University of Central Florida

STARS

Electronic Theses and Dissertations, 2004-2019

2018

Development of an MRM Federation System Using COTS Simulations

Jaeho Kim
University of Central Florida

Part of the Industrial Engineering Commons
Find similar works at: https://stars.library.ucf.edu/etd
University of Central Florida Libraries http://library.ucf.edu

This Masters Thesis (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2004-2019 by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation
Kim, Jaeho, "Development of an MRM Federation System Using COTS Simulations" (2018). *Electronic Theses and Dissertations, 2004-2019*. 6421.
https://stars.library.ucf.edu/etd/6421

Showcase of Text, Archives, Research & Scholarship

DEVELOPMENT OF AN MRM FEDERATION SYSTEM USING COTS SIMULATIONS

by

JAEHO KIM
B.E. Korea Naval Academy, 2007

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science
in the Department of Industrial Engineering and Management Systems
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2018

Major Professors: Gene Lee

© 2018 Jaeho Kim

ABSTRACT

The goal of this research is to build an experimental environment for the Simulation Interoperability Laboratory (SIL) of the University of Central Florida (UCF). The Simulation Interoperability Laboratory (SIL) is researching about multi-resolution modeling(MRM), with a focus on military field uses. This thesis proposes steps to develop an MRM federation system and build two different MRM systems using COTS simulations (SIMBox, VR-Forces, and MASA Sword). This report is written to provide the basis for a time-based MRM federation study in the Simulation Interoperability Laboratory.

The report describes many definitions and notions related to Multi-Resolution Modeling(MRM) and discusses examples to make better understanding for further research. MRM is relatively new research, and there are high demands for integrating simulators running in military field purposes. Most military related research is based on simulators currently being used in the military; this poses a problem for research because the data is classified, resulting in many limitations for outside researchers to see the military's process for building an MRM system or the results of the research. Therefore, development of the MRM federation using COTS simulations can provide many examples of MRM issues for future research.

Keywords: Multi-Resolution Modeling, MRM, aggregation, disaggregation, MRE, MRM approach.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 Research Motivation

Military simulators have been developed for several reasons; economic, for avoiding a dangerous situation, and so on. Connecting two or more different simulations has been attempted for certain purposes. It is expected that the interworking of these simulators will be important in the future, and the MRM federation system technologies will be a good method to achieve this goal in the military.

This research investigates the military motives of MRM system, current methods, and recommendations on how to build an MRM for military purposes. This report also proposes the standard steps for developing the MRM federation system.

1.2 Problem Statement

It is difficult to make different simulations work together because each simulation is designed to fit their own unique requirements. Each simulation has different logistics in every aspect, such as engagement, representation, time, and so forth. For this reason, only a few studies have been made, despite various motivations for and the feasibility of MRM.

Davis Paul and Bigelow articulated the obstacles of MRM in their paper titled ‘Experiments in multiresolution modeling’ (Davis & Bigelow, 1998) as follows;

(1) The Model developer had no guideline for MRM design.

(2) When switching to low resolution, the poor conversion strategy discards important information degrade MRM system.

(3) *Aggregated models sometimes yield the wrong result because of unreasonable reason. This problem can be described as a data mapping problem or inconsistency problem.*

(4) *People are looking forward many things from MRM system, set high standard goals, and then reject the MRM system when it fails to meet their requirement.*

However, the goal of modeling is to recognize good approximation, though such approximation is only useful in context for some degree of accuracy. For this reason, the MRM study is considered worthy of continuing.

1.3 Research Objectives & Contributions

The objectives of this research are:

- To provide in-depth, comprehensive literature review about MRM for the future research
- To put various notions and approaches of MRM together in this report.
- To build the environment for the MRM experiment for the Simulation Interoperability Laboratory (SIL) of the University of Central Florida (UCF). Two different MRM systems using COTS simulators will be built.
- To define “trigger” or “triggering”, and its function in the MRM system through the literature review.
- To propose the steps to build an MRM federation system.
- To build a naval scenario to implement COTS simulations in the MRM federation system.

This research provides the basis for a time-based MRM system of further experiments to develop a new approach of MRM to overcome current obstacles. It is important to put scattered notions and approaches related to MRM together in one report for future research. Although many types of research have been conducted with military simulations, it is difficult to see the details because those research results are withheld as they are confidential. This research can provide many observations of MRM systems through the use of COTS simulators. Also, this work provides a comprehensive literature review about current MRM approaches to contribute to future investigation of MRM.

1.4 Thesis Overview

This research has six overall chapters. The research motivation and the context of this research are described in Chapter 1. Chapter 2 explains the basic concept of modeling and simulation (M&S), what is a military simulation, its classification, and MRM motivations. Chapter 2 articulates the definition and deals with various current MRM approaches with examples. Chapter 3 explains the steps to develop an MRM system and presents a methodology for how to construct two COTS simulations. Experiment environments are specified in Chapter 4, and the design of two MRM federations and observations of case studies are described in Chapter 5. Chapter 6 summarizes the research, and presents limitations and future works.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

The ultimate goal of the military is to win a battle and protect the country. Therefore, the military needs effective tactics, verified and proven operation plans, skillful soldiers, and powerful weapons to achieve their goal. As high-tech weapons emerge in the battlefields of today, everything (tactics, plans, training, etc.) has to be changed to keep abreast with those high-tech weapons.

Modern warfare can be described as a technology war. As we have seen in the Iraq war in 2003, advanced weapons overwhelm older battlefield tactics. However, it's difficult to train soldiers in a short period of time on how to use those weapon systems, as some of the systems are beyond the human cognitive abilities boundary. Furthermore, the price of certain weapon systems is too high to give every soldier first-hand experience (such as missile launching or escaping the cockpit of an aircraft).

M. Petty accurately articulated that Modeling & Simulation(M&S) is very valuable when used in an area where it is generally difficult to exercise, too expensive, or too dangerous to experiment in the real world, especially in the military field (M. Petty et al., 2012). For instance, one Tomahawk missile (land attack missile which has a long-range) costs around US$1.3M, an F-35 lightning II, all-weather stealth multirole fighter, costs roughly US$100M. An Ageis ship is US$ 1.843B per ship. Thus, using these weapon systems only for training purposes is extremely limited. The more sophisticated the weapon system becomes, the greater the military demand becomes for intensive training or testing those weapons to maintain their combat readiness. Those are the reasons why militaries have started to build and develop many kinds of simulators.

2.2 Modeling and Simulation (M&S)

What is modeling? Anu Maria explained it in her 1997 winter simulation conference paper ‘Introduction to modeling and simulation’ as follows;

“Modeling is the process of producing a model; a model is a representation of the construction and working of some system of interest. A model is similar to but simpler than the system it represents. (Maria, 1997)”

Maria (1997) says that an ideal model is a wise trade-off between realism and simplicity; that a model should not be too complex to understand or experiment with, but at the same time, it should be close to the real world. That’s why MRM is valid for building a good model.

“*A simulation of a system is the operation of a model of the system. In its broadest sense, simulation is a tool to evaluate the performance of a system, existing or proposed, under different configurations of interest and over long periods of real-time (Maria, 1997).”*

The purposes of simulation are to minimize the chance of failure, to optimize the use of resources, to curb unexpected bottlenecks, and to achieve optimization of system performance. Simulation can give an answer for what the best kind of simulation design is for a new project, or what resources are needed for the project. Simulation can be used before a legacy system is modified or needed for building a new system. It is built according to a model and allows for the visualization of the estimated result or accumulated data.

As society becomes more complex, everything goes beyond previous human cognitive ability. Thus the importance of Modeling and Simulation (M&S) will grow. Especially in the military field, its importance weighs more than in other fields. M&S can be used for War games, training, decision making, analysis, weapon acquisition, and so on.

These simulation systems can be divided into two large categories, as you can see in Figure 1; Stand-alone simulation or Distributed simulation system/Federated simulation (Park TaeWoong, 2015). Most simulation systems would be developed as a stand-alone simulation system as one product. Figure 1 (left) depicts the fundamental concept that a stand-alone simulation which is designed to simulate a complex model while operating independently. For example, K-1 tanks simulator and F-15 fighter training simulator are typical stand-alone simulations[1].

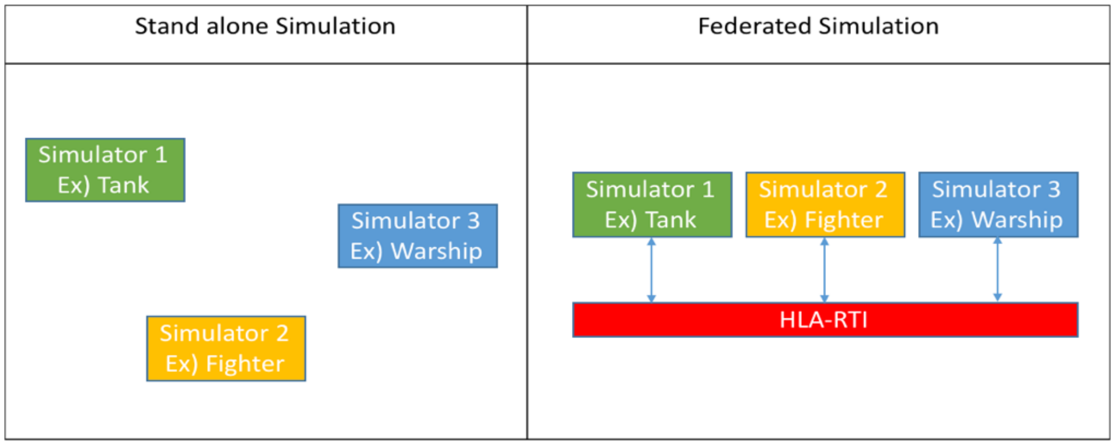


Figure 1 Stand-alone & Federated Simulation

A builder might create a stand-alone simulation for a specific purpose; for example, SIMBox F-15 pilot training simulation is designed for training a pilot without loss of an expensive aircraft and human life. But others have also found benefits through linking together

[1] Strictly speaking, the recent modularity of the simulation cannot be called it is a stand-alone simulation, because one simulator can consist of multiple modules of sub simulation.

different simulations. That’s the reason why simulations keep on expanding within the distributed/federated simulation area. Federated simulation systems have tremendous potential. Federated simulation systems can have massive groups of pilots participate together in the same exercise, or it can help provide a more precise analysis for future warfare. Federated simulation systems can help users to take advantages of reusability for economic reasons.

However, to make stand-alone simulators work together in the same federation, there needs to be a way to share the logic and information. Standard Simulation Architectures (SSA) are needed for this work. SSAs allow individually working models to interoperate through a network, so as to simulate in a similar scenario or situation collaboratively. High-Level Architecture (HLA) is the representative SSA of building a simulation when it comes to making a federation. RTI is a software to help interact between federates. HLA and RTI will be discussed in later chapters.

2.3 Military Simulations

Military consists of multi-rank personnel, different branches based on their weapons and missions. This diversity of ranks and branches makes demands for their bespoke simulators to train their soldiers.

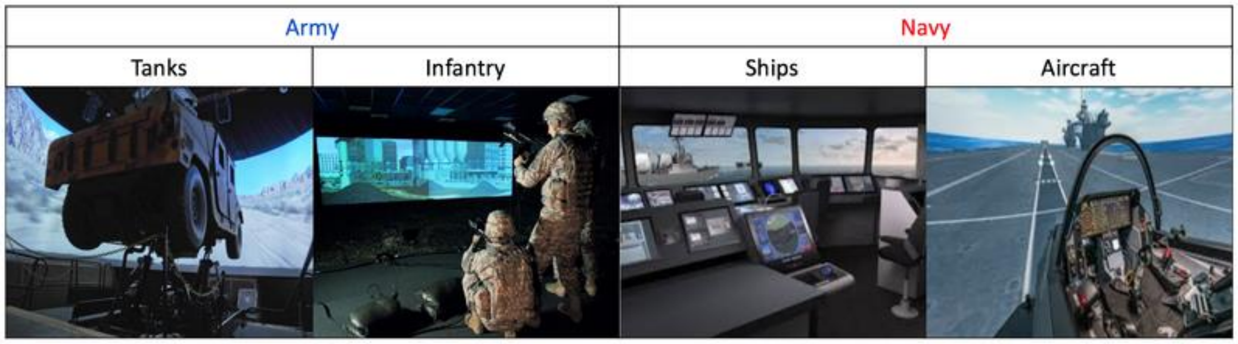


Figure 2 Diverse simulators for different branches and weapons

For instance, as you can see in Figure 2, different simulations are needed depending on the characteristics of each group. Even under the same branch, a battalion of tanks and battalion of infantry need different simulators to train their soldiers.

2.3.1 Classification of simulation from the perspective of human involvement

U.S. Department of Defense (DoD) classified the military training simulation as Live, Virtual and Constructive simulation. The commonly used definitions of Live, Virtual and Constructive simulation systems are shown in Table 1.

Table 1 The definition of Live, Virtual, and Constructive Simulation (DoD Directive, 1995)

Classification	Definition
Live Simulation	A simulation involving human operating real systems in a real environment.
Virtual Simulation	A simulation involving human operating simulated systems. Virtual simulations inject human-in-the-loop (HITL) in a central role by exercising motor control skills (e.g., operating a fighter), decision skills (e.g., commanding post-exercise), or communication skills (e.g., C4I exercise).
Constructive Simulation	Models and simulations that involve simulated human operating simulated systems. Although human makes inputs to such simulations, they are not involved in determining the outcomes.

According to Park TaeWoong (2015), the military simulation system can be categorized based on the type of human involvement. Table 2 shows the characteristics of differences in each simulation category.

Table 2 Characteristics of differences in each simulation category

Factor	Live	Virtual	Constructive
Human	Real	Real	Simulated
System	Real	Simulated	Simulated

A human who is operating the real systems participates directly in Live simulation such as a warship or aircraft. According to Jung Wonil (2017), a good example of live systems is the Air Combat Maneuvering Instrumentation (ACMI) system. The system is attached to real aircraft; for example, the communication module which can provide information such as location, speed, acceleration, system orientation, and weapon status of the aircraft sends real aircraft information to the commanding post for commanding-post related exercises. Daly and Thorpe (2009) distinguished Live simulation from Synthetic training which is executed with real people using real equipment in a virtual environment.

Virtual simulation involves real people in a simulated system world. Humans in the virtual loop simulation are designed to train people how to control, monitor and make the decision for the specific weapon system. Most of the training simulators are included in this category of simulations. Constructive simulation involves simulated people using simulation equipment in a simulated world, but they do not participate in determining the result of the simulated world.

Even though there is a distinction of classification, some simulators cannot be clearly placed in one of those categories. For example, a virtual simulation usually includes both virtual and constructive entities. This is the reason why many simulations look like a hybrid system that contains a mix of entity types (Hodson & Baldwin, 2009).

2.3.2 Classification of simulation from decision-making level

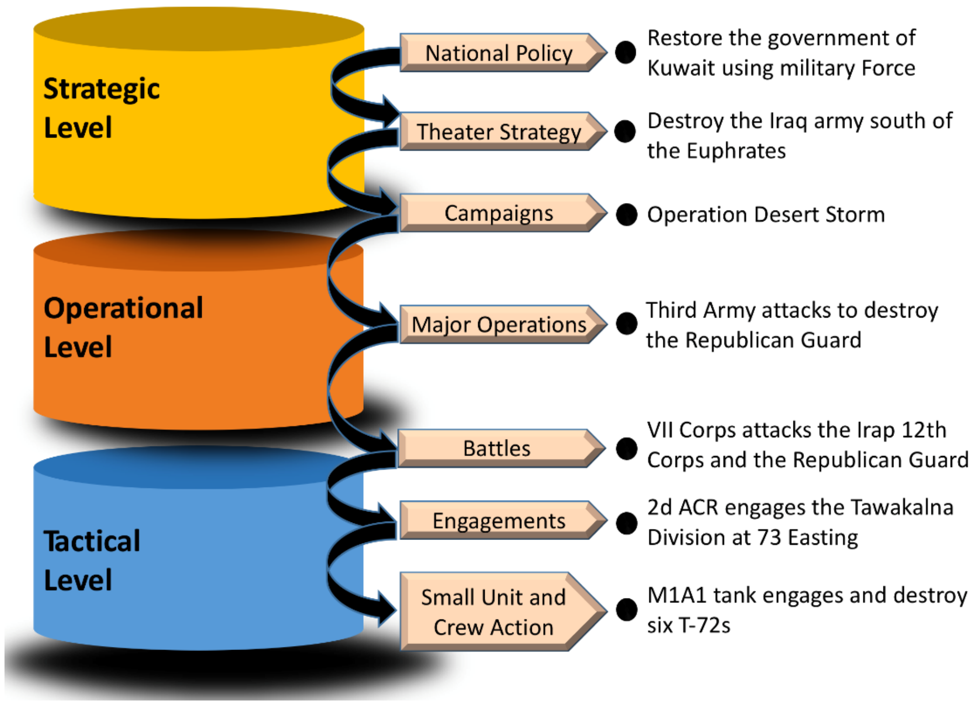

Strategic Level
Operational Level
Tactical Level
National Policy
Theater Strategy
Campaigns
Major Operations
Battles
Engagements
Small Unit and Crew Action
Restore the government of Kuwait using military Force
Destroy the Iraq army south of the Euphrates
Operation Desert Storm
Third Army attacks to destroy the Republican Guard
VII Corps attacks the Irap 12th Corps and the Republican Guard
2d ACR engages the Tawakalna Division at 73 Easting
M1A1 tank engages and destroy six T-72s

Figure 3 Major decision level of the military

Military decision making has three main levels: strategic, operational, and tactical. The military needs a multi-level simulator to train their multi-rank personnel. Figure 3 shows that there were three major levels of decision making in the military field with the Iraq war in 2003. To achieve the highest strategic goal, a sub level of decisions was set up.

At higher level of this hierarchy, the operational ability of commanders and staff is key. Therefore, the simulator should give them a big picture of the theater and change of battlefield situation. However, at a lower level of this hierarchy, the proficiency of a small unit and crew for actions are the most important factors to win in a battlefield. Thus, these kinds of simulators focus on the detail of reality to give soldiers vivid experiences without risking their lives.

U.S. DoD often classifies modeling and simulations into four levels: engineering level, engagement level, mission level, and campaign level. It is usually described as an M&S pyramid (Figure 4)

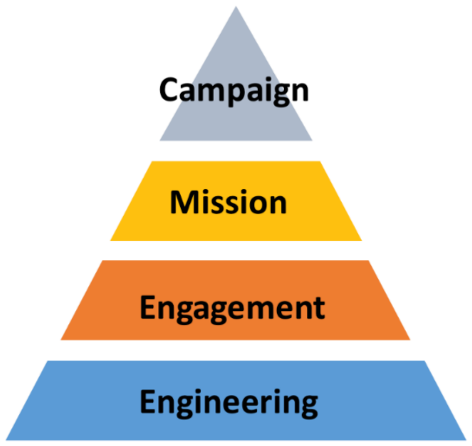


Figure 4 Traditional Military Modeling and Simulation pyramid

Coolahan, J.(2003) explained those four different levels of the M&S pyramid. The campaign level simulation should be able to represent the overall combat situation because it usually is used in warfare analysis for higher rankers. Mission level simulators are designed for relatively smaller areas such as missile defense, air defense, and power projection from the sea. Engagement simulators simulate the ability of specific weapons in most DoD weapon system projects. Engineering-level simulations are used in a more specific and scientific area, as well as in defense communications, space applications, and strategic systems test and evaluation (T&E). Some simulators cannot fit perfectly into specific levels, but there are many different simulators in the military for their purpose.

2.3.3 Classification of simulation from purpose perspective

Simulators can be differentiated based on the purpose of the simulator; for example, training purpose simulation and the weapon system test simulation. Jung S.C (2001) mentioned that there are four different M&S applications systems for Korea's Military;

(1) Massive troop movement exercise,

(2) Small group training,

(3) Analysis of Military capability, and

(4) Weapon acquisition.

These are the categorization of military simulators in terms of their purpose, although it can be argued that the analysis of military capability and acquisition of weapons can be merged together under the category of weapon tests purpose for a simulation.

A massive troop movement exercise simulator, in general, is designed to try and test a military operation plan for high-rank officers. Through these types of simulations, they want to find out defections of operation plans and optimize it. For example, US Joint Theater Level Simulation (JTLS) and KOR Chang Jo-21 model are representative of this massive troop movement simulator. These simulators use Lanchesterian equations to adjudicate conflict between massive troops, typically battalions or brigades (Bowers, 2003). This method enables fast game operation but doesn't give accurate results, which means less reality.

Small group training simulators can be designed for relatively small battle or training on how to use a military's weapon system. For example, some models are designed to train pilots for a new aircraft, whereas others are designed to provide a second experience in warfare without any loss of soldiers and properties. The US Joint Conflict and Tactical Simulation (JCATS) and KOR virtual simulator of K-1 tanks are examples of small group training simulators. They typically depict an individual weapon or entities and adjudicate combat between individual objects, such as individual vehicles or combatants, using individual shot probabilities (Bowers, 2003). This method can produce reasonable results that reflect reality better than the Lanchester equation.

The two aforementioned simulators have different sized battlefields. Massive troop exercise simulators describe a theater or campaign of warfare. It is mostly focused on the maneuver of massive troops. This simulator is fit for the commander or his/her staff of regiments exercise. They don’t need the specific details regarding the movement of enlisted soldiers. Thus, this simulator mostly uses symbols of troops; in this case, they do not want to spend much time seeing what happens at the end of the simulation, so they do not use real-time simulation. On the other hand, small group training simulators describe small size battles, and the emphasis is on details that depict the real world as it is. For example, the fighter training simulator provides details that closely resemble a real aircraft experience; this allows the pilot to have a richer experience.

Analysis of Military capability and Acquisition of weapons simulators are designed to test or examine the performances of weapons or develop tactics optimized for the weapons. These kinds of simulators are designed for a specific weapon, and it should describe a very detailed performance of its weapon system. This kind of simulator requires scientifically proven data and algorithms in order to obtain an analysis of its performance in the simulation. These types of simulators can be used for the development stage of weapons to optimize their design to achieve their purpose. Figure 5 is an example of categorized military simulators based on its purposes.

