using the various criteria presented already, that there still remains unanswered questions concerning techniques when they are compared to each other. As seen in the previous section, there emerge several techniques that have the same profile given the typology available from the literature, but from among those an engineer would still want to make an education selection concerning which technique should be employed for which task. In addition, although there is with each of the techniques chosen (all of which exhibit information about dynamic systems) a representation of some dynamic element about the system that is changing, this is universally (at least with the techniques surveyed) a change in state about the system (either the system itself, or some subset that is being changed), and only in one third of the surveyed techniques a transformation of some object of the system represented in the model.

The Gap Identified. Consider that in a system that is modeled, as per Law and Kelton (2000), the *reason* for modeling is because the system (for any of a variety of reasons) is unavailable to allow for a closed-form solution useable for analytics. If this is the case, then there can potentially be representable by the model many ways in which the state of the system can change. Although the surveyed techniques all show some state change, none give the specifics of how that state change took place – what it is about the system, specifically, that changed. Being able to answer these questions in a manner useful for a systems engineer seems a requirement in order to advance the ability to select from among available techniques. Not having the ability to answer these questions with available typological methods represents a gap in the current state of knowledge.

The gap in the state of the art is then identified as:

"There exists an inability, given current techniques for describing and comparing conceptual modeling techniques one to another, to quantify the specifics of a modeling technique so that it can be compared to another" (CITE).

Gap Specifics. The gap in the knowledge identified here needs to be given more specificity in order to be addressed by a research question. Two general questions have been identified concerning conceptual modeling techniques that a modeler might be required to answer, formally, in order to make an education selection of a specific technique for a specific modeling task. Those two questions are (1) comparing the potential for a technique to show different **aspects of dynamic behavior**, as identified in the typology set forth by Hester and Tolk (2010); and (2) comparing the **specific contents of a modeling technique** that can be represented by a technique to those that

another technique can represent, so that what a conceptual model is representing about a system that changes from state to state can be assessed and evaluated. These are given more specific meaning here.

Aspects of Dynamic Behavior. From Hester and Tolk (2010), the typological attributes that they give to models, even assuming all dynamic (vs. static) conceptual models, are all elements describing the nature of the dynamic change within the model. From this, it is assume that there are elements of distinction among models that have to do with their dynamic behavior. A formal method for describing a conceptual modeling technique that is capable of showing dynamic behavior must include such dynamic elements in its representation.

A formal method that accounts for all the dynamic elements, for describing and comparing modeling techniques, would close this part of the gap in the knowledge. The dynamic elements from Hester and Tolk (2010) are a good starting place, but they will be have to be evaluated along with other elements later in the dissertation.. They are all describing the transformations that can take place within the modeling technique, and they individually address definitional aspects of those transformations that can affect the other aspects. Those transformations have been identified, earlier, as the system operations called processes in the literature – a term that will have to be defined later. Using that nomenclature, this part of the gap can be satisfied if there could be found a formal method for describing the processes of a conceptual modeling technique. This formal method would include information concerning the distinctions of dynamic behavior so it can be identified, and compared to that of other modeling techniques.

Specific Contents of a Conceptual Modeling Technique. As motivated by the realization that all of the modeling techniques surveyed represent some sort of state change within the system being modeled, but none of those techniques give the specifics of what about the state of the system is changing from state to state, this second part of the gap becomes necessary. This necessity is to be able to show, by a formal method, what there is that is different about the state of the system after a change of state. In order to formally exhibit this, specifying what the technique can exhibit about the system in one state, and then in another state must be part of an effort to close this part of the gap. As shown earlier (p. 49), the parts that make up a system are the objects, processes and relations, a formal method that represents what a system can state about a system, should specifically show what the technique can represent about each of these parts of the system. Taking this as a way of showing the state of a system, it could then be formally identified what a state change represents within that conceptual modeling technique.

CHAPTER 3

RESEARCH QUESTION AND METHODOLOGY

This chapter examines the gap in knowledge that was presented at the end of the preceding chapter, and then presents a research question that will motivate the remainder of the dissertation to provide a solution addressing that gap in the knowledge. It does this, first, by deriving the research question from the gap in the knowledge, as informed by the results of the literature review and analysis performed in the preceding chapter. Next it shows the outlined method of answering that research question found in the remainder of the dissertation, chapter by chapter, and what they do to provide an answer to the research question. Finally, this chapter presents a research design methodology that fits the work that was done, and again ties each piece of the dissertation to established steps making up the research design.

3.1 RESEARCH QUESTION FORMULATION

The gap in the knowledge that arose out of the literature review is that although there are many modeling techniques, and even methods for typologically identifying those techniques and some of their aspects, there is not currently a method that can address the specifics of a technique, formally enough to enable specific comparison between techniques. The two identified specific aspects of that gap addressed are that it should be, but currently is not, possible to describe two different classes of information about a modeling technique. The first is an identification of the specific characteristics of the dynamic aspects that the modeling technique enables representation of. The second is a complete description of all of the components of a modeling technique, in such a way

that they can be identified and compared between techniques. The second of these two aspects of the gap in the knowledge, if addressed, would mean that all parts of a model would be describable. That suggests that however the first part of the gap in the knowledge is answered, if possible, it should be based in identification of parts of the second in ways to identify the dynamic elements.

One of the concerns raised in identifying the gap is that many conceptual modeling techniques have, as a dynamic element, the change in state of a system that a model in the technique's style is describing. As there is not currently a formal way of capturing this for the purpose of describing and comparing modeling techniques, a feature of the research question that addresses the gap should ensure that the method of identifying and describing the dynamic element of a modeling technique is able to show dynamic change to any part of a model. Taking this as the beginning of the formulation of the research question, we see as a requirement, that the two parts of the gap should be represented in the research question; that the two parts of the gap influence each other, so that if the first part is answerable, it almost necessitates the second part also being answered; and finally that if a way of identifying the dynamic capacity of a modeling technique does result, that it should be capable of showing dynamic change to any part of the model.

Research Question Specifics. Taking these requirements for the research question into account, we can see that there are several requirements that should be addressed. Since the gap is the absence of a method to formally describe and compare conceptual models, the research question should first be targeted at determining if such a

formal method is possible. Such a method would have these qualities, if it were to answer the identified gap:

1. Describe all of the components of a conceptual model, using a meta language that could be applied to any conceptual modeling technique.
2. Describe the dynamic capacity of a conceptual modeling technique with specificity (the components discussed in the preceding chapter, and others as they are derived from the literature).
3. Not limit the ability to have the dynamic capacity of a conceptual modeling technique to address changes to any part of the model, including the model itself – meaning the structure and nature of the components from point number 1.

This finally leads to a research question. Taking into account all three points and the gap in the knowledge, the research question arises as the following.

"Is a formal method possible that can fully describe the components, structure and dynamic behavior of a conceptual modeling technique?"

This is the research question that will serve to guide the remainder of the dissertation. Answering it, however, must be accompanied by some metrics in order to determine success. For those metrics, measures of merit will be presented.

Measures of Merit Defined. The most direct way of answering this research question would be to derive such a formal method. This is the approach taken in this dissertation. However, such a method once suggested would have to be evaluated to see whether it could satisfy the requirements identified that determine whether or not the gap is addressed. This means that some measures of merit should be determined, and relied

on, in determine whether or not the research question, and hence the gap in the knowledge, are addressed by the formal method that is derived.

Capability for compositional description - First, as a measure of merit, the formal method should be able to describe modeling techniques with enough specificity so that similar techniques can be addressed, in order to make an informed decision between those techniques. To satisfy this measure, some definable differences between techniques should be identifiable from application of the technique. An example question that should be answerable would be "Is it exhibited that technique 'A' has more representable capacity than technique 'B'? If so, then specifically in what area is 'A' more expressive than 'B'? This measure of merit will be called the measure of "Capability for compositional description".

Capability for definition for dynamic content - Second, as a measure of merit, the formal method should be able to describe the definitional parameters of the dynamic change capacity of a modeling technique, again so that such capacity could be compared between techniques. The parameters should include at a minimum, but perhaps more, characteristics such as those identified in the later part of Chapter 2 – timing questions, whether the dynamic behavior is stochastic or dynamic, and so on. An example question that should be answerable would be "In what specific ways does the dynamic change capacity of modeling technique 'A' differ from that capacity in modeling technique 'B', and what are the parameters of that difference?" This measure of merit will be called the measure of "Capability for definition for dynamic content".

Capacity for technique enhancement - Third, as a measure of merit, in addition to applying the technique to comparing techniques of differing sorts, the formal method that

is derived should suggest what is possible in representation within a modeling technique, so that once the method is used to describe a technique, it should be able to suggest in specific ways how that technique could be modified, or extended, in order to make it capable of representing more about a system. An example question that should be answerable would be "If modeling technique 'A' is described, in what specific ways could technique 'A' be extended in order to allow for more expressivity than it currently has as a technique?" This measure of merit will be called the measure of "capacity for technique enhancement".

3.2 STAGES OF ANSWERING THE RESEARCH QUESTION

In order to attempt to answer this question, the remainder of the dissertation is split up into a number of chapters that each contribute to closing the gap in the body of knowledge. These are addressed in the following list.

Developing Formal Method (Chapter 4) – A second literature review is the first part in the following chapter. It will explore what the literature has to say concerning the identifiable components of a model, and how they work together. In performing this investigation, and synthesizing the findings from the review, a formal method of describing the components and the ways in which they merge in order to describe the behavior of a system is also presented. The analysis and synthesis of these findings are what enable the formal method to be derived. The sections involved in this stage are an understanding, from the literature, of what the composition of a model entails; what objects are identified as; what processes are identified as; and what relationships are identified as. This is followed up

with the presentation of the formal method itself, involving a series of definitions, axioms, formal statements, and corollary observations (Appendix 2).

Evaluation of the Formal Method (Chapter 4) – The formal description method presented in the chapter on the second literature review is a rational construct – rationally ordering all that is said in the literature about the possible components of a model, and how they merge together. This formal method has the intended use of describing the elements of a conceptual modeling technique, and formally identifying what about a system they are representing. In order to assess if that is accomplished and possible by the method, it is applied as intended to a cross section of different conceptual modeling techniques, and its descriptive capability is shown (p. 143). The "capability for compositional description" measure of merit will be able to be answerable by the work presented in (p. 162).

Implications for Dynamic System Modeling (Chapter 4) – From the evaluation of the formal method, some interesting patterns begin to emerge about what the formal method has to say about the processes (components of the model that are responsible for dynamic changes) of a model. These are identified and formally quantified and categorized in this chapter, giving an even richer treatment of processes within the formal method. As the research question is concerned with what about a model a process can address, this is providing the means to answer that question. Applying the richer process formal description to the conceptual modeling techniques that made up the evaluation in the chapter preceding shows how this is useful. The process formal description is then applied to a variety of the changing elements (processes) from the surveyed

modeling techniques, in order to capture some specifics about what is addressable. This gives a more formal treatment to the question asked earlier in this chapter – what is it about a dynamic system that a modeling technique describes? Some specifics not addressed by any of the surveyed techniques are identified (p. 174). The "capability for definition for dynamic content" measure of merit will be answerable from the work presented at this point in the dissertation (p. 186).

Identification of New capabilities (Chapter 5) – The formal method and its rich description of processes is used to investigate and evaluate a conceptual modeling technique not included in the original survey. The identification of specifically what the technique can show about a dynamic system is made. Then the quantification of what a technique is capable of showing is used as a motivation for extension of the technique into a new area – showing that the formal method, and the answering of the research question, enables the dissertation to identify and present new knowledge to the discipline of modeling and simulation. The "capacity for technique enhancement" measure of merit will be answerable by the work presented in Chapter 5.

3.3 RESEARCH METHODOLOGY

Choosing a methodology for a dissertation sets it firmly in the regularly recognized practices of academic research. For this dissertation, the choice was made to adopt a method from (Cresswell, 1994). There are a number of respected guides to methodology, but the specifics for this dissertation were accommodated by Cresswell. Following the Cresswell (1994) book on research design, the dissertation as presented

follows a qualitative design. Qualitative research design is a research project that explores, in depth, a particular topic (in the case of this dissertation, process representation in modeling techniques) and constructs some new theory concerning the subject matter by analyzing and synthesizing some observed data. In this case, the observed data consists of two literature reviews. The analysis of the literature review (p. 18) resulted in showing what a survey of existing conceptual modeling techniques can exhibit. The Developing of a Formal Method begins with further, directed literature review, when analyzed (Chapter 4), shows what the identifiable components of any model can consist of. Synthesis of the analyzed results of the second literature review resulted in the construction of a theory – the formal method for describing the functionality of conceptual modeling technique elements. Evaluation of the Formal Method is then performed by applying it (p. 128), in the manner it was designed for, against existing methods, in order to assess if it provided the information predicted in its creation. It did, showing that the formal method is useful. In analyzing the evaluation application, Implications for Dynamic System Model became apparent and further refinements to the theory were derived (p. 143), and these were then incorporated and the theory (p. 172) was re-applied for an evaluation. This second evaluation will be shown to have been successful, and also shown to have uncovered new knowledge about the state of conceptual modeling that was unidentifiable previously in a formal way (p. 182). That new knowledge was then combined with the theory in order to show the Identification of New Capabilities could be applied to an existing system in order to extend it in the direction of the new knowledge (Chapter 5). This is the final test of the developed theory, and is the precursor to stating the conclusion of the dissertation (Chapter 6).

CHAPTER 4

THEORY GENERATION AND EVALUATION

This chapter provides the main qualitative offering of the dissertation. In order to answer the research question that was presented in the preceding chapter, the formulation of a method that can describe all of the aspects of a modeling technique that could represent any sort of dynamism that a modeling technique could exhibit. Failing the formulation of such a method, if it could be shown that such a method is NOT possible, then that would also answer the question. To review, that question has been identified as the following.

"Is a formal method possible that can fully describe the components, structure and dynamic behavior of a conceptual modeling technique?"

The effort to answer the question by developing and exhibiting the method will be attempted. It will be shown that this attempt has been successful. This chapter reveals the second literature review that was completed, in order to identify what the formal method most include, and also assembles that method in such a way that it can answer the question by satisfying the measures of merit identified in Chapter 3. After applying the measures of merit to the resulting formal method, it will be shown that further details concerning the nature of dynamic change within a model can be further described, based on analysis, and that extension in the particular area of interest for this dissertation is also presented here. This refinement is again based on analysis, and revealed again by induction from the grounded information that we receive from the literature.

4.1 COMPOSITION OF A MODEL

In order to answer the research question by presenting a formal method, exploration of the appropriate literature is required to uncover the philosophical basis for what potential component categories are necessary to make up any model, and the potential characteristics for those components. The dissertation's task then, would be to analyze and synthesized those components into a meaningful whole that satisfies the measures of merit.

It is first necessary to identify what the components are that make up a model. From Definition 1, it has been seen that "a model is a purposeful process of abstracting and theorizing about a system, and capturing the resulting concepts and relations in a conceptual model". As such, the model is a sign standing in place of the system it represents. Viewed this way, models are tokens for knowledge – about the system they are representing. Each component that is identified will either have to be useful for the model to either represent some aspect of the system being represent, or necessary in order to internally support the whole representation of that system. Each component, then, can be considered in isolation from all the rest of the model, yet it represents knowledge about some part of the system – as the model itself is representing knowledge about the whole system. In this relation (part to whole) although each component can be considered individually, they are all intended to contribute (and be part of) the whole model.

Definition 3: A model component is an identifiable part of the model that represents some part of the knowledge that makes up the whole model, but which can be considered individually as well.

As discussed in Chapter 2, the model must be representative of the system, and encapsulate enough of its complexity in order to be expressive (to answer whatever question is being asked of the model), yet simple enough to warrant the extra effort of using the model to represent the system, rather than work with the system itself. Taking this as a departure point forth is second literature review, what is said about the components of a knowledge representation, and then more generally, any model, is

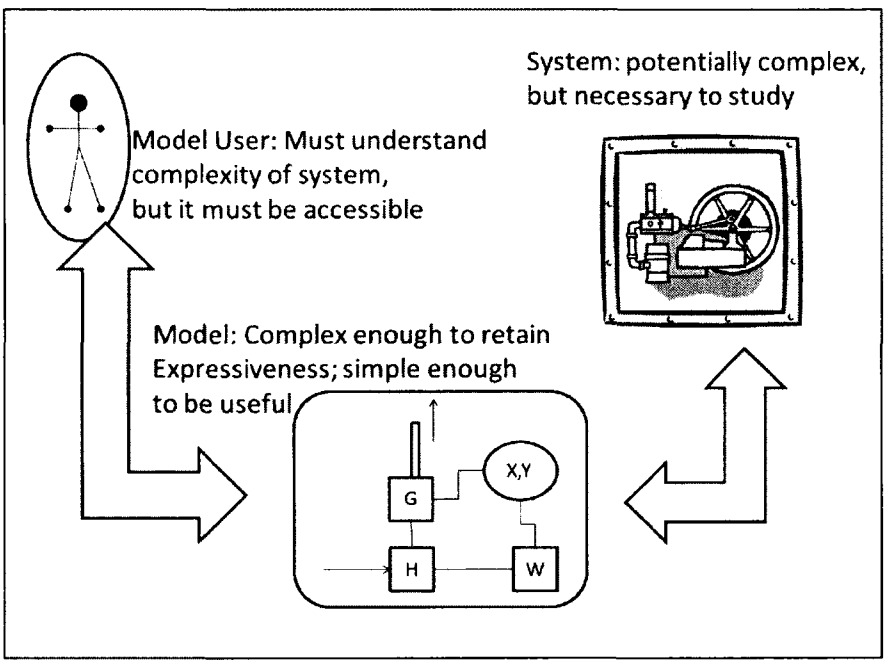

Figure 23 **Models represent expressive representation but reduce complexity**

brought out of the literature.

The literature of conceptual models identifies many elements that are part of models. In (Robinson, 2008) these are enumerated as:

Objectives

- Inputs
- Outputs

- Content
- Assumptions
- Simplifications

Examining that list, there are two categories of informational elements about models. First are the elements that make up the model – inputs, outputs and content. Second are the elements that describe the character of the model – objectives, assumptions and simplifications. For the purpose of exploring the nature of the components of a model, the first set is of interest for consideration here. The effects of the elements that describe the character of the model are derived may influence the elements that make up the model, but those effects will be present in the definitions and parameters of the elements that make up the model.

As discussed earlier, and visualized in (**Error! Reference source not found.**), the common components for the modeling techniques of a wide variety of different domains are all objects, processes or relations. The literature on knowledge representation, and the ontological representation of knowledge, is relied on to show that a system can be described with these three components.

From the literature of information theory, we can see in Shannon (1948) that although the transferring of information includes a speaker, a listener, and a medium of transfer, it is implied that information will exist in "symbols". This is also held up by the earlier (Ogden and Richards, 1923), which holds that a representation of a conceptualization is a symbol. This idea is also held up in the literature on ontology (Smith and Grenon, 2004), which identifies that the symbols for describing a world consist of symbols for enduring entities and symbols for occurring entities. These are

described as objects and processes. Objects are distinguished and characterized as an entity that has enduring identity over time. This enduring identity will be known as an endurant – the term used in Smith and Grenon (2004). Processes are characterized as an entity that occurs within the lifespan of the system being described (hence, an occurent, the term used), as the transition between two states of being (Smith and Grenon, 2004). As processes may be perdurants, which is to say that they have some time span to their occurrence, they aren't by definition limited to being point transformations. In regards to the identity measure of endurants, it is necessary that occurents and perdurants also have an identity, else it could not be the point of knowledge representation (in a model or otherwise), but that identity is only realized during the instant (or duration, in the case of a perdurant) of occurrence.

Taking these two views (whether a system should be viewed as a number of objects, and they change because of processes; or where everything is in a constant state of change) together, under the frame of considering a model as a representation of knowledge, we see that a model (Robinson, 2008) should accept input, produce output, and have some content that describes how the output is produced from the input. As the entities of a world are (at least) objects or processes (Smith and Grenon, 2004) then the input, output and content should address these (as will be demonstrated later, a third possibility exists, and that is the association of those objects and processes – which will be referred to as relations). As the model includes as separate things the content, input and output, then it follows that all three of these are distinguishable from each other. The introduction of input into the content is identifiable as being separate from the content, as is the output. If the content contains a description of how the input is changed to become

the output, then regardless of what the input consists of, something must occur that changes the identity of that input. That change is represented by one or more processes within the model content. It is in this way, at a very coarse level of consideration, that the modeled operation (the content) is a function. It accepts some input, and produces some output.

Objects – Persistent Components. There are parts of a system that may be identified within a model that maintain identity over time, and these are objects (Corcho and Gomez-Perez, 2000). The term object here is not to be confused with the computer science term object. In having an understanding of a system and its persistent elements, we can assume that they will remain stable, unless operated on by something that is within the system (a modal change) or something from without the system (extra-modal change). It is reasonable to expect this persistence (Chandrasekaran, et al., 1999). The persistence of these elements is what allows for the meaningful capture of them into artifacts (Niles and Pease, 2001). A method to distinguish objects from similar objects is the identification of the various qualitative and quantitative attributions that identifies the object (Guarino and Welty, 2000).

The model describes that meaning of what the system it describes is, and what it does. The model uses symbols to express that meaning, and those symbols are in at least two different categories – objects and processes (Whitehead, 1978). It is clear that ontologically objects are things that *are*, that is – they exist. Processes, ontologically, are things that *occur* – either at a single point, or over some time. While it may be enough for now to state that objects are the portions of a model that persist in their existence – the

parts, components, etc., what the meaning a process conveys is not yet clear. A definition of process, and what a Process component can do, is required.

Processes – Dynamic Behavior Components. If there is change – that is, objects changing because of something represented within the model – then that change is a dynamic element of the model. In order to derive a definition for what that dynamic element is when it is considered as part of a model, it is required to look to the literature for definition. To simplify things, although the word has not been defined yet in the dissertation, the word process will be used for that dynamic element. In order to see what this is as the literature is guiding us to a formal description of what the model consists of, defining the word process is necessary. So far, whatever it is that changes within a model – and it has just been shown that a dynamic model is one that is responsible for some change of the input to the model into the output from the model – has had the method of enacting that change called either the dynamic aspect, or dynamic element of the model – or alternatively it has been referred to "what it is about the model that changes". From the National Institute of Standards and Technology the formulation of the IDEF system descriptions came about (NIST, 1993), with the IDEF0, and subsequent definitions of a process. Within that work, the process is a function that changes input into output, again with some constraining mechanism, and employing some resources. Within Dori (2002) we find that Dori, in his treatment of systems descriptions using an equal combination of process and objects, defines a process as something that has a change on something else in the system, but that objects can be related as effects on the process, or something to be consumed by the process. Within Sowa (2000) we find that Sowa, in his treatment of the subject (more later in this section) refers to a process as something that has a starting and

a stopping point and has some change that takes place in between. He goes further to provide a categorization of different sorts of processes, dividing them up into first continuous processes, and then discrete processes and so on. That this is possible shows that it will be something the dissertation's treatment of the subject will have to address later on. For the remainder of the dissertation this act of change will be attributed to a model component called a process. A process is defined in layman's language, in the sense meant here, as "a continuous action, operation, or series of changes taking place in a definite manner" from the website dictionary.com which amalgams definitions from a number of different popular lay dictionaries. As the dissertation has, so far, referred to the processes of a model as change, this meets the layman's language test of what the term means. In more specific and formal terms, however, there are several sources that treat with the word.

