TRANSFORMING CONCEPTUAL MODELS OF THE MISSION SPACE INTO
SIMULATION SPACE MODELS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

FATİH KÜÇÜKYAVUZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

MARCH 2011

Approval of the Graduate School of Informatics

_____

Prof. Dr. Nazife Baykal
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Prof. Dr. Yasemin Yardımcı
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____   _____

Dr. N. Alpay Karagöz   Assoc. Prof. Dr. Onur Demirörs
Co-Supervisor   Supervisor

Examining Committee Members

Prof. Dr. Semih BİLGEN   (METU, EEE) _____

Assoc. Prof. Dr. Onur DEMİRÖRS   (METU, IS)   _____

Dr. N. Alpay KARAGÖZ   (METU, IS)   _____

Assist. Prof. Dr. Aysu Betin CAN   (METU, IS)   _____

Assoc. Prof. Dr. Altan KOÇYİĞİT   (METU, IS)   _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this wok.

Name, Last Name: Fatih KÜÇÜKYAVUZ

Signature   :   _____

ABSTRACT

TRANSFORMING CONCEPTUAL MODELS OF THE MISSION SPACE INTO SIMULATION SPACE MODELS

Küçükyavuz, Fatih
M.S., Department of Information Systems
Supervisor: Assoc. Prof. Dr. Onur Demirörs
Co-Supervisor: Dr. N. Alpay Karagöz

March 2011, 86 pages

Helping to abstract a valid model from real system, conceptual modeling is an essential phase in simulation development lifecycle. With the development of the KAMA framework, a new methodology was presented to develop mission space conceptual model for simulation systems. It provides metamodel elements represented by graphical diagrams to develop conceptual models of mission space. BOM (Base Object Model), developed by SISO (Simulation Interoperability Standards Organization), is another conceptual modeling concept serving for simulation space.

KAMA models are very close to problem domain and intend to model real world concepts in requirement analysis and development phase. Whereas, being vital inputs for the simulation design phase, BOM models are closer to solution domain. Hence there is no defined way of using the captured mission space knowledge in simulation space, problem arises when moving from requirement analysis to design phase. In

this study, to solve this problem, we propose a method for transforming mission space conceptual models in simulation space. Our solution approach will be mapping the KAMA mission space models to BOM simulation space models for automatically transport real world analysis results to simulation designers.

ÖZ

GÖREV UZAYINDAKİ KAVRAMSAL MODELLERİN SİMÜLASYON UZAYINDAKİ MODELLERE DÖNÜŞÜMÜ

Küçükyavuz, Fatih
Yüksek Lisans, Bilişim Sistemleri
Tez Yoneticisi: Doç. Dr. Onur Demirörs
Yardımcı Tez Yöneticisi: Dr. N. Alpay Karagöz

Mart 2011, 86 Sayfa

Kavramsal model, gerçek sistemden doğrulanmış bir model oluşturmayı sağlamasıyla benzetim geliştirme yaşam döngüsünün vazgeçilmez bir parçası olmuştur. KAMA aracı da görev uzayı kavramsal model geliştirme sürecine yeni bir metodoloji sunmuştur. Grafiksel olarak temsil ettiği metamodel elemanları sayesinde KAMA, kavramsal modelin geliştirilmesine olanak sağlamıştır. SISO (Simulation Interoperability Standards Organization) ise benzetim uzayında kavramsal modelleme yapabilmek için BOM (Base Object Model) adında yeni bir konsept ortaya koymuştur.

KAMA gereksinim analiz ve geliştirme safhalarındaki modellerin oluşturulmasını amaçlarken, BOM benzetimin tasarım ve geliştirme safhalarındaki modeller için kullanılmaktadır. Dolayısıyla benzetimin geliştirme yaşam döngüsü boyunca, gereksinim analiz safhasındaki elde edilen bilginin benzetimin tasarlanması ve

geliştirilmesi safhalarına aktarılma gereksinimi ortaya çıkmaktadır. Biz de bu çalışma kapsamında KAMA ile geliştirilmiş görev uzayı kavramsal modellerinin BOM benzetim uzayına dönüştürülmesi için bir yöntem önermekteyiz. Çözüm önerimiz de KAMA modellerini BOM modelleri ile eşleyerek KAMA'dan BOM'a dönüşüm yapmak olacaktır.

Anahtar Kelimeler: Kavramsal Model, KAMA, BOM, Görev Uzayı, Benzetim Uzayı

ACKNOWLEDGMENTS

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVATIONS

BOM: Base Object Model

CMMS: Conceptual Models of the Mission Space

DCMF: Defense Conceptual Modeling Framework

DMSO: Defense Modeling and Simulation Office

FEDEP: Federation Development and Execution Process

FOI: Swedish Defense Research Agency

HLA: High Level Architecture

MOF: Meta Object Facility

OMT: Object Model Template

OWL: Web Ontology Language

OWLS: Web Ontology Language Service

RF: Radio Frequency

RWR: Radar Warning Receiver

CHAPTER 1

INTRODUCTION

Nowadays, computer simulations are widely used in many distinct contexts and domains from training, finance, sales to engineering and military. Besides evaluating effects of the changes in real life systems, they are also used to get the idea of how a system works. As all of them are complex in their nature, much effort is required to make a valid abstraction from real world models to simulation models.

In general, simulation developers do not have enough knowledge about the domain they develop for. They implement the models which are conceptually created and validated by domain experts of the simulation systems. Therefore; it is very crucial that there exists a formal way of creating conceptual models for simulation domain experts and well formed communication bridges between domain experts and simulation developers.

In this study, we try to find an effective way to transform the models developed by domain experts into models which can be captured and used by simulation developers for further simulation development phases.

In the following parts of this chapter, we first present the context by giving the definitions of simulation and modeling and explain why conceptual modeling is crucial and then we state the problem we aimed and finally we describe our solution approach for the problem.

1.1. The Context

Rothenberg states that "modeling in its broadest sense is the cost effective use of something in place of something else for some cognitive purpose" [1]. In this definition there are two important points; first, modeling is cost effective way of replacement of real entity. In terms of cost, it must be reasonable to use models instead of real entity. Secondly, model has a purpose according to its referent in real life.

Robinson describes the simulation with four aspects; operations systems, purpose, simplification and experimentation and defines it as: "experimentation with a simplified imitation (on a computer) of an operations system as it progresses through time, for the purpose of better understanding and/or improving that system" [2]. The first aspect, operations systems, is related with the system to be simulated. Entities in real world and their relationships constitute the operations systems of simulations. Second, he describes the rationale of simulations with purpose aspect. The aim of developing simulations can be summarized as promoting better understanding of the real system and making improvements. In his third aspect, for the sake of simplification, he states that it is not likely and even not desirable to include all the details of real system in the simulation. Only the required details should be modeled in the simulation. And in his final aspect, Robinson explains the experimentation as changing the variables in the simulation model and predicting the results of these changes in real system.

From the definition of the simulation, especially from first three aspects, it can be deducted that system analysis is required in-depth before making valid experimentations on simulation. Defining the entities, their capabilities and relationships in the system (operations systems), intention of their usage (purpose) and finally deciding the boundaries of the system by including only the intended details (simplification), all require modeling the system conceptually before moving on further steps of simulation study. That is why this study focuses on enhancing the usage of conceptual models in simulations.

2

Zeigler describes the conceptual modeling as abstraction of a model from a real or proposed system. He also identifies the abstraction process as simplification of the reality [3]. Borah explained the key terms of simplification and abstraction in conceptual modeling. He states the simplification as eliminating the unnecessary details to form simpler links in an analytical way and he defines the abstraction as establishing base functions of real world system and representing those functions in a different form [4].

Having emphasis on the simplification, description of Zeigler is not seemed to be complete by Robinson and he offers a more descriptive definition as "The conceptual model is a non-software specific description of the simulation model that is to be developed, describing the objectives, inputs, outputs, content, assumptions and simplifications of the model" [2]. This definition is important as it puts a light on conceptual model's being independent of the software specifications (languages, frameworks and platforms) and its aim of unveiling simulations' purposes, input outputs, content and defining the assumptions and simplifications made for the system.

Nance, supporting Robinson, states that conceptual models are separate from simulation execution. They have no dependency with the development platform [5].

Highlighting the role of conceptual modeling as a way of transforming simulation objectives into simulation capabilities, FEDEP (Federation Development and Execution Process) describes conceptual modeling as software independent abstraction of real world. Entities and their capabilities, interactions with others, constraints and assumptions are documented for ensuring the simulation to meet users' requirements. Also in FEDEP, conceptual models are mentioned to be forming the links between real world, requirements and design. It helps specifying the requirements from real world and constructing the design models from those requirement sets [6].

When we look at the definitions of conceptual modeling, we can list common key points in these definitions as;

- It is the simplified abstraction of real world
- It is separate from execution, implementation and development platforms
- It provides a means of communication by presenting a common language for users, domain experts and simulation developers
- It constructs the links between the real world, requirements and model

One might argue that do we really need to have conceptual models in modern simulation systems. Actually, the power of hardware is growing fast which results in bigger expectations from modern software. Demand for smarter software, that enables solving complex problems by utilizing distributed calculations, requires building effective conceptual models more than before.

Both Salt and Chwif explain the increasing complexity in simulation systems with "possibility" factor [7] [8]. While building, complexity and the size are not important concerns while building simulations any more. Processing power of computers is increasing fast and it results in enabling modelers to create complex models. Although advances in hardware bring faster simulation runs with complex calculations, it also causes excessive complexity in simulation models. Today, to cope with complexity of simulations, careful conceptual modeling is needed more than ever.

Conceptual modeling not only helps creating complex simulation models systematically, it also enables to define simulation objectives to draw the boundaries of the simulation. It helps to specify requirements by eliminating unnecessary details and minimizing the probability of moving onto further simulation development phases (design, implementation, verification and validation) with unclear and inconsistent requirements. It is claimed to be vital phase for enhancing the possibility of simulation's success [9].

Conceptual modeling is comprised of two levels of abstractions, namely mission space and simulation space. Both have different objectives in simulation study; however they have some common characteristics.

Pace defines mission space as the intention of simulation in its application area. Through the abstraction of real world, it forms first step by describing the important entities with specifying possible states, behaviors, characteristics and linkages with other ones. Likewise, he describes the simulation space conceptual modeling as the next level of abstraction. He highlights its objective as supporting the simulation development with building bridges between the requirements and specifications [10].

Aysolmaz explains the distinction of mission space and simulation space conceptual modeling. Mission space conceptual models consist of real world concepts and entities with related relations and capabilities. Therefore, it is mostly utilized in requirement analysis phase. On the other hand, simulation space conceptual models are simulation oriented. They involve simulation objectives along with operational and functional capabilities and they deal with the objectives, assumptions and limitations of the simulation. Consequently, simulation space conceptual modeling is mostly applied on simulation design activities rather than requirement analysis [11].

To sum up, we can list and compare the characteristics of mission and simulation space as in Table 1;

Table 1: Comparison of Mission Space and Simulation Space Conceptual Models

Mission Space Models	Simulation Space Models
Related with the representation of the simulation	Concerned with simulation control
Involve real world entities and processes	Define the entities and capabilities of simulation system
Closer to problem domain	Closer to solution domain
Exploited by subject matter experts	Exploited by simulation modelers and developers

Table 1 (continued)

Used in requirements development and analysis phases	Used in design and implementation phases

In this study, we have selected KAMA mission space models as the source of our transformation and BOM simulation space models as the target.

KAMA is a framework with modeling methods, notation and supporting tool. It was developed with an aim of developing mission space conceptual models and exploiting the usages of these conceptual models in simulation systems. It is UML based and generated models are simulation and implementation independent. It was also developed as a research project with the collaborations of academia, industry and military. KAMA framework is validated by case studies and real world simulation projects [12].

KAMA targets developing mission space models which of those represent the missions with their objectives in summation and real world entities with corresponding states. It is mostly utilized in requirements analysis and development phase and has very little information about the implementation of simulation models.

BOM (Base Object Model), devised by SISO, is a component design framework for efficient development of simulation models by promoting reusability, interoperability and composability. In BOM Specification Document, its objective is defined as "provide a mechanism for defining a simulation conceptual model and optionally mapping to the interface elements of a simulation or federation using HLA OMT constructs". This objective supports the ideas of rapid development of simulation models and federations by assembling the building blocks of simulation components [13].

In this study, we are mostly interested in the conceptual model definition part of the BOM specification. This portion describes the entities' characteristics along with their relations, possible states and interactions in between. Besides entity modeling, BOM also describes the patterns of interplay to be realized while developing the

simulation system. Main concerns of these representations are simulation system rather than real world. Furthermore, BOM conceptual models can be direct inputs of simulation design phase. As it is very close to simulation system, conceptual models defined in BOM are called as simulation space conceptual models.

1.2. The Problem

In literature, there are different studies that suggest methodologies and frameworks for developing either mission space or simulation space conceptual models. Although mission and simulation space models have much in common, there is no accepted study describing the way of transforming mission space into simulation space.

In a typical simulation study, simulation domain experts begin with analyzing the real world system to be simulated. They define the real world entities by listing their attributes and capabilities and find out how they interact with each other. Furthermore, they specify the objectives of the simulation and whether or how the simulation system will satisfy the user needs. While studying on these real world conceptual issues, they use some kind of modeling notation to represent the models for being inputs for further simulation development phases. As this conceptual analysis is mostly concerned with missions of the simulation, we call it as mission space conceptual modeling.

Completed the mission space conceptual modeling, domain experts let simulation designers to design simulation components. Simulation designers again start with the conceptual modeling, but this time, they develop the models for direct usage in system implementation rather than representing the real world. Since their main concern is documenting how simulation developers implement the simulation models, we call it as simulation space conceptual modeling.

Despite the fact that both mission space and simulation space modeling serve for different purposes, they hold similar information about simulation. As shown on Figure 1, both define the entities, task flows and states. Nevertheless, there is no

formal way of utilizing the conceptual modeling information of mission space in simulation space. For that reason, duplicate efforts are spent for forming the similar information. Moreover; to avoid inconsistencies, error prone manual synchronization of models is needed throughout whole simulation study.
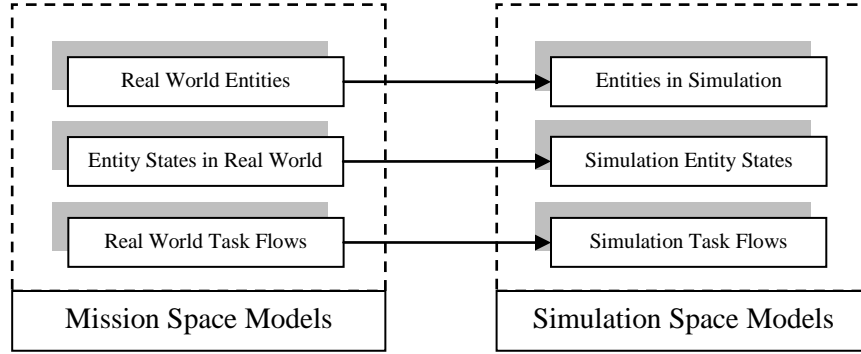
Figure 1: Mission and Simulation Space Models Sharing Similar Information

The problem of lacking transition of conceptual study findings from mission space into simulation space modeling can be depicted in Figure 2. Briefly, it shows that there is no defined methodology from promoting usage of mission space conceptual models in simulation space.
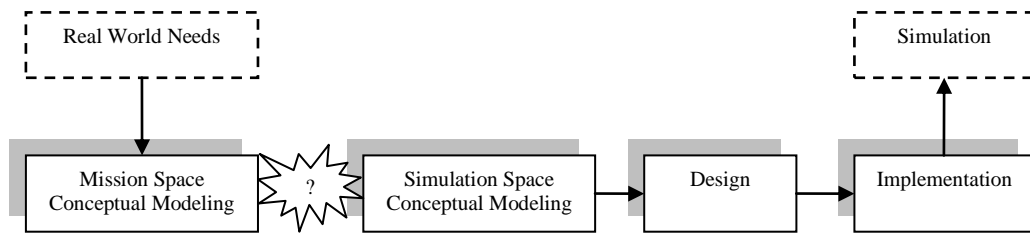
Figure 2: Problem of missing Linkage between Mission and Simulation Space

To sum up, the problems we aim to solve in this study are;

- Lack of a bridge between the mission and simulation space conceptual modeling.

- Usage of mission and simulation space modeling is degraded as it is very hard to establish and maintain manual links from mission space to simulation space.
- Same studies have to take place to represent similar conceptual information in both mission and simulation space.
- There is no formal means of communication between simulation domain experts and simulation designers.

1.3. Solution Approach

With the aim of solving problems mentioned in previous section, we propose an approach for transforming models of mission space in simulation space. In this study, we selected KAMA models as source of our transformation and BOM models as target.

Firstly, to identify which models may be transformed, we analyzed the metamodel of both KAMA and BOM. After such analysis, we have found several model mappings between KAMA and BOM as shown in Figure 3.
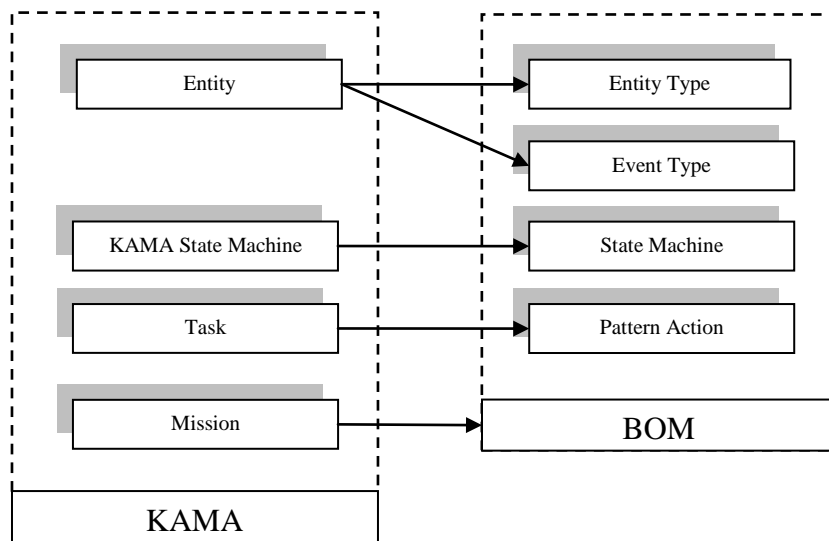
Figure 3: High Level KAMA and BOM Mapping

Having decided model mappings, we deeply analyzed the KAMA and BOM model attributes to be matched. Then we realized that, there are many model properties can be directly or indirectly mapped. Considering the fact that they are not same models having same purposes, of course there are some model properties with no match.