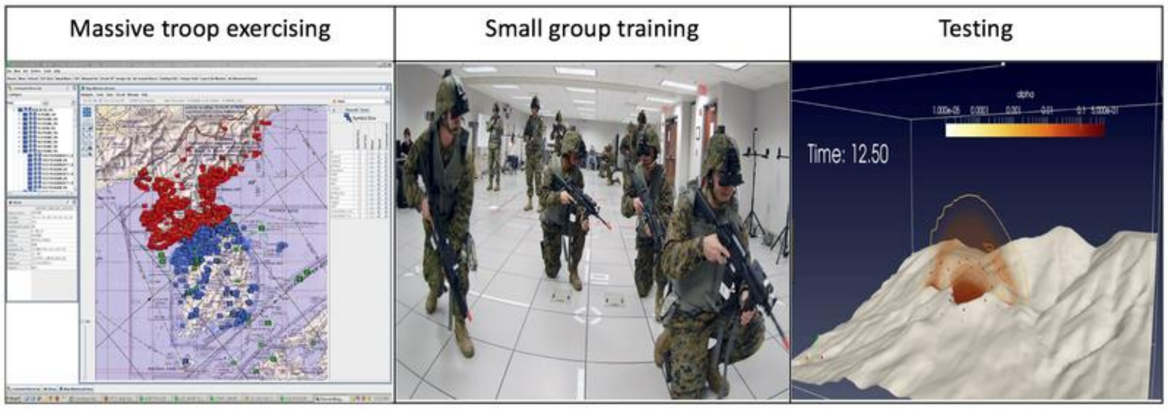


Figure 5 Example of Military simulators for different purposes

The military has already developed a lot of different simulators based on multi-level requirements, but in the end, all of their efforts with these simulators go towards one ultimate goal: to win a future war and defend their nation. Therefore, there is a high demand for linking and making a federation of these simulators. MRM can be a solution to link alliances or multi-nation military simulations for joint exercises.

2.4 Definition of Multi-Resolution Modeling (MRM)

Multi-resolution modeling is a relatively new research field. At the first stage of MRM, it was called VRM. The definition of VRM is addressed in Davis (Davis, P. K., and Hillestad R., 1993). VRM is building brand new models or federated model families so that users can change resolution at which phenomena are treated.

Hong S. Y(2007) define “Multi-Resolution Modeling (MRM) as represents a real system as a set of models at a different resolution in different abstraction levels from the viewpoint of simulations objectives.”

Davis and Bigelow(1998) define multi-resolution modeling as follows;

"(1) building a single model with alternative user modes involving different levels of resolution for the same phenomena; or

(2) building an integrated family of two or more mutually consistent models of the same phenomena at different levels of resolution; or

(3) both."

An MRM approach can be conceptualized with this definition and understanding that different levels of resolution can be implemented into a single model or family of models[2]. The level of abstraction desired to depict a model is related to the different levels of resolutions. The level of abstraction in an MRM federation system can be associated with the number of input/output parameters related to a specific simulation model or reality.

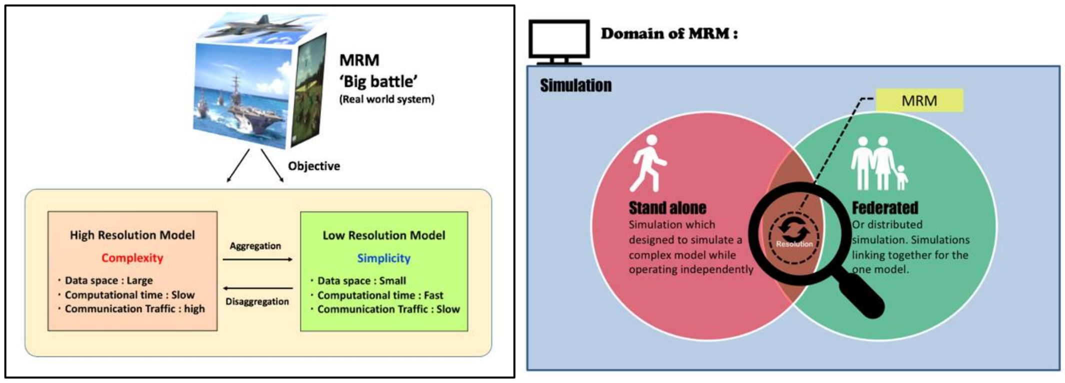

MRM
'Big battle'
(Real world system)
Objective
High Resolution Model
Complexity
· Data space : Large
· Computational time : Slow
· Communication Traffic : high
Aggregation
Disaggregation
Low Resolution Model
Simplicity
· Data space : Small
· Computational time : Fast
· Communication Traffic : Slow
Domain of MRM :
Simulation
MRM
Stand alone
Simulation which designed to simulate a complex model while operating independently
Resolution
Federated
Or distributed simulation. Simulations linking together for the one model.

Figure 6 Multi-Resolution Modeling Concept and Domain

[2] That means MRM can be implemented into a stand-alone simulation and federated simulation.

Hong and Kim (2013) as shown in Figure 6, express that MRM can be a key technology to describing a large-scale and complex model in a various application field. MRM is a simulation system which can be operated either by the changing of models in different resolutions or by the communication of information among different levels of models from different levels of resolution perspectives.

In summary, as demonstrated by Figure 6 (right), MRM is a simulation which can be a stand-alone or federated simulation but should have a changing resolution to meet certain requirements of users.

2.4.1 Dimensions of Resolution

Dimensions of Resolution is an important notion in MRM. According to the MRM definition of Davis & Bigelow, the resolution is key to distinguish from other simulators. Thus, we need to figure out what is a resolution in M&S.

All models can be described as abstractions of reality; some models are more detailed than others. Those details can vary depending on the scope (input/output domain range treated, and the extent of the system) and on the resolution. Models also can vary from the perspective that they represent, and in their user modes (Davis, P. K., and Bigelow, J. H., 1998, p-4).

The resolution can be defined as "the degree of detail and precision used in the resolution of real world aspects in a model or simulation"(Hong S.Y, 2007). The question remains what kind of details there should be, such as entities, time, description, and so forth, and what parameters to use to measure whether it is low or high resolution.

The resolution is a multifaceted and relative concept, as indicated in Figure 7 (Davis, P. K., and Hillestad R., 1993). For instance, high resolution usually describes fine-grained entities such as individual fighters rather than a group of fighters, and each entity has more attributes than units (e.g., number of air to air missile, position of the missiles, rate of fire and so forth), a more detailed process of movement, speed, time, terrain and attrition. (Davis, P. K., and Bigelow, J. H., 1998).

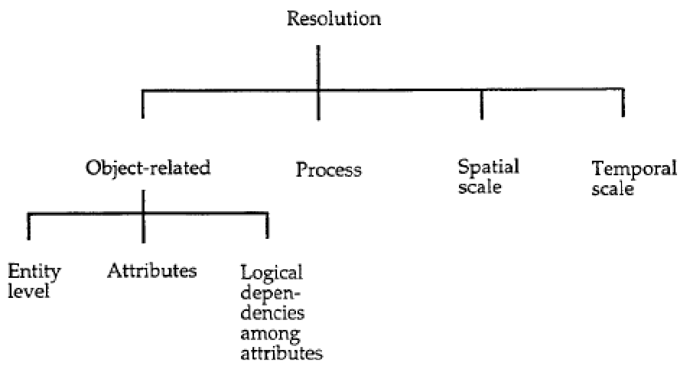


Figure 7 Aspects of Resolution *(Retrieved from Davis, P.K., and Bigelow, J. H.,1998)*

Mullen et al. (2013) suggested an example of the temporal scale using the modeling and simulation pyramid shown in Figure 8. The degree of resolution decreases as one moves from the engineering to the campaign level. Execution times, as a multiple of real time, varies greatly across the levels (Mullen et al., 2013).

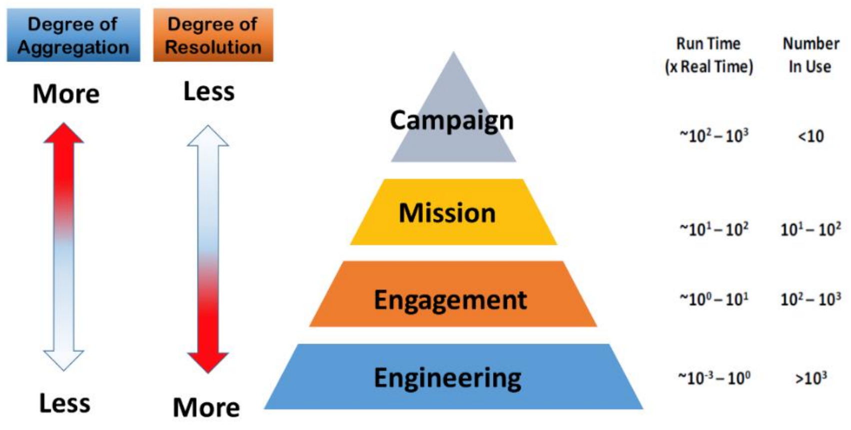


Figure 8 The overall degree of resolution

Campaign level simulations must be able to represent the order of 90 days of combat, and because they are typically executed many times for uncertainty analysis, there needs to be an ability to execute very quickly in order for the simulation to be useful. If a simulation is needed in order for a decision maker to decide which method is the best, a quick-run simulation would be the most useful. If the simulator runs in real time, this increases the amount of time that the decision maker needs in order to make a decision as he/she has to wait for a long time to see the result. Mission level simulations typically execute 10-50 times faster than real time. Engagement level simulations typically operate in the real-time range or slightly faster, but simulate a shorter period. For instance, a simulator which is designed for training pilots should run in real time in order to provide a similar environment and experience compared to the real world. At the engineering level, run-times are diverse as there is a mix of the other three levels combined with some very computationally intensive simulations. For example, the simulator for the missile test is recommended to operate at the desired time level. This is because the user wants to be able to manipulate the ability to slow down or speed up to a certain period that they want to observe.

The relationships of resolution among federates can be confusing and complex because the resolution has a multi-aspect and relative concept. Resolution of a certain model can be lower in some aspects, higher in others, and the same in the rest. Aggregation and disaggregation are closely related to resolution, but the quality of resolution in a simulation cannot be judged solely based on whether it has an aggregated unit in its model; this does not equate to a low-resolution model since resolution is a relative concept (Davis, P. K., and Hillestad R., 1993). When it comes to comparing two models A and B, model A could have a higher resolution in some aspects, but lower resolution in other aspects. Therefore, it is not an easy task to decide which models are higher than others, and MRM is a part of the art of deciding what level of detail is needed in various aspects (Davis, P. K., and Bigelow, J. H., 1998).

The crux of resolution in MRM is not a matter of which simulator is higher or lower resolution than others. The important thing is if there is a difference in resolutions among federated simulations and if it has meaningful connections in MRM. The goal of MRM is to achieve users' requirements through connecting different resolution simulations.

2.4.2 Composability and interoperability

When researching an MRM approach implementation for a simulation system, the concepts of composability and interoperability arise. Composability is a system design principle that deals with the relationships of components and it is the capability to select simulation components and assemble them in variety of combinations to make a valid simulation system in order to meet user requirements (Petty and Weisel, 2003).

The definition of Interoperability is the ability of a computer system to run programs from different vendors, and interact with other simulations or computers across local or distance-area networks regardless of their operating systems and physical architecture (Petty and Weisel, 2003).

Davis and Tolk(2007) distinguished composability from interoperability. Interoperability is the ability to handle software and implementation details containing data exchange elements based on data interpretation, which can be distributed to the levels of semantic and syntactic interoperability. Interoperability focuses on how the specific models are executed. Composability, on the other hand, describes the components of issues on the modeling level. The model is a meaningful abstraction of the reality used in the conceptualization implemented by the resultant simulation system. Composability treats the contextualized common conceptual models. In summary, it is convenient to refer to the interoperability of simulations and the composability of models.

2.4.3 Multi-Resolution Combat modeling (MRCM)

A multi-resolution combat model, as defined in Tolk(2012), is a federated model that connects an aggregated unit level combat model (i.e., company, battalions, and brigades), with a disaggregated entity level military model (i.e., fighter, ships, helicopters, and tanks).

2.5 Current Issues of MRM

Multi-Resolution Modeling is focusing on resolving representational and conceptual gaps arising from multiple resolution levels in federated simulations. The main issue is simulating an object or unit accurately and its attributes concurrently. If an abstraction of the object and its attributes are simulated concurrently, then all interactions in overlapping specific periods with the same abstraction and its constituents will be at both levels of models. (Natrajan A., 1997). However, many issues arise with simulation couplings: temporal inconsistency, mapping inconsistency, and so on.

2.5.1 Data consistency

Data consistency has many facets that need to be addressed, such as data distribution problems, syntax, semantics, pragmatics, assumptions, and validity consistency. All of these facets are required to build a good MRM system. The following research will highlight two types of reasons in which data inconsistency arises in MRM at different stages.

2.5.1.1 Inconsistency from the Development stage of Modeling

Davis & Tolk explained composability of M&S in their 2007 winter simulation conference paper (Davis & Tolk, 2007). There is a fundamental cause in which inconsistency happens when two different simulations are linked. It is a domain of composability. Five issues involve composability.

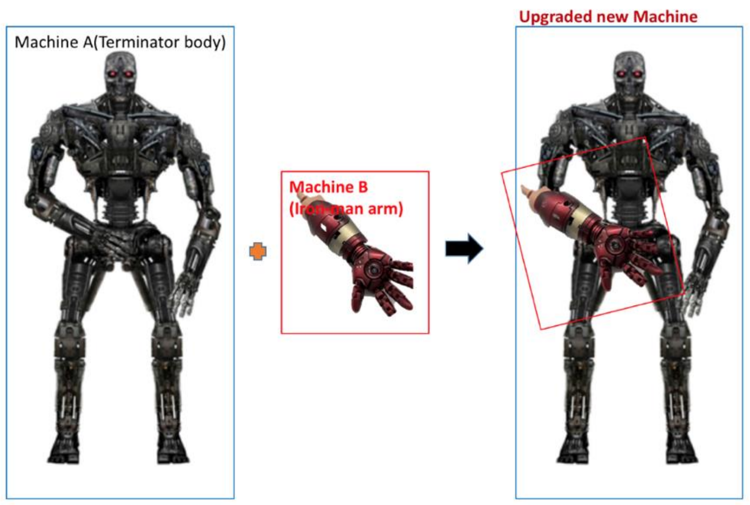


Figure 9 MRM concept example

First, keeping the consistency of syntax means that two simulations or models can work together on an engineering level (Davis & Tolk, 2007). For example, suppose that L corporation wants to build a new powerful humanoid weapon and they want to re-use legacy machine A(Terminator body) and B(Iron-man's arm) as shown in Figure 9. The resulting new machine represents a new MRM system and Terminator and Ironman both show a different simulator. Suppose machine A(Terminator body) is drive by a biological signal, whereas, machine B(Iron-man's arm) only accepts an electrical signal. In this case, it would be a meaningless composition even if the outer physical connection is successfully connected. For it to be a successful composition, the digital output from model A should be compatible with the digital input of model B. In other words, they should use the same computing language or protocol. If there are differences with syntax, then one sub-model cannot compute or understand information from the other sub-model. Therefore, the same protocols are essential to deal with this problem.

Second, consistency of semantics ensures that data have the same meaning in the sending and receiving model (Davis & Tolk, 2007). This means that the data which is produced by model A has same meaning when it is used in model B. Back to the terminator example, if the user ordered the new machine (MRM system) to ‘Attack’, this could have a different meaning to both model A and B. For model A, ‘Attack’ might translate to hitting a target with its fist. However, model B might have more than one option, such as using its fist or firing a laser beam. Tolk(2003) recommended using a common reference model to overcome these issues by distributing information exchange objects to standardized data elements. Solutions are languages based on the Protocol Data Units of the Distributed Interactive Simulation Protocol, a standardized common reference model, or the object model of the Test and Training Enabling Architecture (TENA).

Third, consistency of meaning is not always the same, because the same word could mean different things depending on the context. (Davis & Tolk, 2007). Even though semantics make sure that the individual data are straightforwardly identified, pragmatics puts those data into the context of how they are used in the simulation or model. Although both models agree on the meaning of all data elements to exchange, these data elements can be used in different situations, making it difficult to exchange these data in the runtime environment. For instance, a “defense position” may refer variously to the location of the planned defense, the current position of the defending unit, the location of the units to be defended from enemies attack, and so on. The context is not ambiguous only in the context of a complex element, such as the location of the defending unit.

Forth, the difficulty continues even after consistency is maintained across models, due to the way words are used. The method to calculate the data may not be suitable for federates. Sometimes it is a matter of accuracy or precision, but in other cases, it is much more subtle.

Fifth, there remains a question about whether the model is correct. The assumptions of the model may actually be incorrect, the logic or computation used may be incorrect, or both. Also, a model that is properly valid in one context may not be valid in another context (Davis & Tolk, 2007). Here are some overlaps. Configuration validation issues can also be studied at the formal level. Weisel at el(2003) did so and showed difficult problems. Tolk(1999) provides some examples of the validity of a federation that does not guarantee the validity of the outcome.

In summary, high composability help to solve MRM connecting problems and to prevent inconsistency in MRM. Hence specific protocols for building simulation is needed for MRM. However, this may sacrifice various simulation approaches which help represent the real world, because while strict rules of building simulation might be helpful for MRM, it may make the builder's innovation shrink. Therefore, demanding pure composability is seen to be unreasonable.

2.5.1.2 Inconsistency from Data distribution stage of the simulation

According to Natrajan A. (1997), the MRM problem has been known as the aggregation/disaggregation problem in distributed simulations (Natrajan A., 1997). That is because a common solution is to change the resolution of an LRE (or HRE) dramatically to match the resolution of other encountered entities (Reynolds P., F., 1997, p. 369). When it comes to disaggregation (LRE → HREs), the MRM have to generate more attributes to create HREs in order to fit in the high-resolution model. Mapping mismatch happens when an entity undergoes a series of transitions across levels of resolution resulting in a state it could not have achieved in the simulated time spanned by that sequence. Every scheme that an object converts along its resolution level (e.g., aggregation - disaggregation - aggregation) must distribute attributes consistently across levels. Conversions in particular should enable switching levels without changing attributes unless other interactions occur. Poor translation strategies cause "jumps" in the entity state. A sharp increase in the visual perception occurs when the state in which the state change is detected violates the simulation meaning due to a rapid transition between states. Low-level resolution information (aggregated level) may not be sufficient to provide consistency in a high-level resolution (disaggregated level) entity. When disaggregated into aggregated conversions, some information related to the HRE may be lost (e.g., sub-entity's position). As a result, the second transition (from the aggregation to the disaggregation) can be a separate state that does not match the first separated state. This is because the standard algorithm or principle has been applied to specify the location of the item. A perfectly maintained translation strategy is desirable, but often it may not be easy to find. In such cases, any potential discrepancies that may arise from translations from one state to another should be treated differently.

For example, as depicted in Figure 10, model A assumed a simulator which is designed for a commanding post exercise. Therefore, model A simulates large-scale troop movement, such as regiments, a battalion of infantry, or amphibious assault forces in the navy. Model B assumed a simulator which is designed for training soldiers or a team. Therefore, Model B simulates relatively small-scale troop movement, and it is relatively more detailed than Model A, such as depicting tanks, helicopter, or warships.

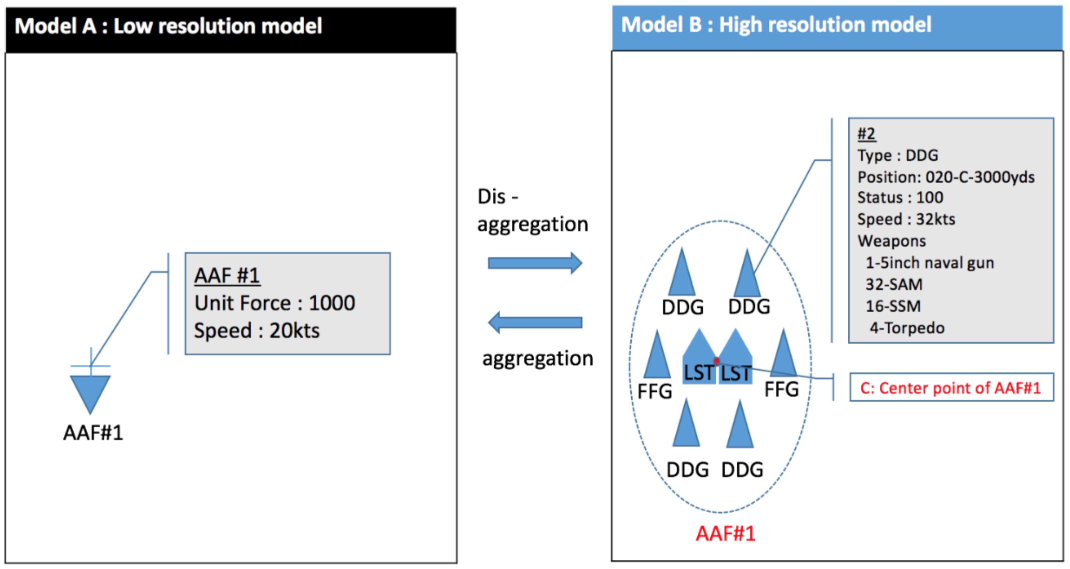


Figure 10 Imaginary example of data inconsistency

When disaggregation occurs, LRE unit (in this situation AAF[3]#1) in Model A needs more information to become HREs entities (in this situation, 4DDG, 2FFG, and 2LST)[4], this process is known as disaggregation. In this case, type of ships, relative position, status, speed, and weapon attribute should be produced for to adapt in Model B because Model B requires those attributes to run the simulation. Properly initializing these new objects requires information from the aggregated units using some rules and algorithms that describe the proper military doctrine of how the unit will respond to a particular situation (Sven Skoid, 1998). The method to create attributes will be case by case depending on models. This process can be simple or problematic depending on the property/attribute (e.g., the location of the disaggregating unit and its formation, their weapon states and so on.). For example, AAF#1(LRE) has undergone a disaggregation process due to encountering several enemy aircraft HREs as depicted in Figure 11, and #2 DDG

[3] AAF: Amphibious Assault Forces: amphibious assault force is a naval force to achieve a type of offensive military operation. This force is design for delivering troops to shore. AAF is consists of several amphibious ships and defensive ships.
[4] DDG, FFG, and LST are abbreviation of naval warships: DDG represents a guide missile destroyer, FFG means a guide missile destroyer, LST is a landing ship tank for carrying troops.

has engaged with those enemies. From this engagement, #2 DDG received a minus 20 status damage from the enemy aircraft and fired 8 SAM missiles. Then #2 DDG's attributes will change in Status (100→80), and Weapons (32 SAM → 24 SAM). After engaging, HREs would undergo an aggregation process to move to the next point.