From these definitions, then, the thing that remains through them all is the defining identity that "a process is responsible for some transformation or change". The characteristics of the process will be described further on in this chapter. When considering the system as a whole, then, each process is a marker of the system being in one state, and then the transformation of the process occurs, and then the system is in another state. If the process exists over time, then the system is in a dynamic state during that time – neither the pre-transformation state, nor the post-transformation state. Here, when the state of the system is referred to, the same sense is implied as for the definition for "system state" from earlier (p. 71).

Relations – Associating Components. With regard to a model, then it is seen that an object is something that retains a state of identity within that model. A process is

something that is a marker between two different states of the model. In the case of the object, the word state is the identity of the object. In the second case of the process, the word state is the identity of the model. Since we have seen that the model consists of objects and processes, then a process is the change to one or more components that make up the content of the model. The component(s) that the process affects must be identified by the model, to express information about the system.

As the content is some collection of objects and processes, and processes occur that mark a change to the state of one or more of these, then there must be a way of associating a process with what it is changing. That association indicates a third entity that is part of the model's content. It is referred to in the computer science literature (Codd, 1970, 1974) as a relation. This view was refined, and given further consideration (as the basis for relational database management systems) in Chen (1976). The reliance on a relation is identified as a key to representing information in all of the fourteen (or more, considering SysML) diagramming techniques presented in both UML and SysML (OMG, 2002, 2010). Finally, it is represented within both the knowledge representation literature (Sowa, 2000) as well as the systems modeling literature (Dori, 2002) as the connection between objects and processes. In a static model, such as a data model or an entity-relation model, the relation is used to associate together component objects into new aggregate objects that consist of one or more mereological parts. In a dynamic model, as we have changes to the components of the model, the association of processes with what they are changing also becomes the subject of relations. This has been identified within both Sowa (2000) and Dori (2002). It is quite possible that the identification of components that work at different times either individually, or combined

into a larger element is part of the meaning of a system. Because of this, relations are also required to associate together not only objects that have a mereological relationship to a larger construct, but also processes that may be related to one another in a certain sense, but also must be described separately. This new component, then, the relation is defined as "the means of associating together other parts of the model".

4.2 DEVELOPING A FORMAL METHOD

Following the preceding presentation of findings from the literature on what the components of a model can be, all of the required investigation has been done in order to assemble these into a formal method that can serve to answer the research question. A formal statement of the basic components follows (p. 130) and then this is accompanied with a formal statement of the defining qualities for those components (p. 131). This is then followed up by the formal method itself, expressed as a series of axioms and formal statements (p.133). The expression of the formal method, upon reflection of the entire system together, presents several implications and these are then presented as corollaries to the axioms, bringing out some of the more apparent (or perhaps, more useful) implications (p. 143).

Objects, Processes and Relations. To begin with, the three identified definitions of object, process and relation components are these:

- Object component
- Process component
- Relation component

Based on the definition given for a component (Definition 3) and model (Definition 1), each of these is expected to be an identifiable portion of the model that

fulfills some role in expressing the information about the system that the model is describing.

Definition 4: An Object component is a model component that has continued existence.

The Object component is a part of the model that is representing some part of the system that will retain its identity (have continued existence), until that identity is altered by some other component.

Definition 5: A Process component is a model component that is responsible for describing change or transformation.

As has been shown in the preceding section defining process, there are dynamic aspects to some systems, and therefore models of those systems must represent that change. A Process component is the component that captures that information in a model.

Definition 6: A Relation component is a model component that is responsible for associating other components.

The most immediately apparent reason for associating together components comes from the definitions alone – Object components do not change identity until operated on by some other component. Process components are responsible for some change. In order to represent within the model which Process is affecting an Object, association of the two is required, and the component that describes it is the Relation. However, other associations are also possible, such as what has been shown already in defining what a Relation is.

Defining Qualities. Although the Defining Qualities are described here, with emphasis on their structural role in the OPR system, their functional role deserves more exposition. Objects are, as described herein, loci of identity. This matches with the earlier definition given of Objects as a component of the model that has continued existence. They are the things that can be observed independently within the Model, with identity separate from all other components. Giving them definition, however, are their Attributes. These give the Objects their qualitative and quantitative distinction from other objects.

Definition 7: Attributes are defining qualities for Objects, and grant them qualitative and quantitative distinction from other objects.

This is directly corroborated by the literature on ontological views of domains (Chandrasekaran, et al. 1999), and the language of that community in defining what an object is (Guarino and Welty, 2000).

Processes are, as described herein, the descriptions of changes within the Model. What is being changed by the Process is not limited, nor are there other means within the model to address change. The defining qualities of Processes in the formal method described here are referred to as Characteristics. These are necessary to identify the behavioral nature of the Process, and to distinguish one Process from another (Sowa, 2000). These define for the Process what it changes, and the existential definition of when the change occurs, and how it takes place. That these distinctions between Processes exist, and can be captured is shown within (Whitehead 1978)).

Definition 8: Characteristics are defining qualities for Processes, and they describe the behavior of the Process as well as providing qualitative and quantitative distinction from other Processes.

Relations are the means for associating together components. Although association between components may be in effect, there are likely to be conditions to that association, and that is what the Defining Qualities of Relations define. This is addressed in the literature on the philosophy of information representation (Smith and Grenon, 2004) and also in the literature of computer science, in how entities are related to each other (Codd, 1970, 1974), and refined in Chen (1976). These qualities are specifically granted identity in the formal method presented in this dissertation,and will be called Rules. The literature is clear that Processes are affecting other components (Whitehead, 1978); to indicate this, the relation between the two is necessary to identify (Sowa, 2000). The role of associating together mereological related components (such as objects that make up an aggregate object, or processes that make up an aggregate process) is also served by Relations, and may equally be affected by the Rules that define the nature of the association. This association of mereological parts into an aggregate whole is exhibited in the computer science literature on relations Codd (1970, 1974) and Chen (1976).

Definition 9: Rules are the defining qualities for Relations, they serve to identify the nature of association the Relation is making, and to provide qualitative and quantitative identity to the Relation.

It is interesting that while all of the components share the stated feature of Objects that they are expected to retain identity (absent some change brought about by a Process

component), that the other two components (Processes and Relations) are reliant for their distinguishing definition on the presence of other components. Frege's (1982) study on the Relations between a changing quality and the identity it affects illustrates the need for this association, and the requirement for a Relation component. This is made clear in modern literature discussing the representation of a process based system (Haller et al., 2006). A Relation that does not have defining characteristics identifying its nature within a temporal framework is meaningless in a system that exhibits change, as a dynamic system must.

OPR Formal Method. This entire view of objects, processes and relations as the elements that make up the content of a model is corroborated up by the literature. Each of these exists in order to describe the role of a particular element of a model, and should be examined with more precise language, in order to form a complete understanding of what a model is composed of. What follows is a synthesis of the views from the literature presented in the preceding section. They offer up a definitional view of what the components of a model are, and how they work together. They are presented here as a formal description, albeit in an elementary fashion. The method of presentation is by indexed outline, with statements presented as indexed entries, occasionally resulting in axioms that formally capture the nature of the statements. Throughout the remainder of the dissertation, whenever the specific named items from this formal method are addressed, either in formula, diagram or text they will be capitalized. Those terms include: Model, Sub-model, Input, Output, Object, Process, Relation, Defining Quality, Identity Quality, Attribute, Characteristic, Rule, and Value. If these terms are

encountered in capitalized form, then they can be understood to be referring to the specific definitions given for them in the following formal description.

1. A Model is a representation of a system (see Definition 1).

    a. A Model has content describing the behavior of the represented system.

    b. A Model's content describes the transformation of some Input into some Output as the operation of the represented system.

    Let Z be a system.

    Let M be a Model.

    Let Z(M) be the Model of System Z.

    Let Φ be the contents of the Model.

    Let $\underline{M}$ be the modeling relationship, showing that Model's contents are related to a System that the Model Represents.

    Axiom 1: There exists for every Model, some content that is related to the system it represents.

    $\forall M, \underline{M}(\Phi(M),Z(M))$ (eq. 1)

    c. A simulator may exist that implements the model.

    d. A simulation is an instantiation of the simulator, accepting specific input, and then producing specific output (see Definition 2).

    Let α be the Input to a Model, as it is implemented by the simulator.

    Let ω be the Output of a Model, as it is implemented by the simulator.

    Let S be a simulator based on a model. SM is a particular simulator based on the particular model, M.

Let ξ be a function representing simulation; ξS being the simulation of the simulator S, accepting Input to a model, and Contents of a model, and producing Output of a model.

Axiom 2: For every Model, there exists a simulation such that the content of the Model describes how the Input to a simulator based on that model would transform into the Output that such a simulator would produce.

$\forall M \exists SM, \xi S(\alpha,\Phi(M)) = \omega$ (eq. 2)

2. A Model's content is comprised of components.

    a. There exist three types of components; Objects, Processes and Relations.

    Let O be the set of all possible Objects, O{o1, o2, o3, ...}

    Let P be the set of all possible Processes, P{p1, p2, p3, ...}

    Let R be the set of all possible Relations, R{r1, r2, r3, ...}

    Let Ω be the set of all component sets, Ω{O, P, R}

    Let Ω_M be a subset of Ω

    Let the operation $\rightarrow$ indicate that the pre-term is composed of the post-term.

    Axiom 3: The content of each Model is comprised of some (non-empty) subset of all these possible components.

    $\forall M (\Phi(M) \rightarrow \Omega_M(M))$ (eq. 3)

    b. Each component is a non-empty set of some Defining Qualities.

    Let Θ be the relationship between a component and its Defining Qualities.

        i. Objects are sets of Attributes.

            1. Attributes may have associated qualitative or quantitative Values.

2. An Attribute may appear in any number of Objects.
3. Each particular Object-Attribute pairing has a unique identity.
4. The Attributes and/or Attribute-Values of an Object will not change, unless acted on by an outside component.
5. An Object describes a thing in the model that can be considered in isolation from all else in the Model.

Let A be the set of all possible Attributes, A{a1, a2, a3, ...}

Let A_O be a subset of A

Axiom 4: Every Object O is defined by some set of Attributes.

$\forall O \; \exists A_O \Theta(O, A_O)$ (eq. 4)

ii. Processes are sets of Characteristics.

1. Characteristics may have associated qualitative or quantitative Values.
2. A Characteristic may appear in any number of Processes.
3. Each particular Process-Characteristic pairing has a unique identity.
4. A Process describes a change to a component in the model.

Let C be the set of all possible Characteristics, C{c1, c2, c3, ...}

Let C_P be a subset of C

Axiom 5: Every Process P is defined by some set of Characteristics.

$\forall P \; \exists C_P \Theta(P, C_P)$ (eq. 5)

iii. Relations have Rules.

1. Rules may have associated qualitative or quantitative Values.

2. A Rule may appear in any number of Relations.
3. Each particular Relation-Rule pairing has a unique identity.
4. A Relation describes the association of two or more components and/or Defining Qualities.

Let Γbe the set of all possible Rules, $\Gamma\{\gamma 1, \gamma 2, \gamma 3, \ldots\}$

Let Γ_R be a subset of Γ

Axiom 6: Every Relation R is defined by some set of Rules.

$\forall R\ \exists \Gamma_R \Theta(R, \Gamma_R)$ (eq. 6)

iv. There is a set of Defining Qualities for a model

1. Each unique pairing of a component and a Defining Quality for a Model M is a member of the set Q_M.
2. This is the set of all Defining Qualities (A, C, or Γ) that are associated with a particular member (O,P, or R) of set Ω_M. (see eq. 3)
3. Each Value that is associated with a member of Q_M exists in the set V_M.
4. The first Defining Quality for each component is the Identity Quality that has as a Value, the identity of the component.

Definition 10: Defining Qualities are the means for expressing the meaning of what a Component represents within the model. A Defining Quality may have an associated Value with it, when appropriate, providing parameterization for that aspect of the Component's meaning.

3. Input to and Output from the Model are sets of Defining Qualities

a. The collected Defining Qualities of all of the components in the Model (as well as any Values that may be associated with those Defining Qualities) at the beginning of a simulation are together termed the Input to that simulation based on that Model.
    i. Not all Defining Qualities have paired Values.
    ii. For the Defining Qualities that do have paired Values, and that are represented in a simulator based on that model at the initialization of a simulation, then all of those Defining Quality/Value pairings would represent the Input to the simulation.
b. The Output from the simulator is some subset of all of the Defining Qualities of all the components in the model (as well as any Values that may be associated with those Defining Qualities) at the time the Output is produced by the simulator, during the particular simulation resulting from a particular Input.
c. Output can be produced once the simulation halts, or at any point during the instantiated implementation of the simulation.
d. Both Input and Output may be empty sets.

Corollary Observations. Consideration of this brief formalism will show that the three types of components can be combined in order to show a broad variety of systems and their activities. It covers the input, output and content discussed earlier, but also makes provisions for describing space, time, multiple references for either, entities and their behavior/activities within the chosen framework of space and time. It does not address the elements of the model captured in Robinson (2008) defined earlier as

describing the character of the model – its assumptions, motivation, and operational constraints. These have been addressed in King and Turnitsa (2008), and proven in King (2009) and have been shown to be outside of the components of a model.

A minimal Model – one that is only a depiction of a non-changing Object – is one that has contents consisting of exactly one component – a singleton Object.

Corollary 1: The simplest Model is one consisting of a single Object.

$\exists M_{simple}\ (\Phi M_{simple} = o_n)$ (eq. 7)

This Model does not exhibit any change, either in definition or in time. It is simply a description of an Object, without reference to or consideration of time and space (each of which require other components to be in the model). From the definitional statements earlier, the Object would require at least one Defining Quality, which is the Attribute known as the Identity Quality.

Time in a model is represented by a set of three related components – an Object, a Process, and a Relation associating the two of them. Together the three of these components can make up a simple dynamic model called "time". For convenience, we can refer to the object as the "time Object", and the process as the "time Process". The Relationship can be called the "temporal Relation". Together these three components represent a common example of the simplest dynamic model that can exist – one with one of each of the components.

Corollary 2: The simplest dynamic Model is one consisting of three components – an Object, a Process, and a Relation.

$\exists M_{SimDyn}\ (\Phi M_{SimDyn} = \{o_1, p_1, r_1\})$ (eq. 8)

When the three components of M_{SimDyn} are those that are suggested above (the "time Object", the "time Process", and the "temporal Relation") then it can be called M_{Time}.

If a model has some conception of time as part of it, then it incorporates M_{Time} as a Sub-model. A Sub-model is portion of a Model that can be conceived of in isolation from the rest of the Model. The Sub-model cannot have any of its Defining Qualities have definitional Relations to the main model. An example of a sub model from the literature would be an atomic DEVS component that is part of a coupled DEVS specification (Wainer, 2009). A definitional Relation is one where the Value of the Defining Quality derives from some other Defining Quality, through a Relation.

Corollary 3: A Sub-model does not have any definitional Relations for any of its components to other components outside of the Sub-model.

Other components outside the Sub-model may be reliant on the Sub-model for definition (for example – a model that has processes that occur at a certain time, may have a relationship between those processes and the time Sub-model), but not the other way.

Let sM be a Sub-model of Model M

The contents of M are Ω_M (as per eq. 3)

The contents of sM are a subset of Ω_M, or Ω_{sM}

The contents of M that are not contents of sM are $\Omega_{\sim sM}$

Corollary 4: If there is a Relation, r1 that exists between some component of Ω_{sM} and some component of $\Omega_{\sim sM}$ then r1 can only be definitional from $\Omega_{sM} \rightarrow \Omega_{\sim sM}$. The Relation r1 cannot be definitional from $\Omega_{\sim sM} \rightarrow \Omega_{sM}$.

Although each component is a non-empty set of one or more defining qualities, and each defining quality may have an associated value, the value does not have to be a singleton value. It can be a range, or a set or some other numeric construct. This allows for components that are types for other components to exist. A component that is an instance of another component (which itself is a type component) has all Defining Qualities that the type component has. It may have additional Defining Qualities, or it may have separate Values for its Defining Qualities, but in some way it differs from the type component. Correspondingly, sibling components, that is, components that are all instances of the same type component, will all have some difference from each other, whether it is additional Defining Qualities, or simply different values for the same set of Defining Qualities, or some combination of these two. The Relation component can be used to associate components that share a type-instance relation. This would be a member-of Relation.

Let R_{Type} be a relation between 2 or more members of the set Ω_M

Let the relation R_{Type} distinguish the type component to the instance component(s).

Corollary 5: The existence of types and instances of components are so indicated by the R_{Type} relation.

The possible range for the Defining Quality Values of the instance components can be defined as Values of the similar Defining Qualities of the type component. As the corresponding Values of a Defining Quality are not required to be a single value (2.b.iv), but can be a range or even a set, this can be the case for a component that serves as a type for other components. The defining qualities from set Q_M that are members of the type

component may have their values either not defined or defined as a range. The instance component will then have all of the same set of Defining Qualities as the type component they are associated with through R_{Type} but as pointed out, may have additional Defining Qualities, or may just differ in the Values assigned to the Defining Qualities it shares with the type component.

Once declared within the Model, the components of that model Ω_M, and all of their Defining Qualities, and associated values, do not change unless acted upon by a Process – the components of the Model that are responsible for change. This is only done, however, if there is a Relation associating the Process with what is being changed. As in all cases, the Relation may be given further qualitative and quantitative definition as to when it applies by their Defining Qualities, the Rules. Rules may be a qualifier on when the Relation is to be allowed, that is to say, the Rule may be a subjective statement, based on some conditions in the Model (such as, Defining Quality DQ_1 having its Value set to a specific quantity...). This will be referred to as a subjective Rule for the Relation. The Relation associates the defined components only when the conditions of the subjective Rule are evaluable to true. Only when the subjective Rule evaluates to True is the Relation considered to be associating the components and/or Defining Qualities as described.

Corollary 6: There are subjective Rules, a subset of Γ, that describe a subjective truth condition for the Relation they are paired with to be in effect.

Finally, from Axiom 2 it can be seen that the Simulation function, ξ, that connects some specific Input with a specific Simulator will produce some specific Output. If the Simulator in question is a Turing compliant construct, such as a computer program for a

digital computer, then recreating the same Input will necessarily create the same Output. What this shows is that everything that is subjective to a particular Simulation, of a particular Simulator, is necessarily part of the Input. That means the choices (seeds) being followed for stochastic events, and also for associated Inputs (such as a human interaction or from a separate simulator based on a separate model, connected via federation). All of this is necessarily considered as part of the Input to the Simulator, but when it is recreated, the same Output can be expected, meaning it is the same Simulation function. If the Input is varied, then the Output may or may not be the same, but it is a different Simulation function.

Corollary 7: Given the same Simulator S_n and the same Input α_n, there will always be the same Output ω_n, when the Simulator is a Turing compliant construct. A Simulator of this type can be considered a formal Simulator.

4.3 EVALUATION OF THE FORMAL METHOD

The formal method presented here, along with its corollary observations is a rational construct based on the findings of a second literature review concentrating on what the literature has to say about the components of a model. In order to evaluate the usefulness of this formal method, it should be applied against several CMTs, as that is its intended use, and measured to see if it can adequately explain the definition and role of each of the elements in those CMTs. Examination of the following application will be done in Chapter 5.

The next chapter will deal with this evaluation. It will apply the OPR formal method definitions to four different CMTs. Those chosen will be behavioral diagram types from the UML 2 [54] suite of modeling techniques. The reason for the behavioral

diagram types is because this dissertation is interested in exploring the role of change and process in modeling techniques, in order to identify what a modeling technique must be capable of doing in order to treat the whole model itself as a degree of freedom. The behavioral diagram types, as the modeling techniques from UML intended to show the dynamic nature of a system being modeled is a strong candidate for showing the sorts of change desired to be understood and analyzed here. That those CMTs come from UML 2 is to strengthen the evaluation, because UML 2 is such a popular modeling approach for both software engineers and systems engineers.

Assessment of the Formal Method. In order to assess the OPR formal description of what constitutes a CMT, the application of the formalism to a series of CMTs is necessary. The goal at this stage is in checking OPR for internal consistency, and to see if it can be applied to a CMT and reveal what its component parts are. Once this is done, the application can be compared to the measures of merit (Chapter 3) to evaluate the formal method, and its application, to determine if it addresses the research question. Results of the comparison to measures of merit are discussed later (Chapter 5). As suitable candidates for this effort, the Unified Modeling Language (UML) family of modeling types will be selected from. The UML family consists of a number of different modeling techniques, divided up in to two broad groups. The first of those groups are the structure diagrams, which are models that identify the static structure of the elements of the system being modeled (OMG, 2002). These include such diagrams as the class diagram, the deployment diagram and so on. The second group includes the behavior diagrams, which are diagrams showing the behavioral aspects of the system being

modeled. As the dissertation is interested in processes, the second group is of interest here, and will be used for the evaluation of the OPR formal description technique.

The behavior diagrams of UML 2 include the activity diagram, the use-case diagram, the state diagram (very closely related to the activity diagram), and a subclass of behavior diagrams known as the interaction diagrams, which are the sequence diagram, interaction-overview diagram, timing diagram and communication diagram. The communication diagram was known as the collaboration diagram, prior to UML 2. With the beta version of UML 2.4 being reviewed at the time of this dissertation being written, the features of that extension of the language will not be part of this evaluation, only up through UML 2.3. In this evaluation, the activity diagram, the use-case diagram, the sequence diagram and the communication diagram will be described, using the OPR formalism to classify each component of the diagrams, and the role that they play in modeling a system.

The motivation for making this selection is in light of two reasons.

- The first is because the research question:

"Is a formal method possible that can fully describe the components, structure and dynamic behavior of a conceptual modeling technique?"

As describing modeling techniques, and their dynamic behavior, is the topic, then choosing techniques for evaluation that are identified as specifically serving to address dynamic behavior is the first criterion. That these methods are common to a modeling language from one domain (UML for software engineering) and also another domain (SysML for systems engineering) gave this criterion additional weight.

