



2013-09

# Hybrid architectural framework for C4ISR and Discrete-Event Simulation (DES) to support sensor-driven model synthesis in real-world scenarios

Chen, You-Quan

Monterey California. Naval Postgraduate School

---



Calhoun is a project of the Dudley Knox Library at NPS, furthering the precepts and goals of open government and government transparency. All information contained herein has been approved for release by the NPS Public Affairs Officer.

**Dudley Knox Library / Naval Postgraduate School**  
**411 Dyer Road / 1 University Circle**  
**Monterey, California USA 93943**



**NAVAL  
POSTGRADUATE  
SCHOOL**

**MONTEREY, CALIFORNIA**

**DISSERTATION**

**HYBRID ARCHITECTURAL FRAMEWORK FOR C4ISR AND  
DISCRETE-EVENT SIMULATION (DES) TO SUPPORT  
SENSOR-DRIVEN MODEL SYNTHESIS IN REAL-WORLD  
SCENARIOS**

by

You-Quan Chen

September 2013

Dissertation Supervisors:

Don Brutzman  
Phillip Pace

**This dissertation was performed at the MOVES Institute  
Approved for public release; distribution is unlimited**

THIS PAGE INTENTIONALLY LEFT BLANK

| <b>REPORT DOCUMENTATION PAGE</b>  |   |  | <i>Form Approved OMB No. 0704-0188</i>                  |  |
|---|---|--|---|--|
| Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.   |   |  |   |  |
| <b>1. AGENCY USE ONLY (Leave blank)</b>   |   | <b>2. REPORT DATE</b><br>September 2013                        | <b>3. REPORT TYPE AND DATES COVERED</b><br>Dissertation |  |
| <b>4. TITLE AND SUBTITLE</b><br>HYBRID ARCHITECTURAL FRAMEWORK FOR C4ISR AND DISCRETE-EVENT SIMULATION (DES) TO SUPPORT SENSOR-DRIVEN MODEL SYNTHESIS IN REAL-WORLD SCENARIOS   |   |  | <b>5. FUNDING NUMBERS</b>                               |  |
| <b>6. AUTHOR</b> Chen, You-Quan   |   |  |   |  |
| <b>7. PERFORMING ORGANIZATION NAME AND ADDRESS</b><br>Naval Postgraduate School<br>Monterey, CA 93943-5000  |   |  | <b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>         |  |
| <b>9. SPONSORING /MONITORING AGENCY NAME AND ADDRESS</b><br>N/A   |   |  | <b>10. SPONSORING/MONITORING AGENCY REPORT NUMBER</b>   |  |
| <b>11. SUPPLEMENTARY NOTES</b> The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number: N/A.   |   |  |   |  |
| <b>12a. DISTRIBUTION / AVAILABILITY STATEMENT</b><br>Approved for public release; distribution is unlimited   |   |  | <b>12b. DISTRIBUTION CODE</b>                           |  |
| <b>13. ABSTRACT</b><br>While the application of a time-step approach in modeling C4ISR in Missile Defense Warfare (MDW) suffers inaccurate time estimation and relative slow speed, Discrete Event Simulation (DES) can elegantly satisfy these shortages. However, current DES frameworks typically rely on detailed efforts in event analysis for numerous replications before software modification of the simulation scenario can be meaningful. Such approaches have limited adaptability, especially regarding flexibility of scenario design and customizability of entity definition. This dissertation proposes an improved DES framework, Adjustable and Extensible Modeling Framework DES (AEMF-DES), which embeds the primary principles of a topical theme into a program to perform adjustable and extensible studies that can be explored by the analyst. To prove the feasibility of AEMF-DES, a Missile-Defense Simulation application (MDSIM) is also developed during this research. MSDIM simulates the C4ISR processes in Missile Defense Warfare and can estimate the overall effectiveness of a defender's deployment or attacker's strategy. Additionally, based on the interest in sensor deployment evaluation, a k-coverage rate problem is also studied. Current k-coverage algorithms can only deal with binary and omnidirectional sensor models which cannot provide enough simulation fidelity if higher resolution is needed. An improved k-coverage rate algorithm is proposed in this research to handle the probabilistic and directional sensor models. A separate simulation test successfully demonstrates the feasibility of this new calculation algorithm in estimation of the k-coverage rate problem with probabilistic and directional sensor models. Considered together, the architecture implemented in this example software illustrates the value of integrating hybrid simulation techniques to support C4ISR analysis related to Missile Defense Warfare. |   |  |   |  |
| <b>14. SUBJECT TERMS</b><br>Simulation, Discrete Event Simulation, DES, Computer Managed DES, AEMF-DES, missile defense warfare, MDW, K-Coverage Rate   |   |  | <b>15. NUMBER OF PAGES</b><br>193                       |  |
|   |   |  | <b>16. PRICE CODE</b>                                   |  |
| <b>17. SECURITY CLASSIFICATION OF REPORT</b><br>Unclassified  | <b>18. SECURITY CLASSIFICATION OF THIS PAGE</b><br>Unclassified | <b>19. SECURITY CLASSIFICATION OF ABSTRACT</b><br>Unclassified | <b>20. LIMITATION OF ABSTRACT</b><br>UU                 |  |

THIS PAGE INTENTIONALLY LEFT BLANK

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

**HYBRID ARCHITECTURAL FRAMEWORK FOR C4ISR AND DISCRETE-  
EVENT SIMULATION (DES) TO SUPPORT SENSOR-DRIVEN MODEL  
SYNTHESIS IN REAL-WORLD SCENARIOS**

You-Quan Chen  
Lieutenant Commander, R.O.C. (Taiwan) Navy  
B.S., National Defense University, R.O.C. (Taiwan), 2000  
M.S., Naval Postgraduate School, 2007

Submitted in partial fulfillment of the requirements for the degree of

**DOCTOR OF PHILOSOPHY IN  
MODELING, VIRTUAL ENVIRONMENTS, AND SIMULATION**

from the  
**NAVAL POSTGRADUATE SCHOOL**  
**September 2013**

Author:

\_\_\_\_\_  
You-Quan Chen

Approved by:

\_\_\_\_\_  
Don Brutzman  
Associate Professor of  
Applied Science  
Dissertation Supervisor

\_\_\_\_\_  
Phillip Pace  
Professor of  
Electrical and Computer Engineering  
Dissertation Co-Supervisor

\_\_\_\_\_  
Thomas Lucas  
Professor of  
Operational Research

\_\_\_\_\_  
Xiaoping Yun  
Professor of  
Electrical and Computer Engineering

\_\_\_\_\_  
Arnold Buss  
Research Associate Professor of  
MOVES

\_\_\_\_\_  
Deborah Goshorn  
Research Assistant Professor of  
Electrical and Computer Engineering

Approved by: \_\_\_\_\_

Christian J. Darken, Chair, MOVES Academic Committee

Approved by: \_\_\_\_\_

Peter J. Denning, Chair, Department of Computer Science

Approved by: \_\_\_\_\_

Douglas Moses, Vice Provost for Academic Affairs

THIS PAGE INTENTIONALLY LEFT BLANK

## ABSTRACT

While the application of a time-step approach in modeling C4ISR in Missile Defense Warfare (MDW) suffers inaccurate time estimation and relative slow speed, Discrete Event Simulation (DES) can elegantly satisfy these shortages. However, current DES frameworks typically rely on detailed efforts in event analysis for numerous replications before software modification of the simulation scenario can be meaningful. Such approaches have limited adaptability, especially regarding flexibility of scenario design and customizability of entity definition. This dissertation proposes an improved DES framework, Adjustable and Extensible Modeling Framework DES (AEMF-DES), which embeds the primary principles of a topical theme into a program to perform adjustable and extensible studies that can be explored by the analyst. To prove the feasibility of AEMF-DES, a Missile-Defense Simulation application (MDSIM) is also developed during this research. MSDIM simulates the C4ISR processes in Missile Defense Warfare and can estimate the overall effectiveness of a defender's deployment or attacker's strategy. Additionally, based on the interest in sensor deployment evaluation, a k-coverage rate problem is also studied. Current k-coverage algorithms can only deal with binary and omnidirectional sensor models which cannot provide enough simulation fidelity if higher resolution is needed. An improved k-coverage rate algorithm is proposed in this research to handle the probabilistic and directional sensor models. A separate simulation test successfully demonstrates the feasibility of this new calculation algorithm in estimation of the k-coverage rate problem with probabilistic and directional sensor models. Considered together, the architecture implemented in this example software illustrates the value of integrating hybrid simulation techniques to support C4ISR analysis related to Missile Defense Warfare.



THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS

|             |  |           |
|-------------|--|-----------|
| <b>I.</b>   | <b>INTRODUCTION.....</b>   | <b>1</b>  |
| <b>A.</b>   | <b>OVERVIEW OF THIS DISSERTATION .....</b>   | <b>1</b>  |
| <b>B.</b>   | <b>MOTIVATING PROBLEMS.....</b>  | <b>2</b>  |
|             | <b>1. Time-Step Approach: Advantages and Limitations .....</b>   | <b>2</b>  |
|             | <b>2. Discrete Event Simulation (DES): Advantages and Limitations.....</b>   | <b>3</b>  |
|             | <b>3. New Framework Proposal.....</b>  | <b>4</b>  |
| <b>C.</b>   | <b>RESEARCH OBJECTIVES .....</b>   | <b>5</b>  |
| <b>D.</b>   | <b>CONTRIBUTIONS.....</b>  | <b>5</b>  |
|             | <b>1. An Improved DES Framework:<br/>Adjustable-and-Extensible-Modeling-Framework DES .....</b>                        | <b>5</b>  |
|             | <b>2. A Simulation Architecture for C4ISR in Missile Defense<br/>Warfare.....</b>                                      | <b>6</b>  |
|             | <b>3. An Improved <math>k</math>-Coverage Rate Algorithm.....</b>  | <b>8</b>  |
| <b>E.</b>   | <b>NOMENCLATURE.....</b>   | <b>9</b>  |
| <b>F.</b>   | <b>EXAMPLE SCENARIO .....</b>  | <b>10</b> |
| <b>G.</b>   | <b>DISSERTATION ORGANIZATION .....</b>   | <b>11</b> |
| <b>II.</b>  | <b>RELATED WORK.....</b>   | <b>13</b> |
| <b>A.</b>   | <b>TIME-ADVANCE MECHANISMS (TAMS) .....</b>  | <b>13</b> |
|             | <b>1. Time-Step Approach: Discrete-Time Simulation (DTS).....</b>  | <b>13</b> |
|             | <b>2. Discrete-Event Simulation (DES) .....</b>  | <b>14</b> |
|             | <b>3. Comparison Summary.....</b>  | <b>15</b> |
| <b>B.</b>   | <b>DISCRETE EVENT SYSTEM SPECIFICATION (DEVS).....</b>   | <b>15</b> |
| <b>C.</b>   | <b>REAL-TIME SIMULATION AND CONTROL .....</b>  | <b>16</b> |
|             | <b>1. Real-Time Control .....</b>  | <b>17</b> |
|             | <b>2. Real-Time Simulation with Time-Step Approach .....</b>   | <b>17</b> |
|             | <b>3. RT Simulation with DES Approach.....</b>   | <b>17</b> |
| <b>D.</b>   | <b>PHYSICAL FIDELITY.....</b>  | <b>18</b> |
| <b>E.</b>   | <b>AGENT-BASED MODEL (ABM).....</b>  | <b>18</b> |
| <b>F.</b>   | <b>LOOSE COUPLING .....</b>  | <b>19</b> |
| <b>G.</b>   | <b>HIGH-LEVEL ARCHITECTURE (HLA) SIMULATION .....</b>  | <b>19</b> |
| <b>H.</b>   | <b>COMMAND, CONTROL, COMPUTER, COMMUNICATION,<br/>INTELLIGENCE, SURVEILLANCE, AND RECONNAISSANCE<br/>(C4ISR) .....</b> | <b>19</b> |
| <b>I.</b>   | <b>C4ISR-RELATED SIMULATIONS.....</b>  | <b>22</b> |
| <b>J.</b>   | <b>VERIFICATION, VALIDATION AND ACCREDITATION (VV&amp;A)...</b>  | <b>25</b> |
| <b>K.</b>   | <b><math>k</math>-COVERAGE RATE PROBLEM FOR SENSOR<br/>EFFECTIVENESS.....</b>  | <b>26</b> |
| <b>III.</b> | <b>AN IMPROVED DES FRAMEWORK: ADJUSTABLE AND EXTENSIBLE<br/>MODELING FRAMEWORK DES (AEMF-DES).....</b>                 | <b>29</b> |
| <b>A.</b>   | <b>LIMITATIONS OF CURRENT DES APPROACHES .....</b>   | <b>29</b> |
| <b>B.</b>   | <b>DESIGN CHALLENGE AND SOLUTION STRATEGY .....</b>  | <b>31</b> |

|      |  |     |
|------|--|-----|
| C.   | <b>ADJUSTABLE-AND-EXTENSIBLE-MODELING-FRAMEWORK<br/>DES (AEMF-DES)</b> .....         | 32  |
| 1.   | <b>Concept of AEMF-DES</b> .....   | 33  |
| 2.   | <b>Program Architecture</b> .....  | 43  |
| D.   | <b>SUMMARY</b> .....   | 44  |
| IV.  | <b>DEMONSTRATED SIMULATION PROGRAM: MISSILE DEFENSE<br/>SIMULATION (MDSIM)</b> ..... | 45  |
| A.   | <b>MDSIM CONCEPT AND SCOPE</b> .....   | 45  |
| B.   | <b>DEVELOPMENT STEPS</b> .....   | 47  |
| 1.   | <b>Build Fundamental Theme Agents (FTAs)</b> .....                                   | 47  |
| 2.   | <b>Build Function for Custom Entity Templates</b> .....                              | 64  |
| 3.   | <b>Build Function for Scenarios Design</b> .....                                     | 66  |
| 4.   | <b>Build Function for Simulation Execution</b> .....                                 | 68  |
| C.   | <b>FUTURE GOAL</b> .....   | 70  |
| 1.   | <b>Simulation Societies and C4ISR</b> .....  | 70  |
| 2.   | <b>Simulation and VV&amp;A Processes</b> .....                                       | 72  |
| D.   | <b>SUMMARY</b> .....   | 72  |
| V.   | <b>EXPERIMENT SIMULATIONS</b> .....  | 75  |
| A.   | <b>EXPERIMENT SERIES 1 : IMPROVED FLEXIBILITY FROM<br/>AEMF-DES</b> .....            | 76  |
| 1.   | <b>Sub-Experiments and Results</b> .....   | 77  |
| 2.   | <b>Conclusions</b> .....   | 85  |
| B.   | <b>EXPERIMENT SERIES 2 : C4ISR PROCESSES IN MISSILE<br/>DEFENSE WARFARE</b> .....    | 85  |
| 1.   | <b>Sub-Experiment and Results</b> .....  | 86  |
| 2.   | <b>Conclusions and Limitation</b> .....  | 106 |
| C.   | <b>SUMMARY</b> .....   | 106 |
| VI.  | <b>SENSOR ANALYSIS: <math>k</math>-COVERAGE RATE PROBLEM</b> .....                   | 109 |
| A.   | <b>PROBABILISTIC AND SECTORIAL SENSOR MODEL</b> .....                                | 109 |
| 1.   | <b>Probabilistic Sensing</b> .....   | 109 |
| 2.   | <b>An Example of Probabilistic Sensor: Pulse Radar System</b> .....                  | 110 |
| 3.   | <b>Sectorial Sensing Coverage</b> .....  | 112 |
| B.   | <b>CHALLENGES AND SOLUTIONS</b> .....  | 112 |
| 1.   | <b>Probabilistic Sensing</b> .....   | 113 |
| 2.   | <b>Sectorial Coverage</b> .....  | 113 |
| 3.   | <b>Improved Algorithm</b> .....  | 115 |
| C.   | <b>EXPERIMENTS</b> .....   | 116 |
| 1.   | <b>Experiment 1</b> .....  | 116 |
| 2.   | <b>Experiment 2</b> .....  | 118 |
| D.   | <b>SUMMARY</b> .....   | 121 |
| VII. | <b>CONCLUSIONS AND RECOMMENDATIONS</b> .....   | 123 |
| A.   | <b>IMPROVED DES FRAMEWORK: AEMF-DES</b> .....  | 123 |
| 1.   | <b>Conclusions</b> .....   | 123 |
| 2.   | <b>Limitations</b> .....   | 123 |

|   |   |     |
|---|---|-----|
| 3.  | Recommendations for Future Work .....   | 124 |
| B.  | A SIMULATION ARCHITECTURE OF C4ISR IN MISSILE<br>DEFENSE WARFARE: MDSIM ..... | 125 |
| 1.  | Conclusions .....   | 125 |
| 2.  | Limitations .....   | 125 |
| 3.  | Recommendations for Future Work .....   | 125 |
| C.  | AN IMPROVED $k$ -COVERAGE RATE ALGORITHM .....                                | 127 |
| 1.  | Conclusions .....   | 127 |
| 2.  | Recommendations for Future Work .....   | 127 |
| APPENDIX A: USER MANUAL OF MDSIM .....  |   | 129 |
| A.  | INTRODUCTION .....  | 129 |
| 1.  | Purpose of This Document .....  | 129 |
| 2.  | Content Organization .....  | 129 |
| B.  | WHAT MDSIM IS .....   | 129 |
| 1.  | Background .....  | 129 |
| 2.  | Installation and Execution .....  | 129 |
| 3.  | Operation Process .....   | 129 |
| C.  | TUTORIAL: FROM DETECTION TO FIRE .....  | 130 |
| 1.  | Review Loaded Part Templates .....  | 130 |
| 2.  | Build Custom Entity Templates .....   | 131 |
| 3.  | Design a Scenario .....   | 138 |
| 4.  | Run a Simulation .....  | 146 |
| 5.  | Review the Result .....   | 148 |
| APPENDIX B: USER MANUAL OF THE $k$ -COVERED RATE CALCULATION<br>PROGRAM ..... |   | 153 |
| A.  | INTRODUCTION .....  | 153 |
| B.  | WHAT THIS PROGRAM IS .....  | 153 |
| 1.  | Background .....  | 153 |
| 2.  | File Structure .....  | 153 |
| 3.  | Installation and Execution .....  | 154 |
| C.  | TUTORIAL: FROM DETECTION TO FIRE .....  | 154 |
| APPENDIX C: SOFTWARE AVAILABILITY OF EXAMPLE SIMULATION<br>PROGRAM .....      |   | 157 |
| A.  | ACCESS ELIGIBILITY .....  | 157 |
| B.  | SOFTWARE LICENSE AND DISCLAIMER .....   | 157 |
| C.  | AVAILABLE AT SOURCE ARCHIVE .....   | 158 |
| D.  | LIST OF ASSETS .....  | 158 |
| E.  | VIDEO DESCRIPTION .....   | 158 |
| LIST OF REFERENCES .....  |   | 161 |
| INITIAL DISTRIBUTION LIST .....   |   | 171 |

THIS PAGE INTENTIONALLY LEFT BLANK

## LIST OF FIGURES

|            |   |    |
|------------|---|----|
| Figure 1.  | Potential future hybrid architecture where DES can cooperate with time-step methods to support C4ISR simulation. ....   | 8  |
| Figure 2.  | A theme can contain multiple scenarios which include entities and parts.....  | 10 |
| Figure 3.  | In this simple example scenario, a radar station and an anti-air weapon system cooperate to intercept an invasive missile. ....                                     | 11 |
| Figure 4.  | Step-wise executions of the time-step approach (from Al Rowaei et al. 2011). ....   | 14 |
| Figure 5.  | OODA Loop contains 4 processes in a military operation. ....  | 20 |
| Figure 6.  | Simplified information flow in each C4ISR process contains its generation, delivering, and utilization. ....  | 21 |
| Figure 7.  | A rational behavior includes sense, decide, and act.....  | 21 |
| Figure 8.  | The ratio of time-advance mechanism (TAM) methodologies used in C4ISR related research examined in this dissertation.....   | 25 |
| Figure 9.  | Improved grid scan method separates grids into three states (after Sheu et al. 2008). ....  | 28 |
| Figure 13. | Fundamental Theme Agents (FTAs) are concluded from the target theme and can derive possible events for a specific scenario which must belong to the same theme..... | 35 |
| Figure 14. | Entity can contain multiple concrete part classes and represent a combat unit while abstract part type classes define the general behavior of each part type. ....  | 36 |
| Figure 15. | A class diagram of the sample scenario shows three portions of FTA: entity class, abstract part type classes, and concrete part classes .....                       | 38 |
| Figure 17. | The PES of the sample scenario shows that a scenario contains multiple entities which also contain multiple parts to offer different functions. ....                | 40 |
| Figure 20. | A scenario derived from user’s need is described by PES, which utilizes FTA and then loads it into a DES engine to generate simulation results. ....                | 44 |
| Figure 21. | This simulation architecture is contributed from AEMF-DES and previous DES works. ....  | 45 |
| Figure 22. | Besides the Entity class, Abstract part classes and Concrete part classes construct the FTA of MDSIM.....   | 48 |
| Figure 24. | Core methods of the Sensor class along with pseudo codes.....   | 51 |
| Figure 25. | Core methods of the CommNode class which uses head and tail sign to model information transmission. ....  | 55 |
| Figure 26. | An illustration of the headquarters model concept. ....   | 57 |
| Figure 27. | The core methods of the abstract Headquarters class. ....   | 58 |
| Figure 28. | The core methods of the abstract Weapon class. ....   | 62 |
| Figure 29. | Illustration of part interface displaying successfully loaded information.....  | 64 |
| Figure 30. | Interface lets users edit an entity’s properties and arrange part settings.....   | 65 |
| Figure 31. | Available entity templates are listed. ....   | 66 |
| Figure 32. | Interface design for Scenario Add/Edit.....   | 68 |
| Figure 33. | Design interface for executing a simulation. ....   | 69 |
| Figure 34. | Design interface for viewing simulation results. ....   | 69 |

|            |  |     |
|------------|--|-----|
| Figure 35. | Communities related include C4ISR, DES, and the time-step approach.....  | 70  |
| Figure 36. | C4ISR and the time-step communities have cooperated to manage the needs on analyzing practical C4ISR processes. ....   | 70  |
| Figure 37. | Both DES and the time-step approach are required to fully satisfy the needs in C4ISR analysis. Achieving this comprehensive hybrid architecture is future work. ....   | 71  |
| Figure 38. | A time-step-event mediator can be added into FTA to enable those customized functions that need a time-temp mechanism .....  | 72  |
| Figure 39. | Add simulation to the validation process.....  | 72  |
| Figure 40. | In this user-defined scenario, a missile attempts to fly across the coverage of a radar station.....   | 77  |
| Figure 41. | Simulation result (excerpted from detailed simulation event log) shows that the radar station can detect the missile within the coverage region.....   | 79  |
| Figure 42. | Before it destroys the warehouse, the missile traverse the radar's coverage (subjected to geometry limitation) (also see Figure 44 for more detail).....   | 79  |
| Figure 43. | Attacker's missile arrives at its destination and destroys the warehouse. ....   | 80  |
| Figure 46. | After the warehouse entity is added with a new sensor part, it now can act like the radar to detect the missile.....   | 82  |
| Figure 47. | After the missile traverses the coverage of the radar, it enters the range of the modified warehouse which also has a radar sensor part (subjected to geometry limitation)(also see Figure 49 for more detail). .... | 83  |
| Figure 50. | A missile attempts to fly across the coverage of a radar station which has a communication link to HQ.....   | 87  |
| Figure 51. | Simulation event-log results show that the user-defined scenario is been simulated functionally (subjected to geometry limitation) (also see Figure 54 for more detail). ....  | 89  |
| Figure 54. | The HQ gets message after the radar perform detection (also see Figure 51 for simulation summary).....   | 90  |
| Figure 55. | A missile attempts to fly across the coverage of a radar station which has a communication link to an HQ with C2 processors. ....  | 91  |
| Figure 56. | HQ makes decisions when receiving new detection reports (subjected to geometry limitation)(also see Figure 57 for more detail). ....   | 92  |
| Figure 57. | The HQ makes decisions periodically after receiving message from the radar (also see Figure 51 for simulation summary). ....   | 93  |
| Figure 58. | Radar, headquarters, and an AAA cooperate to intercept an incoming missile. ....   | 94  |
| Figure 59. | The beginning of the simulation event-log results show that the AAA can receive information from a sensor and headquarters (subjected to geometry limitation) (also see Figure 61 for more detail). ....             | 95  |
| Figure 61. | The end of simulation event-log results show that the missile is intercepted by the AAA (also see Figure 59 for simulation summary). ....  | 97  |
| Figure 62. | A hostile missile with jammer onboard attempts to penetrate the radar's coverage. ....   | 98  |
| Figure 64. | The missile arrives at its target location under cover from a radar jammer. ...  | 99  |
| Figure 65. | Two successful detections are performed only when the missile is close to the radar (also see Figure 63 for simulation summary). ....  | 100 |

|             |   |     |
|-------------|---|-----|
| Figure 66.  | The radar cannot detect the existing missile even it is within coverage. ....   | 101 |
| Figure 68.  | A hostile communication jammer attempt to block communication among defender units. ....  | 103 |
| Figure 69.  | The communication link from the radar entity to the HQ entity is jammed (subjected to geometry limitation) (also see Figure 71 and Figure 72 for more detail). ....             | 104 |
| Figure 70.  | The missile destroys the warehouse under the cover from a communication jammer. ....  | 104 |
| Figure 71.  | The radar entity attempts to distribute a message every time it detects a target (also see Figure 69 for simulation summary). ....  | 105 |
| Figure 73.  | Binary sensor model contains two states: seen (100%) or not seen (0%). ....   | 110 |
| Figure 74.  | Probabilistic sensor model has a monotonically decreasing probability of detection as the distance increases. ....  | 110 |
| Figure 75.  | Detection probability vs. SNR (from Qi et al. 2008). ....   | 112 |
| Figure 76.  | Three possible examples illustrating the limitations of Sheu’s 4-corners algorithm. ....  | 114 |
| Figure 77.  | Beside the four corners, three additional monitor points are recommended to make sure all points with the highest number of covering sensors are taken into consideration. .... | 115 |
| Figure 79.  | The first grid scan result shows that the $k$ -covered state of all grids is correctly identified. ....   | 117 |
| Figure 80.  | The fifth grid scan result shows the $k$ -covered rate of this scenario is 7.0%~3.07% with an estimation error in 3.03%. ....   | 118 |
| Figure 81.  | The first grid scan shows that mixing sensor types can be estimated correctly. ....   | 120 |
| Figure 82.  | The fifth grid scan result shows the $k$ -covered rate of this scenario is 65.61%~56.84% with an estimation error in 8.77%. ....  | 120 |
| Figure 83.  | AEMF-DES can be extended to include real-time system and time-step approach in combination with DES by loosely coupling TAMs ....   | 124 |
| Figure 84.  | Operation Process Flow Chart of MDSIM ....  | 130 |
| Figure 85.  | In Part tab of MDSIM, part dll files loaded are listed. ....  | 131 |
| Figure 86.  | Entity tab of MDSIM ....  | 132 |
| Figure 87.  | A dialog to select entity affiliation. ....   | 132 |
| Figure 88.  | Entity Add/Edit Form. ....  | 133 |
| Figure 89.  | A new entity named “City” is added into entity table. ....  | 133 |
| Figure 90.  | Add Missile Entity ....   | 134 |
| Figure 91.  | Add two parts with connection between them. ....  | 135 |
| Figure 92.  | Customized Properties Edit Form for a part ....   | 136 |
| Figure 93.  | Add a HQ entity template ....   | 136 |
| Figure 94.  | Add an Anti-Air Artillery entity template ....  | 137 |
| Figure 95.  | Add a Jammer entity template ....   | 137 |
| Figure 96.  | Entity Table with Several Entities ....   | 138 |
| Figure 97.  | Scenario Tab of MDSIM ....  | 138 |
| Figure 98.  | An Example of Scenario Setting ....   | 139 |
| Figure 99.  | A city, a missile, and a jammer have been dragged into map. ....  | 140 |
| Figure 100. | Maneuver Sub-tab ....   | 141 |



|             |  |     |
|-------------|--|-----|
| Figure 101. | Dialog for Selecting a Target.....   | 141 |
| Figure 102. | Dialog for Setting Navigation Point .....  | 141 |
| Figure 103. | An Example of Attacker Setting.....  | 142 |
| Figure 104. | An example of Defender’s Deployment .....  | 142 |
| Figure 105. | In the Electromagnetic Setting, three types of setting are available: Sensor, comm., and Jammer. ....            | 143 |
| Figure 106. | Dialog for setting Communication Setting .....   | 144 |
| Figure 107. | Dialog for Selecting Communication Network .....   | 144 |
| Figure 108. | Several dash lines with arrows appear on map to indicate the communication network among defender entities. .... | 145 |
| Figure 109. | Sensor and jammer settings using default value. ....   | 145 |
| Figure 110. | Adjust customized properties to fit the entity state.....  | 146 |
| Figure 111. | Simulation Tab of MDSIM.....   | 147 |
| Figure 112. | Messages Displayed in Step-by-Step Mode .....  | 148 |
| Figure 113. | Dialog for Run-All-At-Once Mode .....  | 148 |
| Figure 114. | Result Tab of MDSIM .....  | 149 |
| Figure 115. | An Example of Event Detail Message .....   | 150 |
| Figure 116. | Data collected in simulation can be selected to generate a diagram.....  | 151 |
| Figure 117. | The generic diagram form can show data collected in a simulation.....  | 152 |
| Figure 118. | The initial program interface .....  | 154 |
| Figure 119. | The result of grid scan is shown in the right hand side.....   | 155 |
| Figure 120. | The scan result and estimation result are updated after a further division.....                                  | 156 |

## LIST OF TABLES

|           |   |    |
|-----------|---|----|
| Table 1.  | Comparing to typical DES approach, the proposed DES framework enables more dedicate entity relation settings and changeable entity definition after a simulation program is released..... | 6  |
| Table 2:  | Comparison among simulation programs that made by different time advance mechanisms (TAMs).....   | 7  |
| Table 3.  | Mixture of different detection and coverage models can be handled by the proposed $k$ -Coverage Rate algorithm .....  | 9  |
| Table 4.  | DES and the DES framework proposed in Chapter III focus on different aspects of DES simulation program development. ....  | 16 |
| Table 5.  | DES events can be seen as agents' actions. ....   | 35 |
| Table 6.  | Example of event history (excerpted) of sample scenario shows that the startup method initializes all events.....   | 43 |
| Table 7.  | Typical C4ISR Processes in MDW.....   | 46 |
| Table 8.  | MDSIM covers most primary capability of C4ISR in MDW.....   | 47 |
| Table 9.  | Abstract methods of the Sensor class that must be implemented by Concrete subclasses. ....  | 52 |
| Table 10. | The abstract method that must be implemented by subclasses.....   | 55 |
| Table 11. | The abstract method of the abstract Headquarters class and must be implemented by concrete subclass.....  | 59 |
| Table 12. | Sheridan Levels of Authority for Decision and Action Selection. (after Sheridan, 2002).....   | 60 |
| Table 13. | The abstract methods of the abstract Weapon class must be implemented by any concrete subclass. ....  | 63 |
| Table 14. | Entity symbols used covers 3 battle dimensions and 2 affiliations. ....   | 67 |
| Table 15. | Different line/curve patterns are used to represent paths, coverage, and links. The color can be either cyan or red depending on its friendly or hostile.....                             | 67 |
| Table 16. | Two experiment series demonstrate the feasibility and potential of AEMF-DES and MDSIM .....   | 75 |
| Table 17. | Two sub experiments are used to answer questions about AEMF-DES to improve the limitation of released DES programs.....   | 76 |
| Table 18. | Attributes of entities in experiment 1a.....  | 78 |
| Table 19. | A new PulseRadarPart is added to the warehouse entity to testify the entity customizability.....  | 83 |
| Table 20. | Summary of five sub experiments to incrementally introduce C4ISR processes regarding MDW.....   | 86 |
| Table 21. | Attributes of entities in experiment 2a.....  | 88 |
| Table 22. | A Headquarters part is added to experiment 2b.....  | 92 |
| Table 23. | An AAA entity with a Weapon part and a CommNode part is added into experiment 2c.....   | 95 |
| Table 24. | The previous missile entity is replaced by a missile entity with a jammer part. ....  | 98 |

|           |  |     |
|-----------|--|-----|
| Table 25. | A communication jammer is added to block the communication between the radar entity and the HQ entity..... | 103 |
| Table 26. | Three sensors are used to illustrate challenges for current algorithm.....                                 | 116 |
| Table 27. | Four different types of sensors are used in experiment 2. ....   | 118 |
| Table 28. | File structure of the calculation program .....  | 153 |
| Table 29. | Attributes of sensors .....  | 155 |

## LIST OF ACRONYMS AND ABBREVIATIONS

|          |  |
|----------|--|
| AAA      | Anti-Air Artillery   |
| ABM      | Agent-Based Model  |
| BPS      | Bits per Second  |
| C4ISR    | Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance |
| AEMF-DES | Adjustable and Extensible Modeling Framework DES   |
| DES      | Discrete Event Simulation  |
| DTS      | Discrete-Time Simulation   |
| EM       | Electromagnetic  |
| EMW      | Electromagnetic Wave   |
| FTA      | Fundamental Theme Agent  |
| FOM      | Federation Object Model  |
| HLA      | High-Level Architecture  |
| MALD     | Miniature Air-Launched Decoy   |
| MDSIM    | Missile Defense Simulation   |
| MDW      | Missile Defense Warfare  |
| OOD      | Object-Oriented Design   |
| PES      | Part-Entity-Scenario data structure  |
| PRF      | Pulse Repetition Frequency   |
| $P_k$    | Probability of Kill  |
| RCS      | Radar-Cross Section  |
| RT       | Real Time  |
| RTI      | Run-Time Infrastructure  |
| SAM      | Surface-to-Air Missile   |
| SEAD     | Suppression of Enemy Air Defenses  |
| SNR      | Signal-to-Noise Ratio  |
| SOP      | Standard Operating Procedures  |
| TAM      | Time-Advance Mechanism   |
| VV&A     | Verification, Validation, and Accreditation  |

THIS PAGE INTENTIONALLY LEFT BLANK

## **ACKNOWLEDGMENTS**

This dissertation would not be complete without the assistance of many people. First are my dear thesis advisors, Professor Don Brutzman and Professor Phillip Pace, whose guidance and knowledge were essential in this research.

In addition, I'd like to thank Professor Tom Lucas who taught me the concept in Combat Modeling, Professor Xiaoping Yun who showed me how to model agent movement, Professor Deborah Goshorn who shared with me her knowledge of systems engineering, and Professor Arnold Buss who demonstrated the concept of DES.

Of course, I also would like to thank my lovely wife, Fang-Yin Hsu, for the motivation she gives me and the love she shows me every day.

THIS PAGE INTENTIONALLY LEFT BLANK

# I. INTRODUCTION

## A. OVERVIEW OF THIS DISSERTATION

This research is inspired by considering a time-advance mechanism (TAM) for a simulation of C4ISR in Missile Defense Warfare (MDW). While Discrete Event Simulation (DES) can provide accurate time estimation and fast simulation speed, models utilizing it often suffer limitations in their flexibility of scenario design and customizability of entity definition. These limitations are due to the scenario-oriented design style and can happen in both DES and the time-step approach. A new DES framework, namely Adjustable and Extensible Modeling Framework DES (AEMF-DES), is proposed to improve these limitations by embedding primary principles of a topical theme. Since the primary principles describe the nature of events related to the topical theme, such as transaction rules of events and algorithms to determine occurrence time, a released simulation program that is designed according to AEMF-DES principles can generate events for a user-defined scenario with custom entities.

An example program that simulates C4ISR progress in MDW is developed in this research to demonstrate the feasibility of AEMF-DES and explore its potential. The simulation (MDSIM) program architecture covers typical C4ISR processes related to MDW such as missile detection, message delivering, and decision making, and this architecture can help in defense deployment or attacker strategy efficiency estimation.

In addition to the program architecture of C4ISR simulation, the algorithm to estimate sensor deployment efficiency is also of interest. The  $k$ -covered rate problem which calculate the proportion of an area that covered by  $k$  sensors is studied in this research. Current estimation algorithms can handle a binary-omnidirectional sensor model which is simple to design but only provides limited simulation fidelity. This research introduces a probabilistic and sectorial model and proposes a new calculation algorithm to accommodate it. Another program has been developed for experiments and shows that the new calculation algorithm is feasible.



## **B. MOTIVATING PROBLEMS**

C4ISR describes the information flow of military operations from initial generation to the final utilization. Systems involving in C4ISR processes such as radars, communication networks, and weapons work closely to form a system of systems to perform joined and complex operations in battlefield. Failure of any one may make the entire system fail and cause loss of property or even human lives. To manage the complexity and tremendous uncertainty, simulation techniques are critical to estimate whether forces have been organized efficiently.

Missile Defense Warfare (MDW) is studied in this research to explore more capabilities in C4ISR simulation. The processes involved may at least include missile detection by sensors, data communication among units, intelligence analysis and decision making at headquarters, guidance of anti-air weapons, and computer assistance within all these processes.

### **1. Time-Step Approach: Advantages and Limitations**

When simulating C4ISR in Missile Defense warfare, one of the important issues is choosing a suitable time-advance mechanism (TAM). The time-advance mechanism controls the time variable that determines when the model state is evaluated and updated. The time-step approach and DES are the two main mechanisms that have been used.

Time-step approach has been well known and popularly used. This approach advances simulation time in fixed intervals and updates all model states at each increment. For example, when a radar station attempts to detect an incoming missile the position of all combat units in battlefield must be recalculated at every time step. Then the distance between the radar station and the missile is measured. Detection occurs if the missile is within the range of the radar station. Model states are updated periodically in small time increments until the end of simulation. The advantages of the time-step approach are it is easy to understand and simple to develop simulation programs.

However, considering simulations in regard to the C4ISR processes, we found the time-step approach might not be fully applicable. First, while time intervals are applied to incremental time values, model states are updated according to these “rounded” time

values. Since many important simulation state variables (such as location of a moving missile, progress percentage of a communication, or accumulated probability of detection) are related to the time variables, any inaccurate time value may cause serious aggregated simulation error or miss some important states that only happen in a specific time value. Second, the time-step approach is relatively slow to compute since it has to go through all time steps and states during the simulation. Due to the need to simulate many C4ISR processes like communication with high fidelity, a short time step is needed. Adapting such short time intervals in the time-step approach would slow down simulation speed. This often results in difficulty for simulations requiring a large number of runs to generate meaningful statistical results. Since the statistical result is a quite important tool to help us in managing the uncertainty of the battlefield, this is an undesirable and serious shortcoming (Lucas 2000; Buss and Al Rowaei 2010; Al Rowaei et al. 2011).

In addition the multi-rate time-step approach has been proposed to improve overall simulation computational speed. By adjusting the time intervals, multi-rate time-step simulation saves much time in trivial model state updates. However the short time events might happen during all a simulation of C4ISR in MDW, for example, defender's routine communication. Multi-rate time-step approach may not be able to benefit all possible scenarios.

## **2. Discrete Event Simulation (DES): Advantages and Limitations**

Another type of time-advance mechanism is Discrete Event Simulation (DES) which has been long used in military simulation (Ong et al. 2010; Seo et al. 2012). DES does advance time incrementally but goes through important events only. Simulation analysts first analyze all events related to a scenario and model event responses within the program in development. While a simulation is run, customized parameters and possible random variables are then provided to the models. The value of the time variable jumps ahead to the precise time of each event sequentially. Therefore, DES can be significantly faster than the corresponding time-step approach and can provide more rapidly computable time estimations. These properties make it a possible option for simulating C4ISR in MDW (Buss and Sánchez 2005).

However, applying DES also brings some challenges even when its advantages elegantly satisfy our needs. First, since the DES simulation program is designed for events occurring in a given scenario, most of time the simulation program cannot provide great flexibility for related scenarios. For example, a simple scenario contains a radar station intercepting an invasive missile. If a succeeding scenario requires a second radar station or two new missiles coming from other direction, the source code for the underlying simulation program might need to be modified to accommodate the possible new events. This is due to that fact that a typical DES framework utilizes a scenario-oriented design style so that program refactoring is needed while the topical scenario is modified and the relevant events are changed. Note that this limitation does not only happen to the DES programs but also to the time-step programs with scenario-oriented design style.

Another limitation, which also can exist in a time-step simulation program, is that the entity customizability cannot satisfy the need to model new combat units. Military technology progresses every day. New combat units can be improved designs of an existing model or an entirely new design including several functions. In a military simulation containing invasive missiles, radar, and an anti-air weapon system, end users might need to model a new destroyer consisting of two radar systems and two anti-air weapons. Although the concepts of radar detection and weapon attack are the same as those in previous scenarios, such program modification is unavoidable.

### **3. New Framework Proposal**

It is been observed that for the many possible scenarios found under the same theme, the majority share similar primary principles inherited from the theme topics. For instance, in sea surface engagement, ships might be attacked by torpedoes, and aircraft can be taken down by anti-air artillery. These principles do not only apply to one or a few specific scenarios but to all scenarios belonging to this simulation theme. This research intends to explore the possibility of surmounting the limitations mentioned earlier by letting the computer utilizing the primary principles to determine events happen in a user-defined scenario with custom entities principles. Once a theme has been studied and a simulation program created to model it, the simulation program can provide better

flexibility in related scenarios, and customizability in entities definition without any additional human analysis.

## **C. RESEARCH OBJECTIVES**

First, this research introduces the limitations of the current DES approach followed by a discussion of a strategy to embed primary principles into a simulation program. The techniques to accommodate related issues such as principle embedding and arbitrary scenario description are mentioned. All the findings are combined as a new DES framework that can be used to guide DES simulation program development.

Second, given the framework proposed, a new C4ISR model of Missile Defense Warfare (MDW) is developed to test the feasibility of the new DES framework and find out its potential. Model concept and program architecture are introduced as well as experiment simulations.

Last, based on the interest in sensor deployment strategy, this research also studies the  $k$ -coverage rate problem which is used as a metric of the optimization degree of sensor deployment. The current calculation approach can only handle binary-omnidirectional sensor models and cannot provide high simulation fidelity. An improved calculation method to accommodate probabilistic-sectorial sensor models is proposed.

## **D. CONTRIBUTIONS**

### **1. An Improved DES Framework: Adjustable-and-Extensible-Modeling-Framework DES**

A new DES framework is proposed to help surmount some limitations of current DES models. Its advantages include improved flexibility in scenario design and entity modeling. Therefore, this new DES framework has been named Adjustable and Extensible Modeling Framework DES (AEMF-DES). Compare to typical DES models, AEMF-DES enable more dedicate entity relation settings within a scenario and changeable entity definition after a simulation program is released (shown in Table 1), which result in two properties:

- *Increased flexibility in related scenarios*: Typical DES software is dedicated to a specific scenario and has restricted flexibility in related scenarios. By

using AEMF-DES, end users can define scenarios that belong to the topical theme without additional analyst effort in event analysis and programming. The flexibility of a DES program is improved.

- *Improved customizability in entity modeling:* To model new types of combat units, AEMF-DES allows end users to define custom entities with an arbitrary combination of fundamental functions and to use them in simulation. This ability again needs no additional effort from analysts.

Table 1. Comparing to typical DES approach, the proposed DES framework enables more dedicate entity relation settings and changeable entity definition after a simulation program is released

| Flexibility   | Typical DES approach   | Proposed DES Framework   |
|---|--|--|
| Change simple value parameters, such as motion speed                              | Yes, value parameters can be adjusted easily.  | Yes, value parameters can be adjusted easily.  |
| Change entity numbers, such as number of missiles                                 | Yes. Objects can be instanced from classes.  | Yes. Objects can be instanced from classes.  |
| Change relations among entities, such missile targets, to form a related scenario | No, since classes in a released program is designed for a specific scenario.                                     | Yes, properties of each object can be adjusted to satisfy difference scenario definition.  |
| Change entity definition  | No, once the events of the target scenario are analyzed, entity functions are determined to support simulations. | Extensible. By using different parts, end users can define their own entities. New part files are possible by additional programing. |

## 2. A Simulation Architecture for C4ISR in Missile Defense Warfare

A simulation program named Missile Defense Simulation (MDSIM) is also developed in this research to demonstrate the feasibility of the AEMF-DES architecture. This program is based on AEMF-DES and can simulate C4ISR processes in MDW as follows:

- The detection actions of sensors

- The communication among data links
- The information storing and sorting of intelligence processes
- The decision-making time delay of a headquarters
- The probability of kill for weapon systems
- The noise jamming to sensors and communication performance

Table 2 compares simulation programs that follow different time advance mechanism (TAM). Inheriting from typical DES framework, MDSIM has both relatively faster simulation speed and precise time estimation. Moreover, end users can design new scenarios without any human analysis work or program refactoring. New entities can also be modeled without professional programming techniques. It becomes possible for end users to evolve scenario and system design based on these simulation flexibilities. Note that all the flexibilities are limited to the theme that MDSIM was design for.

Table 2: Comparison among simulation programs that made by different time advance mechanism (TAMs)

|                                  | Simulation programs made by the time-step approach | Simulation programs made by typical DES framework | MDSIM made by the proposed DES framework |
|----------------------------------|--|---|--|
| Relatively easy to understand    | √  |   |  |
| Relatively easy to implement     | √  |   |  |
| Relatively fast simulation speed |  | √   | √  |
| Precise time estimation          |  | √   | √  |
| Flexible in scenario definition  |  |   | √  |
| Flexible in entity definition    |  |   | √  |

In addition, a new concept is also triggered by this simulation architecture. When simulating complex physical phenomena, such as electromagnetic (EM) wave

propagation, second-order or higher-order mathematical equations might needed to be applied. Due to the need of the time-step oriented Euler method to estimate such equations, pure DES framework cannot provide enough simulation fidelity. The agent-based modeling (ABM) architecture of AEMF-DES explores a possible solution that not only uses DES to simulate C4ISR events but also contains the time-step approach to provide high simulation fidelity. A potential hybrid architecture to serve CISR simulation is shown in Figure 1. For these events that don't use Euler method, for example, a random time delay DES framework is sufficient for accurate time estimation and fast simulation speed. For the other events that need the Euler method to generate high-fidelity values, the time-step approach may be necessary.

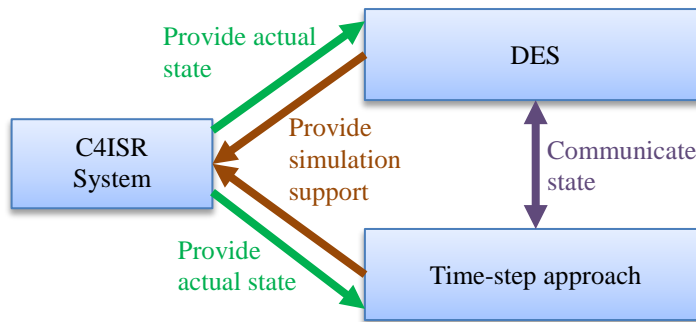


Figure 1. Potential future hybrid architecture where DES can cooperate with time-step methods to support C4ISR simulation.