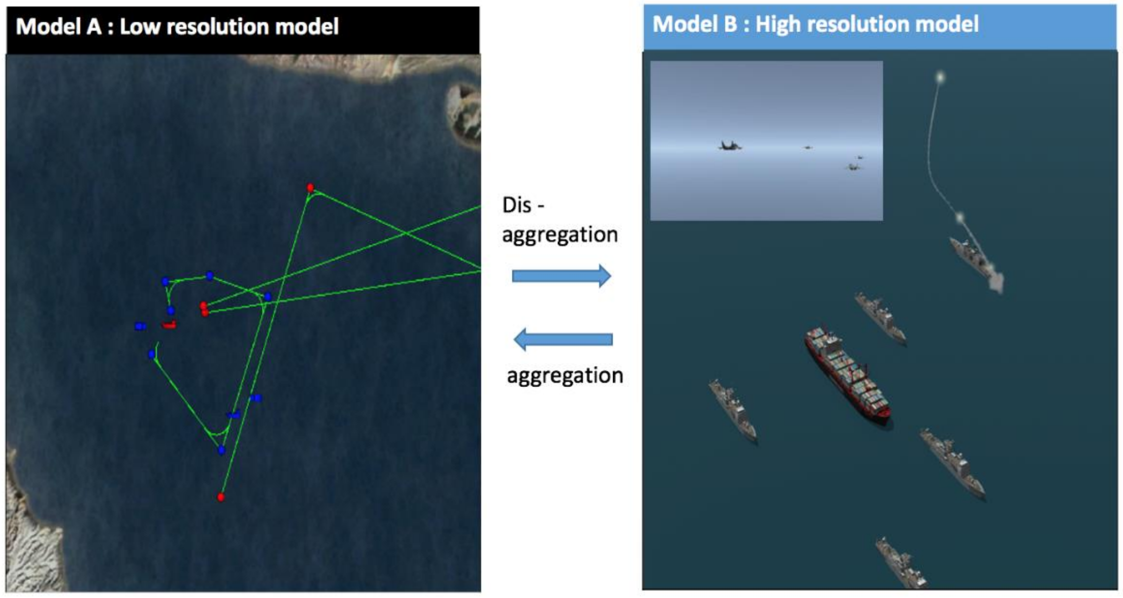

Model A : Low resolution model
Dis - aggregation
aggregation
Model B : High resolution model

Figure 11 Interaction in multi-resolution level (concept)

A problem will arise at the end of this interaction. When HREs aggregate to be LRE as AAF#1, this aggregation process should reflect the result of the engagement just a moment ago, #2 DDG's damage and ammunition attrition should be reflected into the AAF#1's(LRE) attribute. The aggregate-level information/attributes may not be sufficient enough to provide disaggregate-level consistency; some information about the HRE may be lost in the transition from disaggregation state to aggregation state, for example, specific vessel's status, and weapon attributes. Furthermore, when an entity undergoes a series of transitions across changing resolution, it results in a state that cannot be achieved within the time simulated by that sequence.

Every scheme that an object switches along a resolution level (such as aggregate-disaggregate) must map their attributes consistently across levels (Reynolds P. F., 1997).

2.5.2 Data flooding

Chain disaggregation is a continuously happening disaggregation phenomenon. Chain disaggregation causes unnecessary simulated entities to be involved in the disaggregation process rapidly. It is related to network flooding and thrashing. Typically, Low-Resolution Entity (LRE) is disaggregated when they meet HREs to interact at the disaggregate level. Other LRE units in the vicinity will also disaggregate as the LRE changed to HREs entities. Suppose HREs (E_a) meets LRE (E_B). Then, LRE (E_B) disaggregates to become a legitimate entity to interact with HREs (E_a) in Figure 12. LRE (E_B) become HREs (E_b). The other LRE (E_C) which is adjacent to HREs (E_b) also undergo disaggregation procedure to be HREs (E_c).

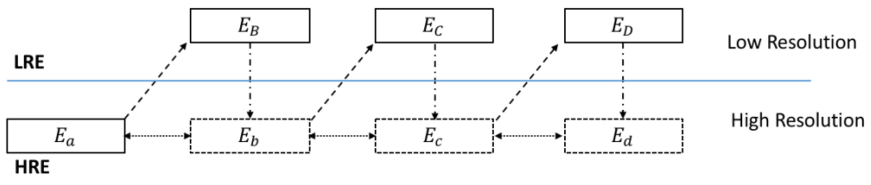


Figure 12 Chain disaggregation

These chains of disaggregation can occur repeatedly. Because the LRE has interacted and been disaggregated, it can easily extend to other LREs that have been forced to isolate this reaction. The initial disaggregation causes the domino effect. Chain disaggregation quickly increases the number of simulated entities. This chain disaggregation results in a load on processors and the network. There are some approaches to relieve data flooding; e.g. setting an adequate trigger condition, hybrid MRE, partial/pseudo disaggregation, cross-level interactions, high-performance hardware, etc. These concepts will be discussed later.

2.5.3 Time synchronization

Temporal inconsistency can occur when two entities have been conflicting or inconsistent representations of the state of a third entity at overlapping simulation times (Reynolds P.F., 1997). Typically, a high-resolution model runs its simulation based on real-time, but a low-resolution model used to implement time-stepped. Time synchronization issues are typically observed in linkages that run at very different time levels with different resolution levels.

For example, entities A1 and A2 interact with each other once every minute in low-level resolution model. A1 and A3 interact together in every second in high-level resolution model. Temporal inconsistency occurs in this case because A1 interacts with A2 once a minute, but A3 interacts with A1 sixty times a minute. A3 forces A1's state to change sixty times but those interactions do not adequately reflect into the low-resolution model. This causes a temporal inconsistency because A2 and A3 have inconsistent representations of A1 at the end of the larger time-step. Temporal inconsistency directly degrades the reality of the MRM federation.

2.6 Motivations for MRM

The motivations for MRM are increasing, even though there are various problems which are difficult to solve. If a high-resolution model depicting the underlying phenomenon is a perfect model, then why is it that the low-resolution model requires a more aggregated model (Davis & Bigelow, 1998)? If the military builds a simulation from scratch in a strict protocol, then it could easily solve the problems of MRM, however, they do not do this because of the following reasons.

2.6.1 Economic Reason

There are common economic reasons. Frequent use of high-resolution models is expensive, and time and data consuming. High-resolution models are voracious in data and run as in real time. Furthermore, the military has already developed a large number of simulators. If they can reuse those legacy simulators with minimum modifications, it can be a way to save a significant amount of money on the M&S for the military. Davis and Tolk (2007) explain that MRM is economically advantageous if it able to reuse existing software whenever possible because the software is expensive.

2.6.2 To get a new insight

Humans require the model to be cognitive because they have a different level of detail. Humans need different models to exploit the knowledge that comes from many different levels of detail. Integrating constructive (Low-resolution simulation) and virtual simulations (High-resolution simulation) provide comprehensive and realistic simulation exercises that allow different classes of soldiers to concentrate on the abstraction or level of detail that suits their needs (Tan et al., 2001). Multi-Resolution Modeling allows the training environment for a various rank soldier in the same simulation. Such a system can also extend the reach of the training environment. Low-level of detail with high abstraction aggregated federates are used to provide training to higher rankers who command an entire division or battalion. Therefore, these low-resolution models provide the whole picture of the campaign. A high-level of detail with low abstraction disaggregated entity level federates can be used to train lower-ranking soldiers, such as tank unit crews, Multi-resolution modeling federation, which combines all federates, enables high-level commanders to build training environments to interact with vehicle crews participating in the same exercise through command and control hierarchies (Tan et al., 2001). Dahmann (1997) said that no one-simulation can provide a satisfactory simulation to every user. Therefore, if a user or builder can link every simulator according to their needs, then the new system can satisfy the user's requirements easily. The synergy from the new environment may provide insights about new and efficient strategies, understanding some weapons problem, and so on. Achieving reliability in complex and sophisticated systems, even if the economy is not a problem, depends on not reworking unnecessary problems (David & Tolk, 2007).

2.6.3 To overcome the limitation

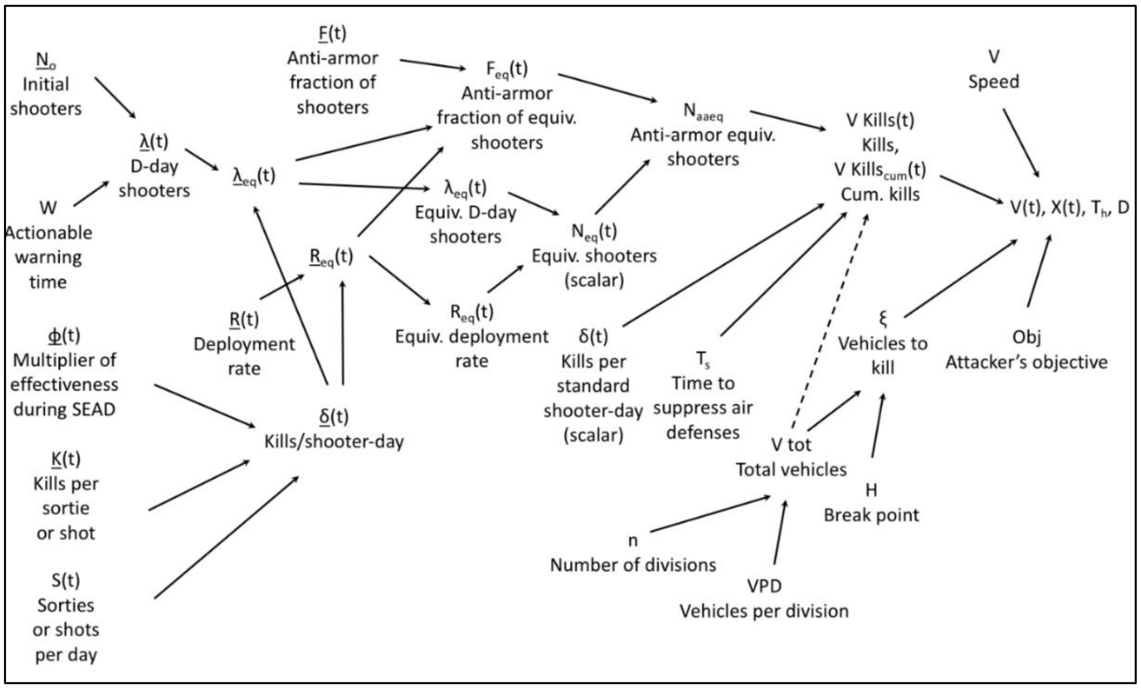
High-resolution models in the real world typically have limits that imply an important factor in determining a higher level of behavior. For example, most detailed models are "scripted", missing critical information about adaptive behavior and strategy. Consolidating simulations at different levels of abstraction also helps reduce workload and network traffic (Tan et al., 2001). The low-resolution process can reduce the workload on the system. For example, suppose a battalion moves toward a certain battlefield as per their commander's order, then using low-resolution simulation will save more computing power compared to a high-level resolution model. This also diminishes data traffic among their sub-simulations by reducing the number of updates that must be sent to each other. In some applications it is highly desirable to be able to use alternative sub-models developed with different perspectives and ideas from different people (Davis & Tolk, 2007).

Therefore, Low-resolution models are needed for exploratory analysis to provide a comprehensive high-level understanding of the problem and provide the potential value of alternative decisions. A low-resolution model is required for analytical agility for relevant reasons. That's why MRM is needed in the military field.

2.7 Current MRM Approaches

2.7.1 Integrated Hierarchical Variable Resolution (IHVR)

Integrated Hierarchical Variable Resolution (IHVR) modeling is an MRM approach developed that describes a procedure-oriented method that uses a hierarchical variable tree. The main variables are depicted in hierarchical trees with the lowest-resolution variables at the top. This hierarchical variable tree goes from low resolution to high resolution. Therefore, aggregation is at the top, and the most disaggregated areas are at the bottom.



$\underline{N}_o$ Initial shooters
W Actionable warning time
$\underline{\lambda}(t)$ D-day shooters
$\underline{\lambda}_{eq}(t)$
$\underline{F}(t)$ Anti-armor fraction of shooters
$F_{eq}(t)$ Anti-armor fraction of equiv. shooters
$\lambda_{eq}(t)$ Equiv. D-day shooters
$\underline{R}_{eq}(t)$
$\underline{R}(t)$ Deployment rate
$R_{eq}(t)$ Equiv. deployment rate
$N_{eq}(t)$ Equiv. shooters (scalar)
N_{aaeq} Anti-armor equiv. shooters
V Kills(t) Kills, V Kills$_{cum}$(t) Cum. kills
V Speed
V(t), X(t), T_h, D
$\underline{\phi}(t)$ Multiplier of effectiveness during SEAD
$\underline{\delta}(t)$ Kills/shooter-day
$\underline{K}(t)$ Kills per sortie or shot
S(t) Sorties or shots per day
δ(t) Kills per standard shooter-day (scalar)
T_s Time to suppress air defenses
ξ Vehicles to kill
Obj Attacker's objective
V tot Total vehicles
H Break point
n Number of divisions
VPD Vehicles per division

Figure 13 Hierarchical variable tree

One can clearly specify the procedures required to associate variables at various levels and calibrate based on modules (Davis & Bigelow, 1998). This represents the "everything-affectseverything" aspect of the real world. However, it is very hard to find its relationship equation between trees of variables. For instance, if an individual wants to get a regiment unit speed in a low-resolution simulation (i.e., the variable speed of regiment), then many variables in a high-resolution simulation level would have to be considered to determine the regiment unit speed, such as the speed of the soldiers or tanks in the regiment, the degree of manage, etc.

Davis and Bigelow (1998) argue that object-oriented modeling (OOM) focuses primarily on object hierarchies and the real challenges in MRM implementation are process-based hierarchical problems. It is very clear that this thought was before OOM was fully implemented and developed in the late 1990's and early 2000's. David and Bigelow (1998) discussed three cases in which the IHVR MRM method can be implemented because it may arise in a given modeling system. The specific three cases will not be discussed in this paper.

2.7.2 The module-based approach

According to Tan et al. (2001), the crux of MRM are aggregation/disaggregation interactions to make federates work together. Tan et al. (2001) explain that the information flow from individual objects must be aligned and integrated so that the overall representation is correct. Events drive the key point of disaggregation and aggregation. For example, the aggregation can be triggered by an event that causes the control of the entity being joined to be sent to the aggregated model, and the model starts to simulate the aggregation unit (Tan et al., 2001).

In short, modifications are essential to building a seamless federation. The point is where those modifications or modifiers are placed in order to facilitate a smooth resolution change. Tan et al., (2001) explains that this modification is the regulator. Regulator defines a set of entities or entities that handle aggregate/disaggregate procedures among federates during simulation. The regulator is responsible for providing the relevant information necessary for aggregation or disaggregation of a federate (Tan et al., 2001).

The first and basic concept is the module-based approach. In this approach, it is suggested to attach the regulator onto each federate as shown in Figure 14. This regulator is a modification to make a smooth connection to other simulators in the federation. Thus, the attached regulator module determines whether interactions that perform disaggregation requests should be serviced by a federate.

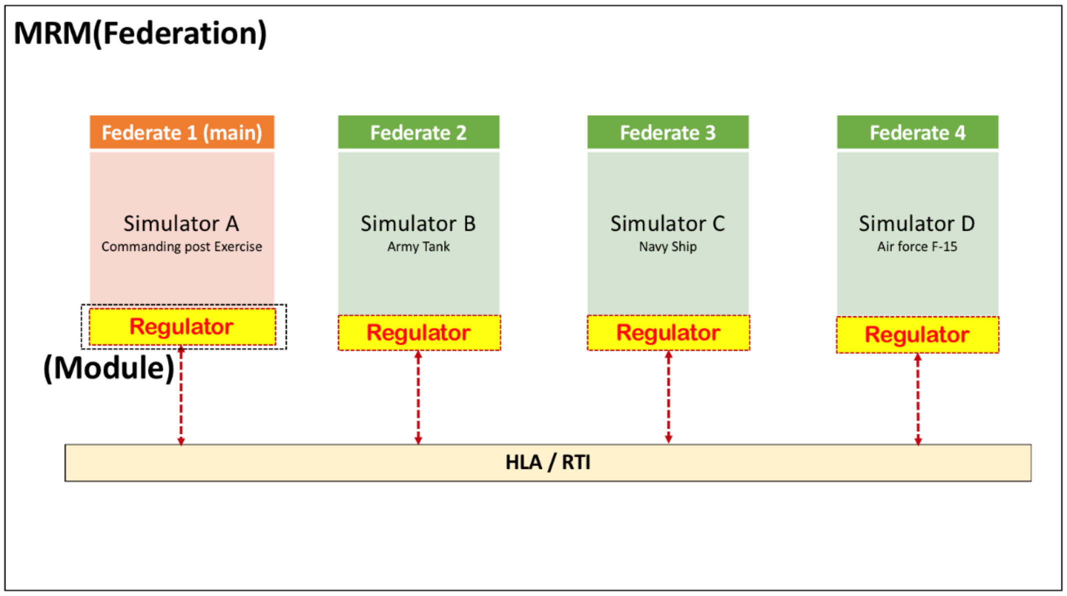


Figure 14 The module-based approach

Tan et al., (2001) discussed that this module-based approach has several problems. First, whenever an event triggers a disaggregation request in the main federate, then the rest of the federates also undergo a disaggregation procedure. This can cause a significant load on the network; a feasible federate is required whenever possible disaggregation events occur. Second, separate regulator modules spend more time to build an MRM system. This approach is useful when implementing a number of codes that perform tasks such as debugging, or performs similar tasks. More time and effort is required to establish this type of MRM. Third, if they want to add more sub-federate simulators, there would be the difficulty of making a bespoke regulator for the additional sub-federate simulator. Last, location problems regarding the storage of the database can arise (such as military doctrine to share, or algorithms for interaction).

2.7.3 Regulation as Middleware

The Regulator as Middleware was introduced by Tan et al. (2001) as an alternative method instead of the module-based approach. The regulation as middleware is a system which puts the modifier, such as the aggregation/disaggregation approach, on the middleware. Thus, middleware controls the aggregation/disaggregation procedures among federates.

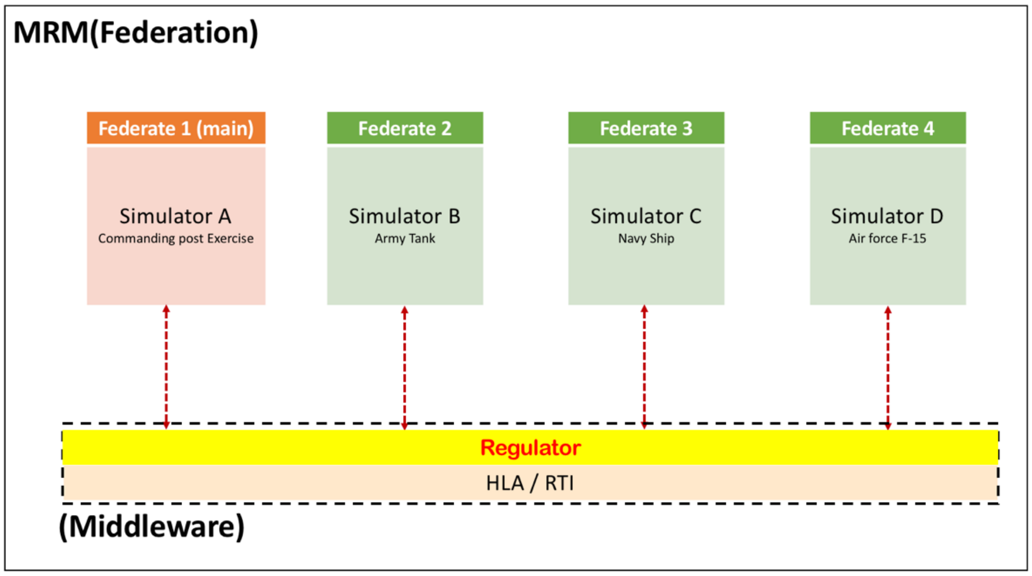


Figure 15 The middleware approach

The middleware approach attaches another layer of interface between federates and the RTI (as depicted in Figure 15). This middleware regulator layer handles all disaggregation and aggregation requests from federates. Therefore, the RTI is not involved in those interactions. This method reduces the workload of the RTI.

It is necessary to consider how to communicate with the federates when it comes to implementing the middleware method. This can be accomplished using two potential choices:

1. *A seamless link between the federates and the middleware.*
2. *An interface that can be designed with standard interfaces, such as the RTI applications programming interface (API) or "home-made" interface, with some modification.*

Since both choices are dependent implementations, there is a need to find a way to run in any environment without relying on RTI.

An example of regulation as middleware is shown in Figure 15. Assuming that 'federate 1' (main) is a low-resolution simulation (commanding post-exercise), and the 'federate 2' is a high-resolution simulation (army tank), the disaggregation request will cause the API to regulate not only the low-resolution data from 'federate 1' but also the high-resolution data from the 'federate 2'. The RTI will support operations of a federation execution with the standardized data through API.

By requiring a standardized, common interface between federates and the RTI, the middleware such as API has been developed and the HLA allows program developers to work independently. Therefore, the federate developers and RTI developers can improve the interfaces without regard to RTI implementation and explicit consideration of federate development. This approach can reduce RTI workload by screening unnecessary aggregation/disaggregation processes in RTI (Tan et al., 2001).

2.7.4 Regulator as Federate

The regulator as the federate approach is also one of the alternative methods instead of the module-based approach in Tan et al., (2001) paper. This approach suggests that building a regulator as a separate federate will control all required interactions of disaggregation. The regulator federate stores the database of information within itself. The concept of this approach is in Figure 16.

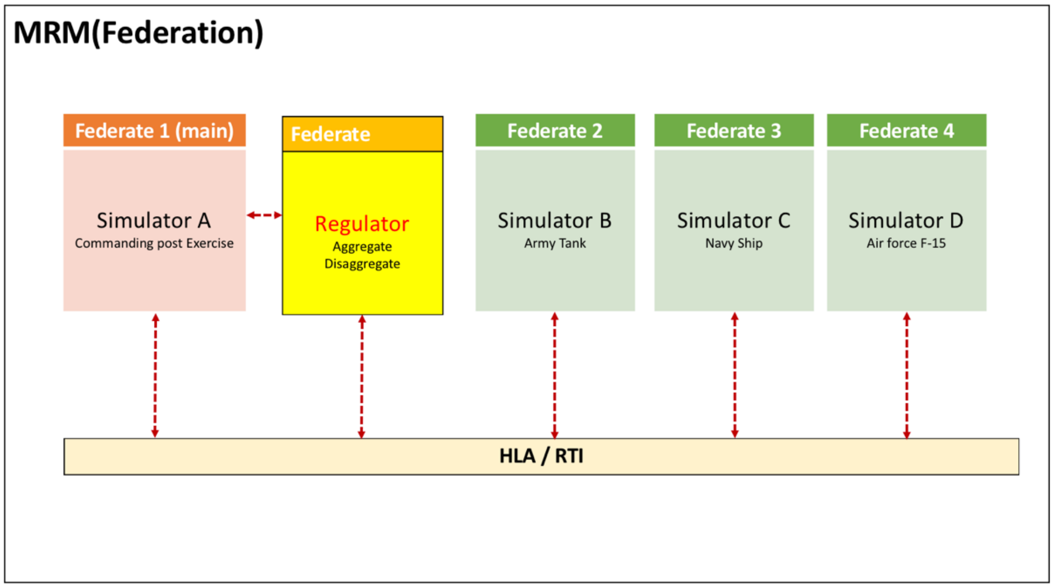


Figure 16 Regulator as federate

The regulator as federate is not only in charge of the interaction of aggregation/disaggregation, but it also decides which lower level federate associations will handle the interaction when disaggregation is needed. If it is not necessary, the federation will ignore the interaction (dynamic change), and the federation will perform normally. If necessary, federation accesses the database and extracts the information that should be sent as input parameters to the low-level federate that will process the disaggregation.