In short, we can conclude our solution approach as;

- Finding and documenting model matching between KAMA and BOM
- Guiding to use these matching for information transition from mission space to simulation space.

By applying this solution, we believe that we will be constructing a bridge between mission space and simulation space and automatically promote the information gathered in requirements analysis phase in simulation development.

Secondly, hence this study provides synchronization of models automatically, it will enhance the usage of conceptual modeling in both mission and simulation space. Changes in KAMA models will be automatically reflected to BOM models and then it will be great motivation for modelers to use conceptual modeling in simulation studies.

Third problem this study attacks is eliminating the duplicate efforts spent whilst developing conceptual models in KAMA and BOM. When developing BOM models, simulation designers will be using the domain analysis information formed in KAMA.

Finally it will constitute a formal language for simulation domain experts and simulation designers. The artifacts of domain experts will be converted to artifacts of simulation designers.

To show the applicability of transformation method, we will apply our rules in a real life example and evaluate the results. This application will be about modeling the

mission space conceptual models with KAMA notation and transforming these models into BOMs with executing defined transformation rules.

1.4. Outline of the Thesis

This thesis study consists of five chapters.

In chapter 1, the context of the study, problem we targeted and our solution approach is presented.

In chapter 2, background information and related studies are presented. In this section, previous studies for conceptual modeling and similar studies aimed handling mission and simulation space modeling is discussed.

Chapter 3 is the section in where we present our proposed solution. We detail our solution approach with first enlisting the matching fields of KAMA and BOM and then clarifying our transformation method.

Application of transformation method is provided in chapter 4. Also, findings and comments on these findings are discussed.

In chapter 5, final chapter, conclusion of overall study is evaluated. Finally limitations and possible future works are given.

CHAPTER 2

BACKGROUND

This chapter first introduces the previous studies about KAMA and BOM. Then in succeeding parts, related works about integrating mission and simulation space conceptual modeling are presented.

2.1. KAMA

Aiming to develop a conceptual modeling tool which provides a common modeling approach and a repository, KAMA project is devised by MODSIMMER; Modeling and Simulation Center of METU.

Karagöz summarized the project with three main components. First component of project is an UML based modeling notation that enables effective development instrument for conceptual modeling. Secondly, it presents a conceptual modeling process for getting best gain from mission space conceptual modeling phase. Final component of KAMA project is a tool supporting not only the notation but also conceptual modeling process [12].

2.1.1. KAMA Notation

Before forming the notation three design goals are specified to bring up an effective method of developing models of the mission space.

First goal is introducing a domain specific modeling notation designed for simulation domain experts and conceptual modelers. To achieve that goal, KAMA involves several diagrams and model elements specific to mission space conceptual modeling.

Reusing the UML elements wherever possible is the second goal of notation. Again to carry out that target, MOF based KAMA metamodel is extended from UML foundation package.

Last design goal to be achieved is making it simulation and implementation independent. Supporting this goal, KAMA contains no model element for implementation of simulation. That is why we call the models of KAMA as mission space models instead of simulation space.

In this study we are mostly interested in the metamodel of the KAMA notation hence we use KAMA metamodel elements for finding valid mappings to BOM metamodel.

In Figure 4, all KAMA metamodel elements and their relations are depicted.

Figure 4: KAMA Metamodel Elements [12]

KAMA notation built upon three packages; mission space, structural and dynamic behavior. In all packages there are several model elements and diagrams to define notation. Details of elements and diagrams will be presented in following sections.

a. Mission Space Package

Mission space package contains elements to model mission and task details in conceptual models. The main concern of this package is representing the high level missions of simulations.

In mission space package there are two diagrams to model missions and task flows. Mission space diagram is used to depict high level missions along with objectives

and measures. Another diagram to model missions in detail is task flow diagram. In this diagram, sequence of task execution, conditional branches and parallel flows can be defined for better perception of missions.

Model elements used in mission space package are presented in following parts.

1) Mission

Mission represents the high level requirement that simulation should meet. It is generally realized by sequence of tasks in task flow diagrams. Attributes of missions listed in Table 2.

Table 2: Mission Attributes

Attribute	Type	Detail
Input List	List of Work Product	Work Products which are used by mission
Output List	List of Work Product	Work Products which are produced by mission
Preconditions		Conditions which are met before executing the mission
Post Conditions		Conditions that occur after executing the mission
Objective List	List of Objective	Objectives that are achieved by the mission
Role List	List of Role	Roles who are responsible or who realizes the mission
Measure List	List of Measure	Measures which are quantified by the mission objective
Notation	«Mission»	

2) Task

Task is used to detail subsequent parts of mission. Sequence of tasks forms the mission of conceptual model. Attributes of tasks are similar to mission and listed in Table 3.

Table 3: Task Attributes

Attribute	Type	Detail
Input List	List of Work Product	Work Products which are consumed by task
Output List	List of Work Product	Work Products which are produced by task
Preconditions		Conditions which are met before executing the task
Post Conditions		Conditions that occur after executing the task
Objective List	List of Objective	Objectives that are achieved by the task
Role List	List of Role	Roles who are responsible or who realizes the task
Measure List	List of Measure	Measures which are quantified by the task objective
Is Extension Point	Boolean	Marks the task if it is an extension point
Extension Point ID	Number	Identifier number of extension point if task is marked as extension point
Notation	«Task»	

3) Role

Roles represent actors who are responsible or realize the connected mission or task. It can whether a human being or a system component in simulation. Notation of roles is shown in Table 4.

16
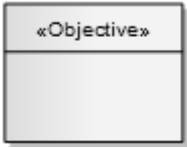
Table 4: Role Attributes

Attribute		
Notation		

4) Objective

Objectives denote the aim of mission or task. They are quantified by the connected measures. In Table 5, attributes of objectives are given.
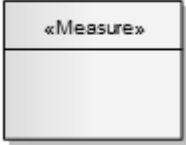
Table 5: Objective Attributes

Attribute	Type	Detail
Measure List	List of Measure	Measures which indicates the performance of objective
Performance Criteria		Criteria or formula to be used to calculate performance
Notation	«Objective»	

5) Measure

Measure is a performance indicator for objective. Attributes of measure is listed in Table 6.
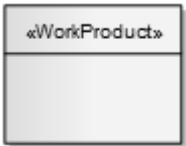
Table 6: Measure Attributes

Attribute	Type	Detail
Unit		Quantifiable unit of measure
Notation	«Measure»	

6) Work Product

It represents the entities which are whether inputs to or outputs from missions and tasks. Work product notation is shown in Table 7.

Table 7: Work Product Attributes

Attribute		
Notation	«WorkProduct»	

b. Structure Package

Structure package is used to model static components of conceptual model. Model elements are generally reflection of real-world entities in conceptual model.

There are four type of diagrams can be used in structure package. Although first two of them (ontology and relationships diagrams) semantically same, they have different purposes in conceptual modeling. Entity ontology diagrams are used for modeling the entities and relations which can be shared with other simulation systems, however entity relationships defines models specific to mission. Also in entity relationship diagrams, associations with work products can be modeled. Command hierarchy diagram is the third diagram used in structure package. As name implies, in this diagram, hierarchy of commands associated with the actors can be depicted. Last

diagram of structure package is organization structure. In this diagram, the relationships of actors and roles are defined.

Model elements of structure package are given in subsequent sections.

1) Entity

Entity is used to represent any kind of entity in real-world. Metamodel details of entity are given in Table 8.

Table 8: Entity Attributes

Attribute	Type	Detail
Attributes	List of KAMAAttribute	List of entity characteristics
Capabilities	List of KAMACapability	Capabilities of entities which specifies what can related entity do in real-world
Association List	List of Association	Associations of entities
Notation	«entity»	

2) Actor

Inherited from entity, actor element models the active elements in mission space. They are used in command hierarchy diagrams and details are presented in Table 9.

Table 9: Actor Attributes

Attribute	Type	Detail
Role List	List of Role	List of roles which can be connected to missions and tasks.
Notation		

c. Dynamic Behavior Package

Focusing on complex dynamic behaviors, this package describes the possible states of entities and transitions in between.

Almost all of the entities in mission space have dynamic behaviors. With the help of KAMAStateMachine diagram, those dynamic behaviors can be modeled with state nodes and transitions from one node to another.

Model elements of dynamic behavior package are generally focused on state machine concept and listed below.

1) KAMAState

KAMAState is used to represent the nature of an entity under some specified circumstances. Throughout lifetime of an entity, KAMAState can model any possible states which describes whether satisfying a condition, performing an activity or waiting for a triggering event.

In KAMAStateMachine, state transitions are started with InitialState and ends with FinalState nodes. InitialState node has no incoming transitions whereas FinalState node has no outgoing one.

Attributes of KAMAState model element are given in Table 10.

Table 10: KAMAState Attributes

Attribute	Type	Detail
Is Simple	Boolean	Specifies if the state is a complex or simple one. Unlike UML State element, KAMAState can only represent simples so this attribute is always set to true.
Is Submachine State	Boolean	Specifies if the state is a complex or simple one. Unlike UML State element, KAMAState can only represent simples so this attribute is always set to false.
Notation	«State» «InitialState» «FinalState»	

2) Transition

For modeling the changes of entity states, connections between KAMAState nodes are represented by Transition model element. Showing the changes from source state to target, Transitions are defined by a triggering event, a guard condition and an action. These attributes are detailed in Table 11

Table 11: Transition Attributes

Attribute	Type	Detail
Triggering Event		The event that initiates the state change.
Guard Condition	Boolean	Before leaving one state and advancing to another, guard condition is evaluated and required transition takes place if only guard condition is satisfied.
Action		The action which takes place after state change is applied.
Context Mission	Mission	The mission in which the transition is valid.

Table 11 (continued)

Notation	
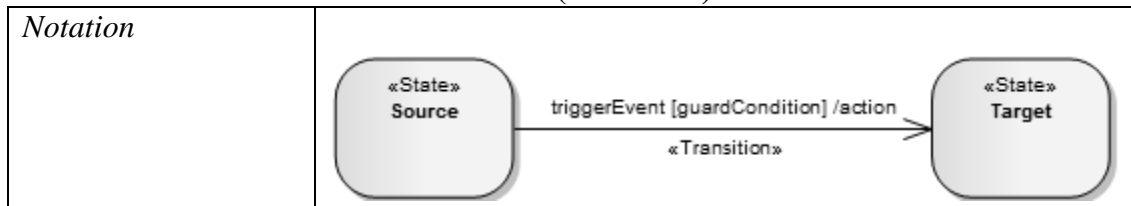
2.1.2. KAMA Conceptual Modeling Process Definition

To help modelers for constructing effective conceptual models, conceptual modeling process is introduced as second component of KAMA project. Modeling process is defined with sequence of tasks and their inputs and outputs.

1) Acquire Knowledge About the Mission Space

Process of conceptual modeling is started with getting information about the related mission space. It is very crucial to get enough knowledge about the domain in which the simulation will be developed.

This phase is defined with several sub steps in KAMA.

- Identifying the objectives,
- Specifying the context,
- Listing the source of authoritative information

2) Define Context

This second phase is the iterative construction phase of recommended conceptual modeling process. Mission space model elements and diagrams including missions, roles, objectives, measures and the relationships among them are produced throughout this phase.

Every mission in this phase is detailed by task flow diagrams. They are all related with an objective and achievement of this objective is determined by connected quantifiable measures.

3) Develop Content

After the phase 2 is completed, the behavioral features are shown by task flow and entity-state diagrams. Besides, the structural features are modeled with entity-ontology, entity relationship, command hierarchy and organizational structure diagrams.

4) Verify and Validate Conceptual Model

Joint review with sponsor and domain experts can take place to validate the finalized conceptual model. Results of this review are reported to be resolved in next phase.

5) Update the Conceptual Model According to V&V Results

Final step of recommended conceptual modeling process is reflecting the findings of V&V activities in final conceptual model elements and diagrams.

2.1.3. KAMA Tool

Third and last component of KAMA project is a supporting tool to enable modelers for developing conceptual models of the mission space. The high level functional requirements of tool are depicted in Figure 4.
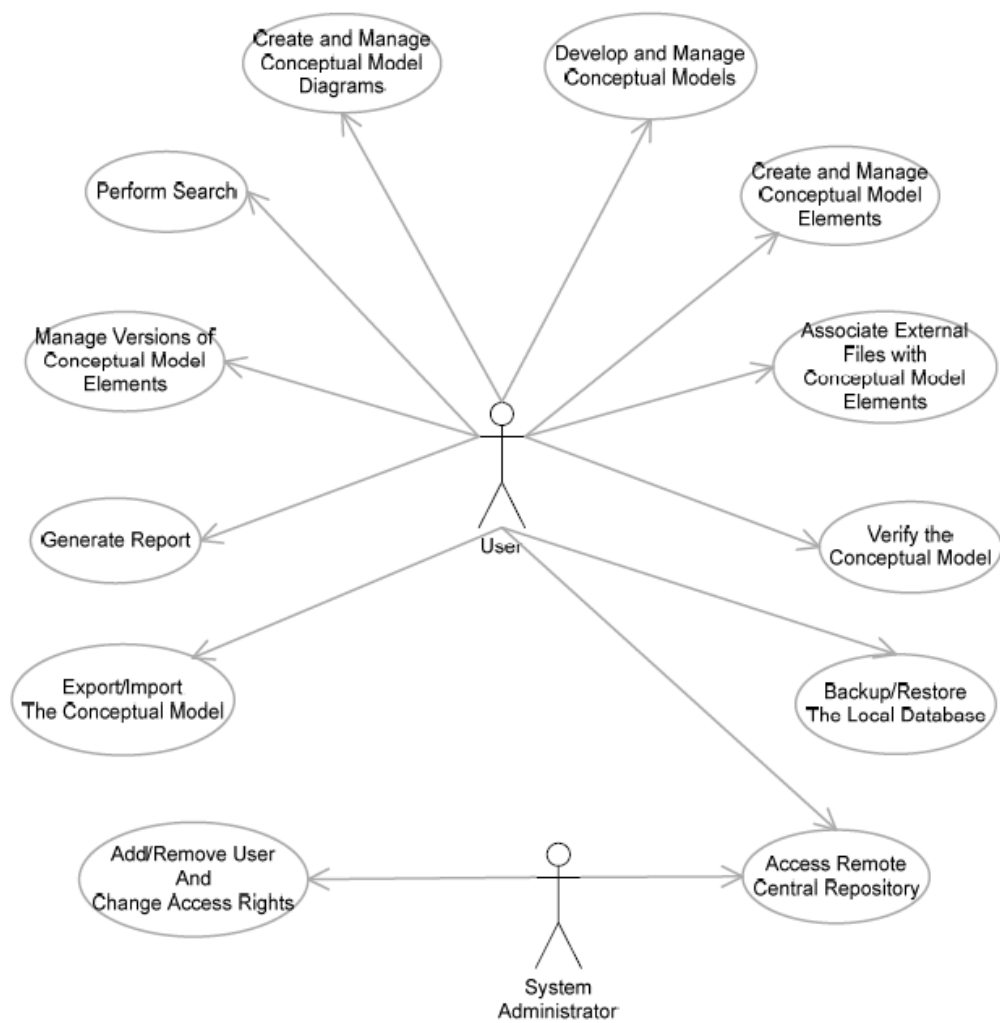
Figure 5: Requirements of KAMA Tool [12]

2.2. BOM

Having focus on interoperability of simulation space conceptual models, BOM is an open standard by SISO. The rationale behind BOM methodology is rapid development of simulations with enabling composability and interoperability. This idea is depicted in Figure 6.
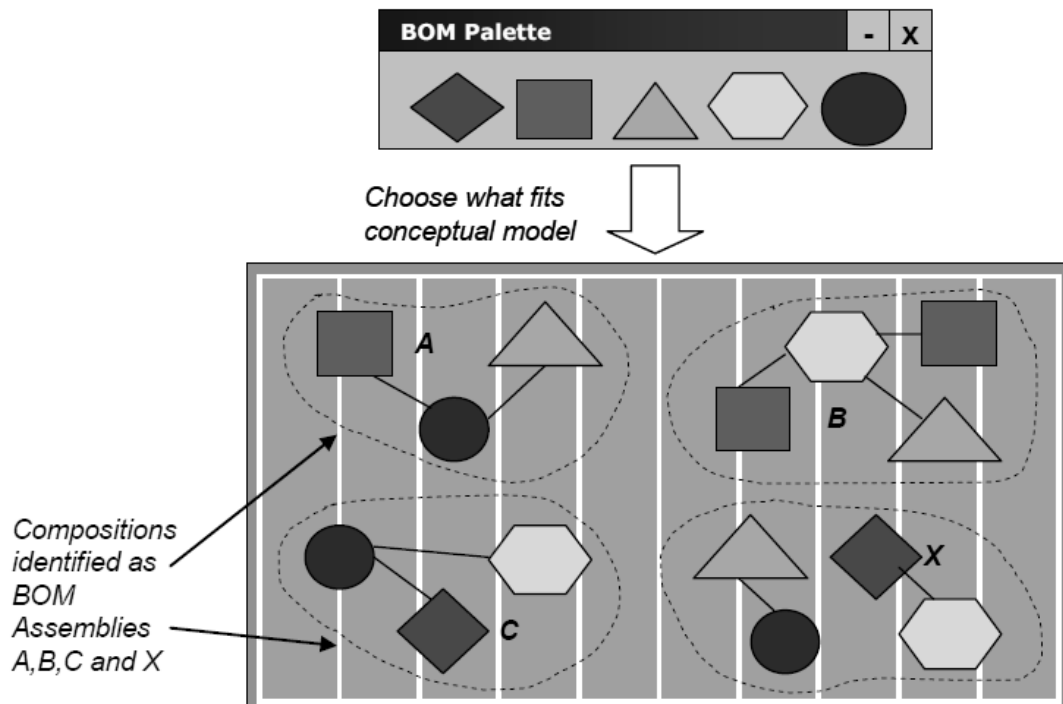
Figure 6: BOM Composition [14]