The second criterion is that the four techniques have very similar profiles (Table 4

- System Aspects Represented by Model Techniques) given existing categorizing efforts (p. 106). Being able to identify specific differences, using the OPR formal method, when the original categorizing efforts show the similarity of the techniques, would exhibit the usefulness of the formal method for the first measure of merit (capability for compositional description).

The purpose here is to evaluate whether the OPR formalism is applicable to a number of different CMTs, and whether the essential components of those CMTs can all be addressed using the OPR formalism.

OPR Assessment. In order to apply the Object-Process-Relation formal, it is necessary to map each of these components from the four selected CMTs to OPR, as well as their intended roles. This is not a formal validation, just an assessment to see if each of the identified elements of the CMTs can be described using the OPR formal method. The evaluation is to determine if OPR is sufficiently capable of definite and unambiguous role identification of the various components of the chosen CMTs. In this case the CMTs are four of the techniques designed to capture dynamic systems, from one of the currently most popular suites of modeling tools (UML 2) in use by both software engineers and system engineers (in the related form of SysML).

Activity Diagram and OPR. The components of the activity diagram from UML 2 are identified as follows, with identification of which areas within OPR that identifies their role for the activity diagram CMT. First a textual identification of each component is given, and following that, a table summarizing the whole activity diagram.

Activities within the activity diagram satisfy the definition of Processes from OPR. The reason for this assessment and not the other possibility (that they are Objects from OPR) is because (Definitional Statement 2.b.ii.4 from the formal description in Chapter 4) Processes describe a change to a component of the model. Whenever an activity is encountered in an activity diagram, the current "state" or focus of the model is changing. At a minimum, the activity represents change in the state of the model. It is implied from the UML 2 literature that there are things going on inside the model – changes to objects, through methods etc. – but these are not made explicit within the activity diagram. What is made explicit, however, is the change in current "state" of the model. This information would be captured in the Characteristics of the Process, as identified in (Definitional Statement 2.b.ii). Sub-activities, from the activity diagram are therefore also Processes. Their association with other Processes would be defined both by their Characteristics (Definitional Statement 2.b.ii), and that association would be marked by a Relation (Definitional Statement 2.b.iii), but this is left to the arrows of the activity diagram, as addressed below.

Work-flow terminators in an activity diagram satisfy the definition of two different things within OPR. In the case of initialization of work-flow, they represent Input to the identified activity (which is minimally a Process). By OPR this is some set of Defining Qualities (with their associated Values) for the Process and whatever other OPR components are related. While this will include the Characteristics of the Process, it will also include the Defining Qualities of other related components. This is as per (Definitional Statement 3.a) in the formal definition of OPR. In the case of final halting of the work-flow, they represent the Output from the identified last activity (which is,

again, a Process, but may have other associated components). Within the language of OPR this represents the final Process halting, and then producing Output, or some set of Defining Qualities (with their associated Values), as per (Definitional Statement 3.b) in the formal definition of OPR.

Signals are the association (by Relation, 2.b.iii.4) of the outcome of a Process (2.b.ii) with the input into another Process, which is part of the activity that the signal is linked to in the activity diagram. As Output and Input are Defining Qualities, which of these within the model that the Processes in question are affecting, or the Relations are associating, is to be defined by the Characteristics of the Processes, and the Rules of the Relations, in question.

Timing signals are as signals above, but they are waiting for a time-spanning Process to reach a certain amount of passed time as the change the Process is effecting. Again, from the OPR definition, this is to be defined at the Process within its Defining Qualities.

Object flows are the movement of objects within the system modeled by the activity diagram, from one activity to another. As they are not, then, relying on any one activity for their definition, but keep it throughout the model, absent an activity affecting it, it satisfies the OPR definition of an Object, in particular(Definitional Statements 2.b.i.4, and 2.b.i.5). As the identity of the object within the activity diagram remains the same, even though activities are operating on that object, there must be some parameterization of the object that is being affected; this follows the OPR definition of Attributes and Attribute Values (the Defining Qualities of Objects), as per (Definitional Statements 2.b.i.1 and 2.b.i.3).

Arrows within an activity diagram represent the temporal sequencing of the activities within the system that the activity diagram is modeling. As the activities are defined by, at minimum, one Process from OPR, then this temporal placement of the Processes, with relation to one another, should be defined by the Characteristics (Definitional Statements 2.b, 2.b.ii.1) of the Processes – that timing is part of the Defining Qualities (Definitional Statements 2.b.ii). The association of the Processes that is represented by the arrows also maps to the role of the Relations from OPR. So we have the Defining Qualities of Processes defining the temporal relation to the model, and the association of Processes with each other based on Relations.

Given that we have identified that the temporal relations of activities will be handled by the Defining Qualities (Characteristics) of Processes (Definitional Statements 2.b.ii.1) within OPR, the thick lines in an activity diagram that show activity concurrency can be explained via the Defining Qualities and their Values of OPR, in the Processes that the Activities map to.

Finally, the decision points that allow for parameterized branching of temporal association of activities within the activity diagram can be seen to be a product of having subjective Rules for the Relations that OPR would represent the arrows as. The Rules are the Defining Quality of Relations, and can be subject to certain conditions being true for the association of the Relation to hold true. This is as per (Definitional Statements 2.b.iii.1 and 2.b.iii.4) of the OPR formal description.

Table 5

Activity Diagram elements Described as OPR Components

Activity diagram elements	OPR Definitional Statements
activities, sub-activities	Processes (2.b.ii)
work-flow terminators, initialization	Process Characteristics (2.b.ii); Defining Qualities of components being initialized (3.a)
work-flow terminators, halting	Process Characteristics (2.b.ii); Defining Qualities of components being affected by the halting Process (3.b).
signals, events	Processes (2.b.ii)
timingsignals	Processes (2.b.ii)
objectflow	Objects (2.b.i)
arrows	Process Characteristics (2.b.ii); Relations (2.b.iii)
concurrency lines	coordination of Process Characteristics (2.b.ii)
decision points	Relations and Rules (2.b.iii)

As an example application of this classification to an actual activity diagram, developed to render some knowledge about a system, the model activity diagram from Tolk, et al., 2008) is presented below (*Figure 24*. Example Activity Diagram). As can be seen, the activities, arrows, and workflow terminators (both initialization and halting) are present in the example. Of note, also, are the dotted lines that identify different portions of the overall system as belong to some identifiable mode. These modes, in this example are "mounted movement", "dismounted movement", "fire elements" and "urban warfare"

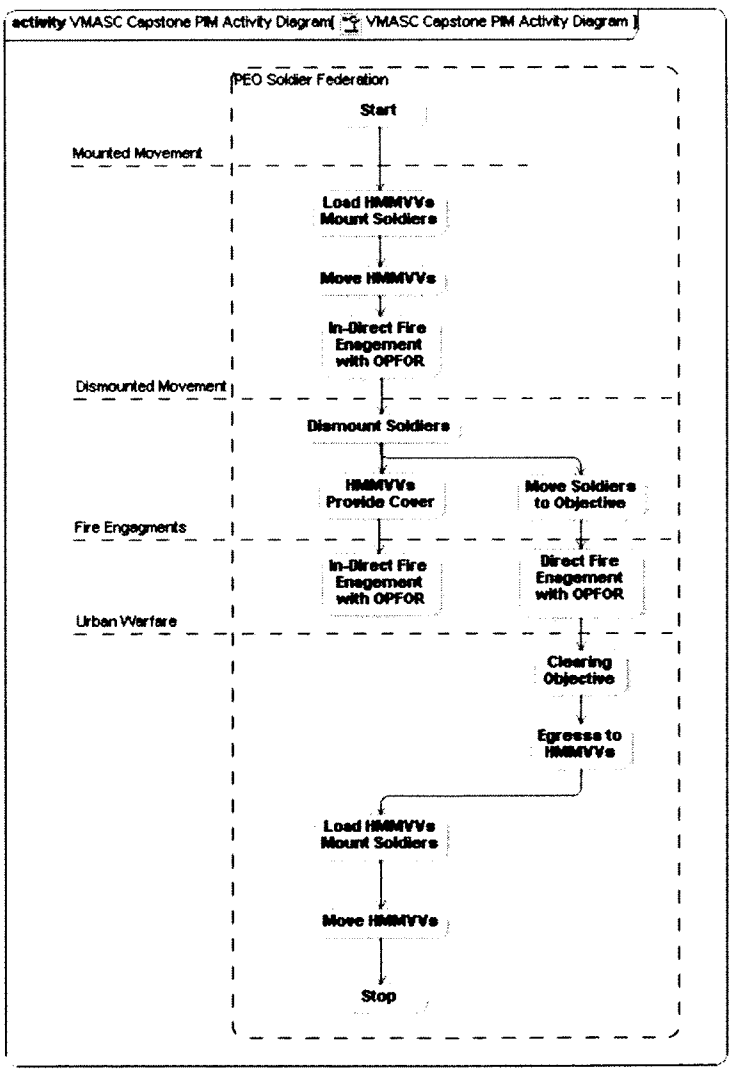

Figure 24. **Example Activity Diagram**

– these are each different modes that the system whose activity is being described (the system is infantry engagement in an urban environment). The modes could be thought of as "states" following the definition given earlier for state (Welkiens, 2006). All of the Processes (Definitional Statements 2.b.ii) that are associated within a state would be (under OPR) associated with a Relations (Definitional Statements 2.b.iii), identifying the association they share (identity within a state).

Use Case Diagram and OPR. The use-case diagram of UML (OMG, 2002) is a very useful diagramming technique, to model how an entity, either human or another system, makes use of part of the system being modeled. The intended use of the system, or part of the system, that the employing entity makes use of is called the use-case. As described in the section above about the elements of the use-case diagram, there are only a few elements; however one of those elements, the actor, is actually outside the formal boundaries of what the system entails. Also, as mentioned, it is common that several actor/use-case combinations will actually involve the same entity – either the same human or the same outside system – but in each case, that entity is taking on a different role.

The actor, as mentioned, is not part of the system. However, in a use-case diagram, the intentionality of the actor employing some or all of the system to do achieve a goal is modeled. This means that some sequence of activity (given more detail in an activity diagram or sequence diagram, or both) will be initiated by the actor to perform steps in pursuit of a goal, and that the end result of that sequence may result in output. In light of this, the definition of an actor's interaction with a use-case, in terms of how it is defined by OPR, is as Input and Output to the Process or Processes (and associated other

components) that make up the activity or activities within the use-case. As mentioned above, this will be the Input to the identified activity (which is minimally a Process). By OPR this is some set of Defining Qualities (with their associated Values) for the Process and whatever other OPR components are related. While this will include the Characteristics of the Process, it will also include the Defining Qualities of other related components. This is as per (Definitional Statement 3.a) in the formal definition of OPR. If it is so that the use-case will result in some Output from the activity or Activities that define the steps towards the Actor's desired goal, then by OPR definition, the Output is useful, as well. This will be the Output from the identified last activity (which is, again, a Process, but may have other associated Components). Within the language of OPR this represents the final Process halting, and then producing Output, or some set of Defining Characteristics (with their associated Values), as per (Definitional Statement 3.b) in the formal definition of OPR.

If it is desired for the Actor (whether Human or some other system) to be modeled, in its own OPR terms of objects, Processes and Relations, then it is defined as its own Model (Definitional Statements 1.a, 2.a, etc.) but with regard to the system in question that is described by the main body of the use-case diagram, it can be treated as a Sub-model (Corollary 3) for Input purposes. If there is to be Output, back to the Sub-model, then it should be properly considered a separate Model altogether (as this would violate Corollary 4).

The use-cases of a use-case diagram represent some activity or a sequence of activities that are undertaken in order to pursue a goal that the actor desires. As per the activity diagram, such activities are, when defined by OPR, either a single Process, or a

series of Processes (Definitional Statement 2.b.ii), with associated other Components; the possible candidates are Objects (Definitional Statement 2.b.i) and or Relations (Definitional Statements 2.b.iii).

Links in a use-case diagram would represent the identification of the use-case with the actor that is making use of it. As we have seen that OPR defined actors in a use-case diagram are the Input and/or Output to the Processes and other components that OPR would define the use-case as, then it is reasonable to identify the Link as a catalog of the components and their Defining Qualities that would be the subject of the Input collection required to initiate the use-case (Definitional Statements 2.b and 3.a). When the Actor is modeled on its own (Definitional Statements 1.a, 2.a), or as Sub-model (Corollary 3), then the link represents the Model to Model (or to/from Sub-model) relations (but not to violate Corollary 4).

Arrows in a use-case diagram are used for convenience, when a larger use-case is conveniently divided up into two or more sub use-cases, such that each individual part can be addressed separately for other actor/use-case pairings. In the case of defining this CMT element by OPR terms, the arrow represents how Defining Qualities that are required as input into the target activity's initial Process (and associated components) come from the Defining Qualities that must be either supplied by the source activity's final Process (and associated components) or provided by the actor as new Input. This is according to (Definitional Statements 2.b, 3.a, and 3.b).

When a type-extension of an actor is identified in a use-case diagram, then it is defined, as per the OPR formal method, as the same as however the actor was treated (either as a set of Input and Output conditions; as a Sub-model, or as a completely

separate Model), but with the additional elements required to accommodate the use-cases identified for the extended type.

Table 6

Use-Case Diagram elements described as OPR Components

Use-case diagram elements	OPR Definitional Statements
actor as user	Defining Qualities for Input and/or Output (3.a, 3.b)
actor as model(optional)	Actor treated as a separate Model (1.a, 2.a)
actor as sub-model (optional)	Actor treated as a Sub-Model (1.a, 2.a, Corollary 3)
use-case	Processes (2.b.ii), also possibly Objects (2.b.i) and Relations (2.b.iii)
actor/use-caselink	Defining Qualities (2.b, 3.a, 3.b); in the optional case, Relations between the Models (2.b.iii)
use-case/use-casearrow	Defining Qualities as Output, becoming Input (2.b, 3.a, 3.b)
actor type extension	Following the OPR definition for the actor option chosen, but with additional requirements for the extension's use-cases

As an example of a use-case diagram, see (*Figure 25.* Example Use-Case Model) from Cloutier, et al., 2003) presented here. As can be seen, there is an actor which is a

sub-model from another model (from a business case model, as indicated in the diagram), which would be represented as a sub model (following Corollary 3). The three use-cases would be (at least) Processes (2.b.ii), and might themselves be sub-models. The actor/use-case links are definitely represented as Relations (Definitional Statement 2.b.iii). As can be seen from this example of (what appears to be) a fairly simple model, without having the perspective of the modeling known, it is not clear what is represented

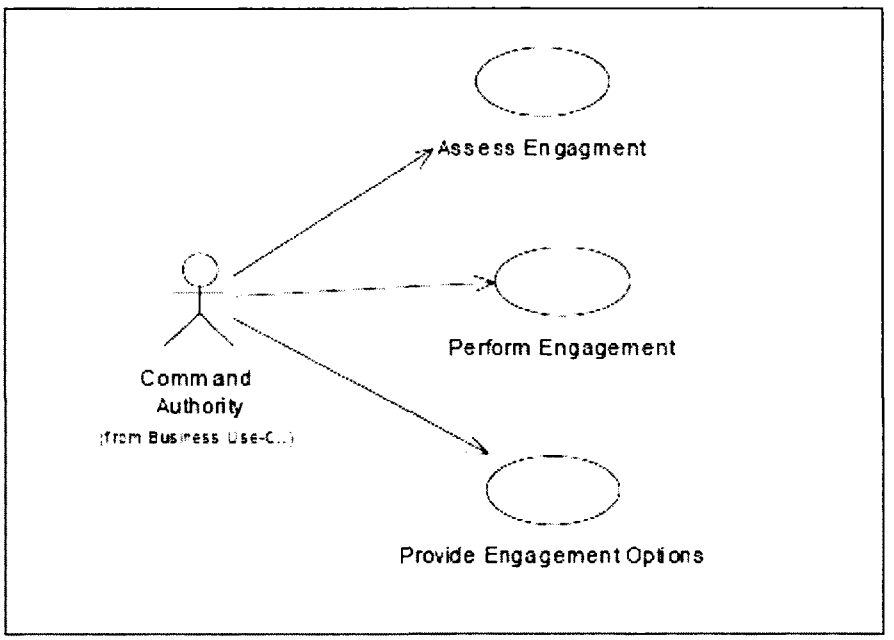

Figure 25. **Example Use-Case Model**

by the elements in this model – making it an open question as to how to represent those elements as components by the OPR formal method. In this case – the CMT being a diagramming type from UML or SysML – the case would be cleared up by having a full suite of models from which to draw the knowledge from.

Sequence Diagrams and OPR. Sequence diagrams have a number of elements that help define the information being modeled in the diagram. These elements are all defined by components from the OPR formal method. The communication diagram section that follows is designed to capture the same information as the sequence diagram, but since it views the knowledge from a different perspective (element-centric, vs. sequence-centric) the differences for OPR are interesting.

The objects of a sequence diagram follow the definition of an object from object oriented design (OOD) methodology, this is not surprising since UML is designed to support such a methodology (Booch, 1999). Though they appear in the sequence diagram as just the object, they also have (implied) all of their class-associated methods along with them. OPR defines both Objects and Processes as having their own independent existential claims, so if the implied methods of an OOD object are to be accessed by elements in a sequence diagram, then their OPR defined component will need to be defined independently. Having stated that caveat, the objects of a sequence diagram are the Objects of the OPR formal method (Definitional Statement 2.b.i), along with their Attributes and Values (Definitional Statement 2.b.i.1)

Messages and returns are, as mentioned above, identified as originating with objects in a sequence diagram, but they represent the operations of the Methods that are affiliated with the objects. As methods are change, then it is quite plain that the OPR definition of messages and returns would be Processes (Definitional Statement 2.b.ii). The fact that they are affiliated with an OOD object means that they in some way have characteristics that are associated with the object. When defining these using the OPR definitions, the Characteristics (Definitional Statement 2.b.ii.1) and their Values of the

Processes that represent these methods must be relied on to detail a lot of things concerning the operation of the Process – temporal concerns, effects, dependencies. Where these are related to other components, such as the Object they are associated with, a Relation (Definitional Statements 2.b.iii) would be involved.

Activation frames defined in OPR would be depending on the Processes that are operating within them. If the message and return pairing that is possible in an activation frame are aspects of the same OPR Process, then the definition is already taken care of in the temporal and effect Characteristics (Definitional Statement 2.b.ii.1) of the Process, but when they are separate Processes, then there will be a Relation (Definitional Statement 2.b.iii) that is associating the Characteristics of those Processes that have to do with their temporal spacing within the overall model.

Branches are handled by both the Defining Qualities (Characteristics and Values) of Processes, and also the Relations associating them. When they branch from the same originating object, at the same temporal point, then the separate Processes will be operating with the same temporal Characteristics defining their initiation, and also their operation. When the branching has some sort of parameter to choose between alternative branches, then Relations (Definitional Statement 2.b.iii), with subjective Rules (Definitional Statement 2.b.iii.1) are the OPR proscription.

Loops are also handled by situations described by the Characteristics of the Processes (Definitional Statement 2.b.ii.1) involved, when defined in OPR terms. If the loop consists of a single Process, then its temporal defining Characteristics would identifying it as repeating until a certain point is reached. When there are several Processes that iterate, and then loop, they have Relations (Definitional Statement 2.b.iii)

associating them through their temporal defining Characteristics, and then a subjective Rule for a Relation (Definitional Statement 2.b.iii.2) going from the termination of the last to the re-initialization of the first.

Interaction frames are a marker of convenience, showing which elements in a sequence diagram belong together conceptually. They would be handled within OPR through Defining Qualities (Definitional Statement 2.a) of the components involved.

Call-backs within a sequence diagram are a case where an object has a method that sends a message (making a call) back to itself. Within OPR as the Defining Qualities of all the components that would make up the OOD object (as an OPR Object – Definitional Statement 2.b.i) and its associated methods (as OPR Processes Definitional Statement 2.b.ii), the Characteristics of the Process (Definitional Statement 2.b.ii.2) that defines the call-back could be associated with any of the Defining Qualities of the OPR components (Definitional Statement 2.a) that it is affecting within the definition of the OOD object.

Table 7

Sequence Diagram elements described as OPR Components

Sequence diagram elements	OPR Definitional Statements
objects, object lifelines	Objects, Attributes, Values (2.b.i, 2.b.i.1)
messages, returns	Processes, Characteristics, Values (2.b.ii, 2.b.ii.1)
activation frames	Characteristics, Relations (2.b.ii.1, 2.b.iii)
branches	Relations, Rules, Values (2.b.iii, 2.b.iii.1)

Table 7 Continued	
loops	Characteristics, Relations, Rules (2.b.ii.1, 2.b.iii, 2.b.iii.1)
interaction frames	Defining Qualities (2.a)
call-backs	Characteristics, other Defining Qualities (2.b.ii.1, 2.a)

The diagram (*Figure 26.* Example Sequence Diagram) is presented as an example of a sequence diagram, illustrating some of the features of the CMT, and

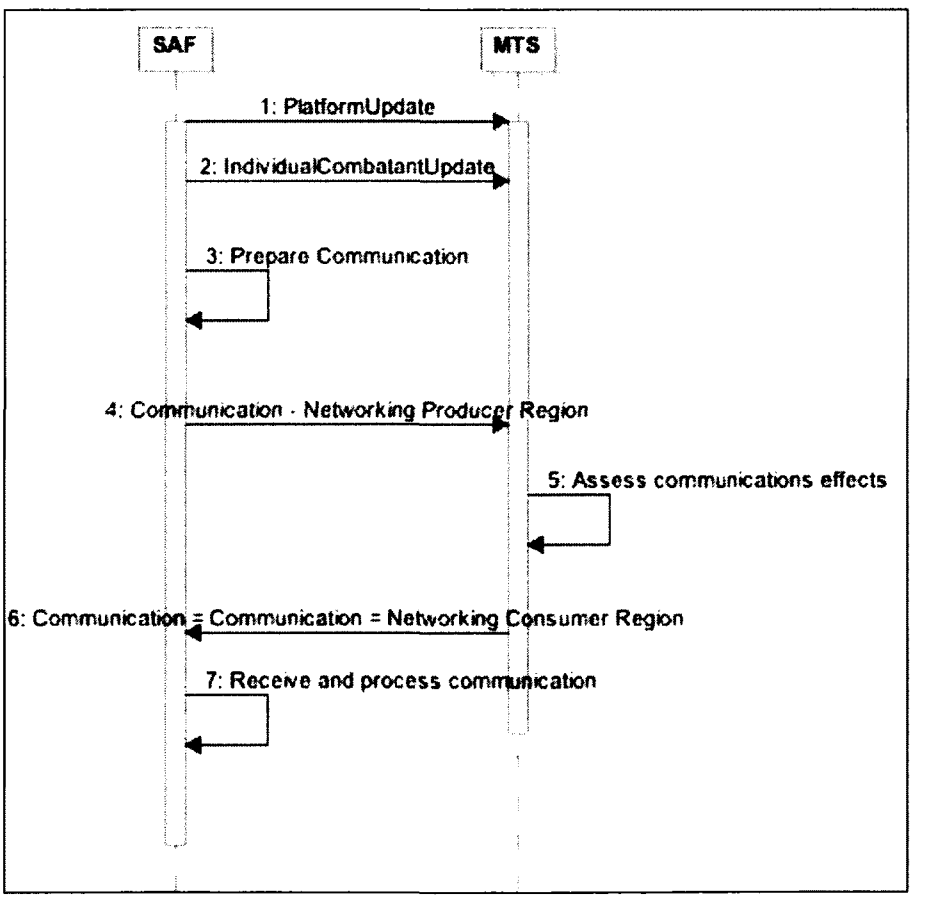

Figure 26. **Example Sequence Diagram**

illustrating how they can be represented (for comparison) by OPR components. The figure is from Kewley and Tolk (2009) and represents the sequencing of activity between two different simulation systems for infantry training. As can be seen, object lifelines are present for the two elements within the systems, which would be represented by Objects (Definitional Statement 2.b.i), Attributes (Definitional Statement 2.b.i.1) and specific Values (Definitional Statement 2.b.i.1) for those Attributes. Messages are present, which would be represented by Processes (Definitional Statement 2.b.ii), and refer to the activation frames via Relations (Definitional Statement 2.b.iii). Of interest in this particular example, are some messages (numbers 3, 5, and 7) that refer back to the activation frame they originate from. This is the perfect case for seeing that some of the identifiable Characteristics for a Process are not only when and how it originates (initialization characteristic, to be defined in 4.7.1) but also what it is in the model that is being affected by the dynamic behavior the process represents (effect characteristic, to be defined in 4.7.2).