### 3. An Improved $k$ -Coverage Rate Algorithm

The  $k$ -coverage rate problem evaluates sensor deployment efficiency by considering the coverage state of a given region. Current calculation algorithms can only handle a binary-omnidirectional sensor model which cannot offer sufficient simulation fidelity. This research proposes an improved calculation algorithm that can accommodate following properties which are incapable to current algorithm.

- *Probabilistic Sensor Model*: For most type of sensors in real world, the probability of detection is probabilistic since natural noise exists to affect light, sound, electromagnetic (EM) wave, or any other energy formation that people use in sensing.
- *Sectorial Sensor Model*: For the interests in particular direction, some sensors

only monitor sectorial area for better sensing efficiency.

Table 3 shows that by using the new calculation algorithm, a mixture of different detection and coverage models can be handled at the same time. The small sensor coverage that might cause errors in the current algorithm is also managed.

Table 3. Mixture of different detection and coverage models can be handled by the proposed  $k$ -Coverage Rate algorithm

| Sensor Models            | Current Algorithm | Proposed Algorithm |
|--------------------------|-------------------|--------------------|
| Binary Detection         | √                 | √                  |
| Probabilistic Detection  |                   | √                  |
| Omnidirectional Coverage | √                 | √                  |
| Sectorial Coverage       |                   | √                  |
| Small Coverage           |                   | √                  |

## E. NOMENCLATURE

While some terms represent multiple meanings when people use them in everyday life, they may only stand for specific concepts in this dissertation. For the reader's convenience, these terms are introduced here.

- *Theme*: A topic domain of simulation. For example, Missile Defense Warfare (MDW), Anti-Submarine Warfare (ASW), etc.
- *Theme Primary Principle*: The general rules of a given theme. For instance, in Missile Defense Warfare, sensors are assumed to be able to detect missiles and missiles can attack targets.
- *Unit*: An independent unit in the battlefield. For example, a ship, a tank, an aircraft.
- *Component*: A functional assembly of a unit. For example, a radar system or an anti-air gun of a destroyer.
- *Scenario*: A specific simulation setting of a theme. For example, a scenario that simulates a radar station in position A to detect an aircraft in position B.
-



- *Entity*: A model of a unit in a scenario, such as a ship model, a vehicle model, etc.
- *Part*: A model of a component in an entity. For instance, a ship entity could contain communication parts, weapon parts, and more.
- *Agent*: A computer model that has its own data storage and interacts with other agents. In this dissertation, entities and parts are both agents.

The relation among theme, scenario, entity, and part are shown in Figure 1. A theme can contain various scenarios which share the same primary principles and include multiple entities. These entities also can contain multiple parts within them. Note that both entities and parts can be regarded as agents which keep their own data and interact with other agents within the same scenario.

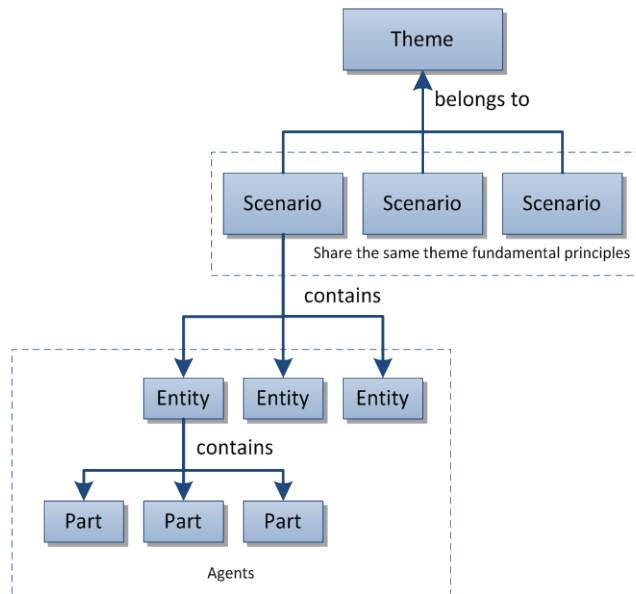


Figure 2. A theme can contain multiple scenarios which include entities and parts.

## F. EXAMPLE SCENARIO

In this dissertation many simulation techniques are discussed and illustrated. An example scenario shown in Figure 3 is used in many places in this dissertation to help readers understand these concepts. In this example, an invasive missile attempts to fly through a defender area in which a radar station and an anti-air weapon are deployed. The

radar station can detect the existence of the missile within coverage then commands the anti-air weapon to perform an intercept operation. All processes in this simulation may contain several probabilistic variables, such as probability of detection, detection error, and probability of kill, that ultimately affect the simulation results.

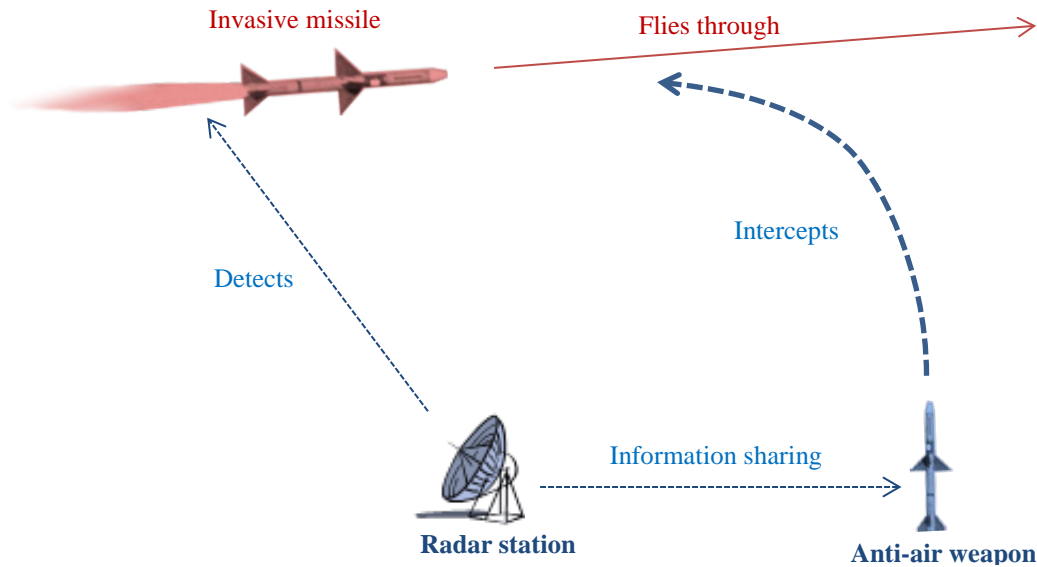


Figure 3. In this simple example scenario, a radar station and an anti-air weapon system cooperate to intercept an invasive missile.

Note that for simplification, all defender entities in the scenario are assumed static, only invasive missiles can maneuver during the simulation. This assumption makes it easier to evaluate the time when a missile enters a radar's coverage.

## G. DISSERTATION ORGANIZATION

Several important related works which support and inspire this research are discussed in Chapter II. These works provide a general introduction to time-advance mechanisms of simulation, the C4ISR concept for military application, and the  $k$ -coverage rate problem, etc. In Chapter III the proposed DES framework, AEMF-DES, is introduced. The discussion begins with the limitations of the current DES approach and the strategy for embedding theme primary principles. The challenge of this notion and our proposed solution are then disclosed. The AEMF-DES framework is then shown. Chapter IV contains a discussion of a simulation program, MDSIM, for simulating

C4ISR processes in MDW. The background and basic concept of the theme is introduced, followed by a series of development steps. Several experiments illustrating how MDSIM that adopted AEMF-DES can create and schedule events automatically are detailed in Chapter V. Functions of C4ISR are then demonstrated. To go through the findings of the  $k$ -coverage rate problem, Chapter VI opens with the introduction of the current calculation approach. The probabilistic-sectorial sensor model is then discussed and followed by the introduction to a new calculation to accommodate probabilistic-sectorial sensor models. Conclusions and recommendations for future research are summarized in Chapter VII.

## II. RELATED WORK

The research in this dissertation has benefited from many works. This chapter first introduces the time-advance mechanisms (TAMs), discrete event system specification (DEVS), real-time (RT) simulation and control, physical fidelity, agent-based model (ABM), loose coupling, and high-level architecture (HLA) to support the improved DES framework. Command, control, computer, communication, intelligence, surveillance, and reconnaissance (C4ISR) and related simulations are then mentioned for the demonstrative program. Verification, validation, and accreditation (VV&A) is discussed for possible future work of the program. At last work related to the current  $k$ -coverage rate problem is presented.

### A. TIME-ADVANCE MECHANISMS (TAMs)

States of models in simulation are changed when the time value is changed. Many time-advance mechanisms have been used in handling the time variable, and each of them has individual advantages and shortcomings. In fact, Delaney and Vaccari have claimed that a time variable is necessary in any digital computer simulation program (Delaney & Vaccari 1989). Although various types of time-advance mechanisms are identified by different researchers, such as the time/event/process driven approach by Galluscio and time-step/discrete-event/time-parallel architecture by Marquez, they can all be included in either the time-step approach or the DES approach (Al Rowaei et al. 2011; Fishwick 2007).

#### 1. Time-Step Approach: Discrete-Time Simulation (DTS)

The time-step approach is also known as Discrete Time Simulation (DTS). The time value in this simulation advances in a fixed interval and the states of models are changed according to the time value. The occurrence of possible interactions or other events in simulation are examined repeatedly to ensure that simulated actions happen on time. The state of models being changed in each time step is illustrated in Figure 3.

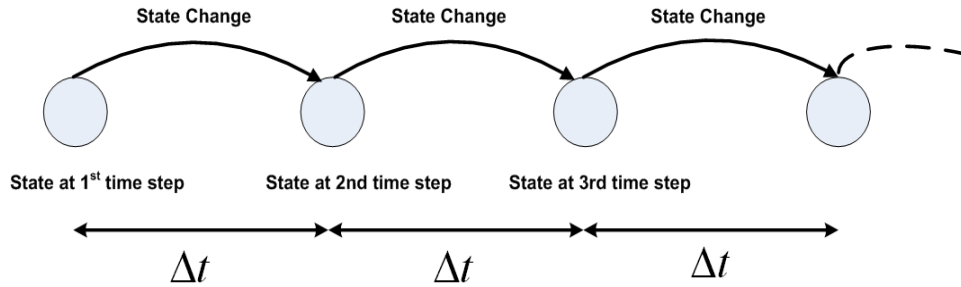


Figure 4. Step-wise executions of the time-step approach (from Al Rowaei et al. 2011).

The time-step approach is the main approach used by most simulations since it has been known for a long time and its intuitive internal algorithm makes it easy for engineers to develop programs using it. On the other hand, it also has the drawback that the time value advances incrementally, which requires a great deal of computational effort and can cause unexpected errors when this “imprecise” time is used to update the state of models (Cellier and Kofman 2006).

## 2. Discrete-Event Simulation (DES)

DES is a relatively new and rare choice of time mechanism. Like the time-step approach, DES also has a time variable to control the simulation progress. This value is determined by the time or duration of predicted critical events, rather than an increment that is repeated. In the scenario shown in Figure 3, such events may include the following:

- The missile starts to move.
- The missile enters the radar station’s coverage.
- The radar station detects the missile.
- The anti-air weapon intercepts the missile.

The events listed above are executed one by one with some random variables, such as detection time or maneuver bias, taken into consideration. If the attempt to detect fails, the detection error is too high, or the anti-aircraft weapon does not intercept the missile successfully, the entire intercept operation fails (Cassandras, 1993; Fishman, 2001; Law & Kelton, 2000).

Comparing to the time step approach, time value in DES doesn’t increment using fixed value but is determined by the prediction of event occurrence. There are many

advantages raises from this approach. First, unlike time step approach, the estimated times in DES of events are not rounded to time intervals. Time and duration values are precisely calculated without time interval round off. Second, since computer only spends calculation resources in critical events, simulation speed of a DES is relatively faster than the speed of a corresponding simulation program using the time step approach. However, DES is also considered not as intuitive and so tends to cost more in technicians training and program development.

### 3. Comparison Summary

Many studies have been dedicated to comparing simulation performance using the time-step approach and DES. Although different research focuses on different properties of the time-step approach or DES, all of them agree that the choice of a time-advance mechanism depends on the problem described and the goal of the pursued simulation (Al Rowaei et al. 2011). In this research, this work is not going to extend the discussion of the comparison. Rather, it aims to push the boundary of the current DES framework, as discussed in Chapter III.

## B. DISCRETE EVENT SYSTEM SPECIFICATION (DEVS)

A discrete event system can be represented by DEVS which is a modular and hierarchical formalism. The DEVS formalism was first introduced in Bernard P. Zeigler's book in 1976. In DEVS, input events, output events, states and related functions of a model are all included within an atomic DEVS formation shown in ( 1 ) (Zeigler et al. 2000).

$$M = \langle X, Y, S, ta, \delta_{ext}, \delta_{int}, \lambda \rangle \quad (1)$$

$M$  : An atomic DEVS model

$X$  : The set of input events

$Y$  : The set of output events

$S$  : The set of states

$ta$  : The function to determine the life span of a state

$\delta_{ext}$  : The function determines how an input change model state

$\delta_{int}$  : The function determines how states change internally

$\lambda$  : The function determines how states triggers an output event.

Comparing to the DES framework proposed in Chapter III, DEVS focuses on a standard representation for a DES design which benefits can document management and developers communication, while the proposed framework gives recommendations in programming architecture such as class architecture. (Shown in Table 4)

Table 4. DES and the DES framework proposed in Chapter III focus on different aspects of DES simulation program development.

|   | DEVS | Proposed DES Framework |
|---|------|------------------------|
| Formalism of a DES design                       | √    |                        |
| Benefits of Consistent Documentation Management | √    |                        |
| Program Architecture                            |      | √                      |
| Class Structure Framework                       |      | √                      |
| Needs Dedicated DES Engine                      |      | √                      |

DEVS is useful in describing a DES system design while training is needed to let users be familiar with the formalism. In this research, all DES concepts are depicted in traditional block diagram style for user's convenience.

### C. REAL-TIME SIMULATION AND CONTROL

Real-time (RT) simulation is a simulation whose internal clock speed is as fast as that of the real world. For example, if it takes two minutes for a missile to fly from A to B, in simulation the time needed for such a maneuver is two minutes. RT simulation matches the human sense of time domain and is helpful in military personnel training such as vehicle driving and pilot training. (Dufour et al. 2010)

### **1. Real-Time Control**

Real-time systems can be implemented by the time step approach. For example, a robot aviation simulator updates model state for every period of time that the robot motion control is calculated. The data is sampled at each time step and then used to generate instruction for following actions. When building an RT simulation with the time-step approach, two possible conditions can happen. First, if the model is simple enough so that computer can execute it faster than a wall clock, the simulation program has to be intentionally slowed down to match the real time step. Second, when there are too many possible details that can be modeled causing slower execution, the model has to be simplified to catch up with the real time step (Asatani, Sugimoto, & Okutomi, 2011; Lei & Hongzhou, 2012; Ouellette, Wierckx, & McLaren, 2008). Reduced fidelity is undesired in real world simulators.

### **2. Real-Time Simulation with Time-Step Approach**

Real time system can be implemented by the time-step approach. For example, the aviation simulator updates model state every period of time or the robot motion control calculation. The data is sampled in each time step and then used to generate instruction for following actions. When building an RT simulation with the time-step approach, two possible conditions could happen. First, if the model is simple enough so that computer can execute it faster than a wall clock, the simulation program has to intentionally slow down to match the real time step. Second, while there are too many possible details that could be modeled causing slower execution, the model has to be simplified to catch up with the real time step (Ouellette et al. 2008; Lei and Hongzhou 2012; Asatani et al. 2011).

### **3. RT Simulation with DES Approach**

Unlike the time-step approach, DES determines time and effect of what will happen before incrementing a time variable. This property makes it faster and more precise, and can be used in manufacturing process control and scheduling. When applying this approach in RT simulation, in addition to the core event queue process, a heartbeat timer is usually used to update model states periodically and to generate



renewed information for users. This timer is also used to postpone future events until the time expires (Sung-Ho Jee 1999; Tavakoli et al. 2008).

#### **D. PHYSICAL FIDELITY**

In the fields of modeling and simulation, fidelity stands for the degree that a model correctly reproduces a real world object and sometimes also been described as "degree of similarity." Physical fidelity is determined by different modeling approaches including algebraic formulas, differential formulas, probabilistic functions, and tabular parameters. The choice of a suitable model approach depends on the nature of the phenomena and finally determines how much precision the simulation can achieve. (Donnell, 2008; Gross, 1999)

Physical activities of C4ISR in MDW simulation often consist of entity motion, electromagnetic (EM) wave propagation, and digital data calculation. These activities assemble multiple C4ISR processes to support a defender's interception operation. Depending on the resolution required, these physical activities can be simulated with detailed algorithms to achieve good fidelity or else just focus on the C4ISR process results to have a more comprehensive estimation of all physical activities related to the processes.

#### **E. AGENT-BASED MODEL (ABM)**

The Agent-Based Model (ABM) is a computer modeling technique used in simulating the actions of individual units. These units are regarded as independent agents and can receive/sense outside information independently, while the information collected can be stored individually and possibly guide future actions. The ABM technique consists of game theory, complex systems, emergence, computational sociology, multi-agent systems, and evolutionary programming. Monte Carlo Methods are often used when these models contain random factors (Niazi and Hussain 2011).

ABM is usually applied in simulating the complex phenomena that contains multiple agents (for example, a combat scenario with numerous soldiers). Due to its flexibility in simulating an individual agent's actions and interactions among them, phenomena with various agents can be predicted more meaningfully than before.

Complex large-scale systems, such as social networks, have been modeled by ABM successfully (Zhang Chengbin et al. 2010; Sanchez and Lucas 2002; Wiedemann 2010; Kotenko 2007).

#### **F. LOOSE COUPLING**

Loose coupling refers to a system in which components have limited or no prior knowledge of each other. Interaction among these components is achieved by third-party software mediators or standard interfaces. This approach makes it easier in system development, modification, and maintenance (Hock-koon and Oussalah 2010).

In this research, the proposed program architecture exhibits a typical loose coupling style and enables future development of its parts.

#### **G. HIGH-LEVEL ARCHITECTURE (HLA) SIMULATION**

The high-level architecture (HLA) is a common framework standard to support distributed simulation systems. In HLA, a simulation is named federation that can include multiple participants in network. Federation Object Model (FOM) represents all possible agents within the simulation. Simulation entities, also known as federates, can interact with each other via a corresponding Run-Time Infrastructure (RTI). Ambassador patterns are used when federates communication with RTI (Kuhl, Weatherly, & Dahmann, 1999; U.S. Department of Defense, 2001).

Time in a simulation moves forward only when all federates have finished current event and agree to go forth. The increment can be adjusted to match the wall clock so that the simulation runs as a real time system or just go to the time of next event, like a typical DES system (Kuhl et al., 1999).

#### **H. COMMAND, CONTROL, COMPUTER, COMMUNICATION, INTELLIGENCE, SURVEILLANCE, AND RECONNAISSANCE (C4ISR)**

Generally speaking, C4ISR represents the elements of the military information process. According to the *Department of Defense Dictionary of Military and Associated Terms*, C4ISR stands for command, control, computer, communication, intelligence, surveillance, and reconnaissance. It is interesting to compare C4ISR to another military

model, the Observation-Orientation-Decision-Action (OODA) loop model shown in Figure 5, C4ISR derives from the previous C2 concept (command and control) and denotes the modern military information architecture. This conceptual model is only applicable in battlefield but also can be used in many other issues such as Counter-terrorism (Dimarogonas, 2004; Jing Liu, Ai-Min Luo, & Xue-Shan Luo, 2009; Tajwer & Shamsi, 2010).

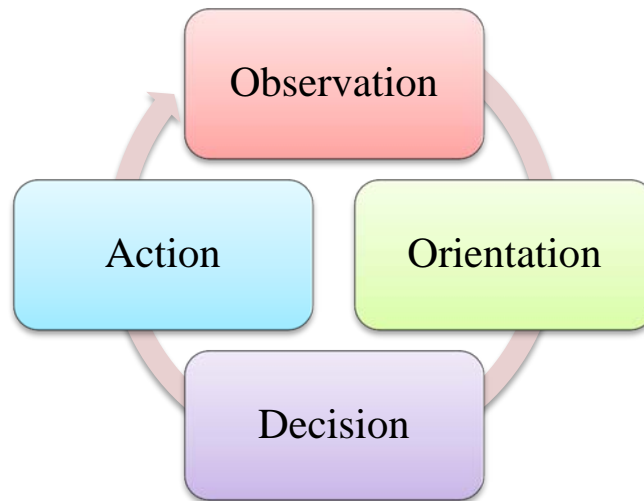


Figure 5. OODA Loop contains 4 processes in a military operation.

Figure 6 illustrates a simplified C4ISR information flow which is derived from the concept of network centric warfare that also gets benefits from simulation analysis (Stein, Garska, & McIndoo, 2000; Yuanzheng Ge, Xiaogang Qiu, & Kedi Huang, 2010). Comparing to Figure 7, the C4ISR processes are performed in three steps, sensing, decision, and action. Information is first sensed by the surveillance and reconnaissance processes. After intelligence analysis, the commander makes a decision and controls subordinates to act by giving commands. Note that all information exchange is done by communication systems, and the computer benefits the digital information process in each step. Of further interest is that the sense-decide-act cycle is commonly used in robotics, and nearly identified to the OODA loop commonly used by human-decision processes.

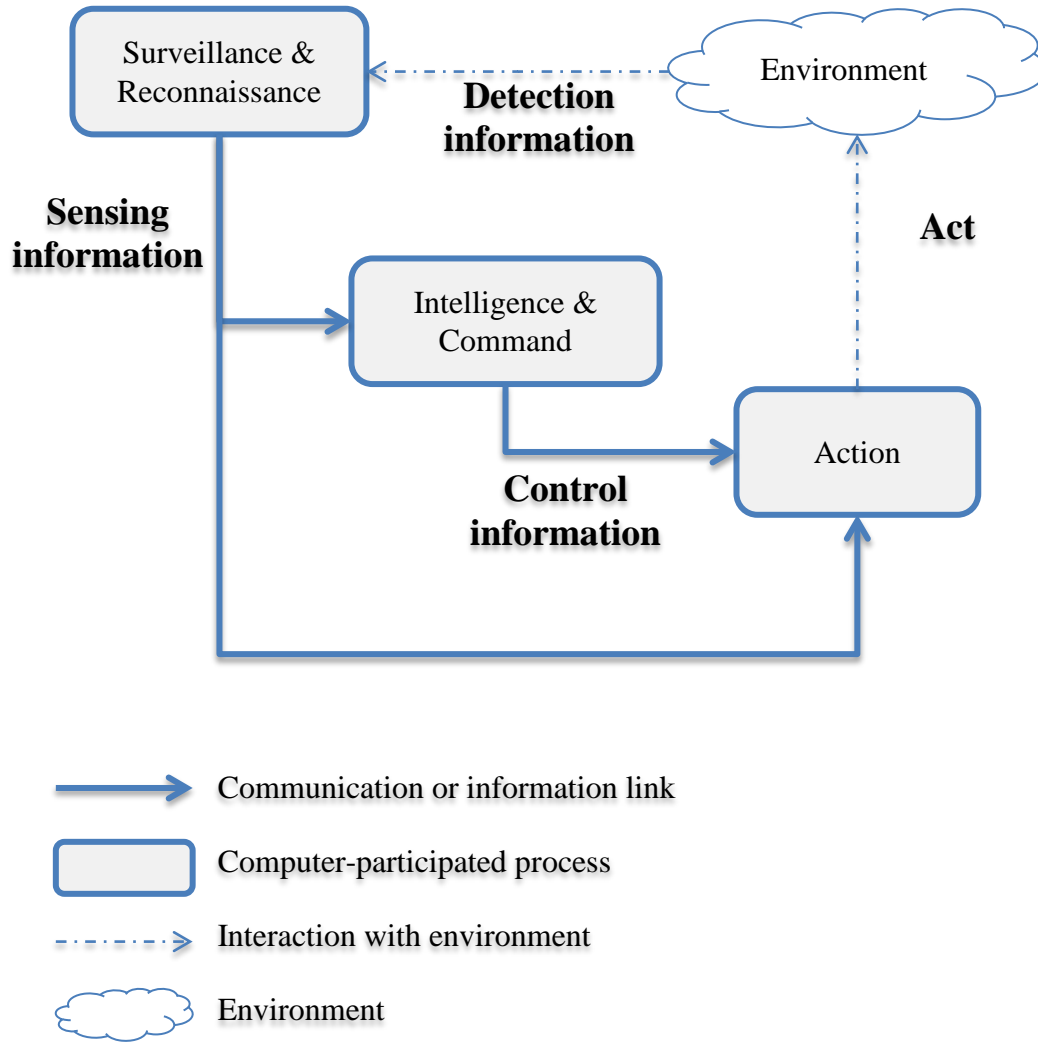


Figure 6. Simplified information flow in each C4ISR process contains its generation, delivering, and utilization.

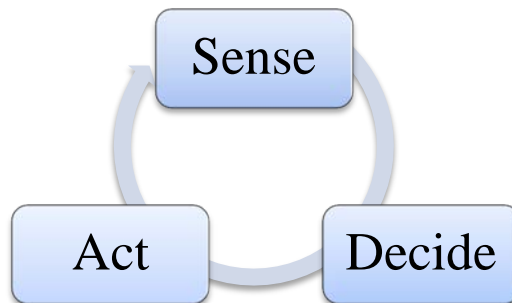


Figure 7. A rational behavior includes sense, decide, and act.

Another similar concept, C5ISR, is also recommended. It includes combat systems additional to C4ISR since more and more digital weapons have been applied in battlefield, and the information process within them is also part of the whole information process architecture. In this research, those concepts are used in Chapter IV, the development of sample simulation software. Although this research focuses on the traditional C4ISR domain, a simple weapon model is also provided to reflect this fact.

## **I. C4ISR-RELATED SIMULATIONS**

Many academic researches and commercial products are dedicated to C4ISR related simulations. This research completes a comprehensive literal study in both computer and electronic engineer area such as:

- Simulation interoperability Workshop (SIW)  
<http://www.sisostds.org>
- Association for Computing Machinery (ACM)  
<http://www.acm.org>
- Command and Control Research Program (CCRT)  
<http://www.dodccrp.org>
- Autonomous Agents and Multi-Agent Systems (AAMAS)  
<http://aamas2012.upv.es> ; <http://aamas2013.cs.umn.edu>
- Institute of Electrical and Electronics Engineers (IEEE).  
<http://www.ieee.org>

While some of these societies focus on physical phenomena and support individual system development, the others cover system of systems (SoS) level analysis. Recent findings of interest are listed below.

- Moffat develops a conceptual model of C2 to allow assessment of cost-benefit across the defense budget (Moffat 2007).
- FEKO: FEKO and COMSOL are commercial products that can solve a wide range of problems that benefits engineer in designing sensor and communication systems (EM Software & Systems 2013; COMSOL 2013).
- Nguyen & Yip proposed a stochastic model which estimates the integrated system effectiveness by summarizing the probability of sensors, command & control, and weapons (Nguyen and Yip 2010).
- Fusano showed his simulation which contains multiple agents to represent combat unit in battlefield (Fusano et al. 2011).

- Zhang discussed a simulation architecture which utilizing the Agent-Based technique to generate the C4ISR effectiveness measurement. Rather than standing for an independent combat unit in battlefield, each agent in this simulation architecture represents an element of C4ISR of an affiliation (Zhang Ying-chao et al. 2010).
- Leskiw introduced a research that models the worldwide internet communication for military application using DES to demonstrate the feasibility of producing a Global Network Simulator (GNS) for Air Force C4ISR. (Leskiw et al. 1999)
- Xie uses Petri Net to model C2 federations' states in a HLA simulation (Likuan Xie et al. 2009).
- Giampapa extended the OneSAF Testbed Baseline (OTB) modeling and simulation environment with agent-based functions to simulation C4ISR actions in battlefield (Giampapa et al. 2005).
- Jiang discussed the effectors of fleet collaborative decision making based on simulation experiments (Xin Jiang et al. 2010)
- Wang analyzed the paralysis conditions of combat system based on complex network (Wang Chang-chun et al. 2010).
- Wei studied the efficiency evaluation method of simulation systems by referring to BP neural network (Chu Wei 2011).
- Sun researched the runtime support platform for the net-centric simulation (Sun Li-yang et al. 2010).
- Shaulov studied the performance robustness for optical gigabit LAN in military application based on simulations (Shaulov and Patel 2007).
- Wood used HLA federation to demonstrate the NATO live, virtual and constructive concept in defense power (Clive Wood et al. 2008).
- Chang implemented computer simulation to explore the employment methods of utilizing ISR systems (Chang 2005).
- Donnelly proposed using the system of system (SoS) common integration approach to bridge live and simulation domain related to military agents (Donnelly 2009).
- Brooks proposed a design approach to implement the implicit traffic flows of the C4ISR communication in a simulation environment (Brooks et al. 2006).
- Davis observed puzzling results from high resolution simulation for military exploratory analysis and then proposed another model for wider circumstances (Davis et al. 2000).
- Bozlu discussed a conceptual modeling methodology to help in the

communication between users and developers when development a system from a conceptual model to design phase (Bozlu and Demirörs 2008).

- Bai introduced an application using object-based petri nets for C4ISR architecture simulation validation (Xiaohui Bai 2008; Xiao-Hui Bai 2008).
- Ferenci proposed a model using HLA to integrate U.S. naval simulation system and another DoD simulation system(Ferenci et al. 2004)
- To improve the information security of c4ISR, Jiang used Gateway-Proxy to design a C4ISR simultion system (Jiang Jin-long et al. 2004).
- Santiago recommended using high Power performance computer (HPC) to add Live C4ISR environment simulations (Santiago 2009).
- Davis explored the issue about using high resolution simulation to calibrate multi-resolution analysis model regarding military operations (Davis et al. 2000).
- Moen discussed the message flow of the real-time simulation for C4ISR ((Moen and Pullen 2005)).
- Wei introduced the issues about integrating simulation systems and C4ISR systems (Chu Wei 2011).
- Mao studied the simulation and evaluation experimental method for C4ISR systems (Shaojie Mao et al. 2008)
- Su introduced the approach to use simulation to help in C4ISR system design (Wei Su et al. 2006).
- Ma proposed using simulation-based training support environment to promote the training with C4ISR systems (Ma Wei-bing and Zhu Yi-fan 2011).
- Wang discussed the method of using UML model to perform C4ISR requirement analysis (Wang Cong et al. 2010).
- Huang explored the C4ISR from the viewpoint of military electronic information system and discussed the entity modeling (Huang Zhonghua and Li Yinlin 2010).
- Yang proposed models for C4ISR and used indexed system theory to research the relation among C4ISR entities (Yang Juniang et al. 2009).

Among these related C4ISR simulation researches, except for some unimplemented research that explore conceptual issues such as design methodology or cost management, the other use time advance mechanism to support their simulation algorithms. Comparing the number of the two kinds of time advanced mechanism—DES and the time-step approach, most use time-step approach and few use DES while none covers both methods

at the same time. This discloses that very little overlap exists between DES and the time step approach to support high-fidelity analysis in real world domain such as C4ISR.

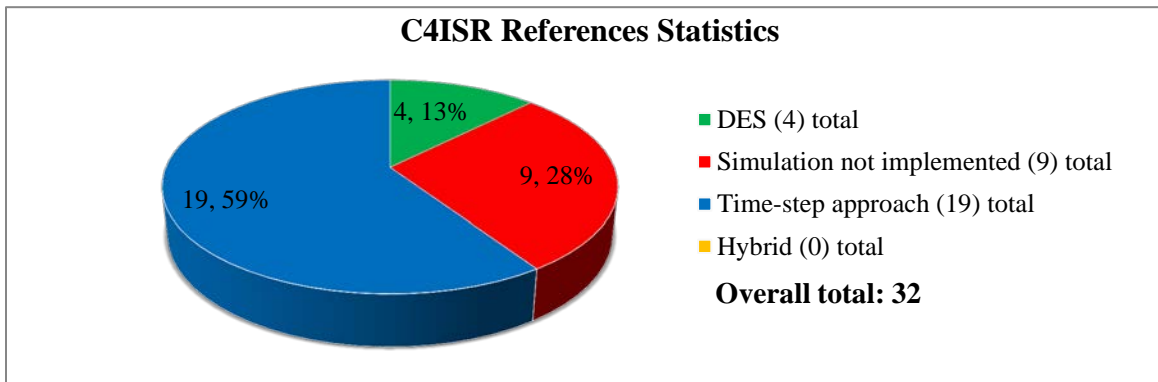


Figure 8. The ratio of time-advance mechanism (TAM) methodologies used in C4ISR related research examined in this dissertation.

The C4ISR simulation program developed in this research does not attempt to provide competitive C4SIR simulation capability such as advanced analysis algorithms or conceptual models, but demonstrate those differences brought by adopting the proposed DES framework. These differences can improve the flexibility of released simulation programs and enable end users to explore design insights further.

## J. VERIFICATION, VALIDATION AND ACCREDITATION (VV&A)

VV&A is a series of steps that are used to estimation the correctness, capability, and usability of a simulation. It contains three interrelated processes: Verification, Validation, and Accreditation. According to the official definition (U.S. Department of Defense 2013):

- Verification – the process of determining that a model implementation and its associated data accurately represent the developer's conceptual description and specifications
- Validation – the process of determining the degree to which a model and its associated data provide an accurate representation of the real world from the perspective of the intended uses of the model
- Accreditation – the official certification that a model, simulation, or federation of models and simulations and its associated data is acceptable for use for a specific purpose



In a viewpoint of semantics, verification is more about mathematics and to prove whether the equation/conceptual idea is corrected modeled in the code and validation is about science and engineering to prove the M&S can represent the phenomena in real word. Accreditation is about engineering and engineering management to refer to the result of verification & validation to determine if the code results support intended use (Rouche 2009; U.S. Department of Defense 2011).

**K.  $k$ -COVERAGE RATE PROBLEM FOR SENSOR EFFECTIVENESS**

When sensors are deployed in a given region for reconnaissance and surveillance purposes, the sensors’ location and orientation determine the efficiency of a deployment. The covered rate is a metric used to estimate the proportion of a given region that is monitored by at least  $k$  sensors. This ratio value can be used not only to understand the efficiency of a sensor network but also as a reference to help in future enforcement design. (Shen, Chen, & Sun, 2006; Sheu, Chang, & Chen, 2008)

Although the definition of the covered state considers all points within it, a simple calculation algorithm is needed to accommodate the fact that it is impossible to infinitely evaluate all points within a given region. A calculation algorithm, called the grid scan algorithm, was proposed by Shen in 2006 to solve the  $k$ -coverage rate problem. The Grid scan algorithm divides the target region into small grids and then identifies whether a grid is covered or not by whether the center of the grid is covered. The covered rate of the region is then available by calculating the proportion of the covered grids area over the target region area. Steps are listed as follows:

- First separate the target region into regular sized grids.
- Identify each grid state:

| Covered State    | Criterion  |
|------------------|--|
| $k$ -covered     | If its center is covered by at least $k$ sensors |
| not $k$ -covered | Else   |

- Then the  $k$ -coverage rate can be calculated as follows:

$$k\text{-coverage rate} = \sum_{g \in M} |g| / |A|$$

Where  $M$  denotes the set of all  $k$ -covered grids and  $A$  stands for the area of the whole region.

The estimation accuracy depends on the size of the grids. Larger grid size implies lower accuracy but faster calculation speed. On the other hand smaller grid size results in better accuracy and takes more time (Shen et al. 2006).

Shen’s algorithm is simple but slow when high-resolution estimation is needed. To pursue better efficiency, in 2008 Sheu proposed an improved algorithm which can increase the calculation speed with the same accuracy standard. Generally his approach can be summarized by the following (Sheu et al. 2008):

- First separate the target region into regular sized grids.
- Identify each grid state:

| Covered State          | Criterion   |
|------------------------|---|
| $k$ -fully-covered     | If all corners are covered by at least $k$ sensors          |
| $k$ -partially-covered | If only partial corners are covered by at least $k$ sensors |
| not $k$ -covered       | Else  |

- Evaluation error is calculated as:

$$\text{evaluation error} = \sum_{g \in P} |g| / |A|$$

Where  $P$  denotes the set of all  $k$ -partially-covered grids and  $A$  stands for the area of the whole region.

- If the evaluation error is acceptable, for example less than 10%, then go to the next step; otherwise divide the  $k$ -partially-covered grids into smaller grids and evaluate repeatedly until the evaluation error is acceptable.
- Then the  $k$ -coverage rate can be calculated as follows:

$$k\text{-coverage rate} = \sum_{g \in F} |g| / |A|$$

Where  $F$  denotes the set of all  $k$ -fully-covered grids and  $A$  stands for the area of the whole region.

An example of Sheu’s approach is shown in Figure 9. Three sensors 1~3 are deployed in a square region. When  $k$  is given by 2, the first grid scan given a  $k$ -covered value = 25% with estimation error = 37.5%. If the error is not acceptable, further grid division can be performed on the grids  $k$ -partially-covered to get more precise estimation. Note that because only the  $k$ -partially-covered grids are divided into smaller grids for

better estimation, the calculation speed is significantly improved in contrast to the original grid scan method proposed by Shen.

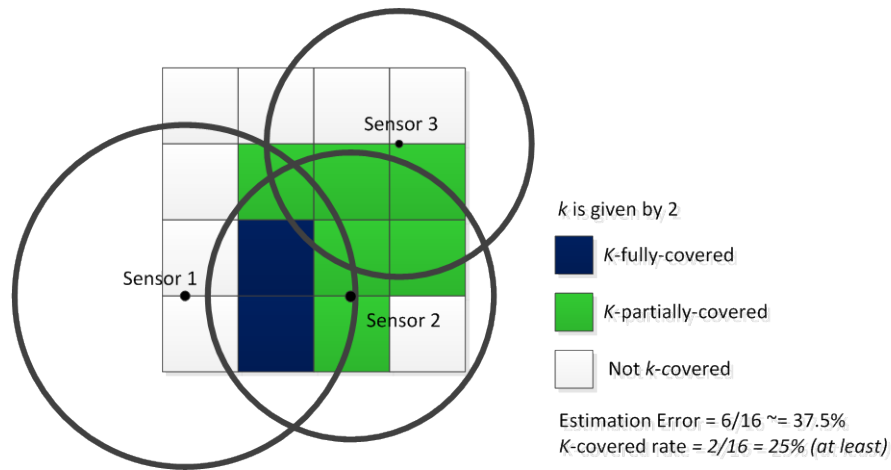


Figure 9. Improved grid scan method separates grids into three states (after Sheu et al. 2008).

Comparing these two grid scan approaches, while Shen only uses the grid center to identify two possible grid states: covered or not, Sheu uses four grid corners to provide three possible grid states and can estimate error range. The different utilizations of limited monitor points of a grid provide different estimation fidelity.

Although Sheu's method can provide estimation error range and improved calculation speed in solving the related  $k$ -coverage rate problem, his algorithm only accommodates simple binary-omnidirectional sensor models which might not provide enough resolution in modeling modern sensor systems. This research follows Sheu's work and develops an improved algorithm which can adopt a probabilistic-sectorial sensor model that provides better modeling fidelity to practical application.

### **III. AN IMPROVED DES FRAMEWORK: ADJUSTABLE AND EXTENSIBLE MODELING FRAMEWORK DES (AEMF-DES)**

This chapter introduces an improved DES framework that attempts to improve some limitations of the current DES approach. First, common limitations are discussed. Then the need and possible strategy for proposing this framework are mentioned. Finally, the propose framework is shown.

#### **A. LIMITATIONS OF CURRENT DES APPROACHES**

Consider a simulation of C4ISR in MDW; two properties of DES are preferred due to (1) its precise time estimation that allow the time-sensitive parameters to be analyzed accurately, and (2) relatively fast simulation speed leading to the possibility of adopting the Monte Carlo method to accommodate the uncertainty of the battlefield. However, because typical DES is scenario-oriented design which requires human effort for event analysis and programming in each individual scenario, the application of DES is limited. Note that time-step simulation program suffers same limitations while using similar design style.

The workflow of a typical DES simulation program is shown in Figure 10. The workflow begins with the confirmation of a client's need for the simulation (for example, a request to estimate the efficiency of a missile defense system for a coastal city). The theme of the simulation, such as MDW, is then known intuitively. After clients plot a scenario that can generate simulation results related to simulation needs, the simulation analysts analyze the scenario concept and identify the events that is included as well as the occurrence of their interactions. Once the detail of scenario is known, simulation specialists then develop a program and perform the simulation. After performing a simulation, the simulation is terminated if the simulation need has been satisfied. Otherwise, additional scenarios which could contain new agent models or different scenario settings are input to pursue a more relevant or plentiful simulation result to satisfy the client's need.

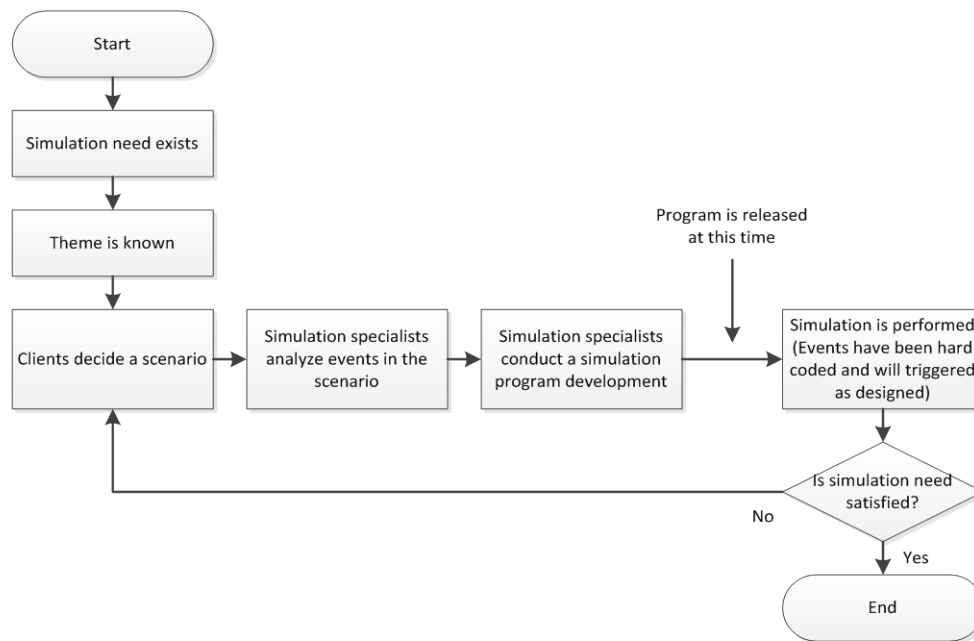


Figure 10. Typical DES framework requiring analysts’ effort in events analysis of each scenario.

During the workflow described above, simulation specialists use their sharp perception to analyze events and their interactions within the given scenario. More specifically, by studying the concept of the scenario, specialists gain a sense of the theme’s primary principles and then use these principles as the criteria for judging the events creation and interaction. For example, in the sample, specialists (1) build a theme with primary principles in mind, such as missiles can maneuver and radar can detect missiles; then (2) they apply these rules into the scenario setting and identify what events should happen and how they should interact. This fact prompts an idea for an improved DES framework:

*Rather than focus on a specific scenario, simulation specialists develop a simulation program for a theme based on its primary principles. As a result, simulation programs can manage events automatically by referring to those primary principles.*

A recommendation for a DES workflow using the proposed framework is illustrated in Figure 11. After the simulation need and the target theme are identified, simulation

specialists study the concept of a given theme and analyze its primary principles and subsequently embed these primary principles into a simulation program. Once the program has been completed, clients can request any scenarios under the given theme and can revise setting as well without any additional analysis or programming (Buss and Sanchez 2002).

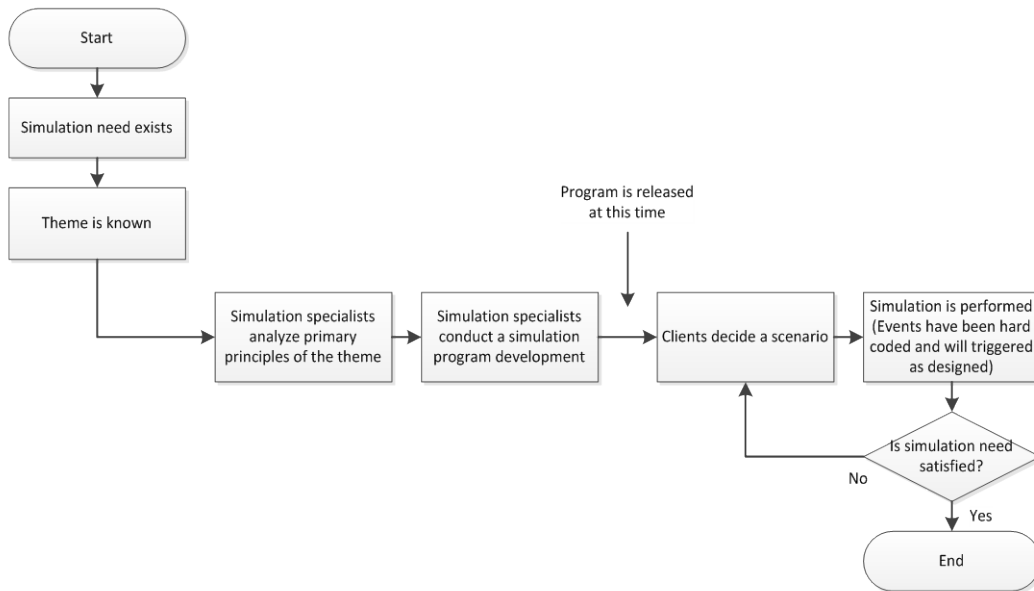


Figure 11. Improved DES framework allows the computer to manage events automatically.

## B. DESIGN CHALLENGE AND SOLUTION STRATEGY

To implement the idea described in the previous section, from the software engineering terms, the following questions must be answered:

- How can a simulation program know the primary principles?
- How can a simulation program understand all possible scenarios offered by end users?
- How can a simulation program apply the primary principles to execute a simulation to generate a result?

This research first comes up with a basic strategy:

- Create a set of classes as the fundamental unit types of a given theme and define the primary principles of a given theme as the methods of these classes.

- Provide a data structure type for which users can define various scenarios by assembling the fundamental unit type classes.
- Build a DES engine that can read the scenario data structure and execute the methods as events.