Like KAMA, BOM methodology defines the models with metadata based representations. High level composition of BOM metamodel is given in Figure 7. The part which this study is mostly interested is the Conceptual Model Definition part. It constitutes the target artifact of proposed transformation method.
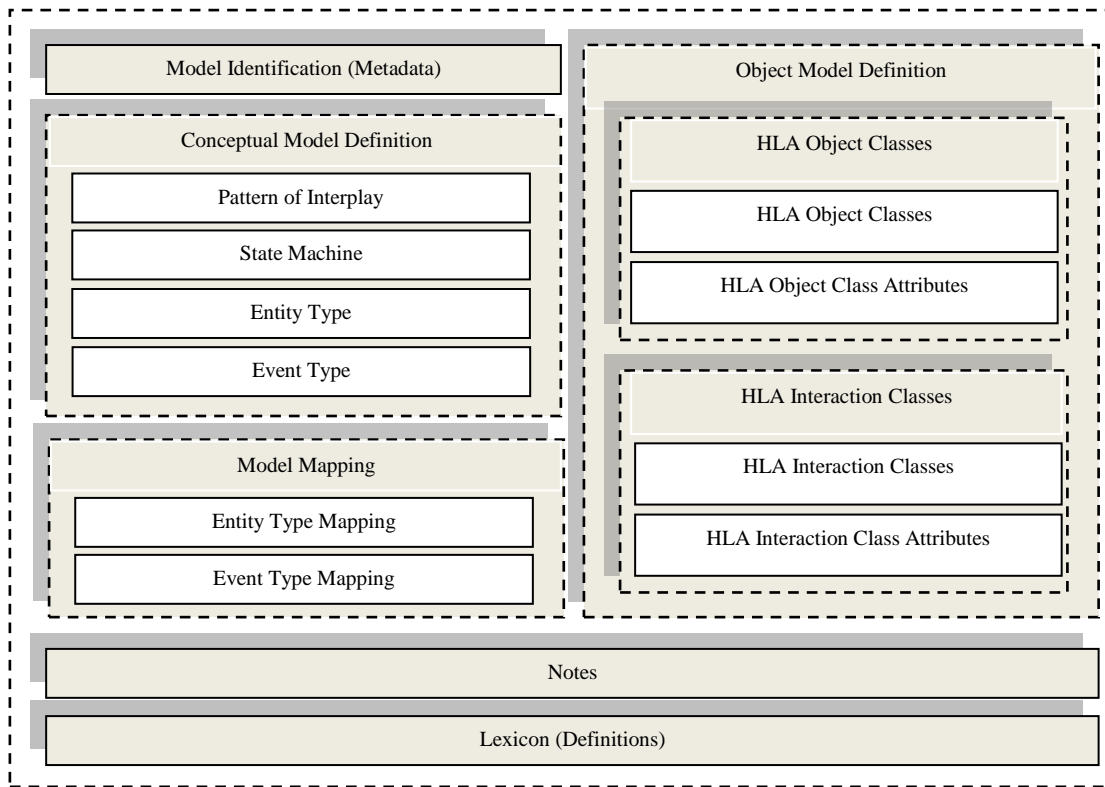
Figure 7: BOM Components

In BOM Template Specification document, high level metamodel components of BOM and these components' associations with each other are depicted in Figure 8. Using this figure, which components are required and which can be omitted is deducted by means of multiplicity identifiers. This deduction has a big significance in our study as it shows us that Conceptual Model Definition must have at least one or more Pattern of Interplay, State Machine, Entity Type and Event Type component.
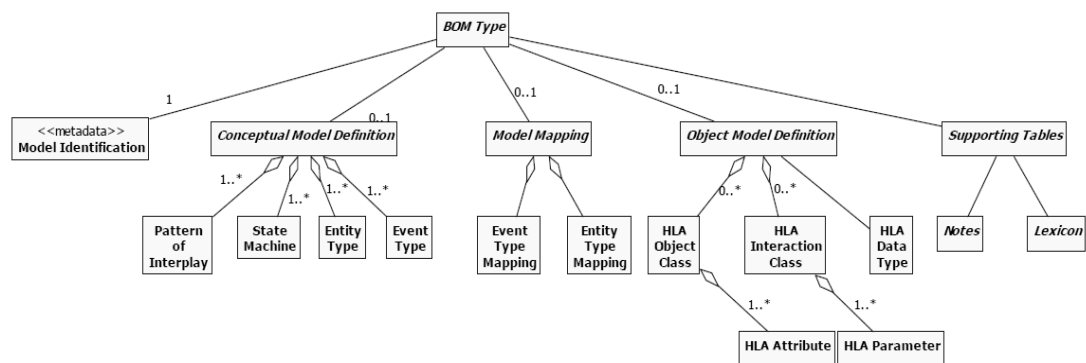
Figure 8: BOM Template Components [13]

26

Fields of BOM components, which are primary outputs of this study, are detailed in following sections.

2.2.1. Model Identification

Model Identification section of BOM documents describes the general information about the model like, what it simulates, its purpose, how it can be found for reuse and who the point of contact to be communicated is. Importance of this identification lies under the information it holds about the use limitation and restrictions. This portion of BOM has to be checked before of using this model in other simulations.

In SISO's Guide for BOM Use and Implementation document, purpose field of Model Identification component is defined to be reflective of simulation objectives [14].

2.2.2. Conceptual Model

Conceptual modeling part of BOM describes the simulation space conceptual model via pattern of interplay descriptions, state machines, entity and event specifications. This is the part which our transformation method tries to form automatically. Therefore our intention towards this section will be high compared to other parts of the model.

Conceptual modeling in BOM is closely related with 1st and 2nd steps of FEDEP. Conceptual model, developed within BOM methodology, provide descriptions about what the simulation will present together with limitations and assumptions and other capabilities needed to satisfy requirements [6].

In following subsection, all fields of conceptual modeling components (pattern of interplay, state machine, entity type and event type) will be presented in details.

1) Pattern of Interplay

Listing variations and exceptions, Pattern of Interplay component describes the pattern actions which are aligned with other BOMs or event types. Providing actions in a list and enabling modeler to assign sequence numbers to actions, it helps to figure out task flows for accomplishing simulation objectives. It is also useful to model the interactions between conceptual entities by identifying the sender and receiver of associated pattern actions.

Element relationships of pattern of interplay are depicted in Figure 8.
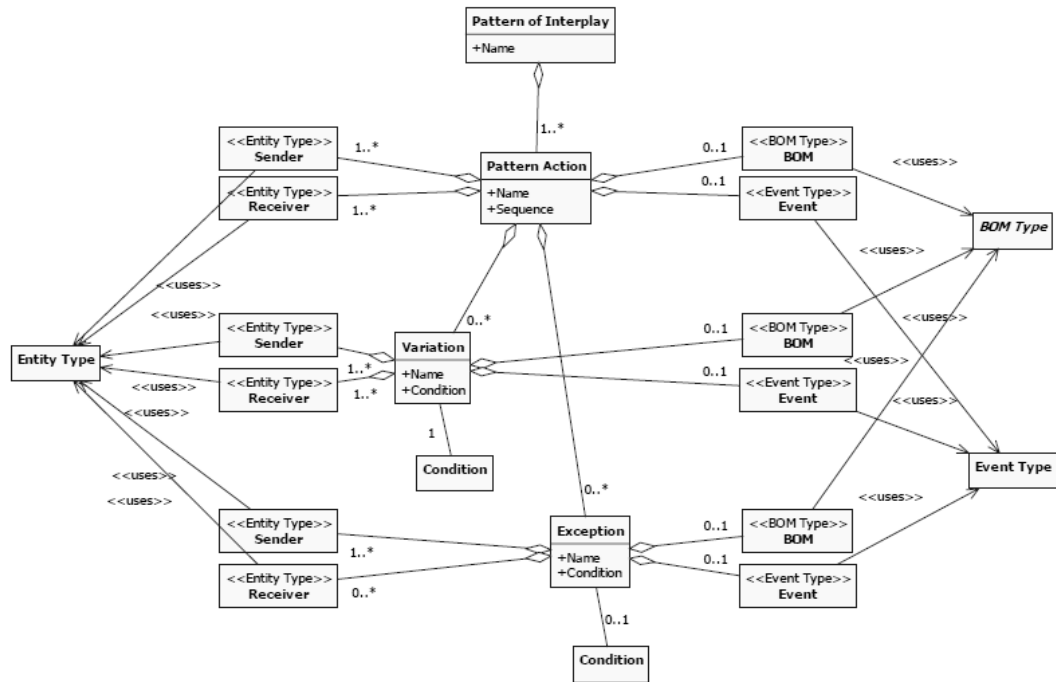
Figure 9: Pattern of Interplay Relationships [13]

Field details of Pattern of Interplay component is listed in Table 12, Table 13, Table 14 and Table 15.

Table 12: Pattern of Interplay Attributes

Attribute	Type	Detail
Name	Text	Name of the pattern of interplay.
Pattern Action	List of PatternAction	An action, within series of action, to accomplish named pattern of interplay. Pattern of interplay has to have at least one pattern action.

Table 13: Pattern Action Attributes

Attribute	Type	Detail
Sequence	Number	Specifies the order of sequence in pattern of interplay
Name	Text	Name of pattern action.
Sender	List of EntityIdentifier	Identifies the conceptual entities which trigger the action. Pattern action has to have at least one sender.
Receiver	List of EntityIdentifier	Identifies the conceptual entities which receive the interaction of senders. Pattern action has to have at least one receiver.
Event	EventTypeIdentifier	Identifies the event type for related activity. Pattern action may have no event if it is realized by another BOM.
BOM	BOMIdentifier	Identifies another BOM that realizes the action. Pattern action may have no BOM if it is realized by an event in current BOM.
Exception	List of Exception	Specifies the exceptional flow. Pattern action may have not any exception if has no exceptional branches.
Variation	List of Variation	Specifies the alternative way of accomplishing the pattern action.

Table 14: Exception Attributes

Attribute	Type	Detail
Name	Text	Name of exception.
Sender	List of EntityIdentifier	Identifies the conceptual entities which trigger the action's exceptional flow. Exception has to have at least one sender.

Table 14 (continued)

Receiver	List of EntityIdentifier	Identifies the conceptual entities which receive the interaction of senders. Exception may have no receivers.
Event	EventTypeIdentifier	Identifies the event type for related exception. Exception may have no event if it is realized by another BOM.
BOM	BOMIdentifier	Identifies another BOM that realizes the action's exceptional case. Exception may have no BOM if it is realized by an event in current BOM.
Condition	List of Boolean	Specifies the condition which has to be satisfied for initiation of the exceptional flow. Exception has to have at least one condition statement.

Table 15: Variation Attributes

Attribute	Type	Detail
Name	Text	Name of variation.
Sender	List of EntityIdentifier	Identifies the conceptual entities which trigger the action's variation flow. Variation has to have at least one sender.
Receiver	List of EntityIdentifier	Identifies the conceptual entities which receive the interaction of senders. Unlike exception, variation has to have at least one receiver.
Event	EventTypeIdentifier	Identifies the event type for related variation. Variation may have no event if it is realized by another BOM.
BOM	BOMIdentifier	Identifies another BOM that realizes the action's variation. Variation may have no BOM if it is realized by an event in current BOM.
Condition	List of Boolean	Specifies the condition which has to be satisfied for initiation of the variation flow. Unlike exception, variation may have no condition.

2) State Machine

State Machine component is used to model dynamic behavioral states of conceptual entities. All the possible states in which the entities can be and which action(s) trigger the state change shall be identified via State Machine. Also, the condition which must be satisfied before the state change takes place is allowed in State Machine. State Machine gives a clear understanding for simulation developers to get the information of entities possible states during simulation run.

Relationships between State Machine elements are depicted in Figure 10.
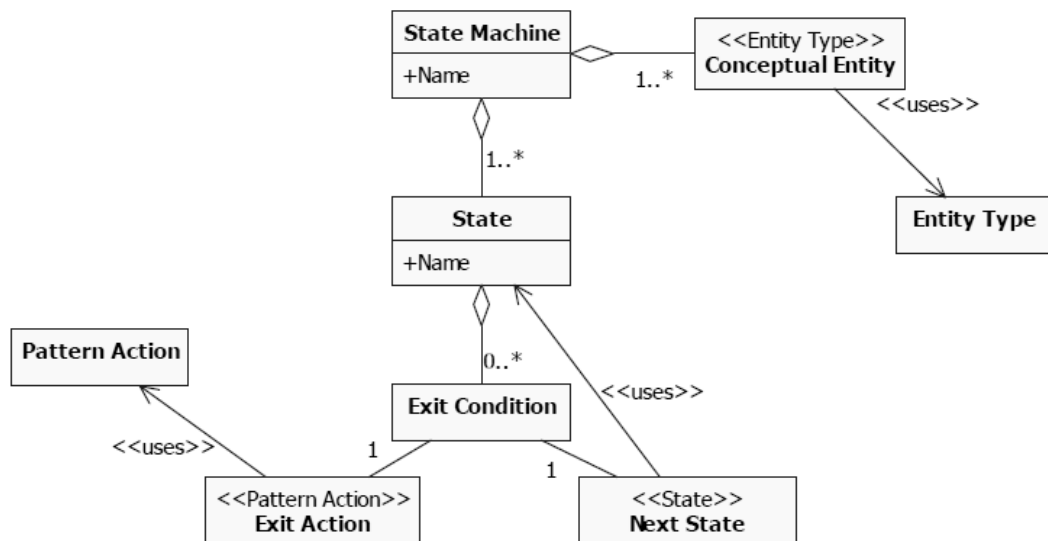
Figure 10: State Machine Relationships [13]

State Machine attributes are detailed in Table 16, Table 17 and Table 18.

Table 16: State Machine Attributes

Attribute	Type	Detail
Name	Text	Name of state machine.
Conceptual Entities	List of Conceptual Entity	Identifies the conceptual entities which can be in associated states. State machine has to have at least one conceptual entity.

Table 16 (continued)

State	List of State	Identifies the states which may belong to a conceptual entity of state machine. State machine has to have at least one state.

Table 17: State Attributes

Attribute	Type	Detail
Name	Text	Name of state.
Exit Condition	List of ExitCondition	Specifies the condition which has to be satisfied for initiation of state change. State may have no exit condition.

Table 18: ExitCondition Attributes

Attribute	Type	Detail
Exit Action	PatternAction	The action in which the exit condition has been satisfied. Exit action must be aligned with a pattern action.
Next State	State	Specifies the succeeding state when the exit condition is satisfied.

*3) **Entity Type***

In BOM, simulation conceptual entities which drive the pattern of interplays and which hold the states defined in state machines are represented by Entity Types. They are actually reflections of entities to be used in simulation system. Entity types can be a real world entity, concept, system or process.

Relationships of Entity Type element are depicted in Figure 11.
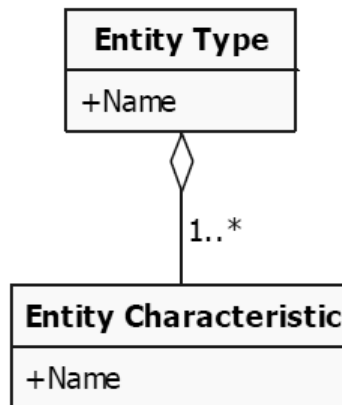
Figure 11: Entity Type Relationships [13]

Attributes of entity type model element are also given in Table 19 and Table 20.

Table 19: Entity Type Attributes

Attribute	Type	Detail
Name	Text	Name of entity.
Characteristic	Characteristic	Specifies the attributes of conceptual entities. Entity type has to have at least one characteristic.

Table 20: Characteristic Attributes

Attribute	Type	Detail
Name	Text	Name of characteristic.

4) Event Type

All the conceptual events occurred in simulation space conceptual model can be expressed with Event Types. While defining the pattern of interplay, pattern actions along with variations and exceptions are modeled with event types.

In BOM methodology, there can be two types of event types: trigger and message.

- BOM Trigger: They are the event types which in general show the state change of conceptual entities. The distinction about triggers is having no

33

target characteristics. Sender entity does not care about the receiver of published event.

- BOM Message: Message event type is a directed event having both source and target characteristics. Sender publishes the event to a specified receiver entity.

Event type element relationships are given in Figure 12.
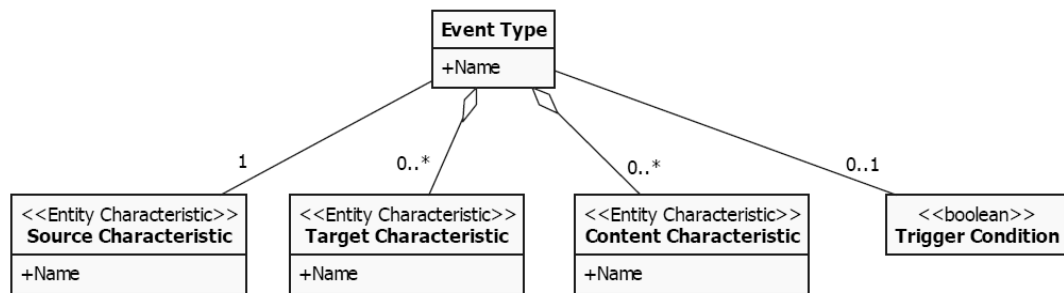
Figure 12: Event Type Relationships [14]

Details of its fields are listed in Table 21.

Table 21: Event Type Attributes

Attribute	Type	Detail
Name	Text	Name of event.
Source Characteristic	Characteristic	Specifies the sender of event. It is a reference to characteristic of entity type.
Target Characteristic	Characteristic	Specifies the receiver of event. It is a reference to characteristic of entity type. Event type may have no target characteristic.
Content Characteristic	Characteristic	Specifies the content of event. It is a reference to characteristic of entity type.
Trigger Condition	Boolean	Condition which triggers the publishing of event. Event type may have no trigger condition.

2.2.3. Model Mapping

This component of BOM is used for mapping the simulation conceptual entities and events with HLA OMT constructs. It is the key component when moving from conceptual analysis phase into simulation software design and implementation phases.

BOM enables modelers to construct mappings for entity and event types.