For example, Simulator A is designed for an exercise of commanding post, so this simulator is a low-resolution model and the main simulator in this MRM federation. Simulator B is a tank simulator for training soldiers, Simulator C is a navigation simulator for naval training, and Simulator D is an F-15 simulator for training a pilot. Suppose that an MRM will be built using the regulator as a federated method. As depicted in Figure 17, a federate which functions as a regulator to determine and execute the aggregation/disaggregation needs to be built. It is also one of the federates from the whole MRM federation perspective.

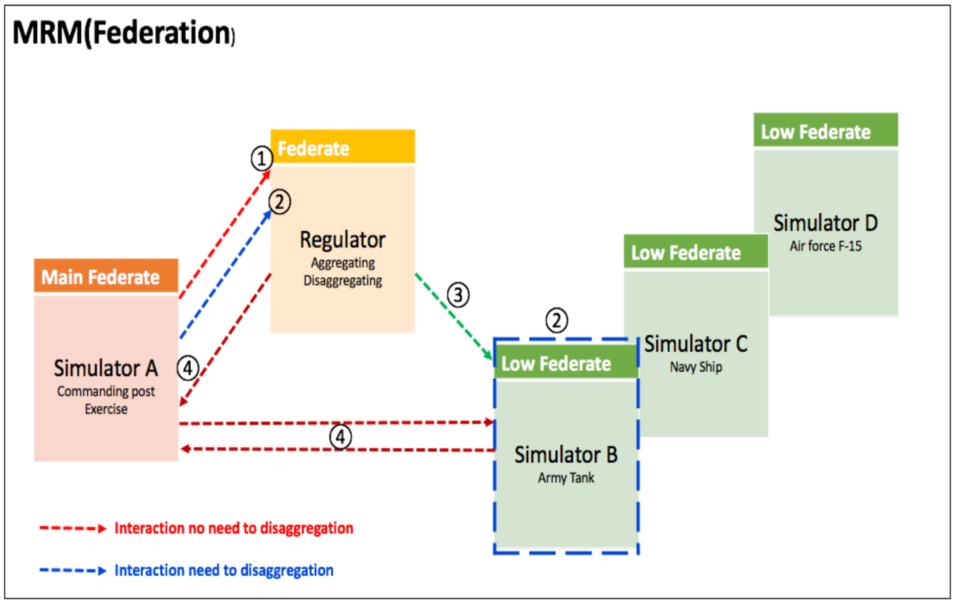


Figure 17 An example of MRM regulator as federate

Simulator A interacts with the Regulator, sending information about a certain situation to the regulator. The regulator first extracts the parameters of interaction and, depending on their logic, decides whether the disaggregation is necessary or not.

- (1) If disaggregation is not necessary, then the regulator (one of the federates) ignores the process (dynamic change), and the federation (the whole MRM) carries on as per normal, using only main Simulator A.
- (2) If disaggregation is needed, then the regulator determines which low-level federation is fit for this interaction and chooses one; for example, the regulator chooses Simulator B, which is the Army tank simulation.
- (3) Then, the regulator tries to access its database and extract the information necessary to be sent as input parameters to the high-resolution model (Simulator B/Army tank) that will handle the disaggregation.
- (4) The selected high-resolution model (low-level federate) will retrieve the parameters to start to create the individual entities. Then, the regulator shares the situation with main federate (Simulator A), and now disaggregation has taken place in Simulator B. The main federate then starts to send each aggregated units' attributes (which are important start values of their entities in simulation B) in an interaction to the high resolution model (low-level federate). Main federate (Simulator A/CP exercise) and selected low-level Federate (Simulator B/Army tank) will interact with each other. The high-resolution model (Low level federate) will keep updating the attributes in each time interval to the main federate.

Like this example, all simulations want to make an MRM federation that will be one of the federates as a part of the MRM system. Also, the regulator participated in this MRM federation as a member of the federate, but this regulator was designed by the MRM builder to help the process of aggregation/disaggregation. When it comes to designing this regulator, the builder should consider all characteristics of the main and sub federates (simulator).

2.7.5 Resolution Converter

A resolution converter is one of the methods to solve the problem of resolution mis-matching in communication between simulators for distributed simulation of multi-resolution models. The approaches act as a solution by transforming data formats and managing time synchronization in the multi-resolution modeling. Hong (2007) suggested developing the architecture of the multi-resolution converter to HLA/RTI to exchange data among distributed simulation models. However, the details of this concept will not be handled in this paper. It is a similar concept to the regulator which was discussed before, and can vary according to what kind of methods the system builder wants to add. Furthermore, it cannot be a cookie cutter method for every simulator or MRM system. As reiterated throughout this paper, there is no holy grail method for the regulator or converter.

2.7.6 Selective Viewing

Selective Viewing (SV) is another one of the traditional MRM approaches, in which the most detailed, highest resolution model, is running at all times, zooming and un-zooming capability presenting a display of various resolution are included. It is based on the principle that having more detail is more important than the entities performance.

Selective viewing depends on the highest resolution model that interacts with the low-resolution model after a short while in the simulation. It is similar to the construction of MVC (model-view-controller) in software design. MVC is a software architectural pattern for implementing user interfaces (Burbeck, 1992). It has three parts of a software application (see Figure 18): Model, View, and Controller.

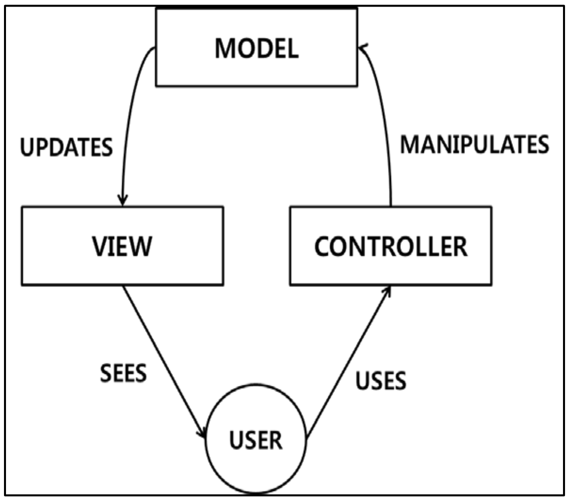


Figure 18 MVC approach (Retrieved from Burbeck, 1992)

In the selective viewing approach, commanders can see the battlefields from different perspectives of the model set at any time by simply choosing a resolution. (Sharma et al., 2014). That means a high and low-resolution model run at the same time. Thus, the user can change the view whenever the user wants. Many commanding post-exercise simulators have adapted this selective viewing to a certain degree.

The biggest advantage of this method is consistency because the variable of low-resolution is exactly the aggregations of high-resolution. According to Sharma et al. (2014), effective and accurate decision making can be conducted by a command agent based on very high consistency. On the other hand, there are some disadvantages. This method lacks flexibility. Usually, this selective viewing can be implemented in one simulation which is modeled or designed with the same rules from scratch. Therefore, this method cannot be used easily for building an MRM system using different types of simulations. Additionally, when modeling at higher resolutions computing resources are intensely used and often wasted (Davis & Hillestad, 1993).

2.7.7 Aggregation and Disaggregation

Multi-resolution combat models need the capacity to change a low-resolution level to a high-resolution level, or vice versa, respectively while simulating models. One of the various resolution change methods is aggregation-disaggregation.

2.7.7.1 Basic Aggregation and disaggregation

Aggregation objects mean a unit level consisting of various entity levels, while disaggregation objects represent an entity level which cannot be further decomposed (Rumbaugh, 1991). Disaggregation can be triggered by an event where aggregate units are separated into a lot of entities at a lower abstraction level, each representing a higher resolution component (Tan et al., 2001). Aggregation is also triggered in some situations where the federates need to control the aggregated unit, and then the entity is simulated independently. The following definitions and explanations are retrieved from Tolk, A., (2012):

1) Aggregation*. Aggregation is a reversed process of disaggregation means the method by which an entity level model is transferred into a unit level model. The aggregation has five steps below: a) If an aggregation trigger condition is satisfied, the interface module detects it and sends a message of aggregation to an entity level model, b) The interface module receives information about the aggregating entities from the entity level model, c) The entity level model* *information is translated to unit level model information by interface module which requests an instantiation to each unit level object, and d) The aggregating entities are aggregated according to each instantiation request by the unit level model which starts operating those entities, while the entity model stops operating the entities that were aggregated.*

***2) Disaggregation**. Disaggregation means the method by which a unit level model is transferred into an entity level model. The disaggregation has five steps below: a) If a disaggregation trigger condition is satisfied, the interface module detects it and sends a message of disaggregation to a unit level model, b) The interface module receives information about the disaggregating unit from the unit level model, c) The unit level model information is translated to entity-level model information by interface module which requests an instantiation to each entity level object, and d) The disaggregating units' entities are disaggregated according to each instantiation request by the entity level model which starts operating those entities, while the unit model stops operating the unit that was instantiated*.

2.7.7.2 Modified approaches

Aggregation and disaggregation are representative resolution change methods. This method includes some problems related to computing overload, and efficiency by chain disaggregation phenomenon. Thus, some modified approaches were presented, such as pseudo-disaggregation/aggregation, partial disaggregation, and subset disaggregation. The details are described in the following:

3) Partial Disaggregation. *Partial disaggregation is a useful method for resolution change by not fully transferring the representation of a unit to the entity level. Entities which are described in the partial disaggregation have not complete functions at the entity level. The important disaggregation shortcoming can be resolved regarding efficient data management by this flexible method (Natrajan et al., 1997). The partial disaggregation has been applied to the entity level model using ModSAF (Schricker et al., 1998).*

4) Pseudo-disaggregation. *Pseudo-disaggregation allows almost none of the representation of the unit to be transitioned to the entity level as a special case of partial disaggregation. Namely, active representation of the unit level model is not changed to the entities in the pseudo-disaggregation operation. However, a process such as disaggregation is only applied to generating the entity level information which is used at the unit level composed of entities over time.*

5) Pseudo-aggregation. *Pseudo-aggregation is a reversed concept when being compared to pseudo-disaggregation as described earlier. Almost none of the representation of the entities is transitioned to the unit level. Namely, active representation of the entity level model is not changed to the unit in pseudo-aggregation operation. But, a process such as aggregation is only applied to* *generating the unit level information for the unit combined by the entities over time.*

6) Subset Disaggregation. *In subset disaggregation operation, all entities of a unit are not applied to disaggregation. A subset of entities, which composes a unit, is instantiated in the entity level model. Therefore, the unit level model can have some representation of the unit for the entities which was not disaggregated. Partial disaggregation separates some of the capabilities for entities, whereas subset disaggregation separates some of the entities for a unit.*

2.7.8 Multi-Resolution Entity (MRE)

There is a need for the Multi-Resolution Entity (MRE) concept for interacting at multiple levels of resolution simultaneously, referred to as MRM from here on. This concept is one of the methods to maintain consistency of constituents during a series of aggregations and disaggregation. It is how to keep records for entities or units while avoiding inconsistency in MRM. According to Reynolds Paul. F.(1997):

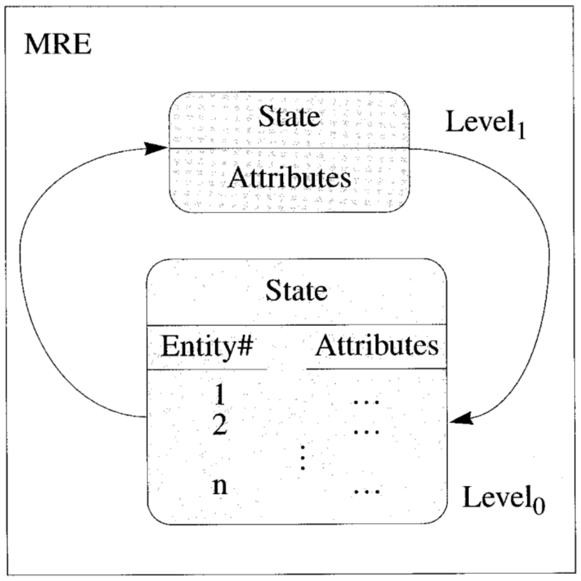


Figure 19 Design of an MRE (Retrieved from Reynolds Paul. F.,1997)

"*an MRM interacts with consistent properties across matched properties at different levels of resolution. Each MRE maintains state information at all desired levels of resolution or provides the requested level of information immediately…*"

Figure 19 describes a notion of MRE design that can interact with two levels of resolution. Reynolds holds the view that the problems of aggregation and dis-aggregation can be solved by holding all information in the MRE. The simulation of the MRE reflects the impact of incoming interactions that any level user wants. Each MRE is responsible for maintaining logical consistency at any resolution level; the effect of the incoming interaction must be consistently reflected in the attributes of all levels of the MRE (Reynolds Paul F., 1997).

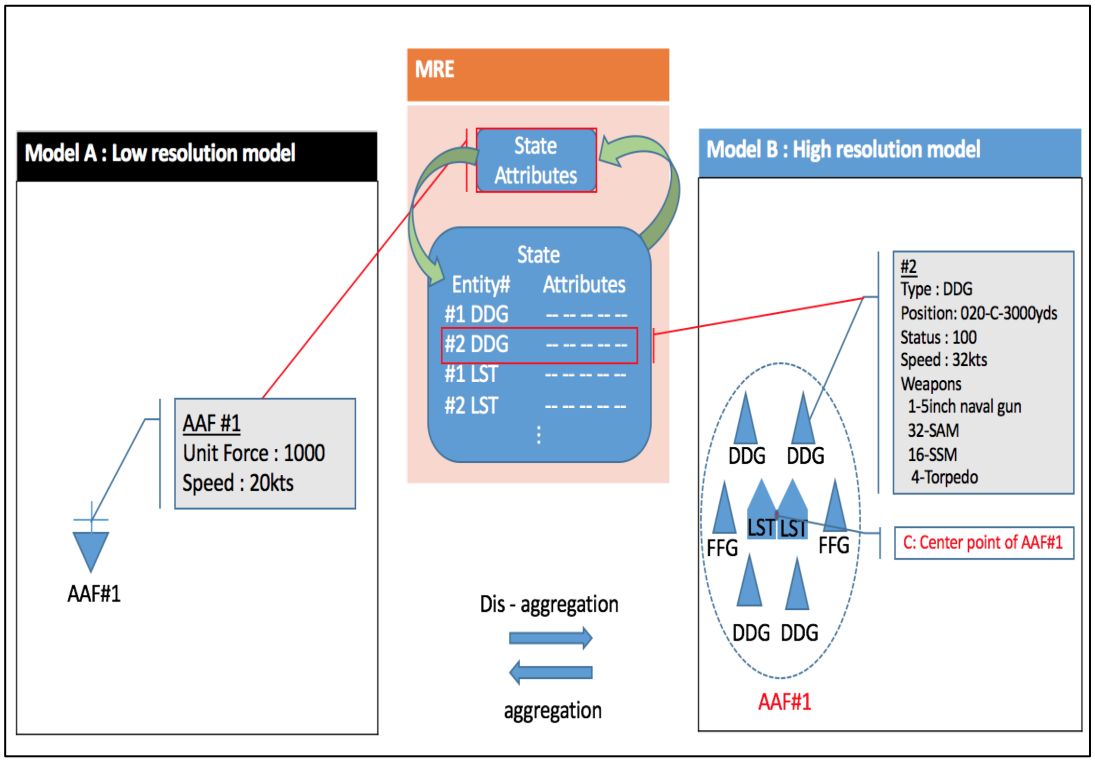


Figure 20 An example of MRE concept

Take a look at the simple example of the MRE concept. As shown in Figure 20, all attribute information will be updated in the MRE. The MRE is continuously updating its information in a single time-step and interacts with every aggregate unit (in Model A) and disaggregate entity (in Model B).

Updating MRE's information will be done by a Consistency Enforcer. MRE Consistency Enforcer is designed to have a function calculating every interaction with each level. The effects of interactions coming from a single time-step are distributed across the two levels and map the effects of interactions at one level to the properties at a different level, etc. A detailed design of the Consistency Enforcer depends on the particular models selected at various levels and is beyond the scope of this paper. The MRE method has been proposed as an object that can reflect the effects of simultaneous interactions at multiple levels in a consistent manner like in Figure 21.

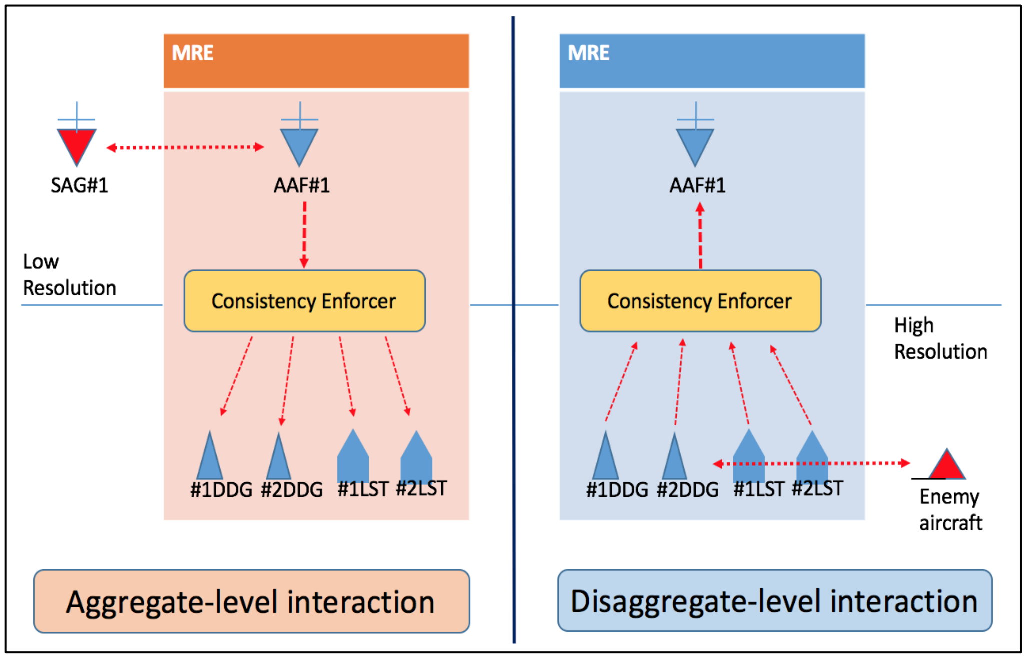


Figure 21 Consistency Enforcer in MRE

The logic of Consistency Enforcer would depend on simulations. The crux of MRE application is laid on how reasonable the consistency enforcer logic is and consistently reflecting in its attributes and the effects of interaction at all levels promptly on demand.

2.7.9 Hybrid (Disaggregation / MRE)

Aggregation-disaggregation and Multi-Resolution Entity (MRE) are methods of MRM as explained above. The aggregation-disaggregation was the commonly used approach and was considered to represent the nature of MRM best. However, this approach may lead to inconsistencies between the various levels of resolution. On the other hand, the MRE approach always maintains the attributes of an entity consistently at all levels of resolution. However, MRE can be consistent while sacrificing more resources and software costs.

The hybrid method is designed to reduce or eliminate transition overheads, in other words, chain disaggregation like in Figure 22 (left). In the perspective of interacting at multiple levels of resolution simultaneously as depicted in figure 22 (right), it is a similar concept with MRE, but they have diverse ways to maintain consistency.

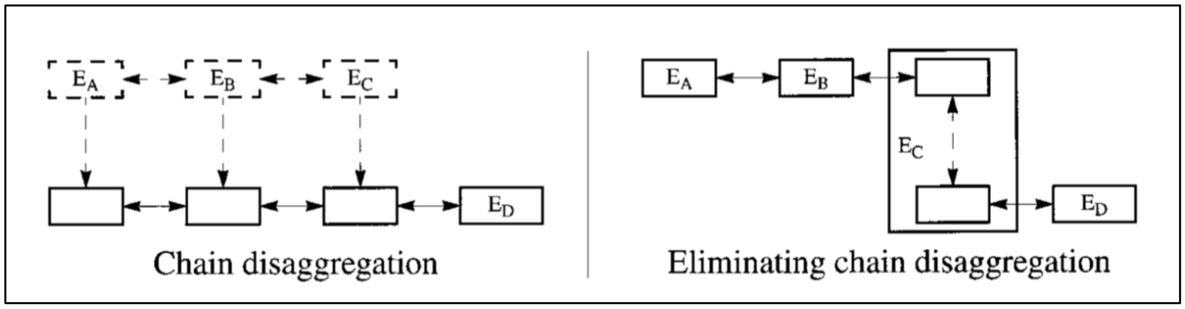


Figure 22 Reducing transition overheads (Retrieved from Reynolds P. F., 1997)

According to Reynolds Paul.F.(1997), a hybrid approach maintains a consistent attributes *core* across the resolution level. The core is a subset of the entire set of attributes, consisting solely of attributes that are considered essential. As needed, the values of additional attributes are generated from values at the core level.

Hybrid method's core is has a similar function to the MRE Consistency Enforcer. MRE keeps all attributes at each level and deals with incoming interaction results at any level promptly by the Consistency Enforcer. However, the Hybrid method keeps the core index of some attributes as shown in Figure 23. This core set may keep updating on every interaction to reflect a state of the MRE that is consistent at multiple levels of resolution (Reynolds Paul F., 1997).