This approach seems to provide a capability to restore the primary principles in predefined classes and also lets users define their scenarios with unit objects related to the target theme. While the scenario data structure is loaded into the DES engine, the program can perform a simulation by executing a method predefined in the fundamental unit classes. However, although the predefined classes are useful in handling primary principles and can help users build various scenarios, limited classes cannot satisfy daily renewal inventions. Therefore, another strategy is proposed:

- Create a set of classes as the fundamental agent types of a given theme and define the primary principles of a theme as the methods of these classes. These agents include an Entity class as a platform and multiple part classes as functional components of an entity object.
- Allow end users to model a new combat unit by assembling an entity object along with several part objects in it. For example, a destroyer (entity) may contain several radar and weapon systems (parts).
- Provide a data structure that users can use to define various scenarios. A scenario consists of multiple entities which contain multiple parts to represent functions of the entity.
- Build a DES engine that can read the scenario data structure and execute the methods as events.

With this strategy, end users are allowed to define custom scenarios and entities based on extensible parts. The flexibility of current DES approach is significantly improved.

### **C. ADJUSTABLE-AND-EXTENSIBLE-MODELING-FRAMEWORK DES (AEMF-DES)**

Following the discussion in the previous section, an improved DES framework named Adjustable and Extensible Modeling Framework DES (AEMF-DES) is proposed. This name comes from the basic concept that the released programs conducted by AEMF-DES have better flexibility in user-defined scenarios and more extensibility in

custom entities. This section discusses the concept behind AEMF-DES and its architecture in a simulation program.

## **1. Concept of AEMF-DES**

As mentioned previously, rather than focusing on an individual scenario, this work pays attention to themes. This research proposes an improved DES architecture, the Adjustable and Extensible Modeling Framework DES (AEMF-DES) framework, which is named due to its adjustability in scenario and extensibility in custom part classes.

AEMF-DES includes several components:

- First, a general descriptive model contains a set of classes named Theme Fundamental Agent (FTA), which describes the primary principles of a given theme.
- Second, a data structure named Part-Entity-Scenario (PES) contains a set of objects instanced from FTA and can represent a specific scenario defined by end users.
- Finally, also essential, a DES engine that can interpret PES data and refer to FTA to manage events automatically. After finishing the simulation, the simulation result should be generated by the DES engine.

As shown in Figure 12, AEMF-DES are contributed from many works including Agent-Based Modeling, Object-Orientation programming, and the typical DES framework.



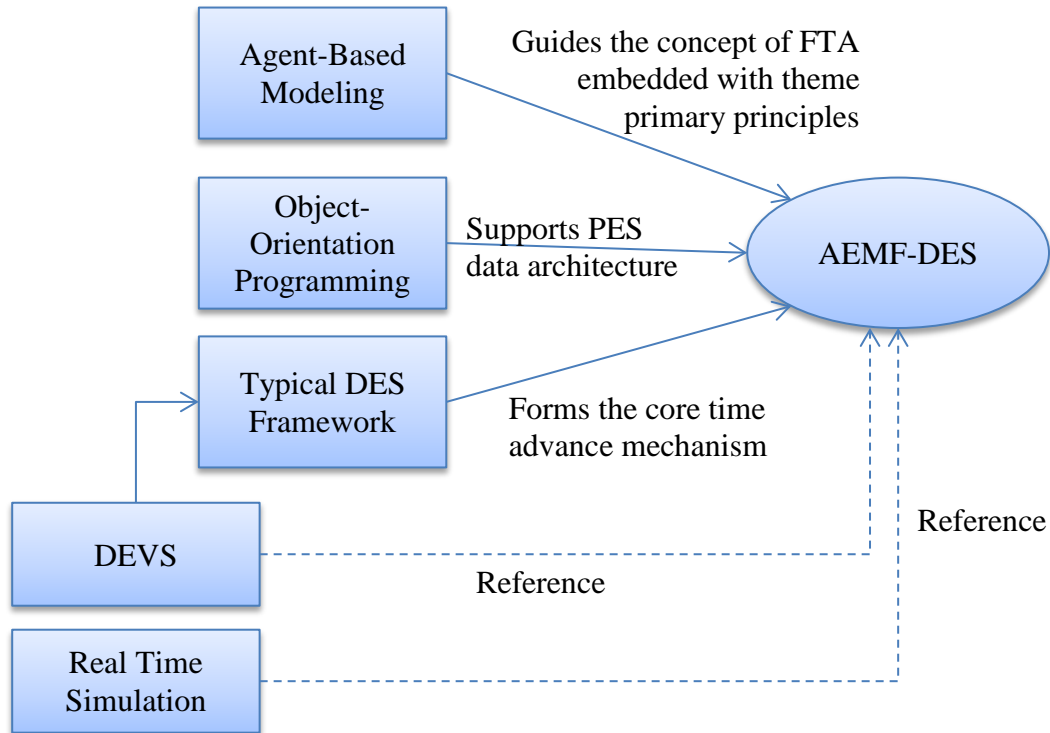


Figure 12. AEMF-DES elements are contributed from many works.

*a. Fundamental Theme Agents (FTAs)*

It has been noticed that events of military simulation can be related to an entity's individual actions or the interactions among multiple entities. For example, in an MDW theme infinite events could exist from infinite possible scenarios, while all of these events can be related to combat unit actions. In other words, although it is impossible for the events of a given theme to be explicitly listed and managed, it is possible to derive them from the related unit type of the theme (see Figure 13). This research proposes using a set of agents to represent these unit types. High simulation fidelity may need more agent types, while less simulation fidelity might result with fewer agent types.

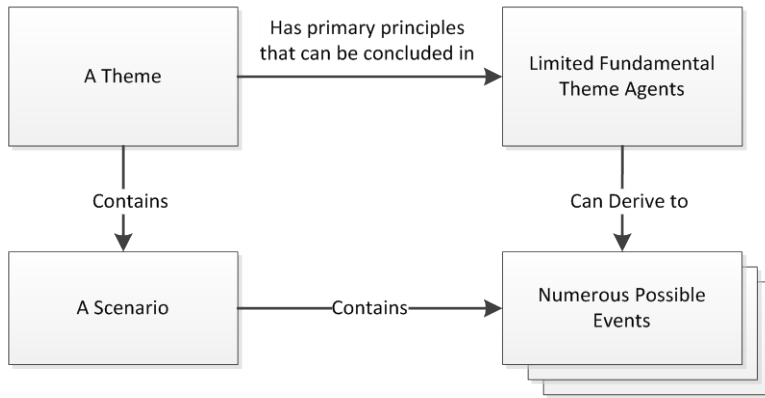


Figure 13. Fundamental Theme Agents (FTAs) are concluded from the target theme and can derive possible events for a specific scenario which must belong to the same theme.

Fundamental Theme Agents (FTAs) is a set of classes of these fundamental agents, and it describes the theme’s primary principles. Several example events from the missile-radar-weapon scenario are listed in Table 5. These are typical DES events and may be hardcoded in a simulation program to support simulation execution. While they can be related to agents’ methods, agents with methods can then be used to represent the primary principles of a given theme.

Table 5. DES events can be seen as agents’ actions.

| DES Events    | Description                                 | Related Agent Activities |                      |
|---------------|---|--------------------------|----------------------|
|               |   | Agents                   | Method               |
| Move          | A missile moves to a position               | Missile                  | Move                 |
| EnterCoverage | A missile enters a radar station’s coverage | Radar station            | AddTargetInCoverage  |
| Detection     | A radar station detects a target            | Radar station            | Detection            |
| ExitCoverage  | A missile leaves a radar station’s coverage | Radar station            | LostTargetInCoverage |

However, as mentioned earlier, having a few basic agents is not sufficient to fulfill the need of this research since using only limited agent types to represent all possible combat units in a battlefield is unsatisfactory. Three components of agents exist in FTA to solve these issues (see Figure 14).

- First, a general entity class represents platforms of combat units such as a battleships and vehicles. An entity can move along with parts on it. If an entity is destroyed, then all the parts on it are invalid. A combat unit is defined only with an entity object while additional functions such as detection and attack are provided by additional part objects attached to this entity.
- Second, several abstract part type classes are used to define the basic types of parts such as sensors or weapons. These part type classes do not represent any specific equipment but stand for a general component type which is related to a kind of function, such as detection or attack. They contain concrete methods to represent primary principles and abstract methods to let sub-classes decide their own detailed algorithm. For example, an abstract sensor class can contain a concrete Detect method which describes how a sensor interacts with a target. Further an abstract GetDetectionProbability method is left for a subclass to provide the detailed algorithm.
- Last, the unlimited concrete part classes that inherit from the abstract part type classes and must implement those abstract methods to model specific equipment such as an AN/SPY-1 naval radar system or Phalanx Close-In Weapon system (CIWS).

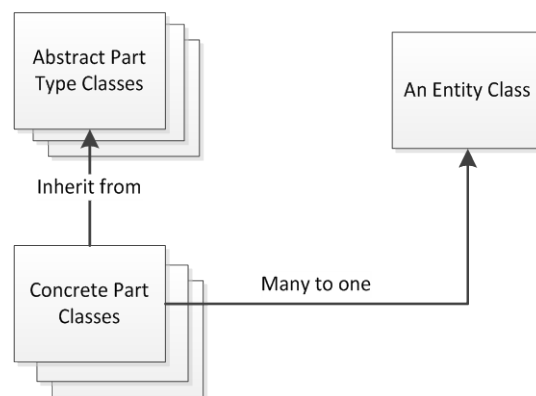


Figure 14. Entity can contain multiple concrete part classes and represent a combat unit while abstract part type classes define the general behavior of each part type.

For instance, the theme of the missile-radar-weapon example scenario mentioned in Chapter I is “simple missile defense.” Similar scenarios that belong to this same theme include the following:

- A pulse radar station, a phase radar station and an anti-aircraft weapon attempt to intercept the missile.
- An anti-aircraft destroyer which contains a radar system and a AAA system attempts to intercept the missile.
- Multiple missiles traverse the defense area.

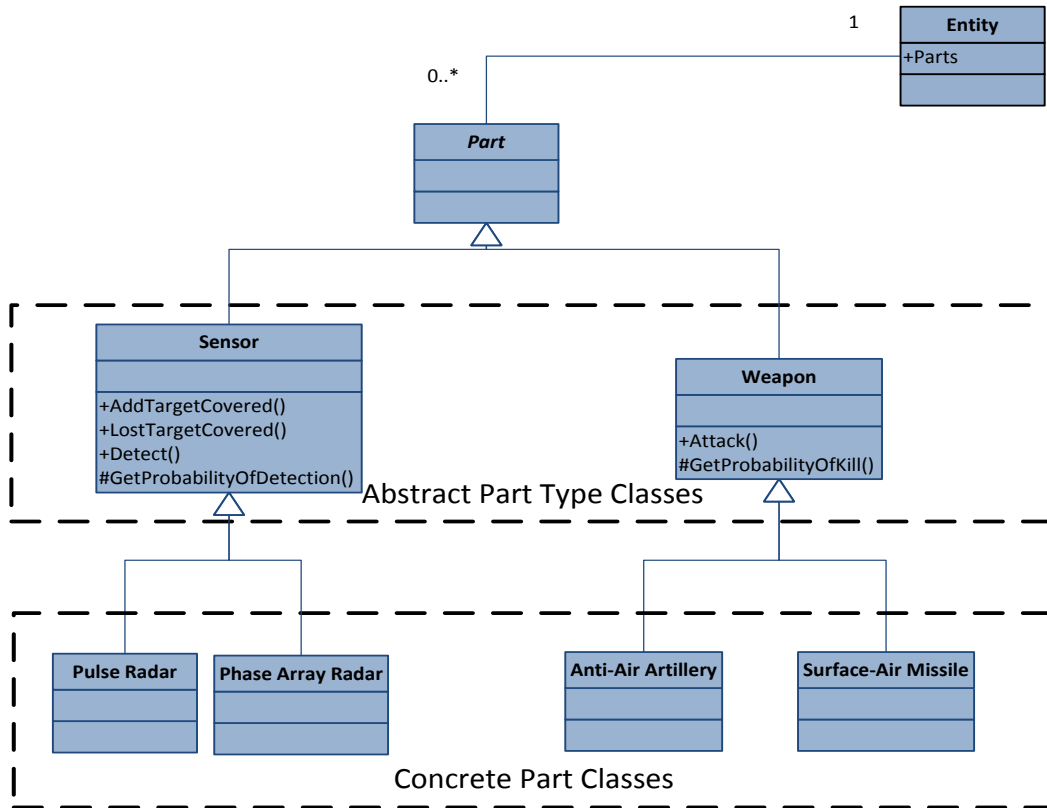
Although all these scenarios result in different event analyses, they share the same primary principles of the target theme:

- Invasive missiles can maneuver.
- Defender sensors can detect missiles.
- Defender weapons can attack missiles.

By observing the primary principles, the FTA of the simple missile defense can be concluded as follows:

- An entity class as platform of missile, radar station, and weapon
- Two abstract part type classes: Sensor and Weapon
- Four concrete part classes inherit from two abstract classes to model specific equipment

A simplified class diagram for the FTA of the simple-missile-defense theme is shown in Figure 15. The entity class is in the upper right and can contain multiple parts. Abstract part type classes in the middle area consist of only sensors and weapons, which define the general method of these two basic part types, while the specific method (`GetProbabilityOfDetection` and `GetProbabilityOfKill`) are implemented by sub-classes. Concrete part classes are located at the bottom of the diagram. Pulse radar and phase-array radar both inherit from the sensor but provide different algorithms to the `GetProbabilityOfDetection` method to reflect their individual properties. The same situation exists in the weapon part type class. Anti-aircraft artillery and surface-to-air missiles offer a different algorithm in probabilities of kill when they are both derived from the abstract weapon part class.



+: concrete method  
#: abstract method

Figure 15. A class diagram of the sample scenario shows three portions of FTA: entity class, abstract part type classes, and concrete part classes

### b. Part-Entity-Scenario Architecture (PES)

Since AEMF-DES focuses on the primary principles of a given theme and provides flexibility for end users to define scenarios to fit different needs, a general data structure is needed that can describe all possible scenarios of the given theme along with custom entities. The Part-Entity-Scenario Architecture (PES) is proposed as such a data structure. It contains a scenario object along with multiple entity objects that belong to either the attacker or defender. These entity objects also contain their own part objects which offer them different functions.

The class diagram illustrated in Figure 15 is extended to support user-defined scenarios, as shown in Figure 16. Here the Scenario class is added and associated

with the Entity in one-to-many relation. When a user defines a scenario, first a scenario object is instantiated in the Scenario class. Then entity objects along with part objects are created by referring to the Entity and Concrete part classes. The attributes of these objects can be modified to fit the scenario design. Compared to FTA which is built as a set of classes during the program development for “templates,” PES is the object hierarchy that uses these “templates” to describe their own scenarios.

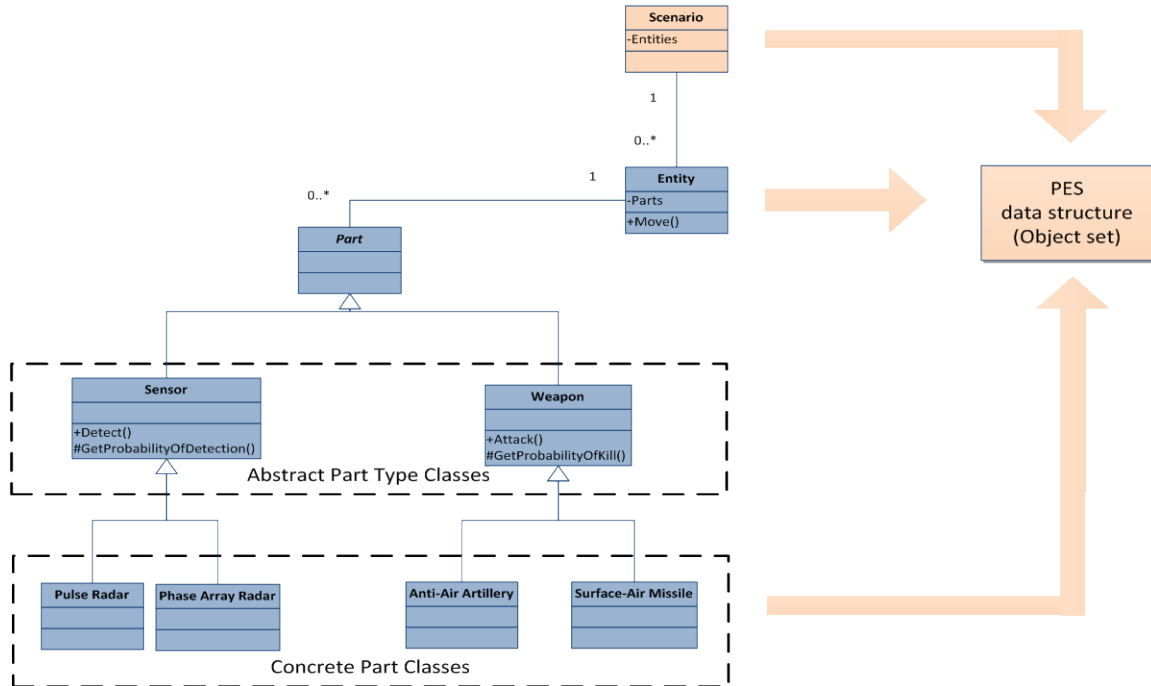


Figure 16. The Scenario class, the Entity class, and Concrete Part classes support PES data structure to represent user-defined scenarios.

For instance, the PES of the sample scenario can be seen in Figure 17. A scenario object contains one attacker entity, the missile, and two defender entities, the radar station and the anti-aircraft weapon. While the missile entity only performs the “Move” action during the simulation, the radar station and anti-aircraft weapon have detection and attack ability via the part objects attached to them.

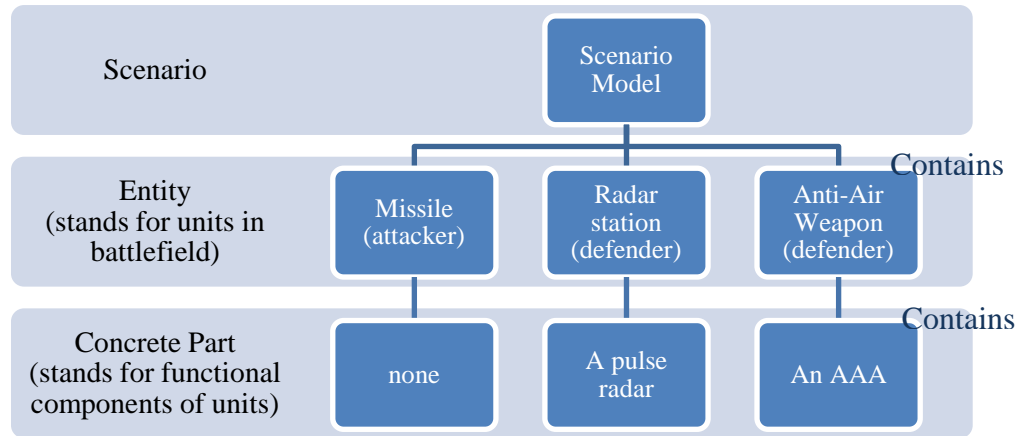


Figure 17. The PES of the sample scenario shows that a scenario contains multiple entities which also contain multiple parts to offer different functions.

Therefore, by using PES, users' custom entities can be defined by assembling different part objects to an entity's objects. Then a user's scenario of the given theme can be described in an object hierarchy that contains a scenario object and multiple entities along with attached part objects.

### c. *DES Engine*

A typical DES framework has a software engine to handle event execution. As shown in Figure 18 after a simulation is started, initial events are scheduled. Then the engine will check to see if the end condition is satisfied. If so, the simulation is stopped; otherwise the model state transition is performed, and the next event is executed. Subsequently the engine checks the end condition again and repeats the same process until the end condition is satisfied.

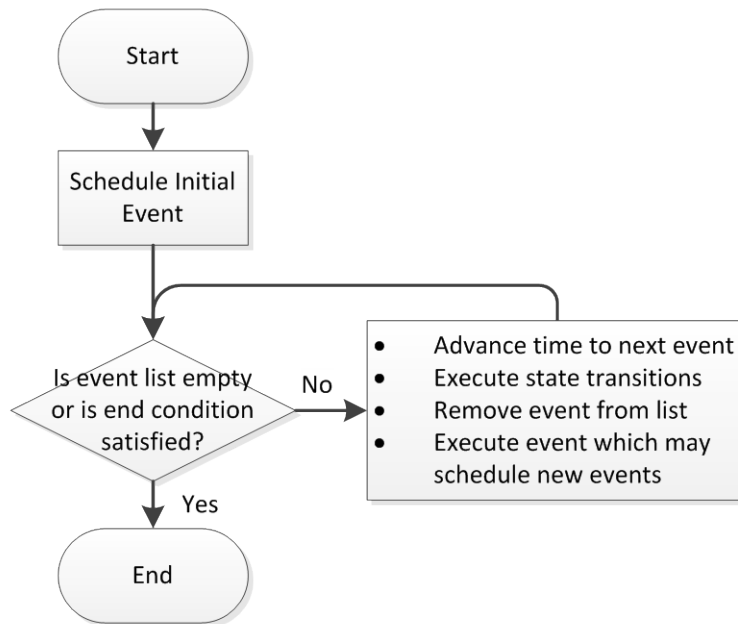


Figure 18. Typical DES engine handles events during a simulation.

To accommodate FTA and PES, a step in the DES process has to be modified. Recall that events in AEMF-DES are in the form of method of objects. Since AEMF-DES has to accommodate arbitrary user-defined scenarios of the given theme, the initial event is unknown. A simulation begins with the startup methods of all defender entities and those of all attackers to trigger initial events such as a missile moving. This sequence is decided by the assumption that a defender's entities exist earlier than an attacker's entities. Then, similar to a typical DES flow, the end condition is verified and used to determine whether the simulation should continue (as shown in Figure 19).



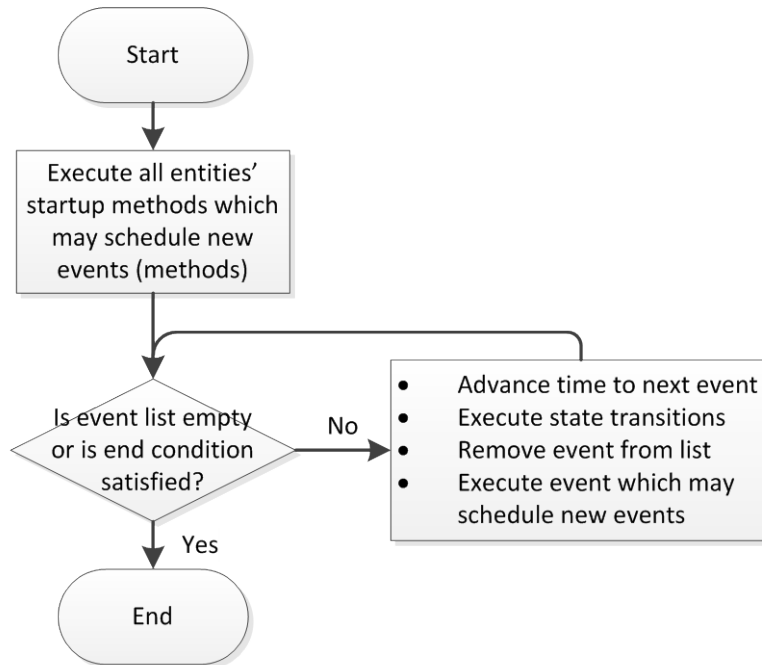


Figure 19. The DES engine for AEMF-DES starts simulation with the startup methods of all entities.

An example of an event history for the missile-radar-weapon scenario is provided in Table 6. The simulation begins with the two defender entities' startup methods. Here no special actions need to be done. Then the startup method of an attacker's missile is executed in that the future event of arriving at the destination is first scheduled, then the path is compared with all the defender's sensor coverage. If any path traverses might exist, schedule the corresponding sensor events as well.

Table 6. Example of event history (excerpted) of sample scenario shows that the startup method initializes all events.

| Sequence | Time  | Events          |                   | Description  |
|----------|-------|-----------------|-------------------|--|
|          |       | Entity          | Method            |  |
| 1        | 00:00 | Radar Station   | Startup           |  |
| 2        | 00:00 | Anti-Air Weapon | Startup           |  |
| 3        | 00:00 | Missile         | Startup           | <ul style="list-style-type: none"> <li>Schedule move method. (event 4)</li> </ul>  |
| 4        | 00:00 | Missile         | Move              | <ul style="list-style-type: none"> <li>Schedule a new event for arriving destination.</li> <li>Figure out whether the path will enter any defender's sensor's coverage. If so, schedule corresponding event. (event5,7)</li> </ul> |
| 5        | 01:00 | Radar station   | AddTargetCovered  | <ul style="list-style-type: none"> <li>Add the missile to a list, every time this sensor attempts a detection, all targets in the list are considered. (event 6)</li> </ul>  |
| 6        | 01:34 | Radar station   | Detection         | <ul style="list-style-type: none"> <li>This sensor attempts to detect. All targets in the list have a chance to be detected.</li> </ul>  |
| 7        | 02:00 | Radar station   | LostTargetCovered | <ul style="list-style-type: none"> <li>Remove the missile from the possible target list.</li> </ul>  |

## 2. Program Architecture

The characteristics of four elements described in the previous section are illustrated in Figure 20. Guided by a given theme, FTA can be regarded as a language to describe the primary principles of the given theme; PES is a more specific model of a particular scenario. Using FTA as the parts, various entities can be created and used to assemble fruitful scenarios. The DES engine is used only in the simulation phase. The PES model is interpreted, and the DES engine can create and schedule events by referring to FTA. After all events are executed, the simulation result is then generated. Through the

cooperation of these three elements of AEMF-DES, it is believed that events can be managed automatically after the simulation program is developed by simulation specialists. If other scenarios belong to the same theme that is proposed, no additional human effort is needed for analysis or reprogramming.

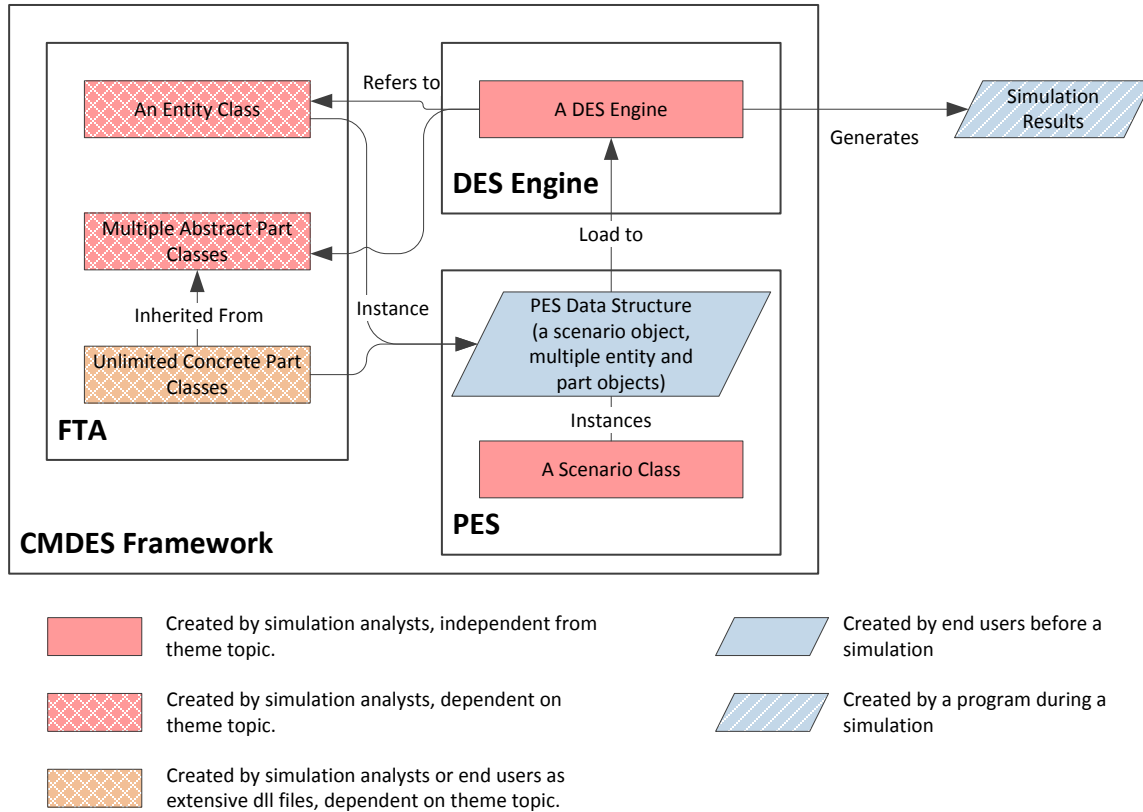


Figure 20. A scenario derived from user’s need is described by PES, which utilizes FTA and then loads it into a DES engine to generate simulation results.

#### D. SUMMARY

In this chapter, we discussed the limitations of the current DES approach and proposed a new framework, AEMF-DES, which mainly provides for the automation of the events creation and schedule and enables the incorporation of related scenarios as well as custom entities into the program. The implementation of a sample program is demonstrated in next chapter providing more detail regarding program development.

#### IV. DEMONSTRATED SIMULATION PROGRAM: MISSILE DEFENSE SIMULATION (MDSIM)

AEMF-DES, which is an improved DES framework, attempts to minimize some limitations of the current DES approach. Along with a discussion of AEMF-DES in this chapter, a simulation program MDSIM is introduced. This program not only demonstrates the feasibility of AEMF-DES but also illustrates an architecture for C4ISR processes simulation for an MDW theme. Note that the simulation architecture introduced here can be adapted to a more complex one if more accurate results are desired.

In 2005 Buss first introduced a simple movement and detection model using DES (Buss and Sánchez 2005). Although the time-step approach was most commonly used, he noticed that the DES approach is not only applicable but also has unique advantages. Basic algorithms of movement and detection were proposed. Based on this achievement, the following study mentions more about C4ISR related issues and ends up with simulation architecture for C4ISR in MDW (shown in Figure 21).

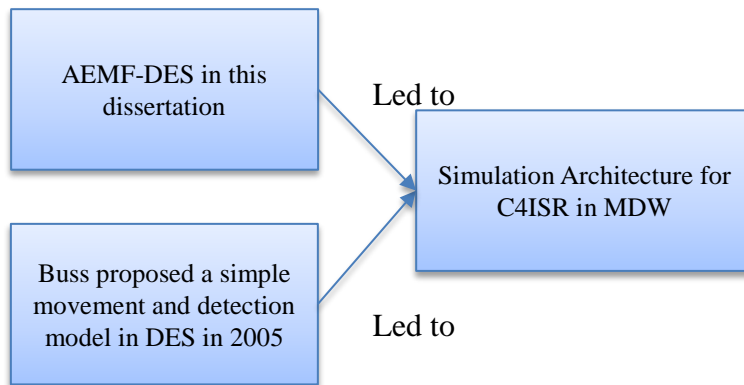


Figure 21. This simulation architecture is contributed from AEMF-DES and previous DES works.

##### A. MDSIM CONCEPT AND SCOPE

There are many aspects to consider in MDW, such as prior intelligence collection, communication network connection, radar detection, and personnel skill training

(Chasteen 2012; Katopodis, Katsis et al. 2007; Mei Dan et al. 2008). This work focuses on the C4ISR which is all about information generation, delivery, reference, usage, and utilization in battlefield. Rather than using a comprehensive model to describe the probability in each steps, ABM is used to provide better simulation resolution (Nguyen, Yip 2010; Wijesekera et al. 2005; Jun Wang et al. 2011).

Such an aviation invasion might be for supporting land battle for an Iron Hand also known as Suppression of Enemy Air Defenses (SEAD)(Bolkcom 2005; Lambeth Summer 2002). In a typical engagement scenario, incoming missiles are first detected by sensors (e.g., radar). Detection reports are then delivered via communication links to headquarters, which has the right to initiate all defense activities. Detection reports might be sent to weapon systems (e.g., anti-aircraft artillery) at the same time if communication links between sensors and weapons systems exist. After headquarters makes a decision, commands are subsequently transferred to weapon systems to guide interception with incoming missiles. Typical C4ISR processes in MDW are illustrated in Table 7 (National Research Council 2006).

Table 7. Typical C4ISR Processes in MDW

| C4ISR Processes                 | Description   |
|---------------------------------|---|
| Surveillance and Reconnaissance | Incoming missiles are detected by sensors, such as radars systems.  |
| Intelligence                    | Detection reports are restored, integrated and analyzed to generate meaningful and comprehensive information for commander.   |
| Command                         | A commanding officer orientate environmental situation and make commands for subordinates.  |
| Control                         | Once a command has been generated, it must be delivered to weapons to guide actions like fire or stop.  |
| Communication                   | Communication represents delivering message among distributed units on the battlefield. This might be wireless radio communication or a wired network on ground.  |
| Computing                       | Inside each agent on the battlefield, modern computers help to speed up information processes and data analysis. Typical examples include the signal processing computer in a radar station, digital switch in communications equipment, and an intelligence program at headquarters. |

Due to the time limitations and security considerations, the scope of MDSIM is determined as shown in Table 8.

Table 8. MDSIM covers most primary capability of C4ISR in MDW.

| C4ISR Processes                 | Capability included   | Capability not included  |
|---------------------------------|---|--|
| Surveillance and Reconnaissance | <ul style="list-style-type: none"> <li>• Sensor detection distance</li> <li>• Sensor radiation power</li> <li>• Target radar cross section</li> <li>• Electromagnetic noise jamming</li> </ul>  | <ul style="list-style-type: none"> <li>• 3D coordinate</li> <li>• Terrain elevation</li> <li>• Sensor types other than pulse radar system</li> </ul> |
| Intelligence & Command          | <ul style="list-style-type: none"> <li>• intelligence gathering</li> <li>• Decision time delay</li> </ul>   | <ul style="list-style-type: none"> <li>• Intelligence analysis</li> <li>• Content of command</li> </ul>  |
| Control                         | <ul style="list-style-type: none"> <li>• Sending command from headquarters to subordinates</li> </ul>   | <ul style="list-style-type: none"> <li>• Command ambiguity</li> <li>• Message error</li> <li>• Defender jammer to missile seekers</li> </ul>         |
| Communication                   | <ul style="list-style-type: none"> <li>• Deliver message from one unit to multiple units</li> <li>• Wireless communication</li> <li>• Electromagnetic noise jamming</li> <li>• Transmission bandwidth</li> <li>• Encode and decode delay</li> </ul> | <ul style="list-style-type: none"> <li>• Voice communication</li> <li>• Frequency overlapping</li> </ul>   |
| Computing                       | <ul style="list-style-type: none"> <li>• Process time delay in all other process</li> </ul>   | <ul style="list-style-type: none"> <li>• Individual computer properties</li> </ul>   |

## B. DEVELOPMENT STEPS

### 1. Build Fundamental Theme Agents (FTAs)

The FTA of MDSIM is summarized in Figure 22. In addition to the Entity class, five abstract part classes including Sensor, CommNode, Headquarters, Weapon, and Jammer classes describe primary principles. Several Concrete part classes inherit from these abstract part classes and subsequently define some key algorithms (such as how to determine the probability of detection, how long to deliver a message, how often a headquarters can make a decision, and how efficiently a weapon can intercept a target). Note that these parts are not only determined by the theme chosen, but also according to

the required simulation resolution. Fewer parts may be sufficient for simple analysis, whereas more parts are recommended for a detailed analysis.

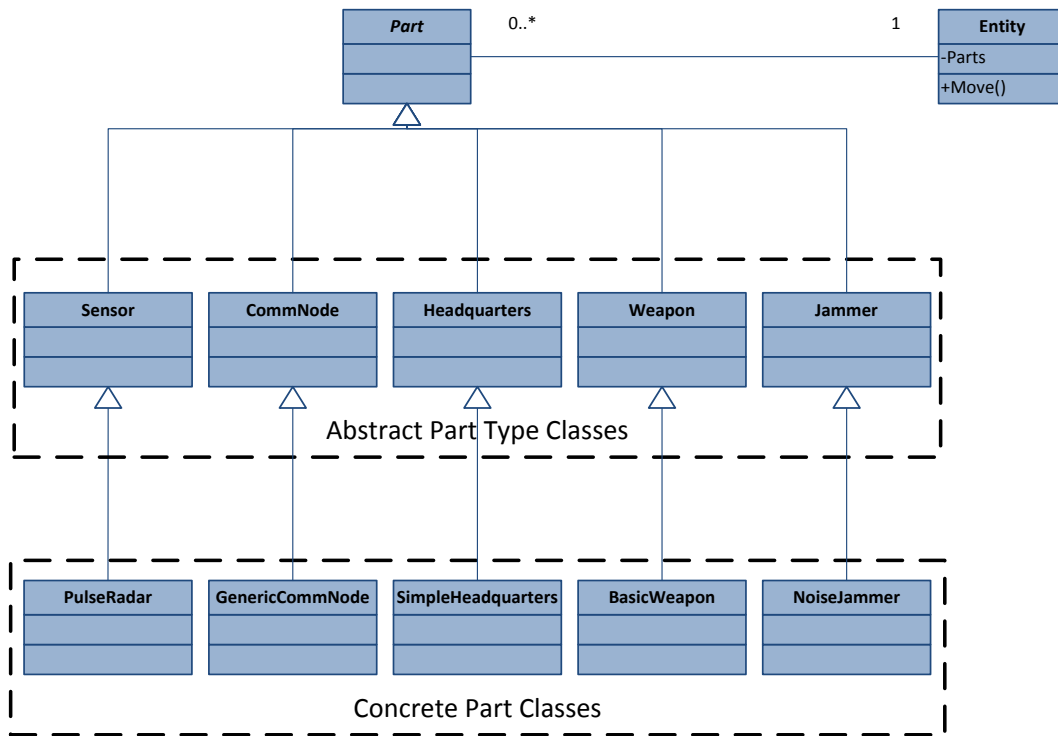


Figure 22. Besides the Entity class, Abstract part classes and Concrete part classes construct the FTA of MDSIM.

The Sensor, CommNode, Headquarters, and Weapon classes can only be used in defender entities to represent those C4ISR processes in the defender aspect. On the other hand, the Jammer class is used in attacker entities to simulate an electromagnetic noise source. The following section introduces these part types briefly. The concept of application and key preformation considerations are covered also.

*a. Entity Class*

In AEMF-DES, an entity is regarded as a platform of a combat unit. In MDSIM, the Entity class models a generic platform which could have multiple parts attached to it. An attacker’s entity can represent for a missile that does not need any parts insides—it completes its mission just by maneuvering to preset destination. During the route to the target, the missile only can be stopped by defender’s weapon attack, missile seek is not yet included in this program (Zarchan 2002). Jammer parts can be added to

attacker's entities to provide jamming capability. On the contrary, a defender entity can contain Sensor, CommNode, Headquarters, or Weapon parts to simulate various processes in C4ISR. For simplicity, MDSIM assumes that only attacker units can maneuver during the simulation.

Core methods of the Entity class along with pseudo codes are shown in Figure 23. The Startup method of an entity object is triggered at the beginning of a simulation. In the Showup event, if an attacker's entity object has been discovered by any sensors, this unit is added to the sensor's possible target list and is considered every time the sensor attempts to detect. If a navigation point exists, which means this unit maneuvers during simulation, then the BeAtNavigationPoint event is scheduled. BeAtNavigationPoint represents the event that a unit is ready to move or has just finished its movement. The following BeAtNavigationPoint events are scheduled depending on whether more navigation points exist.

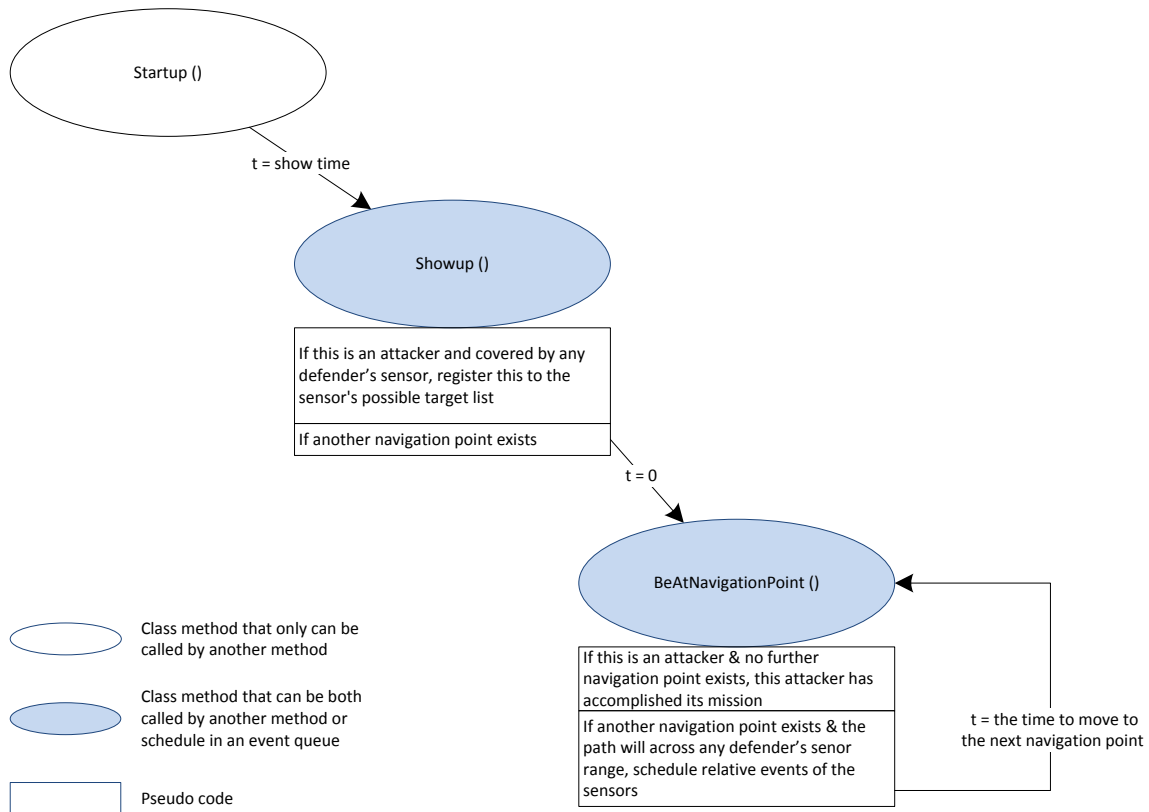


Figure 23. Core methods of the Entity class along with pseudo codes.



***b. Abstract Sensor and Concrete Subclass***

Sensors in the battlefield take the responsibility of reconnaissance and surveillance. Sensor examples include soldiers (with human eyes), night vision cameras (with an infrared system), or radar systems. Each sensor type has its own advantages and disadvantages in cost, coverage, or accuracy. Since radar is the main sensor type used in MDW, an abstract sensor part and all the subclasses are assumed to be like radar.

(1) The Abstract Sensor Class. Radar is an electromagnetic system that transmits electromagnetic waves through space and receives possible echoes from a target within coverage. Detection results could include target location, speed, and motion direction. Since a radar system detects each echo to “feel” the existence of a possible target, the performance of detection is governed mainly from (1) the power transmitted, (2) the radar-cross-section which determines the ratio of reflection, (3) the coverage from radar to the target, and (4) relevant environmental factors (Skolnik 2001).

The core methods of the Sensor class are shown in Figure 24. The two initial methods on the left side are CheckPointCover and CheckPathCover, which are triggered by the Entity class when an attacker’s unit shows up or attempts to maneuver. If a location point or a path is covered by this sensor, AddTargetCovered and LostTargetCovered events are then scheduled. Detect event takes place repeatedly only when at least one target is discovered within coverage. As long as a target is detected, a report message is then delivered to listening parts within the same entity object.

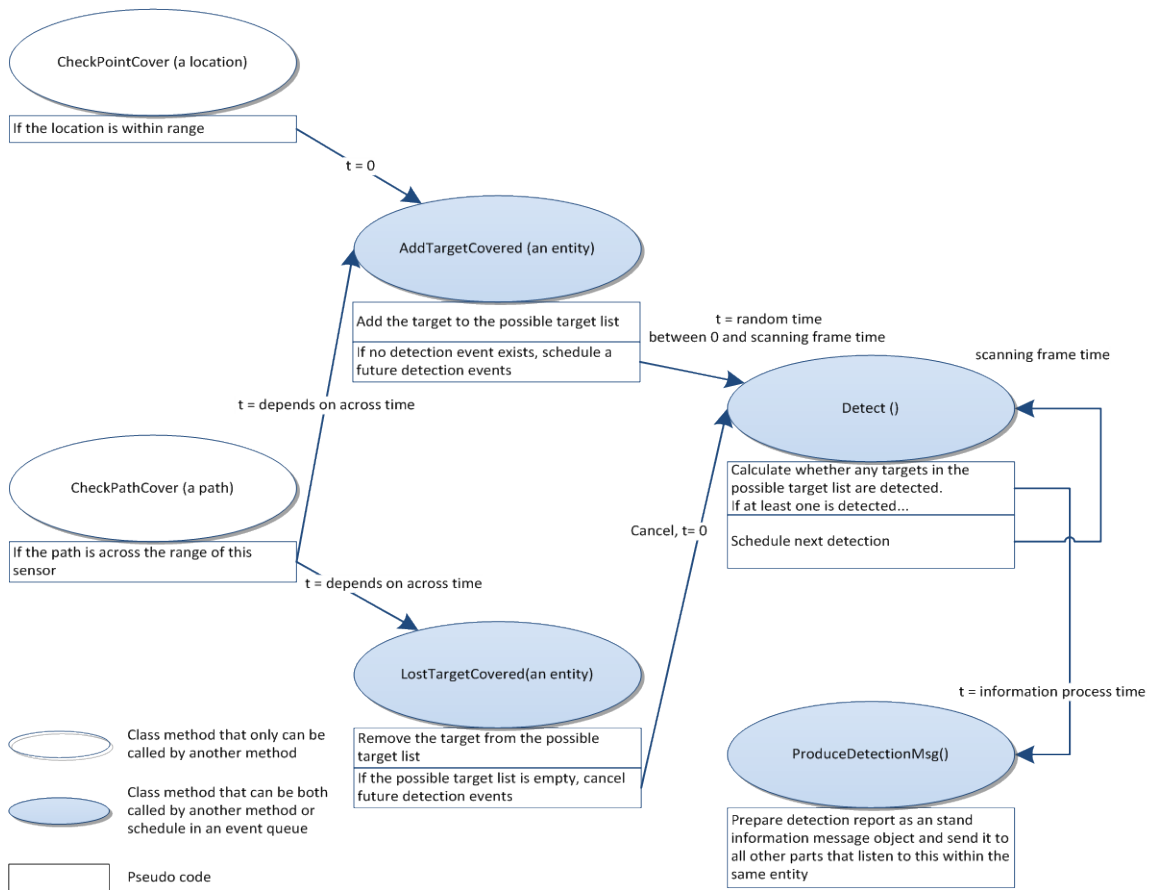


Figure 24. Core methods of the Sensor class along with pseudo codes.