- Entity Type Mapping: It is the mapping of conceptual model entities with HLA OMT class structures. Briefly, entities are mapped with HLA object and interaction classes while entity characteristics are mapped with attributes of these objects and interactions.
- Event Type Mapping: Likewise, event type mapping is used for mapping the conceptual events with HLA objects and interaction classes. However, in event type mapping, these HLA constructs are also mapped with source characteristics, target characteristics, content characteristics and trigger condition of event types.

2.2.4. Object Model Definition

Fully conforming to HLA OMT, Object Model Definition component of BOM includes definitions about simulation elements in terms of HLA objects and interactions. This part of BOM supports the simulation design phase directly as the resultants are ready to be used in development of simulation systems.

2.3. Previous Efforts

In past, to overcome the problem of duplicate representations of mission and simulation space models, efforts are spent for unifying the conceptual modeling that supports both mission and simulation space.

There are two studies that proposes new methods to extend existing methodologies to support both mission and simulation space.

First study is aimed to extend KAMA methodology to support simulation space modeling and second one's objective is extending BOM to support mission space modeling.

2.3.1. An Extended Methodology for KAMA

Aysolmaz introduces problem statement of her thesis study as; although KAMA provides solid mission space conceptual modeling methodology, it does not support any mechanism to support simulation space. What's more; she states that there is no study explaining how to benefit conceptual modeling artifacts in design [11].

Identifying the problems above, she proposed an extension for KAMA to support simulation space conceptual modeling and a method to utilize the conceptual modeling constructs in simulation design phase. Instead of devising a new methodology to support mission and simulation space together, she built her solution on KAMA framework as an extension. Because she believes that KAMA has a strong conceptual modeling infrastructure and provides a ready to use method.

Specifications of her extensions and the rationale behind them can be summarized as:

- Two new steps added to KAMA conceptual modeling process which are "develop simulation space conceptual modeling diagrams" and "develop high level design" respectively. These steps are used to embody extensions into modeling process of KAMA.
- "Kind" property is added to entity element for specialization in different parts of the simulation.
- Behaviors and attributes of entities are classified as whether simulation or mission space. So modelers are able to define separate mission and simulation space models.

- Modelers can use "unit" properties of attributes to avoid inconsistencies.

- Initial and possible values as can be provided to be used in simulation design.

- Attributes can be defined to be whether "fixed" or "variable". So in design phase, variable types can be specified (like derived, static, constant vs.).

- A new "object" element is introduced. It is used to model specific instantiations of entities.

- To detail the dynamic behaviors in conceptual model, "if", "when" and "while" conditions are added to "mission" and "control flow" elements and disjoint property is included in "control flow".

- Sub-task concept is developed to enable modeling the hierarchy between tasks.

- To model the processes which can be shared between more than one entities, "algorithm" element is added.

- With an aim of utilizing conceptual modeling in design, quantity and role attributes are added to generic relationship.

- To help developers building well formed elements, a guide is presented for extracting elements via requirement statements.

- A new activity is added into process to classify model elements as mission space or simulation space conceptual modeling element. So, modelers will be enabled to develop mission and simulation space diagrams separately.

- "Entity Ontology" diagrams are modified to support environment conceptual model. This modification lets the environment to be used in simulation design.

- Five conceptual modeling diagrams used in mission space are also used in simulation space, but this time having different perspectives. In this way, concepts, regarding simulation design and construction, can be modeled conceptually.

- Finally, new process step is introduced as "Develop High Level Design". This phase is vital for transporting mission space models to simulation design. Also at the end of this process, simulation designers will be having a base for critical design artifacts. This extension is one of the most important

improvements to construct bridges between the phases of mission space conceptual modeling and simulation design.

To prove the effectiveness of her study, Aysolmaz conducted a case study in which she models the both mission and simulation space elements for synthetic environment simulation. At the end of her study, she concludes with the findings of this case study.

It is experimented that entity ontology diagrams can be used to model simulation software entities and utilities. Likewise, organization structure diagrams are utilized to depict the roles in simulation system and entity state diagram is benefited from representing the possible states of simulation software system. In addition, entity relationship diagrams served for their objectives of modeling the simulation entities to show how they work on facilities. Finally, all the tasks in simulation are shown on task flow diagrams with their corresponding roles and input outputs.

She not only shows that the extension diagrams can be promoted to support simulation space but also she proves conceptual modeling phase artifacts can be utilized in simulation design. The justification she rely on is; at the end of conceptual modeling process, the resultant diagrams are mature enough to be inputs into simulation design phase.

2.3.2. Enriching BOM to Support Mission Space Models: BOM++

Attacking the same problem of being incapable of representing both mission and simulation space, Mojtahed proposed enriching BOM to support mission space conceptual modeling beside its primary support of simulation space [15]. Although he tried similar solution of extending a methodology to support both mission and simulation space, unlike Aysolmaz, he suggests semantically enriching simulation space modeling method to serve also for mission space.

Due to the fact that this study is emerged from the need of DCMF end product formalization, we want to give brief information about DCMF before explaining the details of BOM++.

After DMSO, with an unknown reason, has ended the research studies about the development of Conceptual Models of The Mission Space (CMMS) [16], Swedish Defense Research Agency (FOI) has decided to research on this concept in 2002. Having seen the unfinished and ambiguous components of CMMS, they have started a new project which they call DCMF, Defense Conceptual Modeling Framework.

In DCMF document, the objective is expressed as "to capture authorized knowledge of military operations; to manage, model and structure the obtained knowledge in an unambiguous way and to preserve and maintain the structured knowledge for future use and reuse" [17].

The end products of DCMF, which needs better formalization, are conceptual model elements of real world entities, environmental factors and processes and these elements are the foundations of missions' critical entities and interactions [18].

In this study, it is observed that the knowledge representation requirements of DCMF can be only partially supported by BOM. It can meet only the interoperability, reusability, composability and shareability requirements of DCMF. Therefore; extending BOM with new capabilities is offered to support other requirements as well.

To extend BOM, they propose two possible approaches; first one is adding semantics for changing the structure incrementally and the second is applying larger changes for including the ontology conceptualization of DCMF.

In the context of approach 1, they follow two aspects of "level of quality" and "improved reusability" to enrich original specification of BOM.

They realized that, unlike DCMF process which produces reusable and composable mission space models with level of quality attribute, there is no definition that describes the level of quality for BOM products in BOM standard documentation. To eliminate this problem, they offer adding a new metadata attribute (ProductionProcess) in BOM model identification table which refers the formal DCMF process by which the BOM was designed.

For the sake of "improved reusability", they believe BOM should have such constructs that support mission space ontology. Then, they offer extending the "EntityType" model element of BOM with new attributes (dcmfRelation and dcmfEntity) that enables for referring to DCMF mission space conceptual model ontology.

In contrast to approach 1, major changes are made in approach 2 to use the ontology for knowledge representation and semantic interoperability. They convert the xml document of BOM into ontology language in first step with OWL (Web Ontology Language) [19], and then in second step they extend the ontology to capture domain aspects by OWLS (Web Ontology Language Service) [20].

In step 1, they offer making the changes below to build BOM ontology:

- Removing the object instances of "sender" and "receiver" in BOM and introducing single "entity" with the metadata attributes of "isSender" and "isReceiver".
- Adding new characteristics ("action-resource", "context") to model for supporting domain specific concepts.

Finally in second step, following changes are proposed to add behavioral aspects to BOM:

- Instead of class "next state", semantic aspect of next state is added.
- Adding the "consistOf" relation with removing the aggregation of states.

- "Conceptual entity" is replaced with "has conceptual entity".
- For synchronization of "pattern action" and "entity type", "has sender" and "has receiver" attributes are introduced.
- "Exit condition" is modified to refer another "pattern action".

The final proposed BOM template applying the second approach can be shown graphically on Figure 13.
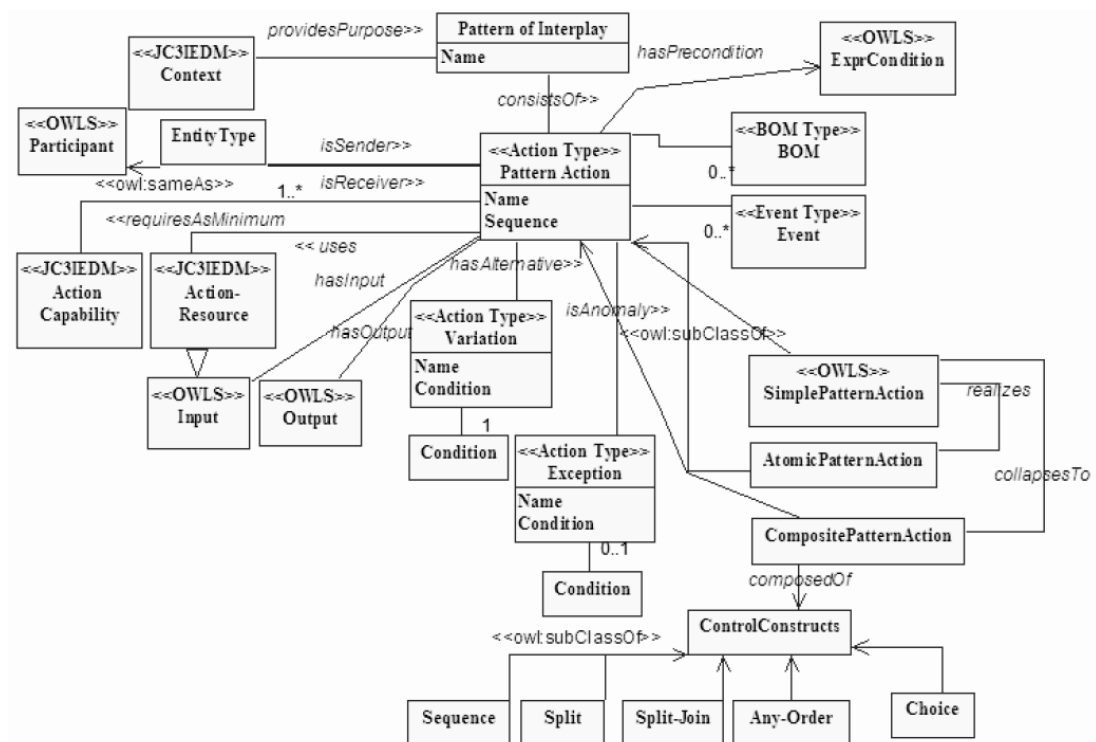
Figure 13: BOM++ [15]

To conclude, applying whether first or second approaches to extend BOM, they claim BOM will be supporting mission space conceptual models beside the constructs of simulation space by referring to or encapsulating the mission space ontology.

2.3.3. Comparison of Previous Works and Our Proposed Solution

In Figure 14 the problematic situation of having no relationship is depicted. Mission space conceptual models are developed with KAMA and simulation space conceptual models are developed with BOM. There is no way of using the models in mission space in simulation space.
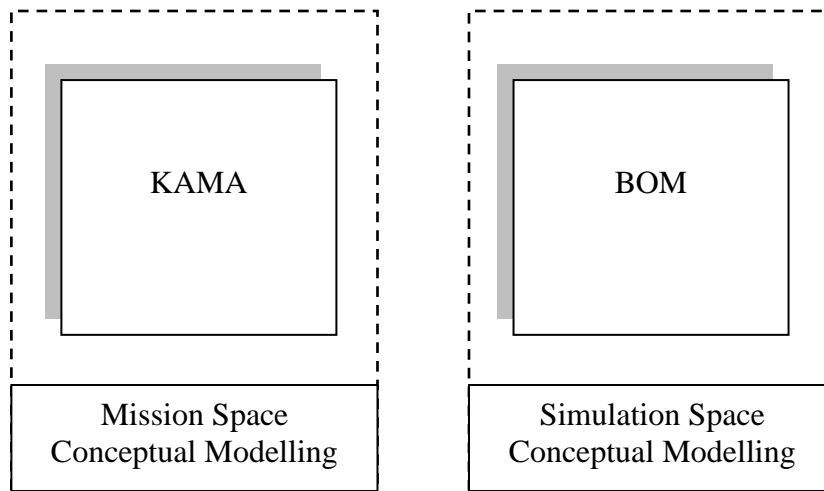
Figure 14: No Relationship between Mission and Simulation Space

For solving the problem, Aysolmaz proposes extending KAMA to also support simulation space [11]. She offers to make extensions to KAMA for enabling the development of simulation space models as shown on Figure 15.
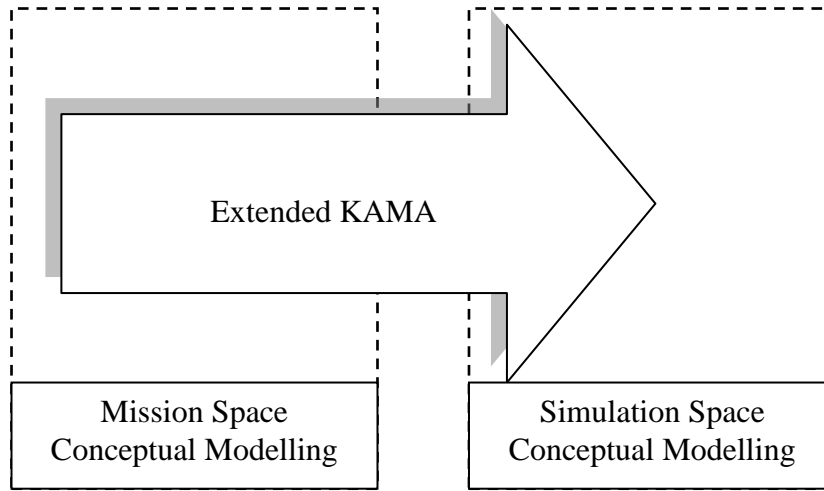
Figure 15: Extending KAMA to Support Simulation Space

As depicted on Figure 16, aiming to solve same problem, Mojtahed presents his solution as enriching BOM semantically to make it not only model simulation space conceptual models but also mission space [15].
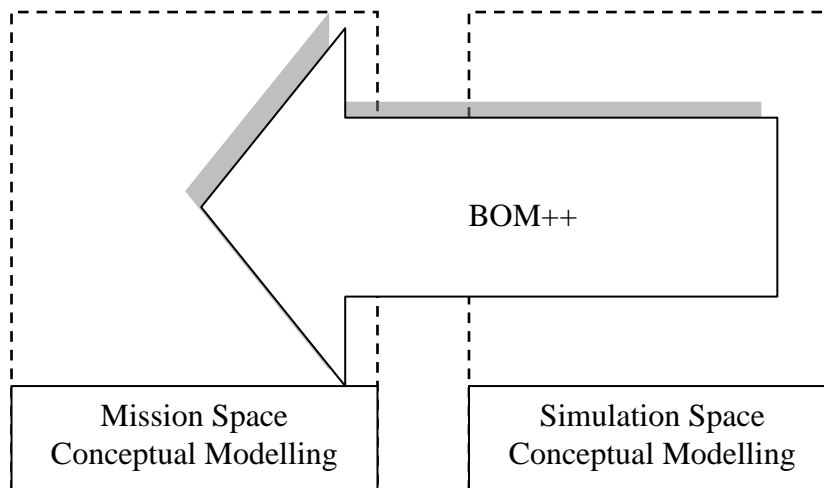
Figure 16: Extending BOM to Support Mission Space

Aysolmaz and Mojtahed realized that current tools and frameworks are incapable of covering whole conceptual modeling process, so they proposed to unify it with extending the current methodologies to embrace both mission and simulation space.

Unlike them, as shown on Figure 17, we offer to transform mission space models into simulation space by mapping the fields of corresponding models.
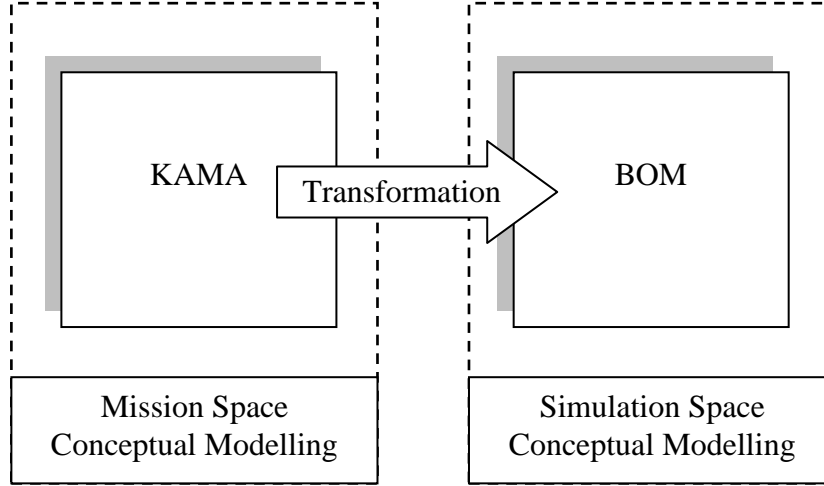
Figure 17: Our Proposed Solution

The fundamental advantage of our solution over two others is that we continue to separate problem domain from solution domain in contrast to trying to cover both. However, by unifying the mission and simulation space methodologies, Mojtahed and Aysolmaz also unify the problem and simulation domain along with their stakeholders. According to their approaches, subject matter experts should also consider the models specific to simulation design and simulation designers should also model the real world entities and concepts.

By applying our solution approach of transformation, subject matter experts use KAMA tool to model real world concepts in problem domain and in solution domain, simulation designers use BOM to model simulation concepts. Consequently, the abstraction between mission and simulation space is preserved in a way that simulation space models are constructed with the information gathered and analyzed in mission space.

CHAPTER 3

SOLUTION APPROACH

This chapter explains our method to transform KAMA mission space models into BOM simulation models. As our solution approach is to find mappings from KAMA to BOM, all the details of those proposed mapping rules are explained in following parts.

We categorize our mapping rules into direct and indirect mappings. As its name implies, direct mapping rules can be used to find direct correspondences from KAMA to BOM model element attributes. However, with contrast to direct mappings, indirect mapping rules can be inferred with associations of KAMA model diagrams and elements to find valid correspondent fields in BOM.

To visualize the rules of direct mappings, we use a tabular notation to show which fields are inputs/outputs of transformation. In this notation;

- Direct mappings are represented by the rows with normal text font,
- Indirect mappings, which indicate associated source attribute is transformed into attribute of another BOM element, are shown by bold and italic text font,
- Finally stroked text font is used to specify that there is no mapping.