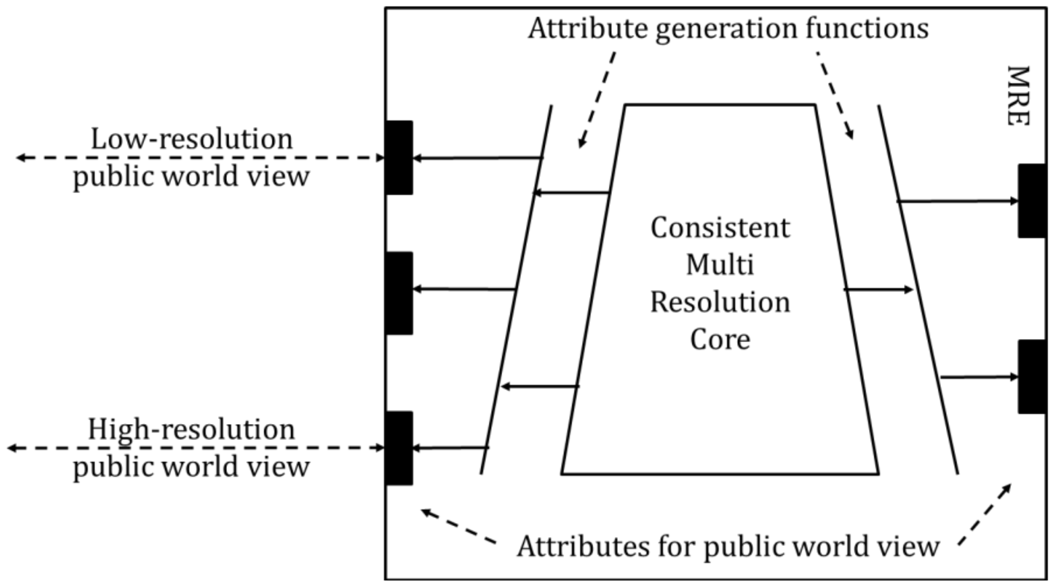

Attribute generation functions
MRE
Low-resolution public world view
Consistent Multi Resolution Core
High-resolution public world view
Attributes for public world view

Figure 23 Core attributes (Reynolds P.F., 1997)

The hybrid method categorizes all attributes into the core set attributes and others. Reynolds Paul F.(1997) explained about the core. ‘*Because the core is a subset of all the properties at all levels, you need to develop a criterion that identifies the properties that should be at the core.’* This means the hybrid method does not have all the attributes in their core. Reynold Paul F.(1997) said four criteria had been identified which should be considered when identifying core variables such as reversibility, reduced time effectiveness, cost ratios, and frequency of access.

Conceptually, it seems to be an economic method for MRE when it comes to handling less information. However, realistically, it would be hard to find that kind of *core* which perfectly matches at all level attributes. Figure 24 shows how difficult it is to build those cores in comparison to the MRE.

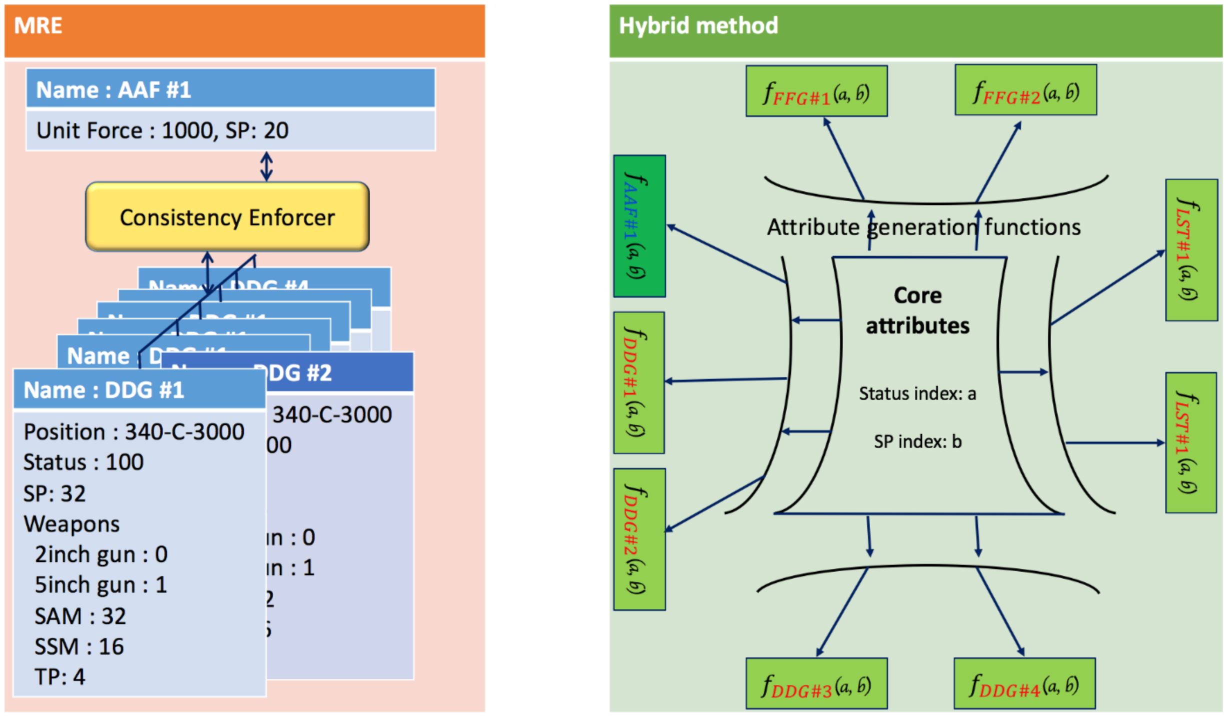


Figure 24 Comparing methods between MRE and Hybrid

As depicted in Figure 24, the MRE keeps all attributes and information at all levels in the MRM, and the Consistency Enforcer is in charge of changing attributes between LRE and HREs when it comes to incoming interactions. On the other hand, Hybrid picks core attributes according to some criteria that can be updated in all interactions to reflect the state of the matching MRE at different levels of analysis. These core attributes will change with every incoming interaction, and those changes will affect all related attributes of entities. The hybrid method keeps less information, but it is hard to choose core attributes. Furthermore, it will take much more effort to develop the attribute generation functions, such as $f_{AAF\#1}, f_{DDG\#1}, f_{DDG\#2}, f_{DDG\#3}, f_{DDG\#4}, f_{FFG\#1}, f_{FFG\#2}, f_{LST\#1}, f_{LST\#2}$, which allow attributes to maintain consistency at every level of resolution at any time.

2.7.10 Agent-based

Agent-based simulation has gained more popularity recently, especially in areas where behavior is important, because of its powerful capability of capturing behavior in detail and imitating the system interactions and dynamics (Brailsford, 2014). The increasing numbers of conference proceedings and journal articles that call for agent-based models are evidence of its popularity and growth (Macal C. & North, 2009).

The agent-based simulation would provide better results under certain scenarios, because the legacy modeling tools may not be enough to capture the complexity of the real world. Although there is no universal agreement on the exact definition of the agent, the fundamental characteristic of an agent is the ability to make independent decisions that are working itself.

The agents have a behavior and can make decisions. They also interact with the objects and other agents. Furthermore, agents can form groups and act individually. In this perspective, we can use an agent-oriented approach to more closely match the characteristics required by the military and operational view of problems.

Agents consist of many sub-compartments. For example, an aircraft is composed of engines with their respective modules and components. Components are implemented as agents, like aircraft and engines, and we can implement different levels of details with an agent-based system. Then, we can select the level of detail (i.e., aggregation and disaggregation); selective viewing is straightforward using agents.

An agent which is built by the same rule may be able to employ multiple models and agents and may execute multiple models jointly. In a sense, the agent-based simulation paradigm can be used effectively in MRM.

2.8 Comparison

The comparison among the current MRM approaches, except the module-based approach, was accomplished in the research of comprehensive survey & tutorial on Multi-Resolution Combat Modeling by Gene Lee & Luis Rabelo (2017).

They analyzed and provided the table in their paper, which has a summary/comparison of the different MRM schemes available. The comparison was made considering the Multi-Representation Consistency, Multi-representation Interactions, Cost Effectiveness, potential utilization of COTS for a particular implementation, and scalability.

In their conclusion, MRE and Agent-based approaches are in general the best approaches. The reasons are MREs reduce simulation and consistency costs. MRE can be implemented using COTS, and has been utilized in HLA environments. The scalability and cost are also crucial factors to be considered for MRM. Minimum modifications can achieve MRE. Therefore, MRE is one of the best approaches for MRM available today (Gene Lee & Luis Rabelo, 2017).

Furthermore, if an individual considered building a new simulator, then the agent-based technologies are also a good option to implement plug and play functionality. The agent-based simulation technologies can provide a suitable environment for MRM. However, this has a limitation to building MRM by using legacy simulations.

Even though they conclude that MRE and Agent-based technology can be a good option to build an MRM, that does not mean that it is the panacea for building all MRM federation. Gene Lee & Luis Rabelo clarify that:

> *"determining whether a multi-models are satisfied is a form of the Turing test because ultimately only the end-users can determine whether the multi-model*

meets their requirements. That means: besides the general requirements of our proposed comparison scheme, the particular requirements and technological knowledge of the end-user are sometimes more relevant in determining the MRM approach to be utilized."

This would mean that the approach must be considered depending on what kind of simulations the user wants to use in the MRM federation.

CHAPTER 3: METHODOLOGY

3.1 Introduction

To briefly describe the MRM concept, most MRM is a federation of more than two different simulations for the user requirements. If the simulations that make up the MRM are made in different computer languages and different communication methods, then the modification cost will be increase. If building some MRM system required a prohibitive cost when they connect two or more simulations, it could mean a fail. To achieve interoperability among simulators, they need Standard Simulation Architectures (SSA) that have been developed in order to achieve interoperability among independently developed simulations (Jung Wonil, 2017). DIS, HLA, and TENA are major SSAs. However, only the specific research of HLA that is related to COTS will be discussed.

3.2 High-Level Architecture (HLA)

The Defense Modeling and Simulation Office (DMSO) addressed the ongoing need for interoperability between new and existing simulations in the U.S. The U.S. Department of Defense (DoD) needed some standards for their military simulations. Without a standard, they could not connect their simulation easily, so they started to research about High-Level Architecture.

3.2.1 Definition of HLA

HLA is infrastructure, like a highway which already has been constructed by the government for enhancing transportation. According to Reid (2000), "*The High-Level Architecture (HLA) is a current U.S. Department of Defense and an industry (IEEE-1516)* *standard architecture for modeling and simulations. It provides a framework and set of functional rules and a common interface for integrating separate simulators and heterogeneous simulations into a large federation* (Reid, 2000)." HLA wants to generalize and build related efforts, such as the Distributed Interactive Simulation (DIS) world and aggregation-level simulation protocols. The basic HLA definition was approved in 1996 as the standard technology architecture for all US Department of Defense Simulations, and HLA is continues to be updated today. Therefore, most military-related simulations built their simulation systems following HLA. Below, the explanation of HLA references are from Dahmann et al. (1997), unless otherwise specified.

3.2.2 Motivation of HLA

The HLA is based on the premise that all simulations and users cannot satisfy every certain requirement. That is the same premise as MRM. The purpose of the HLA is to provide a standard that improves the performance by reducing the cost and development time of the simulation system and facilitating the recyclability and interoperability of the component simulator (Reid, 2000). The HLA is designed for a wide range of resolutions at varying levels of resolution in various levels of orientation simulation applications, including training, analysis, and engineering capabilities (Dahmann et al., 1997).

3.2.3 History of HLA

The U.S. DoD developed the HLA based on processes related to government, academia, and industry. The HLA is widely used in various fields and nations. For instance, NASA also adopted this architecting rule for their simulation programs. South Korea, which is one of the closest allies of the US, have become similar as time goes by. Korea’s DoD also adopted a resolution of HLA as a standard and it is one of the requirements when they choose a military simulation.

3.2.4 Characteristics of HLA

The HLA does not specify a specific implementation and does not require the use of a specific set of software or programming languages. As technology develops over time, new and diverse implementation within the framework of HLA will be possible. However, the HLA requires that all federates and simulators integrate specific functions, and allow the objects of the simulation to interact with objects of the other simulations through the exchange of service supported by the RunTime Infrastructure (RTI).

A functional view of an HLA federation consists of three parts. The first principle component is the simulations themselves. The second functional component is the RTI. The third functional component is the runtime interface that provides a standard method for federated interfaces and responds to requests from the RTIs.

3.2.5 RunTime Infrastructure (RTI)

The RTI is software for the Federation of the distributed operating system. The RTI provides a set of services that support the simulations when performing this federate-federate interaction and federation management support functions. All interactions among the federate have to flow through the RTI.

<u>3.3 Building a meaningful MRM federation</u>

As was shown in chapter 2, MRM is defined by Davis and Bigelow(1998) as:

(1) building a single model with alternative user modes involving different levels of resolution for the same phenomena; or

(2) building an integrated family of two or more mutually consistent models of the same phenomena at different levels of resolution; or

(3) both.

According to this definition, most recent simulators can be assumed as MRM because they adapted selective viewing, or a certain amount of resolution change itself. Furthermore, the concept of resolution has many facets. Thus it cannot be directly compared or distinguished which simulator is a low-resolution or high-resolution. Therefore, if two different simulators are connected and constructed with an environment allowing them to work together and share their information, then it can be considered an MRM system.

A meaningful connection of MRM for military purposes, however, is not a single selective viewing simulator or a simple connection with different resolution simulators. When U.S DoD developed the High-Level Architecture (HLA), they sometimes wanted to establish interoperability between different types of simulations at different locations to simulate interactive activities. Therefore, a meaningful MRM for the military field is a connection that clearly facilitates the purpose of achieving the usage of the federation. For example, it is ideal to combine commanding-post exercise simulations and individual training simulations in one federation to provide realistic training for soldiers of other classes in the same federation movement.

3.4 Steps to develop an MRM

The current MRM approaches which were discussed in the previous chapter, are solutions which have a different focus on the MRM issues/problems. Some approaches are solutions for simulation design which can be MRM, such as IHVR, Agent-based and so on. Though there are some specific problem resolutions like MRE or Hybrid. These approaches are methods which need to be reviewed at different steps during development of an MRM. A meaningful MRM federation is to connect simulations (or federates) with different levels of resolution (High or Low) to meet the needs of the user. Therefore, the following three steps are proposed to build an MRM system and categorize the current MRM approaches into each step.

3.4.1 Step 1: Decide to develop a new simulation or reuse

Most existing simulations have inherently limited interoperability because they were not designed to work together from scratch. If one wants to build a whole simulation from scratch to make it work in a federation, then it can be built seamlessly and is an ideal MRM federation in some aspect. As shown in Figure 25, Integrated Hierarchical Variable Resolution (IHVR), Selective Viewing, and Agent-based approaches can be adapted if one wishes to build a new simulator or simulator for MRM.

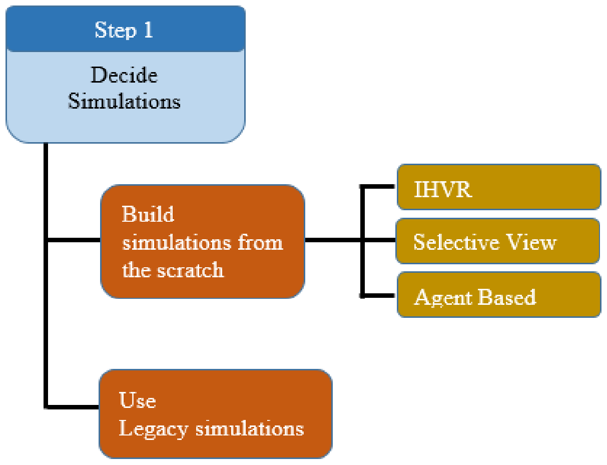


Figure 25 Step 1: Decide simulations

The IHVR approach is difficult to apply to the already made-simulation because the IHVR approach tries to achieve a seamless link in low and high-resolution entities by building a new relationship between low and high-resolution entities attribution variable. It is hard to find an adequate relationship equation in the tree of the variable (Figure 13). Therefore, if you build a new low and high-resolution simulator from scratch, then you can consider its tree of the variable to build a seamless MRM federation.

Agent-based approach is also very powerful when one builds every sub-simulator with the same rule for fitting them together. For example, suppose one wants to build an aircraft simulation as an MRM federation according to the agent-based approach; one would develop multiple modules consisting of the aircraft-like engine module, flight control module, and so on. These modules can also be broken down into sub-compartments like fuel tank simulators, ejection simulators, and so on. Those small compartment sub-simulators are working for the main module simulator, and module simulators make the whole aircraft simulator. Therefore, the Agent-based approach is also more suitable for building the MRM from scratch strategy.

Selective viewing is an MRM approach to change resolutions through viewpoint changing. It is also a suitable approach to build MRM from scratch because it is hard to modify into the existing simulator to have selective viewing function. It requires fundamental re-engineering to the already built simulators. Therefore, it is not a good method from an economic perspective.

Building an MRM system from scratch can create a seamless MRM federation due to it being made from scratch. It can be designed to avoid data inconsistency or time synchronization issues between low and high-resolution models. On the other hand, it can cause tremendous cost problems. If a user wants to build the MRM federation system by using the legacy simulators, then the builder must consider which simulators will participate in the federation to meet the user's requirements. This paper focuses on using legacy simulators. Therefore, two commercial simulators, MASA Sword and VR-forces, will participate in the MRM federation.

3.4.2 Step 2: Federation Architecture

If the builder decides to use legacy simulators for an MRM system, then they might face the problems that were discussed in chapter 2, the current issues of MRM. SSAs can solve the basic problems; protocol to communication, definitions of the formats of the messages to be exchanged, the data formats, logics of action, and sequences to be performed. However, each federate can vary in data, especially attributes to their unit or entities. As was discussed in Chapter 2, a high-resolution model needs more data to operate their units or entities than the low-resolution model does. These kinds of data gaps are different depending on what kind of simulators participate in the federation. Therefore, the chance of finding and building one solution for relieving data gaps in general, is highly unlikely to happen.

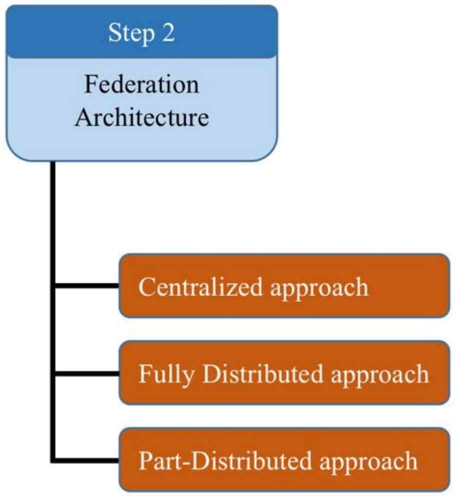


Figure 26 Step 2: Federation Architecture

When the developer wants to build a federation among the different levels of simulations, they have to consider the federation architecture at first. According to Martin A. & Pierre S.(2001), there are three federation architecture approaches; centralized approach, fully-distributed approach, and part-distributed approach (Figure 26).

3.4.2.1 Centralized architecture

Martin A. & Pierre S.(2001) said that a single federate ensures the simulation of both low and high-resolution entities in the centralized approach. The method is to absorb a number of high-resolution simulations in a low-resolution simulation and make them work as part of a low-resolution simulator engine as depicted in Figure 27. This approach needs re-engineering of all simulations which participate in the federation. Suppose that simulation A is a tank company level simulator engine, and simulation B is individual tank level simulator engine. Then, the centralized approach requires that simulation A is re-engineered so that it is made out of several simulation Bs. This approach does not need the HLA/RTI service to make a federation, but it costs the re-engineering of both simulator engines.

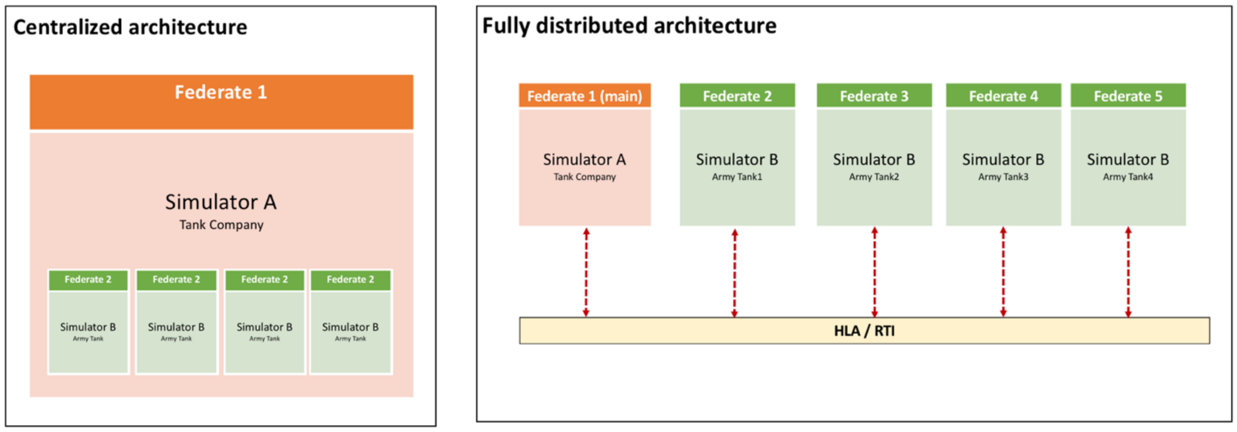


Figure 27 Centralized architecture vs. Fully-distributed architecture

3.4.2.2 Fully distributed architecture

Conversely, fully distributed architecture is designed for each aggregate or disaggregate entity to interact with its own federates. In this architecture, each federate may run on different hosts of the network. Figure 27 depicts the fully distributed architecture. For example, when the tank company consists of four individual tanks disaggregated into tank entities, each entity is designated to interact with a specific tank simulator B. This architecture still needs re-engineering of simulator A because separated tanks need to be assigned to a simulator to interact with. The number of messages exchanged between the federates and the difficulty of extracting global state at any given time is a major drawback of this architecture. However, it is easy to expand the federation.

3.4.2.3 Part distributed architecture

Part-distributed architecture can provide a compromise between the amount of messages exchanged and the distribution. The architecture is simple; each federate takes charge of a specific resolution. For instance, in Figure 28, federate 1 takes charge of everything that happens in low-resolution, and federate 2 controls everything in high resolution. It is a sort of role sharing. Furthermore, it does not require re-engineering of each federate; even if this architecture requires modification, such as aggregation and disaggregation, it can be achieved through minimum modifications or a separated regulator.