Communication Diagram. As mentioned above, in the section on the sequence diagram, the communication diagram from UML 2 covers much of the same information. It displays information about which objects (and their associated methods) are communicating with which other objects, and also the order of such communication. The elements included in a communications diagram are simple – they include the objects that are communicating, as well as communication lines showing the messages and responses between objects and their associated methods.

The objects in a communication diagram, as with objects in other UML 2 diagrams, are OOD objects. The strong implication, of course, is that they are class

derived and so have associated methods. If strictly considering the objects from an OPR definition viewpoint, they would simply translate directly as an OPR Object, with associated Defining Qualities (Definitional Statement 2.b.i). However, as this evaluation also seeks to define the use each element of the diagrams will receive, the methods associated with the communication diagram object that are responsible for the modeled communications would be included, and as defined by OPR these would be Processes, associating Relations and their Defining Qualities (Definitional Statements 2.b.ii, 2.b.iii).

The communication links that are depicted in the communication diagram represent the methods and returns that take place. Again, as these are only available to an object through the associated methods, these would be defined by OPR to be Processes (definitional statement2.b.ii) and their Defining Qualities (notably, Characteristics and Values). The ordering of the communication steps, which provides for temporal sequencing, would be defined by OPR as part of the Characteristics of the Processes.

Table 8

Communications Diagram elements described as OPR Components

Communication diagram elements	OPR Definitional Statements
objects	Objects, Processes, Relations, Defining Qualities (2.b.i, 2.b.ii, 2.b.iii)
communication links	Processes, Relations, Defining Qualities (2.b.ii, 2.b.iii)

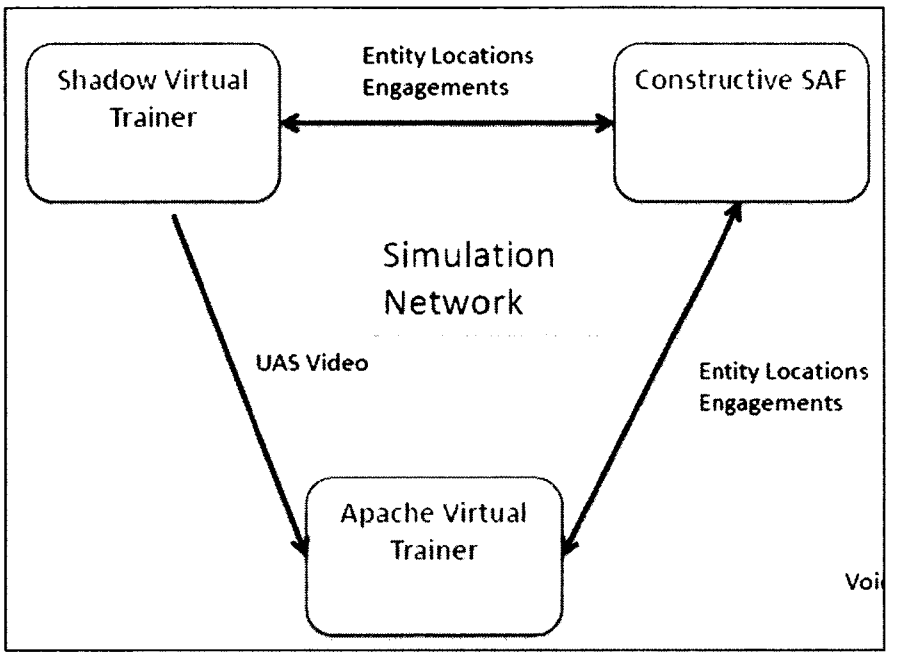

Figure 27. **Example Communications Diagram**

As an example of a communications diagram, refer to the diagram (*Figure 27.* Example Communications Diagram) which is from Kewley and Tolk (2009). As can be seen, the communicating objects ("Shadow Virtual Trainer", "Constructive SAF", and "Apache Virtual Trainer") can be seen. As defined, these would be represented by OPR as Objects (Definitional Statement 2.b.i). They are connected via communication links, which would be directly represented as Relations (Definitional Statement 2.b.iii), but they also imply that Processes associated with the Objects will be responsible for the communications (Definitional Statement 2.b.ii), as an object itself is not capable of producing change, and generating a communications event is certainly change that occurs during the operational span of the model.

Assessment Results. With the four different CMTs chosen from the UML 2 family of modeling techniques, it has been shown that every element of those four techniques – chosen because of the two criteria given in (p. 144) – can be defined using

the components of the Object-Process-Relation formal method for describing conceptual modeling techniques.

Each element of the CMTs was shown to have been addressed with the associated statements from the formal method (backed up by the axioms, definitions and corollaries of the method). By providing a look from outside any of the modeling techniques intended to be evaluated, the OPR formal method gives the ability to describe, and compare, in a neutral language the components and their definition of a modeling technique. This allows for the direct comparison of techniques, for evaluation purposes. The capability that application of this formal method has to address the "capability for compositional description" measure of merit has been satisfied.

Because the second measure of merit, the "capability for definition for dynamic content" is to investigate processes and change to models and model components, the dynamic element of OPR is of particular interest in this evaluation. While we have seen that OPR Processes, and their Defining Qualities, the Characteristics, are part of each of these CMTs, in each case the role that the Processes play is different, but there is some similarity in the role that the Characteristics play (i.e. – defining change effects that the Process introduces to the model; temporal characteristics; initialization; etc.). Because of this, the next chapter will synthesize those definitional elements into specific Characteristic definitions for Processes.

Defining the Borders of a Multi-Model. To consider how the OPR method can assist with evaluating a multi-model application, the abilities of the method to satisfy the first two measures of merit come into use. As pointed out (definitional statements 3.a, 3.b) in the sections of the formal method dealing with input to and output from a model,

these are both in terms of defining qualities that are accepted into the model, and produced to come out of the model. As with other considerations of defining qualities within OPR, the set can be a complete component (object, process or relation), if it includes the identity quality, and all other defining qualities, that make up such a component.

In the case of modeling approaches, such as UML and SysML, that make use of a variety of different modeling techniques, in order to specifically assemble a multi-model view of a system, the same holds true. In order to represent a representation of the model that captures as much information as is expressible, the border conditions (where output from one of the models serves as input to another) should be true to the OPR described elements of the modeling techniques in question. Consider (**Error! Reference source not found.**) which shows part of a UML multi-model representing a system. The use-case diagram (p. 154) has as an element a use case (Table 6), which is then represented by an activity diagram. The activity diagram (p. 146) has many activities (Table 5) that are part of it, and one of those is represented by a sequence diagram (p. 160). In each of these cases, there are conditions where the temporal (in this case, but it could also be spatial) borders between the models also serve as delineators between what the different models of the multi-model are representing about the overall system. In the use case, the beginning and end of the use case correspond to the starting and ending points of the activity diagram. In this case, the time sub-model (corollary 3) of the activity diagram model would have, as its initialization of the time object in that sub-model, a value to an attribute of that time object that is imported from the use-case. The same applies for all

other components and defining qualities that may be necessarily defined as input to one model, from the model that it borders (temporally or spatially).

Such an arrangement is tempting to think of as a series of sub-models within an overall model, but the term used here of multi-model has been chosen intentionally. While OPR could be shown to be useful, in the process listed previously, to describe the

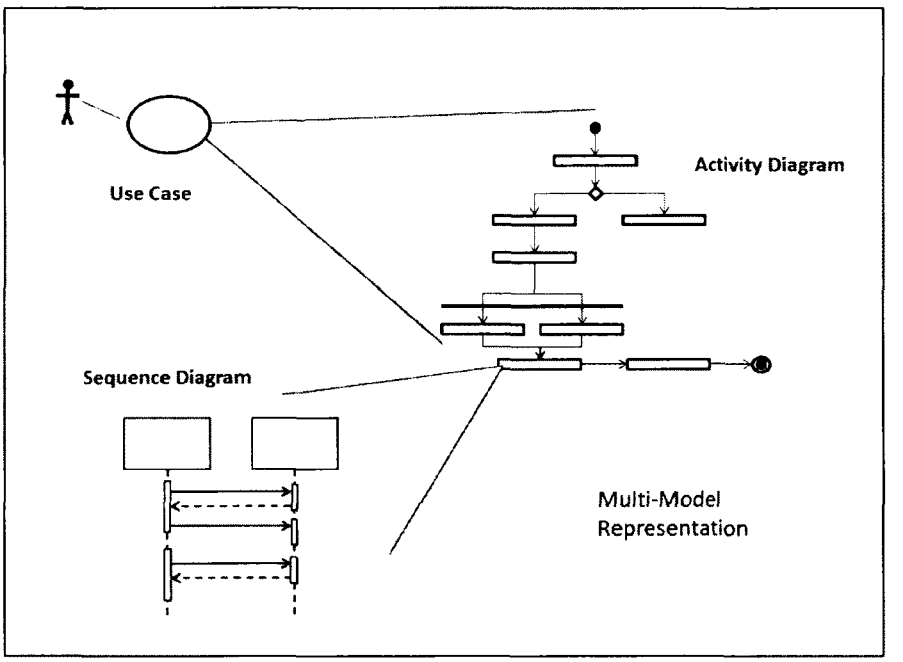

Figure 28. **Multi Model Representation**

relationships between a sub-model and a model, in the case of the multi-model, as there may be overlapping relationships between the various models, and there may be definitional dependency on the input, as it is output from another model, this violates the description of sub-model from (Corollary 3).

4.4 ENHANCING THE FORMAL METHOD

Having evaluated the OPR formal method for defining the elements of a conceptual modeling technique, the results show that, as in the case of the sample CMTs chosen, the elements of such a technique can be defined by the OPR formal method. What the evaluation also highlighted, is that the Characteristics of a Process, can themselves be given definition within the formal method. This chapter does that, by examining what the likely Characteristics of a modeled Process can be, and describing them formally in a way complimentary to the formal method (p. 128).

It has been shown that the elements of a CMT can be described as some set of components from the categories Object, Processes and Relations. It has been defined what role these components serve within the modeling technique. In all three types of components, no change to any definition is possible without that change coming from the one component that causes change – a Process. Therefore, a Process is required to have three things (perhaps more) at least, when described using the OPR terminology. Those are identity (see Definitional Statement 2.b.iv.4), a defined change to some part of the Model, and one or more associative Relations components that connect the Process with the part of the Model it is defined to change. The associative Relations are each their own defined component, however the definition of what the Process will affect in the associated component is part of the Process' Defining Qualities. The identity of the Process derives from P being a member of Ω_M. The change, as part of the Process' definition, is to be a Definitional Quality (a Characteristic by definitional statement 2.b.ii of the formal method in Chapter 4). A return to the discussion from the chapter on the formal description of OPR as a function definition of a Model is needed to see what else

is required of a Process, and further identify all of the standard types of Process Defining Qualities (Characteristics). It should be pointed out that the definition of a Process from Chapter 4 (Definitional Statement 2.b.ii) and what follows in this chapter do not limit additional Defining Qualities that grant qualitative or quantitative understanding to the modeled Process.

A note on terminology is warranted here. As with the beginning of the exposition on the formal method of OPR, as we explore Processes here, the various names of the identified Characteristics will be capitalized when referring to them. This is to avoid ambiguity when seeing these words in the text. The words are: Initialization, Effect, Behavior and Halting.

Process Initialization. The sequence of all changes that occur within M, during a simulation instantiation of M, must be ordered. The ordering of the sequence of changes may be relative to all other changes in the Model, or may be marked against a time Sub-model of Model M. There is nothing that requires M to have only one time Sub-model, especially in cases where there may be components in the Model that have different time references. All changes that occur within M occur as the result of a Process that is part of M. If something external to M (some undefined external agent, such as another system outside the Model, or a human operator) causes a change, then that change is considered to be an extra-modal change. All other changes (that occur because of processes defined as part of M) are modal changes. Everything that is part of the Model is part of the normal mode of that Model. This includes all of Ω_M, Q_M, and V_M.

Q_M and V_M and all of their members can change at any time point T, or over an interval T_{int} in the instantiated simulation that implements the Model M. T, or T_{int}, is

some identified value in the "time object" of a time sub-model, M_{Time}, of M. The term used when the Process takes place at a point T is that it is an "instant process". The term used when the Process takes place over an interval T_{int} is that it is an "extant process". The process component P, that is responsible for the member of Q_M and/or V_M that changes, does not necessarily need to have T (or T_{int}) defined, but it nevertheless occurs at some time.

These considerations of a time sub-model and how modal changes are ordered against it show that another of the definitional qualities of a Process component is when it occurs, in relation to some M_{Time} sub-model. As expressed already, this need not be objectively defined by the Process, but may be subjective based on some other combination of Q_M, and V_M being in a particular state. It has been asserted that the two possibilities of P occurring are either at some point T, or over some interval T_{int}. In order to be general, and encompass both of these, the Characteristic that defines when this occurs for the Process should be referred to as the Initialization point. Both time possibilities, T and T_{int}, have a beginning point. In the case of T, that is the only point it is concerned with, but with T_{int}, there is some interval that follows the beginning point.

As a Process component may have more than one Effect Characteristic that defines changes to more than one Defining Quality, when there are multiple changes defined to be part of one Process, then that Process would have a single Initialization Characteristic, but the Behavior Characteristics would define when each Effect were to take place, sometime after that Initialization Characteristic, and before the Halting Characteristic.

Process Effect. The change that a Process component defines is referred to here as the Characteristic called Effect. It will be referred to symbolically as C_E. The effect of the Process is defined as some change to one or more defining qualities within the model. This includes any member of Q_M (Defining Qualities) or the Value affiliated with that member, from V_M. So an effect can be a change to any Characteristic, Attribute or Rule – or it's associated Value. The possible changes that an Effect can include are of three types – creation, alteration and destruction. As an effect can be change to more than one Defining Quality, it can be defined as also more than one type of change. For instance, a Process might have the Effect of both creating one Attribute (or some other Defining Quality) and destroying another. When creation of a Defining Quality, it is up to the definition of the Effect to describe if a Value will be affiliated with it, and what its Value might be. When change to more than one Defining Quality is part of what the Process Effect describes, it may be that all of the changed Defining Qualities or their associated Values are from the same component, or they may be associated with more than one component from Ω_M. The way to either create a new component, or to destroy an existing component, is through its paired Identity Defining Quality. If an Identity Defining Quality is destroyed, then the component that it granted Identity to is also destroyed.

There are some observations that can be made about the Effect Characteristic. First is that although the Effect describes some change to one or more Defining Qualities in the Model, a Process itself cannot associate with another component, but he implication is that there will be a defined Relation component between the Process and whatever component or components that the Defining Qualities to be changed by the

Effect belong to. Regardless of the defined Effect, the Rules of the Relation associating the Process with the component that is to be changed by the Effect, and if that Relation is itself somehow changed this would keep the Effect from happening. In other words, there is no change allowed between a Process and another component, unless there is also a Relation established between those two components (the Process and the other). Equally, the Rules of the Relation, if they are subjective, only allow the Relation between the two components to be active if the subjective case of the Rule is satisfied (see Definitional Statement 2.b.iii and Corollary 6 from Chapter 4).

Process Behavior. In the case of an extant Process, where an effect takes place over an interval T_{int}, it is not determined that the effect should take place at a single point, T, during that interval. Because of this, an additional defining quality is required. The Characteristic that describes the Behavior of the process – that is, how the extant effect (or multiple instant or extant effects) occurs during the interval existence of an extant Process during the simulation implementation of the model – is called the Behavior Characteristic, and will be represented as C_B. As mentioned, this is only necessary when the Process is extant in nature. This Characteristic would describe the temporal Behavior of the Effect (or Effects) of the Process with reference to a particular time Sub-model of Model M. While the specific point on the "time Object" of the time Sub-model in question need not be stated, the temporal Behavior should be expressed as having a rate of progress in relation to the "time Process" of a time Sub-model. There is nothing that compels a model to assume that defined temporal Behavior (which is the Behavior Characteristic of the "time Process") will continue within a simulation that implements that model, even from modal disruptions. It is useful to consider from the literature on

time certain indicators of the state of an extant process, as seen in Haller and Oren (2006) and Haller, et al. (2006). Within systems modeling literature, this is also seen in ISO (2004). These include the possible process states:

- Active
- Suspended
- Resumed
- Cancelled
- Aborted
- Halted

When the time state of the time Sub-model is considered to be *Active* then the changes to the "time Object" by the "time Process" are proceeding according to the Behavior Characteristic of the "time Process". This means that the "temporal Relation" has its subjective Rule evaluating to "True", so that the relationship between the "time Process" and the "time Object" can proceed as normal.

When the time state of the time Sub-model is considered to be *Suspended*, then the subjective Rule of the "temporal Relation" is evaluated to "False" so that the relationship between the "time Process" and the "time Object" is not allowed, and does not proceed as the Behavior Characteristic of the "time Process" would normally dictate.

If the subjective Rule of the "temporal Relation" is restored to "True", then the Behavior Characteristic of the "time Process" begins motivating the Effect of the "time Process" on the "time Object" once again – time is *Resumed.* Operationally, there is little difference between Active and *Resumed*, however it is here because of additional

information that may be captured when a *Suspension* of Time is planned as part of the Model's operation.

Some Process that has its Initialization point defined in terms of the "time Object" being in a particular state, but the "time Object" is somehow prohibited from reaching that state can be said to be *Cancelled.* Note that *Cancelling* applies to Processes, and not necessarily the entire Model, although it could. A Process could be *cancelled* by somehow affecting the Relation between the Process's Behavior Characteristic and the "time Object" if it is defined that way, or other Processes if defined subjective to their operation.

If either the "time Process", the "time Object" or the "temporal Relation" are destroyed by some Process, then the Time Sub-model that they were part of is considered to be *Aborted.* There is little difference between the *Aborting* taking place because of destruction by a Modal cause or an Extra-Modal cause, however with a Modal cause it can be part of the Model. The Model is not aware of Extra-Modal causes.

Finally, if the "time Process" is somehow stopped from advancing the "time Object" either because it has completed its proscribed Behavior, or some modal change to the Relation between them, or some other reason – then the Time Sub-model is considered to be *Halted.* When the Time Sub-Model of a Model is *Halted* is the expected means of indicating that a Model that has its Processes based on the Time Sub-model (for Initialization, Behavior, and Halting) has reached its end.

Process Halting. Each Process will eventually end. This may be during the temporal life of a Simulation based on the Model, or it may be when that Simulation ends. If it is something that can be anticipated within the system, then the Model

describing that system should describe a Halting Characteristic. It is in form identical to an initialization Characteristic, in that it can be either objective or subjective. It is not necessary that a Process have a Halting Characteristic. The temporal behavior of when the effects of the Process take place are described by the Behavior Characteristic, and do not necessarily coincides with the Halting Characteristic. If another Process has its Initialization or Halting Characteristics defined as being subjective to the Halting of another Process, then however that Process Halts will be the indicator to the subjective Characteristics.

Formal Treatment of Characteristics. Restating what has been previously shown about the Characteristics of a Process in a formal manner similar to the formal method of defining OPR results in the following formal statements. These are presented in the same numbered statement format, and begin with (Definitional Statement 2.b.ii.5), following after the earlier (Definitional Statement 2.b.ii.4).

5. For every Process component of model M, there may be an Initialization point Characteristic, known as C_I
    a. C_I is defined as either an objective time point, or a subjective condition.
        i. An objective time point has some identified value on some Time Sub-model's "time Object"
        ii. A subjective condition is defined as some set of Defining Qualities of model M that have some specific values.
            1. The subjective condition would be associated by a Relation between the Process and the components that contain the Defining Qualities to be evaluated

2. The associating Relation (Definitional Statement 2.b.iii of Chapter 4) would have a subjective Rule (Corollary 6) that would define the requisite terms of the subjective condition.

b. The Process will occur each time the objective time point occurs or subjective conditions are satisfied during a simulation implementation of the model, M.

i. This allows for an objective point or subjective conditions to be defined in such a way that they are a set of points or conditions

ii. In the case of an objective point, if the Time Sub-Model has its "time Object" repeat the same point more than once, each time would satisfy the conditions of the objective point Initialization definition.