Every KAMA and BOM model elements have *Name* attribute in common, so we will not mention this mapping over and over while explaining the element transformations in next parts.

In next subsections, first we will elaborate the mappings from KAMA *Entity* to BOM *Entity Type* and *Event Type,* and then from *KAMAStateMachine* to BOM *State Machine* and finally from KAMA *Mission* to *BOM* and KAMA *Task* to BOM *Pattern Action* elements.

3.1 Mapping Rules from KAMA Entity to BOM Entity Type

In Table 22, mappings between the elements of KAMA *Entity* and BOM *Entity Type* are listed. In this transformation, all *KAMAAttribute* fields of KAMA Entity can be mapped with *Characteristic* attributes of *EntityType*. Additionally, to represent the associations of entities in simulation space, every *Association* of KAMA *Entity* is also mapped with a new *Characteristic* in BOM *Entity Type.*

Table 22: Mapping Table from KAMA Entity to BOM Entity Type

KAMA Entity	BOM Entity Type
Name	Name
KAMAAttributes	Characteristics
Name	Name
Associations	Characteristics
Name	Name
KAMACapabilities	*-Mapped With Another Model-*
Name	

3.2 Mapping Rules from KAMA Entity to BOM Event Type

KAMACapabilities of *Entity*, which have no mappings with BOM *Entity Type* (Table 22)*,* can be used as inputs of the transformation into BOM *Event Type*s*.* All the *Event Types* in simulation space are constructed with corresponding *KAMACapabilities* (Table 23)*.* Hence the capabilities trigger the actions directed from owner entity; name of *KAMA Entity* should be set into *Source Characteristic* attribute of events.

Table 23: Mapping Table from KAMA Entity to BOM Event Type

KAMA Entity	BOM Event Type
Name	Source Characteristic
KAMACapabilities	
Name	Name
-Indirectly Mapped-	*Target Characteristic*
	Content Characteristic
	Trigger Condition
KAMAAttributes	*-Mapped With Another Model-*
Name	
Associations	

Beside these direct mappings between the KAMA capabilities and BOM event types, new mappings can be deduced via Task Flow diagram of KAMA notation. Following the sequence of tasks, *Target Characteristic* attribute can be mapped with name of entity which is responsible for next task in KAMA *Task Flow* diagram and have an *Association* with source entity. Again, list of inputs and outputs of KAMA *Tasks* can indicate the *Event Type's Content Characteristic.* Final missing attribute, namely *Trigger Condition,* is also found with branching guard conditions of KAMA *Task Flow* diagrams and *StateMachines*.

3.3 Mapping Rules from KAMAStateMachine to BOM State Machine

As shown on Table 24, there is almost one to one mapping between the state machines of KAMA and BOM. *KAMAState*s are transformed into BOM *State*s and *KAMAInitialState* and *KAMAFinalState* can represent the first and last *States* in BOM State Machine. Moreover, *Exit Condition* can be created with *KAMATransition* attribute and while creating, *TriggeringEvent, GuardCondition* and *Action are* transformed into *Exit Action* of *Exit Condition.* Finally *KAMATransition's TargetState* is used to specify *Next State* attribute of BOM condition.

Table 24: Mapping Table from KAMAStateMachine to BOM State Machine

KAMAStateMachine	BOM State Machine
Name	Name
KAMAState	State
Name	Name
KAMAInitialState	*- mapped with first state*
KAMAFinalState	*- mapped with last state*
KAMATransition	Exit Condition
TriggeringEvent	Exit Action
GuardCondition	*- part of Exit Action*
Action	*- part of Exit Action*
TargetState	Next State
ContextMission	*-Indirectly Mapped-*

The only field which has no direct mappings to BOM *State Machine* is the *ContextMission* attribute of *KAMATransition.* However, as it is used to refer associated KAMA *Mission,* this field can be mapped with name of target *BOM*.

3.4   Mapping Rules from KAMA Tasks to BOM Pattern Actions

Despite the fact that there are only Name and Sequence direct mappings between KAMA Task and BOM Pattern of Interplay, many indirect mapping inferences can be made.

Mapping rules between KAMA *Tasks* and BOM *Pattern Actions* are listed on Table 25. KAMA *Tasks* can be transformed into *Pattern Actions* of BOM and *Initial Task* and *Final Task* elements in KAMA specify the first and last pattern sequences in *Pattern of Interplay*. *RoleList* of *Tasks* and control flows in *Task Flow Diagram* help finding the responsibilities of entities to be *Sender* and *Receiver* of *Pattern Actions.* This responsibility information is also used to find transformation rules into *Events* which are aligned with *KAMACapabilities*. In short; finding the entities, which are responsible for task execution, assists to find corresponding *Sender*, *Receiver* and

48

Event attributes of BOM *Pattern Action.* In addition to *Event* mapping, *BOM* field can be inferred via *Name* attribute of associated *Task's* owner *Mission.*

Exceptions and *Variations* of BOM *Pattern Actions* can be decided with the help of decision and synchronization points of *Task Flow* diagram. *Exceptions* can be figured out via the decision point branches that end up in *FinalTask*, and *Variations* are mapped with the decision points whose guard condition is evaluated to be "*No*". Finally, *Conditions* in both *Exception* and *Variations* are mapped with decision point's guard conditions.

What's more; *InputList* and *OutputList* which represents whether consumed or produced work products are used to map *Content Characteristic* field of BOM *Event Types.*

Table 25: Mapping Table from KAMA Task to BOM Pattern Action

KAMA Task	BOM Pattern Action
Name	Name
-*Sequence of task execution*	Sequence
	Sender
	Receiver
	Event
	BOM
	Exception
	Name
	Sender
	Receiver
	Event
	BOM
	Condition
-*Indirectly Mapped*-	*Variation*
	Name
	Sender
	Receiver
	Event
	BOM
	Condition

Table 25 (continued)

InputList	
OutputList	*-Indirectly Mapped-*
RoleList	
~~*Preconditions*~~	
~~*Postconditions*~~	
~~*ObjectiveList*~~	*-No Mapping-*
~~*IsExtensionPoint*~~	
~~*ExtensionPointID*~~	
InitialTask	*- mapped with first action*
FinalTask	*- mapped with last action*

Beside these indirect mapping rules, there are also some fields that have no match in simulation space. These fields are the ones with stroked texts. In *Pattern of Interplays*, there is no way to describe the preconditions and postconditions of *Pattern Actions*. Only flow of pattern actions and conditional branches can be modeled. Therefore preconditions and postconditions of KAMA *Tasks* have no match in BOM. Likewise, in BOM there is no metadata attribute in *Pattern Actions* that describes its objectives, so *Task* objectives cannot be transported into BOM models. Finally as there is no mechanism for inclusions of *Pattern Actions* in another one, *IsExtensionPoint* and *ExtensionPointID* attributes cannot be mapped to an attribute of *Pattern Action*.

3.5   Mapping Rules from KAMA Mission to BOM (Model Identification) Mapping

As shown in Table 26, all the *Missions* in KAMA can be represented with a new BOM. Thus, all the missions in *Mission Space* diagram can form a new BOM in BOM assembly of simulation space. Besides, *ObjectiveList* of missions is source of information to fill the *Purpose* field of *BOM Model Identification.*

Table 26: Mapping Table from KAMA Mission to BOM Model Identification

KAMA Mission	BOM Model Identification
Name	Name
ObjectiveList	Purpose
TaskList	*-Indirectly Mapped-*
InitialTask	
FinalTask	
RoleList	
InputList	*-No Mapping-*
OutputList	
Preconditions	
Postconditions	
-No Mapping-	Type
	Version
	Modification Date
	Security Classification
	Release Restriction
	Application Domain
	Description
	Use Limitation
	Use History
	Keyword
	POC
	Reference
	Other
	Glyph

CHAPTER 4

APPLICATION OF TRANSFORMATION

To show applicability of our proposed solution, we applied our transformation method in a real life example. The models we use in this study belong to domain of electronic warfare in which we model RWR (Radar Warning Receiver) and RF guided missiles. The reason of selecting this domain is our personal experience in this field.

4.1. Research Questions

Identifying the research questions is an important activity to successfully define the scope and boundaries of the study.

We specify following research questions for this study:

1) Which simulation space models can be constructed via the execution of transformation?
2) How sufficient is our transformation to form simulation space models?
3) How does our transformation method serve for composability and reusability of simulation models?

For answering the questions above, we will apply our transformation rules which are defined in Chapter 3 – Proposed Solution.

After executing the rules, first question can be answered by listing the end-products of our transformation. For second question, fields of simulation space model will be analyzed to be whether automatically generated or not. After that, the last question will be answered by introducing the two independent BOM models.

4.2. Introduction to Application

In short, we model the RWR behavior on rotary wing airborne platforms when a RF guided missile is fired from an enemy.

In real life, RWR system is used to warn host platform if it detects any RF signals in battle field. While it is operational, it begins to seek for any RF signal in the environment. And, if it finds any, it categorizes the RF signal whether it belongs to a known threat or not. Afterwards, if detected signal belongs to RF guided missile which is approaching to host platform, it sends commands to countermeasure systems of host platform to begin countermeasure and also it warns pilots of host platform through a threat warning display about threat.

As a countermeasure system, we use chaff model which is used to deceive approaching missiles by imitating that it is the real platform.

Another major model we used in this study is RF guided missile. They are fired from host platforms to targets which are detected by tracking radars. After being launched, they seek for threats which are emitting RF signals and they head towards these threats if they are successfully engaged. Finally they detonate when they reached to target.

After introducing the major models of application, we can express its scope as firing RF guided missile to detected target platform on which RWR system is located. This RWR system will detect the missile and alert countermeasure system to dispense chaff for avoiding being hit.

4.3. Implementation

This section describes how we apply our solution method to experiment our transformation rules and the details of conceptual modeling artifacts of both source and target.

From a higher perspective, our application involves mission space modeling with KAMA and applying proposed transformation rules to generate BOM constructs. All details of these transformations and model details are provided in next subsections.

For the sake of readability, we used tabular notation rather than XML to represent BOM models. This notation is also introduced in BOM Template Specification document [13].

4.3.1. Transforming into BOMs from KAMA Missions

It will be convenient to start with transformation of KAMA Missions into separate BOMs hence other transformations will be applied in the context of these detached BOMs.

Our experimental simulation will be modeled for achieving two missions of firing RF guided missiles to detected targets and applying countermeasure with the help of RWR system. These missions are modeled with KAMA framework as given on Figure 18 and Figure 19.

After we applied the rules defined in section 3.5, missions of simulation formed two separate BOMs whose model identification components are constructed as Table 27 and Table 28. *Purpose* field of *Model Identifications* are directly mapped with *Objectives* of KAMA *Missions*.
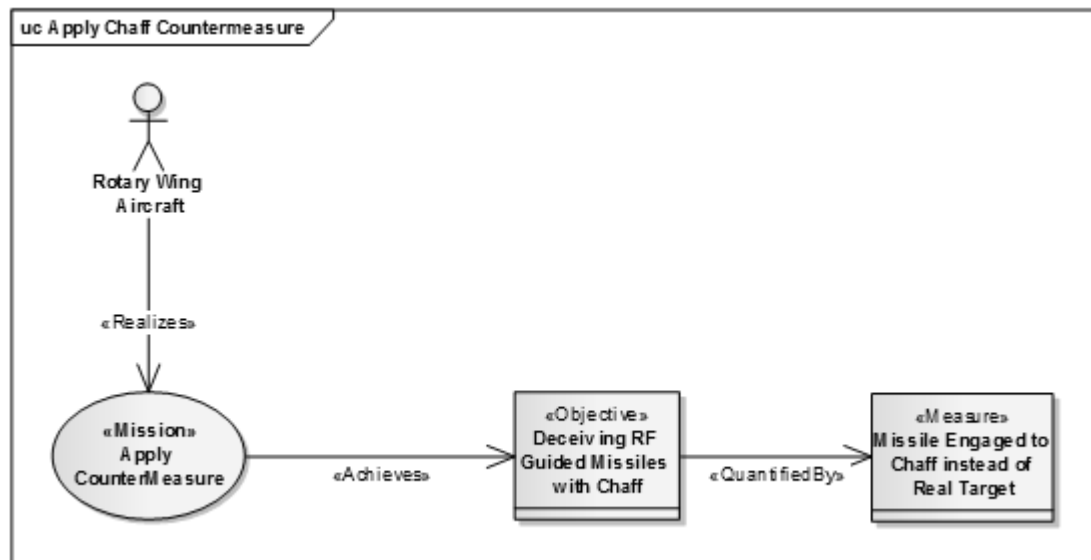
Figure 18: KAMA Mission Space Diagram for Applying Countermeasure

Table 27: Apply Countermeasure BOM Model Identification

Model Identification	
Name	ApplyCountermeasureBOM
Purpose	Deceiving RF Guided Missiles with Chaff
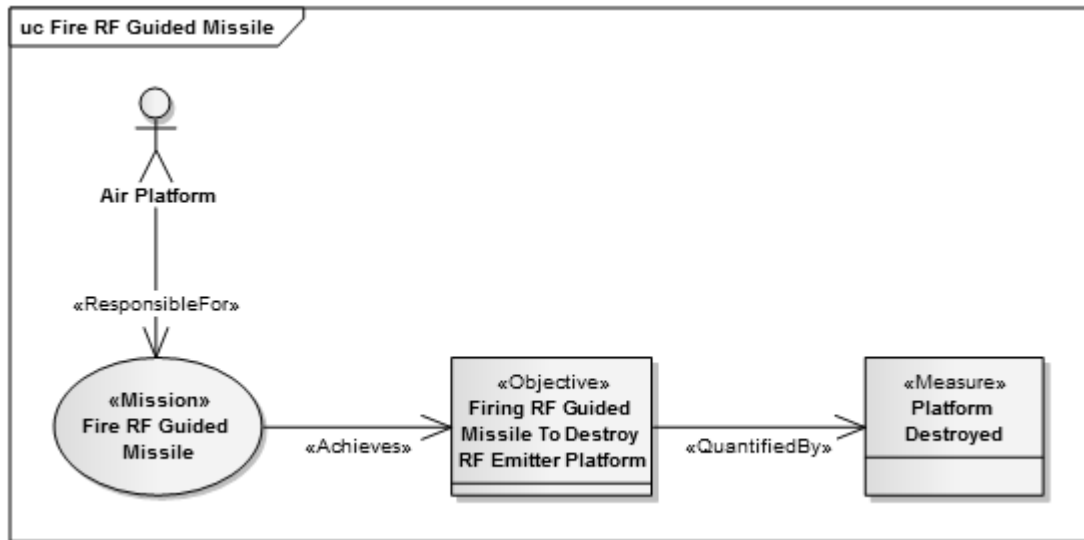
Figure 19: KAMA Mission Space Diagram for Firing RF Guided Missile

Table 28: Fire RF Guided Missile BOM Model Identification

Model Identification	
Name	FireRFGuidedMissileBOM
Purpose	Firing RF Guided Missile To Destroy RF Emitter Platform

Having divided conceptual models of mission space into two BOMs, we have constructed composable simulation space piece parts which can be used in different simulations independently. Thus in next sections, we grouped other transformation artifacts into two BOMs of *Applying Countermeasure* and *Firing RF Guided Missile*.

4.3.2. Transforming into BOM Entity Types from KAMA Entities

For building up static models of mission space, we used KAMA's Entity Ontology diagrams to show which entities exist in the context of missions and what are the relationships between these entities.

First of all, we have identified only the entities, their attributes and the relationships and then; while developing the state machines and task flows, we found their

operations. The resulting mission space models are depicted in Figure 20 and Figure 21.

Mapping rules which are defined in section 3.1 are applied to form the basis of BOM *Entity Types*. In this transformation, not only the *KAMAAttribute* fields but also *Associations* are directly converted to BOM *Characteristics*.