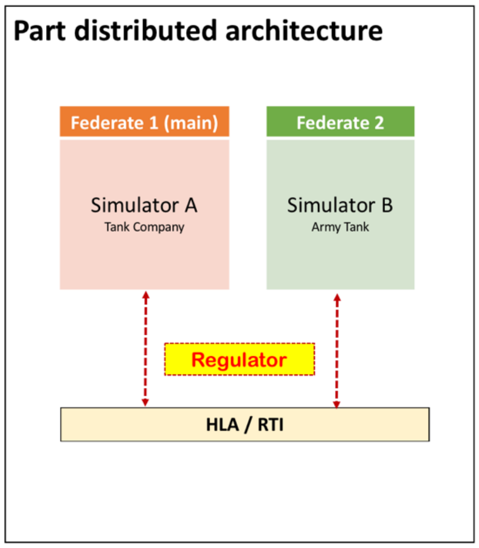


Figure 28 Part distributed architecture

3.4.3 Step 3: Design Regulator

The regulator controls the ownership and changes the information in order for the sub-federates to fit each other. As shown in Figure 29, the regulator can vary based on the function of the regulator, and the where the builder placed it. In some papers, it is called a resolution converter. It works like an interpreter and coordinator in the federation.

Engineers can build without a regulator, but it will result in a less powerful MRM. An MRM without a regulator has major data inconsistency problems, or they cannot share any data among federates because each federate could have different attributes for running their simulation engines. Only a certain amount of information can be shared if the federates build same Standard Simulation Architecture (SSA) like an HLA. However, there will be a limitation in the MRM federation without a regulator.

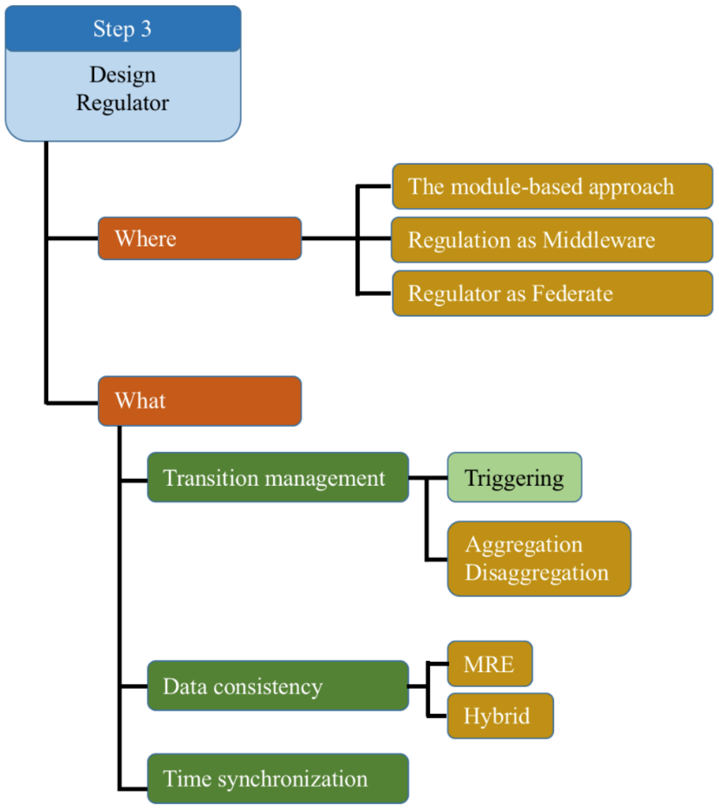


Figure 29 Step 3: Design Regulator

The designs of regulators is the crux of building a seamless MRM federation system. Standard Simulation Architecture (SSA) provides a rough linkage among federates, but the higher resolution gap among federates causes more problems in the federation. Therefore, adequate modifications are needed. Re-engineering of simulations would be expensive, so the regulator can be an alternative method to relieve MRM problems. There are two major considerations; what to put in the regulator and where to put the regulator.

3.4.3.1 What to put in a regulator

Transition management is a way to manage the level gap of information among federates. When MRM changes a low-resolution level to a high-resolution level, MRM will change one side's information to make it suitable for the other side. For example, in Figure 30, there is a tank squadron unit which consists of four individual tanks in a low-resolution model (MASA Sword). A high-resolution model (SIMBox Simigon) is a simulation which can only represent individual tanks; therefore the low-resolution model information needs to be changed to be compatible information for the high-resolution model or vice versa.

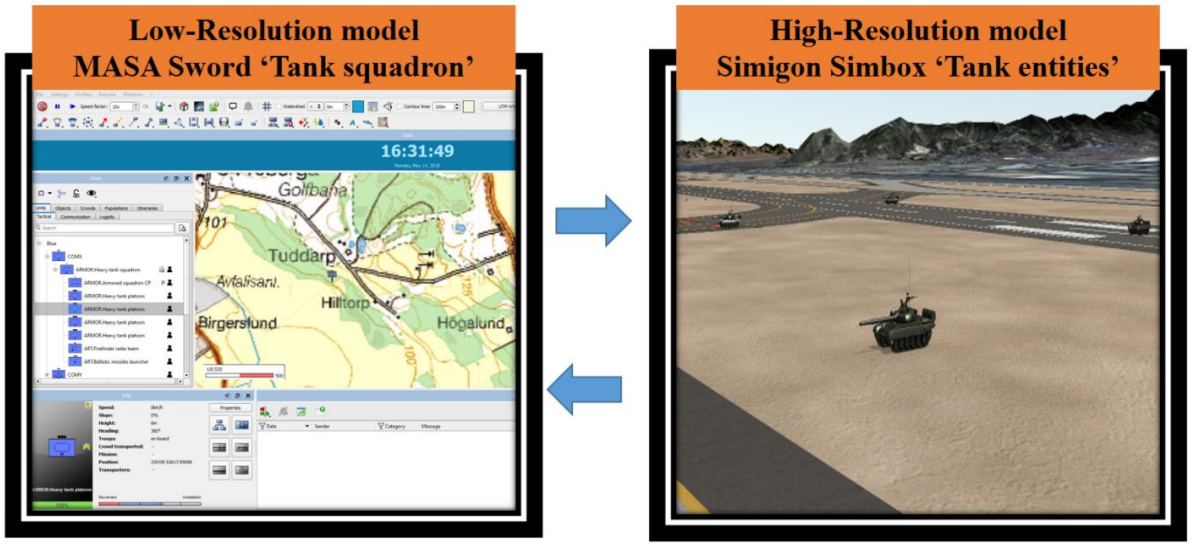


Figure 30 Transition management; Aggregation & Disaggregation

As you can see in Figure 30, low-resolution model, MASA Sword, has one symbol which represents the tank squadron in their map. If the MASA Sword wants to send this tank squadron information to Simigon SIMBox, MASA Sword would have to create individual tank information such as position, armor, status and so forth for the Simigon SIMBox. It is an example of disaggregation process; the detailed process will depend on how to modify the transition management. In general, the low-resolution model generates specific information to fit into the high-resolution model. For instance, an individual tank's position will be generated randomly or through doctrines of the military. The detailed logic of generating in this disaggregation process is up to the builder.

A 'Trigger' in MRM means a specific condition for changing the resolution. There is not a particular definition of 'trigger' for MRM. However, we can extract the general definition of trigger through several papers about MRM;

> *"For example, in an indirect fire situation, two entities could engage in combat without direct interaction (as in long-range artillery fire). Due to the indirect nature of the engagement, disaggregation is not triggered." (Reynolds, 1997)*

> *"Since such an entity can simultaneously operate at multiple levels of resolution, some sub-models can be active at a time. Shifts in the phase of a problem trigger model updating by changing the active set of entities."(Yilmaz, 2007)*

> *"In our doctrine-based multi-resolution converter, a resolution change trigger is dynamically activated when an object performs different resolution actions or when it enters an area of aggregation or disaggregation. (Paul, 2017)*

> *"Disaggregation is also triggered by some event, causing control of the aggregated unit to be transferred to the models (federates) representing the individual entities, and each entity is then simulated individually"(Tan, G., Ng, W. N., & Moradi, F., 2001)*

"Disaggregation requests are <u>*triggered*</u> *within the main federate by the contact zones, which are the radar detection zones of the units." (Tan et al., 2001)*

In general, trigger means a condition for changing the resolution or simulator in MRM. There are several types of 'Triggers' when it comes to building an MRM system. Such as fixed geographical area, spheres of influence, manual triggering, event-based triggering, specific period, and so forth. The type of trigger can be whatever the developer designed, to meet the user's requirements. The trigger is an essential factor when it comes to developing an MRM system. The moment in which a resolution change is needed, relates to the reason why the users want to build an MRM system. The trigger also can prevent computing traffic overload. The trigger limits disaggregation and aggregation action locally or temporally, thus avoiding unnecessary increases in computer overloads. If you look closely at these triggers, the trigger can be limited to three characteristics; Geographical, Time and Commander.

- Geographical-based trigger (spatial size)
- Time-based trigger
- Commander-based trigger

For instance, 'proximity to hostile units or entities trigger' can be described from the viewpoint of a limited geographic area, specific unit's sensing area, or engaging area. The commander-based trigger also could be an option for triggering, because the resolution should change when the user (commander) wants it to change to meet the user's requirements. Therefore, every trigger can converge into those three properties.

Data inconsistency and time gaps cause problems about how simulations are reflected realistically. Data inconsistency and what kind of approaches are currently suggested (MRE, hybrid) were already discussed in the previous chapter 2. Time synchronization is out of this research boundary. Thus, the details of these approaches will not be mentioned in this chapter.

3.4.3.2 Where to put the regulator

If the developer decided on what kind of MRM approaches they want to include in the regulator, the next thing to consider is where to put the regulator. There are three different options to attach the regulator in an MRM federation system.

- The module-based approach
- Regulation as Middleware approach
- Regulator as Federate approach

As shown in Figure 31, each approach has a different position for the attaching regulator. The details of each approach were already discussed in Chapter 2. Each approach has pros and cons; there is no way to determine which approach is better than the other because the best method depends on the user requirements and federate (simulator) characteristics.

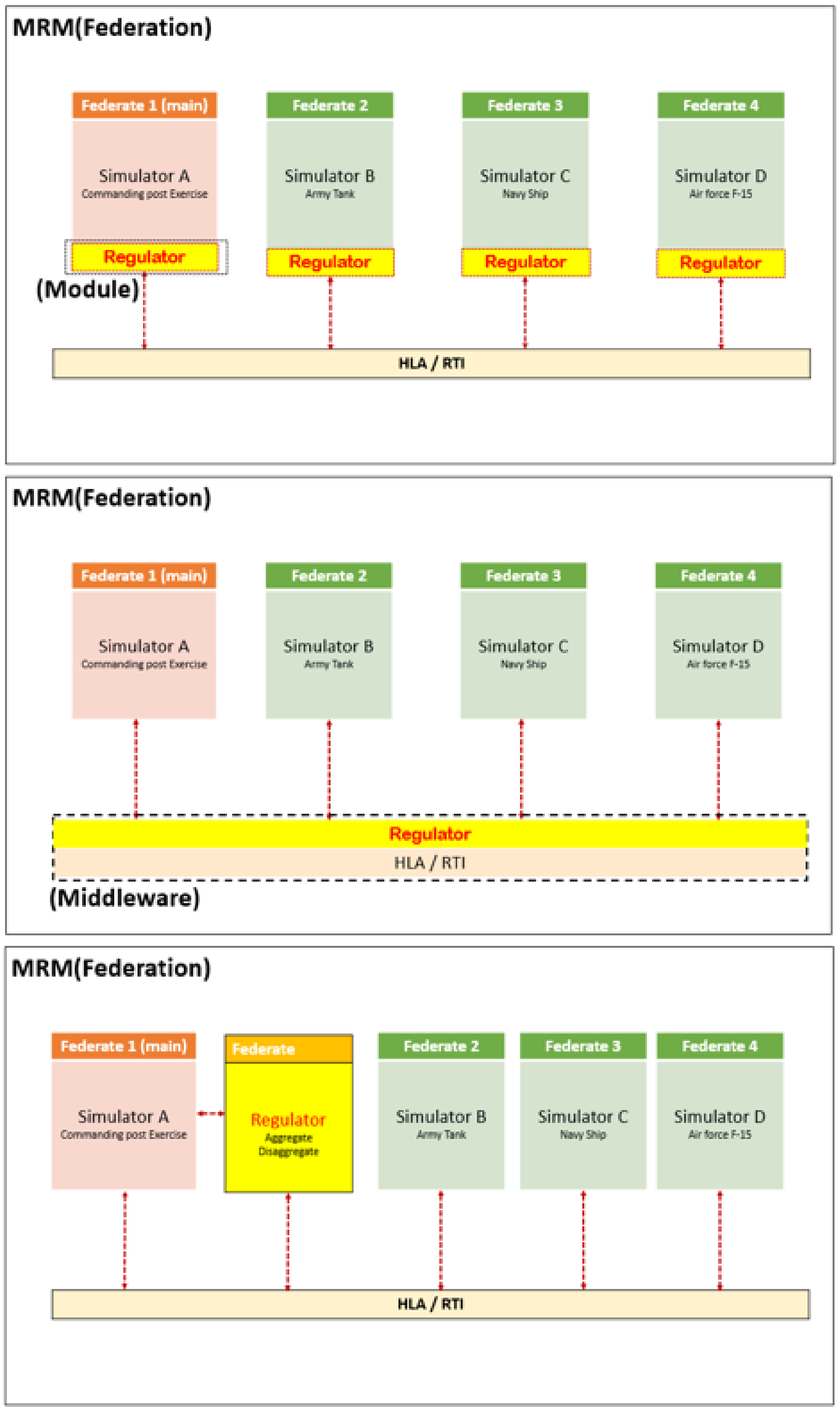


Figure 31 Where to put the regulator

CHAPTER 4: EXPERIMENT ENVIRONMENT

4.1 Commercial Off-The-Shelf (COTS)

COTS, MOTS, GOTS, and NOTS are abbreviations which depict pre-packaged software or hardware purchase. It depends on who is the targeted user.

A COTS product means commercial off-the-shelf. COTS products are designed to be easy to install and interoperate with existing system components. Almost all software purchased by the average computer user is a COTS product. The advantages of COTS software are that they can mass-produced, are easily accessible, and are relatively low cost.

MOTS means modified, modifiable, or military off-the-shelf. The meaning depends on the context, but basically, MOTS describes a modified software from a COTS product to achieve a specific purpose. A GOTS product means government off-the-shelf, which is typically developed by the technical staff of the government agency. Therefore, it is hard to access that software. NOTS is the abbreviation of NATO off-the-shelf. It is a software which is developed to meet specific requirements of NATO.

These scenarios were developed to show a basic concept of MRM. For this purpose, sword from MASA, VR-Forces from VT MÄK Technologies, and SIMBox from Simigon were all initially used for this research paper. They are all COTS simulators. The simulations were connected through HLA/RTI because DIS and HLA are international standards based on a commercial off-the-shelf (COTS) implementation strategies (Park TaeWoong, 2015).

4.2 Hardware

Four computers will be used for the experiment of two different MRM federations. In order to minimize the collision caused by using heterogeneous RTIs, the arrangement of the computers in the Simulation Interoperability Laboratory (SIL) of the University of Central Florida (UCF) is shown in Figure 32. Table 3, 4, 5 and 6 describe the hardware operation environment.

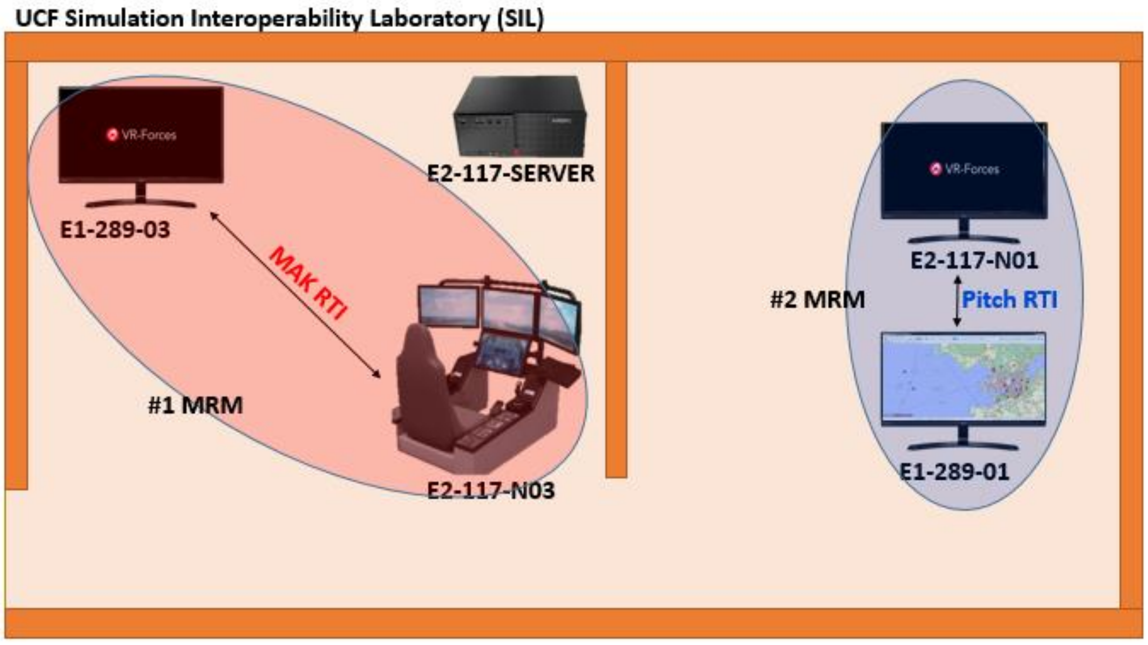


Figure 32 Arrangement of computers in UCF SIL

Table 3 Operation environment for #1 MRM (Low-Resolution simulation)

Computer	Equipment	Description
E1-289-03	Desktop Computer	- CPU: Intel® Core i7-4770K Processor 3.5Ghz - HDD/RAM: 1TB/16GB - VGB: NVIDIA GeForce GTX 770 (2GB) - Monitor: 23inch LCD(1920x1080)
	O/S	- Window 7
	Operation	- VR-Forces from VT-MAK - MAK RTI

	Complier	- Microsoft Visual Studio 2010
	Purpose	-#1 MRM (Low-resolution)

Table 4 Operation environment for #1 MRM (High-Resolution simulation)

Computer	Equipment	Description
E2-117-N03	Desktop Computer	- CPU: Intel® Core i7-4770K Processor 3.5Ghz - HDD/RAM: 1TB/16GB - VGB: NVIDIA GeForce GTX 770 (2GB) - Monitor: 4x23inch LCD(1920x1080)
	O/S	- Window 7
	Operation	- SIMBox Knowbook from simigon - MAK RTI
	Complier	- Microsoft Visual Studio 2010
	Purpose	-#1 MRM (High-resolution model)

Table 5 Operation environment for #2 MRM (Low-Resolution Simulation)

Computer	Equipment	Description
E2-117-N01	Desktop Computer	- CPU: Intel® Core i7-4770K Processor 3.5Ghz - HDD/RAM: 1TB/16GB - VGB: NVIDIA GeForce GTX 770 (2GB) - Monitor: 4x23inch LCD(1920x1080)
	O/S	- Window 7
	Operation	- MAK VR-Forces - Pitch RTI
	Complier	- Microsoft Visual Studio 2010
	Purpose	-#2 MRM (low-resolution model)

Table 6 Operation environment for #2 MRM (High-Resolution Simulation)

Computer	Equipment	Description
E1-289-01	Desktop Computer	- CPU: Intel® Core i7-4770K Processor 3.5Ghz - HDD/RAM: 1TB/16GB - VGB: NVIDIA GeForce GTX 770 (2GB) - Monitor: 23inch LCD(1920x1080)
	O/S	- Window 7
	Operation	- SIMbox Knowbook - Pitch RTI
	Complier	- Microsoft Visual Studio 2010
	Purpose	-#2 MRM (High-resolution model)

<u>4.3 Software</u>

Three COTS simulations are used for the study in UCF Simulation and Interoperability Lab. Two MRM federations are going to be constructed using these three simulations differently.

4.3.1 SIMbox

The Simigon developed a flight and surface to air missile (SAM) simulation system on the SIMBox simulation platform, a commercial off the shelf (COTS) simulation system for education. SIMbox is simulation software for military and civilian applications, and it provides a distributed simulation solution. SIMbox is an HLA compliant software.

4.3.2 MÄK VR-Forces

MÄK VR-Forces graphical user interface (GUI) provides the user with a 2D and 3D view of a simulated environment where the interaction between all entities can be observed. It is a powerful and flexible simulation tool of environment for generation of certain scenarios. VR-Forces satisfy the requirements of both DIS and HLA standards. VR-Forces also support HLA 1.3, 1516 and 1516 evolved specifications and HLA PRP-FOM through the mapping feature.

4.3.3 MASA Sword

MASA Sword is designed for the operational level of warfare. It is a good simulation for commanding post exercises, using an aggregated unit like division, brigade, battalion, and company. Opportunistic behaviors drive units in this simulation, and the user can modify the behavior based on their military doctrine. Therefore, it can reduce human intervention for every unit movement.

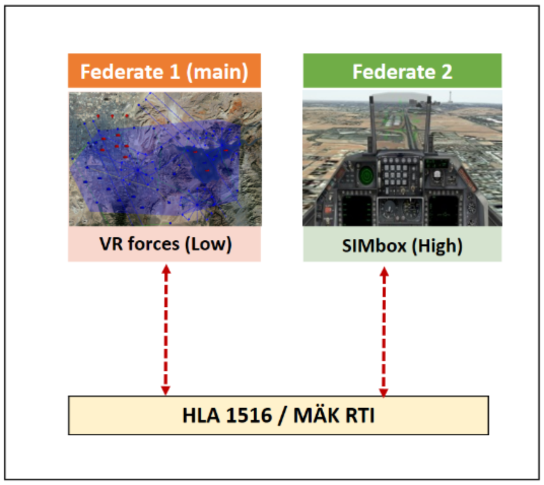
CHAPTER 5: CASE STUDY

Two different MRM federation systems will be presented; #1MRM federation (SIMBox & VR-Forces), and #2MRM federation (MASA Sword & VR-Forces). In this paper, several case studies will be conducted to observe a few issues of MRM. #2MRM is an ongoing project, and the progress to date will be discussed.

5.1. #1 MRM Federation System

5.1.1 Configuration

SIMBox simulation is a high-resolution model, and VR-Forces is a low-resolution model in #1 MRM federation. Those two legacy simulations are used for MRM federation system (Step 1), and part-distributed architecture is applied (Step 2). There is no regulator for this federation, as seen in Figure 33.



Contents	#1 MRM Federation
Low Resolution	VR-Forces from VT-MAK
High Resolution	SIMBOX from Simigon
RTI	HLA 1516 MAK RTI
Step 1	Reuse Legacy simulation
Step 2	Part-Distributed architecture
Step 3	No regulator

Figure 33 #1 MRM federation configuration

HLA 1516 MAK RTI is used for their communication & link. It is simple and loosens MRM. This allows for the observation of some issues of MRM through #1 MRM federation.