Table 9 lists some important abstract methods of the Sensor class. These methods support the event execution depicted in Figure 24 and must be implemented by inheriting concrete subclasses.

Table 9. Abstract methods of the Sensor class that must be implemented by Concrete subclasses.

| Method Name        | Input Parameter(s) | Return Variable(s)    |
|--------------------|--------------------|-----------------------|
| ifPointCovered     | A location point   | Boolean               |
| ifPathCovered      | A maneuver path    | Enter time, exit time |
| GetFrameTime       | N/A                | Time span             |
| GetProcessTime     | N/A                | Time span             |
| ExportMessage      | A message          | Void                  |
| GetDetectionReport | N/A                | Detection report      |

(2) A Concrete Subclass: PulseRadar. A radar system utilizes electromagnetic waves to detect targets. Detection report is decided by examining the echo signal. Strength of echo power can be calculated by the following equation (Pace 2009):

$$P_r = \frac{P_t \sigma_T G_t A_e}{L_{RT} L_{RR} (4\pi R_T^2)^2} \quad (W) \quad (2)$$

$P_r$  : Radar power received

$P_t$  : Radar power transmitted

$L_{RT}$  : Loss factor of radar transmitter

$G_t$  : Antenna gain

$R_T$  : Range from antenna to target

$\sigma_T$  : Radar cross section (RCS) of target

$A_e$  : Effective area of receiver antenna

$L_{RR}$  : Loss factor of radar receiver

Notice that RCS is the variable standing for the visibility of a target to radar system. An invisible design aircraft would have lower RCS than the one of a traditional design aircraft. This visibility could be reduced by modifying the shape to prevent reflective echo toward radar station or by using special materials to absorb electromagnetic waves.

Except for the electromagnetic wave created by humans, natural noise exists due to the agitation of the charge carriers inside conductors. Nature noise is also known as thermal noise or white noise and can be represented as

$$\text{Thermal noise} = kT_0 B_{Ri} F_R \quad (W) \quad (3)$$

$k$  : Boltzmann's constant

$T_0$  : Standard temperature 290K

$B_{Ri}$  : Receiver bandwidth

$F_R$  : Noise Effector of receiver

Another source of noise is hostile jamming which attempts to affect the radar performance by increasing the noise level (Erdemli 2009; Mofrad, Sadeghzadeh 2010; Shen Tong-yun et al. 2011). While the radar is jammed, the in-band jamming signal increases the noise power of the received signal. As a result, the detection capability is degraded and the detection coverage reduced. Jamming noise power at the receiver is defined by (Pace 2009; Chen and Pace 2008):

$$P_n = \frac{P_j G_j A_e}{4\pi R_j^2 L_{RR}} \quad (W) \quad (4)$$

$P_n$  : Jamming power received

$P_j$  : Jamming power transmitted

$G_j$  : Antenna gain

$R_j$  : Range from jammer to antenna

$A_e$  : Effective area of receiver antenna

$L_{RR}$  : Loss factor of radar receiver

Echo from possible targets can be detected only when it is explicit from natural noise and jamming noise. Signal-to-Noise Ratio (SNR) is the common metric used to measure the strength of the echo received by the radar as (Pace 2009; Difranco and Kaiteris 1981):

$$SNR = \frac{P_r}{P_n + P_j} \quad (5)$$

$P_r$  : Echo power received from target

$P_n$  : Nature Noise power

$P_j$  : Jamming Noise power

When the SNR of an echo is greater than the minimum SNR required of a receiver, the existence of a target may be observed. Otherwise, it is invisible. Similar to the Pulse Radar model, this Generic Comm model does not include the loss of propagation such as absorption and scattering.

*c. Abstract CommNode and Concrete Subclass*

Simulation for communication network in battle has been proved to be an effective to analyze and design systems (Fanfan Yao al. 2012; Yang Qing-wen et al. 2009). Communication delivers information among different units and can be wired, such as land-line telephones, or wireless radios. The link can be formed by more than two units, which is usually called a communication network. To simulate the communication link/network among units in the battlefield, we focus on the communication equipment of a terminal (i.e., a communication node), and then consideration the signal channel among nodes (Yi Deng et al. 2012; Haitao Yang et al. 2006). If two communication nodes share the same communication setting, then a link is assumed to exist between them. If more than two communication nodes share the same communication setting, they reside on a network.

(1) The Abstract CommNode Class. Figure 25 shows the core methods of the abstract CommNode class. ImportMsg is triggered while a message is delivered to this CommNode object. The incoming message will first be stored in a queue and then be popped up when this CommNode has delivered all previous messages. Message transmission for a period of time in MDSIM is modeled by two signs to form a simulation protocol: head and tail signs which stand for the beginning and ending of the transmission (Larocque and Lipoff 1996). Both two sign must be collected by the listeners so that the target message can be restored. However either or both signs could fail if communication signal strength is too low or hostile jamming exists. Table 10 lists the abstract method that must be implemented by subclasses.

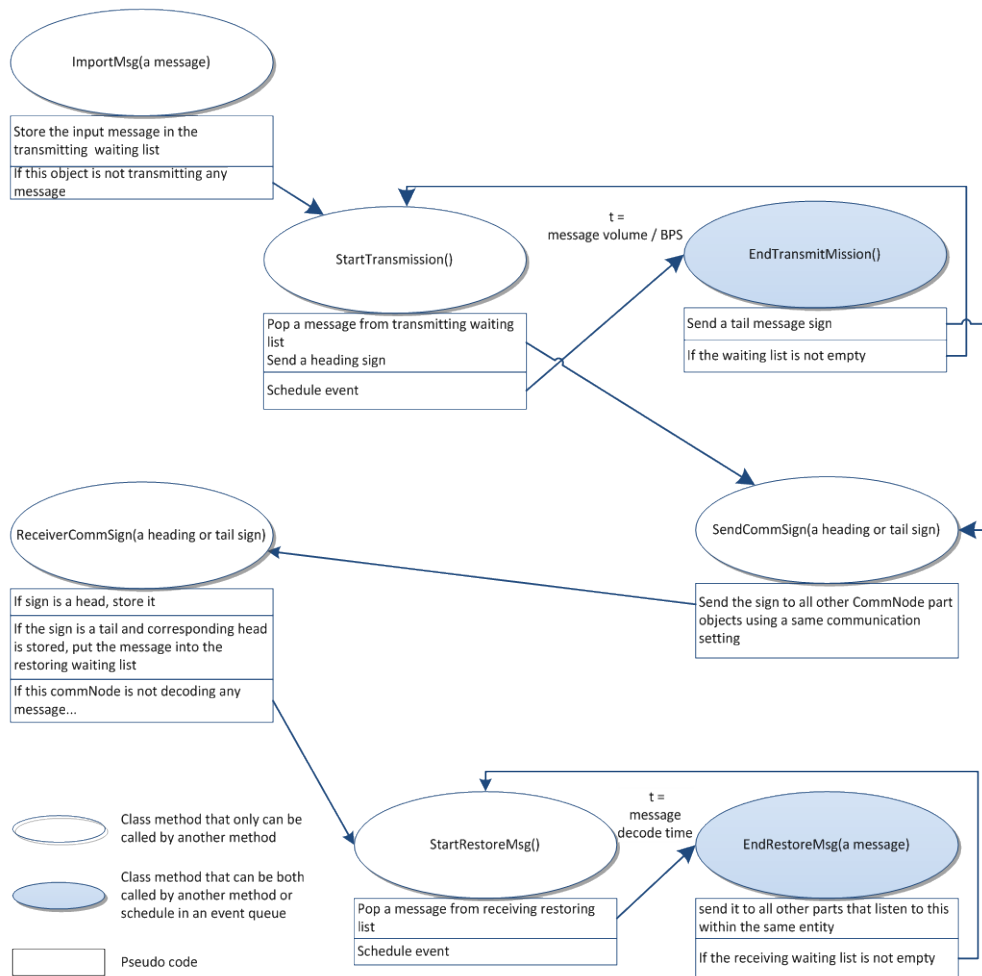


Figure 25. Core methods of the CommNode class which uses head and tail sign to model information transmission.

Table 10. The abstract method that must be implemented by subclasses

| Method Name           | Input Parameter(s) | Return Variable(s) |
|-----------------------|--------------------|--------------------|
| ValidateCommunication | Another CommNode   | Boolean            |

(1) A Concrete Subclass: Generic CommNode. Because it can accommodate the demands of most types of environments, wireless communication has

been widely used on the modern battlefield. Bits-per-second (BPS) is used to represent the rate of communication over a network. The time needed to deliver a specific amount of data (in bits) is shown in (Sklar, 2001):

$$\text{latency of data delivery} = \frac{D}{BPS} \quad (\text{Sec}) \quad (6)$$

$D$  : Data volume (in bits)

$BPS$  : Bit per second, the effective transmission capability of data link

Similar to a radar system, wireless communication is also bound by the strength of signal received. The definition of power received in a wireless communication system is similar to that of a radar system, except that wireless communication is a one-way transmission shown in equation ( 7 ) (Sklar, 2001).

$$P_r = \frac{P_t G_t A_e}{L_{RT} L_{RR} 4\pi R_T^2} \quad (\text{W}) \quad (7)$$

$P_r$  : Communication power received

$P_t$  : Communication power transmitted

$L_{RT}$  : Loss factor of the transmitter

$G_t$  : Antenna gain

$R_T$  : Range from antenna to target

$A_e$  : Effective area of receiver antenna

$L_{RR}$  : Loss factor of the receiver

Communication systems also have similar noise power which may include the thermal noise, shown in Equation ( 3 ), and hostile jamming, shown in Equation ( 4 ). When the SNR of the receiving signal is greater than the minimum SNR required, this information delivery is valid. Otherwise the communication links/networks could be prohibited if a signal is too weak or it encounters jamming. The SNR equation is shown as:

$$SNR = \frac{P_r}{P_n + P_j} \quad (8)$$

$SNR$  : Signal to Noise Ratio

$P_r$  : Echo power received from target

$P_n$ : Nature Noise power  
 $P_j$ : Jamming Noise power

Similar to the Pulse Radar model, this Generic Comm model does not include the loss of propagation such as absorption and scattering.

*d. Abstract Headquarters and Concrete Subclass*

A headquarters model includes two components: intelligence process and commander. In MDSIM all incoming information is stored in intelligence storage, and then a commander refers to it to make a decision. Subsequently a command is generated; it is sent to subordinates to perform. The concept of a headquarters model is illustrated in Figure 26.

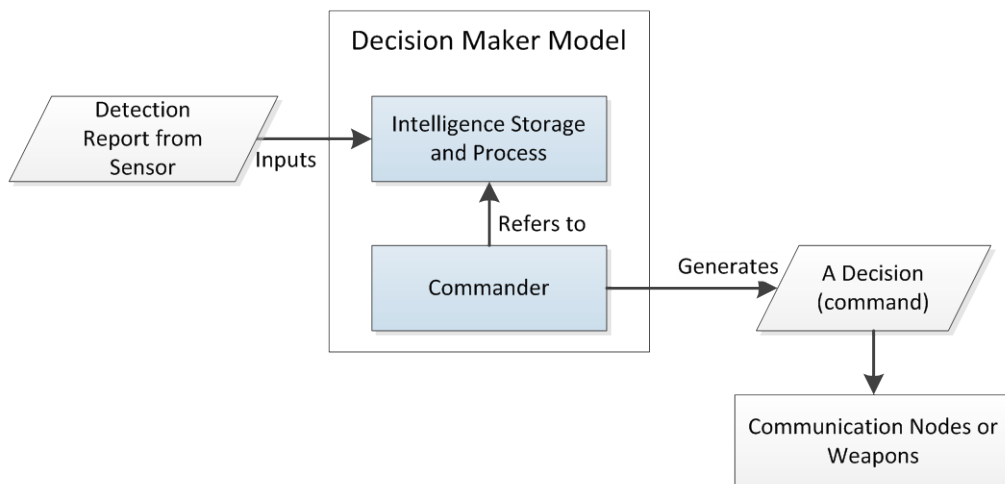


Figure 26. An illustration of the headquarters model concept.

(1) The Abstract Headquarters Class. The core methods of the abstract Headquarters class are shown in Figure 27. The ImportMsg method is called whenever a message is delivered to this object. But only the detection report is stored in intelligence storage. If this headquarters object is not making any decision, the making decision loop will then be activated. Each decision making action takes a specific period of time and can only refer to the intelligence storage this headquarters object owns. In



other words, no decision is made if no information is offered to a headquarters object. Once a command is generated, it will then be delivered to other parts listening to this object.

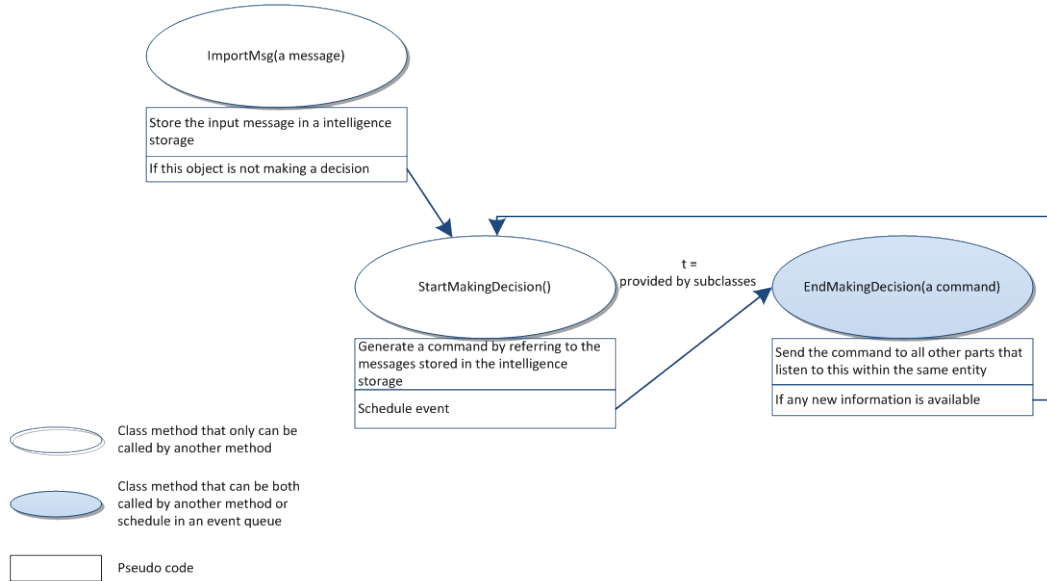


Figure 27. The core methods of the abstract Headquarters class.

The abstract method which is used by the abstract Headquarters class and must be implemented by concrete subclass is identified in Table 11. The GetCommand method not only determines the algorithm used to generate a command, such as fire on a target or ignore it, but also returns the time needed for making such a decision. The time can be a probabilistic distribution depending on the assumption of properties of a headquarters.

Table 11. The abstract method of the abstract Headquarters class and must be implemented by concrete subclass.

| Method Name | Input Parameter(s)         | Return Variable(s)  |
|-------------|----------------------------|---|
| GetCommand  | An intelligence collection | <ul style="list-style-type: none"> <li>• A command</li> <li>• Time Duration (needed for making this command)</li> </ul> |

(2) A Concrete Subclass: SimpleHeadquarters. The decision making progress can be modeled by a computer agent model to explore the internal algorithm inside a C4ISR system (Sari, Kuspriyanto, & Prihatmanto, 2012). However the related factors may include various possible considerations such as culture and strategic goal. To focus only on the information flow of C4ISR in the simulation, this research ignores the qualitative variation of commands, assumes that an incoming missile can always be recognized and assumes that commanders always give the same instruction to take down threat. That is, the quality of the command decision will not be a factor in to the simulation. However, the variable time latency caused by the command process can be considered.

Most modern information processing is conducted according to a predesigned standard operating procedure (SOP) with designated computer equipment. Compared to the purely human operations of the past, the SOP with partial full automation provides advantages in both quality and processing speed. The level of automation of decision and action selection is categorized in Table 12 (Parasuraman, Sheridan, & Wickens, 2000).

Table 12. Sheridan Levels of Authority for Decision and Action Selection (after Sheridan 2002).

| Level | Computer Task         | Description             |
|-------|-----------------------|-------------------------|
| 1     | No assistance         | Does all                |
| 2     | Suggest alternatives  | Chooses                 |
| 3     | Select way to do task | Schedules response      |
| 4     | Select and execute    | Must approve            |
| 5     | Executes until vetoed | Has limited veto time   |
| 6     | Executes immediately  | Informed upon execution |
| 7     | Executes immediately  | Informed if asked       |
| 8     | Executes immediately  | Ignored by computer     |

In MDSIM, the decision-making latency is simply modeled as a variable provided by subclass and indicates the automation level of a headquarters.

*e. Abstract Weapon and Concrete Subclass*

Weapon systems in this program play a role of intercepting invasive missiles. As mentioned before, the weapon is not part of C4ISR but we need at least a simple model to take care the work to engage with missile by referring to detection report and command generated from defender’s C4ISR system. The weapon systems regarding MDW include anti-air artillery (AAA) and surface-to-air missile (SAM). AAA fire trajectory projectiles to form a cloud that the target might bump into, on the other hand, SAM fires missiles which can trace targets until destroying or missing the targets (Westermann 2001; Werrell 2005; U.S. Department of Defense 2008).

(1) The Abstract Weapon Class. The core methods of the abstract Weapon class are shown in Figure 28. In the ImportMsg method, Incoming message is handled by its type. If it is a detection report from sensors, the message is

stored for future reference. If it is a command from headquarters, the following rules are applied:

- If this weapon is idle, a warm-up time is needed for system and personnel preparation. This weapon object will perform the ProcessCommand event.
- If this weapon is warming up, then do nothing since a ProcessCommand event has been scheduled.
- If this weapon is active, which means this object is attacking, do nothing since this object will call ProcessCommand later.
- If this weapon is on standby, this object is ready to attack anyone. Call the ProcessCommand method immediately.

In the ProcessCommand method, a target is first decided by an abstract method implemented by subclass. This weapon then attacks it and repeat ProcessCommand method after a fire interval. The event that a projectile might hit the target is triggered latter to decide whether the target is destroyed.

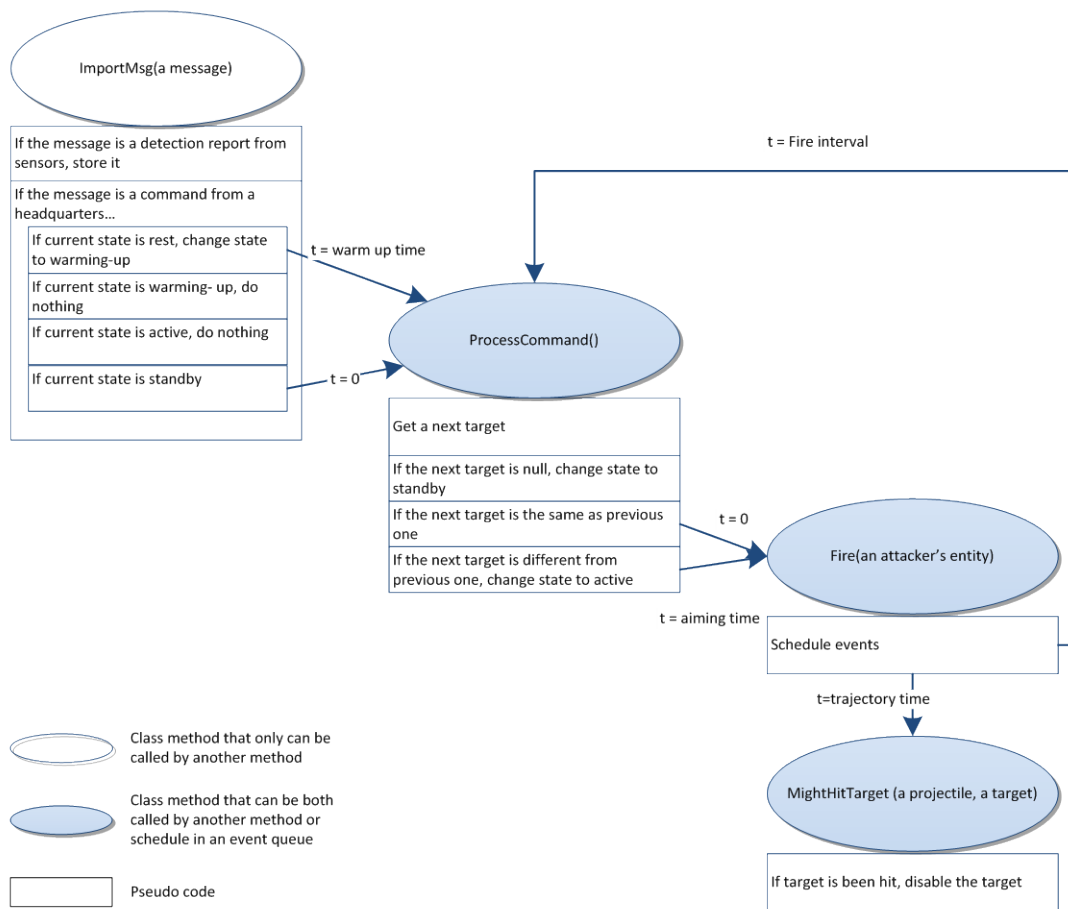


Figure 28. The core methods of the abstract Weapon class.

The important abstract methods of the abstract Weapon class that must be implemented by concrete subclass are listed in Table 13. Important variables such as warm-up time, aiming time, and fire interval must be provided by subclasses. These variables can be expressed as simple constants or complicated probabilistic distributions. The GetNextTarget method is used to offer the suitable target from the detection report this weapon has collected. For example, while two attacker units are within coverage, a weapon system may prefer to intercept the closer one.

Table 13. The abstract methods of the abstract Weapon class must be implemented by any concrete subclass.

| Method Name          | Input Parameter(s)          | Return Variable(s)   |
|----------------------|-----------------------------|----------------------|
| GetWarmingUpTime     | N/A                         | Time Span            |
| GetAimingTime        | N/A                         | Time Span            |
| GetFireinterval      | N/A                         | Time Span            |
| GetNextTarget        | Detection report collection | An attacker's entity |
| GetProbabilityOfKill | An attacker's entity        | Decimal              |

(2) A Concrete Subclass: BasicWeapon. Basically two types of weapons are applied for missile defense operation. They are Anti-Air Gun (AAA) and Surface to Air Missile (SAM). In MDSIM, we only consider a simple model that includes effective coverage, Probability of Kill ( $P_k$ ), and the time for a projectile/missile to reach a destination. Both AAA and SAM can be simulated by offering these variables from subclasses (Devore 2004).

*f. Abstract Jammer and Concrete Subclass*

A jammer radiates electromagnetic waves to increase the noise portion of any electromagnetic signals within the same band. As a result, radar systems and wireless communication systems could be affected by hostile jamming and blocked sensing or communication functions that are critical to the performance of modern C4ISR operation.

In MDSIM, a noise jammer is modeled by offering a noise source when Sensor or CommNode subclasses perform detection or communication. The hostile noise is counted as part of noise and affects the SNR value.

**g. Interface to Show Available Parts To users**

The interface for the part information of MDSIM is shown in Figure 29. From the interface, users can recognize those part files that loaded successfully.

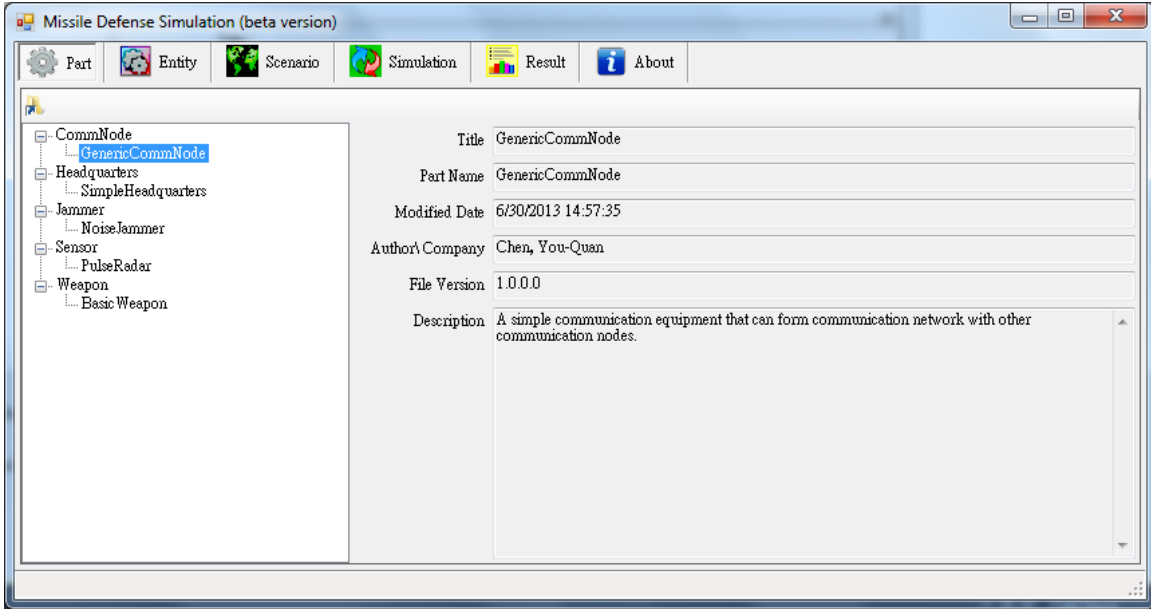


Figure 29. Illustration of part interface displaying successfully loaded information.

**2. Build Function for Custom Entity Templates**

In the AEMF-DES on which the MDSIM is based, combat units in the battlefield are modeled by a platform (an Entity object) and the multiple components (concrete part objects) on it. End users can define a custom entities template that can be used not only in the target scenario but also in future scenarios, too.

The interface designed for adding and editing an entity is shown in Figure 30. Simple attributes are listed in the top area. Part composition is displayed at the bottom where parts can be added, edited, or deleted. Note that the lines between parts illustrate the intercommunication with the entity. For instance, if there was no link between Basic Radar and Basic Weapon, Basic Weapon would be unable to use the detection information generated by Basic Sensor to intercept invasive missile.

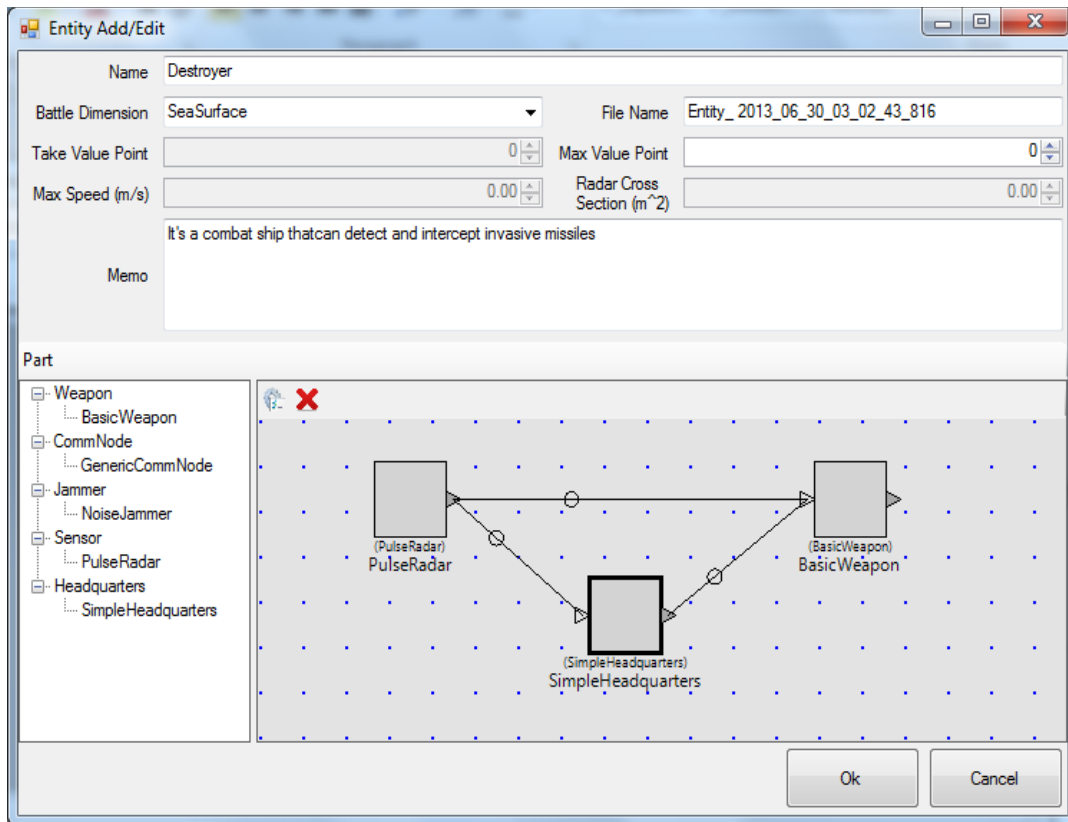


Figure 30. Interface lets users edit an entity's properties and arrange part settings.

The interface design for listing all entity templates is shown in Figure 31. Some important attributes such as affiliation, dimension, and name are shown for easy identification. Users can add new entity templates or select an existing one to edit or delete.



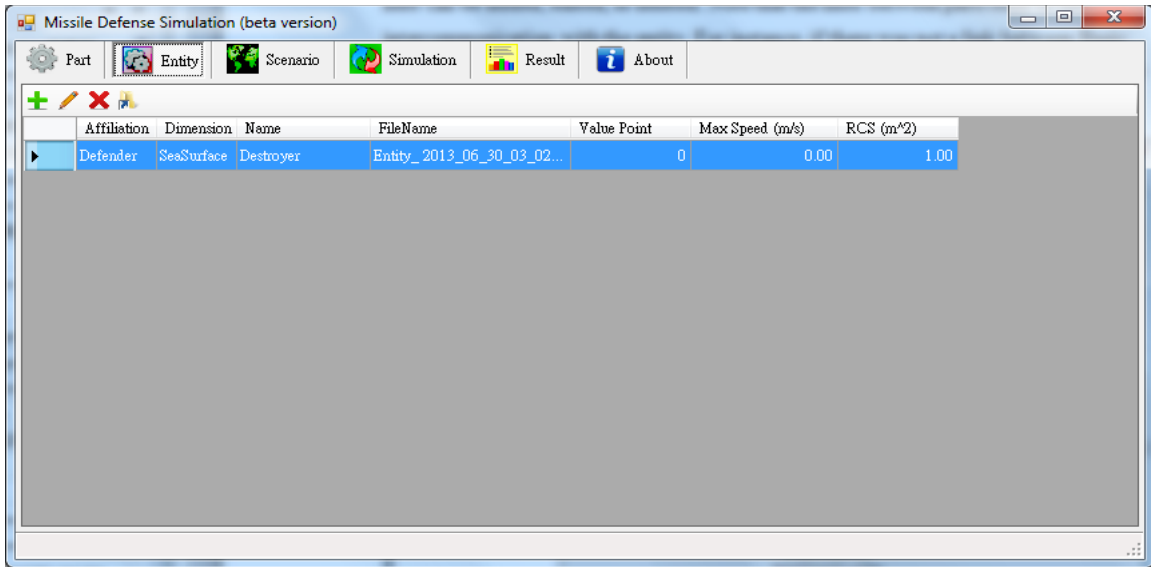


Figure 31. Available entity templates are listed.

### 3. Build Function for Scenarios Design

After defining the entity templates, users are ready to build a scenario. First they create an empty scenario and set its attributes (such as subject, size, and author). Then users add entities for both an attacker and a defender by referring to the entity templates. Later these new entities can be given more detailed settings like maneuver and electromagnetic parameters.

It is recommended to adopt a general symbology for user interface such as MIL-STD-2525C—Common Warfighting Symbology. Table 14 lists 6 symbols used in AEMF-DES. Battle dimensions include air, ground, and sea surface. Affiliations can be friend (Cyan) or hostile (red). Table 15 shows the line/curve patterns to represent paths, coverage, and links. Its color can be either cyan if its entity a friend or red if its entity is a hostile.

Table 14. Entity symbols used covers 3 battle dimensions and 2 affiliations.





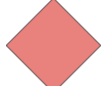



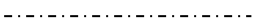

| Standard Identity | Battle Dimension  | Air  | Ground  | Sea Surface   |
|-------------------|---|--|---|---|
|                   | Friend (Cyan)   |   |   |  |
| Hostile (Red)     |  |  |  |   |

Table 15. Different line/curve patterns are used to represent paths, coverage, and links. The color can be either cyan or red depending on its friendly or hostile.

| Affiliation | Pattern   | Description                    |
|-------------|---|--------------------------------|
| Attacker    |    | Directional maneuver path      |
| Defender    |    | Weapon effective coverage      |
|             |  | Sensor effective coverage      |
|             |  | Directional communication link |

The interface design for that allows users to add and scenarios is shown in Figure 32. The buttons at the top of the screen are used to manage scenario files and show the current path. The tabs on the left side list the scenario details. The diagram on the right side of the screen shows entity settings.

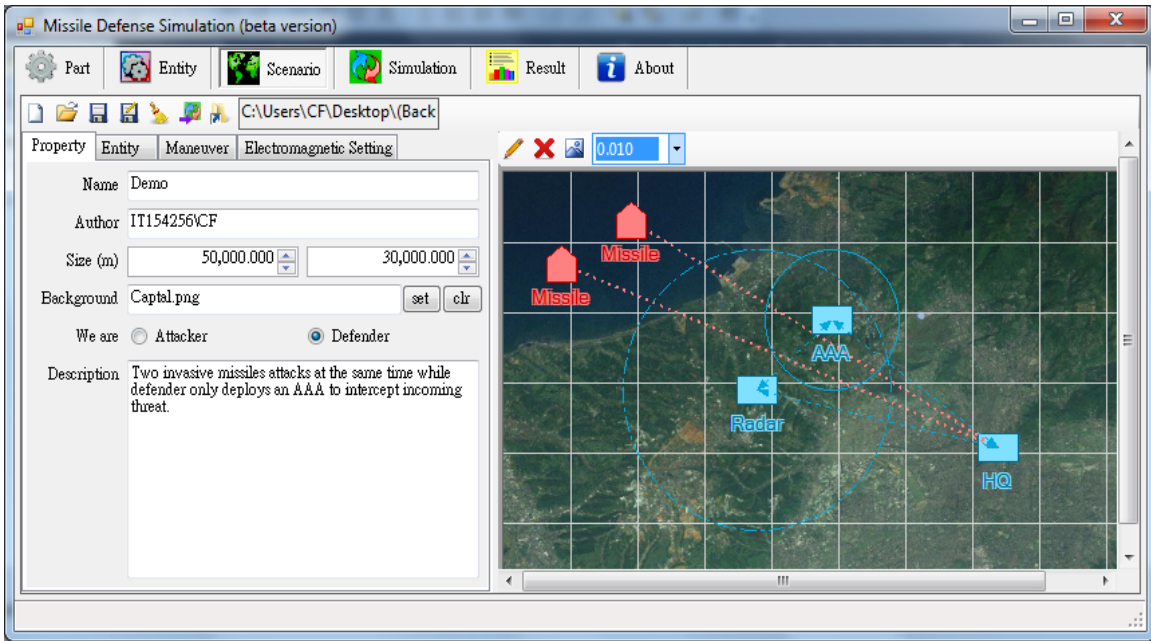


Figure 32. Interface design for Scenario Add/Edit.

#### 4. Build Function for Simulation Execution

A DES engine is used for conducting a simulation by (1) loading a PES defined by end users, (2) executing Startup methods for these entities in the PES and scheduling subsequent events, (3) executing subsequent events until the end condition is satisfied, and (4) generating a simulation result (refer to Figure 20).

Figure 33 shows the design interface of the executing simulation. The textbox labeled DES Engine Message indicates the state of the execution and the Event Queue is shown to its right. In the table below the Event Queue displays an event history. When the simulation is being executed, the state of battlefield is shown on the right panel. Our design for viewing simulation results is shown in Figure 34. It is similar to the design for executing simulation interface since the content of result mostly comes from the execution which saves it for later reviewing.

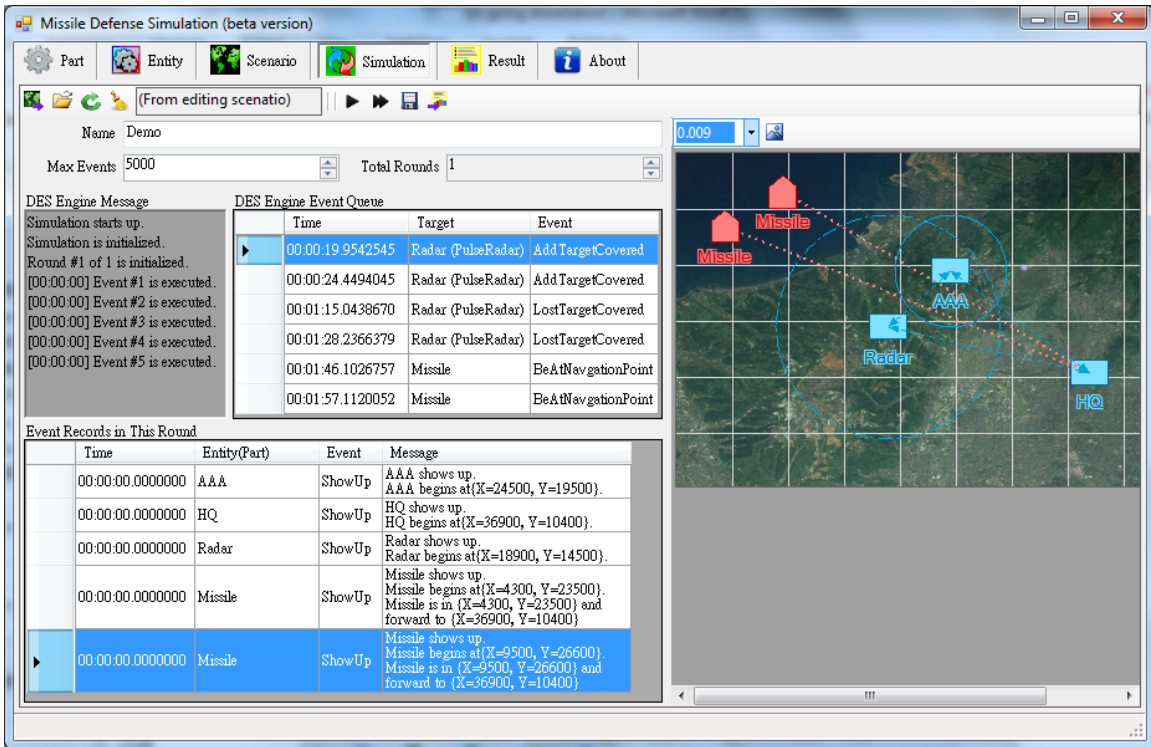


Figure 33. Design interface for executing a simulation.

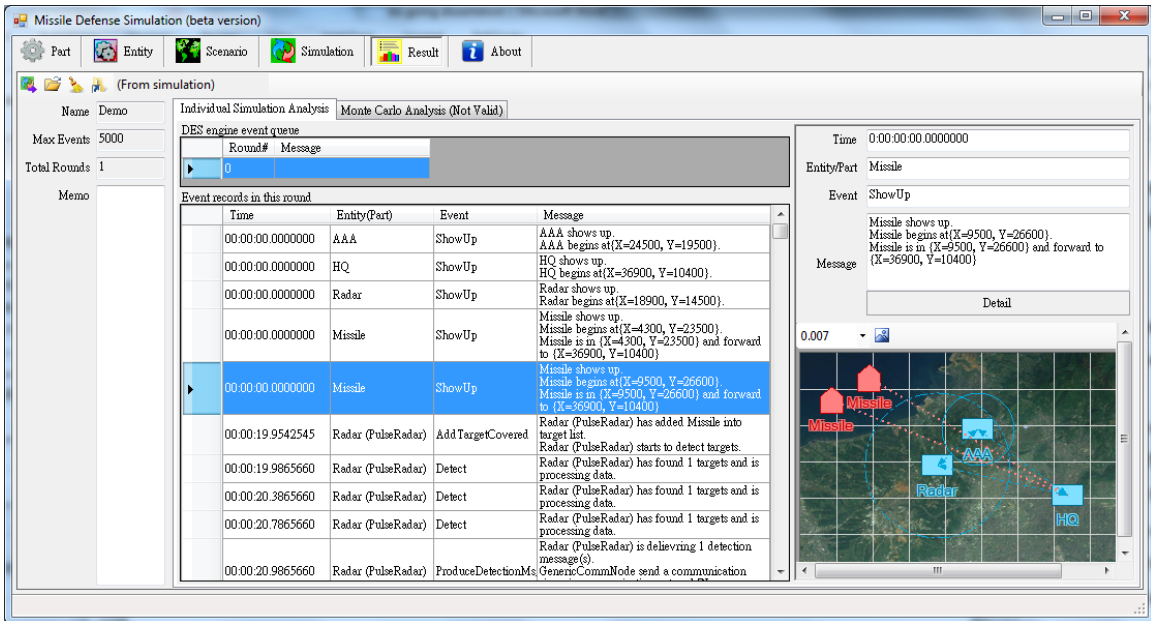


Figure 34. Design interface for viewing simulation results.

## C. FUTURE GOAL

### 1. Simulation Societies and C4ISR

From the viewpoints of academic domains, AEMF-DES relates to three communities. First one is the C4ISR community which covers sensing, decision, and action in real world. The other two are DES and the time-step approach from modeling and simulation community. While DES can simulate relative faster and handle the long-term scenario, the time-step approach can satisfy the complex calculation for physical fidelity. Their relation is shown in Figure 35.

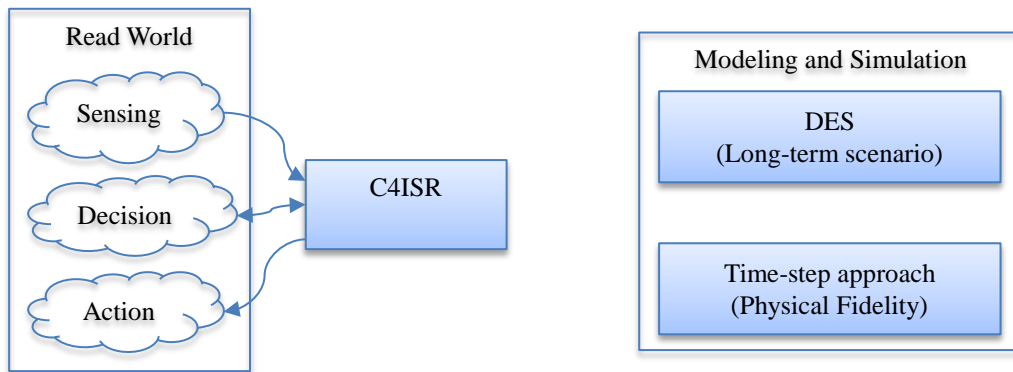


Figure 35. Communities related include C4ISR, DES, and the time-step approach.

To manage the needs in analyzing practical C4ISR process, the C4ISR community has utilized the time-step approach to build many simulation programs. However few studies have been found to use DES in resolving the C4ISR issues. This is believed due to the perceived limitation of current DES framework (shown in Figure 36).

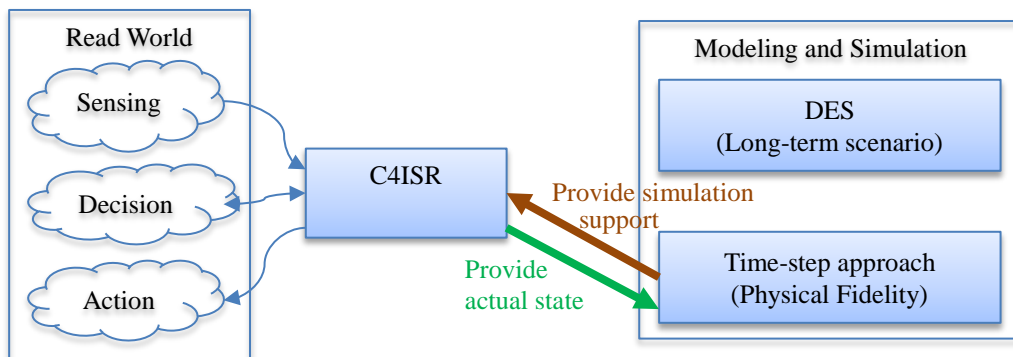


Figure 36. C4ISR and the time-step communities have cooperated to manage the needs on analyzing practical C4ISR processes.

In order to fully satisfy the needs of C4ISR analysis, it is believed that both DES and the time-step approach are required (shown in Figure 37).

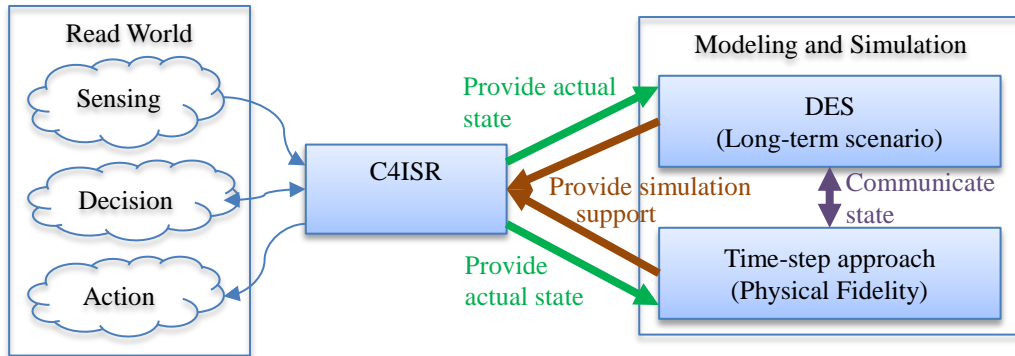


Figure 37. Both DES and the time-step approach are required to fully satisfy the needs in C4ISR analysis. Achieving this comprehensive hybrid architecture is future work.

MDSIM does not only prove the feasibility of AEMF-DES, but also discloses the possibility to include the time-step approach by the extensible part class. Figure 38 shows a pseudo design diagram in that a time-step-event mediator class is added to current FTA design. When the DES engine cannot satisfy second or higher order mathematical equations, the mediator can accept the request from the concrete parts to trigger events periodically and call back a specific function in the concrete parts. For example, if the sensor detection relies on the complex prediction of the EMW propagation, the detection function can ask the mediator to start the time-step mode. Whereas the time advances, the mediator performs a callback to allow the sensor to update the progress of the wave front. Rather than enable concrete part classes trigger period events to imitate the time-step approach, using such a mediator has advantages that the DES engine can be isolated from user-made concrete part classes to reduce system errors and the abstract part classes can also focus on theme primary principles.