6. For every Process component of Model M, there may be one or more Effect Characteristics, known as C_E.

a. The Effect of a Process defines what the change the Process describes in the Model

i. The Effect is some change to one Defining Qualities

1. The change can be creation, as in creating a new Defining Quality

a. When a new Defining Quality is created, it may or may not have a Value paired with it

b. All Defining Qualities must be paired with a component

2. The change can be destruction, as in removing an existing Defining Quality

3. The change can be alteration, as in altering the Value of an existing Defining Quality

ii. If the Process has more than one Effect Characteristic, and the Effects are to more than one Defining Quality, they need not all be of the same component

b. For each Effect to take place, there will be a Relation associating it with the affected Defining Qualities

c. Effects only alter Defining Qualities

i. The Effect can create, destroy, or change all of the Defining Qualities of a component

ii. If the Identity Quality of a component is destroyed, the component itself is destroyed (Definitional Statement 2.b.iv.4)

iii. If a new Identity Quality is created by an Effect, then the new component it is paired with is correspondingly created

7. For every Process component of Model M, there may be one or more Behavior Characteristics, known as C_B.

a. The Behavior Characteristic, for an Extant Process, determines how the Effect takes place over T_{Int}

b. If there is more than one Defining Quality that is changed as an Effect of the Process, then a Behavior may be defined for each

8. For every Process component of Model M, there may be a Halting Characteristic, known as C_H.
    a. C_H is defined as either an objective time point, or a subjective condition.
        i. An objective time point has some identified value on some Time Sub-model's "time Object"
        ii. A subjective condition is defined as some set of Defining Qualities of model M that have some specific values.
            1. The subjective condition would be associated by a Relation between the Process and the components that contain the Defining Qualities to be evaluated
            2. The associating Relation (Definitional Statement 2.b.iii of Chapter 4) would then have a subjective Rule (Corollary 6) that would define the requisite terms of the subjective condition.
    b. The first time that the Halting Characteristic, either subjective or objective, occurs when the Process has already been initialized, is when the Process Halts.
    c. If the Halting Characteristic is equal to the Initialization Characteristic, then the Process is Instant, rather than Extant.

Process Corollary Observations. There are some corollary observations that accompany the formal statements about Processes that have been made (p. 138). These

are presented in the same manner, and following the numbering scheme, of the earlier corollary section. These will begin with Corollary 8.

Although it is not necessary for a Process to have an Initialization Characteristic, the implications are interesting. Within the confines of the model, the Process will never begin. If a Process is intended to be running throughout the temporal period depicted by a simulator within a particular simulation based on the Model, then the Initialization Characteristic for the Process would simply be before all other Processes take place. If there are several such Processes, then one would have that subjective Initialization Characteristic, and the others would have Initialization Characteristics of being equivalent to the first one. A Process with no identified Initialization Characteristic is one that may not occur during the operational span of a Simulator defined by the Model. However, this may be an intentional feature of the Model, and it may be intended to have an Initialization Characteristic created for it by some other Process (since the defined Effect of a Process may result in the creation of a defining quality – see Definitional Statement 2.b.ii.6.a.i.1).

Corollary 8: A Process that does not have an initialization Characteristic as part of its defining qualities will never occur unless an initialization Characteristic is created during a simulation of the Model by a Process.

While Definitional Statement 2.b.ii.5.b states that the Process will occur every time that the initialization Characteristic has a true condition – either objective or subjective – it is possible to define the Relation that would associate that Characteristic (with either the components and defining qualities that make up the subjective condition, or with the Time sub-model that are referenced in the objective condition) as having a

Rule that would keep this from happening. This is also true of subjective halting conditions, or even effects.

Corollary 9: While the defining qualities of a Process describe its behavior, as they are all associated with other components or defining qualities in the model by Relations, the Characteristics of the Process can be given control through Relation and Rule definitions.

It is not necessary for a Process to have a defined Halting Characteristic. In this case, unlike the case where there is no Initialization Characteristic, there is nothing inherently stopping such a Process from being implemented within the span of a Simulation based on the defining Model. The Process, once started, will never expire, and although the Effect Characteristic of the Process may specify a point when the alteration defined by the Effect takes place during the span of the simulation, the Process will be considered to still be "running". This has little practical effect, unless the Effect of the Process is one that may occur either repeatedly or over time. The effect it does have is on other Processes that may have any of their temporal Characteristics (Initialization, Behavior or Halting) subjectively based on the Halting of the continually "running" process.

Corollary 10: Without a defined Halting Characteristic, a Process is considered to be in existence and operating until the simulation based on the Model halts.

Since a Process may have more than one Effect Characteristic defined for it, representing changes to more than one Defining Quality, and that there may be more than one Behavior Characteristic, describing when the internally associated Effect takes place, there is a lot of interesting potential in the types of multi-effect Processes that are

available. If a Process has a number of Effects that occur one after another, it can be thought of as a serial Effect Process. If the Effects all occur simultaneously, then it can be thought of as a Parallel Process.

Corollary 11: A Process may have multiple Effects, each with their own Behavior. Each of these is a separate Defining Quality and therefore may all be the object of other Process Effects separately.

This next observation about Processes concerns the special position of time components within a Model, and should be discussed here to avoid confusion. Although the temporal Characteristics of a Process (Initialization, Halting, and Behavior) all discuss objective and subjective conditions, in reality they are all subjective. That sometimes they are based on values of either the time Object or time Process of a time Sub-model does not make that untrue – it is just that time itself is such a specially considered quality of a Model, that Process Characteristics that refer to a time defining component are given the special indicator of being objective, rather than subjective.

Corollary 12: Although temporal Characteristics of Processes are considered to be objective when they are described in reference to time components of the Model, they are actually subjective, and the time components behave in the defined manner as all other components.

The subjective nature of the temporal characteristics (Initialization, Halting, and to a certain extent, Behavior), as they are related to a time sub-model, brings out a very interesting feature that was briefly mentioned earlier (p. 165). There is no reason why a model is limited to only one time sub-model. On the surface, this may seem like a non-consequential, but curious, corollary to the OPR formal method, but consideration shows

some possible use – in modeling a referent where time is non-constant (for instance, in a model that represents elements with different subjective experiences of time, such as elements moving at near-relativistic speeds).

Corollary 13: A model has Processes whose Characteristics are subjectively related to a time sub-model, yet the model is not restricted to having only one time sub-model.

As the duration of a Process is defined by the difference between referenced points on a time sub-model between the Initialization Characteristic, and the Halting Characteristic, there is no reason why these must be separated in time. When both of the terminal Characteristics (Initialization and Halting) refer to equivalent time reference points (either in the same time sub-model, or aligned between two time sub-models), then the Process can be considered to be an *instant* Process – that is, it has an effect that occurs at a point in time, for the model. There is little requirement left for the Behavior Characteristic in this case, except to point out that all of the Effect (or Effects) occur simultaneously. Otherwise, when the Initialization Characteristic refers to a time reference point that occurs prior to the Halting Characteristic, the results are an *extant* Process. The formal method allows for seemingly aberrant behaviors, such a Process that may occur counter to the flow of time – meaning that the Initialization Characteristic occurs after the Halting Characteristic – but while such is possible with the method, no suitable example could be thought of.

Corollary 14: A Process whose Initialization and Halting Characteristics are equivalent is an instant Process, and one whose Initialization and Halting Characteristics are separated by some measureable distance in a time sub-model, is an extant Process.

The Behavior Characteristic of a Process is concerned with showing how the transformation defined by the Effect Characteristic takes place over the operational duration of an extant Process. It identifies if the Effect Characteristic is at a single point somewhere during the extant duration, or if it is spread out. If it is spread out (that is, occurring gradually) the Behavior Characteristic should define the specifics of how that takes place. It could be smoothly and uniformly over the duration, or it could be in discrete points with equilibrium between those points, or any distribution. If a Behavior is defined over an extant time, and the Process is halted, modally, prematurely, then only the portion of the defined Effect Characteristic that would have taken place prior to the interruption can be said to be defined by the model.

Corollary 15: A modal interruption to a Process will leave an extant Process's Effect Characteristic as only having introduced the change described by the Behavior Characteristic to have occurred prior to the interruption.

It is likely that this series of observations and corollaries is incomplete, and the possibility for more is introduced in the last chapter of the dissertation as future work.

4.5 ASSESSMENT OF ENHANCED FORMAL METHOD

The formal method that is described previously, and that has been evaluated,has as its intended application the identification of CMT elements, and the role that they play. It has been inductively constructed from what the literature describes as being part of a Model describing a system, but is not, as presented in this dissertation, intended to stand as its own modeling technique – other than as a model of other modeling techniques. As such, the observations and corollaries are not meant as existential claims about what might be possible within a CMT, but rather the logical demands that the OPR formal

description presents in its structure. In this way, while some of the statements or corollaries may not be followed by either a CMT or a particular application of the CMT, that is not to suggest that the CMT is not being used as it is intended, or as it is described in its own documentation – only that there may be some internal absences of definition, or inconsistencies of logical relationship among the elements of the technique.

The OPR formal method is not intended to make evaluation of a CMT in order to determine if it is "good" or not – each CMT is designed for a purpose, and the dissertation assumes that it fulfills that purpose, so from an engineering perspective it is "good". What the OPR formal method is intended to do, in identifying the functional role of modeling elements from a CMT, and granting a metric to determine if all of the information about what the element is describing are present, is to allow for a formal comparison of different modeling techniques for the engineering purposes identified earlier (p. 107).

As providing metrics for evaluating or comparing CMTs is the purpose of OPR, and investigating the role of processes within CMTs is the purpose of the dissertation, then it becomes clear that the possibility for OPR to provide specific metrics for CMT Process representing elements is what is left to develop in relation to the formal method. As we have just defined what the Characteristics of a Process are, then these should properly be part of the evaluation.

To evaluate the Process elements of one CMT against another, it is necessary to see what the present Characteristics in each can represent about a system that the CMT allows a Model to describe. In this way, a cross-CMT comparison can be made, in order to satisfy the goals set out earlier (p. 109).

By establishing the criteria compared to the four defined Characteristics we have a quantitative manner to do a comparison, and result in an unambiguous metric of the representative capability of each CMT. The possible criteria for comparing based on the four defined Characteristics are listed here.

Initialization Characteristic – Defined by (Definitional Statement 2.b.ii.5). A CMT should have the capability for its elements that are represented in OPR as a Process, to have a defined initial point, defined in OPR as the Initialization Characteristic. In order to show all that a Initialization Characteristic could show, it should be able to represent what are described above as objective and subjective conditions for the Process to begin.

Effect Characteristics – Defined by Definitional Statement 2.b.ii.6. A CMT that has dynamic changes should have elements that define those changes, which OPR sees as a Process, and the specific details of the change would be represented as the Effect Characteristic. In order to show all that a Process could describe as changing within a Model, the CMT should be able to describe changes to all of the possible Defining Qualities of three possible types of Components, and to represent the three possible alterations that an Effect Characteristic is possible to describe – creation, destruction, and alteration. Whether a CMT allows for Processes with multiple Effects to be defined, or if these are represented as multiple concurrent Processes is a design philosophy, so although this is a point born out in the observations and corollaries about Processes, it is not a required metric.

Behavior Characteristic – Defined by Definitional Statement 2.b.ii.7. A CMT that can show any of the types of change that occur within a system should be able to show

extant Processes, as well as instant Processes, as these are part of many systems. In order to represent extant Processes, the presence of the Behavior Characteristic becomes necessary, in order to define the behavior of the Effect Characteristics over the extant period of the Process. The requirements for all that a Behavior Characteristic can show about an Effect Characteristic include a description of the distribution of the Effect over the extant period of the Process.

Halting Characteristic – Defined by Definitional Statement 2.b.ii.8. A CMT that has Processes with defined periods during the life span of a simulation based on a Model described using the CMT should have the means for describing when that Process finishes. This may be an understood subjective condition based on the Effect of the Process taking place, but as a Model is intended to describe the behavior of a system, then the timing of the Processes of that system should be described, so a Process may have a Halting Characteristic that assists the Process to show what the system behavior is. In order to fully show all that a Halting Characteristic can show about Process termination, the CMT should be able to represent both objective and subjective conditions for the termination of a Process.

From these four criteria for determining what about Process definition a CMT should be able to capture, if it were able to capture everything that has been shown to be part of a model of a system, we can now see what it would take in order to fully answer the research question posed in Chapter 3.

The answer to "What is required for a modeling technique in order for it to address anything about a model that could be subject to change" is that a CMT must possess a full capability to show all of the possibilities captured in these four criteria.

Criteria for Evaluating Formal Method Enhancement. Now that we have identified the criteria for providing specific metrics to what a CMT can state about a Process, and we see that in order to fully satisfy the research question, a single CMT must be able to satisfy all of the possibilities addressed within the Process metric criteria given earlier (p. 164), applying those criteria to some sample CMTs is a way to evaluate their usefulness, and also to discover if they give us any new insight into what CMT processes can show.

The dissertation will apply only the Effect Characteristic criteria, both because they are the most specific, and also because they are the most important to the definition of what a Process describes within a Model. The Effect Characteristic defines what can be in the CMT's specific description of what a Process is changing with the Model, and since the definition given for a Process is "a component responsible for change or transformation" then it seems that perhaps the Effect Characteristic is vital.

The dissertation will evaluate via the criteria for evaluating what the Effect Characteristic within a CMT can display to the original surveyed CMTs from Chapter 2. Going through the list of CMTs, based on what was enumerated (p. 99) as what the CMT describes as changing in the model, that information will not be reproduced here. What can be said, however, is that the specific components, as identified by OPR, that the CMT addresses should be enumerated here.

Activity Diagram – What is changing is the time Object's Attribute, and a number of "activities" which are Processes, have their subjective temporal Characteristics defined in terms of when other Processes begin or end. The Processes themselves do not change, nor do their Characteristics change; the time object of the time Sub-model changes. This

is covered by (Definitional Statements 2.b.ii.5, 2.b.ii.7 and 2.b.ii.8), in that it shows the initialization, behavior, and halting of the processes that make up the activities.

Communications diagram – What is changing is the focus of which communicating object has the focus on data being passed among them via messages and returns. What is changing is the time Object's Attribute, and also the other Object components and their Attributes that represent the communicating objects. This is covered by Definitional Statements 2.b.ii.6 and 2.b.ii.6.a.

Discrete Event System Specification – The state transitions are modeled as changes to both the time sub-model (changes by Definitional Statement 2.b.ii.6 and behavior according to Definitional Statement 2.b.ii.7) and then the new state is represented by changes to the model's Objects and their Attribute (again by Definitional Statement 2.b.ii.6.a).

Flow chart – What is changing here is the time Object's Attribute, and the focus, by having transitions between different states be Processes identifying a "focus" Attribute on the Objects representing the states. Having Relations w Rules (Definitional Statement 2.b.iii) that are subjective to a focus Attribute of the state Objects illustrates how corollary 9 can affect the defined transition effects (Definitional Statement 2.b.ii.6). Those Processes begin and end (Definitional Statements 2.b.ii.5 and 2.b.ii.8) based on the changing time Object and their own subjective Initialization and Halting Characteristics, and the effect of the Process (Definitional Statement 2.b.ii.6) is the change the Attributes (Definitional Statement 2.b.ii.6.a) required to make the Object components of a state in focus.

Sequence Diagram – The time Object's Attribute that helps to order and make sequential the methods and returns described here changes (by the time Process, in a proper time sub-model, including a time Relation), as do the Objects (Definitional Statement 2.b.ii.6) and Attributes (Definitional Statement 2.b.ii.6a) that are the Effect subjects of the methods and returns. Those Effects have affiliated instant Behavior (Definitional Statement 2.b.ii.7), and the attendant identical Initialization and Halting Characteristics (Definitional Statements 2.b.ii.5 and 2.b.ii.8).

State Diagram - What is changing here is the time Object's Attribute, and the focus, by having transitions between different states be Processes identifying a "focus" Attribute on the Objects representing the states. This is another case where the Relations that connect the various Objects will have Rules that make the focus changing Process depending on those rules, illustrating Corollary 9 once again. Those Processes begin (Definitional Statement 2.b.ii.5) and end (Definitional Statement 2.b.ii.8) based on the changing time Object and their own subjective Initialization and Halting Characteristics, and the effect of the Process is to change the Attributes (Definitional Statement 2.b.ii.6.a) required to make the Object components of a state in focus.

Statechart Diagram - What is changing here is the time Object's Attribute, and the focus, by having transitions between different states be Processes identifying a "focus" Attribute on the Objects representing the states. A similar effect, as with State Diagrams, that illustrates the dependence of the Processes involved between states being dependent on the various Relations connecting those states is in play here as well. Those Processes begin (Definitional Statement 2.b.ii.5) and end (Definitional Statement 2.b.ii.7) based on the changing time Object and their own subjective Initialization and Halting

Characteristics, and the effect of the Process is the change the Attributes required to make the Object components of a state in focus.

Use-Case Diagram – As this is also a sequence capturing CMT, the time Object's Attribute is changing here, and other than that the Attributes describing the "Use-Case" Objects. In this CMT the Use-Cases do not actually do anything, it only indicates the sequence and order of when they occur, and which ones occur for each possible actor. This is why they are defined as Objects within OPR rather than as Processes.

All of these techniques illustrate something that is in common to all dynamic models. In a model which is representing information about a system where something changes, some order needs to be represented for that change, so that it can be identified which changes occur before, or after, which other changes. This necessitates some sort of time sub-model. As such, in all of our cases, a time sub model is present, with a time Object, a time Process, and an associating time Relation. The attributes of the time Object do not necessarily represent explicit time advance, but only time as required to be represented in the system – the ordering of states or activities, for instance. Other than the time sub model, the processes responsible for the system changes that the modeling techniques can represent would have to express Effects in terms of Defining Qualities – Attributes, Characteristics or Rules. Exempting any changes to the time sub model, those that are specifically accommodated by the modeling techniques here are shown in the following table.

Table 9

Dynamic elements of CMTs, other than Passage of Time

Conceptual Modeling Technique	Attributes	Characteristics	Rules
Activity diagram	X		
Communications diagram	X		X
Discrete event system specification	X		
Flow chart	X		X
Petri nets	X		
Sequence diagram	X		
State diagram	X		X
Statechart diagram	X		X
Use-case diagram	X		X

Analysis of Assessment Results. The stark omission in this table shows that the OPR formal method has identified a characteristic of systems that none of the surveyed CMTs can address as changing – that is, the definition of a Process within a Model that follows a CMT in question.

If it can be shown that a system may have a Process that changes with the operation of the System, and that a CMT could possibly be modified to represent change to a Process, then this omission in the current state of things, at least as represented by the surveyed CMTs, shows that the OPR formal method has uncovered knowledge about the state of modeling that was previously not part of the body of knowledge.

In order to see that this is necessary, but not accommodated by the modeling techniques analyzed here, consider the case of any model which has a Process whose effect is a derivative. As a result of the Process Effect altering a component (Object?) and changing it to a new state (by altering one or more of its Attributes via Definitional Statement 2.b.ii.6.a), the Effect (Definitional Statement 2.b.ii.6) of the Process itself would be changed as well (when defined in terms of the state of the Object). That none of the modeling techniques surveyed can show this exhibit a gap in those techniques. This has been uncovered by the application of the formal method, and it exists that its usefulness would be repeated by application to other techniques in order to evaluate for this same feature. That this has been done satisfies the "capability for definition for dynamic content" measure of merit.

CHAPTER 5

THEORY APPLICATION

The reason for developing the formal method in the preceding chapter was to answer the research question of the dissertation. The research question, restated, is as follows.

"Is a formal method possible that can fully describe all the aspects of a modeling technique, both static and dynamic, that allows for techniques representing any sort of dynamic change, including changes to the model itself?"

Rather than attempt to answer the question theoretically, it was through best to actually devise such a formal method, through the approach of grounded theory, by reviewing and analyzing existing knowledge, and from that, constructing new theory (the formal method) that is supported by, and refers to, existing theory. In this way the research question is answered, by the development of and presentation of new knowledge (the point of a dissertation), but it remains supported by and corroborated by existing theory. The formal method has been shown to be useful to apply to several existing conceptual modeling techniques – its intended use – and has been shown to reveal information about those techniques. In order to determine if this method and its application can satisfy the research question, however, the measures of merit set out in chapter three will be evaluated.

5.1 SATISFYING THE MEASURES OF MERIT

From Chapter Three, several measures of merit were presented to be used as evaluation tools capable of determining whether or not the research question of the

dissertation has been addressed. These are presented here, along with how the formal method answers them.

Capability for compositional description: Measure 1, the formal method should be able to describe modeling techniques with enough specificity so that similar techniques can be addressed, in order to make an informed decision between those techniques. That the formal method from Chapter Four satisfies this measure should be seen from examining the details in which the representation a model makes of a system is explained by the components of OPR. The differences (and similarities) of the techniques that were subjected to the OPR application in the preceding chapter illustrate this. For instance, from the modeling techniques OPR was applied to, both the sequence diagram and the communications diagram for UML illustrate the same phenomena – the passing of communications between one system element to another, yet the focus of each diagramming technique is different (giving the user of the model different perspective, and different information about the system being modeled). These differences are explicitly laid out, in terms of what the elements can and cannot show, using the OPR technique. This explicit difference can give the practitioner the specifics required to select between different techniques. This is beyond what could be shown given the typologies investigated in chapter two.

Capability for definition for dynamic content: Measure 2, the formal method should be able to describe the definitional parameters of the dynamic change capacity of a modeling technique, again so that such capacity could be compared between techniques. This is an extension of the first measure, by asking for further information in the area of dynamic change within a dynamic model. In the second chapter, the

typologies given for different modeling techniques identified some areas of difference between modeling techniques. But where several techniques are all in the same broad category of these dynamic characteristics (for instance, all discrete, or all stochastic) then having specific means for expressing and measuring these are desired, and did not exist in previous typologies. Within the formal method developed here, of course, the Process component is responsible for modeling the dynamic change within a model, and the specific means for expressing the Process Characteristics give the extra definition that this measure of merit is looking for. With the addition of the specifics of the process characteristics (p. 164), very specific differences between dynamic modeling techniques can be provided for.

Capacity for technique enhancement: Measure 3, the formal method that is derived should suggest what is possible in representation within a modeling technique, so that once the method is used to describe a technique, it should be able to suggest in specific ways how that technique could be modified, or extended, in order to make it capable of representing more about a system. This has not yet been exhibited, however to see that it is possible in theory is a matter of considering the formal method. It expresses the potential for a conceptual modeling technique to express information about a system. By identifying all of the functional capability of an existing modeling technique, whichever components or characteristics of the formal method were not relied on would represent some room for that conceptual modeling technique to be expanded to more fully satisfy the potential represented by the formal method. In answering this measure of merit, however, an actual application of this approach would demonstrate that the measure of merit is satisfied, in actuality, rather than just in potentiality. The

following section does that, by applying the OPR method to the Object-Process Methodology conceptual modeling technique, and identifying an area where an extension of the technique would give a modeler the means for further expressiveness in describing a system than already exists with the technique.

5.2 THEORY APPLIED FOR TECHNIQUE ENHANCEMENT

As an entire working example of the applicability of the assembled parts of this dissertation, the steps followed so far in exploring the research question will be followed with regards to a candidate case. The conceptual modeling technique chosen should be outside the corpus surveyed and followed throughout the case, such that the methods described herein can be witnessed in application to new territory. In selecting a candidate CMT, making a choice of one that is already strong in representation of processes is desired, such that the method can be seen to be applicable even when used in an area with marginal opportunity for improvement.