5.1.2 Case study #1 (Database mismatch)

Case study #1 experiment was designed to observe the database mismatch phenomenon in MRM federation. Eleven Blue force entities in VR-Forces (Figure 34 above left), and seven Red force entities in SIMBox (Figure 34 above right) were generated. Two simulations were then connected into a federation to see how their entities are represented in the mutual simulation.

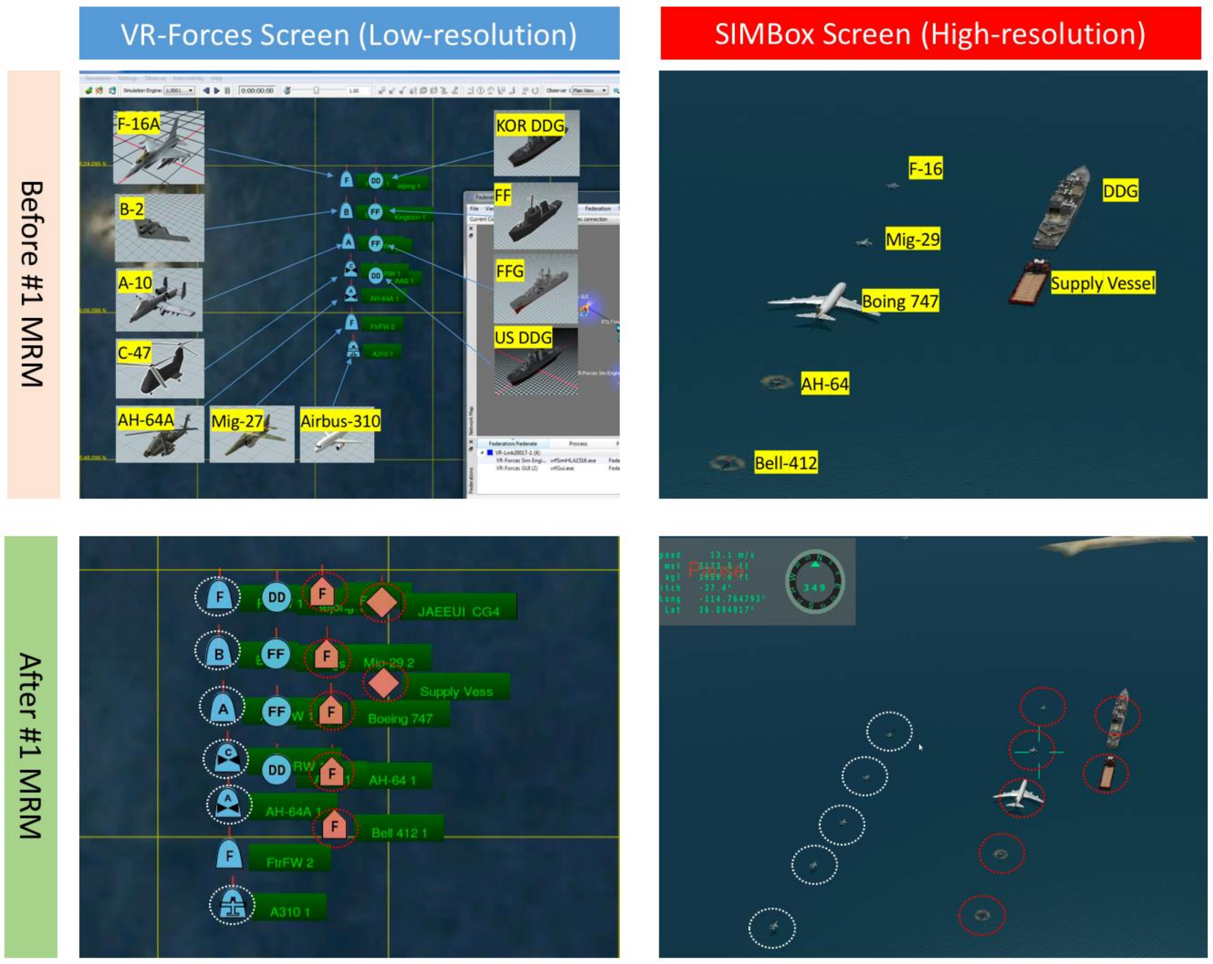


Figure 34 Case Study #1 Experiment (Database mismatch)

It can be seen in Figure 34, 7 of 7 SIMBox created entities were shown in the VR-Forces simulation, but only 5 of 11 VR-Forces entities were shown in the SIMBox simulation because of a database mismatch. Interestingly, the VR-forces information of naval ships not in the SIMBox database had been lost, and all of the aircraft represented were on the SIMBox screen with an F-16 aircraft appearance.

The two federates were designed for different purposes. The VR-Forces simulator was designed for commanding post exercise. Therefore, the VR-forces simulator has various military platform objects to create various military situations. The user can create up to 255 different entities in VR-forces, all entities exist within the context of a force, usually just called friendly, opposing, and neutral. On the other hand, the SIMBox simulation is a simulator for training an F-16 pilot. Therefore, SIMBox is focused on providing the same reality as the real world when the user controls the F-16 entity. Thus, only 62 object entities can be created in SIMBox.

Even though these two simulations share their information through HLA rule, the lack of a database in a certain simulator can create this issue within the MRM federation. This database gap will limit the entities that can be represented by exchanging mutual information. The findings from this research show that in order to make a seamless MRM, it is necessary to carry out the same database between two different simulations or perform data mapping appropriately.

5.1.3 Case study #2 (Resolution Difference)

Case study #2 experiment was designed to observe the effect of resolution differences on MRM federation. A high-resolution model, SIMBox, requires a relatively large number of property values when activating an entity. One F-16 will be generated in each of the two simulations, and two air-to-air (AA)/air-to-ground (AG) missiles will be mounted to see how they are represented in the other’s simulation. The F-16 is one of the units that both of the simulations have on their database and is the best interoperable unit.

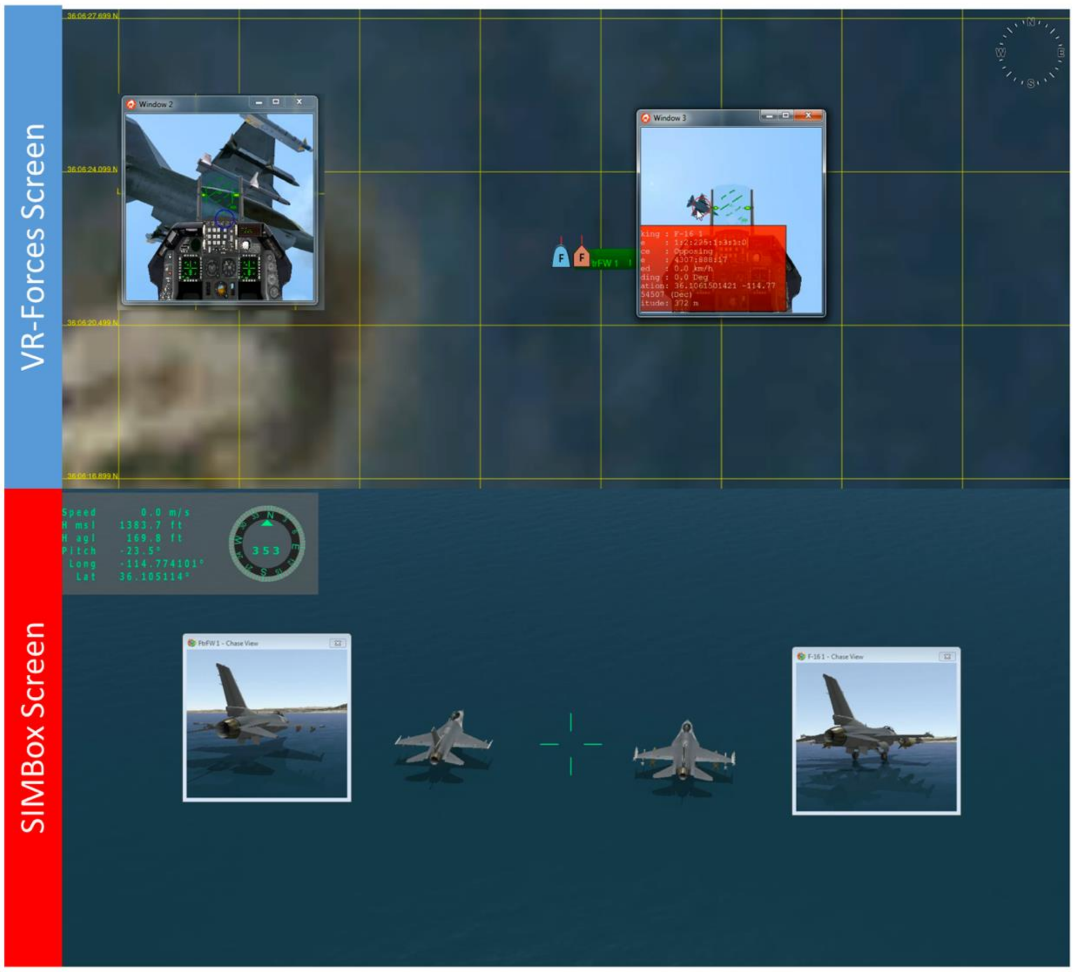


Figure 35 Data inconsistency observation

It can be seen in Figure 35, that the F-16 armed in each simulation has the same appearance in the VR-forces screen, but on the SIMBox, the F-16 generated in the VR-forces looks unarmed in its appearance. The F-16 in the VR-forces is required for attributes such as how many air-to-air missiles were armed in the F-16, but SIMBox F-16 requires more attributes values, such as how many armed, and where the missiles are mounted on the aircraft. A relatively large number of attribute values are required in the High-resolution model SIMBox. The differences in required attribute values created a gap between the two simulations. This gap is an example of data inconsistency in the MRM federation.

5.1.4 Case study #3 (Interactions)

Case study #3 was designed to observe how the results of the interactions between the mutually created units are reflected in both simulations. Three types of engagements were conducted (Figure 36). First, the guided-missile destroyer of SIMBox fire to the F-16 of VR-forces. Second, VR-forces F-16 fire a missile to SIMBox Mig-29. And lastly, a VR-forces surface to air missile launcher (which only appears in the VR-forces and does not exist in the SIMBox) attacks the SIMBox Mig-29.

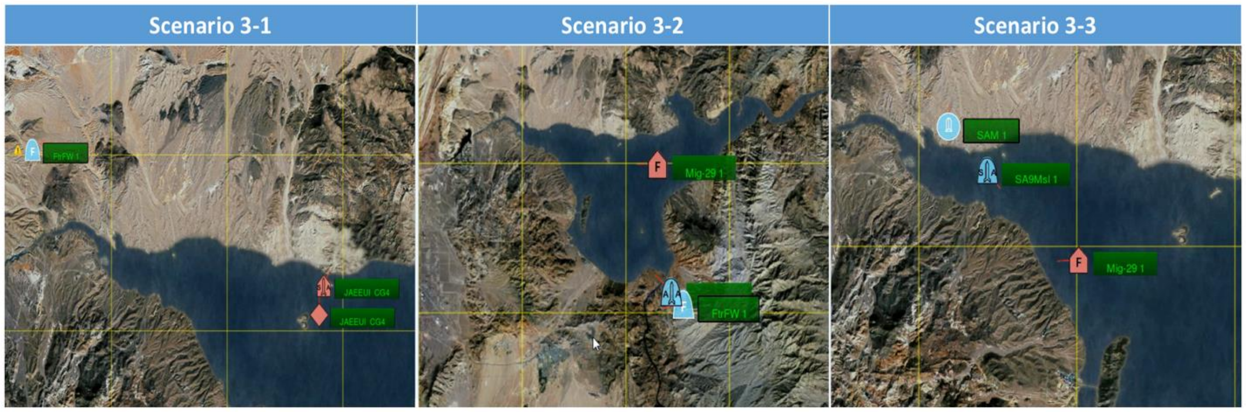
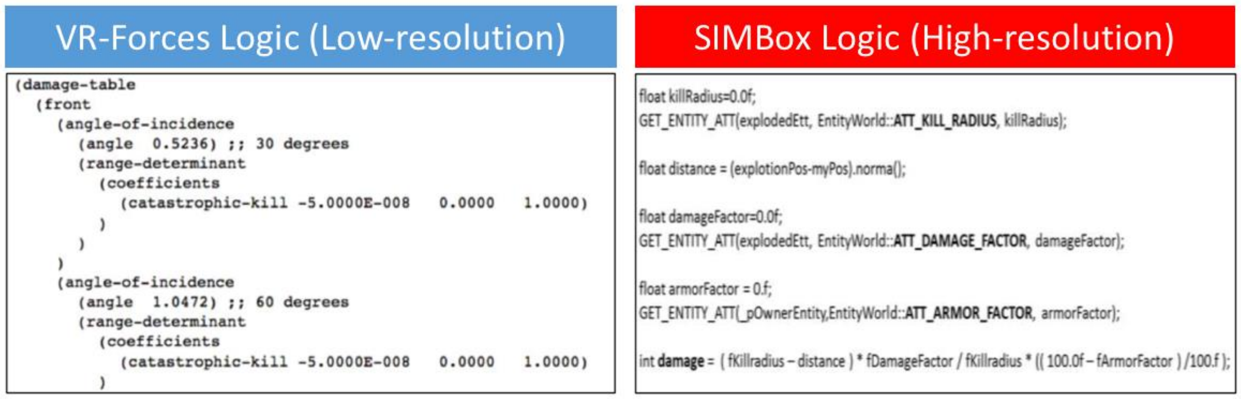


Figure 36 Scenarios for interactions

As can be seen in Figure 37, the engagement logic is different between two federates. In the engagement logic of SIMBox, the damage is decided by using Damage Factor, Armor Factor, and Kill Radius. Damage Value is calculated as a quantitative format from 0 to 100. In the engagement logic of VR-forces, the damage is decided by using the Probability of Hit (POH), Damage Model, and Armor Model. Damage value is determined as 0 (None), 1 (Slight), 2 (Moderate), 3 (Destroyed) (Park H., 2017).


VR-Forces Logic (Low-resolution)

```
(damage-table
  (front
    (angle-of-incidence
      (angle  0.5236) ;; 30 degrees
      (range-determinant
        (coefficients
          (catastrophic-kill -5.0000E-008   0.0000   1.0000)
        )
      )
    )
    (angle-of-incidence
      (angle  1.0472) ;; 60 degrees
      (range-determinant
        (coefficients
          (catastrophic-kill -5.0000E-008   0.0000   1.0000)
        )
```

SIMBox Logic (High-resolution)

```
float killRadius=0.0f;
GET_ENTITY_ATT(explodedEtt, EntityWorld::ATT_KILL_RADIUS, killRadius);

float distance = (explotionPos-myPos).norma();

float damageFactor=0.0f;
GET_ENTITY_ATT(explodedEtt, EntityWorld::ATT_DAMAGE_FACTOR, damageFactor);

float armorFactor = 0.f;
GET_ENTITY_ATT(_pOwnerEntity,EntityWorld::ATT_ARMOR_FACTOR, armorFactor);

int damage = ( fKillradius – distance ) * fDamageFactor / fKillradius * (( 100.0f – fArmorFactor ) /100.f );
```

Figure 37 Different engagement logics of two federates

Although the engagement logics are different, the results were the same in both simulations. That is because Object Model Template (OMT) in HLA/RTI defines the damage status as seen in Figure 38. All damage status will be shared as 0 (No Damage), 1 (Slight Damage), 2 (Moderate Damage), 3 (Destroyed). Therefore, the results of two federate engagements are kept the same in the federation.

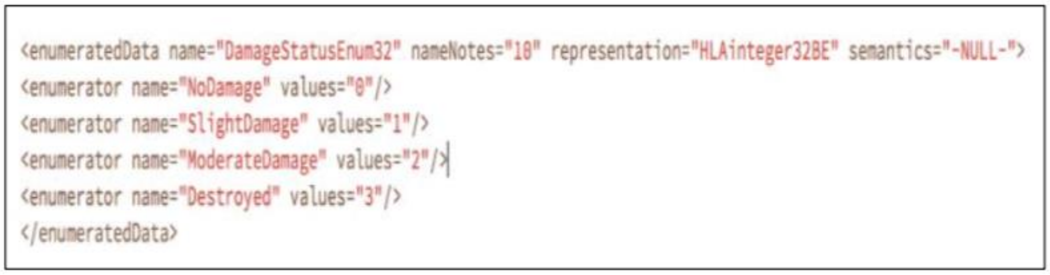

```
<enumeratedData name="DamageStatusEnum32" nameNotes="10" representation="HLAinteger32BE" semantics="-NULL-">
<enumerator name="NoDamage" values="0"/>
<enumerator name="SlightDamage" values="1"/>
<enumerator name="ModerateDamage" values="2"/>
<enumerator name="Destroyed" values="3"/>
</enumeratedData>
```

Figure 38 OMT Damage Status

To update specified values of attributes, the Request Attribute Value Update service should be used. By using this service, the RTI can get the desired values of the specified attributes by using the ‘Provide Attribute Value Update’ from other federates which have ownership of the attributes service (IEEE std, 2010). Figure 39 shows the message communication in an HLA/RTI federation.

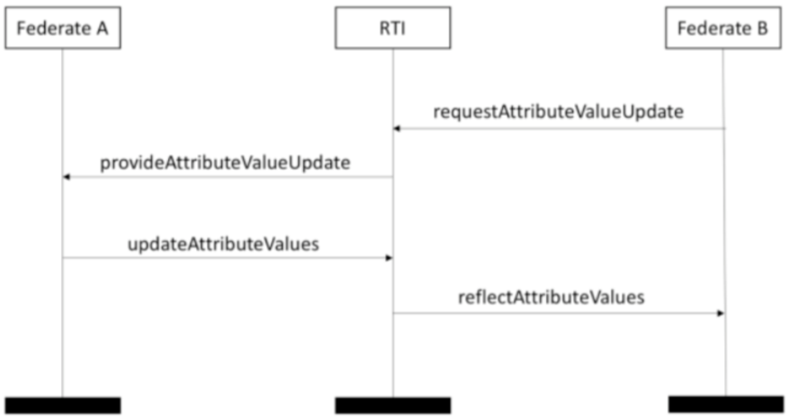


Figure 39 Communication in an HLA/RTI federation

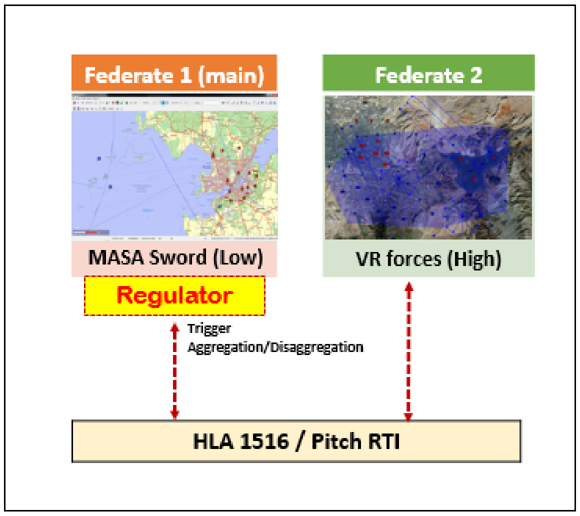
However, in Scenario 3-3, an attacker which is created by VR-forces still engaged and successfully destroyed a SIMBox entity, even though it does not appear in SIMBox because it is not in the database. This phenomenon may degrade the reality of the MRM federation system.

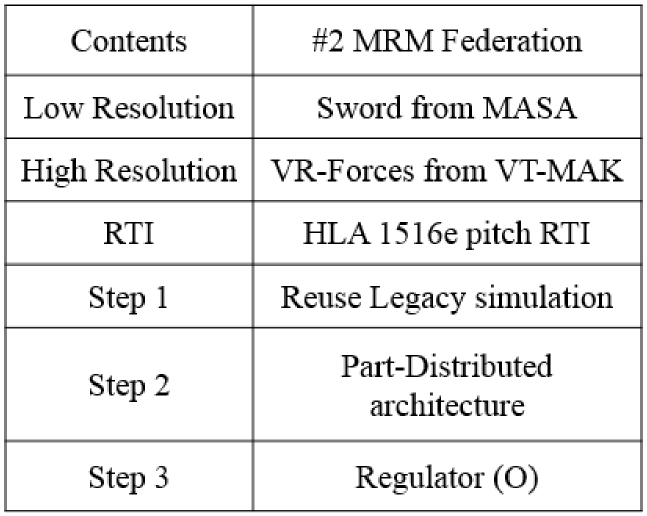
5.2. #2 MRM Federation System

5.2.1 Configuration

MASA Sword simulation is used as a low-resolution federate, and VR-forces participate as a high-resolution federate in #2 MRM federation system. As shown in Figure 40, this MRM federation uses two different COTS legacy simulations, MASA Sword & VR-Forces (Step 1), and the part-distributed approach is used as the architecture of federation (Step 2).

An Aggregation-Disaggregation approach is applied for dynamic changes from Unit to entities. The geographical and time-based trigger is applied to these dynamic changes. Therefore, the MRM with regulator Aggregation-Disaggregation, geographical trigger, and time trigger are included (Step 3, what to put in the regulator) in the module-based approach (Step3, where to put the regulator).



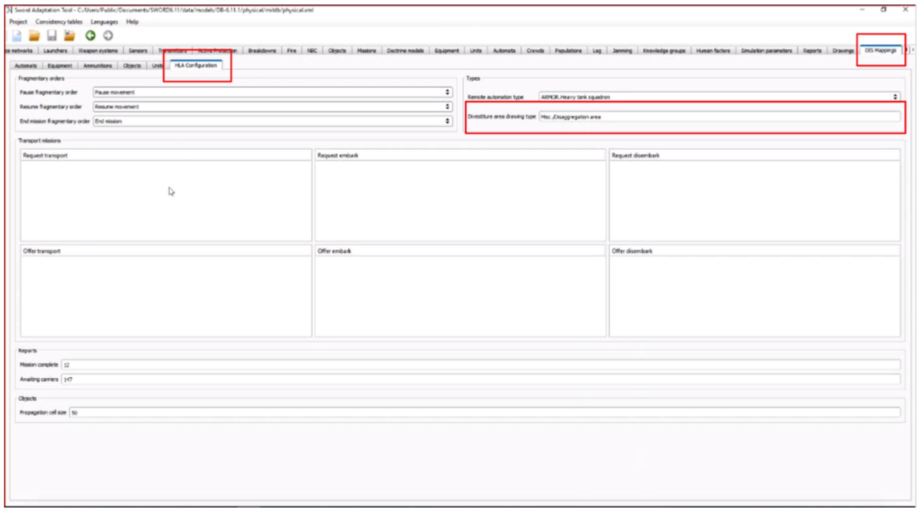


Contents	#2 MRM Federation
Low Resolution	Sword from MASA
High Resolution	VR-Forces from VT-MAK
RTI	HLA 1516e pitch RTI
Step 1	Reuse Legacy simulation
Step 2	Part-Distributed architecture
Step 3	Regulator (O)

Figure 40 #2 MRM federation configuration

5.2.2 How to set up the disaggregation-aggregation in MASA Sword

The modification was applied for dynamic changes in #2 MRM federation. The principle of how the MASA Sword can do the dynamic change is the use of the ‘Divestiture’ function, which the MASA Sword already has (Figure 41). Divestiture function in MASA Sword can hand its unit’s control over to the other federate in certain circumstances.