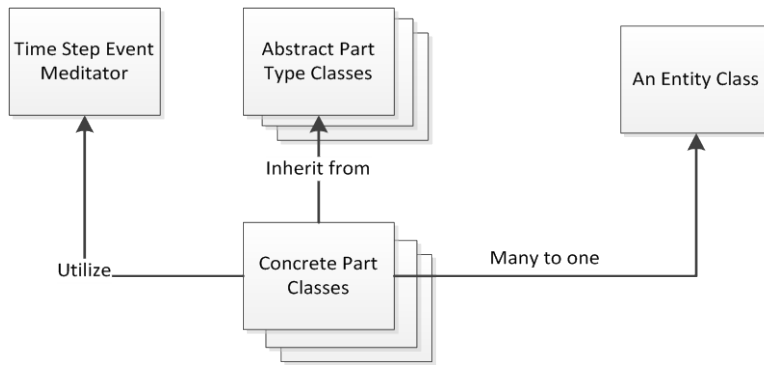


Figure 38. A time-step-event mediator can be added into FTA to enable those customized functions that need a time-temp mechanism

## 2. Simulation and VV&A Processes

With the possibility of building a simulation with high physical fidelity, it would be useful to include it into the standard VV&A processes (Show in Figure 39). For example, a new comprehensive simulation that is design to replace multiple existing validated simulations can be validated by these simulations. Another example is that before implementing an expensive validation in real word, doing a validation simulation might be helpful to prevent unnecessary cost.

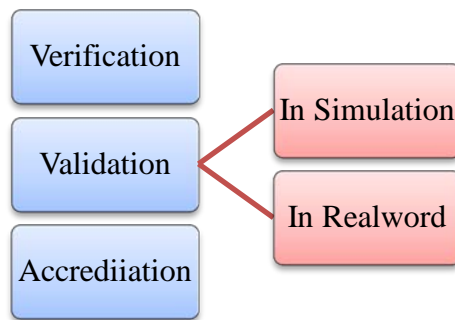


Figure 39. Add simulation to the validation process.

## D. SUMMARY

In this chapter, an architecture of MDSIM has been briefly introduced. Steps of development are shown including FTA, custom entity template, scenario design, and simulation execution. The program framework can be utilized to develop a simulation program and benefit military analysis work. Comparing to other public C4ISR simulation

architecture, MDSIM focuses on a high level viewpoint that considers information generation, delivery, and utilization. In addition by adapting AEMF-DES, MDSIM offers not only relative faster simulation speed but also improved capability in complex scenarios, related scenarios, and custom entities.

In the next chapter, we will discuss several simulation experiments that were run to verify the framework we have proposed. We will also demonstrate the design goal of this program using sample applications.



THIS PAGE INTENTIONALLY LEFT BLANK

## V. EXPERIMENT SIMULATIONS

Chapter IV introduces the development of the demonstrated simulation program (MDSIM) which is designed based on AEMF-DES and aims to simulate the C4ISR processes in MDW. In this chapter several experiments demonstrate the feasibility and potential of AEMF-DES and MDSIM (listed in Table 16).

Table 16. Two experiment series demonstrate the feasibility and potential of AEMF-DES and MDSIM

| Simulation          | Purpose   | Content   |
|---------------------|---|---|
| Experiment Series 1 | Proves that AEMF-DES can improve the limitations of current DES approach. | Contains 2 sub experiments to show these limitations can be improved.       |
| Experiment Series 2 | Demonstrates how MDSIM simulate the C4ISR processes in MDW.               | Contains 5 sub experiments to demonstrate all C4ISR processes step by step. |

The first experiment is devoted to prove the feasibility of AEMF-DES to minimize the limitations of the typical DES framework. To do this, two sub-experiments are designed to answer the following questions:

- Does AEMF-DES allow end users to propose a related scenario to the topical theme but not predefined in program?
- Does AEMF-DES allow end users to customize entity definition to model new battle units?

The second experiment attempts to show how C4ISR processes in MDW are simulated to prove the applicability of MDSIM architecture. The experiment begins with

few entities and then more entities are then added in subsequent steps to gradually build a scenario with all C4ISR processes within it.

**A. EXPERIMENT SERIES 1 : IMPROVED FLEXIBILITY FROM AEMF-DES**

The goal of AEMF-DES is to minimize the limitations of the released program that conducted by current DES approach by. In an effort to prove the feasibility of AEMF-DES, a few questions have to be answered.

First, does MDSIM allow scenarios belonging to MDW when none of them has been specifically considered during software development? If so, then it is proved that AEMF-DES can manage events automatically and resolve the restricted adaptability in complex scenarios. Another question is whether entities can be redefined in MDSIM. If so, then the customizability in entity type for a released DES program is also improved. As shown in Table 17, two sub-experiments are illustrated in the following section to respond to these two questions separately.

Table 17. Two sub experiments are used to answer questions about AEMF-DES to improve the limitation of released DES programs.

| Sub Experiments | Questions   | Corresponding Limitations for a typical released DES Program |
|-----------------|---|--|
| 1a              | Does AEMF-DES allow end users to propose a related scenario to the topical theme but not predefined in program? | Flexibility in scenario definition.                          |
| 1b              | Does AEMF-DES allow end users to customize entity definition to model new battle units?                         | Flexibility in scenario definition.                          |

## 1. Sub-Experiments and Results

### a. Experiment 1a: Accommodation of User-Defined Scenarios

AEMF-DES, which is the governing designed framework of MDSIM, applies FTA to restore the primary principles of a given theme, and PES to describe user-defined scenarios. No specific scenarios are considered during software development. This sub experiment attempts to demonstrate that MDSIM can handle a variety of scenarios proposed by end users.

After starting MDSIM, no scenario is predefined. The new scenario offered for this sub experiment is shown in Figure 40. A defender's radar station locates in the center area while an attacker's missile attempts to fly across its coverage to destroy a warehouse in the right area. The radar station is supposed to detect the missile when the missile is within the coverage. Entities attributes are shown in Table 18.

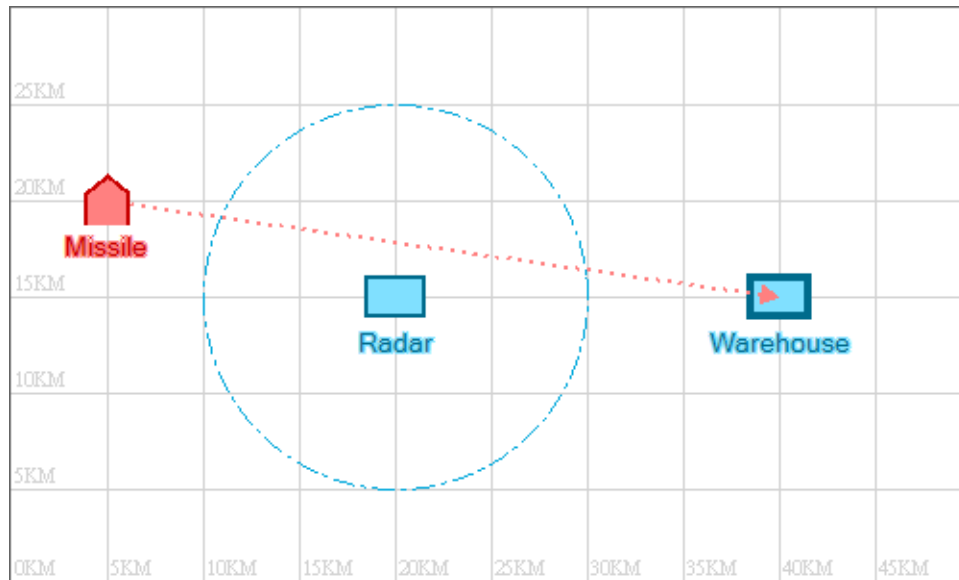


Figure 40. In this user-defined scenario, a missile attempts to fly across the coverage of a radar station.

Table 18. Attributes of entities in experiment 1a.

| Affiliation | Entity   | Part  |
|-------------|--|---|
| Attacker    | Missile<br>Location: 5, 20 Km<br>Speed: 330 m/s<br>RCS: 1 m <sup>2</sup> | None  |
| Defender    | Radar<br>Location: 20, 15 Km   | PulseRadarPart<br>Coverage radius: 10 Km<br>Frequency: 9.375 GHz<br>Bandwidth: 0.512 MHz<br>Transmitter Power: 20 W<br>Transmitter Scan Time: 5 Second/Round<br>Transmitter Antenna Beam: 4 degrees<br>Transmitter PRF: 1000 Hz<br>Transmitter Antenna Gain: 30 dB<br>Receiver Antenna Gain: 30 dB<br>Receiver Temperature: 290 K degree<br>Receiver Noise Factor: 1<br>Receiver Coherent: Yes<br>Minimum SNR: -30 dB |
|             | Warehouse<br>Location: 20, 15 Km   | None  |

Figure 41 shows that after three “ShowUp” events, the defender’s entities are set and the missile entity starts to move. A event “AddTargetCovered” occurs when missile enters the radar’s coverage. Detection events including “Detect” and ProduceDectioMsg” take place repeatedly until the missile leaves the coverage. Then the missile reaches its destination and destroys the warehouse. For readers’ convenience, an event diagram is made according to the event list in Figure 42. The final summary of model state is shown in Figure 43.

| DES engine event queue |                      |                     |  |
|------------------------|----------------------|---------------------|--|
| Time                   | Entity(Part)         | Event               | Message  |
| 00:00:00.00            | Radar                | ShowUp              | Radar begins at{X=20000, Y=15000}.   |
| 00:00:00.00            | Warehouse            | ShowUp              | Warehouse begins at{X=40000, Y=15000}.   |
| 00:00:00.00            | Missile              | ShowUp              | Missile begins at{X=5000, Y=20000}.<br>Missile is in {X=5000, Y=20000} and forward to {X=40000, Y=15000}   |
| 00:00:18.07            | Radar-PulseRadarPart | AddTargetCovered    | Radar-PulseRadarPart has added Missile into target list.<br>Radar-PulseRadarPart starts to detect targets. |
| 00:00:21.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:00:22.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:00:26.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:00:27.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:00:31.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:00:32.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:00:36.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:00:37.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:00:41.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:00:42.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:00:46.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:00:47.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:00:51.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:00:52.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:00:56.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:00:57.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:01:01.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:01:02.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:01:06.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:01:07.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:01:11.70            | Radar-PulseRadarPart | Detect              | Radar-PulseRadarPart has found 1 targets and is processing data.   |
| 00:01:12.70            | Radar-PulseRadarPart | ProduceDetectionMsg | Radar-PulseRadarPart try to sent 1 message(s) to other parts, but no listeners connected.                  |
| 00:01:16.20            | Radar-PulseRadarPart | LostTargetCovered   | Radar-PulseRadarPart has lost Missile<br>Radar has stoped to detect targets.                               |
| 00:01:47.13            | Missile              | BeAtNavigationPoint | Missile attacks Warehouse.   |

Figure 41. Simulation result (excerpted from detailed simulation event log) shows that the radar station can detect the missile within the coverage region.

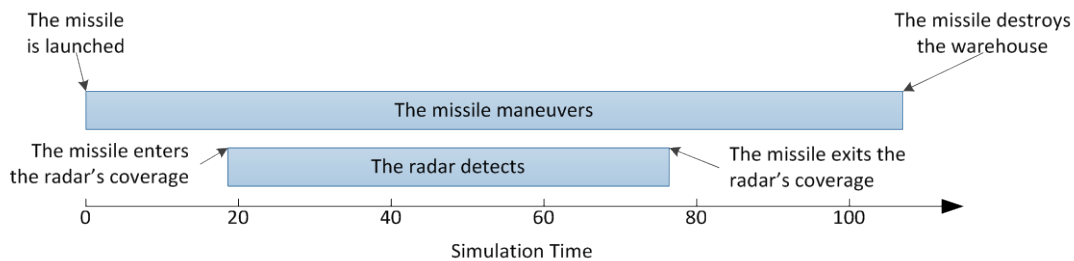


Figure 42. Before it destroys the warehouse, the missile traverse the radar's coverage (subjected to geometry limitation) (also see Figure 44 for more detail).

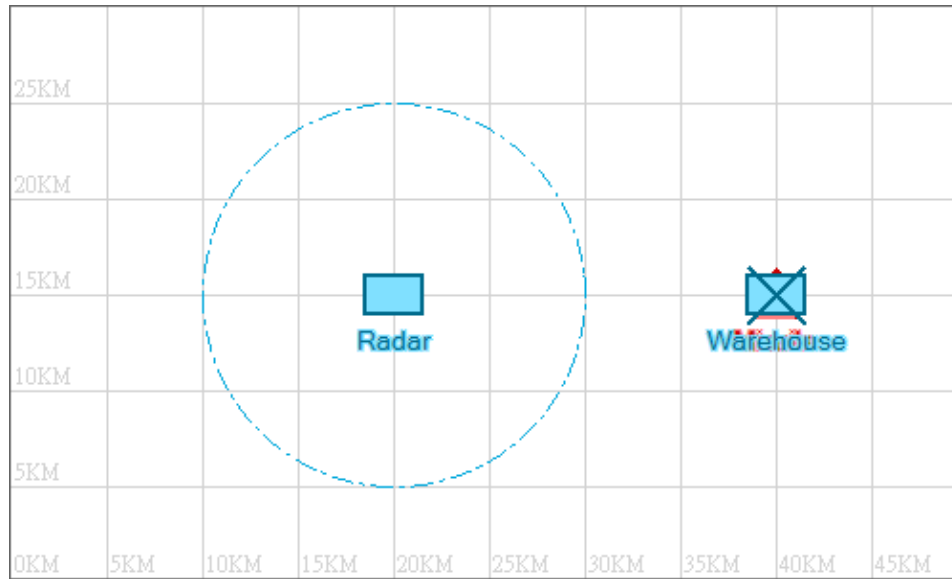


Figure 43. Attacker’s missile arrives at its destination and destroys the warehouse.

Figure 44 also shows the detection signal of the radar when the missile is in radar’s range. The detection signal is in a form of signal-to-noise ratio (SNR) (see equation ( 5 )). It is measured in decibel (dB), equals to  $10 \times \log_{10}(\text{value})$ . Note that the SNR rises while the missile approaches the radar and decrease while it leaves the radar.

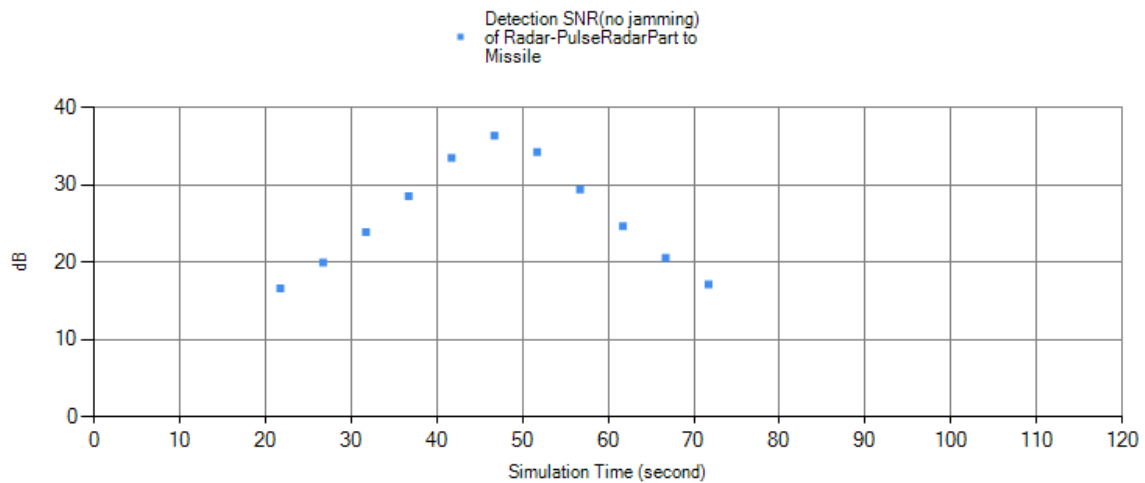


Figure 44. While the missile is in the radar’s range, the radar can detect its existence due to sufficient SNR predicted by scenario geometry (also see Figure 42 for simulation summary).

This sub experiment demonstrates that MDSIM can handle scenarios which is not predefined but provided by end users. During the simulation process with possible scenarios belonging to MDW, MDSIM can manage the events and generate simulation results without additional event analysis efforts by simulation analysts.

Note that in this simulation all EM-related simulation values do not benefit from high-fidelity propagation or attenuation but do reflect geometry trajectory changes and probabilistic effects.

***b. Experiment 1b: Customizable Entity***

To let end users customize entities to model combat units in future battlefields, MDSIM utilizes FTA to represent the fundamental agent types and has PES to define new entities based on the FTA. This sub experiment attempts to demonstrate the applicability of this approach.

The scenario begins with the same setting of the previous sub experiment in that the radar entity can detect and generate detection messages while the warehouse has no part to provide functions. In this experiment, a new sensor part is added to the warehouse entity (see Figure 45). Figure 46 show that after the warehouse entity is added with a new sensor part, a circle to represent the detection coverage shows up. Now the warehouse can act like the radar to detect the missile. The entity setting is shown in Table 19.



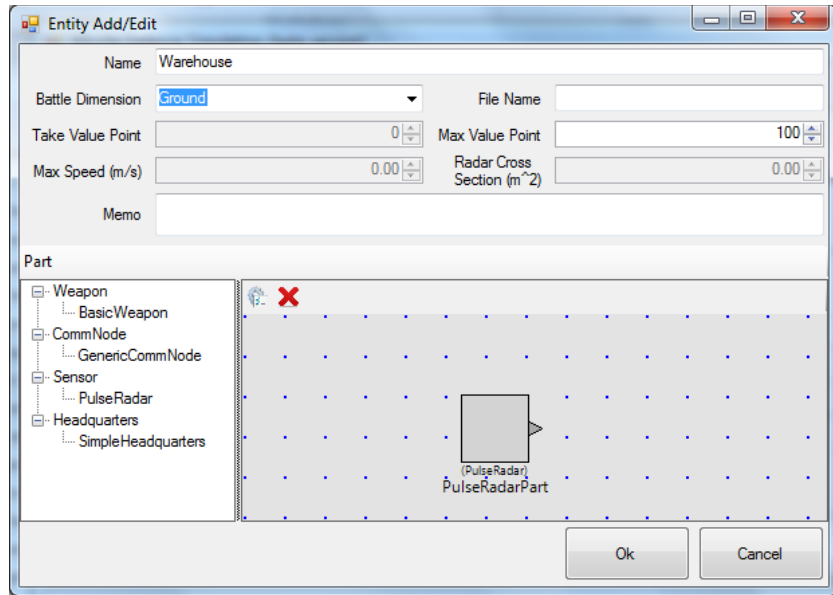


Figure 45. Communication equipment is added into the radar station.

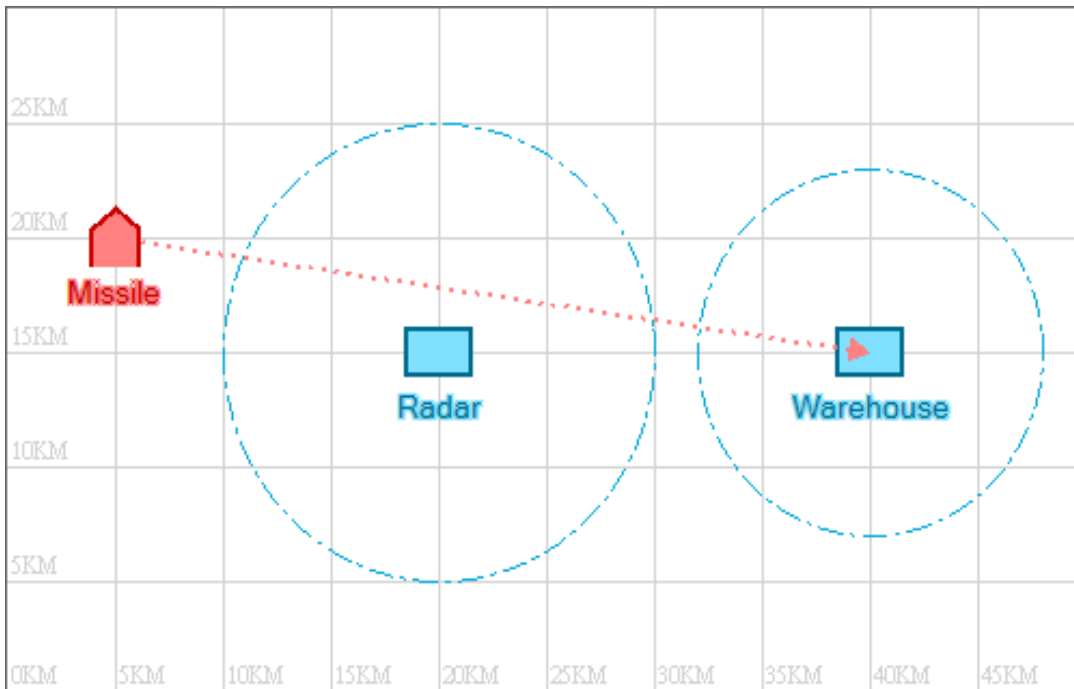


Figure 46. After the warehouse entity is added with a new sensor part, it now can act like the radar to detect the missile.

Table 19. A new PulseRadarPart is added to the warehouse entity to testify the entity customizability.

| Affiliation | Entity                           | Part  |
|-------------|----------------------------------|---|
| Defender    | Warehouse<br>Location: 20, 15 Km | <b>PulseRadarPart</b><br>Coverage radius: 8 Km<br>Frequency: 9.375 GHz<br>Bandwidth: 0.512 MHz<br>Transmitter Power: 1W<br>Transmitter Scan Time: 2 Second/Round<br>Transmitter Antenna Beam: degrees<br>Transmitter PRF: 1000 Hz<br>Transmitter Antenna Gain: 30 dB<br>Receiver Antenna Gain: 30 dB<br>Receiver Temperature: 290 K degree<br>Receiver Noise Factor: 1<br>Receiver Coherent: Yes<br>Minimum SNR: -30 dB |

The result is shown in Figure 47. After the missile traverses the range of the radar, it enters to the range of the modified warehouse which also has a sensor part to detect it. The final model state is shown in Figure 48.

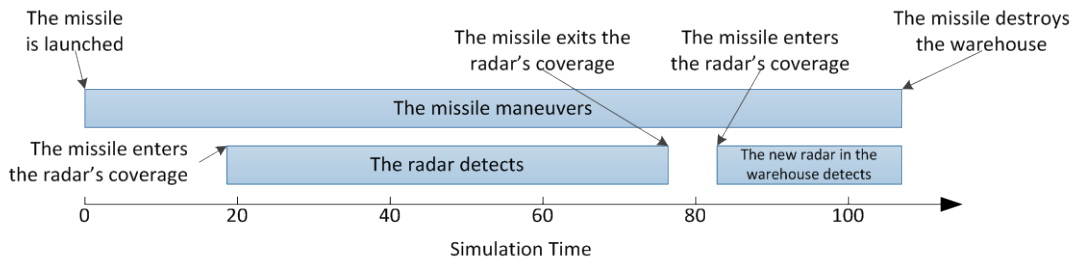


Figure 47. After the missile traverses the coverage of the radar, it enters the range of the modified warehouse which also has a radar sensor part (subjected to geometry limitation)(also see Figure 49 for more detail).

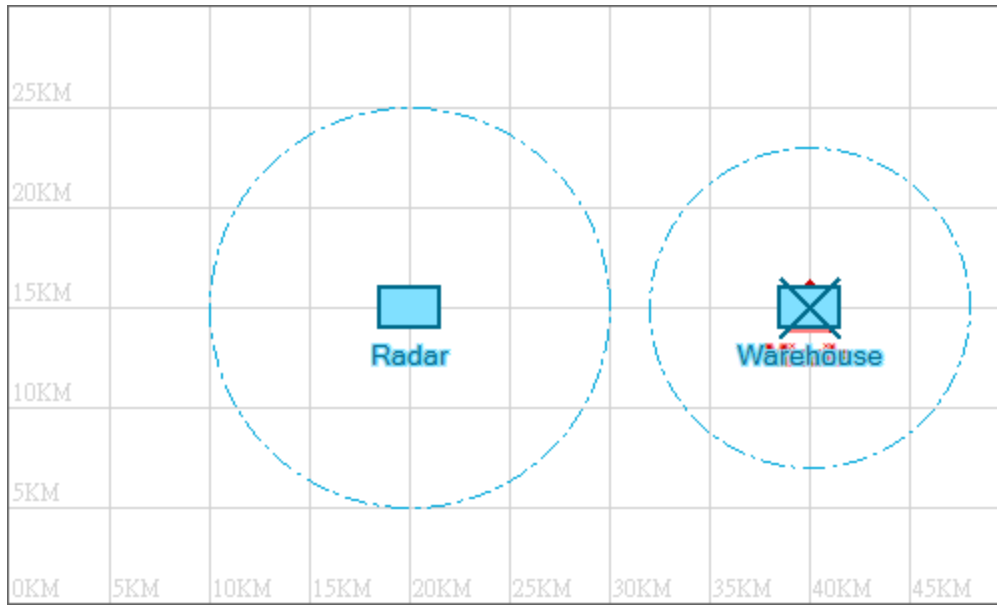


Figure 48. The attacker’s missile arrives at its destination and destroys the warehouse.

The detection SNR of the two sensor parts in the radar and warehouse are shown in Figure 49. While the missile is in the radar’s coverage, the radar gets detection signal as experiment 1a. After the missile enters the coverage of the new sensor in the warehouse, the detection SNR increase as the missile approaches the warehouse.

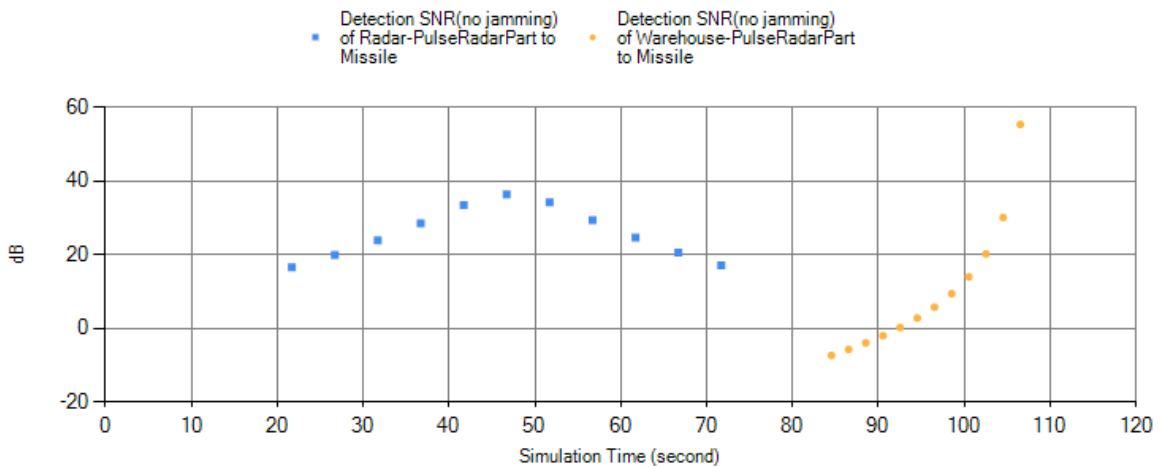


Figure 49. The sensor part in the warehouse entity also provides detection results (also see Figure 47 for simulation summary).

In this sub experiment, the warehouse entity is modified with additional sensor part which then detects the missile as well as the radar. This sub experiment demonstrates that MDSIM can let end users customize entities based on FTA.

## **2. Conclusions**

Through this experiment, it has been demonstrated that MDSIM can successfully accommodate previously undefined scenarios of the given theme and also has the capability to allow customized entities. As a result, it is proved that AEMF-DES provides improved adaptability in complex scenarios and flexibility in related scenarios.

### **B. EXPERIMENT SERIES 2 : C4ISR PROCESSES IN MISSILE DEFENSE WARFARE**

To review the C4ISR process in this simulation program, several processes in following list must be verified. Note that since a computer process has been embedded in each digital information process, it has been simulated implicitly within entity/part agents.

- Surveillance & Reconnaissance (S&R)
- Communication (Comm.)
- Intelligence and Command (I&C)
- Control

The following steps begin with a simple scenario. New properties are added gradually to illustrate these C4ISR functions: (1) a defender radar station and a communication link between the radar and a headquarters are set to detect an invasive missile. Detection should occur while the missile is within the radar's coverage followed by a message being transmitted to headquarters; (2) after the message from radar station is delivered to headquarters, intelligence and command processes take place, and a command is generated; (3) subsequently the command is delivered to an anti-aircraft artillery to execute; (4) the invader applies noise to jam the defender's radar system; (5) the invader applies noise to jam the communication among radar and headquarters. (shown in Table 11).

Table 20. Summary of five sub experiments to incrementally introduce C4ISR processes regarding MDW.

| Sub Experiments | Surveillance & Reconnaissance & Communication | Intelligence & Command | Control & Action | Electronic Warfare    |
|-----------------|---|------------------------|------------------|-----------------------|
| 2a              | √   |                        |                  |                       |
| 2b              | √   | √                      |                  |                       |
| 2c              | √   | √                      | √                |                       |
| 2d              | √   | √                      | √                | Radar Jamming         |
| 2e              | √   | √                      | √                | Communication Jamming |

## 1. Sub-Experiment and Results

### a. *Experiment 2a: Surveillance, Reconnaissance, and Communication*

This sub experiment is dedicated to testing the functions of surveillance, reconnaissance, and communication of MDSIM. Radar attempts to detect and sense a target's existence while invader entities are within coverage. If the target is acquired, a detection message containing information about the target is then generated for a report. As for communication, links between entities can deliver messages so that defender entities can cooperate to defend against incoming missiles.

The layout of this scenario is shown in Figure 50. An invasive missile will maneuver from west to east while a radar station is in its path. A communication link exists between the radar station and headquarters. It is expected that once the missile enters the radar's coverage, the radar will attempt to detect it repeatedly and generate a detection report, which is subsequently delivered to the HQ entity. The entity setting is show in Table 21.

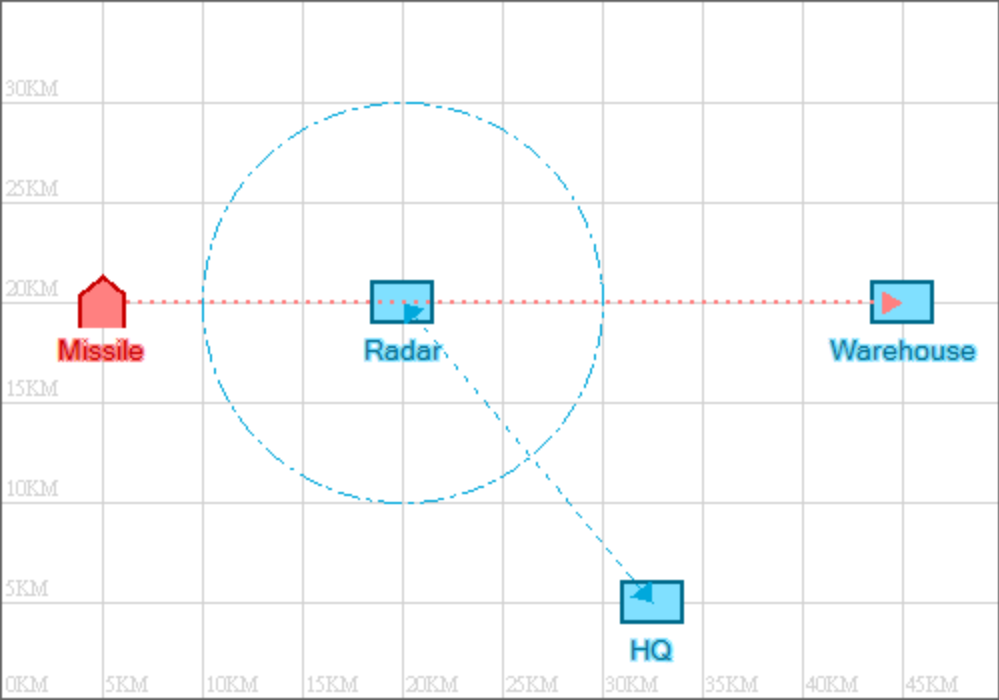


Figure 50. A missile attempts to fly across the coverage of a radar station which has a communication link to HQ.

Table 21. Attributes of entities in experiment 2a.

| Affiliation | Entity   | Part   |
|-------------|--|--|
| Attacker    | Missile<br>Location: 5, 20 Km<br>Speed: 300 m/s<br>RCS: 1 m <sup>2</sup> | None   |
| Defender    | Radar<br>Location: 20, 20 Km   | <b>PulseRadarPart</b><br>Coverage radius: 10 Km<br>Frequency: 9.375 GHz<br>Bandwidth: 0.512 MHz<br>Transmitter Power: 20 W<br>Transmitter Scan Time: 2 Second/Round<br>Transmitter Antenna Beam: 4 degrees<br>Transmitter PRF: 1000 Hz<br>Transmitter Antenna Gain: 30 dB<br>Receiver Antenna Gain: 30 dB<br>Receiver Temperature: 290 K degree<br>Receiver Noise Factor: 1<br>Receiver Coherent: Yes<br>Minimum SNR: -30 dB<br><br><b>GenericCommNodePart</b><br>Frequency: 500 MHz<br>Bandwidth: 10 MHz<br>Transmitter Power: 1 W<br>Transmitter Antenna Gain: 30 dB |
|             | HQ<br>Location: 32.5, 5 Km   | <b>GenericCommNodePart</b><br>Frequency: 500 MHz<br>Bandwidth: 10 MHz<br>Transmitter Power: 1 W<br>Transmitter Antenna Gain: 30 dB<br>Receiver Antenna Gain: 30 dB<br>Receiver Temperature: 290 K degree<br>Receiver Noise Factor: 1<br>Minimum SNR: -10 dB  |
|             | Warehouse<br>Location: 20, 20 Km   | None   |

The simulation result is shown in Figure 51. After the missile enters the coverage of the radar, the radar performs detection periodically and delivers detection message to the HQ via the communication network. A snapshot showing when the missile is within the radar's coverage is shown in Figure 52, and its status at the end of the simulation is shown in Figure 53.

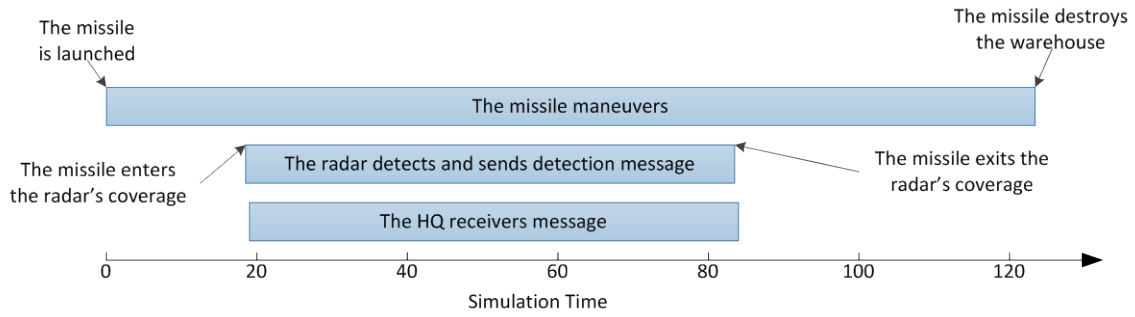


Figure 51. Simulation event-log results show that the user-defined scenario is been simulated functionally (subjected to geometry limitation) (also see Figure 54 for more detail).

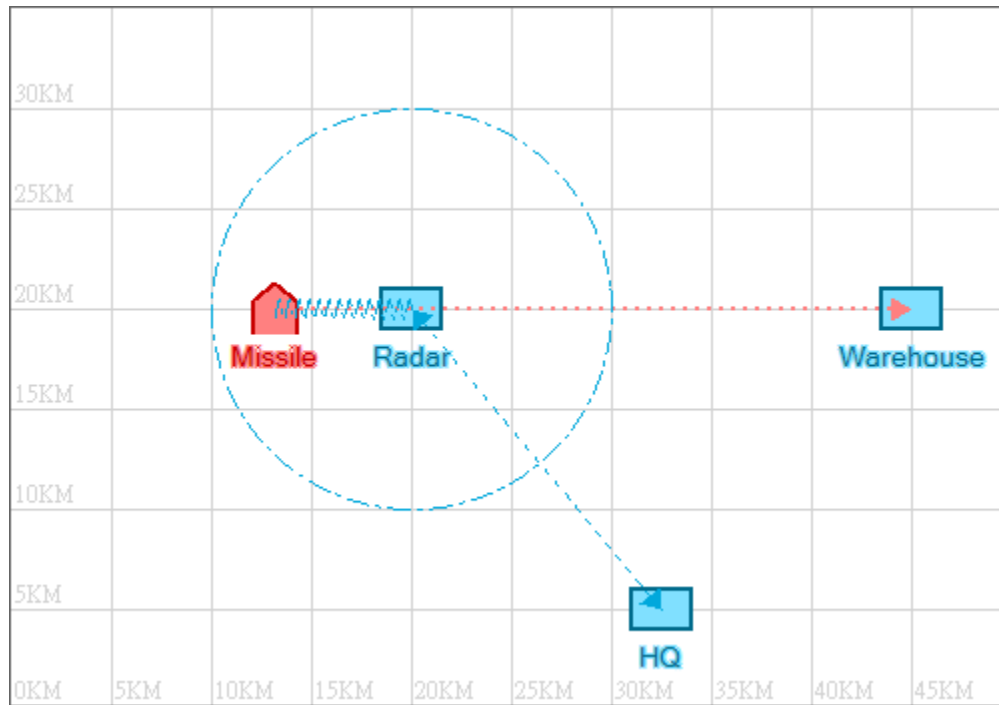


Figure 52. During a portion of the simulation, the missile is flying within the coverage of the radar.



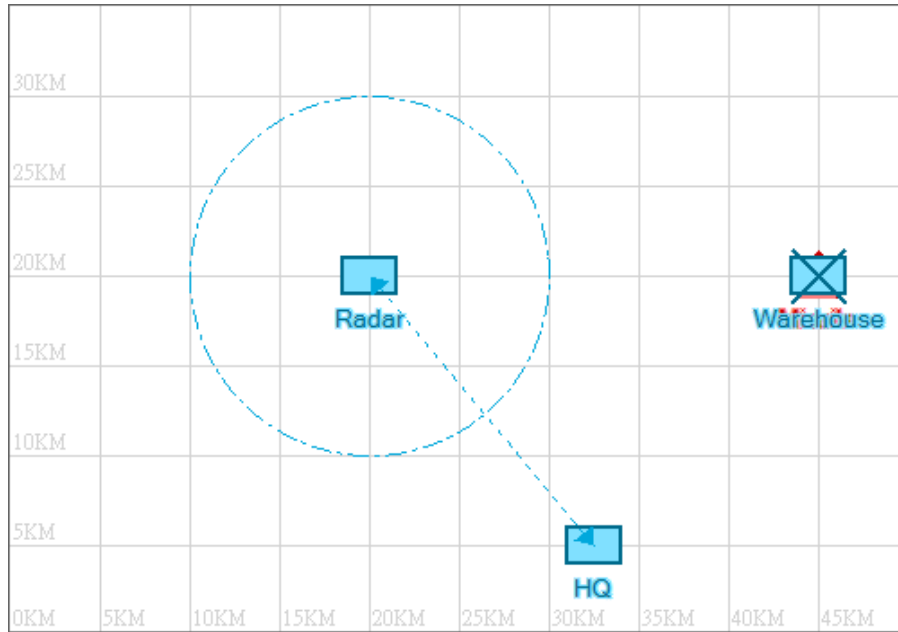


Figure 53. The missile finally stops at its destination, the warehouse.

Figure 54 contains two diagrams. The top one represents the detection SNR of the radar. Every time after the radar performs detection, it delivers a message via the communication network to the HQ entity as shown in bottom diagram.

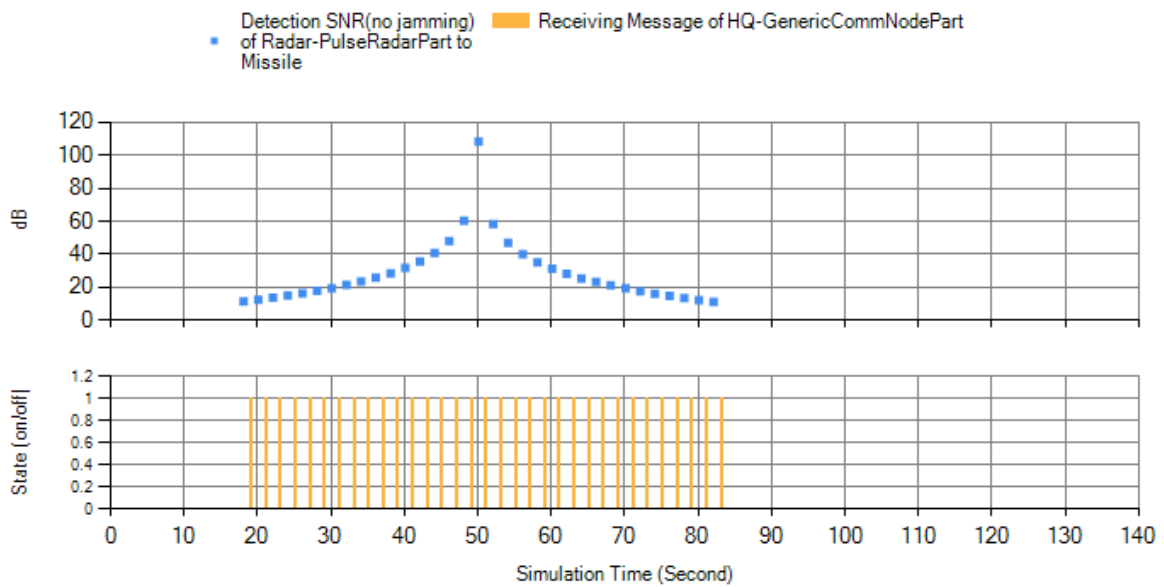


Figure 54. The HQ gets message after the radar perform detection (also see Figure 51 for simulation summary).

In this experiment, the radar station successfully performs detection functions and generates reports through its communication link to the headquarters. It has been demonstrated that the surveillance, reconnaissance, and communication functions are implemented in this MDSIM.

***b. Experiment 2b: Intelligence and Command***

Intelligence and command are performed by the headquarters part type in MDSIM. Every time a new message arrives at headquarters, it is put into an intelligence pool that can categorize and sort messages. If no command process is running, a new command process is initiated and triggered. It is supposed to take some time to generate a command to guide weapon systems. If a command process is running when a new message arrives, the next command process is scheduled by referring the latest message.

In Figure 55, all entities in this scenario are all the same as in the previous one except a headquarters part is added into the HQ entity, which is now capable of processing incoming message and making decisions in response to possible threats. The initial entity setting is show in Table 21.

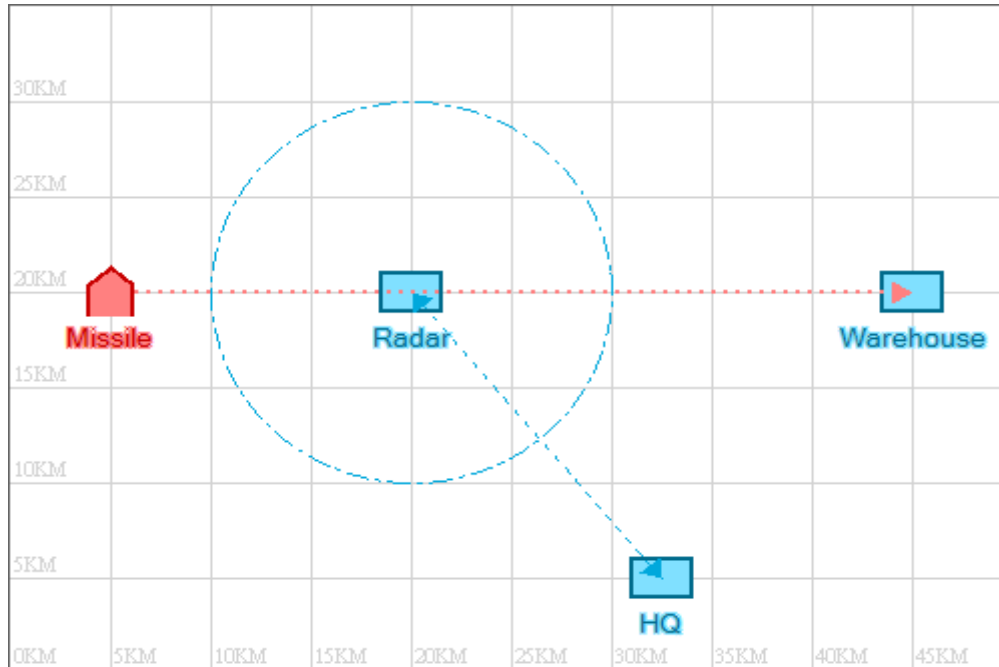


Figure 55. A missile attempts to fly across the coverage of a radar station which has a communication link to an HQ with C2 processors.

Table 22. A Headquarters part is added to experiment 2b.

| Affiliation | Entity                     | Part   |
|-------------|----------------------------|--|
| Defender    | HQ<br>Location: 32.5, 5 Km | GenericCommNodePart<br>Frequency: 500 MHz<br>Bandwidth: 10 MHz<br>Transmitter Power: 1 W<br>Transmitter Antenna Gain: 30 dB<br>Receiver Antenna Gain: 30 dB<br>Receiver Temperature: 290 K degree<br>Receiver Noise Factor: 1<br>Minimum SNR: -10 dB<br><br>SimpleheadquartersParts<br>Decision Making Time: 5 seconds |

Simulation results are shown in Figure 56. Similar to the previous simulation, the missile is detected after flying into the coverage of the radar station. First the message describing target information is sent to command. This time the HQ makes a decision after receiving a new detection. During the time it takes to make a decision, all messages received are postponed until the next decision making period. The intelligence process in this mode will store and sort all message to make sure each initialization of the command process always has the latest information.

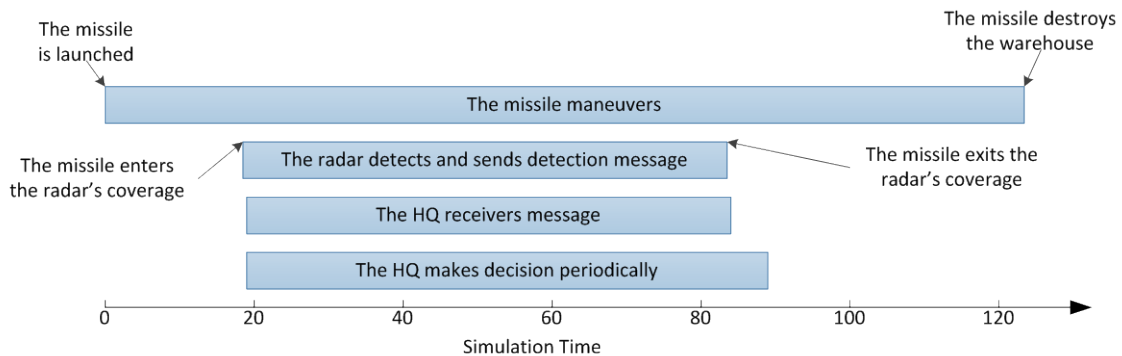


Figure 56. HQ makes decisions when receiving new detection reports (subjected to geometry limitation)(also see Figure 57 for more detail).