The CMT selected for this purpose is the Object Process Methodology, presented by Dov Dori (2002). From that work, a short description of the methodology follows:

"We have seen the three OPM entities: objects, process and state. Objects are things that exist, while processes are things that affect objects. States are situations at which objects can be. Processes transform objects in one of three ways: generating, consuming or affecting them. The effect a process has on an object is manifesting through a change in the object's state. " (Dori 2002, p. 10)

From just this short description, many of the elements that have been discovered and reported in this body of work are present, but not all of them. In order to make a proper study of the CMT as described herein, the following steps will need to be taken:

1. Analyze the CMT in terms of the OPR formal method. This will reveal which elements of the CMT are representing which components of the taxonomy.
2. Evaluate the functionality of the Process elements from the CMT against the Process evaluation criteria presented in the preceding chapter.
3. Having identified the present and missing functionality of the Process elements of OPM, make a decision about how to proceed.

The third step requires that some problem be in mind in order to motivate the next steps to be taken. As the identification of specific functionality results from the evaluation of the process components, that information can be used as the basis for application of various engineering techniques. Those might be evaluation, selection, rejection, extension, incorporation and so on.

Object-Process Methodology and the OPR Formal Method. Taking a look at the elements of the Object-Process Methodology, it can readily be seen which components of OPR that they tie to.

Table 10

Object-Process Methodology elements described as OPR Components

Object-Process Methodology Element	OPR Component
Object Entity	Object component
Process Entity	Process component
State (of an Object Entity)	Snapshot of Attributes (and their qualitative or quantitative values) of a particular Object

Table 10 Continued	
Tagged Directional Structural Link	Relation component between two Objects
Untagged Directional Structural Link	Relation component between two Objects
Tagged Bi-Directional Structural Link	Relation component between two Objects, may have different definition in each direction
Untagged Bi-Directional Structural Link	Relation component between two Objects, may have different definition in each direction
Aggregation Structural Relation	Relation component relating one or more component parts to a component whole, can be Object to Object, or Process to Process
Exhibition Structural Relation	Relation component relating a component to its (one or more) Defining Quality components (used when components fill the role of defining qualities, either Attributes or Characteristics), can be mixed Object and Process
Generalization Structural Relation	Relation component relating one or more parent (type) components to a child (instance) component, can be either Object to Object or Process to Process

Table 10

Continued

Classification Structural Relation	Relation component relating one or more children (instance) components to a parent (type) component, can either be Object to Object or Process to Process
Consumption Procedural Link	Relation from a Process to an Object indicating that the Process consumed the Object as its effect Characteristic
Result Procedural Link	Relation from a Process to an Object indicating that the Process creates the Object as its effect Characteristic
Effect Procedural Link	Relation from a Process to an Object indicating that the Process changes one or more of the Object's Attributes as the effect Characteristic of the Process
Input and Output Procedural Link	Relation from a Process to an Object indicating that it changes one or more of the Object's Attributes from the input state (some value of the Attribute(s)) to the output state (some value of the Attribute(s)) as the effect Characteristic of the Process

Table 10

Continued

Agent Procedural Link	Relation from a Process to an Object, where the Object specifically represents a person necessary as part of the initialization Characteristic of the Process
Instrument Procedural Link	Relation from a Process to an Object, where the Object specifically represents an instrument necessary as the initialization Characteristic of the Process
Invocation Procedural Link	Relation from a Process to another Process indicating an edge temporal relationship between the halting condition Characteristic of the first Process and the initialization Characteristic of the second Process
State Image	Identifies the Attributes of an Object, and in the Value Image variant, indicates the values of those Attributes
Condition State Link	Relation between an Object's Attribute and a Process, indicating that the Attribute must be in a certain state (i.e. have a particular value) as part of the initialization Characteristic for the Process

Table 10 Continued	
Agent Condition State Link	Same as a Condition State Link, but in this case the Attribute-exhibiting Object is an agent, which for OPM is specifically a human
Qualification State Link	Relation component indicating that a certain Attribute of an Object must be in a certain state (i.e. - have a particular value) in order to be a type Object to another Object serving as an instance
Instance Qualification State Link	Relation component indicating that a certain Attribute of an Object must be in a certain state (i.e. - have a particular value) in order to be an instance Object to another Object serving as a type
State Specified Consumption Link	Combined Consumption Procedural Link Relation component with a Rule defining quality that the Process effect Characteristic only is enacted if the related Object Attribute is in a specified state (i.e. have a particular value)
State Specified Result Link	Specific version of a Result Procedural Link Relation component that specifies the created Object and one or more specific Attribute of that Object as the effect Characteristic of the Process
Table 10	

Continued	
Determination Boolean Object	A Rule defining quality that affects the Relation components related to a specific Object – all other Relations are only valid when the Boolean condition of the Rule is satisfied
Condition Boolean Link	A combined Relation component with a Rule between an Object's Attribute and a Process's initialization Characteristic, indicating that the Process initializes when the Attribute is in a particular state (i.e. has a particular value)
Negative Condition Boolean Link	The negative version of a Condition Boolean Link
Both Conditions Boolean Link	Both a Condition Boolean Link and a Negative Condition Boolean Link

A quick survey of the first several OPR axioms (from Chapter 4) reveals the following conditions about what the elements of OPM can represent.

Axiom 1: There exists for every Model, some content that is related to the system it represents. This axiom clearly applies to OPM, as it is intended as a modeling language that captures the objects and processes of a system, and presents them in one of two distinct ways – in graphics, and in text.

Axiom 2: The content of a model describes how the Input to a simulator based on that model would transform into the Output that such a simulator would produce.

Axiom 3: The content of a OPM model is definitely comprised of some non-empty subset of elements representing the possible components, Objects, Processes and Relations.

Axiom 4: When there are elements of OPM that are classified as Objects under OPR, they each have the minimum Defining Quality (Attribute) of an Identity (Definitional Statement 2.b.iv.4), so this is satisfied.

Axiom 5: When there are elements of OPM that are classified as Processes under OPR, they each have the minimum Defining Quality (Characteristic) of an Identity (Definitional Statement 2.b.iv.4), so this is satisfied.

Axiom 6: When there are elements of OPM that are classified as Relations under OPR, they each have the minimum Defining Quality (Rule) of an Identity (Definitional Statement 2.b.iv.4), so this is satisfied.

Evaluation of OPM Process Entity. Now, following the method introduced in (p. 180) in the preceding chapter, the dissertation will examine particularly the Processes of OPM and determine which Defining Qualities that are represented as possible change Effects.

Table 11

Dynamic capacity of the OPM element responsible for change

OPM Element (Process Component)	Attributes	Characteristics	Rules
Process Entity	X		X

We can see, from the enumeration of elements that although there are many that map to the Relations definition provided by OPR, there is only one that maps to the Process definition. In this case, it is also called the "Process Entity" element. Through the very rich set of possible elements that fulfill the Relation value, OPM allows the "Process Entity" element, as a Process Component of OPR, to have Effects on many of the possible Rules of Relations, and also on Objects, and presumably their Attributes (although OPM recognizes Attributes as separate from the Object, more in a mereological sense, than as the Defining Quality sense of OPR).

The Process Entity (as the element is fully referred to as) in OPM has a textual definition as "A **process** is a pattern of transformation that an object undergoes." Consider the extension of this to "A **process** is a pattern of transformation that an entity undergoes." In OPM the Building Block category of "Entities" includes "Object Entities", "Process Entities" and "State Entities". There are already a number of "links" that associate "Process Entities" with both "Object Entities" and also "State Entities" (even though the definition of a "Process Entity" states that it transforms only "Object Entities", there are links that show the "Process Entity" associating with a "State Entity" as both Input and Output – transformation).

Extension to the OPM Process Entity. Based on what OPR tells us should be possible within a Model, we can easily envision an extension to OPM by adding a "procedural link" type that associates together a Process with another Process, which allows for the full realization of our extended definition – "A **process** is a pattern of transformation that an entity undergoes". Now, we can see that OPM can have "process entities" affecting other "process entities" (See Figure 28).

That we can do this, exhibits that it is possible for a CMT to have as its capability the representation of a Process Effect showing change to a Process Characteristic. As OPM is intended to be represented both textually and diagrammatically, such a change must be represented diagrammatically via multiple structures, but this is still an improvement in efficiency in not having the extension.

Consider as an example of this efficiency a system that has a Process which has

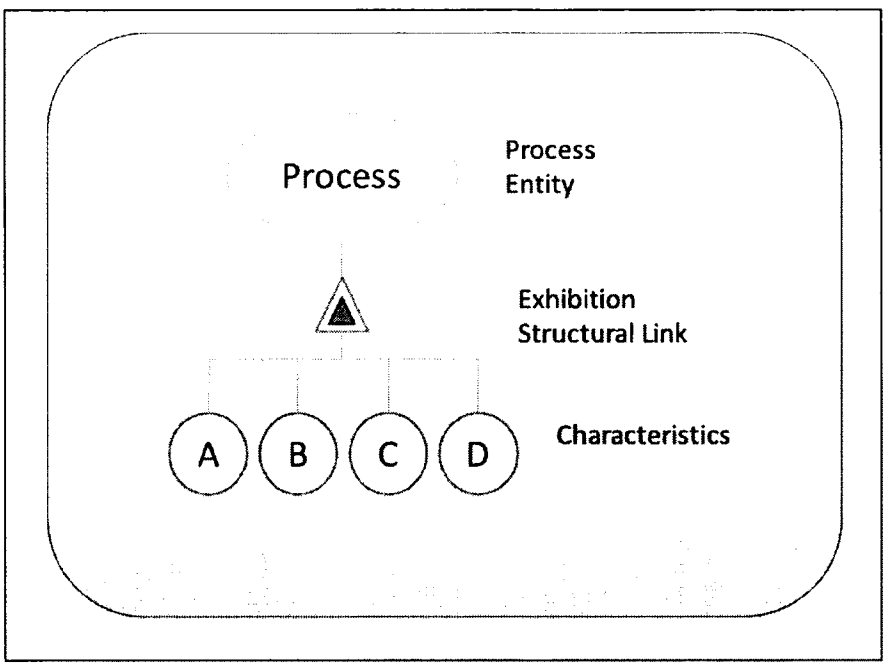

Figure 29 New Process Capability from OPM Extension

an Effect of changing the Attribute of an Object. Now, also consider that the working of the system also has a Process that changes the initial Process, so that over time the Effect that the Process introduces changes. It may change to a different defining quality (perhaps changing an Attribute of Object O2 instead of Object O1), or it may change the values of its Effect (perhaps, incrementing by 6 instead of incrementing by 2), or it could change its Effect Behavior (it changes an Attribute of O1 evenly over a period of time –

this could change to, it changes an Attribute of O1 the whole change all at once). In any case, this would, under traditional OPM required that multiple Process Entity blocks be represented in the model, with a separate Process Entity block that would show the sequential switching between them. Now if a Process could affect a Process – all of the possible changed Processes in the diagram would be replaced by one Process. The multiple Links would remain, but no more than before (each Process in the old case would have a link; the single Process in the new case would have a link for each possible state of Effects). That the Links would be used to represent different existential states of the changing Process is a departure from how they are normally used in OPM, so an indicator with the Link, showing the sequence of change, would be required as part of the extension.

Evaluation of the Extension. What does this change help the Model user to see better about the system? First, it is a more efficient diagram by having a reduced number of elements. More importantly it more accurately represents the system that the Model is describing. In the system there is only one Process, albeit one that is changing over time due to the Effects of another Process. However, by representing that Process as multiple, different sequential Processes in the un-extended OPM technique, the impression is given to the Model user that they are different Processes. Applying the OPR metric to identify an area of possible extension, and then identifying what is needed for the extension has made the possible expressive strength of OPM greater.

An example of how that might occur can be seen if an attempt to model a system that has internally two or more processes whose activities has an effect on each other. Such a system might include a model of linear combat that is based on the Lanchester

laws for determining casualties and attrition. A treatment of the laws themselves, with information on how they may be modeled using a variety of techniques (UML and Systems Dynamics, for instance) is presented in Artelli and Deckro (2008). For the purpose here of illustrating the benefits to OPM of being extended in the area of having a Process that can affect Characteristics, it is enough to state the following:

Lanchester laws provide a pair of differential equations, where two sides in combat, a red side and a blue side, have a current total of troops, a coefficient representing combat effectiveness, a current combat effectiveness factor based on the current state, and an associated attrition function that takes into account all four of those factors and translates them into casualties for the opposing side. These two functions happen simultaneously, such that the function affects the other side's total size, and the other side's state driven combat factor, while having the function's own side's total size and own state driven combat factor reduced.

Extended Object-Process Methodology Applied. In general sense it can be represented by (*Figure 31* Lanchester Laws illustrated with simplified OPM). The attrition factors are just record keeping values (derived from the current force size times the combat coefficient for that side), that are fed into the attrition functions each time step, but because of the limitation of the Process modeling capability of OPM, must be stored as an object, because of the pseudo-concurrent nature of the calculation representation. However, if the attrition Process, which is already using the factor information internally, and reading in the size of both forces, it could calculate and keep as part of the Process Effect characteristic the attrition factor. It would, however, have to be changeable by the opposing attrition function. This would require a Process capable

of modeling a change to another Process's Characteristic. If that were possible, then the diagram could be reduced in complexity to appear as in (*Figure 30* Model of Lanchester Laws improved with Enhanced OPM). What can be seen is that the simplified OPM

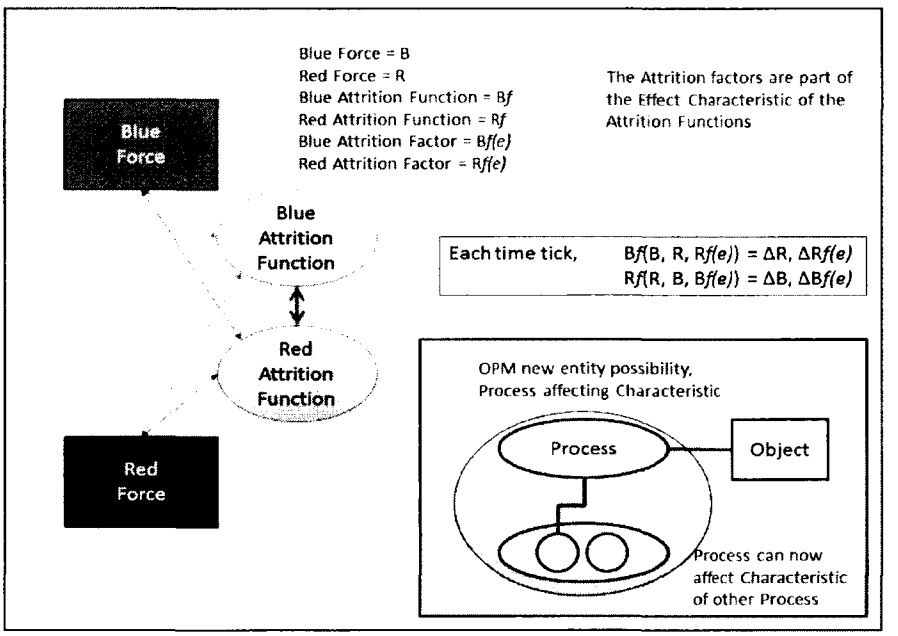

Figure 30 Model of Lanchester Laws improved with Enhanced OPM

model in (*Figure 30* Model of Lanchester Laws improved with Enhanced OPM) is a clearer representation, capturing the meaning of what the Lanchester Square Laws are doing each time step in a linear combat model, by being able to show the effect of one process on another.

It is now, in extended form, a CMT that can, in certain modeling situations (as the one illustrated) depict a better conceptualization of the system being modeled – the CMT has been enhanced. The OPR formal method revealed an area that OPM was not capable

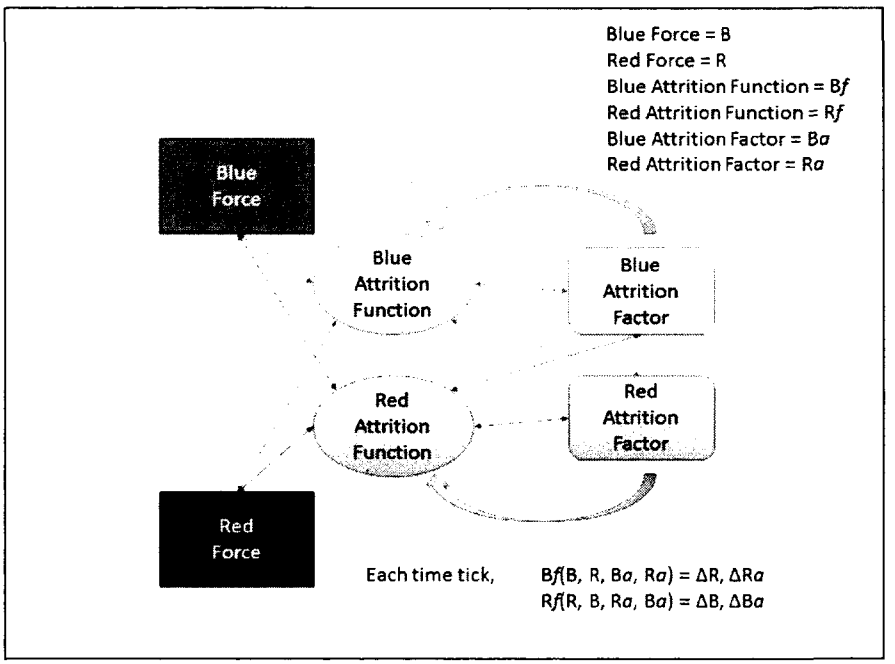

Figure 31 Lanchester Laws illustrated with simplified OPM

of exhibiting – a change by a Process to the defining characteristic (a Characteristic) of another Process. By then extending the CMT being analyzed in that direction, it is shown that the CMT becomes capable of exhibiting knowledge about a system in a more expressive manner. In revealing this new capability, the "capacity for technique enhancement" measure of merit has been shown to have been addressed.

The Reason for Method Extension. As was shown earlier (p. 22), there are likely to be a number of different views of the same referent, and these can be either more or less formal, or more or less explicit in what they show concerning how the model could possibly be implemented. It may be so that some users of a model might be fine with what OPM, or some other technique can show without being extended, as after all this holds with the multiple model view, and also is in support of the fact that the

modeling techniques exist and are useful for their own community as they already exist. However, it has also been shown (p. 19) that all of the components of a model that is to exhibit information concerning a dynamic system are equally important, if we are to gain a complete view of that system. Each model component – Object, Process and Relation – can reveal additional information concerning the system they describe, if accommodated in the modeling technique. In the case of the analysis of surveyed techniques, and their ability to represent dynamic activity through the Process component, it was revealed that none of the techniques are able to show the effects of a Process on another Process. With the previous subsection's example, it was shown that such a technique, OPM, can describe a system that has internal activity based on a Process affecting another Process, even if the modeling technique does not allow for it. However, it required additional elements to be present in the resulting model. The enhanced technique (OPM with the enhanced Process) was able to show a similarly expressive representing about the system; however it did so with fewer elements. In this example, it is up to the ultimate user of the model to decide which view is more useful; however the enhancement did show that the formal method was instrumental in guiding a technique's enhancement that was capable of improvement in at least one dimension – representation of knowledge about a system with fewer components.

It may be, however, in other modeling situations that not having the ability to express information that a modeling technique does not support will lead to not only a different view of a system, but perhaps an incomplete or flawed view. For this, consider the following abstract example. Consider a system that consists of a circular, connected network of four places, and a token that traverses from place to place due to a process

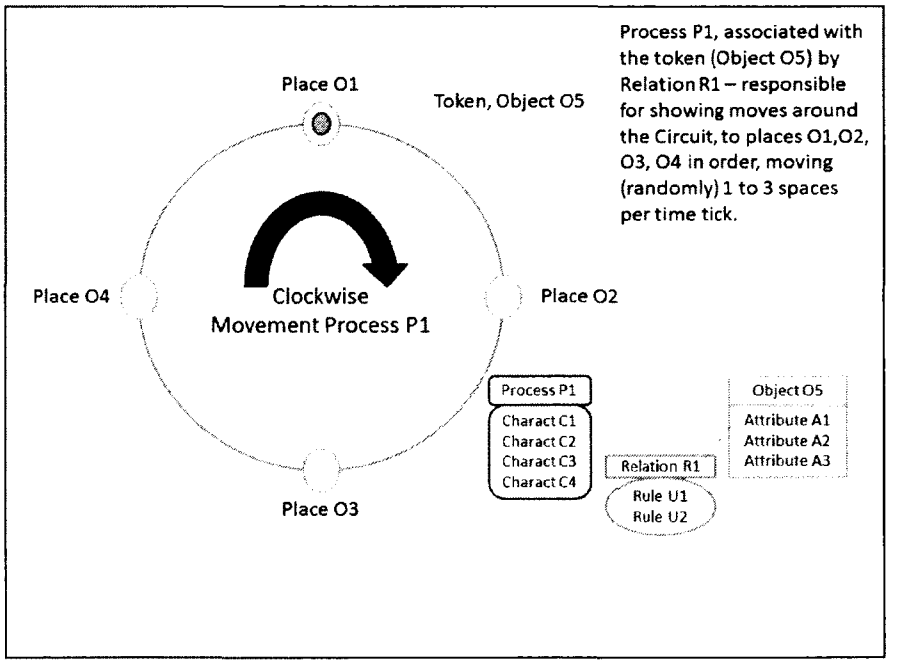

Figure 32. Simple Model Example

that moves the token. Each time tick in the model, the process causes the token to move, clockwise, either one, two or three spaces. In the example shown (*Figure 32.* Simple Model Example) the process, P1, causes the object, O5 to move from the initial place (O1) to one of the other three spaces (O2, O3, O4). If this model is assessed by an outside observer, once it is modeled in a system, it is enough for the model to express the process P1, and its effects on object O5, associated by relationship R1. Assume, to illustrate the case that another model then has to interpolate the mid-point of O5, based on the reported effects that were applied by P1. We can see that this is possible in the case where O5 is affected only by a singular, non-changing P1. Assume that O5 starts as position O1, and then follow the steps in the table below (Table 12

Results of Model with only Process-Object). The Process Effect is a post-fix process, after the Time Tick, so the Interpolated Midpoint is calculated from the position

between the position after the last Time Tick, and the position after the current Time Tick. The Process Effect indicates how many location Objects are traversed, and also if the move is clockwise (CW) or counter-clockwise (CCW). In this first example, since there is no change to the Process, all are clockwise.