At the end of executing the rules of transformation we have constructed BOM models which are listed on Table 29 and Table 30.
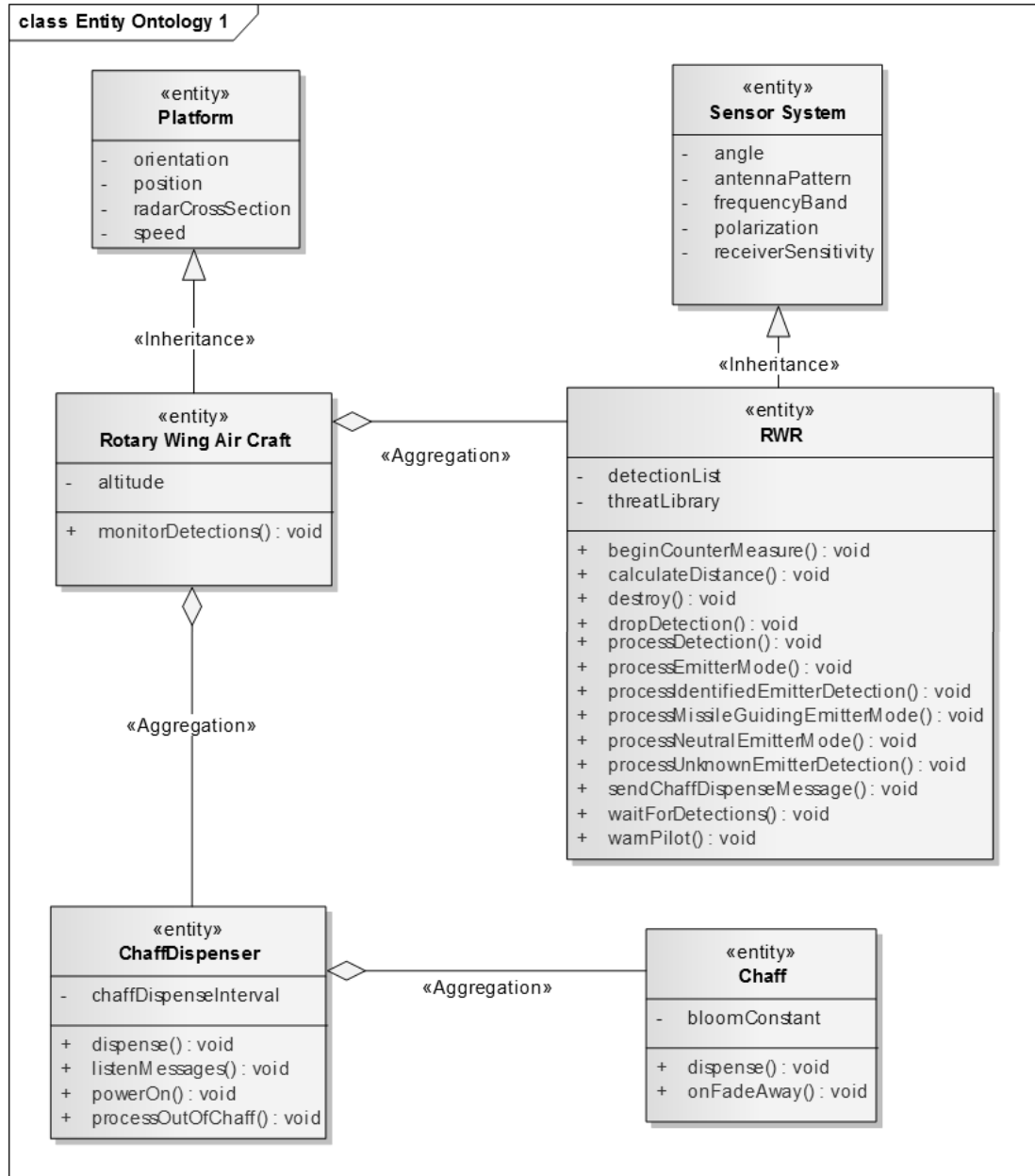
Figure 20: KAMA Entity Ontology Diagram for Mission of Applying Countermeasure

Table 29: Apply Countermeasure BOM Entity Types

Type	Characteristic
RWR	Angle
	AntennaPattern
	FrequencyBand
	Polarization
	ReceiverSensistivity
	DetectionList
	ThreatLibrary
	AssociatedRotaryWingAircraft
Rotary Wing Aircraft	Orientation
	Position
	RadarCrossSection
	Speed
	Altitude
	AssociatedChaffDispenser
	AssociatedRWR
Chaff Dispenser	ChaffDispenseInterval
	AssociatedChaffs
	AssociatedRotaryWingAircraft
Chaff	BloomConstant
	AssociatedChaffDispenser
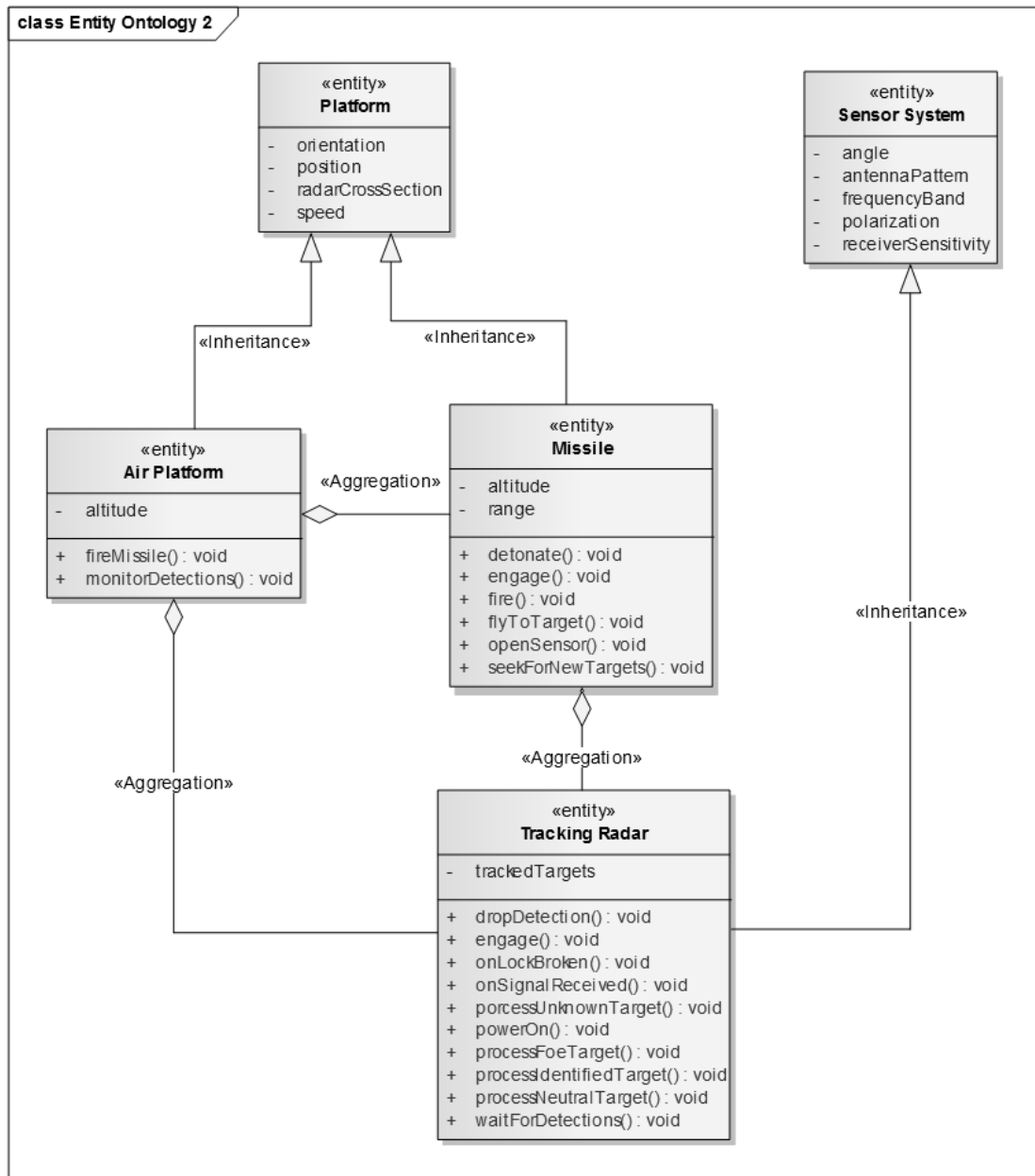
Figure 21: KAMA Entity Ontology Diagram for Mission of Firing RF Guided Missile

Table 30: Fire RF Guided Missile BOM Entity Types

Type	Characteristic
Missile	Orientation
	Position
	RadarCrossSection
	Speed
	Altitude
	Range
	AssociatedAirPlatform
	AssociatedTrackingRadar
Air Platform	Orientation
	Position
	RadarCrossSection
	Speed
	Altitude
	AssociatedMissiles
	AssociatedTrackingRadar
Tracking Radar	Angle
	AntennaPattern
	FrequencyBand
	Polarization
	ReceiverSensistivity
	TrackedTargets
	AssociatedAirPlatform
	AssociatedMissile

4.3.3. Transforming into BOM Event Types from KAMA Entities

In section 3.2, we indicated that; besides forming simulation space *Entity Types,* we use KAMA *Entities* to form *Event Types* of BOM.

To transform into *Event Types*, first we list all the *KAMACapabilities* as BOM events and assign names of corresponding entities to *Source Characteristic* fields. In second step, we use task flow diagrams (Figure 27 and Figure 28) to capture *Target Characteristic*, *Content Characteristic* and *Trigger Condition* fields via task

ownerships and inputs/outputs of tasks. Details of those direct and indirect mapping rules are given in section 3.2.

At the end, we have constructed the *Event Types* of Apply Countermeasure BOM as Table 31 and Fire RF Guided Missile BOM as Table 32.

Table 31: Apply Countermeasure BOM Event Types

Name	Source Characteristic	Target Characteristic	Content Characteristic	Trigger Condition
BeginCountermeasure	RWR	RWR		
CalculateDistance	RWR	RWR	DetectedSignal	EmitterIsGuidingMissile = TRUE
Destroy	RWR	RWR		
OnSignalReceived	RWR	RWR	DetectedSignal	AnyDetection = TRUE
PowerOn	RWR	RWR		
ProcessDetection	RWR	RWR	DetectedSignal	
ProcessIdentifiedEmitterDetection	RWR	RWR	DetectedSignal	UnknownDetection = FALSE
ProcessUnknownEmitterDetection	RWR	RWR	DetectedSignal	UnknownDetection = TRUE
ProcessEmitterMode	RWR	RWR	ThreatLibrary	
ProcessMissileGuidingEmitterMode	RWR	RWR	ThreatLibrary	EmitterGuidingMissile = TRUE
ProcessNeutralEmitterMode	RWR	RWR	ThreatLibrary	EmitterGuidingMissile = FALSE
SendChaffDispenseMessage	RWR	ChaffDispenser	DispenseChaffMessage	
WaitForDetections	RWR	RWR		
WarnPilot	RWR	RWR		IsThreatIdentified = FALSE
MonitorDetections	RotaryWingAirCraft	RWR		
Dispense	ChaffDispenser	Chaff		

Table 31 (continued)

ListenMessages	ChaffDispenser	RWR	DispenseChaffMe ssage	
PowerOn	ChaffDispenser	ChaffDis penser		
ProcessOutOfChaff	ChaffDispenser	ChaffDis penser		
Dispense	Chaff	Chaff		
OnFadeAway	Chaff	Chaff		

Table 32: Fire RF Guided Missile BOM Event Types

Name	Source Characteristic	Target Characteristic	Content Characteristic	Trigger Condition
FireMissile	Airplatform	Missile		
MonitorDetections	Airplatform	TrackingRadar		
DropDetection	TrackingRadar	TrackingRadar		
Engage	TrackingRadar	TrackingRadar	DetectedSignal	IsFoe = TRUE
OnLockBroken	TrackingRadar	TrackingRadar		
OnSignalReceived	TrackingRadar	TrackingRadar		
ProcessUnknownTarget	TrackingRadar	TrackingRadar		UnknownDetection = TRUE
PowerOn	TrackingRadar	TrackingRadar		
ProcessFoeTarget	TrackingRadar	TrackingRadar		TargetIsFoe = TRUE
ProcessIdentifiedTarget	TrackingRadar	TrackingRadar		DetectionIdentified = TRUE
ProcessNeutralTarget	TrackingRadar	TrackingRadar		TargetIsNeutral = TRUE
WaitForDetections	TrackingRadar	TrackingRadar		
Detonate	Missile	Missile		ReachedToTarget = TRUE
Engage	Missile	Missile		AnyDetection = TRUE
Fire	Missile	Missile		
FlyToTarget	Missile	Missile		
OpenSensor	Missile	Missile		ThreeSecondsPassed = TRUE
SeekForNewTargets	Missile	Missile		TargetLost = TRUE

4.3.4. Transforming into BOM State Machine from KAMA StateMachine

With the aim of connecting the state information of mission space with simulation space, we use the rules of transformation defined in section 3.3.

We use KAMA modeling notation again to develop the state machine of real world entities. These models are developed for associated mission space entities and they are shown on Figure 22, Figure 23, Figure 24, Figure 25 and Figure 26.

Rules of mappings (section 3.3) are executed to build simulation space state machines. Transformation starts with mappings of KAMA *Initial* and *Final States* which are represented by the first and last states in BOM *State Machines*. Then, consecutive BOM states, between initial and final states, are directly mapped with all other states of KAMA *StateMachine*. Furthermore, the transitions between source states of KAMA along with *GuardCondition* and *Actions* are transformed into *Exit Conditions* of BOM. Finally *TargetState* of mission space are used for *Next State* of target state machine.

When we applied the rules above, we have developed the state machines for both Apply Countermeasure and Fire RF Guided Missile BOMs as in Table 33 and Table 34 respectively.
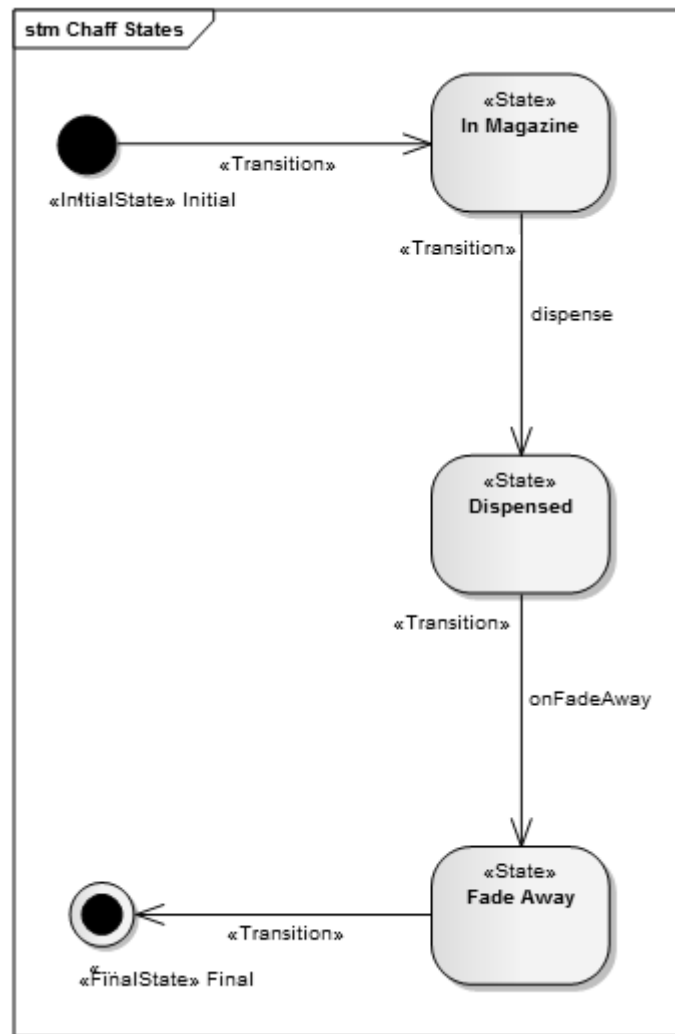
Figure 22: KAMA Chaff States Diagram
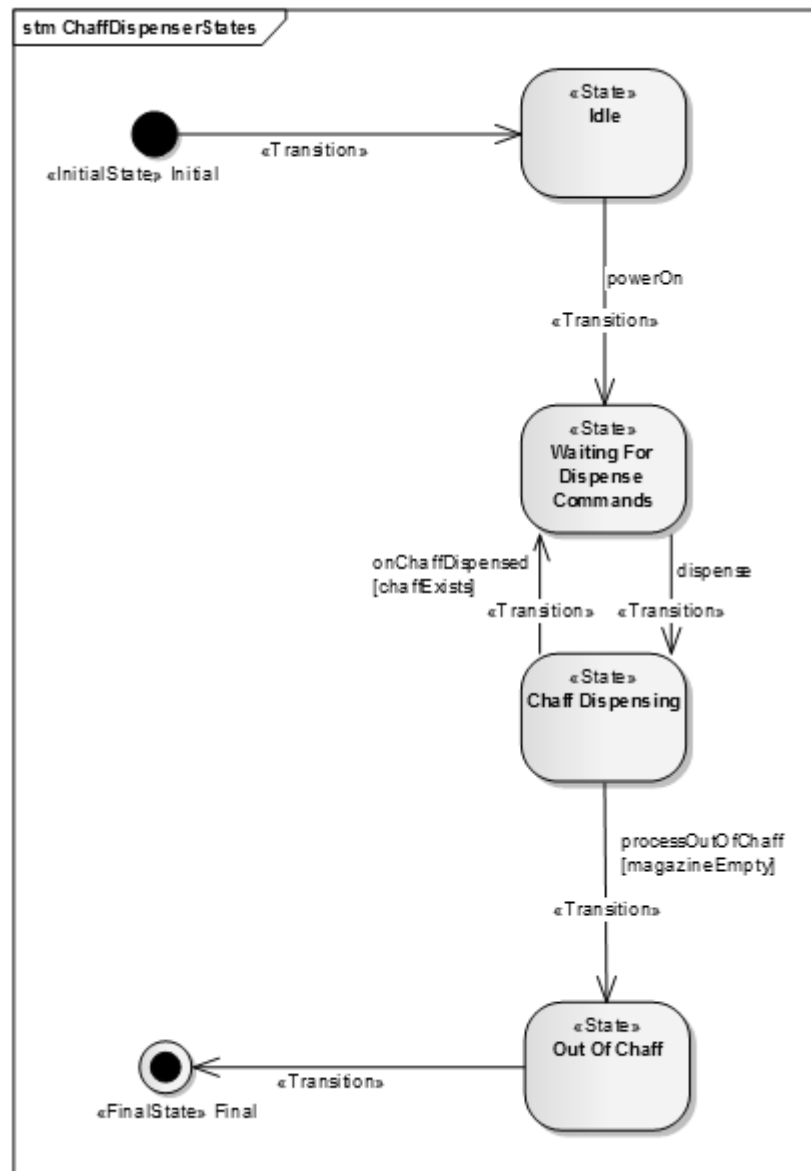
Figure 23: KAMA Chaff Dispenser States Diagram
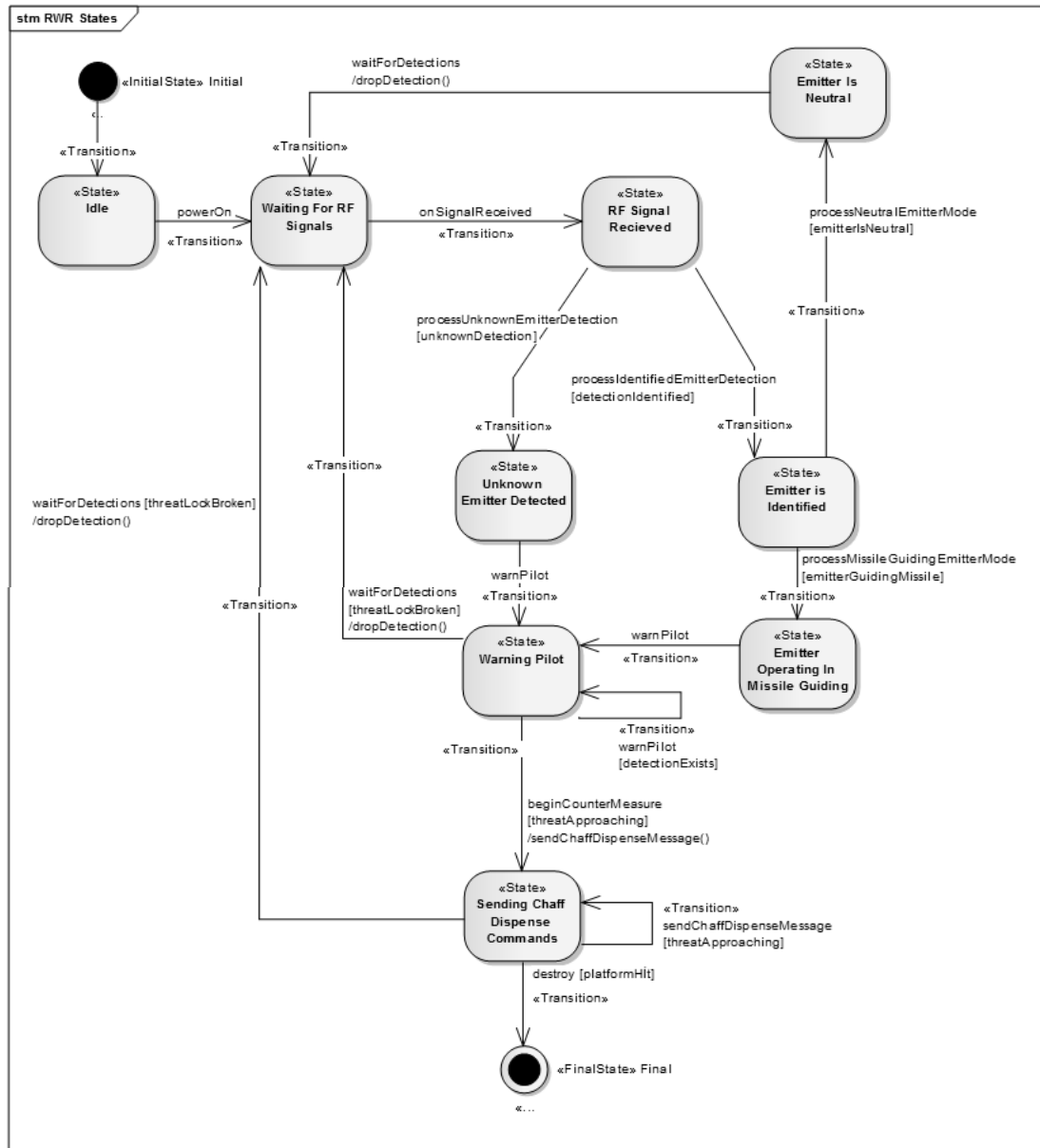
Figure 24: KAMA RWR States Diagram