Figure 57 contains 3 diagrams. Similar to the previous sub experiment, the top plot is the detection SNR of the radar, the middle plot stands for the HQ receiving messages after radar detection. The bottom plot shows that after receiving a message from the radar entity, the HQ starts to make decision according to the latest intelligence. Note that when a decision is being made, all new incoming messages are postponed until next decision making. The HQ does not stop the decision making loop until no more new message are sent by the radar.

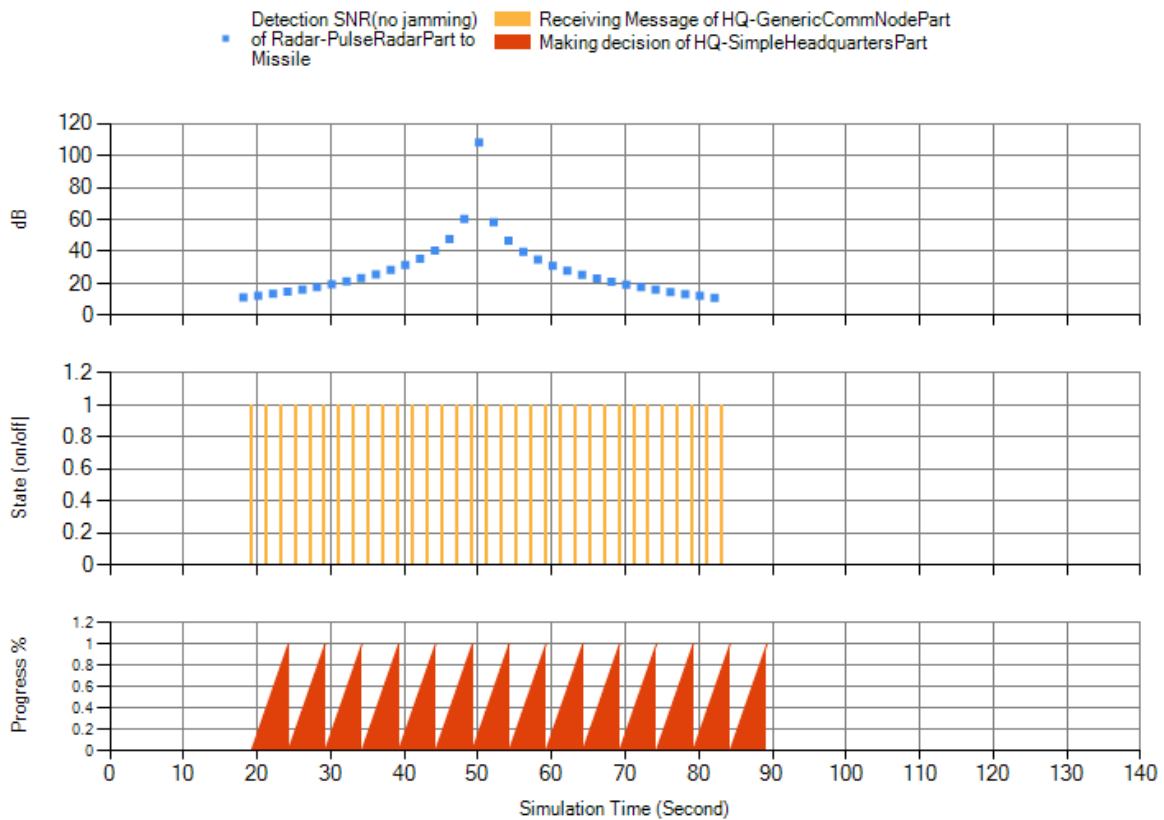


Figure 57. The HQ makes decisions periodically after receiving message from the radar (also see Figure 51 for simulation summary).

In this simulation, after the detection is generated by the radar station and delivered to the HQ entity, a message is processed to generate a command. Due to the time needed when processing the information, new messages might arrive when the current commanding process is in progress. The intelligence process here is designed to store and sort information to support subsequent command processes. Thus, it has been

proved that MDSIM can simulate the information process flow of intelligence and command.

**c. Experiment 2c: Control & Action**

Control in C4ISR stands for the approach used to manage suborders. In our design, this is modeled by simulating the process for the command from headquarters part to be delivered to weapon parts. By authorization from headquarters parts and detection messages generated from sensor parts, weapon parts are then able to engage with targets. Weapon parts can attack invasive entities at every fire interval with a Probability of Kill.

In this scenario, Anti-Aircraft Artillery (AAA) is deployed in addition. Detection messages are sent not only to headquarters but also to AAA. Commands generated from the HQ are delivered to AAA to guide its action. AAA is supposed to intercept the missile when at least a detection message and a grant command are received. The setting of this scenario is shown in Figure 58 and Table 23.

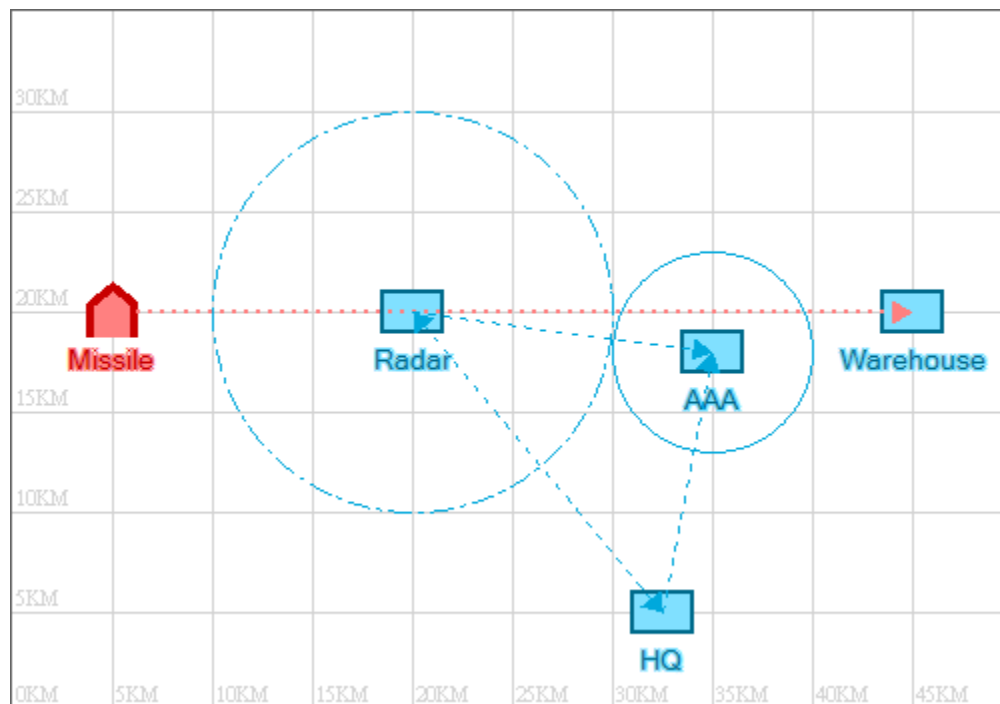


Figure 58. Radar, headquarters, and an AAA cooperate to intercept an incoming missile.

Table 23. An AAA entity with a Weapon part and a CommNode part is added into experiment 2c

| Affiliation | Entity                     | Part  |
|-------------|----------------------------|---|
| Defender    | AAA<br>Location: 35, 18 Km | <b>BasicWeaponPart</b><br>Detection Message Valid Time: 10 seconds<br>Coverage Radius: 5 Km<br>Fire Interval: 1 second<br>Bullet Speed: 1000 m/s<br><br><b>GenericCommNodePart</b><br>Frequency: 500 MHz<br>Bandwidth: 10 MHz<br>Transmitter Power: 1 W<br>Transmitter Antenna Gain: 30 dB<br>Receiver Antenna Gain: 30 dB<br>Receiver Temperature: 290 K degree<br>Receiver Noise Factor: 1<br>Minimum SNR: -10 dB |

Simulation results show that before the missile enters the coverage of the AAA, the AAA can receive detection messages from the radar and a grant command from headquarters. Once the missile is exposed to the AAA, the AAA shoots several rounds referring to the detection information given by the radar. The missile is then intercepted successfully (Figure 59). Figure 60 shows that the missile is intercepted by the AAA entity.

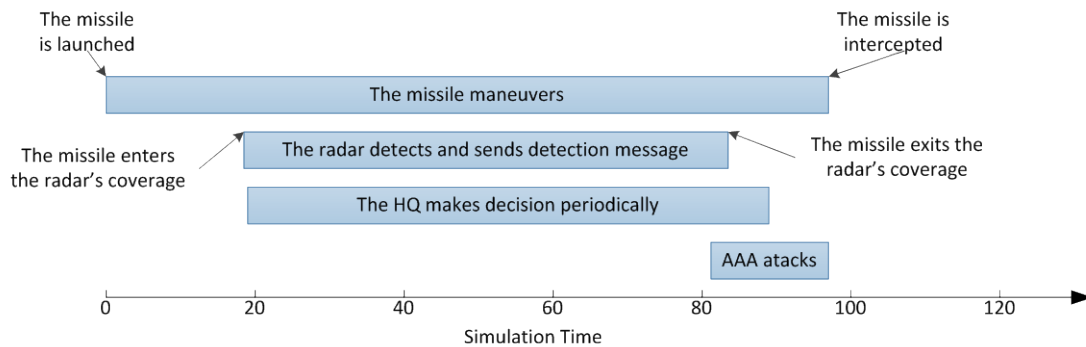


Figure 59. The beginning of the simulation event-log results show that the AAA can receive information from a sensor and headquarters (subjected to geometry limitation) (also see Figure 61 for more detail).

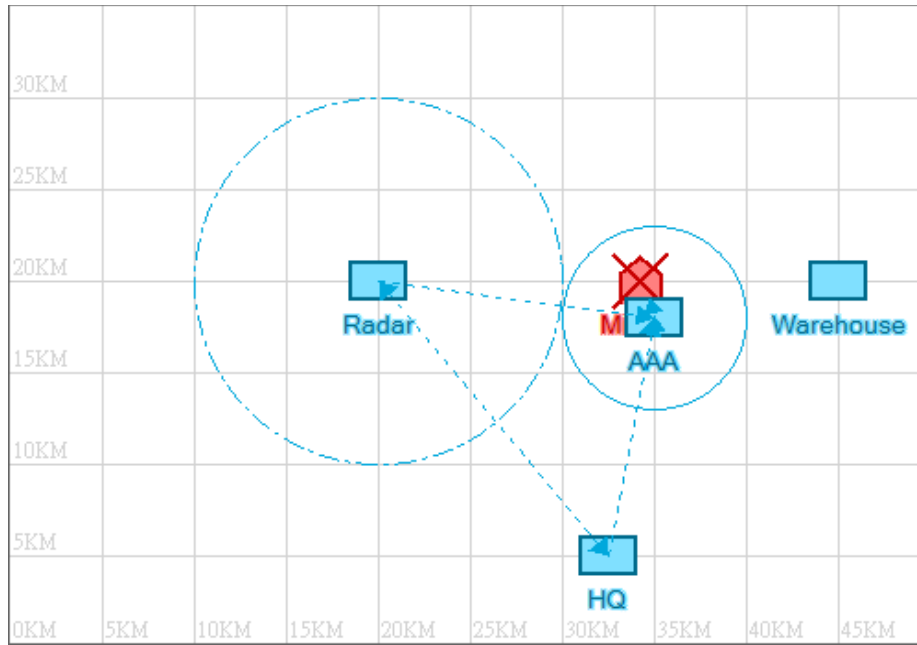


Figure 60. The missile is been intercepted.

As shown in Figure 61, the radar performs detection while the target is in its coverage and then the HQ entity makes a decision. With the detection information from the radar entity and the authority from the HQ entity, the AAA then can intercept the missile.

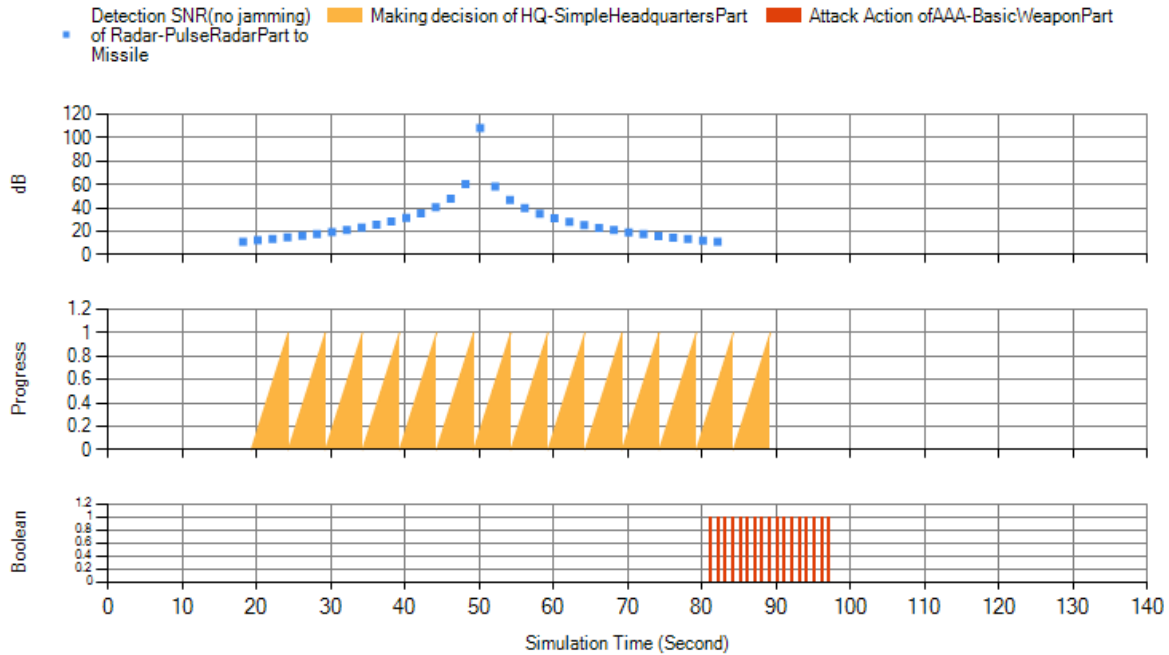


Figure 61. The end of simulation event-log results show that the missile is intercepted by the AAA (also see Figure 59 for simulation summary).

Through this simulation, commands generated from the HQ are delivered to the AAA. This command as well as the detection information sent by the radar then guide the AAA in performing a successful interception operation. This demonstrates the control process and simple action modeled in MDSM.

**d. Experiment 2d: Radar Noise Jamming**

While radar systems use electromagnetic waves to detect targets, it is vulnerable to electromagnetic jamming signals since that value signal can be disturbed. MDSIM provides a noise jamming model which represents the strategy that uses white noise to jam defender radar systems.

In previous scenario, the missile is detected and later taken down by the AAA. Inspired by the state-of-the-art design of Miniature Air-Launched Decoy (MALD) that combines a jammer to an air-launch decoy, in this experiment, the missile is added with a noise jammer (U.S. Department of Defense 2010). By the on-board jammer, the missile now can penetrate the radar coverage with self –protection capability. Simulations



about multiple missiles with advanced jammer functions to invade a more complex defender deployment are possible through additional simulation preparations.

The initial state is shown in Figure 62 and Table 24. When the radar transmits a pulse wave and then attempts to receive echoes from the missile, jamming noise from the jammer part will also be received. As a result the radar of the defender might be blind while the jammer radiates a jamming signal which is strong enough to overflow the echo.

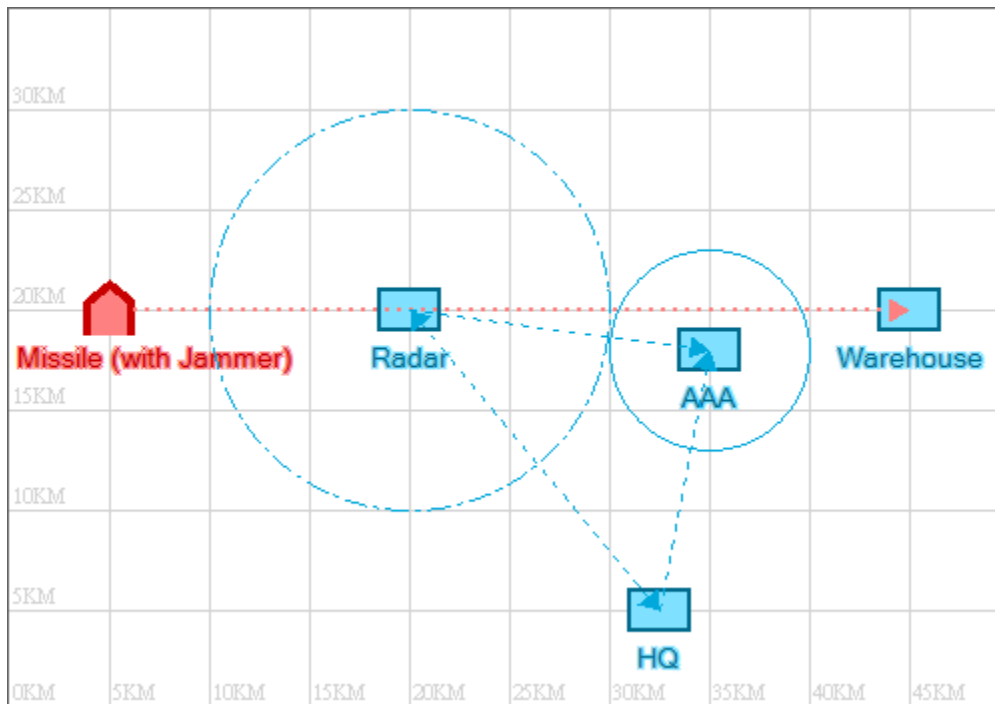


Figure 62. A hostile missile with jammer onboard attempts to penetrate the radar's coverage.

Table 24. The previous missile entity is replaced by a missile entity with a jammer part.

| Affiliation | Entity   | Part  |
|-------------|--|---|
| Attacker    | Missile (with jammer)<br>Location: 5, 20 Km<br>Speed: 300 m/s<br>RCS: 1 m <sup>2</sup> | NoiseJammerPart<br>Frequency: 9.375 GHz<br>Bandwidth: 0.512 MHz<br>Transmitter Power: 1 W<br>Transmitter Antenna Gain: 0 dB |

Simulation results are shown in Figure 63. After the missile enters the coverage of the radar, although the radar attempts to detect it as before, the existence cannot be identified due the invader's jamming strategy. All attempts fail to return a result and therefore no detection messages are generated to offer the command post and the AAA. The missile reaches its destination without encountering any interception event as it has flown through the coverage of the AAA. The final status is shown in Figure 64.

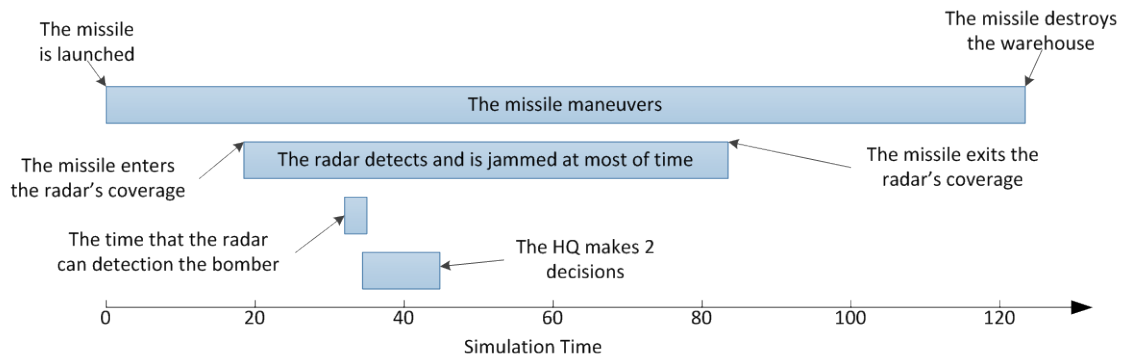


Figure 63. Although the missile traverses the coverage of the radar, the radar can only observe the missile in a short time due to the onboard jammer (subjected to geometry limitation) (also see Figure 65 for more detail).

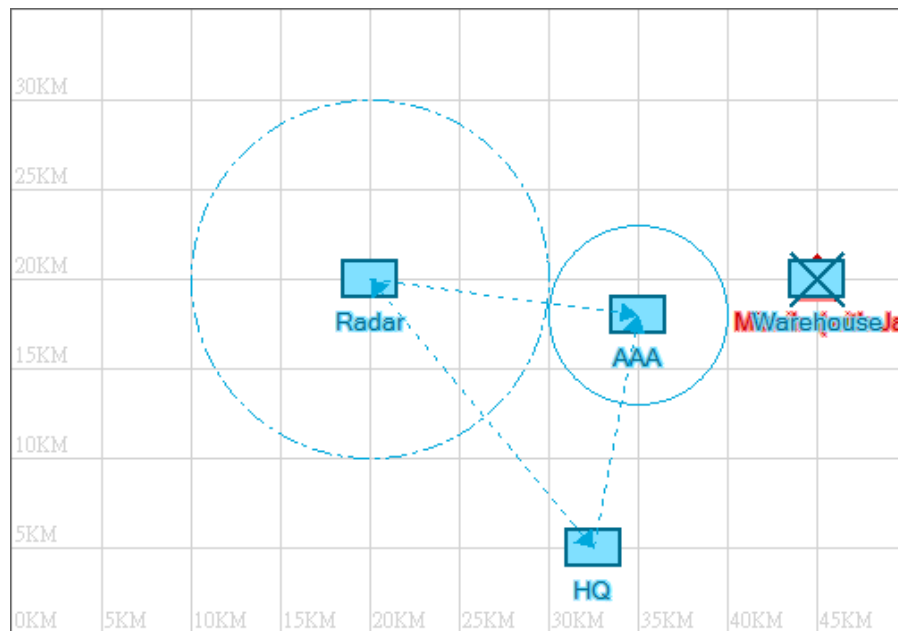


Figure 64. The missile arrives at its target location under cover from a radar jammer.

Figure 65 shows that only when the missile is very close to the radar, the radar can detect it and then offer information to the HQ entity to make decision. However these messages are outdated while the missile enters the AAA's coverage. As a result the AAA cannot attempt to intercept the missile.

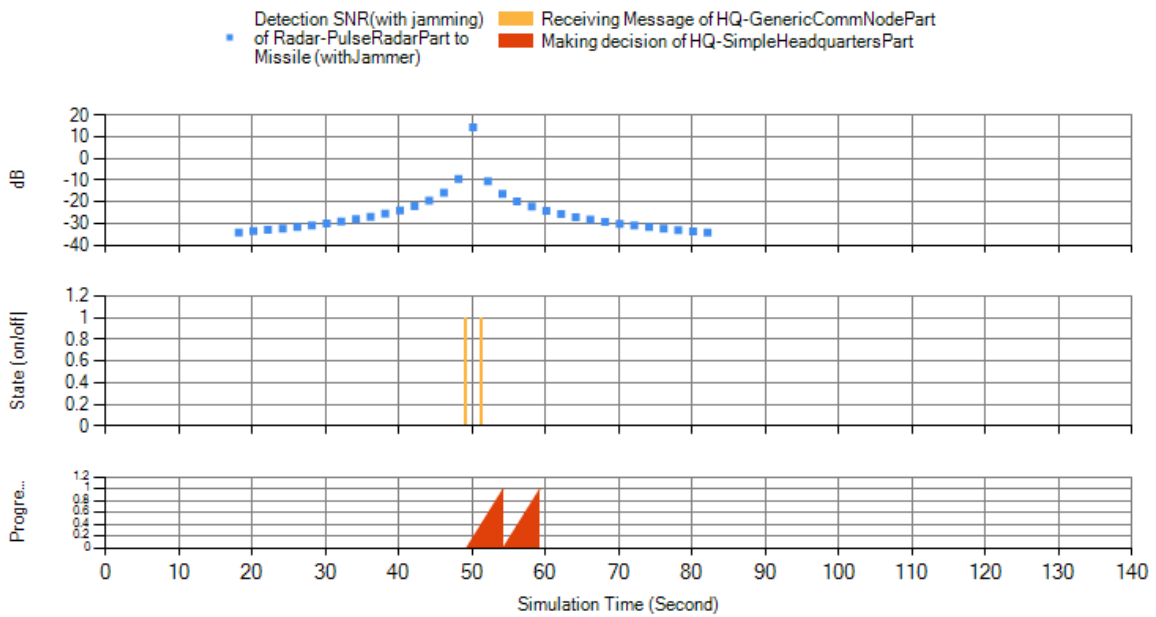


Figure 65. Two successful detections are performed only when the missile is close to the radar (also see Figure 63 for simulation summary).

Figure 66 shows more detail about the jamming to the radar. While the minimum SNR required for detection is -10 dB, the signal power of the radar wave echo is supposed to be above this threshold and helps in target tracking. However due to the jamming effect, the actual signal is degraded about 45dB which makes the missile hard to detect unless it is close to the radar. The crossover range is about 188 meters from the radar transmitter as shown in Figure 67. Note that in this simulation, all simulation values wire relating to communication signal do not benefit from high-fidelity propagation and attenuation.

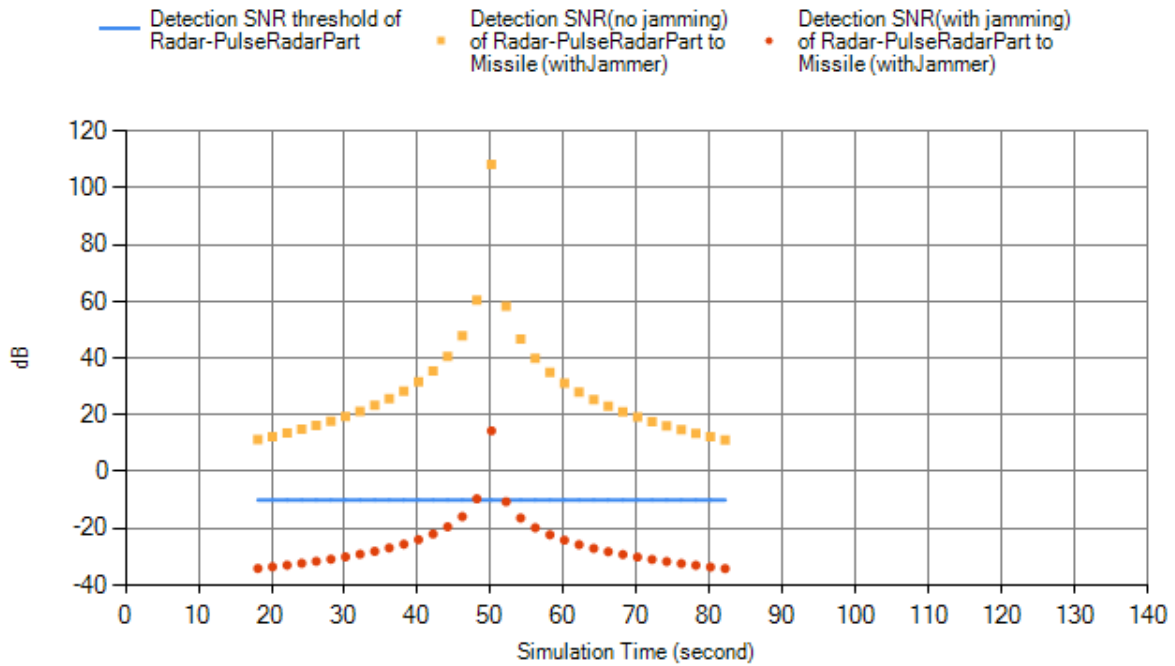


Figure 66. The radar cannot detect the existing missile even it is within coverage.

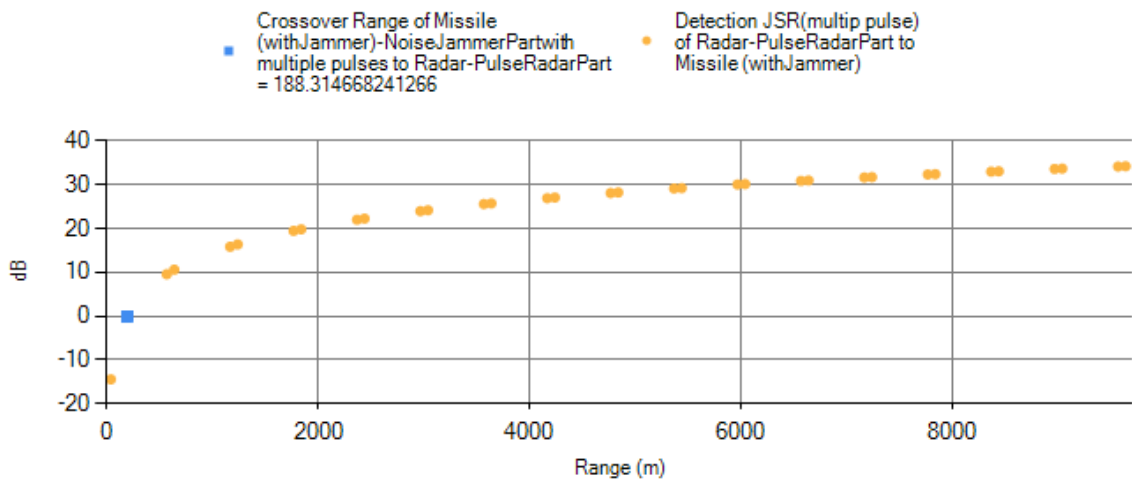


Figure 67. The Jamming/Signal Ratio (JSR) and the crossover range overlaps at about 188 meter from the radar transmitter.

Compared to the results of the previous simulation, the missile in this scenario flies without being intercepted by the defender. By observing the event list, apparently the abnormal behavior of the defender is due to the failure to detect the missile. Without target information, headquarters cannot make any decision since the necessary

information is absent. The AAA cannot intercept because (1) no authorization is given from by command post and (2) no target information is provided from the radar. This simulation demonstrates that the noise jamming to sensor model can be modeled successfully. The simulation analyst upon considering these results might consider changing overall defense configuration to improve the effectiveness of missile defense.

*e. Experiment 2e: Communication Noise Jamming*

Wireless communications utilize electromagnetic waves to deliver message. Therefore, also could be vulnerable to hostile jamming. The way MDSIM models communication jamming is similar to the algorithm of radar jamming. When a communication node receives signal from another communication node, the signal strength, background noise, and possible hostile jamming noise are taken into consideration. Communication links may become blocked if signal is overwhelmed by the combination of thermal and hostile noise.

This scenario is based on experiment 2c as shown in Figure 68 and Table 25. An attacker's communication jammer is added to jam the defender's communication. While the radar detects the target and attempts to deliver messages to the HQ, the communication link remains prohibited from the attacker's communication jammer. Due to the unawareness of the HQ, the missile is not intercepted by the AAA and finally can destroy the warehouse.

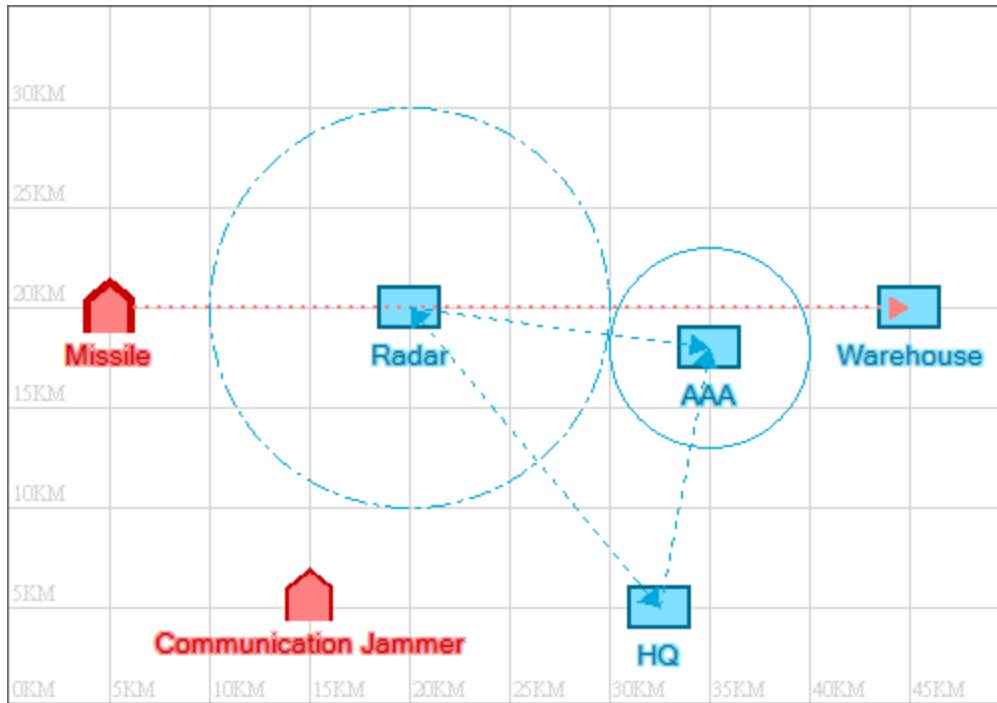


Figure 68. A hostile communication jammer attempt to block communication among defender units.

Table 25. A communication jammer is added to block the communication between the radar entity and the HQ entity.

| Affiliation | Entity   | Part  |
|-------------|--|---|
| Attacker    | <b>Communication Jammer</b><br>Location: 15, 5 Km<br>Speed: 300 m/s<br>RCS: 1 m <sup>2</sup> | <b>NoiseJammerPart</b><br>Frequency: 500 MHz<br>Bandwidth: 10 MHz<br>Transmitter Power: 10 W<br>Transmitter Antenna Gain: 30 dB |

Figure 69 show the simulation results that although the missile is detected by the radar and then detection messages are generated, the communication link to the HQ entity is not available. Therefore the missile successfully flies through the coverage of the AAA entity and destroy the warehouse (shown in Figure 70).

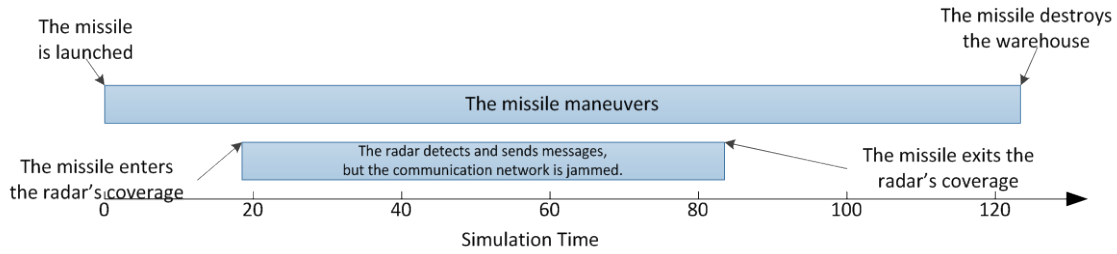


Figure 69. The communication link from the radar entity to the HQ entity is jammed (subjected to geometry limitation) (also see Figure 71 and Figure 72 for more detail).

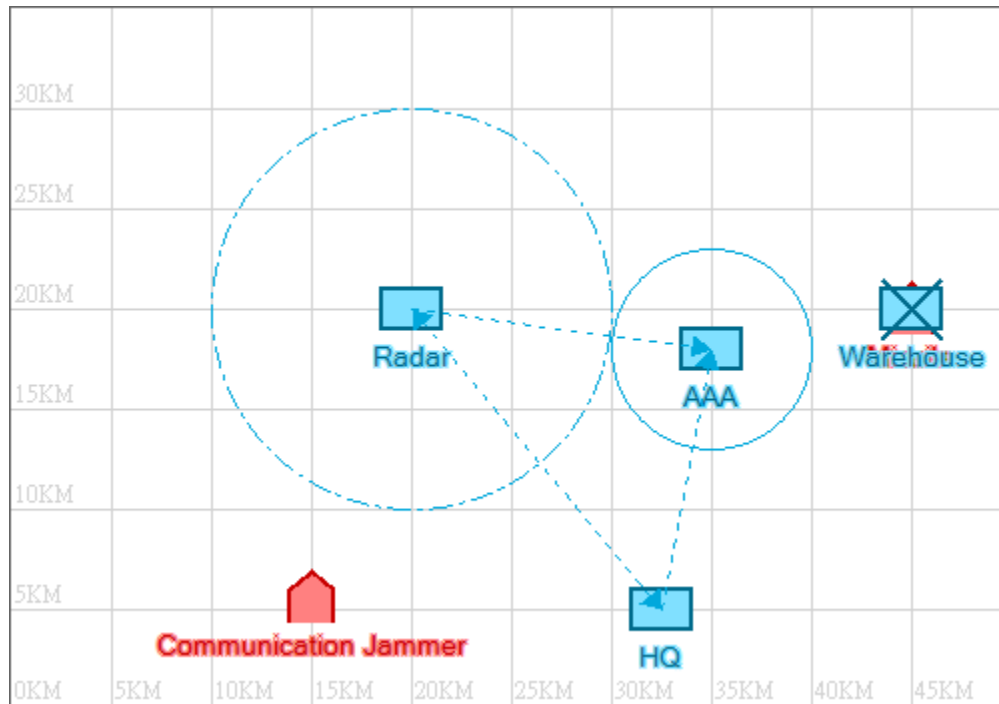


Figure 70. The missile destroys the warehouse under the cover from a communication jammer.

Figure 71 shows that every time the radar entity detects a target, it attempts to distribute a message through the defender's wireless communication network. However, due to the hostile jamming from the attacker's jammer entity, the SNR of the communication signal from the radar entity to the HQ entity is degraded (shown in Figure 72). The HQ therefore is unaware of the invasion and does not give a command to the AAA to activate an interception operation.

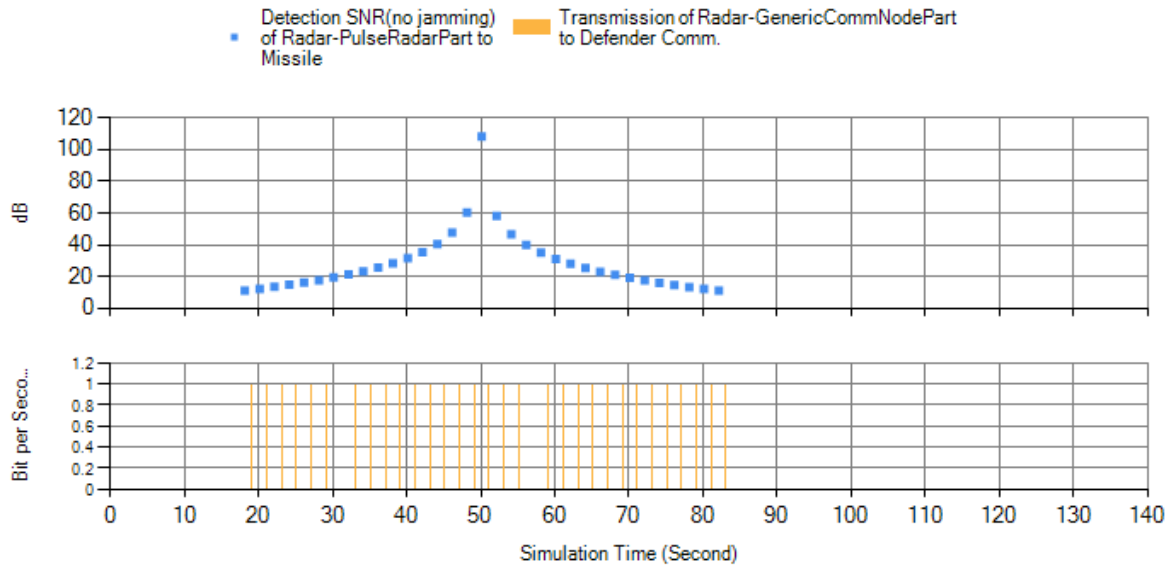


Figure 71. The radar entity attempts to distribute a message every time it detects a target (also see Figure 69 for simulation summary).

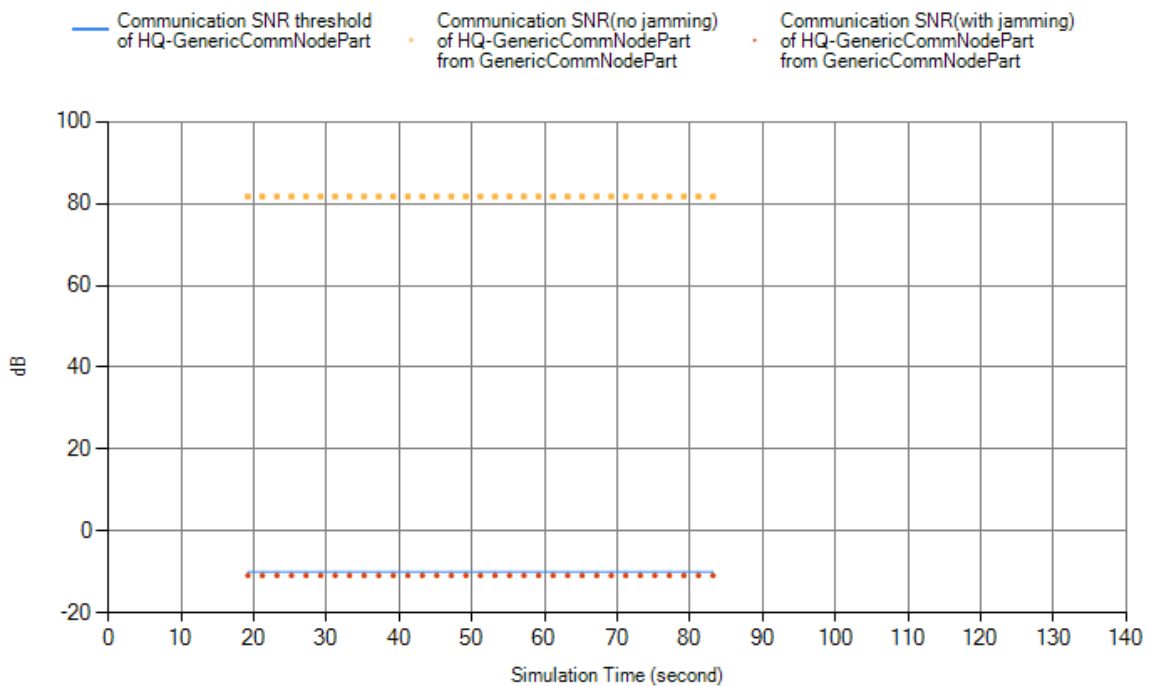


Figure 72. The SNR of the communication signal received by the HQ entity degrades (also see Figure 69 for simulation summary).



In this simulation, the communication among defender entities is disrupted due to the hostile jamming. The command post and the AAA cannot respond to incoming threats since environmental information is cut off. The function in modeling communication jamming of MDSIM is demonstrated.

## **2. Conclusions and Limitation**

In this experiment, the C4ISR processes in MDW have been verified. The surveillance and reconnaissance process are done by sensor parts which can detect targets within coverage and then generate messages describing target information. Communication is modeled by the links/networks formed by communication nodes and can deliver messages among them. Intelligence and command processes happen in headquarters parts while messages about targets are stored and sorted before being used to generate commands. Control is implemented by the delivery of commands to weapon parts and by guiding engagements. Hostile jamming can be set to sensing or communication. Computing takes place in every process that happens during the simulation.

Note that all the electromagnetic-related simulation values such as radar detection and wireless communication links do not include the consideration of high-fidelity electromagnetic (EM) propagation and attenuation.

## **C. SUMMARY**

In this chapter, the feasibility of AEMF-DES is proved. Events raised from agents can be created and scheduled to fire automatically without additional effort by human analysts once the simulation is done. Also, the C4ISR processes of the simulation program developed for these demonstrations are verified to be able to model essential processes in MDW. Sensors, communication nodes, headquarters, weapons, and jammer parts can cooperate together to form a simulation world to benefit military operational analysis.

In the next chapter, additional research in sensor deployment strategy, the  $k$  – coverage rate problem, is investigated. Although the concept has not been integrated within the simulation program, we have disclosed a possible approach for evaluating

sensor deployment with more realistic models. A simple simulation program and an experiment are demonstrated as well.

THIS PAGE INTENTIONALLY LEFT BLANK

## VI. SENSOR ANALYSIS: $k$ -COVERAGE RATE PROBLEM

While performing the modeling and calculation of the  $k$ -coverage rate problem, first introduced in Chapter II, a binary-omnidirectional sensor model is often used. This model is so named because there are only two states, seeing or not seeing, and it contains simple 360-degree coverage. The major advantage of applying this model is its simple structure and ease of calculation. In addition, most types of sensors are probabilistic. The probability of detection decreases as the distance to the target increases. Also quite common, however, is that real-world sensors often use a sector-sensing pattern. Using only binary-omnidirectional sensor models may not offer enough simulation resolution when modeling modern sensors.

The current  $k$ -coverage rate calculation algorithm is discussed in Chapter II. This chapter first introduces the probabilistic-sectorial sensor model and then covers an improved algorithm to accommodate it. Simulation experiments are then provided to demonstrate the feasibility of the proposed algorithm.

### A. PROBABILISTIC AND SECTORIAL SENSOR MODEL

Two of the most important properties of a sensor model are the detection probability and coverage shape. In practice the probability of detection is probabilistic in most sensor types such as radar systems. Also the coverage pattern itself might be in a sectorial shape to pursue better sensing in an interesting direction. This section begins with a discussion about probabilistic sensing, followed by examination of the sectorial coverage shape.

#### 1. Probabilistic Sensing

In a binary sensor model, the target is seen if it is within coverage (as shown in Figure 73). The binary model is widely used in many simulations due to its simple concept and easy application in a simulation program.

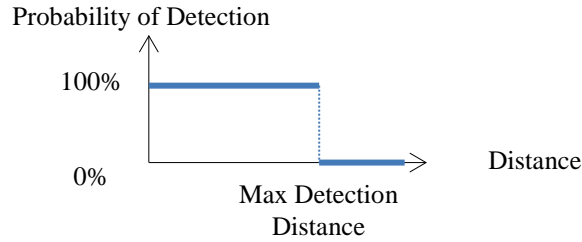


Figure 73. Binary sensor model contains two states: seen (100%) or not seen (0%).

However, most sensors in the real world are stochastic in their detection characteristics. By using active or passive methods, sensors sense environmental signals which contain possible target information. Then the signal is examined to determine whether a target exists. Due to the inevitable noise, the probability of detection is usually probabilistic. An example model whose probability of detection decreases as the distance increases is shown in Figure 74.