Table 12

Results of Model with only Process-Object Effects

Time	Process Effect	New location of O5	Interpolated Midpoint	Interpolation correct?
T0	Move 1, CW	O2	45 deg	yes
T1	Move 3, CW	O1	225 deg	yes
T2	Move 2, CW	O3	90 deg	yes
T3	Move 1, CW	O4	225 deg	yes
T4	Move 2, CW	O2	0 deg	yes
T5	Move 2, CW	O4	180 deg	yes

Now consider the case where there is a process (P2) in the system, so represented in the model, that affects the behavior of the process (P1), by changing its behavior from rotating clockwise, to rotating counter-clockwise. If the outside evaluation of the location of token O5 is based only on the knowledge of the objects, and the processes that affect them, then it cannot know what effect P2 will have on P1, and so will not know that movement could be counter-clockwise, and hence will get the Interpolated Midpoint

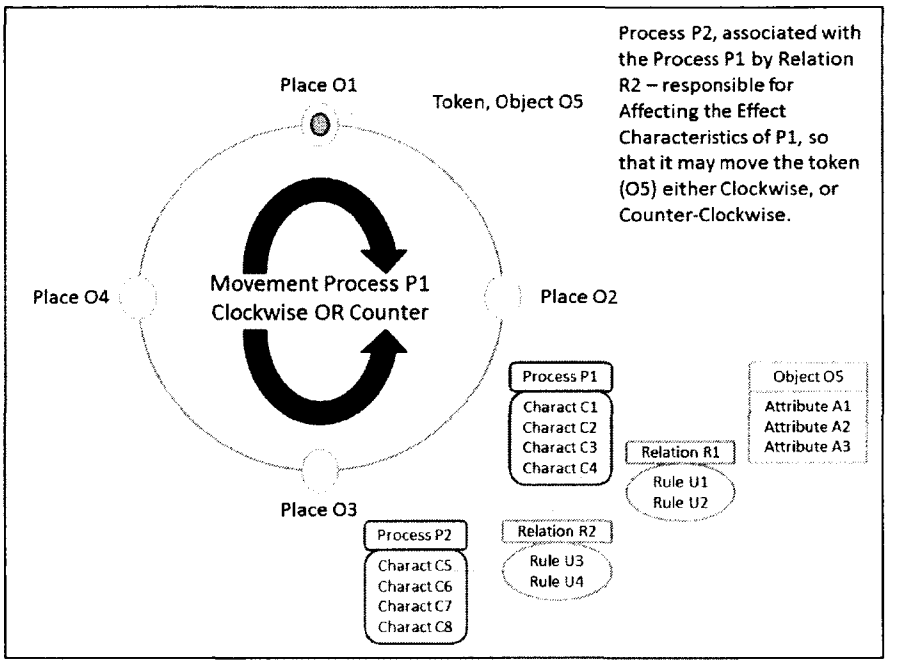

Figure 33 More Complex Model Example

wrong. This sort of modified model is represented by (*Figure 33* More Complex Model Example), and the results are presented in (Table **13**

Results of more Complex Model). As can be seen from the results table, when only the position of the object (as an effect of P1 on the token O5 are shown), then it becomes difficult to have reliance on an interpolation of the midpoint. This leads to interpreting the results as either (1) false, or (2) chaotic and unpredictable. If the knowledge of the second process, P2, is known, however, it is predictable and complexity is reduced. As with the earlier example showing Lanchester based linear warfare modeled with OPM and then enhanced OPM, in a system that does not allow for representation of a process having an effect on another process, it can be done by introducing additional elements into the model to capture the P2 process effects as

somehow involving some object, and that object being relied on by the P1 process. This obfuscates the model, and in a case where the relationship is more than just two processes deep, it quickly gets too complex for understanding. For an actual example, consider a weather system, where all of the processes involved (wind direction and speed; temperature raising or lowering; air pressure raising or lowering; rates of chemical transformations of chemical species in the air of the weather system; changes in the saturation of the air by water vapor; and more)have an effect on each other. In order to represent this in a modeling paradigm that does not allow for the representation of processes affecting processes quickly becomes beyond comprehension, as there are just too many interactions to represent each with a sort of process-to-process interruption into a state conveying object. Such a complex system illustrates what is stated by Shlezinger, et al. (2006) and Sowa (2000) that all elements of the model are equally important to represent, in order to have an accurate view of how the system the model represents is capable of performing. In cases where some of those elements cannot be represented by a modeling technique that is otherwise desirable, technique extension via the principles of OPR application (as described earlier) could serve as one possible solution.

Table 13

Results of more Complex Model

Time	Process Effect	New location of O5	Interpolated Midpoint	Interpolation correct?
T0	Move 1, CW	O2	45 deg	yes
T1	Move 3, CW	O1	225 deg	Yes
T2	Move 2, CCW	O3	90 deg	No (270 deg)
T3	Move 1, CW	O4	225 deg	Yes
T4	Move 2, CCW	O2	0 deg	No (180 deg)
T5	Move 2, CCW	O4	180 deg	No (0 deg)

5.3 OPR AND THE MEASURES OF MERIT

It has been established that the three measures of merit devised to evaluate a formal method against the requirements of answering the research question have been satisfied. The first two from work that was shown in the preceding chapter – the "capability for compositional description" (p. 161), and the "capability for definition for dynamic content" (p. 186), and the third with the work presented in this chapter (p. 199). The OPR formal method is a technique that can serve to provide evaluation method,

either when comparing conceptual modeling techniques, or when examining a single technique for its limits or possible extension.

In addition, the characteristics of OPR that enable it to satisfy the first two measures of merit have also been shown to be useful in defining the border conditions between models in a multi-model application (p. 155).

CHAPTER 6

CONCLUSION

In concluding this dissertation, a look back at what has been stated is warranted. In addition, specifically showing how the research question was addressed is worth stating, as well as identifying the contribution that the work makes to the body of knowledge. Finally, possible directions for future work are identified.

6.1 RESULTS OF THE DISSERTATION

The dissertation has, through the preceding five chapters, moved from a general understanding of what modeling is towards a specific method for describing the role of the components of a modeling technique, for the purposes of analysis, classification or comparison. This was done because the gap in the body of knowledge was identified as the ability, given current methods for describing and classifying modeling techniques, to sufficiently describe them so that they can be compared one to another. The specifics of this gap that were identified were that it would be useful, but currently not possible, to sufficiently describe the dynamic aspect of what a modeling technique can represent about a system it is being used to model and also to sufficiently describe the specific contents of a modeling technique.

This gap was then analyzed to derive the research question:

"Is a formal method possible that can fully describe the components, structure and dynamic behavior of a conceptual modeling technique?

This research question was based on several observations about what would need to exist in order to close the gap. Those things are enumerated here (again, taken from Chapter 3).

4. Describe all of the components of a model, using a meta language that could be applied to any conceptual modeling technique.
5. Describe the dynamic capacity of a modeling technique with specificity.
6. Not limit the ability to have the dynamic capacity of a modeling technique to address changes to any part of the model, including the model itself – meaning the structure and nature of the components from point number 1.

To answer the question, the dissertation followed the research technique of developing grounded theory (Chapter 3), and based on successive rounds of literature review, analysis and distillation, new theory (resulting in the formal method that answers the research question) has been developed. It has been shown to be applicable, and it was then judged according to measures of merit designed to evaluate the resulting formal method. Those measures of merit are listed here:

1. Capability for compositional description - The formal method should be able to describe modeling techniques with enough specificity so that similar techniques can be addressed, in order to make an informed decision between those techniques.
2. Capability for definition for dynamic content - The formal method should be able to describe the definitional parameters of the dynamic change capacity of a modeling technique, again so that such capacity could be compared between techniques.

3. Capacity for technique enhancement - The formal method that is derived should suggest what is possible in representation within a modeling technique, so that once the method is used to describe a technique, it should be able to suggest in specific ways how that technique could be modified, or extended, in order to make it capable of representing more about a system.

The application and evaluation of the formal method (chapters 4 and 5) have shown that all three of those measures of merit have been answered. As such, the dissertation has shown that the research question has been answered, and in such a way that the gap in the body of knowledge is addressed.

6.2 ANSWERING THE RESEARCH QUESTION

As mentioned, the dissertation had a research question, which was couched in terms such that a derived formal method could satisfy it. The OPR formal method from this dissertation has been derived to satisfy the research question. It satisfies the measures of merit designed to provide metrics for success. However, for the benefit of the modeling and simulation practitioner (a serious consideration in identifying the gap in the current body of knowledge, and also in the research question) what does the developed formal method deliver? As seen in the initial literature review of chapter two, there are many different modeling techniques. There are also many different "levels" of understanding about a system that a model can deliver. Likewise, it has been made clear that there are the potential for as many different individual models (even given a specific combination of modeling technique and targeted level of representation) as there are questions about the system in question, and modelers to help answer those questions. This suggests a very rich potential space for models, and modeling techniques, that has at

last three dimensions, probably more. Within that space, if an expert is tasked with selecting a modeling technique he can approach it in a number of different ways. There are practical reasons why a technique may be chosen (it is supported by the practitioner because of familiarity, for example); there are operational reasons why a technique may be chosen (it is supported by the practitioner's organization either through policy or preference); or it may be up to the practitioner to select a method that may have the best job of delivering a model. In this case, the practitioner will want something that can help answer whatever question about the system has been asked, but by using the best model, can achieve the proper mix of expressive capability about the aspects of the system required to be considered for answering the question, and simplicity by abstracting away other distracting elements about the system that do not contribute to answering the question. In this case, having a formal method (such as that developed here) to highlight which elements of the system can, or cannot, be expressed by a modeling technique – and which among several candidate techniques performs that expression with the highest suitability for the question being asked of the system – is what the resulting method of this dissertation will deliver.

6.3 FUTURE WORK

There are two things notably missing from the OPR formal method that resulted out of this dissertation. The first is in depth research and identification of the nature and defining qualities of both object components and relation components (process components are explored in greater depth, because they were the original motivation for the dissertation study). There have been many studies of objects over the years, and some of relations, yet it remains for a future student to provide a rigorous study to

provide the identification of a classification of Attributes and Rules, as they are referred to in this dissertation.

The second thing would be the development of the OPR formal method into its own modeling technique. Such a technique might prove to be too rigorous and formal for most model users, as a model is to abstract away the details of a system, but retain the ability to express what the system does and how it behaves. This is abstraction, and is rendered not useful to the modeler if made excessively rigorous and formal. This is why formal mathematics is not used to describe our modeled systems – it is not abstract enough to provide a common and easy to use language for domain practitioners, even though it provides specificity and rigor that most modeling languages lack. The purpose for having OPR turned into a modeling language on its own would be for formal cases – for further studies into modeling and what can be done with modeling.

Without developing OPR into a formal method, however, future work will most likely be accomplished by using the method to evaluate, and express in the neutral terms it can deliver, explorations of any number of modeling techniques, and presenting these in the literature. In addition, with the application of specificity to the defining qualities of the components of the formal method, it should also be possible to use it to evaluate not only conceptual modeling techniques, but also specific conceptual models themselves. The implications and potential uses of such an application await future exploration.

REFERENCES

Adrion, W., Branstad, M., &Cherniavsky, J. (1982). "Validation, verification, and testing of computer software," *Computing Surveys, 14*(2), pp. 159-192.

Anderson, J. (2006). Automata theory with modern applications. Cambridge: Cambridge University Press.

Artelli, M., & Deckro, R. (2008) "Modeling the Lanchester Laws with system dynamics". Journal of Defense Modeling and Simulation, *5*(1).

Backhouse, R., (1986) *Program construction and verification*, London: Prentice-Hall International (UK) Ltd.

Balci, O. (1998). "Verification, validation and testing". In J. Banks (ed.), *Handbook of Simulation* (pp. 335-393) New York: John Wiley & Sons.

Banks, J. (1998). *Handbook of simulation.* New York: John Wiley & Sons.

Banks, C. (2009) "What is modeling and simulation"? in J. Sokolowski and C. Banks (eds.) *Principles of modeling and simulation* (pp. 3-24), Hoboken, NJ: John Wiley & Sons. .

Barendregt, H., (1981) *The lambda calculus: Its syntax and semantics*, New York: North-Holland.

Black, P. (2008) "Dictionary of Algorithms and Data Structures", U.S. National Institute of Standards and Technology. 12 May 2008. February 9, 2012. http://www.nist.gov/dads/HTML/finiteStateMachine.html

Booch, G., Rumbaugh, J., &Jacobson, I. (1999). *The Unified Modeling Language user guide*, Reading, MA: Addison-Wesley.

Brade, D. (2004). A generalized process for the verification and validation of models and simulation results. Doctoral Dissertation, at the Universitat der Bundeswehr Munchen. Published February, 2004.

Bratley, P., Fox, B., &Schrage, L. (1987).*A guide to simulation. (2nd ed.)*. New York, NY: Springer-Verlag.

Broy, M.(2003). Multi-view modeling of software systems. In B. Aichernig & T. Maibaum (Eds.), *Formal Methods at the Crossroads* (pp. 207-225). Heidelberg, Germany: Springer-Verlag.

Chandrasekaran, B., Josephson, J.,& Benjamins, V. (1999). What are ontologies, and why do we need them? *Intelligent Systems and their Applications. 14(1)*, 20-26.

Chang, C. C., &Keisler, H. J. (1990).Model theory:, Studies in logic and the foundations of math (3rd ed.). Amsterdam, Netherlands: North-Holland.

Chen, P. (1976). The Entity-Relationship Model – Toward a unified view of data. *ACM Transactions on Database Systems, 1*(1), 9-36

Cloutier, R., Winkler, A., Watson, J., &Fickle, C. (2003) "Modeling a system of systems using UML," in Proceedings of the Conference on Systems Engineering Research, Hoboken, NJ, March, 2003..

Codd, E. (1970). A relational model of data for large shared data banks. *Communications of the ACM, 13*(6), 377-387.

Codd, E. (1974). Recent investigations in relational data base systems. In Proceedings IFIP Congress, Amsterdam, Netherlands, August 5-10, 1974...

Corcho, O., Gomez-Perez, A. (2000). A roadmap to ontology specification languages. In R. Dieng and O. Corby (eds.), *Knowledge Engineering and Knowledge Management. Methods, Models and Tools* (pp 80-96). Berlin, Germany: Springer.

Dillon, L.K. (1990). "Using symbolic execution for verification of Ada tasking programs."*ACM Transactions on Programming Languages and Systems*, *12*(4), 643-669.

DoD. (2007). DoD Architecture Framework version 1.5 volume I: Definitions and guidelines. Washington, DC: U.S. Department of Defense.

Dori, D. (2002). *Object-Process methodology*. Berlin, Germany: Springer-Verlag.

Gilbreth, F.B. (1922) "Process charts, first steps in finding the one best way to do work". In: *ASME Transactions, 43*. American Society of Mechanical Engineers.

Franck A, &Zerbe V (2003) "A combined continuous- Time/Discrete-Event Computation Model for heterogeneous simulation systems". LNCS 2834, Springer

Frege, G. (1892). "Über begriff und gegenstand". *Vierteljahresschrift für wissenschaftliche Philosophie*, 16:192-205.

Friedenthal, S., Moore, A., &Steiner, R. (2009). "A practical guide to SysML". Burlington, MA: Elsevier, Inc.

Fujimoto, R.M., (1993) "Parallel discrete event simulation: Will the field survive?" *ORSA Journal on Computing*, *5*(3), 213-230.

Grenon, P., & Smith, B. (2004). "SNAP and SPAN: Towards dynamic spatial ontology". *Spatial Cognition & Computation: An Interdisciplinary Journal*, *4*(1), 69-104.

Guarino, N.,& Welty, C. (2000). "A formal ontology of properties". In R. Dieng and O. Corby (eds.) *EKAW-2000: The 12th International Conference on Knowledge Engineering and Knowledge* Management (pp. 97-112). Berlin, Germany: Springer.

Haller, A. & E. Oren. (2006). A process ontology to represent semantics of different process and choreography meta-models. Published as DERI Technical Report 2006-02-03. Galway, Ireland: Digital Enterprise Research Institute.

Haller, A., Oren, E., & Kotinurmi, P.. (2006). "m3po: An ontology to relate choreographies to workflow models". *Proceedings of the 3rd International Conference on Services Computing*, Budva, Montenegro, June, 2006, pp. 19–27.

Harel, D. (1987), "Statecharts: A visual formalism for complex systems." *Science of Computer Programming, 8*, 231-274.

Hester, Patrick T., &Andreas Tolk. (2010). "Applying methods of the M&S Spectrum for Complex Systems Engineering," In Proceedings, 2010 Spring Computer Simulation Multi-Conference, Boston, MA, April 2010.

Jensen, K. (1987). "Coloured Petri nets," in *Petri nets: Central models and their properties*, Springer-Verlag, 248-299.

Kewley,R., &Tolk, A. (2009) A systems engineering process for development of federated simulations, Presented at the Live Virtual Constructive Conference, El Paso, TX, January, 2009.

King, R., &Turnitsa, C. (2008). The landscape of assumptions. In Proceedings of Spring Simulation Multiconference, 81-88. IEEE Press.

King, R. (2009). On the role of assertions for conceptual modeling as enablers of composable smulation solutions. Doctoral dissertation from Old Dominion University, Modeling, Simulation and Visualization Engineering Department, College of Engineering. Norfolk, VA.

IEEE. (1990). *IEEE standard glossary of software terminology*. Published as IEEE Standard 610.12- 1990. January 25, 2012. http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=2238

International Standards Organization. (ISO) (2004). Industrial automation systems and integration - Process specification language sandard. Published as ISO Standard 18629.

Lowe, E. J. (1998). The possibility of metaphysics: Substance, identity, and time. Oxford, England: Oxford University Press.

Law, A., &Kelton, W. (2000). Simulation modeling and analysis (3rd ed.). New York, NY: McGraw Hill

Manna, Z., Ness, S.,&Vuillemin, J., (1973) "Inductive methods for proving properties of programs," *Communications of the ACM, 16*(8), 491-502.

Mealy, G. (1955). "A method for synthesizing sequential circuits". Bell Systems Technical Journal *34*, 1045–1079.

Model. (2012). Merriam-Webster Dictionary. 7 January, 2012, http://www.merriam-webster.com/ dictionary/model

Modeling and Simulation Coordinating Office (MSCO). (2006). Recommended practices guide for verification, validation, and accreditation (VV&A) Build 3.0.http://vva.msco.mil/. Retrieved from url.

Model. (1989a). *The Oxford English Dictionary* (OED). Oxford University Press. 15 January 2012 http://dictionary.oed.com/.

Moore, E. (1956). Gedanken-experiments on sequential machines. *Automata Studies, Annals of Mathematical Studies 34*, 129–153.

Mylopoulos J (1992) Conceptual modeling and telos. In P. Loucopoulos, R. Zicari (eds) *Conceptual modeling, databases, and case: An integrated view of information systems development* (pp. 49-68) . Wiley: New York.

Von Neumann, J. (1963). "Planning and coding of problems for an electronic computing instrument." In J. von Neumann Collected Works Vol. 5.(pp. 80-151). New York, NY: Macmillan.

National Institute of Standards and Technology (NIST). (1993). IDEF0: Integration definition for function modeling. Federal Information Processing Standard (FIPS) Publication 183. Washington, DC.

Niles, I., &Pease, A. (2001). Towards a standard upper ontology. *2001 Proceedings of the International Conference on Formal Ontology in Information Systems*, Ogunquit, ME, October 17-21, 2001.

Object Management Group (OMG). (2002). *Unified Modeling Language Specification, version 2.2.* Retrieved from http://www.omg.org/

Object Management Group (OMG). (2010). "System Modeling Language Specification, version 1.2." Object Management Group. Retrieved from http://www.omg.org/spec/SysML/1.2/

Ogden, C., &Richards, I.(1923). The meaning of meaning: A sudy of the influence of language upon thought and of the science of symbolism. London, UK: Routledge & Kegan Paul.

Page, E., &Nance, R., (1994) "Parallel discrete event simulation: A modeling methodological perspective," in D.K. Arvind, R. Bagrodia, & J.Y-B. Lin (Eds.), *Proceedings of the Eighth Workshop in Parallel and Distributed Simulation* (PADS '94), pp. 88-93, IEEE Computer Society Press., Los Alamitos, CA, July 1994.

Pace, D.K. (2000). Ideas about Simulation Conceptual Model Development. *Johns Hopkins APL Technical Digest* 21(3), 327-336.

Padilla, J., Diallo, S.,&Tolk, A. (2011) Do we need M&S science? *SCS Modeling & Simulation Magazine* 2011, *4*, 161-166..

Peterson, J. (1981). Petri net theory and the modeling of systems. Saddle River, NJ: Prentice Hall.

Petri, C. (1966). Communication with automata. *DTIC Research Report AD0630125.*

Reisig, W. 1992. A primer in Petri net design. Berlin: Springer-Verlag

Reynolds, C. &Yeh, R.T., (1976) "Induction as the basis for program verification," *IEEE Transactions on Software Engineering*, SE-2, *4*, 244-252.

Robinson, S. (2006). Issues in conceptual modeling for simulation: Setting a research agenda. *Proceedings of the 2006 OR Society Simulation Workshop*, Worcestershire, UK, March 23-24, 2006.

Robinson S. (2008). Conceptual modeling for simulation part I: Definition and requirements. *Journal of the Operations Research Society*. 59(3), 278-290

Sargent, R. (2005). Verification and validation of simulation models. In Proceedings, 2005 Winter Simulation Conference, Orlando, FL, December 4-7, 2005..

Schach, S.R., (1996) *Software engineering (3rd ed.)*, Hornwood, IL: Irwin.

Schichl, H. (2004). Models and the history of modeling. In J. Kallrath (Ed.), *Modeling Languages in Mathematical Optimization* (pp. 25-36). Boston, MA: Kluwer.

Shannon, C. (1948). The mathematical theory of communication. *Bell System Technical Journal,* 27(3), 79-423, 623-656.

Shannon, R.E., (1975) *Systems simulation: The art and science*, Englewood Cliffs, NJ: Prentice-Hall.

Shlezinger, G., Reinhartz-Berger, I., &Dori, D. (2006). Analyzing object-oriented design patterns from an object-process viewpoint. *Proceedings of the 2006 Next Generation Information Technologies and Systems*, Kibbutz Shefayim, Israel, July 4-6, 2006.

Sider, T. (2001). Four-dimensionalism: An ontology of persistence and time. London, UK: Clarendon Press.

Simulation. 1989b. The Oxford English Dictionary (OED). Oxford University Press. 15 January 2012 http://dictionary.oed.com/.

Smith, B., &Grenon, P. (2004). The cornucopia of ontological relations. *Dialectica, 58*(3), 279-296.

Sommerville, I., (1996) *Software engineering (5th ed.)*, Reading, MA: Addison-Wesley.

Sowa, J. (2000). Knowledge representation: Logical, philosophical and computational foundations. Pacific Grove, CA: Brooks Cole Publishing.

Sterneckert, A. (2003). *Critical incident management*. Boca Raton, FL: CRC Press.

Tolk, A., Litwin, T., &Kewley, R. (2008). "A systems engineering process supporting the development of operational requirements driven federations," In *Proceedings of the 2008 Winter Simulation Conference*, IEEE Press, Miami FL,December 7-10, 2008.