Table 33: Apply Countermeasure BOM State Machine

Name	Entity	State		
		State Name	ExitCondition	
			Exit Action	Next State
Chaff States	Chaff	Initial	N/A	In Magazine
		In Magazine	Dispense	Dispensed
		Dispensed	OnFadeAway	Fade Away
		Fade Away	N/A	Final
		Final	N/A	N/A
Chaff Dispenser States	Chaff Dispenser	Initial	N/A	Idle
		Idle	PowerOn	Waiting for Dispense Commands
		Waiting for Dispense Commands	Dispense	Chaff Dispensing
		Chaff Dispensing	OnChaffDispensed	Chaff Exists
			ProcessOutOfChaff	OutOfChaff
		Out Of Chaff	N/A	Final
		Final	N/A	N/A

Table 33 (continued)

|		Initial	N/A	Idle
RWR States	RWR	Idle	PowerOn	Waiting For RF Signals
		Waiting For RF Signals	OnSignalReceived	RF Signal Received
		RF Signal Received	ProcessUnknownEmitterDetection	Unknown Emitter Detected
			ProcessIdentifiedEmitterDetection	Emitter Is Identified
		Warning Pilot	WarnPilot	Warning Pilot
			BeginCounterMeasure	Sending Chaff Dispense Commands
			WaitForDetections	Waiting For RF Signals
		Emitter is Identified	ProcessMissileGuidingEmitterMode	Emitter Operating In Missile Guiding
			ProcessNeutralEmitterMode	Emitter Is Neutral
		Emitter Is Neutral	WaitForDetections	Waiting For RF Signals
		Emitter Operating In Missile Guiding	WarnPilot	Sending Chaff Dispense Commands
		Sending Chaff Dispense Commands	SendChaffDispenseMessage	Sending Chaff Dispense Commands
			WaitForDetections	Waiting For RF Signals
			Destroy	Final
		Final	N/A	N/A
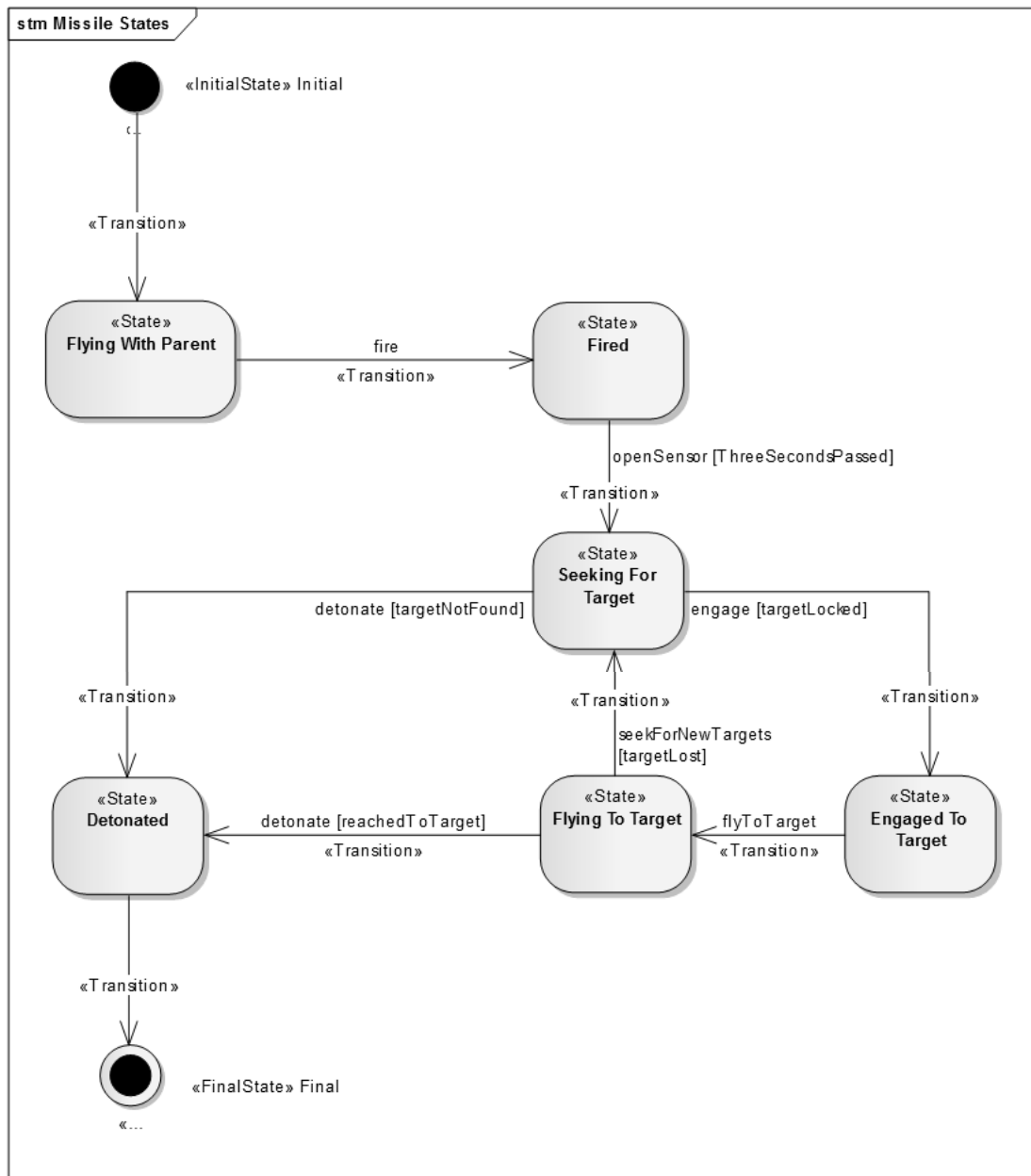
Figure 25: KAMA Missile States Diagram
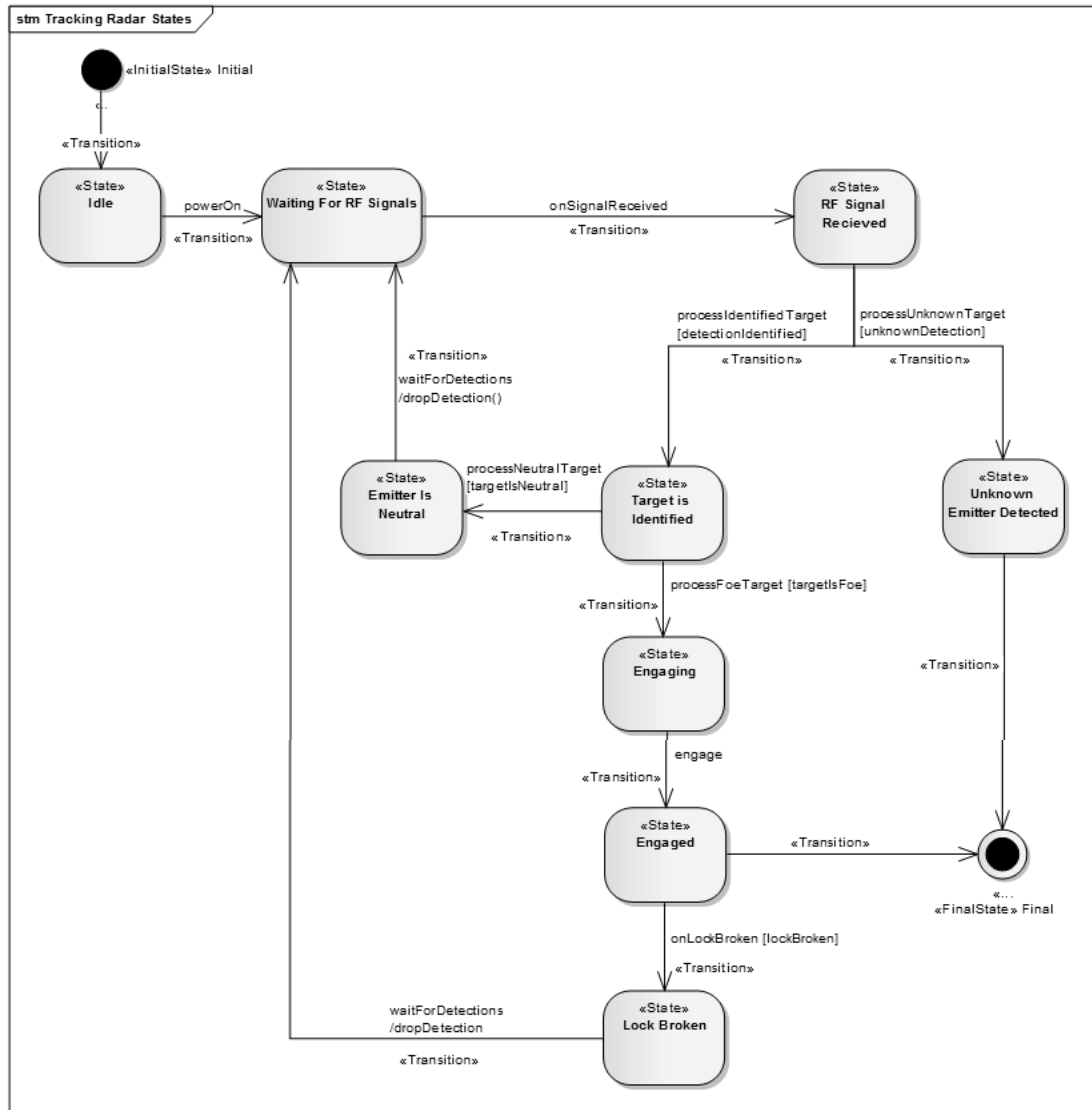
Figure 26: KAMA Tracking Radar States Diagram

Table 34: Fire RF Guided Missile BOM State Machine

Name	Entity	State		
		State Name	**ExitCondition**	
			Exit Action	**Next State**
Missile States	Missile	Initial	N/A	Flying with Parent
		Flying with Parent	Fire	Fired
		Fired	OpenSensor	Seeking for Target
		Seeking for Target	Engage	Engaged to Target
			Detonate	Detonated
		Engaged to Target	FlyToTarget	Flying to Target
		Flying to Target	SeekForNewTargets	Seeking for Target
			Detonate	Detonated
		Detonated	N/A	Final
		Final	N/A	N/A
Tracking Radar States	Tracking Radar	Initial	N/A	Idle
		Idle	Power On	Waiting for RF Signals
		Waiting for RF Signals	OnSignalReceived	RF Signal Received
		RF Signal Received	ProcessUnknownTarget	Unknown Emitter Detected
			ProcessIdentifiedTarget	Target Is Identified
		Target Is Identified	ProcessFoeTarget	Engaging
			ProcessNeutralTarget	Emitter Is Neutral
		Emitter Is Neutral	Wait for Detections	Waiting for RF Signals
		Engaging	Engage	Engaged
		Engaged	OnLockBroken	Lock Broken
			N/A	Final
		Lock Broken	Wait for Detections	Waiting for RF Signals
		Unknown Emitter Detected	N/A	Final
		Final	N/A	N/A

4.3.5. Transforming into BOM Pattern of Interplay from KAMA Task Flows

In this final transformation subsection, to transport dynamic behaviors we use the KAMA Task Flow diagrams (Figure 27 and Figure 28) for finding out BOM pattern of interplays as mentioned in section 3.4.

Like state machine mapping, we began with the *Initial* and *Final Tasks* which are used to indicate first and last *Pattern Actions* of BOMs and then we map all other remaining KAMA *Tasks* with BOM *Pattern Actions*. Next, to decide which entities are the *Receiver* and *Sender*, we use the control flows with associated *Role* owners. Moreover, we used this responsibility information also for finding the related BOM *Events*.

After we developed the primary path of Pattern of Interplay, we used decision points and alternative flows (whose guard conditions are evaluated to be '*No*') of KAMA to find *Variations* in BOM *Pattern of Interplays*.

Finally, applying the transformation which is detailed above, we successfully created *Patterns of Interplay* of Apply Countermeasure BOM as Table 35 and Fire RF Guided Missile BOM as Table 36.
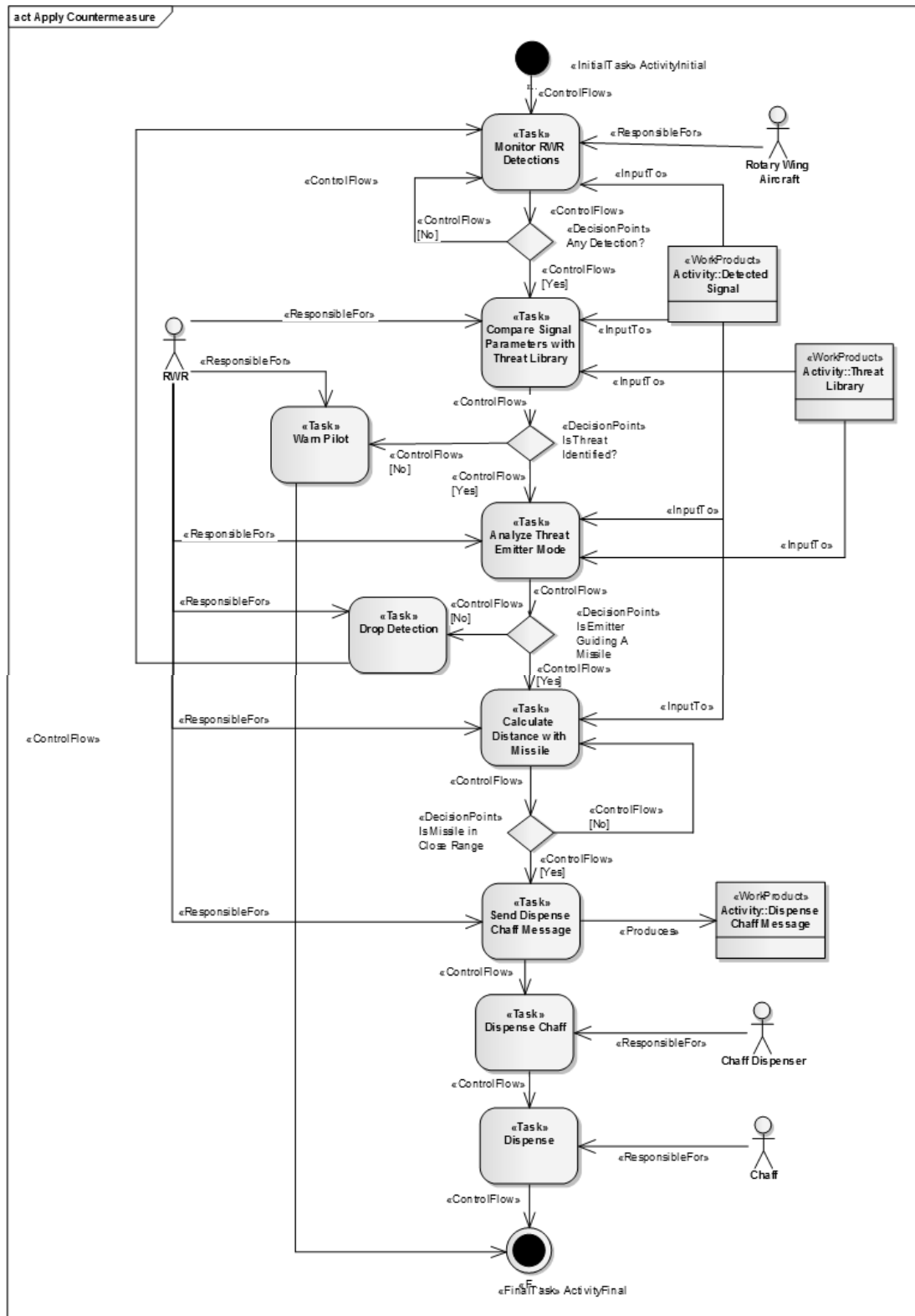
Figure 27: KAMA Task Flow Diagram for Applying Countermeasure