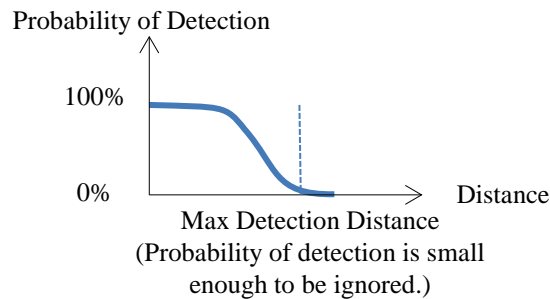


Figure 74. Probabilistic sensor model has a monotonically decreasing probability of detection as the distance increases.

## 2. An Example of Probabilistic Sensor: Pulse Radar System

In a regular pulse radar system, energy in the form of electromagnetic (EM) waves is first emitted. This energy pulse bounces back if a target is encountered. The radar system then determines whether a target is detected by analyzing the echo received. The power of the echo received can be shown as Equation ( 9 ) and signal-to-noise ratio (SNR) as Equation ( 10 ) (Skolnik 2001; Pace 2009; Chen and Pace 2008).

$$P_r = \frac{P_t \sigma_T G_t A_e}{L_{RT} L_{RR} (4\pi R_T^2)^2} \quad (W) \quad (9)$$

$P_r$  : Radar power Received

$P_t$  : Radar power transmitted

$L_{RT}$  : Loss factor of radar transmitter

$G_t$  : Antenna gain

$R_T$  : Range from antenna to target

$\sigma_T$  : Radar cross section (RCS) of target

$A_e$  : Effective area of receiver antenna

$L_{RR}$  : Loss factor of radar receiver

$$SNR = \frac{P_r}{k T_0 B_{Ri} F} \quad (10)$$

$SNR$  : Signal-to-Noise Ratio

$k$  : Boltzmann's constant

$T_0$  : Standard temperature 290K

$B_{Ri}$  : Receiver bandwidth

$F_R$  : Noise Effector of receiver

Due to the thermal noise the echo signal may be too weak to be visible to the signal processor if its noise energy is too low. Since the thermal noise power is a random distribution, it is rational to figure out that the probability of detection for a target is a probabilistic function of the SNR. In 2008, Qi analyzed related issues and proposed a simulation approach to the detection probability of radar system (shown in Figure 75), which gave a more accurate relation between detection probability and SNR of an echo signal. Therefore, since the SNR is inversely proportional to the quartic of range, the detection probability can be viewed as a probabilistic distribution of the range (Qi et al. 2008).

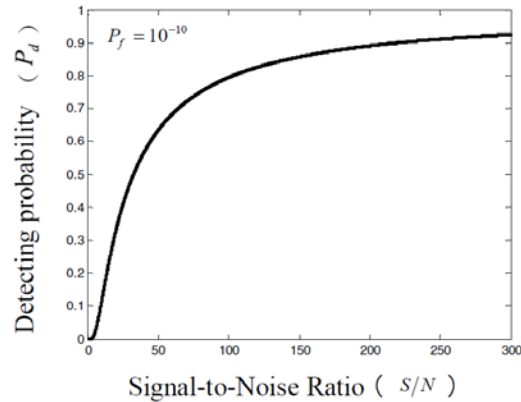


Figure 75. Detection probability vs. SNR (from Qi et al. 2008).

### 3. Sectorial Sensing Coverage

In an omnidirectional sensor model, the coverage is circular in shape. A typical example is the search radar in a battleship. An antenna rotates 360 degrees and forms a circular coverage shape. When using an omnidirectional sensor model, the distance from a sensor to a target is the main consideration while the direction is omitted. This approach is easy, but it also lacks some practical fidelity as well.

It is very common that sensors only focus on a specific direction or sector and ignore received signals in other directions. The possible reasons for this include: (1) sensor constitute, (2) an attempt to extend the detection performance, and (3) terrain shadows in another direction. For instance, most long-range radar stations for missile defense systems face outside a country and have sectorial coverage shapes. Therefore, the necessity of sectorial sensor model exists when high fidelity simulation is needed.

## B. CHALLENGES AND SOLUTIONS

Sheu's method uses four corners to determine the  $k$ -covered state of a grid with binary-omnidirectional sensor models. Two challenges exist when extending it to the probabilistic-sectorial sensor model:

- First, the number of covering sensors of a point within a region is hard to determine since the detection is probabilistic.
- Second, the 4-corners algorithm is not applicable in some extreme situations and sectorial coverage shapes.

The following section first discusses solutions to these challenges, followed by a proposed new algorithm.

## 1. Probabilistic Sensing

Using the probabilistic sensor model seems to offer a challenge for the  $k$ -covered rate problem since the  $k$ -covered state of a point is defined without the uncertainty of detection. In 2012, Wang proposed a numeric definition for a probabilistic sensor model for the  $k$ -coverage problem which, unlike the  $k$ -coverage rate problem, does not focus on the ratio of how much area satisfies the monitoring criteria but just aims to find the latest number of sensors covering a point within a given region, although the  $k$ -covered criterion of a point are identical. Wang proposes that a region is  $k$ -covered when the expected number of covering sensors at all points is at least  $k$  (Wang and Chung 2012):

$$\min_{p \in R} \sum_{s_i \in S} \rho(s_i, p) \geq k \quad (11)$$

$R$  : the given region

$S$  : the set of all sensors

$\rho(s_i, p)$  : the detection probability of sensor  $s_i$  to point  $p$

Due to the identical  $k$ -covered criterion of a point, this research recommends adopting Wang's concept to identify whether a point is  $k$ -covered by if the expected covering sensor number is greater than  $k$ .

## 2. Sectorial Coverage

Sheu's 4-corners algorithm defines a grid as  $k$ -fully-covered if all four corners (monitor points) are covered by at least  $k$  sensors. If only part of them is covered by at least  $k$  sensors, then it is  $k$ -partially-covered. Otherwise, it is not  $k$ -covered. This algorithm is fine with omnidirectional sensor models but might not be applicable in sectorial coverage as well as small sensor coverage. Three situations shown in Figure 59 are examples that the 4-corner algorithm could not handle perfectly since all the coverage of the sensor model does not cover any corners.



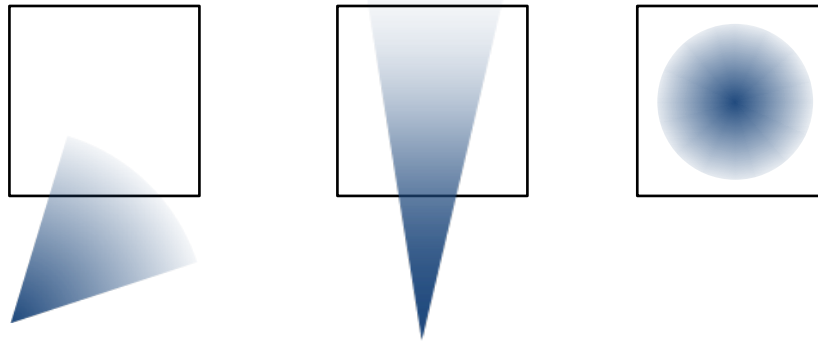


Figure 76. Three possible examples illustrating the limitations of Sheu's 4-corners algorithm.

It has been noted that these limitations arise because four corners are not covered by the sensor models. In other words, four corners to be the monitor points are insufficient. Therefore three additional monitor points, therefore, are recommended to extend the feasibility of the grid scan approach in  $k$ -coverage rate evaluation (shown in Figure 77).

- The interaction point of two sides of sectorial sensor coverage with any sides of a grid. If the sensor model coverage is circular, this monitor point is not necessary.
- The interaction point of the sectorial sensor coverage axis with any sides of a grid. If the sensor model coverage is circular, this monitor point is not necessary.
- The center of each sensor so that a grid with a sensor inside it can be judged whether it is partially covered.

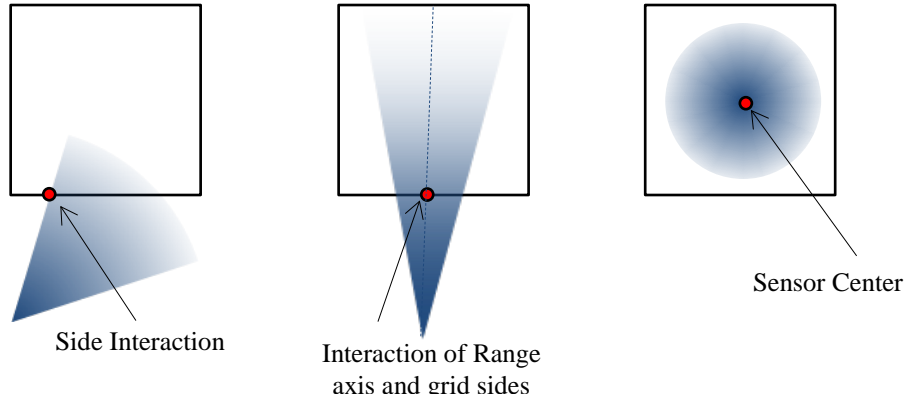


Figure 77. Beside the four corners, three additional monitor points are recommended to make sure all points with the highest number of covering sensors are taken into consideration.

### 3. Improved Algorithm

Based on Sheu’s algorithm, Wang’s definition, and the proposed monitor points, an improved algorithm of grid scan for the  $k$ -covered rate problem is summarized as follows:

- First separate the target region into regular sized grids.
- Determine the covered state of all grids

| Coverage state         | Criterion   |
|------------------------|---|
| $k$ -fully-covered     | If the expected covering sensor numbers of all four corners are greater than $k$  |
| $k$ -partially-covered | If the expected covering sensor numbers of anyone of the following monitor points are greater than $k$ : <ul style="list-style-type: none"> <li>• 4 corners</li> <li>• Location of sensors inside, if any</li> <li>• Interactions of sectorial sensor coverage sides and grid sides</li> <li>• Interactions of sectorial sensor coverage axis and grid sides</li> </ul> |
| Not $k$ -covered       | else  |

- Evaluation error is calculated as:

$$\text{evaluation error} = \sum_{g \in P} |g| / |A|$$

Where  $P$  denotes the set of all  $k$ -partially-covered grids and  $A$  stands for the area of the whole region.

- If the evaluation rate is acceptable, for example less than 10%, then go to next step; otherwise divide uncertain grids into smaller grids and reevaluate repeatedly until the evaluation error is acceptable.

- Then the  $k$ -coverage rate range can be calculated as follows:

$$k\text{-coverage rate range} = \sum_{g \in F} |g|/|A| \sim \sum_{g \in F \cup P} |g|/|A|$$

Where  $F$  denotes the set of all  $k$ -fully-covered grids,  $P$  denotes the set of all  $k$ -partially-covered grids, and  $A$  stands for the area of the whole region.

The relation of this proposed algorithm with Shen’s algorithm, Sheu’s improvement, and Wang’s definition can be shown in Figure 78.

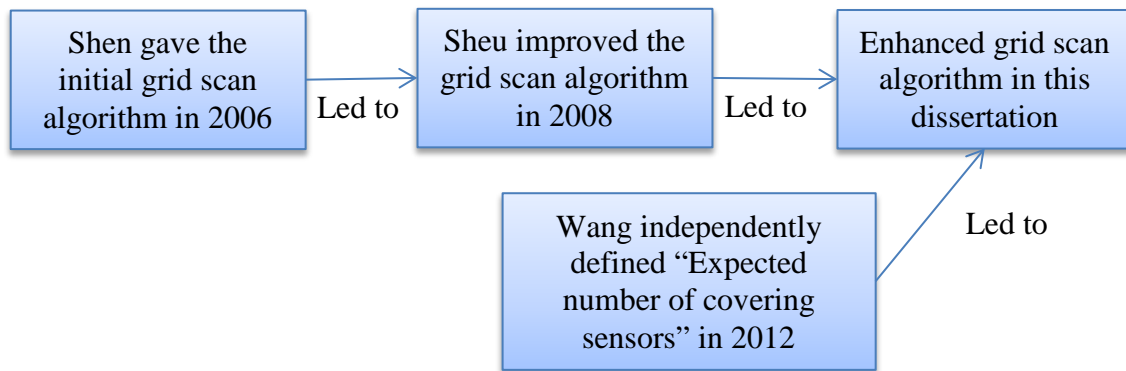


Figure 78. Many works contribute to the proposed algorithm.

## C. EXPERIMENTS

Two experiments are shown in this section. The first one includes three sensors which represent three possible challenges to current calculation algorithm. The second one demonstrates an example that the proposed algorithm can successfully estimate the  $k$ -coverage rate of a scenario including binary, probabilistic, omnidirectional, sectorial sensor properties at the same time.

### 1. Experiment 1

In this experiment, three sensors are used to represent three challenges of the current calculation algorithm shown in Figure 79. The S1 sensor is located just inside a grid that does not cover any one of the four corners. Two side terminals of S2 sensor’s coverage partially covers two grids and also does not cover any corners. S3 sensor’s

coverage directly crosses two grids without covering any corners. Sensor settings are shown in Table 26.

Table 26. Three sensors are used to illustrate challenges for current algorithm.

| Name | Detection Model  | Coverage Shape  | Location | Max Range |
|------|--|-----------------|----------|-----------|
| S1   | Linear probability<br><ul style="list-style-type: none"> <li>• 100% in the sensor's location</li> <li>• 0% in the max range</li> </ul> | Omnidirectional | (10,70)  | 9         |
| S2   |  | Sectorial       | (30,10)  | 60        |
| S2   |  | Sectorial       | (70,50)  | 20        |

Figure 79 illustrates the result of the first grid scan when  $k=0.2$ . Even no corners are covered in this scenario, the  $k$ -coverage state of all grids is correctly identified. Note that due to the given  $k$  value of 0.2, the tail coverage of sensor S3 may be regarded as efficiently covered. As a result the grid at the end of S3's coverage is judged as not  $k$ -covered. The result of fifth grid scan is shown in Figure 80. The  $k$ -covered rate of this scenario is 7.0%~3.07% with an estimation error of 3.03%.

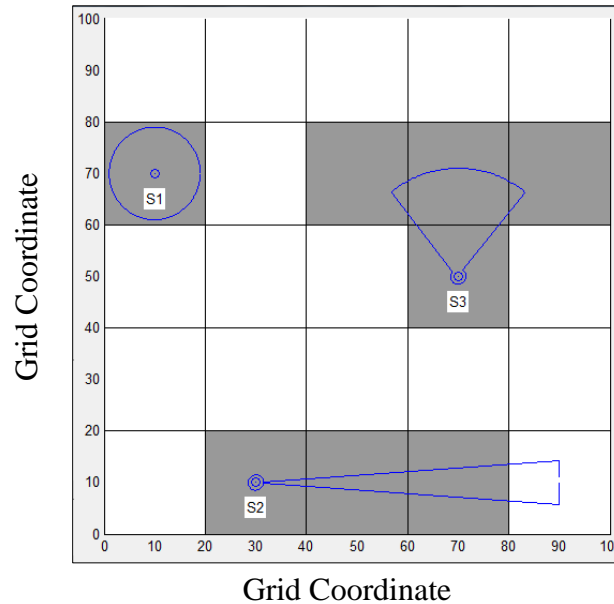


Figure 79. The first grid scan result shows that the  $k$ -covered state of all grids is correctly identified.

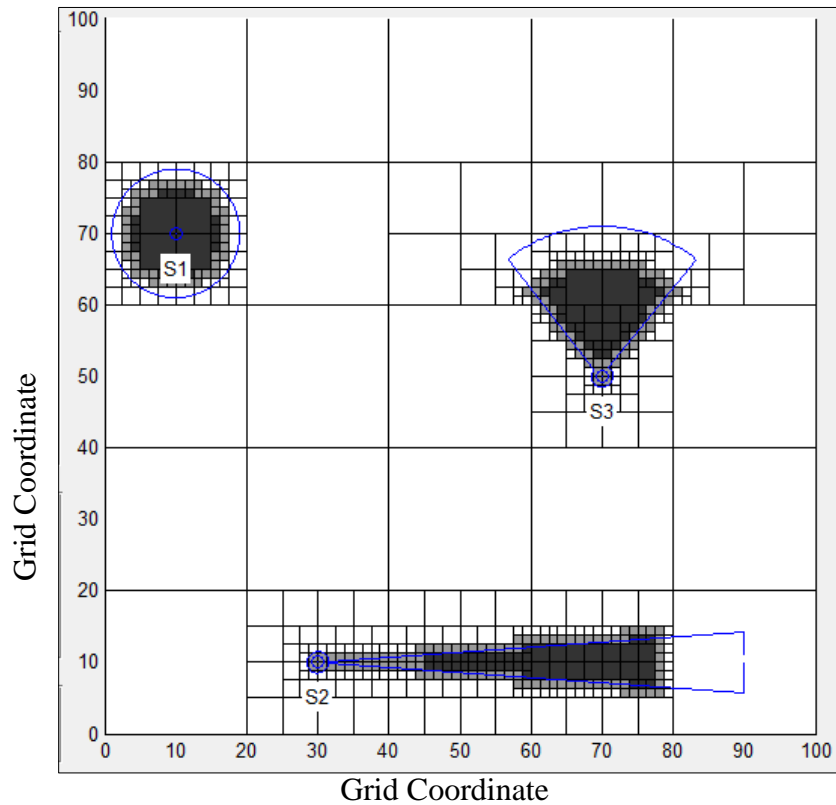


Figure 80. The fifth grid scan result shows the  $k$ -covered rate of this scenario is 7.0%~3.07% with an estimation error in 3.03%.

## 2. Experiment 2

This experiment attempts to illustrate an example of mixing multiple types of sensor models. The four sensors used are detailed in Table 13. Sensors S1 and S2 are both binary detection models; while the first one is omnidirectional, the other one is sectorial. Sensors S3 and S4 are linear probability detection models and also have two coverage shapes.

Table 27. Four different types of sensors are used in experiment 2.

| Name | Detection Model   | Coverage Shape  | Location | Max Range | Memo               |
|------|---|-----------------|----------|-----------|--------------------|
| S1   | Binary  | Omnidirectional | (25,75)  | 25        |                    |
| S2   |   | Sectorial       | (40,40)  | 60        | Looks at northeast |
| S3   | Linear probability <ul style="list-style-type: none"> <li>• 100% in the sensor's location</li> <li>• 0% in the max range</li> </ul> | Omnidirectional | (25,25)  | 25        |                    |
| S4   |   | Sectorial       | (40,30)  | 60        | Looks at east      |

The result of the first grid scan calculation with  $k=0.2$  is shown in Figure 81. While few grids are recognized as  $k$ -fully-covered, most grids are identified as  $k$ -partially-covered. The estimation  $k$ -covered rate is 12%~100% with an 88% error. The result of the fifth improved grid scan algorithm is shown in Figure 82. Estimative  $k$ -covered rate is 65.61%~56.84% with an error 8.77%. Compared to the first estimation, this result discloses more detail for reference. The surveillance and reconnaissance state of the current sensor deployment is known. In addition, possible weakness is represented to help in future enforcement design.

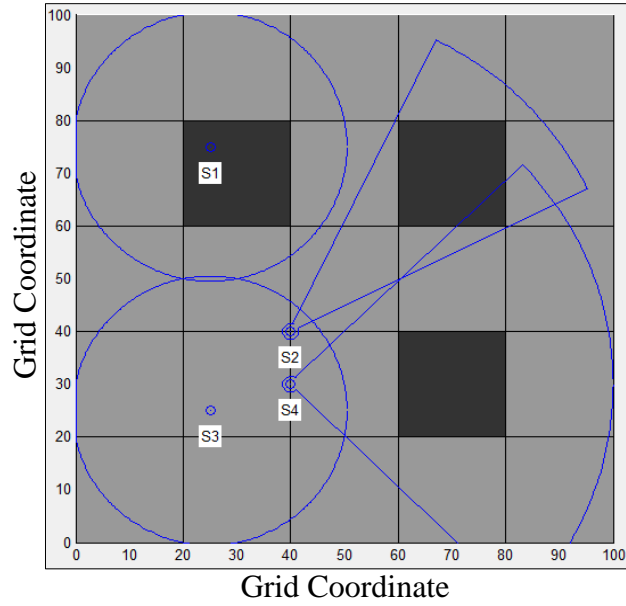


Figure 81. The first grid scan shows that mixing sensor types can be estimated correctly.

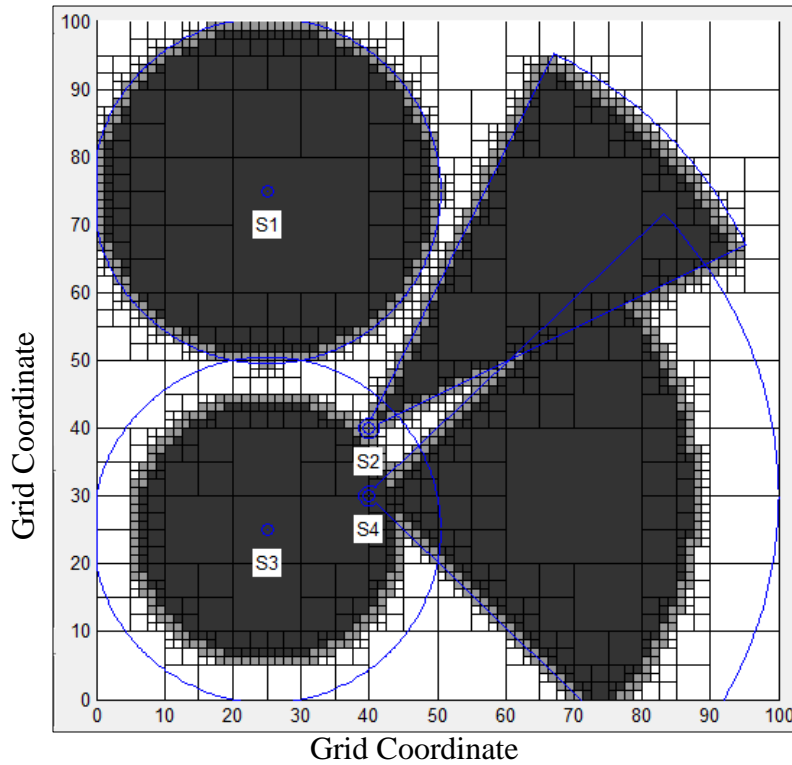


Figure 82. The fifth grid scan result shows the  $k$ -covered rate of this scenario is 65.61%~56.84% with an estimation error in 8.77%.

## **D. SUMMARY**

In this chapter, the concept of the probabilistic-sectorial sensor model is introduced. Possible challenges of the current grid scan algorithm are also discussed. The idea of monitor points is then proposed to regularize the criteria for identification of grid  $k$ -covered state. An improved calculation algorithm is also proposed to accommodate the probabilistic-sectorial sensor model. Experiments show that the proposed algorithm can deal with probabilistic-sectorial sensor model and generate meaningful estimation results when better simulation fidelity is needed.



THIS PAGE INTENTIONALLY LEFT BLANK

## VII. CONCLUSIONS AND RECOMMENDATIONS

This chapter presents the conclusions, limitations, and future works for the three subjects in this research: an improved DES framework (AEMF-DES), a simulation architecture of C4ISR in MDW (MDSIM), and an improved  $k$ -coverage rate problem algorithm.

### A. IMPROVED DES FRAMEWORK: AEMF-DES

#### 1. Conclusions

The challenges of current DES in C4ISR simulation include limited flexibilities in related scenarios and custom entities. These limitations may cause increased development cost and prohibit evolving system design. The reason for these limitations is related to the need for human resources to perform analysis within a scenario-oriented framework. AEMF-DES is therefore proposed in this research to improve these limitations.

AEMF-DES contains several techniques to reduce or eliminate the need for human intervention in DES. These techniques include: (1) fundamental theme agents (FTAs), which is used to embed the primary principles of the target theme, (2) part-entity-scenario structure (PES), which can describe the user-defined scenario along with custom entities, and (3) a DES engine that can interpret PES data structure and generate events automatically by referring to FTA which describe the nature of events relevant to the topical theme.

By using FTA, PES and a specialized DES engine, AEMF-DES not only inherits the advantages of relative faster simulation speed and accurate time estimation from typical DES framework, but also has more flexibility in entity relation setting and extensible part classes. These properties can improve the traditional DES limitations in those simulation programs that are conducted by AEMF-DES.

#### 2. Limitations

AEMF-DES is a generic-purpose DES framework. Although a demonstration simulation program MDSIM has been developed in this research, more simulation

programs in different theme topics are needed to verify its adaptability in various domains. Furthermore, high-fidelity simulation might need the time-step approach to provide calculation values when dealing with complex differential equations. This need leads to the future work of a hybrid architecture including DES and the time-step approach at the same time.

### 3. Recommendations for Future Work

#### a. *Advanced Analysis Tools*

AEMF-DES has been proved to eliminate some limitations related to the current DES framework. Going forward, it is recommended to extend this framework with more advanced analysis tools. For example, add a Monte Carlo procedure that can process the multiple-round simulation results or a recursive procedure that can automatically optimize scenario.

#### b. *Loosely Coupled with other Time Advance Mechanisms (TAMs)*

Although CDMES is a pure DES-style simulation framework, it can be extended to be loosely coupled with other time advance mechanism. Figure 83 shows an extension example that is dedicated to a human observer. The model state describes the battle status. While a DES mechanism handles important events just like what AEMF-DES does, a time-step mechanism is used to only update entity locations by their velocity in each time step. The model state is offered to the observer at the speed that can be adjust to be equivalent, faster, or lower than the real time.

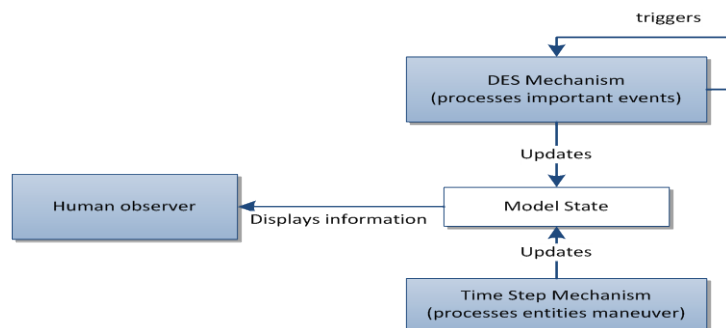


Figure 83. AEMF-DES can be extended to include real-time system and time-step approach in combination with DES by loosely coupling TAMs

## **B. A SIMULATION ARCHITECTURE OF C4ISR IN MISSILE DEFENSE WARFARE: MDSIM**

### **1. Conclusions**

To verify the feasibility of the simulation architecture and explore its potential, in addition to the proposed AEMF-DES framework a simulation program, MDSIM, was also built. This simulation program architecture covers the C4ISR process in MDW including: (1) the detection of sensors, (2) communication among units, (3) the intelligence process and command of headquarters, (4) an attack from weapon systems, and (5) the hostile jamming of sensor and communication systems.

During this dissertation, the concept of C4ISR processes in MDW is discussed and followed by the introduction to each FTA related to this theme. Then the design interfaces of PES and simulation results are also shown for user reference. Experiments show that, based on the AEMF-DES framework, released simulation programs allows end users determine scenario related to a given theme and can customize entity specification which are not possible in those simulation program that conducted by typical DES framework.

### **2. Limitations**

- Demonstrative part classes used to represent components in battlefield are modeled based on general understanding without formal attributes analysis.
- MDSIM has not been testified by a formal user test.
- Although an idea about hybrid architecture is proposed, no implementation is made to show the possibility to contain DES and the time-step approach at the same time.

### **3. Recommendations for Future Work**

#### *a. More Extended Part Classes*

In this research, a few simple models such as a pulse radar system, regular wireless communication equipment, and anti-aircraft artillery have been created and adopted in MDSIM. More extended part classes such as low-probability interception (LPI) radar or more advanced EW parts are recommended for inclusion.

***b. Connection to Virtual Environment***

While this simulation architecture focuses on the fundamental concept of C4ISR processes in MDW, it is possible to extend this architecture to other virtual environment programs to offer more intuitive analysis results for end users.

***c. Connection to Live Simulation***

Not only providing great extensibility, the loosely coupled property of MDSIM also enables connection to live simulators due to the fact that the detailed algorithm of each part is defined within concrete part classes. Parameters of human-involved processes can be determined by human input. For example, when a headquarters part attempt to make a decision, a human tester can be involved to provide following information:

- Instruction for each anti-air entity, such as attacking a new target, changing to another target, stop, etc.
- The time required for each process to make a decision.

The headquarter part can then use this information for scheduling events such as when the headquarters its decision making and generates a command message.

***d. Definition of System Requirement, Refactoring, and Performing Verification & Validation***

Although MDSIM is in purpose of demonstrating the feasibility of AEMF-DES, it also illustrates a basic prototype of a C4ISR simulation system. Interesting future work is possible implementing an applicable simulation program by following steps:

- Define the topical simulation system requirements
- Revise the conceptual model described in this dissertation.
- Refactor the current program according to the new conceptual model.
- Perform verification by comparing whether the program fits the conceptual model, such as event sequences, example numeric results, etc.
- Perform validation by testing whether the program can satisfy the system requirements. User testing may be needed.

This implementation supports the exploration of potential implements to AEMF-DES in realistic simulation scenarios. Additional implement to AEMF-DES is also desired.

## C. AN IMPROVED $k$ -COVERAGE RATE ALGORITHM

### 1. Conclusions

The current  $k$ -coverage rate problem was considered in this research, pointed out that the latest calculation approach is appropriate in simple sensor models but insufficient in probabilistic-sectorial sensor models. Such models are used to describe practical situations more precisely and provide more detailed estimation of sensor deployment strategy. This work proposes an improved calculation algorithm which is capable of accommodating probabilistic and sectorial sensor models.

A simple experimental program has been built to test and verify the improved algorithm. Results of the research experiments showed that by applying the algorithm the  $k$ -covered rate of a given region with probabilistic and sectorial sensor models can be calculated accurately.

### 2. Recommendations for Future Work

#### a. *Embed into Broader Simulation Software*

While this work has proposed an improved  $k$ -coverage rate calculation algorithm, it has not been embedded into any large simulation program to connect to real sensor deployment design for meaningful analysis reports. It is recommended to combine this algorithm into simulation software such as MDSIM, SIMKIT, or MATLAB Simulink. When simulation has been executed, the  $k$ -coverage rate can offer a metric related to how the sensors are deployed.

THIS PAGE INTENTIONALLY LEFT BLANK

## **APPENDIX A: USER MANUAL OF MDSIM**

### **A. INTRODUCTION**

#### **1. Purpose of This Document**

This document is dedicated to introduce MDSIM, a simulation program related to the C4ISR processes in Missile Defense Warfare. Readers should understand how to manipulate MDSIM after reading this document.

#### **2. Content Organization**

The concept of MDSIM is introduced in next section, followed by a tutorial covering fundamental C4ISR processes in Missile Defense Warfare.

### **B. WHAT MDSIM IS**

#### **1. Background**

MDSIM is developed by You-Quan Chen to support his Ph.D. dissertation in the Naval Postgraduate School in 2013. While his research is pursuing an improved Discrete-Event-Simulation (DES) framework, MDSIM is designed to prove the feasibility of a new proposed framework. Note that MDSIM only provides basic analysis functions due to time limitations and security considerations.

#### **2. Installation and Execution**

- Microsoft .NET Framework 4 Client Profile is needed before execution.
- No installation is required; just click MDSIM.exe to run it. (Not the MDSIM.exe.config file)

#### **3. Operation Process**

The operation process of MDSIM is illustrated in Figure 84. After a scenario is designed, a simulation is launched. If the result can satisfy your need, the process is ended. Otherwise, you can redesign another scenario for more simulation. Note that, unlike the traditional DES framework, as long as new scenarios are related to the Missile



Defense Warfare theme defined in MDSIM, you do not need to analyze/create additional events or build new programs.

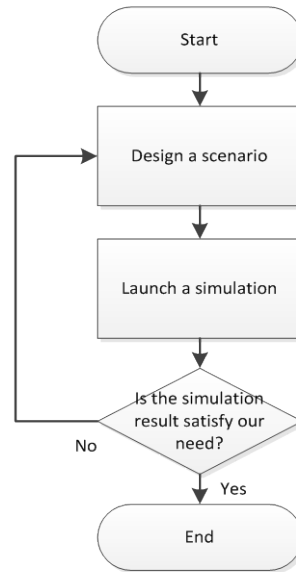


Figure 84. Operation Process Flow Chart of MDSIM

## C. TUTORIAL: FROM DETECTION TO FIRE

A defensive deployment to intercept an invasive missile is testified in this simulation. Defender in this scenario contains a city as a target, a radar station, a headquarters, and anti-air artillery. Attacker has a missile to attack the city, a jammer to window the missile invasion.

It is been expected that the radar station can sense the incoming missile and send message to the headquarters. After the headquarters confirms the threat and gives command to the ant-air artillery, the ant-air artillery will attempt to take the missile down. However the radar detection and communication might be blocked by the jammer.

### 1. Review Loaded Part Templates

#### a. Part Template Tab

After launch the program, switch to the Part Tab (shown in Figure 85). Part listed in the left treeview come from the part files loaded and is the part templates

used in creating an entity. Those files are in dll format and can be offered by MDSIM developer or created by end users (C# program skill is needed). Click any one of them to see the detail information in the right hand side.

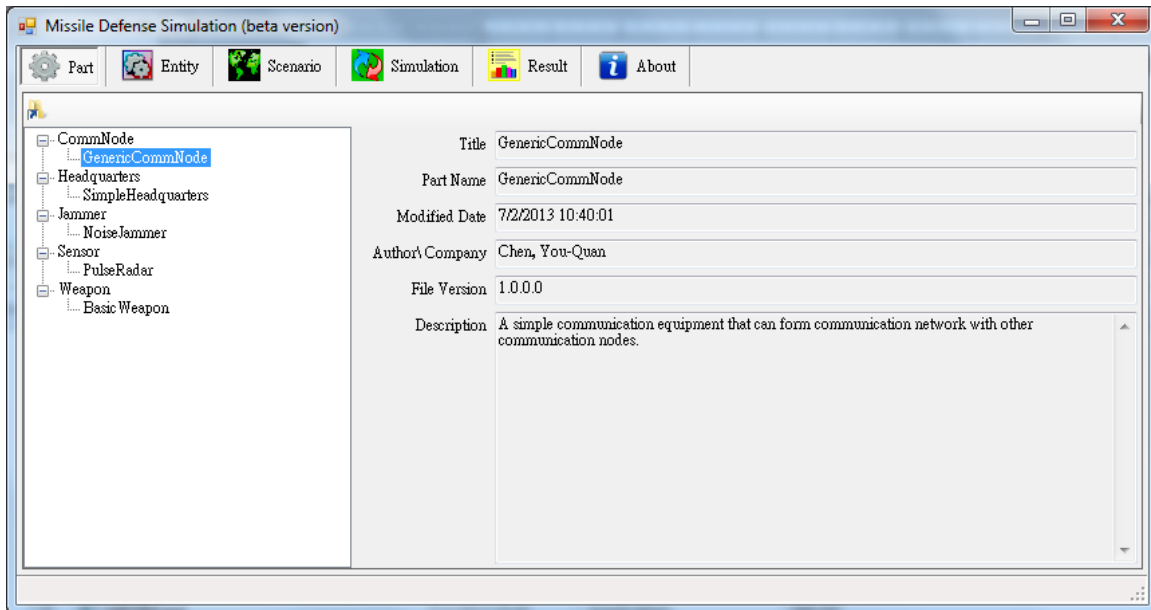


Figure 85. In Part tab of MDSIM, part dll files loaded are listed.

## 2. Build Custom Entity Templates

### a. Entity Template Tab

Before design a scenario, entity templates must be defined first. Switch to Entity Part (See Figure 86). The toolbar below the top tabs consists of several to let users add, edit, and delete custom entities. The last button opens the file folder contains those entity files.

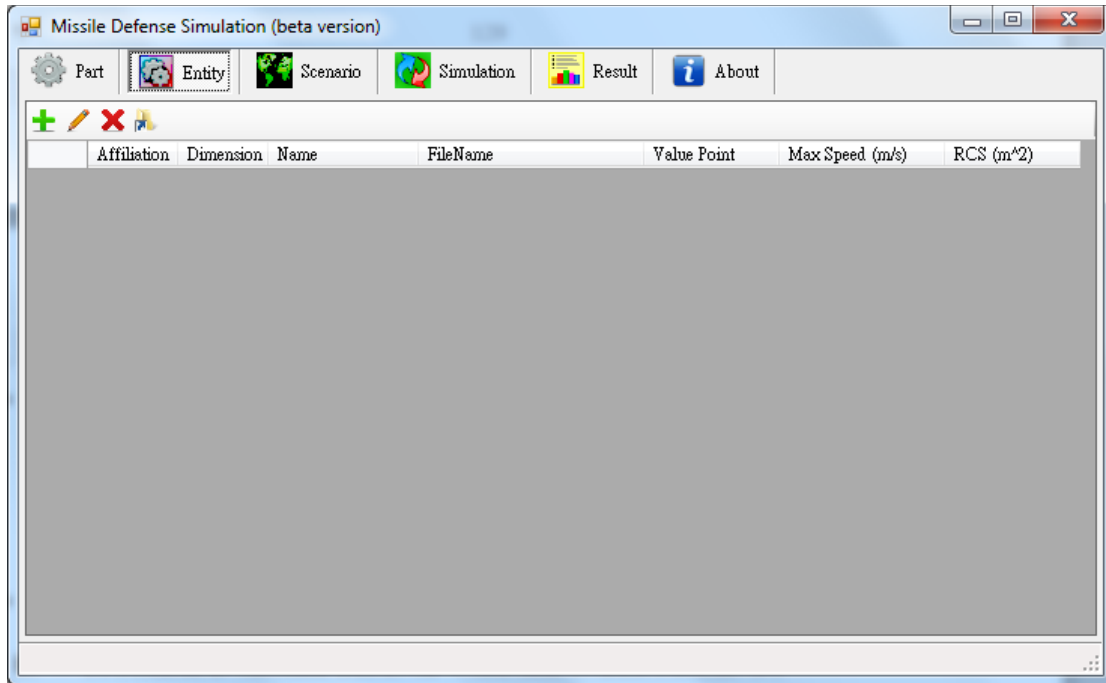


Figure 86. Entity tab of MDSIM

***b. The Entity Edit Form***

Click the add button (the green cross) to create a city entity template. A small dialog will then show up to select the affiliation of this entity (see Figure 87). Click Defender and continue.

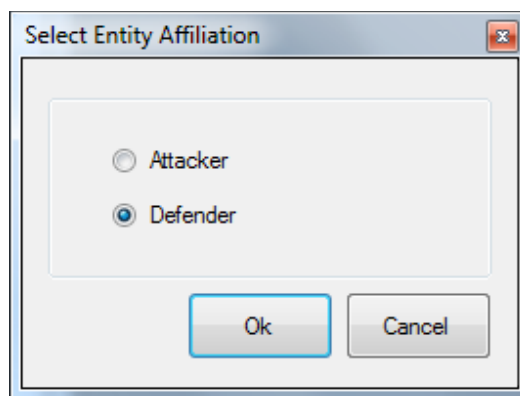
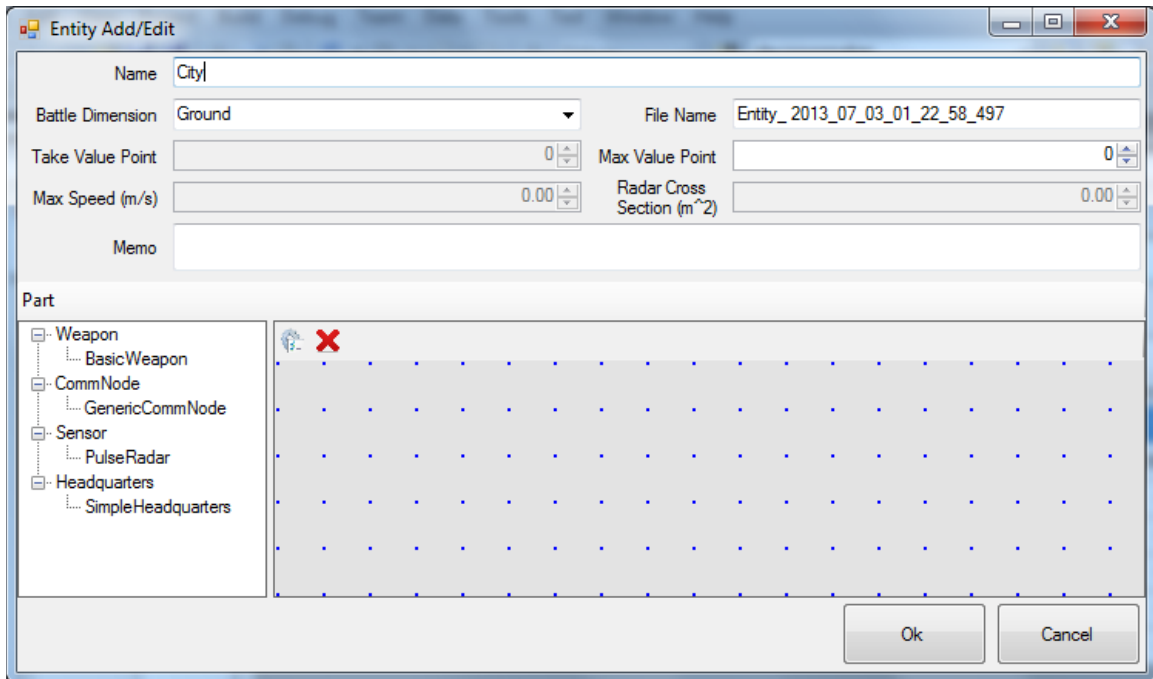


Figure 87. A dialog to select entity affiliation

Figure 88 shows the Entity Add/Edit form. Attributes are listed in top. Bottom area is for part design and the available part templates in the left-bottom are listed

depending on the affiliation. This is introduced later. So far, just change the name to City and click ok.

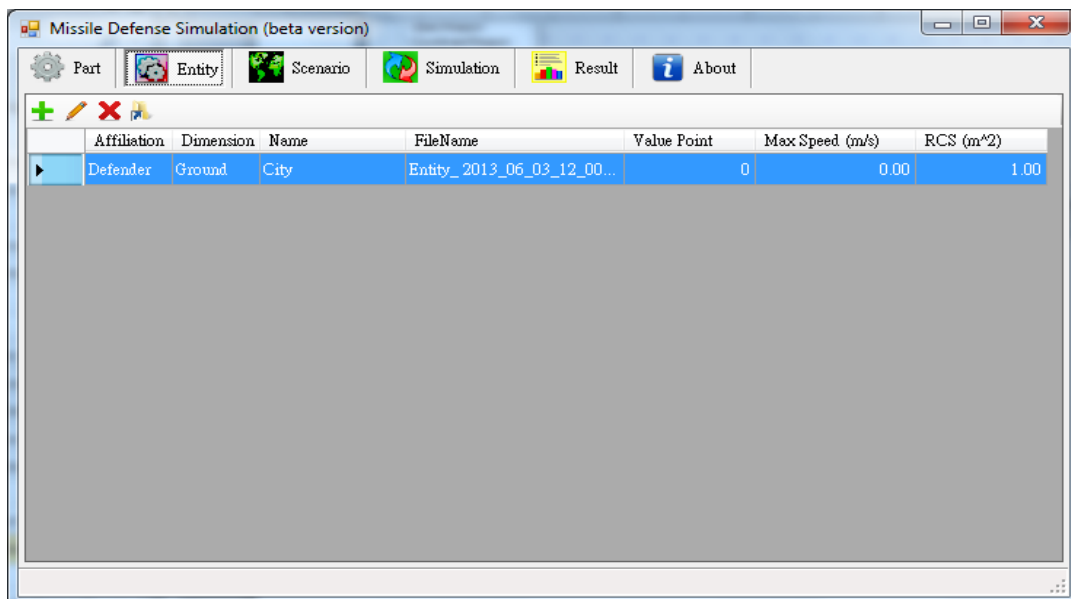


The dialog box 'Entity Add/Edit' contains the following fields and controls:

- Name: City
- Battle Dimension: Ground
- File Name: Entity\_2013\_07\_03\_01\_22\_58\_497
- Take Value Point: 0
- Max Value Point: 0
- Max Speed (m/s): 0.00
- Radar Cross Section (m<sup>2</sup>): 0.00
- Memo: (empty text area)
- Part list (left):
  - Weapon
    - BasicWeapon
  - CommNode
    - GenericCommNode
  - Sensor
    - PulseRadar
  - Headquarters
    - SimpleHeadquarters
- Part grid (right): A grid with a red 'X' icon in the top-left corner.
- Buttons: Ok, Cancel

Figure 88. Entity Add/Edit Form.

Now you may see a new entity has been add into entity table (Figure 89).



The main window 'Missile Defense Simulation (beta version)' shows the 'Entity' tab selected. The entity table contains the following data:

|   | Affiliation | Dimension | Name | FileName                   | Value Point | Max Speed (m/s) | RCS (m <sup>2</sup> ) |
|---|-------------|-----------|------|----------------------------|-------------|-----------------|-----------------------|
| ▶ | Defender    | Ground    | City | Entity_2013_06_03_12_00... | 0           | 0.00            | 1.00                  |

Figure 89. A new entity named “City” is added into entity table.

Click the Add button again. This time we choose attacker affiliation and name this entity “Missile”. Change the battle dimension to “Air” and speed to “300” (see Figure 90). Note that if an attacker entity is set to other than air dimension, a radar may not be able to detect. Click Ok to close this form.