Pause fragmentary order: Pause movement
Resume fragmentary order: Resume movement
End mission fragmentary order: End mission
Remote automaton type: ARMOR.Heavy tank squadron
Divestiture area drawing type: Misc./Disaggregation area
Mission complete: 12
Awaiting carriers: 147
Propagation cell size: 50

Figure 41 Divestiture Setting in MASA Sword

5.2.3 To set Geographical Trigger

The condition of the ‘Divestiture’ function was set to a specific drawing in a section in order to use the method of passing control to another federate using a geographical trigger. The drawing of a polygon was linked to the ‘Divestiture’ function.

The polygon (Figure 42) represents a disaggregation area, which means that whenever a unit enters that area, its ownership is given automatically to another federate. For example, when the aggregated Unit of MASA Sword is passing the polygon section, then the ‘Divestiture’ function activates in the MASA Sword. This means that the MASA Sword takes over its right to control the VR-forces in the #2 MRM federation. The VR-forces may create several entities according to the information from the MASA Sword and control those disaggregated entities in the VR-forces simulation environment. When the unit passes back outside the polygon board, then MASA may retrieve their unit’s control and make them aggregate the unit automatically.

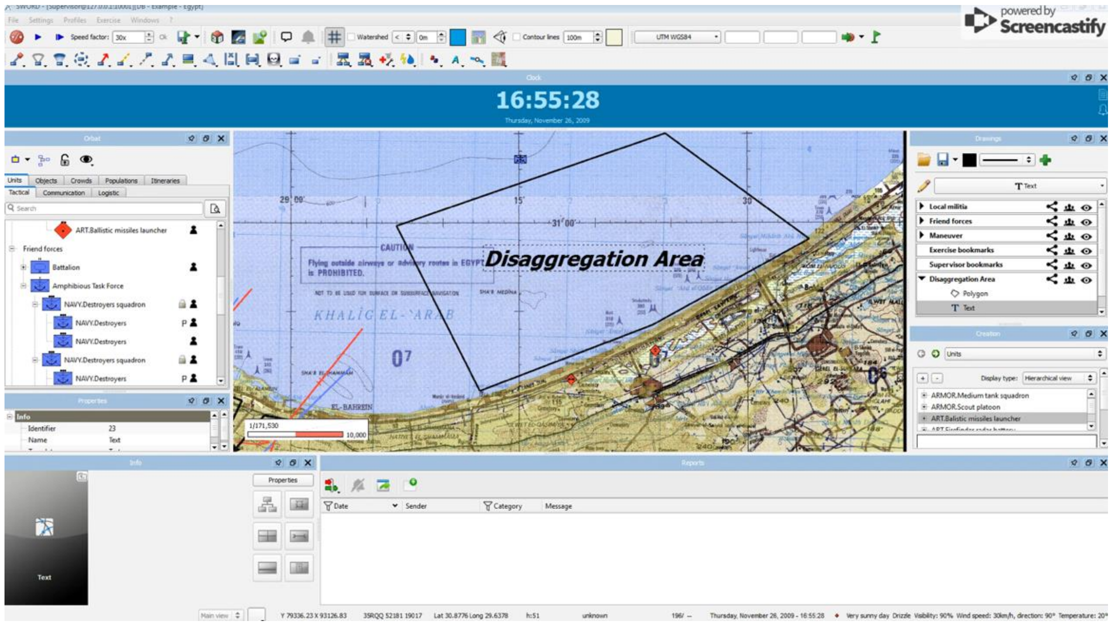
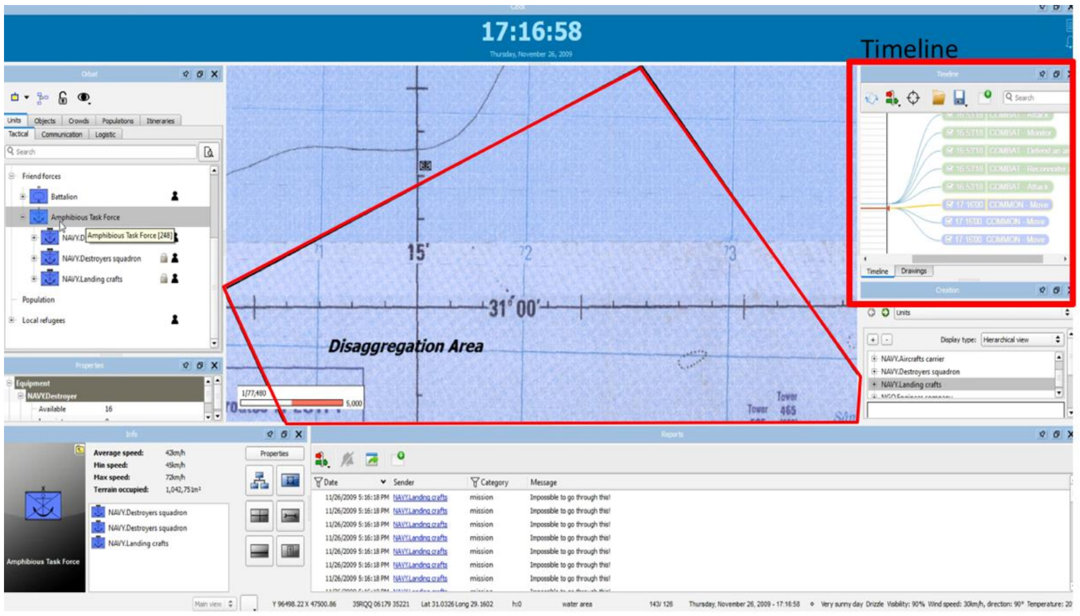


Figure 42 How to set the geographical trigger in MASA Sword

5.2.4 To set Time trigger

This research proposes to use ‘geographic trigger’ and ‘timeline function’ for building a time trigger in MASA Sword simulation. ‘Timeline’ is a function that allows certain units to perform a specific order at a given time. Therefore, if the user gives an order to move the unit beyond the disaggregation area at the designated time, the control of all units is passed to the VR-forces at that time. Thus, the same effect as the time trigger can be achieved.



Date	Sender	Category	Message
11/26/2009 5:16:18 PM	NAVY.Landing crafts	mission	Impossible to go through this!
11/26/2009 5:16:18 PM	NAVY.Landing crafts	mission	Impossible to go through this!
11/26/2009 5:16:18 PM	NAVY.Landing crafts	mission	Impossible to go through this!
11/26/2009 5:16:18 PM	NAVY.Landing crafts	mission	Impossible to go through this!
11/26/2009 5:16:18 PM	NAVY.Landing crafts	mission	Impossible to go through this!
11/26/2009 5:16:18 PM	NAVY.Landing crafts	mission	Impossible to go through this!
11/26/2009 5:16:18 PM	NAVY.Landing crafts	mission	Impossible to go through this!

Figure 43 How to set time trigger in MASA Sword

5.2.5 Scenario

The scenario for the #2MRM federation system is also based on Naval combat. In this scenario, two aggregated units will be disaggregated by the time-based trigger. The time-based trigger will be needed in military simulation because most military operations use the ‘D-day’ notion for building a plan of attack. D-day is a military term which has various meanings; it could be the landing operation day, or starting a raid day like the famous Normandy Beach Landing operation. The military used to set up specific actions for their operation based on D-day. For example, D-2; Deploy amphibious forces in the operation area, D-1; Search and sweep enemy submarines in the operation area, neutralization of all enemy's CDCMs (Coastal Defense Count Missiles), D-day; landing marine corps in the targeted shore. This scenario will be designed according to time-based factors. An aggregated unit will undergo a disaggregation process at a certain time, and when they finish their mission at a specific time, they will aggregate.

As shown in Figure 44, blue force AAF (Amphibious Assault Force) is trying to approach for landing shore in the low-level resolution model. Red forces consist of two different units, a CDCM group and an aircraft group to protect the target area and operate the anti-access strategy. All units are running in the low-level resolution during phase 1. Three different large groups of units maneuver to achieve their tasks; (1) AAF: Approach to the targeted shore, destroy or neutralize enemy CDCM and aircraft group, convoy landing ships. (2) CDCM: destroy or neutralize AAF using by surface to surface missiles. (3) Aircraft group: Destroy AAF.

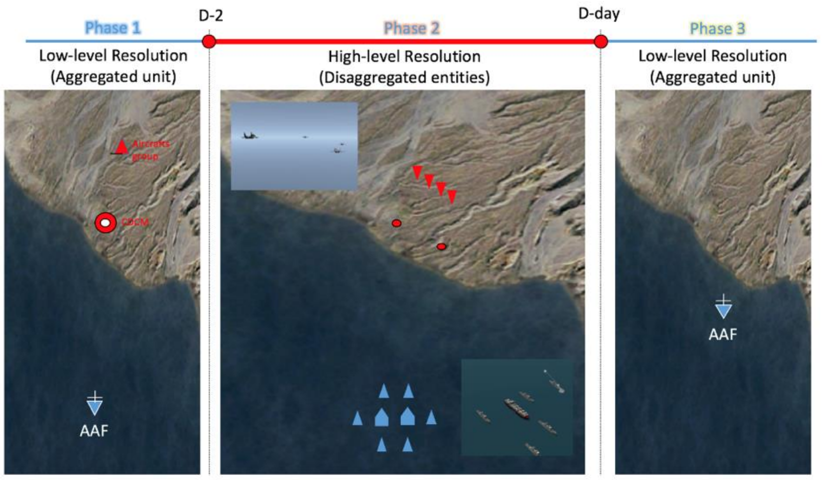

Phase 1
D-2
Phase 2
D-day
Phase 3
Low-level Resolution (Aggregated unit)
High-level Resolution (Disaggregated entities)
Low-level Resolution (Aggregated unit)
Aircrafts group
CDCM
AAF
AAF

Figure 44 Time-based MRM schematic implementation scenario

At a certain time (D-2), All Units (AAF, CDCM, and Aircraft group) are disaggregated. The disaggregation process instantiates the individual entities that make up the disaggregating unit in the entity level model (High-level resolution), and assigns the simulation to a location in the battlefield at entity level based on central location and formation of the unit. The unit level model corresponds to the unit's operating activity.

AAF		CDCM	Aircraft group
Missile defense Ship	Amphibious Ship	Missile launch Vehicle	Aircraft

Figure 45 Example of Disaggregated entities

For example, as shown in Figure 44, the AAF unit disaggregates to eight different entities (six missile defense type ships, two Amphibious type ships) maintaining hexagon formation. Each entity has their specific attributes, the missile defense type ship has SAMs (Surface to Air Missile) like the SM-2 or SM-3, to destroy enemy missiles or aircrafts and defend their HVU (High-Value Unit). It also has SSM (surface to surface or ground Missile) to attack ground entities like Tomahawk type missiles. Amphibious type ships are HVUs which have different attributes from Missile defense ships. CDCM will disaggregate into two SSM (Surface to Ship Missile) launch vehicles. Aircraft groups disaggregate into four aircrafts which have ASM (Air to Ship Missile). Tasks also should be distributed in each entity to achieve their own aggregated unit's task. For example, Missile defense ships have to convoy and defend all missiles which attempt to destroy their HVUs, in order to achieve AAF's goal. Missile launch vehicles and Aircrafts should attack HVUs to achieve CDCM and the Aircraft group's original goal.

After the disaggregation process, the simulation resumes as part of the scenario modeled at the entity level (High-level resolution model). The state of entities is always updated in both environments (Low-level resolution model and High-level resolution model) through the HLA connection. That means all interactions among entities will change attributes in high-level resolution models; those changes should reflect the attributes in low-level resolution models simultaneously.

CHAPTER 6: RESEARCH SUMMARY

6.1 Research Summary

The objectives of this research are:

- To provide an in-depth, comprehensive literature review about MRM for future research purposes.
- To put various notions and approaches of MRM together in one paper.
- To build an environment of MRM experimentation for the Simulation Interoperability Laboratory (SIL) of the University of Central Florida (UCF). During the research, a basic MRM system will be built by using two fundamentally different commercial war game simulators.
- To make define the word trigger, and its function in the MRM system through the literature review.
- To propose Steps for building an MRM system.
- To build a naval scenario to implement in MRM federation system.

Finally, the research objectives are accomplished as below:

- Literature review chapter introduces and explains a comprehensive notion related to MRM, put various notions and approaches of MRM together in this paper.
- The steps for developing MRM federation was proposed with the recommended MRM approaches in each step.
- The definition of a trigger is proposed through the literature review, and its function is explained.

- The environment of MRM experimentation for the Simulation Interoperability Laboratory (SIL) of the University of Central Florida (UCF) was established.
- Navy scenario was implemented in COTS MRM federation system.

6.2 Limitations

Many problems remain when it comes to building a seamless MRM federation. RTI problems occurred when configuring #2 MRM federation.

Pitch RTI did not connect with VR-forces simulation. It is assumed that certain settings of pitch RTI is needed to work with the VR-forces simulation, or VR-forces simulation needs to be modified to fit pitch RTI. MAK RTI was attempted in place of pitch RTI, but as MASA Sword does not connect to MAK RTI, it is presumed that there is a version confliction among them. Therefore, more research on RTI should be done.

In #2 MRM federation, the same map must be created before the two federates share a common situation. MASA Sword and VR-forces use different map formats. Therefore, it is necessary to precede the creation of the same format of map file of either MASA Sword or VR-forces.

As seen in #1 MRM federation case study, there should be efforts made to solve database mismatch problems. The database mismatch problem is difficult to solve without technical support from the two simulation manufacturers because the source-code needs to be disclosed, and the characteristics of the simulation and copyright both need to be considered as well.

The aggregation and disaggregation process with MASA Sword and VR-forces was not able to be shown in this paper (#2 MRM federation), due to a linking problem within the engineering level. However, further research will proceed with the #2 MRM federation.

6.3 Future Works

Future works will be:

- To build #2 MRM federation and implement naval scenario.
- To investigate computing traffic overloading by trigger types
- To design a regulator with the intent of building a seamless MRM federation system.

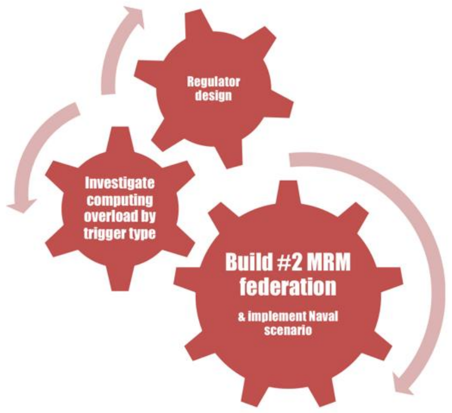

Regulator design
Investigate computing overload by trigger type
Build #2 MRM federation
& implement Naval scenario

Figure 46 Future works

The computer traffic overload will be checked when the trigger for aggregation and disaggregation is changed. Furthermore, future studies will investigate which factors are relevant to the computing traffic overload and the implementation.

SIL should have the ability to design a regulator in the long term. The regulator is the crux of the MRM system because the regulator is the factor that determines how seamless the MRM is. If SIL can design the regulator and incorporate various functions in it, then new and numerous studies and experiments will become possible.

REFERENCES

Bowers, A., & Prochnow, D. L. (2003). Multi-resolution modeling in the JTLS-JCATS Federation. Proceedings of the IEEE Fall Simulation Interoperability; IEEE CS Press.

Brailsford, S. (2014, 7-10 Dec. 2014). Modeling human behavior - an (id)entity crisis? Paper presented at the Simulation Conference (WSC), 2014 Winter.

Burbeck, S. (1992). Applications programming in smalltalk-80 (tm): How to use model-view-controller (mvc). Smalltalk-80 v2, 5.

Connors, C.D., Miller, J., & Lunday, B., (2015). "Using Agent-Based Modeling and a Designed Experiment to Simulate and Analyze a New Air-to-Air Missile." The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology 13(3):321-330.

Coolahan, J. (2003). "Modeling and simulation at APL," Johns Hopkins APL Technical Digest, vol. 24, no. 1, pp. 63–74.

Dahmann, J. S., Fujimoto, R. M., & Weatherly, R. M. (1997, December). The department of defense high level architecture. In Proceedings of the 29th conference on Winter simulation (pp. 142-149). IEEE Computer Society.

Daly, M., & Thorpe, D. (2009). Balancing Simulated and Live Naval Fleet Training. Paper presented at the The Inter service/Industry Training, Simulation & Education Conference (I/ITSEC).

Davis, P. K., & Bigelow, J. H. (1998). Experiments in multiresolution modeling (MRM): DTIC Document.

Davis, P. K., & Hillestad, R. (1993). Families of models that cross levels of resolution: Issues for design, calibration and management. Paper presented at the Proceedings of the 25th conference on Winter simulation.

Davis, P. K., & Tolk, A. (2007). Observations on new developments in composability and multi-resolution modeling. Paper presented at the Proceedings of the 39th conference on Winter simulation: 40 years! The best is yet to come.

DoD. (1995). Modeling and Simulation (M&S) Master Plan. (DoD 5000.59-P).

Hodson, D. D. (2009). Performance analysis of live-virtual-constructive and distributed virtual simulations: Defining requirements in terms of temporal consistency: Air Force Institute of Technology.

Hong, S.-Y., & Kim, T. G. (2007). A Resolution Converter for Multi-resolution Modeling/Simulation on HLA/RTI. In K. Koyamada, S. Tamura, & O. Ono (Eds.), Systems Modeling and Simulation: Theory and Applications, Asia Simulation Conference 2006 (pp. 289-293). Tokyo: Springer Japan.

Jung SangChul. (2001) 한국군 M&S 발전방안. 국방정책연구

Jung Wonil. (2017) An analytic framework of weapon combat effectiveness using the big data generated by live, virtual, or/and constructive(LVC) simulations: Three case studies constructive, live, and limited LVC simulations(Doctoral dissertation, University of Central Florida).

Macal, C., & North, M. (2009, 13-16 Dec. 2009). Agent-based modeling and simulation. Paper presented at the Simulation Conference (WSC), Proceedings of the 2009 WinterSim Conference.

Maria, A. (1997, December). Introduction to modeling and simulation. In Proceedings of the 29th conference on Winter simulation (pp. 7-13). IEEE Computer Society.

Martin Adelantado, & Pierre Siron, (2001). Multiresolution Modeling and Simulation of an Air-Ground Combat Application.

Mullen, F., Allen, G., Coolohan, J., Davis, P., Hanz, D., Hartman, F., Rouse, W., & Kees, S. (2013). Dynamic Multilevel Modeling Framework - Phase I – Feasibility. Report prepared for the Assistant Secretary of Defense, Research and Engineering (ASD(R&E)). The DoD Office of Security Review has cleared this document for public release (Distribution A) (Case No. 13-S-0966).

Natrajan, A., Reynolds Jr, P. F., & Srinivasan, S. (1997). MRE: a flexible approach to multi-resolution modeling. Paper presented at the Parallel and Distributed Simulation, 1997., Proceedings., 11th Workshop on.

Park, H. (2017). Design of a Framework for Sharing and Generating Combat Damage Assessment (CDA) of a HLA/RTI Federation.

Park, T. W., Kim, K., Rabelo, L., & Lee, G. (2015). An Agile Roadmap for Live, Virtual and Constructive-integrating Training Architecture (LVC-ITA): A Case Study Using a Component Based Integrated Simulation Engine (Doctoral dissertation, University of Central Florida).

Petty, M. D., & Weisel, E. (2003). A Composability Lexicon. In Proceedingsof the Spring 2003 Simulation Interoperability Workshop.

Petty, M. D., Dunning, R. L., & Collins, A. J. (2012, May). Expanded Analysis of the Correlation of Characterizing Attributes and Success in Military M&S Standards.

In Proceedings of the 2012 AlaSim International Modeling and Simulation Conference (pp. 1-3).

Reid, M. R. (2000). An Evaluation of the High Level Architecture (HLA) as a Framework for NASA Modeling and Simulation. NASA Software Engineering Workshop 25th.

Reynolds Paul F., N. a., Srinivasan sudhir. (1997). consistency maintenance in multiresolution simulations. ACM Transactions on modeling and computer simulation, 7(3), 368-392.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., & Lorenson, W. (1991). Object-oriented modelling and design.

Schricker SA, Franceschini RW and Adkins MK (1998). Using pseudo-disaggregation to populate a large battlefield for sensor systems. In: Proceedings of the Seventh Conference on Computer Generated Forces and Behavioral Representation, Orlando FL, May 12–14 , Institute for Simulation and Training, pp. 475–483.

Sharma, K. and Sagar, A. (2014). An Enhanced Agent Based Simulation using Selected Viewing Multi-Resolution Modeling. International Journal of Computer Science and Information Technologies, Vol. 5 (4) , 2014, 5345-5350.

Sven Skoid, G. H., Vahid Mojtahed. (1998). Building an Aggregation / Disaggregation Federation using Legacy Models. Fall Simulation interoperability Workshop(SIW)

Tan, G., Ng, W. N., & Moradi, F. (2001). Aggregation / Disaggregation in HLA multi-resolution distributed simulation. IEEE/ACM International Symposium on.

Tolk, A. 1999. Non-monotonicities in HLA-federations. In Proceedings of the Spring Simulation Interoperability Workshop. IEEE CS Press

Tolk, A. 2003. Common data administration, data management, and data alignment as a necessary requirement for coupling C4ISR systems and M&S.Journal for Information and Security 12(2): 164-174.

Tolk, A. (2006). Multi-resolution Challenges for Command and Control M&S Services. Paper presented at the Proceedings of 2006 Spring Simulation Interoperability Workshop.

Tolk, A. (2012). Engineering Principles of Combat Modeling and Distributed Simulation: John Wiley & Sons.

Weisel, E. W., M. D. Petty, and R. R. Mielke. (2003). Validity of models and classes of models in semantic composability. In Proceedings of the Fall Simulation Interoperability Workshop. IEEE CS Press.

Yilmaz, L., Lim, A., Bowen, S., & Oren, T. (2007). Requirements and design principles for multisimulation with multiresolution, multistage multimodels. Paper presented at the Simulation Conference, 2007 Winter.