Table 35: Apply Countermeasure Pattern of Interplay

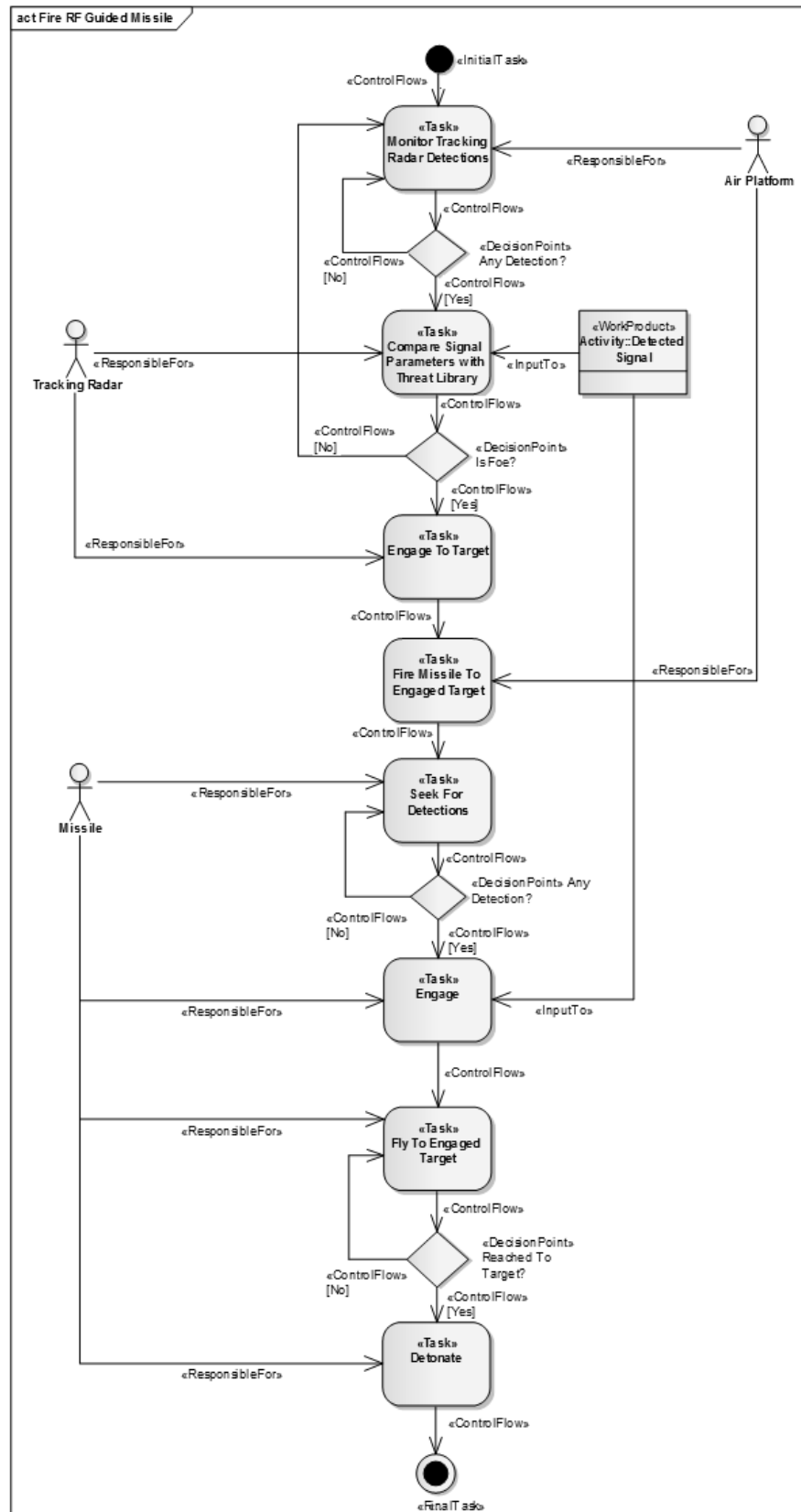| Pattern of Interplay Name		Seq	Name	Sender	Receiver	Event	BOM	Condition
Apply Countermeasure	**Pattern Action**	1	Compare Signal Parameters with Threat Library	RWR	RWR		N/A	
	Variation	N/A	Warn Pilots			WarnPilot	N/A	Is Threat Identified = FALSE
	Pattern Action	2	Analyze Threat Emitter Mode	RWR	RWR	ProcessEmitterMode	N/A	
	Variation	N/A	Drop Detection	RWR	RWR	DropDetection	N/A	Is Emitter Guiding A Missile = FALSE
	Pattern Action	3	Calculate Distance with Missile	RWR	RWR	CalculateDistance	N/A	
	Variation	N/A	Calculate Distance with Missile	RWR	RWR	CalculateDistance	N/A	Is Missile in Close Range = FALSE
	Pattern Action	4	Send Dispense Chaff Message	RWR	ChaffDispense	DispenseChaff	N/A	

Figure 28: KAMA Task Flow Diagram for Firing RF Guided Missile

Table 36: Fire RF Guided Missile Pattern of Interplay

Pattern of Interplay Name		Seq	Name	Sender	Receiver	Event	BOM	Condition
Fire RF Guided Missile	**Pattern Action**	1	Compare Signal Parameters with Threat Library	Tracking Radar	Tracking Radar		N/A	
	Variation	N/A	Monitor Tracking Radar Detections	Tracking Radar	Air Platform	MonitorTrackingRadarDetections	N/A	Is Foe = FALSE
	Pattern Action	2	Engage to Target	Tracking Radar	Tracking Radar	Engage	N/A	
	Pattern Action	3	Fire Missile to Engaged Target	Air Platform	Missile	FireMissile	N/A	
	Pattern Action	4	Seek for Detections	Missile	Missile	SeekForNewTargets	N/A	
	Variation	N/A	Seek for Detections	Missile	Missile	SeekForNewTargets	N/A	Any Detection = FALSE
	Pattern Action	5	Engage	Missile	Missile	Engage	N/A	
	Pattern Action	6	Fly to Engaged Target	Missile	Missile	FlyToTarget	N/A	
	Variation	N/A	Fly to Engaged Target	Missile	Missile	FlyToTarget	N/A	Reached to Target = FALSE
	Pattern Action	7	Detonate	Missile	Missile	Detonate	N/A	

4.4. Summary and Discussions on the Findings

We applied our transformation method in a real life example to explore applicability of our proposed solution. For this study, we selected the RF interactions of missiles and RWR systems in electronic warfare and model their static structures and dynamic behaviors. We model how RF guided missiles can be fired to detected target platforms and how target platforms' countermeasure systems react to this threat.

We start with modeling the real world missions with KAMA framework. We define two missions of *Firing RF Guided Missile* and *Applying Countermeasure* in mission space diagram and then continue with identifying the entities and the relationships in

entity ontology diagrams. Having defined the entities, we specify their state machines. Afterwards, as a final mission space model, we show the flow of actions in task flow diagrams.

At the end of developing mission space models, we execute our transformation rules step by step to construct BOMs. As a result, two BOMs were constructed with model identification, entity types, event types, state machine and pattern of interplay.

With the help of this application, we also gathered some rough numbers for estimating the effectiveness of our proposition.

In such a case that our method is not used, a simulation matter expert begins with developing the mission space conceptual models for both applying the countermeasure and firing RF guided missile missions in KAMA and he spends approximately 12 hours for developing these mission space models. After that, a simulation designer may develop BOM simulation space models for the same requirements in approximately 20 hours. In total, roughly 32 hours are spent for developing the conceptual models.

If our transformation method is used, again simulation matter expert spends 12 hours for mission space models. But this time, instead of developing BOM models from scratch, our transformation rules can be applied within approximately 4 hours for creating BOM models. Simulation designers also spend approximately 3 hours to make the models mature. In total, 19 hours will be spent for developing both mission space and simulation space conceptual models.

To sum up, it can be claimed that our transformation method is approximately %40 more effective than developing mission and simulation space models separately. What's more, if transformation is not applied, simulation designers will spend further efforts to synchronize simulation space models in case of any mission space model updates.

4.4.1. Research Questions

Research questions, defined in section 4.1, can be answered with applied transformation method.

When we execute our rules of transformation, first research question, "*Which simulation space models can be constructed via the execution of transformation?*" was automatically answered by listing the outputs of transformation. At the end of our execution of rules, we have observed that we constructed the following BOM elements:

- **BOM Model Identification:** When we transformed our KAMA Missions, two fields of Model Identification component are filled by mission name and objective. Filling only two fields of simulation space model is not a significant contribution, but the importance of applying this transformation is identifying independent BOMs with respect to KAMA missions. In this experiment, we identified two BOMs of "*Firing RF Guided Missile*" and "*Applying Countermeasure*" with corresponding *Model Identifications*. (See Table 27 and Table 28)

- **BOM Entity Type:** For transforming the static models of mission space into simulation space, we produced BOM Entity Types with the inputs of KAMA Entities. (See Table 29 and Table 30)

- **BOM Event Type:** We have not only used KAMA Entities for BOM Entity Types, but also we have formed BOM Event Types by using KAMA Entities' *Capabilities*. (See Table 31 and Table 32)

- **BOM State Machine:** At the end of state transformations, we produced BOM State Machines according to KAMA StateMachines. (See

- Table 33 and Table 34)

- **BOM Pattern of Interplay:** Dynamic behaviors of real world entities are transformed into the simulation space model of BOM Pattern of Interplays. (See Table 35 and Table 36)

Question of "*How sufficient is our transformation to form simulation space models?*" is answered by analyzing which target fields can be mapped and which fields cannot. Although our mapping tables (Table 22, Table 23, Table 24, Table 25 and Table 26) show which fields are covered by our proposed transformation, end products of experiment proves the applicability of solution. Field based analysis is performed as inspecting the produced models below:

- **BOM Model Identification:** Since there is no mission space conceptual model metadata representation in KAMA and Model Identification component represents metadata of BOM, only two fields (Name, Purpose) of BOM Model Identification can be produced (Table 28 and Table 29).
- **BOM Entity Type:** Entity Type fields of BOM are fully covered by our transformation (See Table 29 and Table 30). Name fields are mapped with the KAMA Entity's name and Characteristics are mapped with Attributes and Associations. There is no indirect mapping between KAMA and BOM entities, all mappings are carried out directly.
- **BOM Event Type:** Although not all of the mappings are direct for filling up target fields of Event Types, all fields can be produced whether directly or indirectly. In our experiment, as listed on Table 31 and Table 32, all applicable fields of BOM Event Types are successfully found.
- **BOM State Machine:** Like Entity Type, all fields of BOM State Machines can be produced with direct mappings. It can be observed from
- Table 33 and Table 34 that all fields of BOM State Machines are constructed by applying our state machine transformation.
- **BOM Pattern of Interplay:** When compared to other transformations, transforming into BOM Patterns of Interplay is the activity which requires indirect inferences most. However, there is always a mapping to construct every single field of BOM Patterns of Interplay as our experiment results indicate on Table 35 and Table 36.

While designing the simulation space models, designers first find out which real world entities they will deal with. By performing our transformation method,

simulation designers will have a solid basement for the simulation entities. They will get great portion of the entities in simulation automatically along with entity relationships, attributes and capabilities. What's more; they will have the state machine information of these entities and flow of tasks to accomplish simulation objectives to begin designing dynamic simulation models.

Our experiment shows that BOM components of Entity Type (Table 29 and Table 30), Event Type (Table 31 and Table 32), State Machine (
Table 33 and Table 34) and Pattern of Interplay (Table 35 and Table 36) will be presented to simulation designers which are fully constructed with real world analysis results (Figure 20- Figure 28).

Shortly, best answer to our third research question of "*How does our transformation method serve for composability and reusability of simulation models?*" is the produced BOMs from KAMA Missions. We list corresponding Model Identifications on Table 28 and Table 29 for the BOMs of Firing RF Guided Missile and Applying Countermeasure.

The idea behind BOM is developing composable and interoperable simulation models [21]. In this study, to serve for that composability idea we defined our transformation as it produces disassociated BOMs with respect to KAMA Missions. So, despite the fact that domain experts model KAMA missions as a whole, produced BOMs can be independently used in different simulations.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1. Conclusion

In this study, we tried to solve the problem of having no linkage between mission and simulation space conceptual modeling by proposing a method to transform mission space models into simulation space.

It is experienced that domain experts develops the conceptual models in mission space to identify and validate conceptual constructs. Then, for enhancing simulation design and implementation phases, simulation designers develop conceptual models in simulation space. The problem arises here as there is no defined way of utilizing the studies of domain experts for simulation designers.

To solve the problem, we have proposed a method to transform the mission space models into simulation space ones. Our method of transformation is based on finding valid mappings from KAMA to BOM models.

For specifying the rules of transformation we first analyzed all models of both KAMA and BOM. Owing to their having similar conceptual models, we have successfully identified the model wise transformations. Next for defining the field based transformation rules, we have analyzed both KAMA and BOM's fields to be matched. Then we observed that there are some direct and indirect field mappings

between KAMA and BOM models. Finally, according to our findings, we constructed our transformation rules of mappings.

Having proposed our solution approach, we applied our transformation method in a real life example to confirm the validity of our methodology. By this application, we have verified that great portion of BOM simulation space models can be directly or indirectly constructed with the mission space model elements and diagrams of KAMA framework.

However, we saw that there is no one to one mapping between KAMA and BOM models. There are some fields in KAMA that have no matching in BOM and there are some fields in BOM which cannot be mapped with KAMA models. It is an anticipated result hence they have different concerns. KAMA models are closed to problem domain and BOM models are ready to be used in simulation design and implementation phases. These unmatched fields are used for modeling the real world mission descriptions and analyzing the requirements. Although there will be some loss of information while transforming into simulation space, these unmatched fields are not crucial fields for the further steps of designing simulation software components. Maybe in future, BOM can be extended or another simulation space conceptual modeling notation can be used to represent all these fields.

Despite having some limitations, we believe that this study offers a number of benefits for simulation studies:

- Real world analysis results will be transformed into simulation design and implementation artifacts.
- Duplicate efforts to model similar, or sometimes the same information will be minimized.
- Usage of conceptual modeling in simulation projects will increase as it will be easier to maintain links between mission and simulation space models.
- A communication link will be established between simulation domain experts and simulation designers.

- Every mission in mission space will correspond to a single manageable BOM. Therefore, transformed conceptual models will serve for the interoperability and composability of simulations.

5.2. Future Work

Although this application covers all our transformation rules, new applications can be developed to improve this study further. In future, transformation rules can be applied in different real life examples by mission space modelers who do not know about BOM. So applicability of our solution will be proved by modelers of having knowledge about neither the transformation nor the BOM.

In this study, to support moving from requirement analysis to simulation design and implementation phases, we propose one way transformation method. For bidirectional synchronization, new rules for the transformation of BOM into KAMA models can be also defined in future. So, the changes made in simulation space can be also reflected into mission space.

Another possible future work can be replacing the KAMA or BOM with new mission or simulation space conceptual modeling frameworks. When we start this research KAMA and BOM was the best choices, however in future better frameworks and concepts can be developed to support conceptual modeling phase.

We have just proposed a method for a sub-step of moving from requirement analysis to simulation implementation. Our study ends in simulation design phase. However as BOM models seem very promising to move into further steps of simulation study, new researches may lead to transform BOM simulation space models into simulation software components.

REFERENCES

[1]   Rothenberg J, Rand Corporation. Knowledge-Based Simulation: An Interim Report. Santa Monica, CA: Rand Corporation, 1989.

[2]   Robinson S. Simulation: The Practice of Model Development and Use. Chichester, West Sussex, England ; Hoboken, NJ: Wiley, 2004.

[3]   Zeigler BP, Kim TG, Praehofer H. Theory of Modeling and Simulation : Integrating Discrete Event and Continuous Complex Dynamic Systems. San Diego, CA: Academic, 2000.

[4]   Borah J. Conceptual Modeling - The Missing Link of Simulation Development. Proceedings of the Spring Simulation Interoperability Workshop, 2002.

[5]   Nance RE. The Conical Methodology and the Evolution of Simulation Model. 1986.

[6]   IEEE Computer Society. Recommended Practice for High Level Architecture (HLA) Federation Development and Execution Process (FEDEP). 2003.

[7]   Salt JD. Simulation Should Be Easy and Fun. Winter Simulation Conference, Proceedings of the 25th conference on Winter simulation. Los Angeles, California, United States: ACM, 1993.

[8]   Chwif L, Barretto MRP, Paul RJ. On Simulation Model Complexity. Winter Simulation Conference, Proceedings of the 32nd conference on Winter simulation. Orlando, Florida, 2000.

[9]   Robinson S, Brooks R, Kotiadis K, Van der Zee D-J. Conceptual Modeling for Discrete-Event Simulation, 2010.

[10]   Pace DK. Conceptual Model Descriptions. 1998.

[11] Aysolmaz B. A Study on Conceptual Modeling in Simulation Systems: An Extended Methodology for KAMA. Department of Information Systems, vol. MSc. Ankara: Middle East Technical University, 2007.

[12] Karagöz NA. A Framework for Developing Conceptual Models of the Mission Space for Simulation Systems. Department of Information Systems, vol. PhD. Ankara: Middle East Technical University, 2008.

[13] SISO (Simulation Interoperability Standards Organization). Base Object Model (BOM) Template Specification. 2006.

[14] SISO (Simulation Interoperability Standards Organization). Guide for Base Object Model (BOM) Use and Implementation. 2006.

[15] Mojtahed V, Andersson B, Kabilan V, Zdravkovic J. BOM++, a Semantically Enriched BOM. 2008.

[16] (DMSO) DMaSO. Conceptual Models of the Mission Space (CMMS) Technical Framework. USD/A&T-DMSO-CMMS-0002 Revision 0.2.1, 1997.

[17] Mojtahed V, Andersson B, Kabilan V. DCMF-Defence Conceptual Modeling Framework. FOI-R--1754--SE, 2005.

[18] Mojtahed V, Andersson B, Kabilan V. BOM and DCMF. Stockholm: FOI, 2008.

[19] W3C. OWL Web Ontology Language Overview. 2004.

[20] Submission WCM. OWL-S Semantic Markup for Web Services. 2004.

[21] Gustavson P, Chase T. Building Composable Bridges between the Conceptual Space and the Implementation Space. In: S. G. Henderson BB, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds, editor. Proceedings of the 2007 Winter Simulation Conference, 2007.