Tolk, A., Diallo, S., King, R.,&Turnitsa, C. (2009) "A layered approach to domposition and interoperation in complex systems," In A. Tolk and L. Jain (Eds.): *Complex systems in knowledge based environments: Theory, models and applications. Studies in Computational Intelligence, SCI 168*, (pp. 41-74), Berlin: Springer.

Tolk, A., Diallo, S., King, R., Turnitsa, C., &Padilla, J. (2010) "Conceptual modeling for composition of model-based complex systems," X. Robinson, X. Brooks, X. Kotiadis, and X. Van Der Zee (Eds.), *Conceptual modeling for discrete-event simulation* (pp. 355-381). City, ST: CRC Press.

Tolk, A., Diallo, S., Turnitsa, C., &Padilla, J. (2011) "How M&S interoperability is different from other interoperability domains," Presented at the *2011 Spring Simulation Conference*, Boston, MA, April 4-7, 2011.

Turnitsa, C., Tolk, A.,& Padilla, J. (2010) "Ontology for modeling and simulation," In Proceedings, 2010 Winter Simulation Conference, Baltimore MD, date, pp 643-651.

Wainer GA (2009). Discrete-event modeling and simulation: A practitioner's approach. Boca Raton, FL: CRC Taylor& Francis.

Weilkins, T. (2006). *Systems engineering with SysML/UML*. Burlington, MA. Elsevier, Inc.

West, M. (1996). Developing high quality data models. Newcastle, UK: University of Newcastle

Whitehead, A. (1978). *Process and reality: An essay in cosmology*. Reprint of the 1929 original. New York, NY: Free Press.

Whitner, R., &Balci, O., (1989) "Guidelines for selecting and using simulation model verification techniques," in E.A. MacNair, K.J. Musselman, & P. Heidelberger (Eds.), Proceedings of the 1989 Winter Simulation Conference, pp. 559-568, IEEE, Piscataway, NJ, December 4-6, 1989.

Yilmaz, L.(2004) On the need for contextualized introspective models to improve reuse and composability of defense simulations. *Journal of Defense Modeling and Simulation, 1*(3), 141-151.

Zeigler, B., Praehofer, H., & Kim, T.G. (2000).*Theory of modeling and simulation, (2nd ed.)*. San Diego, CA: Academic Press.

APPENDIX I

ADDITIONAL CONSIDERED TECHNIQUES

There are, of course, a wide variety of different conceptual modeling techniques. Many were evaluated in the course of analyzing the state of the art, through the literature, and further in deriving the formal method of the dissertation. Some of those that were not included in the list of reported on techniques that make up the survey in the dissertation are listed here, with a brief explanation of why they were not included in the survey. This information, which may prove useful to the reader, is presented here, in the appendix format, because it is not a formally structured, referenced, or formally derived as the main part of the dissertation – it is a collection of observations and reasoned decisions.

- Bayesian Belief Network – A hierarchical model, formally a directed acyclic graph, where the edges (representing transitions between states) have a probability associated with them, representing the chance of THAT particular edge from out of one state will be the next one to occur when the overall system is in that state. Describing these probability markers can be done following Bayes' theorem, and partial information about the state of a system described by a Bayesian Belief Network (or Bayesian Model, although this term while more popular is less precise) can be relied on, through application of Bayes' Theorem, to determine the associated weights of other information within the system being a true representation of what actually happened. In terms of what the model provides, from an OPR perspective, it is very similar to a state machine representation, yet with the difference of having the probabilities associated with

the transitions from one state to another. Because of this, it was rejected for analysis, although the decision to reject was difficult because of usefulness of the technique, and the amount of literature associated with it.

- BOM – The Base Object Model standard is a method for specifying models of HLA objects from the Simulation Interoperability Standards Organization. This technique calls for a conceptual model for the object that each application is modeling, and it calls for the conceptual model to cover the object and techniques that would make up the patterns of activity described by the BOM, however the techniques for doing that conceptual model are not present (rely on UML?). There is a new version, BOM2, of which OPR might make a nice way to evaluate what the standard DOES say about conceptual modeling.
- Business Process Model and Notation – very similar to an activity diagram, as shown by Stephen A. White, "Process Modeling Notations and Workflow Patterns", IBM technical report. Because of the similarity, and that activity diagrams have been included, BPMN was not considered in addition to activity diagrams.
- Conceptual Graph – A conceptual graph is a diagrammatic version of First Order Logic. Proper conceptual graphs are proper graphs (following the definition from graph theory). There are other variations that are not as rigorously defined; those are typically diagrams, rather than graphs. This was originally considered because it is a good candidate for modeling the relationships among concepts, however as with all first order logic based models, the ability to represent truth over time is not possible (which negates its usefulness for a dynamic modeling

system). For insight into why this is so, consider Charles Peirce, "Time has usually been considered by logicians to be what is called 'extralogical' matter. I have never shared this opinion. But I have thought that logic had not yet reached the state of development at which the introduction of temporal modifications of its forms would not result in great confusion; and I am much of that way of thinking yet." – as quoted in Moshe Y. Vardi (2008). "From Church and Prior to PSL". In Orna Grumberg, Helmut Veith. *25 years of model checking: History, achievements, perspectives*. Springer.

- DODAF – This is the U.S. Department of Defense Architectural Framework. The technique is very similar to the U.K. MODAF (Ministry of Defense Architectural Framework). Not considered due to concurrency of Ph.D. research within the MSVE department at ODU, from Ted Schuman – although this would be a good candidate to have the various views subjected to the OPR treatment to evaluate their completeness and similarity of capability to other methods. DODAF is based, somewhat, on the diagrammatic techniques of UML, so some coverage does exist here, in tone, if not in specifics.
- Event driven process chain – very similar to a flow chart, so not considered in addition.
- FEDEP – The Federation Development and Engineering Process, presented with the Standard for the High Level Architecture. This is a specific process model intended to be used in the development and implementation of simulation federations. As such it was not deemed to be a general candidate for conceptual modeling.

- GBKR – Graph Based Knowledge Representation – a form of Conceptual Graph, not considered for the same reasons that Conceptual Graphs were not considered.
- KAMA – The conceptual modeling technique developed by the Middle Eastern Technical University, in conjunction with the Turkish Armed Forces (maintained at the ModSimmer Center of the Middle Eastern Technical University, at Ankara http://www.modsimmer.metu.edu.tr/en/). It is a conceptual modeling tool designed, originally, to model military (especially c2) systems and activities. It is based very strongly on the Meta Object Facility (MOF) from the Object Management Group, and several of the elements from UML. KAMA diagrams are expressed using elements from a UML class diagram, and a few add-ins from other UML techniques (such as borrowing the user representation from use-case diagrams). Because of that, and because it does not express dynamic activity directly (but rather shows relations among objects and classes within the modeled system) it was rejected for analysis, but may prove to be a useful tool to subject to OPR based description and analysis later on.
- Markov Chains – Equally as popular, and also as attractive as a candidate for study, as the Bayesian Belief Network. Markov Chains have a number of very interesting and attractive features (the memory-less nature of the transition functions is one such) that might suggest them as a modeling technique for some problems, however form the OPR perspective, there is little to differentiate it from other FSM modeling methods, with the exception of the Process described transition function having its Characteristics based only on defining qualities of the Object(s) and Process(es) associated with the current state. In way, this

suggests that following OPR, each Markov Chain state and it associated transitions (Objects, Processes, and associating Relations) are each a Sub-Model. This remains as an observation and not a defining quality for Markov Chains, however it appeared close enough to FSMs that analysis of the technique was rejected for the dissertation work, however future analysis of the variations of Markov Chain modeling would be an attractive source of study to employ OPR.

- Predicate Calculus – the symbolic language for representing predicate logic statements. Attractive, especially from a formal study of methods, because of its limited and clear number of elements, however finally rejected because it is not clear that any in the literature regard it as a conceptual modeling language.
- Process Calculus – intended for showing concurrent activity in an electrical system – such as parallel processing in a computer. As such, not suitable for a general conceptual modeling language, and not typically (by the literature consulted) used in that manner. For this reason it was not considered for analysis.
- Process Ontology – The term ontology is a bit overloaded in the more recent literature, meaning on one hand an "ontological representation" for systems science. In this sense it is an artifact that attempts to capture the meaning of the objects and activities of the system. When such a representation is being applied to a "process" – some sequence of activity that a system does – it may be called a process ontology. The second meaning for ontology is the more typical, and older, use of the term from philosophy that refers to an encapsulation of an individual's world view – how that individual perceives the ordering and makeup of the universe around them. In that sense, a process ontology is used much in the

sense as was presented in the dissertation from Grenon and Smith (2004), where a four dimensional view of the universe (everything is in flux, all matter only takes "form" as a temporary characteristic, and processes are changing those characteristics all the time) explains the ontological commitment to understanding reality. In both cases, systems science, and philosophy, the process ontology is not considered to be a conceptual modeling tool, so was not considered for analysis – but as can be seen in the literature of Chapter 2, it did help to inform and provide input into the resulting analysis leading to the OPR formal method.

- Workflow Diagrams – capture the sequencing of work activities. Different activities are represented, and the temporal flow from one to the next is represented, with capacity for concurrency, requirements satisfaction, bifurcation of flow, and so on. It is not considered to be a conceptual modeling technique, although it certainly could be in the realm of representing activities or extended processes. It was rejected for analysis because the technique does not capture any more information than is available in certain UML and SysML techniques, most notably the sequence diagram.

APPENDIX II

SUMMARY OF FORMAL METHOD

Presented here, all together, are the statements, corollaries and definitions that together make up the OPR Formal Method. These are gathered together in one spot, because they appear in the dissertation spread over a number of sections in Chapter 4. Collecting them all together provides a service for the reader to locate any statement, corollary or definition quickly. Axioms are presented in the formal method statements, and are highlighted for ease of location.

DEFINITIONAL STATEMENTS

1. A Model is a representation of a system (see Definition 1).
    a. A Model has content describing the behavior of the represented system.
    b. A Model's content describes the transformation of some Input into some Output as the operation of the represented system.

Let Z be a system.

Let M be a Model.

Let Z(M) be the Model of System Z.

Let Φ be the contents of the Model.

Let $\underline{M}$ be the modeling relationship, showing that Model's contents are related to a System that the Model Represents.

Axiom 1: There exists for every Model, some content that is related to the system it represents.

$\forall M, \underline{M}(\Phi(M),Z(M))$ (eq. 1)

c. A simulator may exist that implements the model.

d. A simulation is an instantiation of the simulator, accepting specific input, and then producing specific output (see Definition 2).

Let α be the Input to a Model, as it is implemented by the simulator.

Let ω be the Output of a Model, as it is implemented by the simulator.

Let S be a simulator based on a model. SM is a particular simulator based on the particular model, M.

Let ξ be a function representing simulation; ξS being the simulation of the simulator S, accepting Input to a model, and Contents of a model, and producing Output of a model.

Axiom 2: For every Model, there exists a simulation such that the content of the Model describes how the Input to a simulator based on that model would transform into the Output that such a simulator would produce.

$\forall M \exists SM, \xi S(\alpha, \Phi(M)) = \omega$ (eq. 2)

2. A Model's content is comprised of components. (see Definition 3)

a. There exist three types of components; Objects, Processes and Relations.

Let O be the set of all possible Objects, O{o1, o2, o3, ...}

Let P be the set of all possible Processes, P{p1, p2, p3, ...}

Let R be the set of all possible Relations, R{r1, r2, r3, ...}

Let Ω be the set of all component sets, Ω{O, P, R}

Let Ω_M be a subset of Ω

Let the operation $\rightarrow$ indicate that the pre-term is composed of the post-term.

Axiom 3: The content of each Model is comprised of some (non-empty) subset of all these possible components – Objects, Processes and Relations.

$\forall M\ (\Phi(M) \rightarrow \Omega_M(M))$ (eq. 3)

b. Each component is a non-empty set of some Defining Qualities.

Let Θ be the relationship between a component and its Defining Qualities.

i. Objects are sets of Attributes. (see Definition 4)

1. Attributes may have associated qualitative or quantitative Values.
2. An Attribute may appear in any number of Objects.
3. Each particular Object-Attribute pairing has a unique identity.
4. The Attributes and/or Attribute-Values of an Object will not change, unless acted on by an outside component.
5. An Object describes a thing in the model that can be considered in isolation from all else in the Model.

Let A be the set of all possible Attributes, A{a1, a2, a3, ...}

Let A_O be a subset of A

Axiom 4: Every Object O is defined by some set of Attributes.

$\forall O\ \exists A_O \Theta(O, A_O)$ (Eq. 4)

ii. Processes are sets of Characteristics. (see Definition 5)

1. Characteristics may have associated qualitative or quantitative Values.
2. A Characteristic may appear in any number of Processes.

3. Each particular Process-Characteristic pairing has a unique identity.
4. A Process describes a change to a component in the model.

Let C be the set of all possible Characteristics, C{c1, c2, c3, ...}

Let C_P be a subset of C

Axiom 5: Every Process P is defined by some set of Characteristics.

$\forall P \exists C_P \Theta(P, C_P)$ (Eq. 5)

9. For every Process component of model M, there may be an Initialization Characteristic, known as C_I
    c. C_I is defined as either an objective time point, or a subjective condition.
        i. An objective time point has some identified value on some Time Sub-model's "time Object"
        ii. A subjective condition is defined as some set of Defining Qualities of model M that have some specific values.
            1. The subjective condition would be associated by a Relation between the Process and the components that contain the Defining Qualities to be evaluated
            2. The associating Relation (definitional statement 2.b.iii) would have a subjective Rule (Corollary 6) that would define the requisite terms of the subjective condition.
    d. The Process will occur each time the objective time point occurs or subjective conditions are satisfied during a simulation implementation of the model, M.

i. This allows for an objective point or subjective conditions to be defined in such a way that they are a set of points or conditions

ii. In the case of an objective point, if the Time Sub-Model has its "time Object" repeat the same point more than once, each time would satisfy the conditions of the objective point Initialization definition.

10. For every Process component of Model M, there may be one or more Effect Characteristics, known as C_E.

a. The Effect of a Process defines what the change the Process describes in the Model

i. The Effect is some change to one or more Defining Qualities

1. The change can be creation, as in creating a new Defining Quality

a. When a new Defining Quality is created, it may or may not have a Value paired with it

b. All Defining Qualities must be paired with a component

2. The change can be destruction, as in removing an existing Defining Quality

3. The change can be alteration, as in altering the Value of an existing Defining Quality

ii. If the Process has more than one Effect Characteristic, and the Effects are to more than one Defining Quality, they need not all be of the same component

b. For each Effect to take place, there will be a Relation associating it with the affected Defining Qualities

c. Effects only alter Defining Qualities

i. The Effect can create, destroy, or change all of the Defining Qualities of a component

ii. If the Identity Quality of a component is destroyed, the component itself is destroyed (Definitional Statement 2.b.iv.4)

iii. If a new Identity Quality is created by an Effect, then the new component it is paired with is correspondingly created

11. For every Process component of Model M, there may be one or more Behavior Characteristics, known as C_B.

a. The Behavior Characteristic, for an Extant Process, determines how the Effect takes place over T_{Int}

b. If there is more than one Defining Quality that is changed as an Effect of the Process, then a Behavior may be defined for each

12. For every Process component of Model M, there may be a Halting Characteristic, known as C_H.

a. C_H is defined as either an objective time point, or a subjective condition.

i. An objective time point has some identified value on some Time Sub-model's "time Object"

ii. A subjective condition is defined as some set of Defining Qualities of model M that have some specific values.

1. The subjective condition would be associated by a Relation between the Process and the components that contain the Defining Qualities to be evaluated
2. The associating Relation (Definitional Statement 2.b.iii of Chapter 4) would then have a subjective Rule (Corollary 6) that would define the requisite terms of the subjective condition.

b. The first time that the Halting Characteristic, either subjective or objective, occurs when the Process has already been initialized, is when the Process Halts.

c. If the Halting Characteristic is equal to the Initialization Characteristic, then the Process is Instant, rather than Extant.

iii. Relations have Rules. (see Definition 6)

1. Rules may have associated qualitative or quantitative Values.
2. A Rule may appear in any number of Relations.
3. Each particular Relation-Rule pairing has a unique identity.
4. A Relation describes the association of two or more components and/or Defining Qualities.

Let Γbe the set of all possible Rules, $\Gamma\{\gamma 1, \gamma 2, \gamma 3, \ldots\}$

Let Γ_R be a subset of Γ

Axiom 6: Every Relation R is defined by some set of Rules.

$\forall R \; \exists \Gamma_R \Theta(R, \Gamma_R)$ (eq. 6)

iv. There is a set of Defining Qualities for a model (see Definition 7)

1. Each unique pairing of a component and a Defining Quality for a Model M is a member of the set Q_M.
2. This is the set of all Defining Qualities (A, C, or Γ) that are associated with a particular member (O,P, or R) of set Ω_M. (see eq. 3)
3. Each Value that is associated with a member of Q_M exists in the set V_M.
4. The first Defining Quality for each component is the Identity Quality that has as a Value, the identity of the component.

3. Input to and Output from the Model are sets of Defining Qualities

a. The collected Defining Qualities of all of the components in the Model (as well as any Values that may be associated with those Defining Qualities) at the beginning of a simulation are together termed the Input to that simulation based on that Model.

i. Not all Defining Qualities have paired Values.

ii. For the Defining Qualities that do have paired Values, and that are represented in a simulator based on that model at the initialization

of a simulation, then all of those Defining Quality/Value pairings would represent the Input to the simulation.

b. The Output from the simulator is some subset of all of the Defining Qualities of all the components in the model (as well as any Values that may be associated with those Defining Qualities) at the time the Output is produced by the simulator, during the particular simulation resulting from a particular Input.

c. Output can be produced once the simulation halts, or at any point during the instantiated implementation of the simulation.

d. Both Input and Output may be empty sets.

COROLLARY STATEMENTS

Corollary 1: The simplest Model is one consisting of a single Object.

$\exists M_{simple} (\Phi M_{simple} = o_n)$ (eq. 7)

Corollary 2: The simplest dynamic Model is one consisting of three components – an Object, a Process, and a Relation.

$\exists M_{SimDyn} (\Phi M_{SimDyn} = \{o_1, p_1, r_1\})$ (eq. 8)

Corollary 3: A Sub-model does not have any definitional Relations for any of its components to other components outside of the Sub-model.

Let sM be a Sub-model of Model M

The contents of M are Ω_M (as per eq. 3)

The contents of sM are a subset of Ω_M, or Ω_{sM}

The contents of M that are not contents of sM are $\Omega_{\sim sM}$

Corollary 4: If there is a Relation, r1 that exists between some component of Ω_{sM} and some component of $\Omega_{\sim sM}$ then r1 can only be definitional from $\Omega_{sM} \rightarrow \Omega_{\sim sM}$. The Relation r1 cannot be definitional from $\Omega_{\sim sM} \rightarrow \Omega_{sM}$.

Let R_{Type} be a relation between 2 or more members of the set Ω_M

Let the relation R_{Type} distinguish the type component to the instance component(s).

Corollary 5: The existence of types and instances of components are so indicated by the R_{Type} relation.

Corollary 6: There are subjective Rules, a subset of Γ, that describe a subjective truth condition for the Relation they are paired with to be in effect.

Corollary 7: Given the same Simulator S_n and the same Input α_n, there will always be the same Output ω_n, when the Simulator is a Turing compliant construct. A Simulator of this type can be considered a formal Simulator.

Corollary 8: A Process that does not have an initialization Characteristic as part of its defining qualities will never occur unless an initialization Characteristic is created during a simulation of the Model by a Process.

Corollary 9: While the defining qualities of a Process describe its behavior, as they are all associated with other components or defining qualities in the model by Relations, the Characteristics of the Process can be given control through Relation and Rule definitions.

Corollary 10: Without a defined Halting Characteristic, a Process is considered to be in existence and operating until the simulation based on the Model halts.

Corollary 11: A Process may have multiple Effects, each with their own Behavior. Each of these is a separate Defining Quality and therefore may all be the object of other Process Effects separately.

Corollary 12: Although temporal Characteristics of Processes are considered to be objective when they are described in reference to time components of the Model, they are actually subjective, and the time components behave in the defined manner as all other components.

Corollary 13: A model has Processes whose Characteristics are subjectively related to a time sub-model, yet the model is not restricted to having only one time sub-model.

Corollary 14: A Process whose Initialization and Halting Characteristics are equivalent is an instant Process, and one whose Initialization and Halting Characteristics are separated by some measureable distance in a time sub-model, is an extant Process.

Corollary 15: A modal interruption to a Process will leave an extant Process's Effect Characteristic as only having introduced the change described by the Behavior Characteristic to have occurred prior to the interruption.

DEFINITIONS

Definition 1:*Modeling* is the purposeful process of abstracting and theorizing about a system, and capturing the resulting concepts and relations in a conceptual model.

Definition 2:*Simulation* is the process of specifying, implementing and executing a model.

Definition 3: A *model component* is an identifiable part of the model that represents some part of the knowledge that makes up the whole model, but which can be considered individually as well.

Definition 4: An *Object* component is a model component that has continued existence.

Definition 5: A *Process* component is a model component that is responsible for describing change or transformation.

Definition 6: A *Relation* component is a model component that is responsible for associating other components.

Definition 7: *Attributes* are defining qualities for Objects, and grant them qualitative and quantitative distinction from other objects.

Definition 8: *Characteristics* are defining qualities for Processes, and they describe the behavior of the Process as well as providing qualitative and quantitative distinction from other Processes.

Definition 9: *Rules* are the defining qualities for Relations, they serve to identify the nature of association the Relation is making, and to provide qualitative and quantitative identity to the Relation.

Definition 10: *Defining Qualities* are the means for expressing the meaning of what a Component represents within the model. A Defining Quality may have an associated Value with it, when appropriate, providing parameterization for that aspect of the Component's meaning.

VITA

Charles Turnitsa completed his Bachelor's of Science degree, from Christopher Newport University in 1991. He also earned a minor in History from that same institution. In 2006, he completed his Master of Science degree, from Old Dominion University, in Electrical and Computer Engineering (concentrating in Modeling and Simulation). That year, 2006, he began his Doctoral studies at Old Dominion University. This dissertation completes a partial requirement towards receiving the Doctor of Philosophy in Modeling and Simulation from Old Dominion University in 2012.

The author has worked in information research and technology development since before earning his B.S. degree in 1991. Much of his career has been involved in modeling, and data modeling, in research and development efforts for the U.S. Department of Defense (mainly the U.S. Army), and also the National Aeronautics and Space Administration (NASA). The author joined the Virginia Modeling Analysis and Simulation Center, as a project scientist, in 2004, and remained there through 2011, continuing to do research and produce a number of publications (journal articles, book chapters, conference proceedings) that have great bearing on what is presented here.