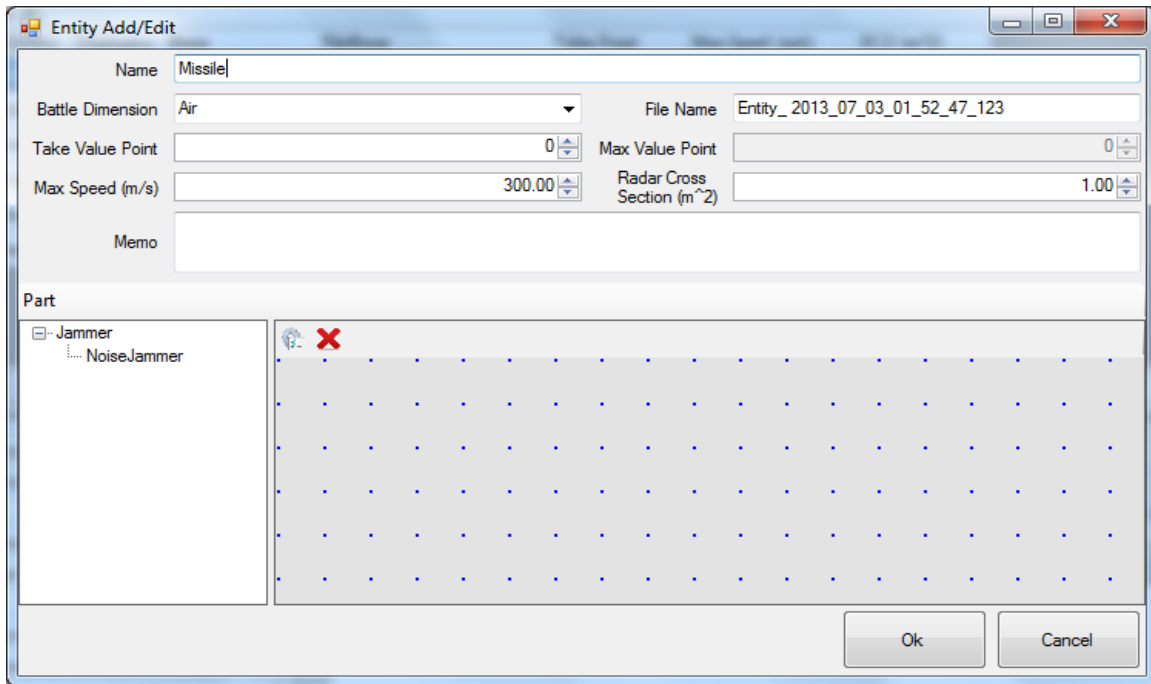


Figure 90. Add Missile Entity

**c. Part Assembling and Custom Properties**

Add another new defender entity and name it “Radar Station.” In the left-bottom treeview, click the “Pulse Radar” part in the treeview in the left bottom side and drag it into the panel on the right hand side. This action adds a “Pulse Radar part into this entity” Similarly, add a “GenericCommNode” from the treeview. These two parts here represent the radar component and communication equipment required in a radar station. Click the gray triangle of the “PulseRadar” and drag to the hollow triangle of the “GenericCommNode”. Therefore, all detection reports generated from the radar part can be delivered to the communication part which then can communicate with other communication parts in other entities. Your form should look like Figure 91.

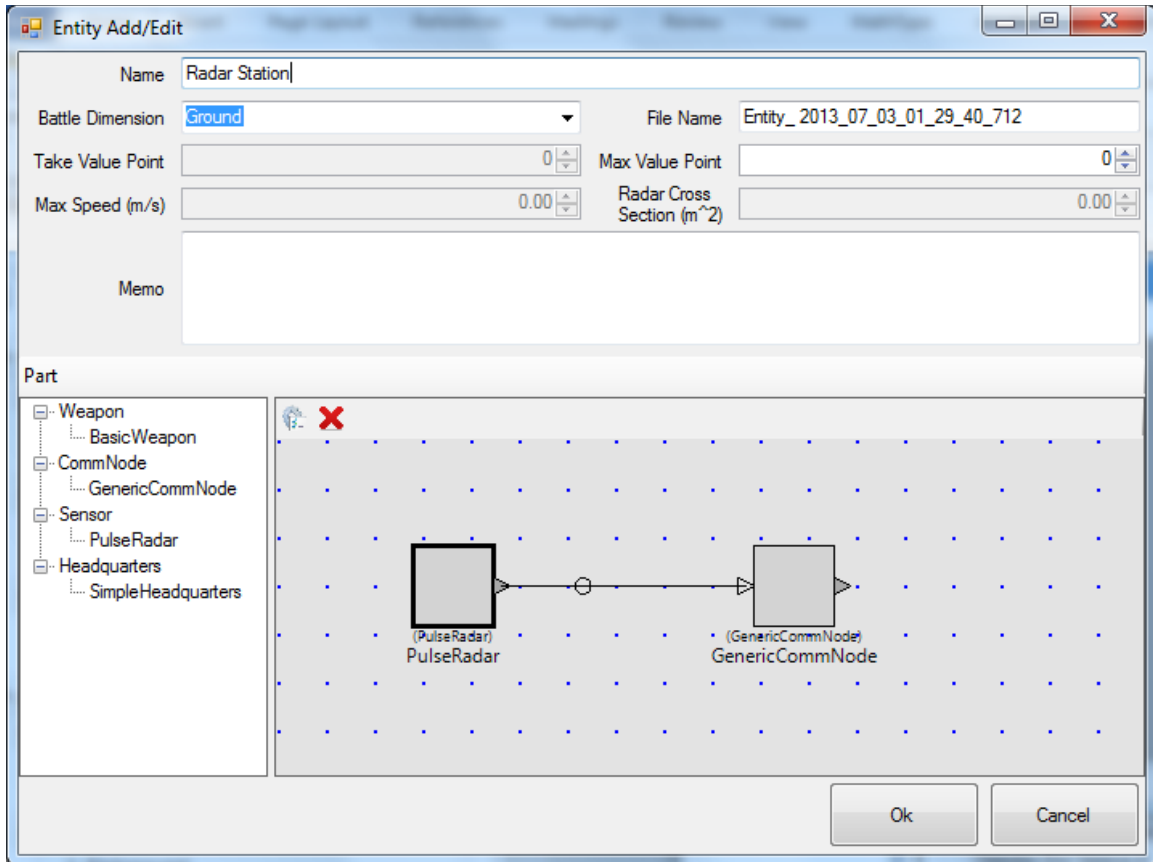


Figure 91. Add two parts with connection between them.

Detail settings of parts can be changed. Select the “PulseRadar” and click the customized properties button above the panel (next to the redcross). Another dialog shows for editing the customized properties. Change the coverage radius to 12000.

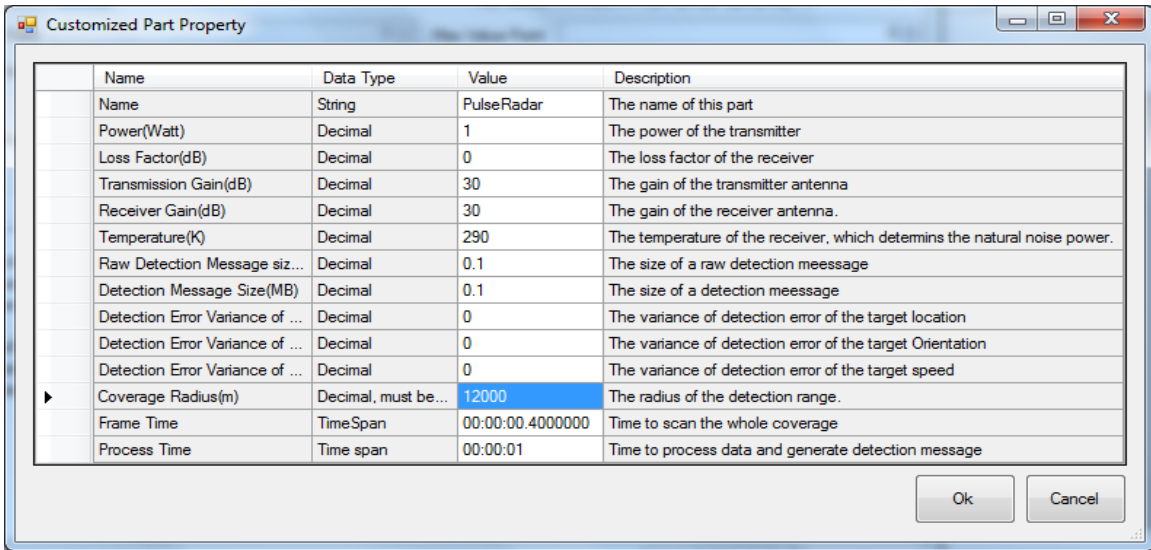


Figure 92. Customized Properties Edit Form for a part

Add two more defender entities, a HQ and an Anti-Air Artillery, one attacker entity, Jammer (shown in Figure 93 ~ Figure 95).

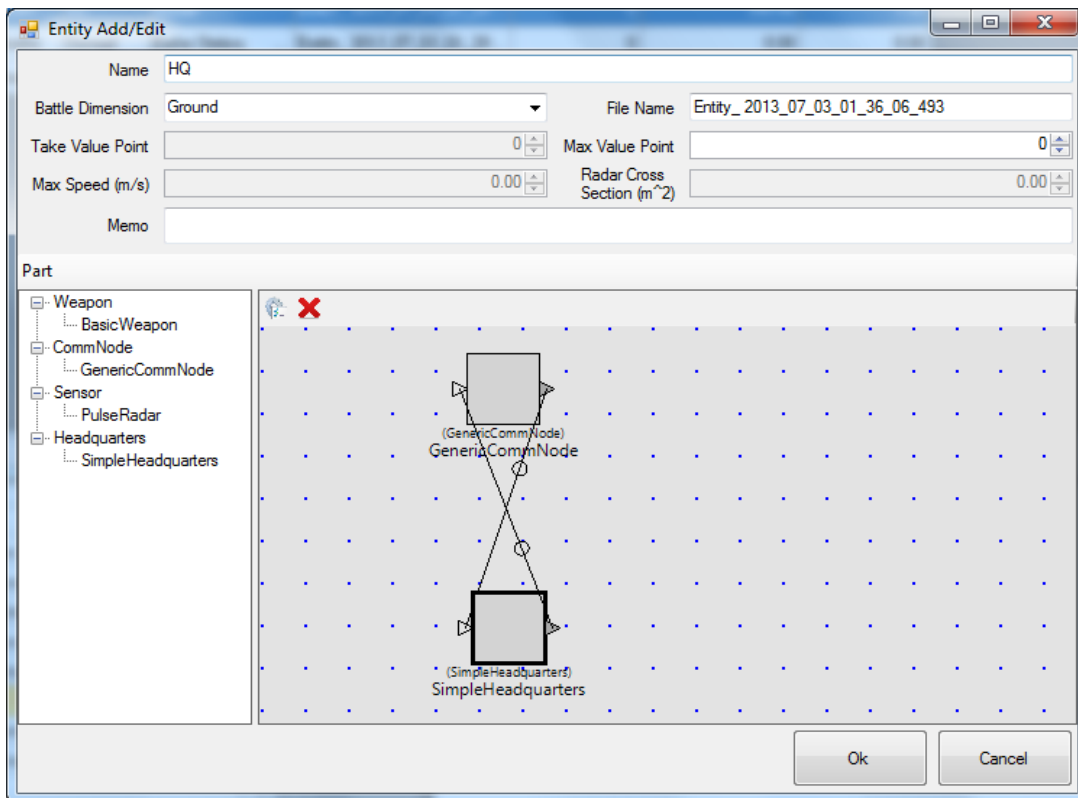


Figure 93. Add a HQ entity template

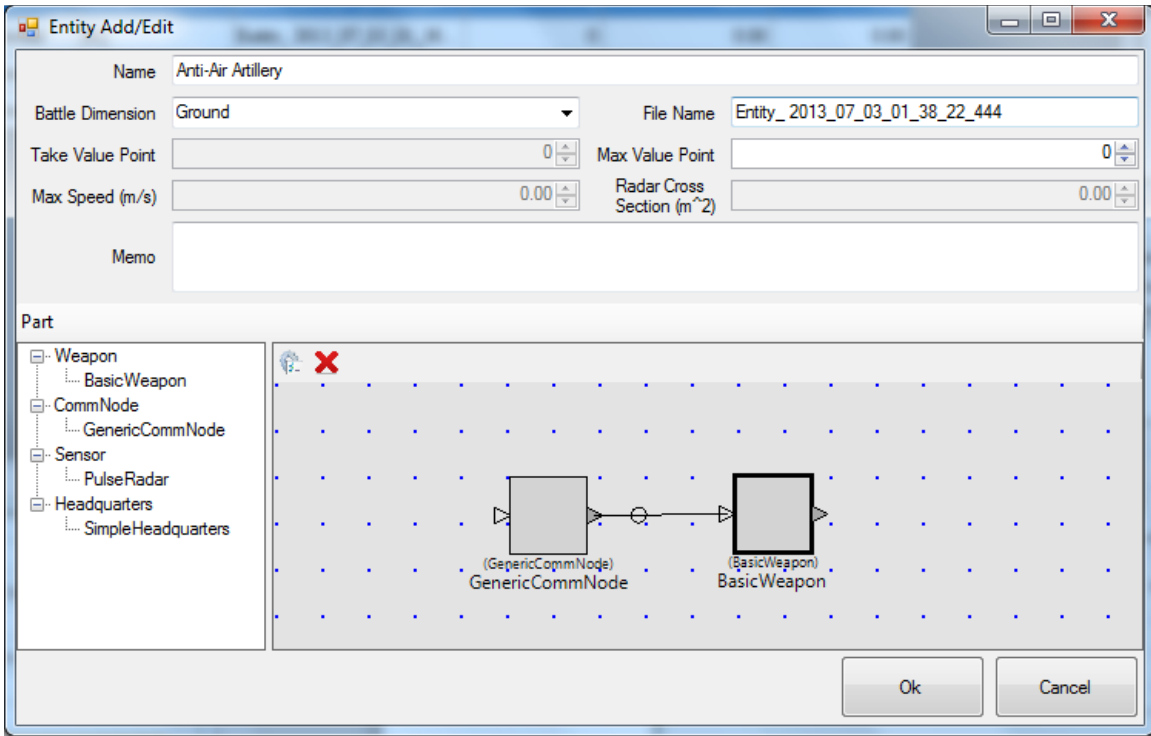


Figure 94. Add an Anti-Air Artillery entity template

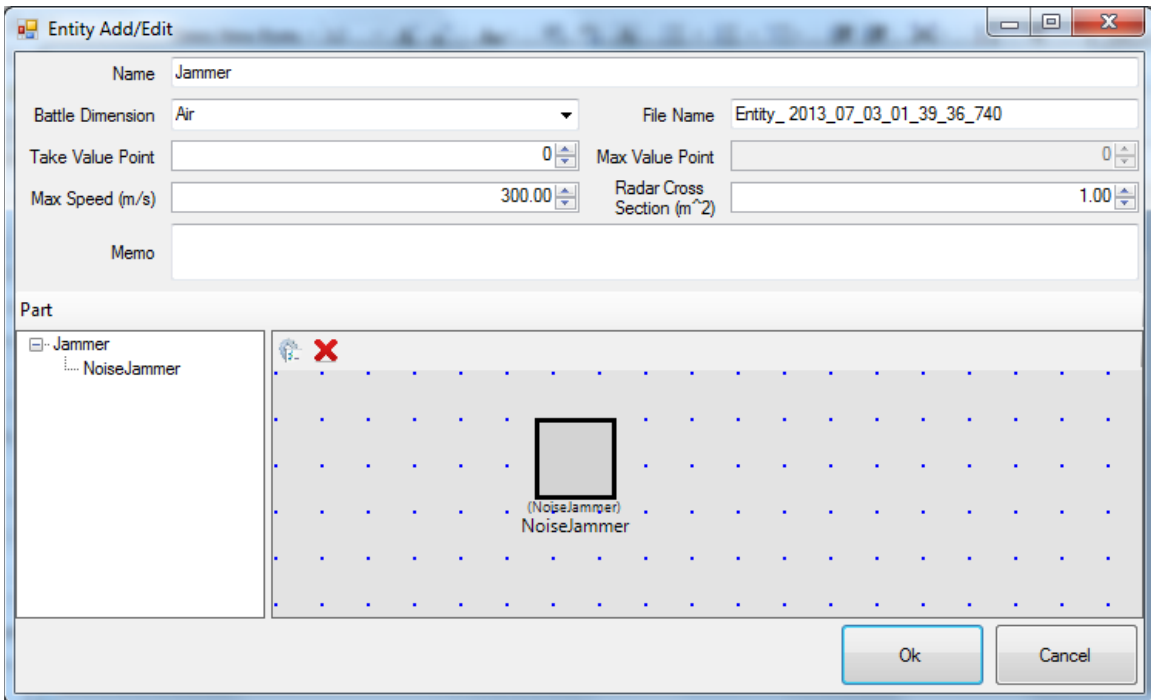


Figure 95. Add a Jammer entity template



Your entity table should look like Figure 96. Entity templates listed in this table is used in following scenario design.

| Affiliation | Dimension | Name               | FileName                   | Value Point | Max Speed (m/s) | RCS (m <sup>2</sup> ) |
|-------------|-----------|--------------------|----------------------------|-------------|-----------------|-----------------------|
| Attacker    | Air       | Jammer             | Entity_2013_07_03_01_39... | 0           | 300.00          | 1.00                  |
| Attacker    | Air       | Missile            | Entity_2013_07_03_01_29... | 0           | 300.00          | 1.00                  |
| Defender    | Ground    | City               | Entity_2013_07_03_01_29... | 0           | 0.00            | 0.00                  |
| Defender    | Ground    | Anti-Air Artillery | Entity_2013_07_03_01_38... | 0           | 0.00            | 0.00                  |
| Defender    | Ground    | Radar Station ...  | Entity_2013_07_03_01_29... | 0           | 0.00            | 0.00                  |
| Defender    | Ground    | HQ                 | Entity_2013_07_03_01_36... | 0           | 0.00            | 0.00                  |

Figure 96. Entity Table with Several Entities

### 3. Design a Scenario

#### a. Scenario Design Tab

Switch to the Scenario Tab (Figure 97). Here is the place you can create and modify scenarios. Right now no scenario is loaded so most of the controls are disable.

Property Entity Maneuver Electromagnetic Setting 0.010

Name

Author

Size (m)

Background

We are  Atta  Def

Description

Figure 97. Scenario Tab of MDSIM

**b. Create Map**

Click the Add bottom in toolbar to create a new scenario. Attributes are listed in left hand side. Change them as you wish. An example is shown in Figure 98.

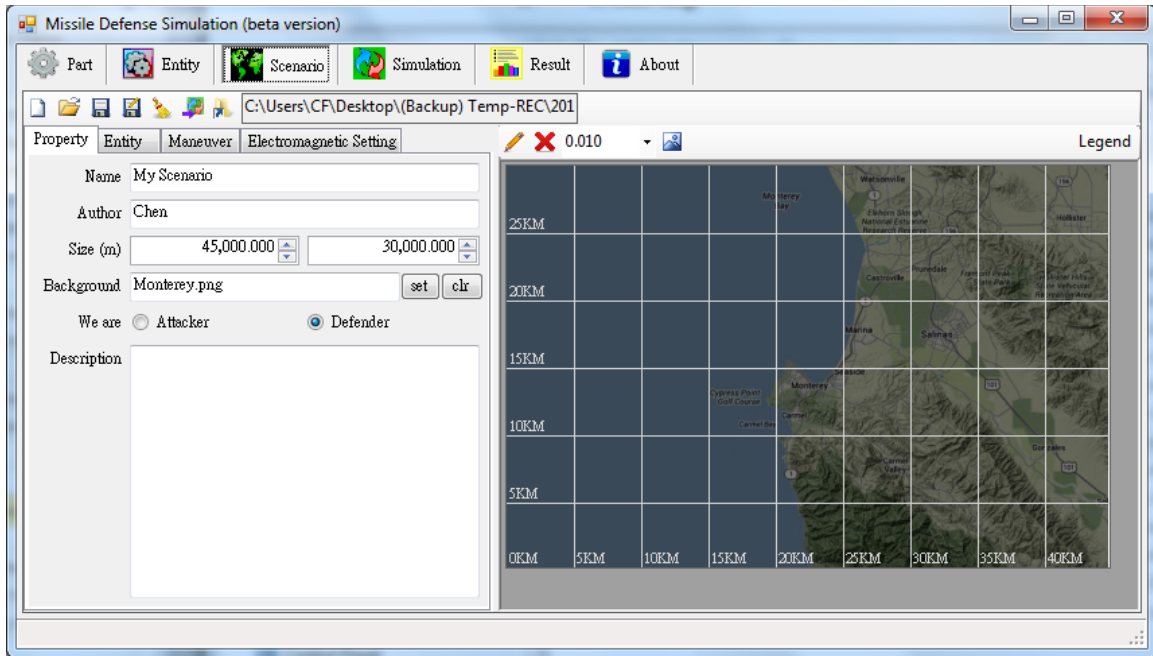


Figure 98. An Example of Scenario Setting

**c. Add Entities**

Switch to the Entity Sub-tab within this Scenario Tab. Drag a city, a missile, and a jammer entity to the map (Figure 99). Note that the legend can be shown by click the legend button on the top-right location

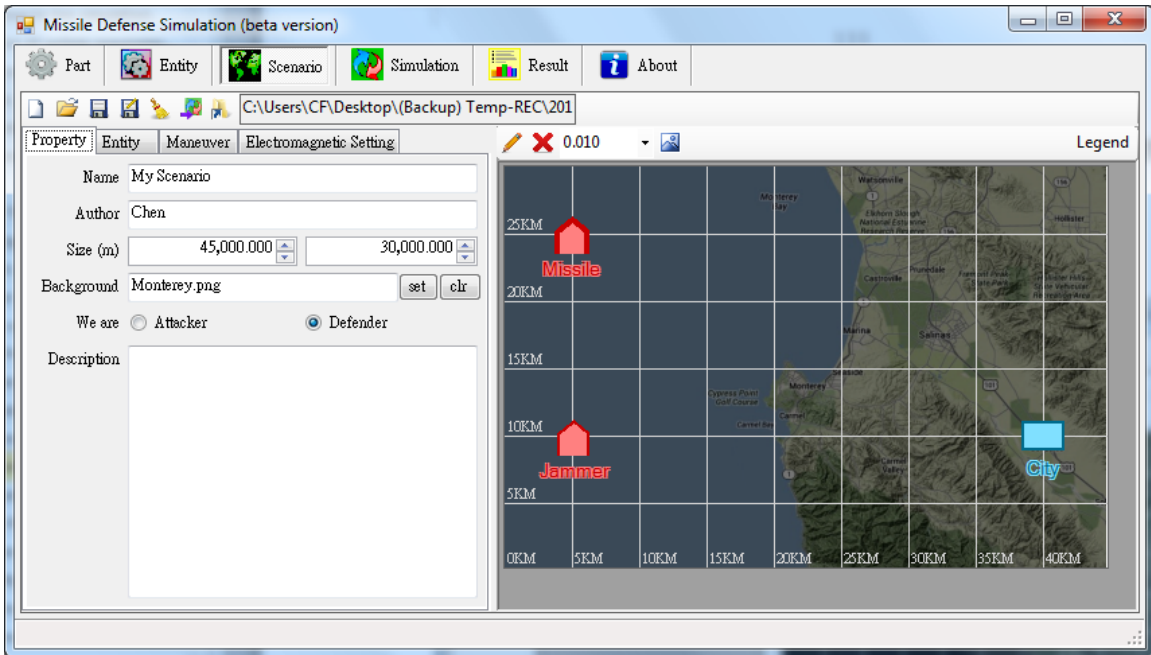


Figure 99. A city, a missile, and a jammer have been dragged into map.

***d. Set Location, Maneuver, and Target***

Switch to the Maneuver Sub-tab (Figure 100). Here you can set the show time, location, maneuver, and target of an entity. Right click the missile to select it and set the city as a target of the missile (Figure 101). Then click the add button of navigation point, set a navigation point (Figure 102). An example setting is shown in Figure 103.

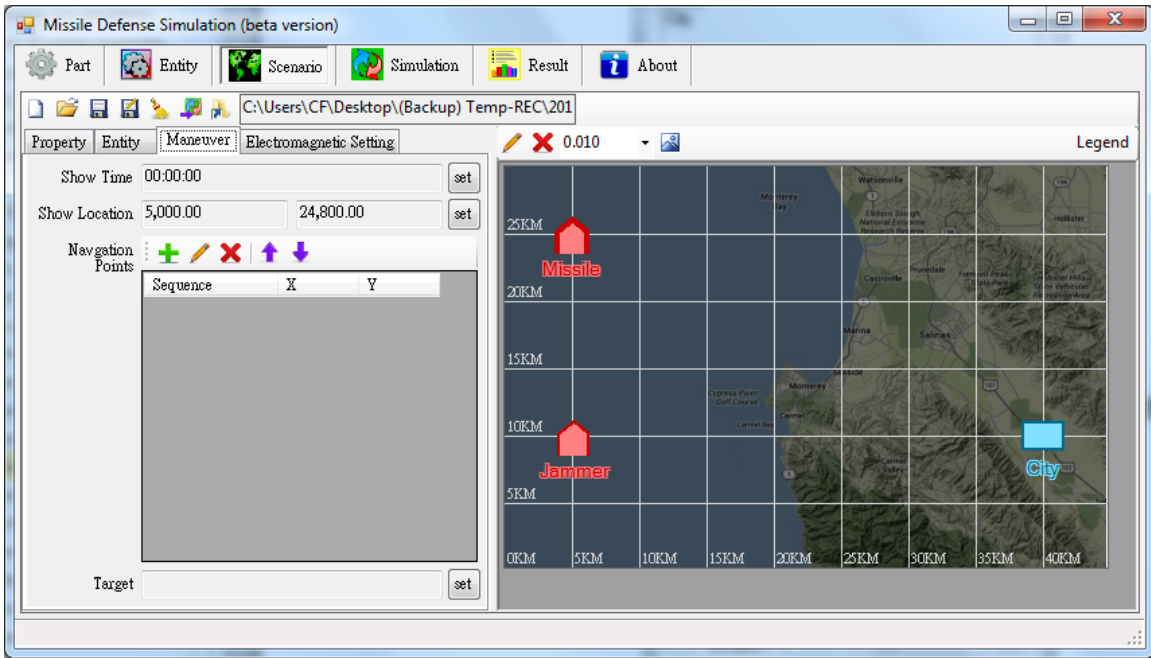


Figure 100. Maneuver Sub-tab

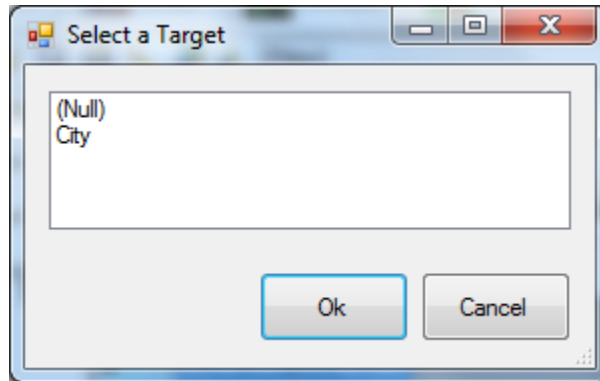


Figure 101. Dialog for Selecting a Target

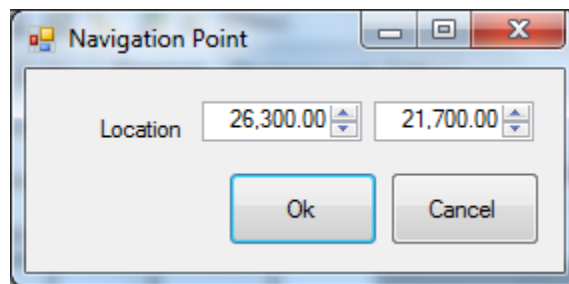


Figure 102. Dialog for Setting Navigation Point

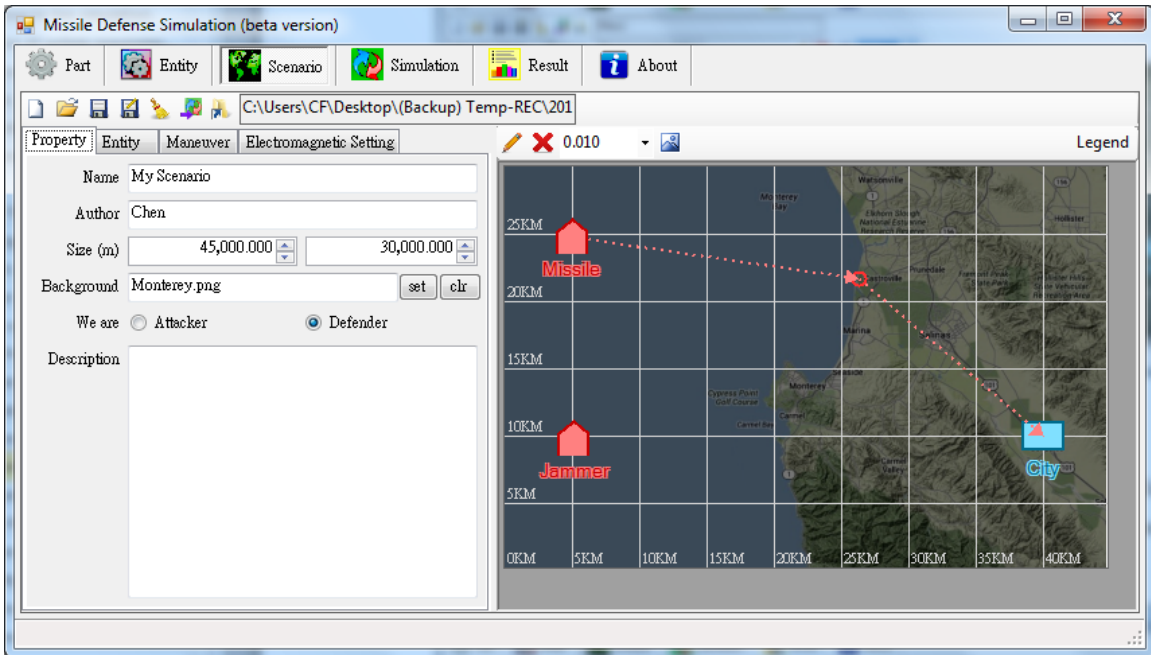


Figure 103. An Example of Attacker Setting

Switch back to Entity Sub-tab, drag a radar station, a HQ, and an anti-air artillery to the map. Make sure the coverage of radar and anti-air artillery cover the path of the missile. Figure 104 shows an example of defender deployment.

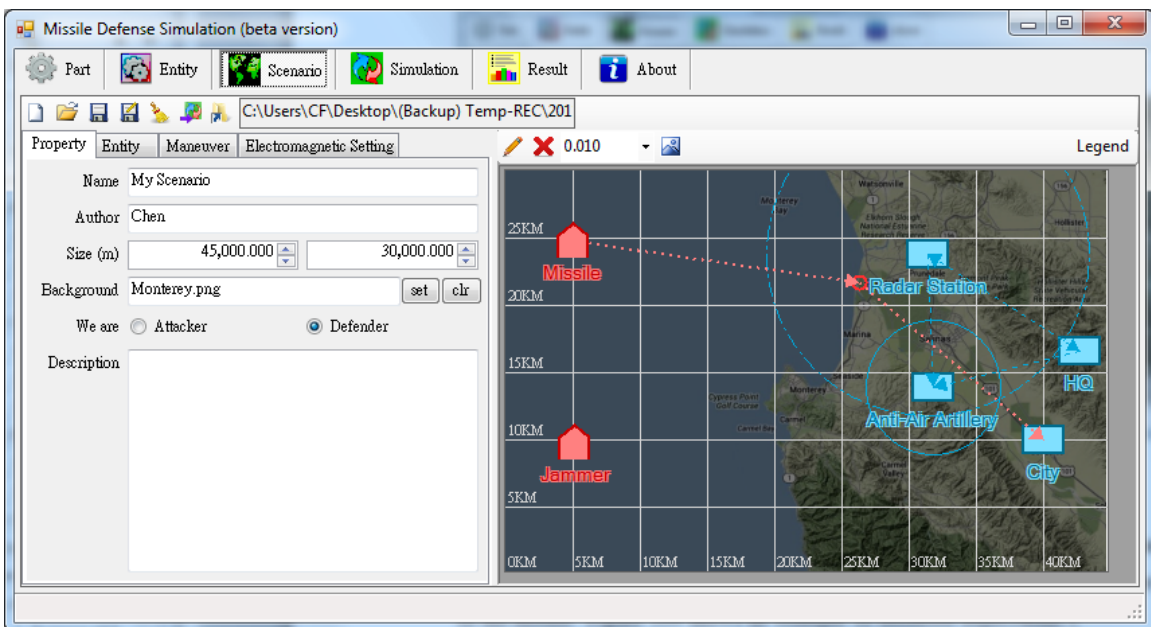


Figure 104. An example of Defender's Deployment

*e. Set Electromagnetic Spectrum*

Switch to Electromagnetic Setting Sub-tab in that three possible setting options are on the top: Sensor, Comm. Jammer.

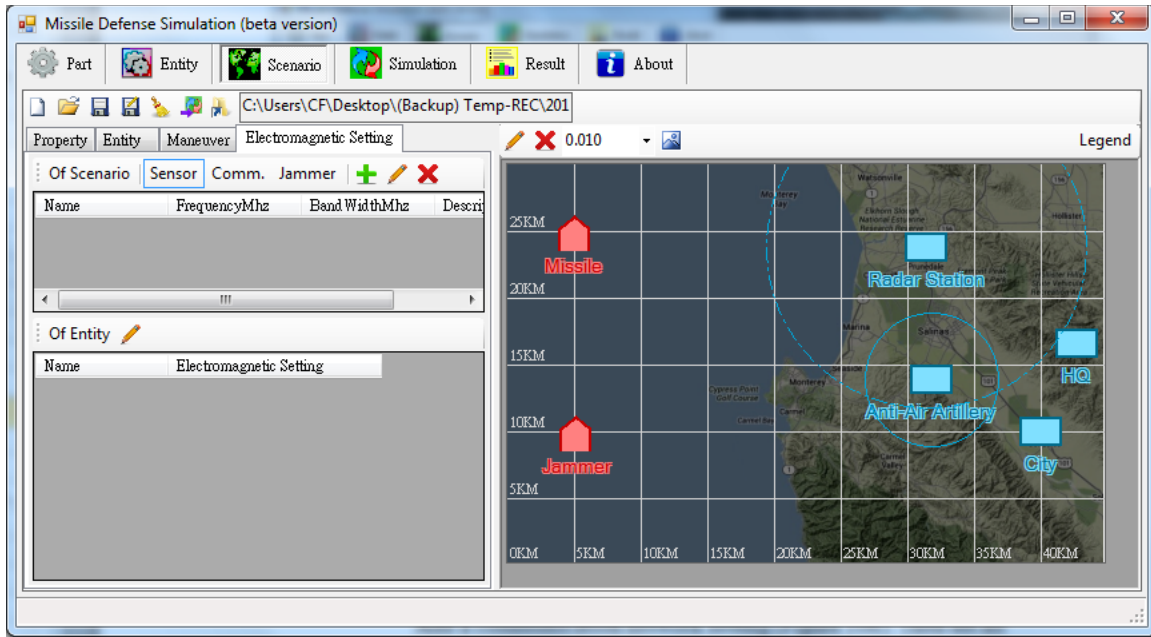


Figure 105. In the Electromagnetic Setting, three types of setting are available: Sensor, comm., and Jammer.

Add a communication network setting (Figure 106). Then set all communication equipment in the radar station, the HQ, the anti-air artillery use same communication network (Figure 107). Several dash lines with arrows should appear on map to indicate the communication network among defenders (Figure 108).

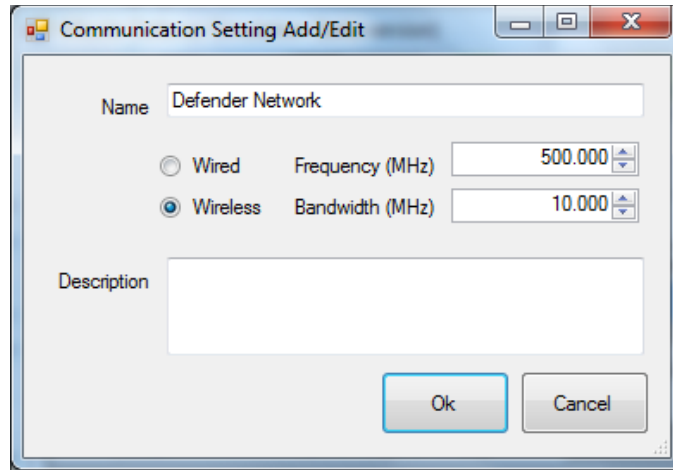


Figure 106. Dialog for setting Communication Setting

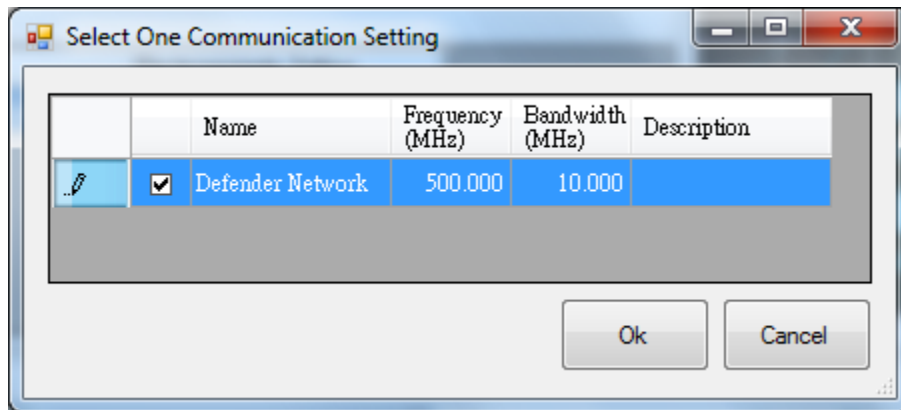


Figure 107. Dialog for Selecting Communication Network

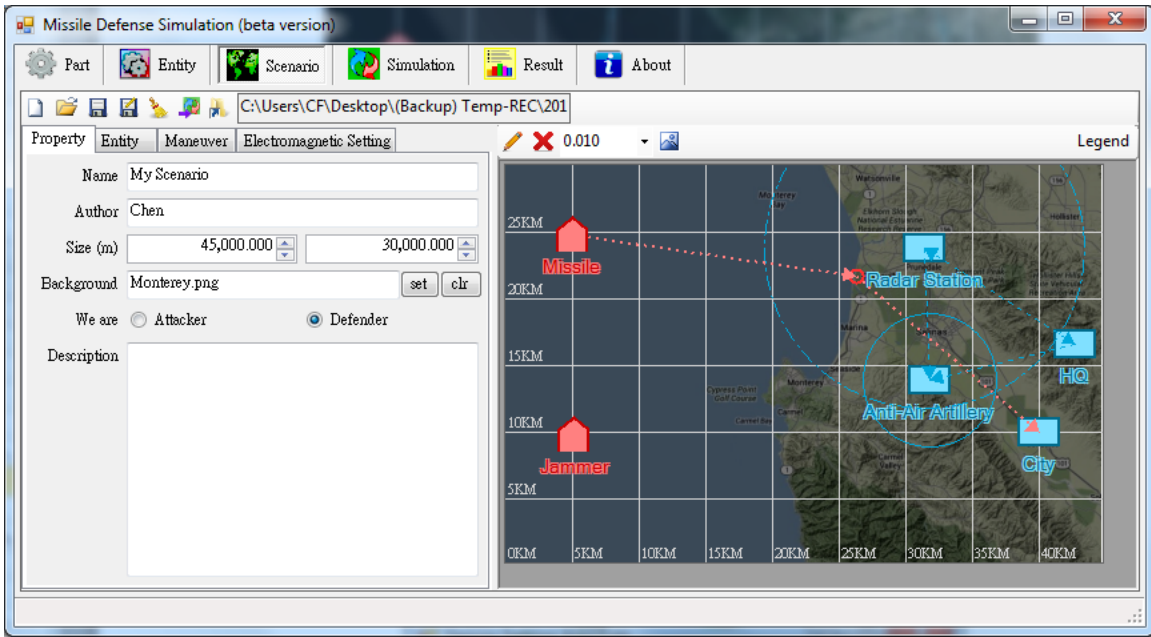


Figure 108. Several dash lines with arrows appear on map to indicate the communication network among defender entities.

Also set frequencies for the radar station and the jammer by using the default value (Figure 109).

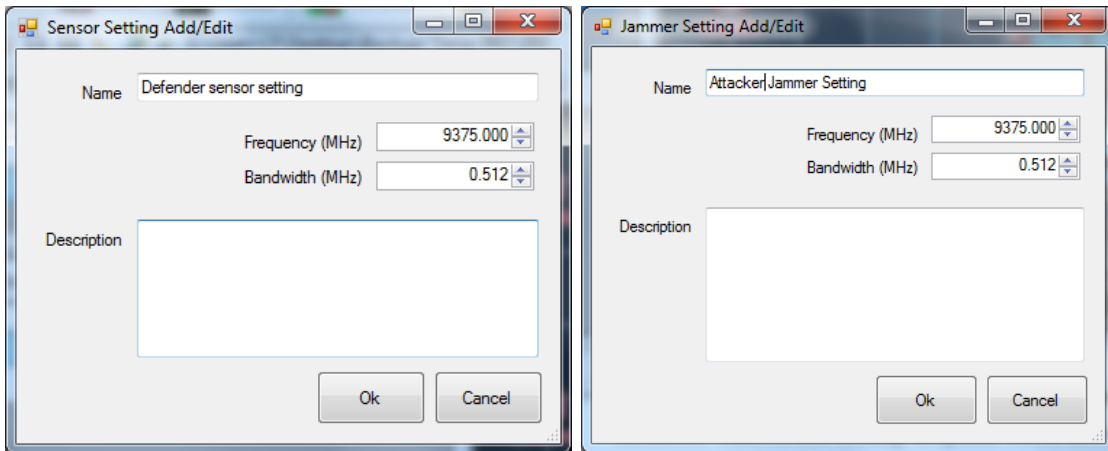


Figure 109. Sensor and jammer settings using default value.



*f. Adjust Entity Properties*

Although it is recommended to set part's properties in related entity template, it is possible to adjust anyone of them even after dragged into a scenario. For example, the anti-air artillery is not expected to send any information during the simulation. Select it and clicking the pencil button above the map to set transmission rate 0 (Figure 110).

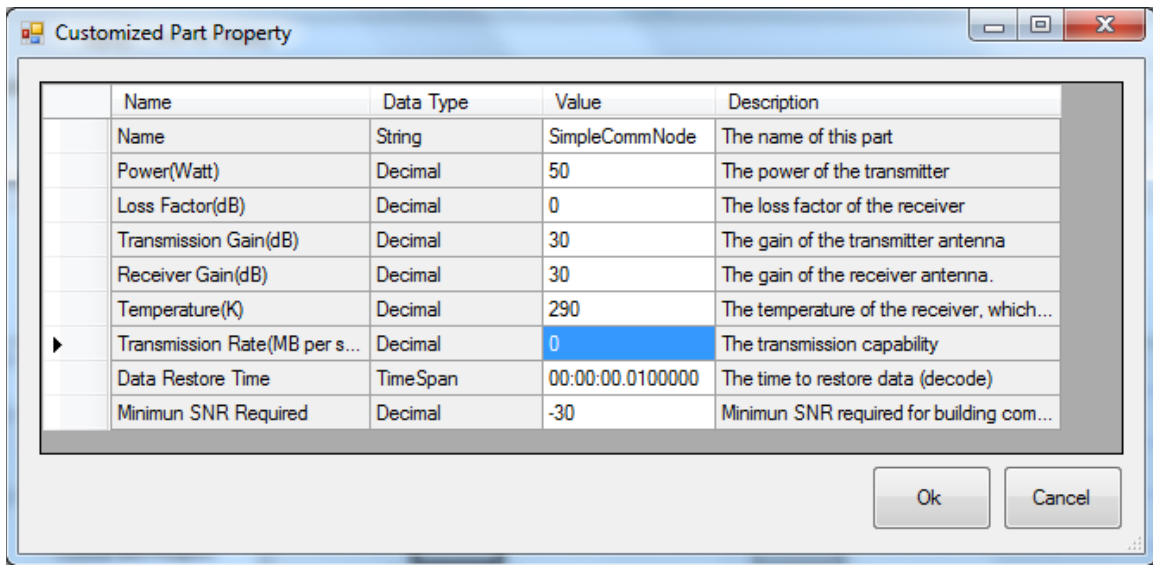


Figure 110. Adjust customized properties to fit the entity state.

*g. Save and Bring to Simulation Execution Tab*

The map now should show the attacker's strategy to the target, the deployment and the communication network of the defender. This scenario is ready to go. Click save button and click the sixth button in the main toolbar to bring this scenario to the Simulation Execution Tab.

**4. Run a Simulation**

*a. Simulation Execution Tab*

Switch to the Simulation tab (Figure 111). The left hand side shows information of a simulation while the scenario mode state is shown in right hand side.

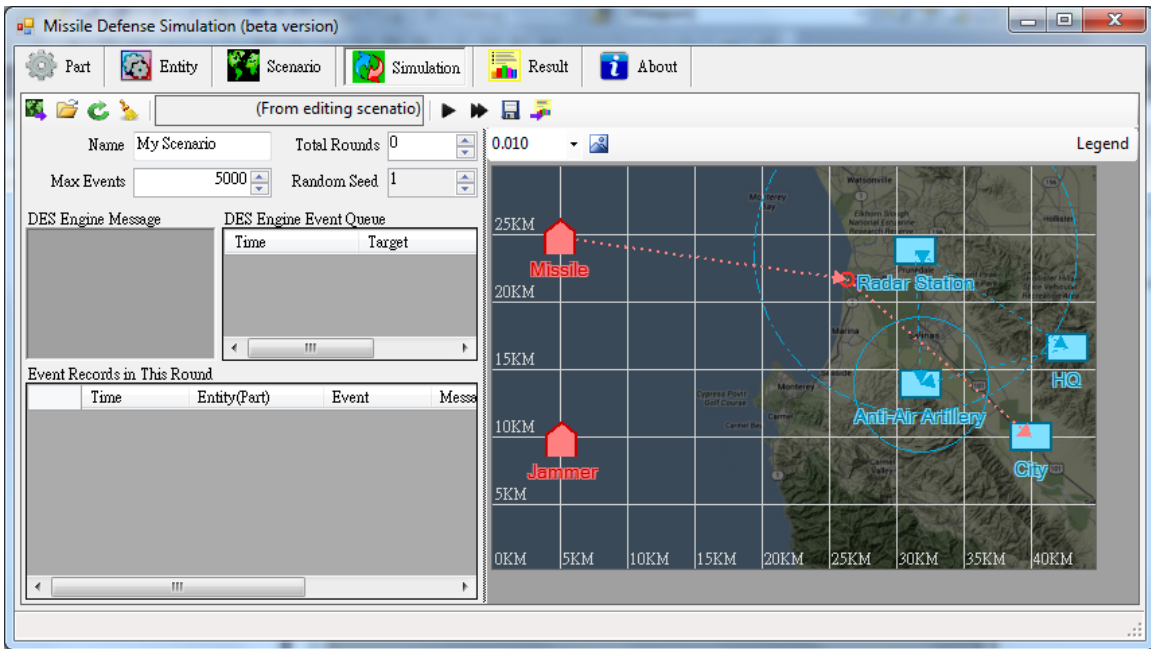


Figure 111. Simulation Tab of MDSIM

***b. Step-by-step or Run-all-at-once***

User may execute simulation in either “step by step” or “run all at once” modes. In step-by-step mode (Figure 112), DES engine messages, events queue, and event history are in the left hand side while the state is shown in the right hand side. It is convenient to observe how events are created and triggered in this mode. In the other hand run-all-at-once mode does not display message during the simulation but just generate the final result as fast as possible (Figure 113). Run the simulation in any model and click save button to save the result file.

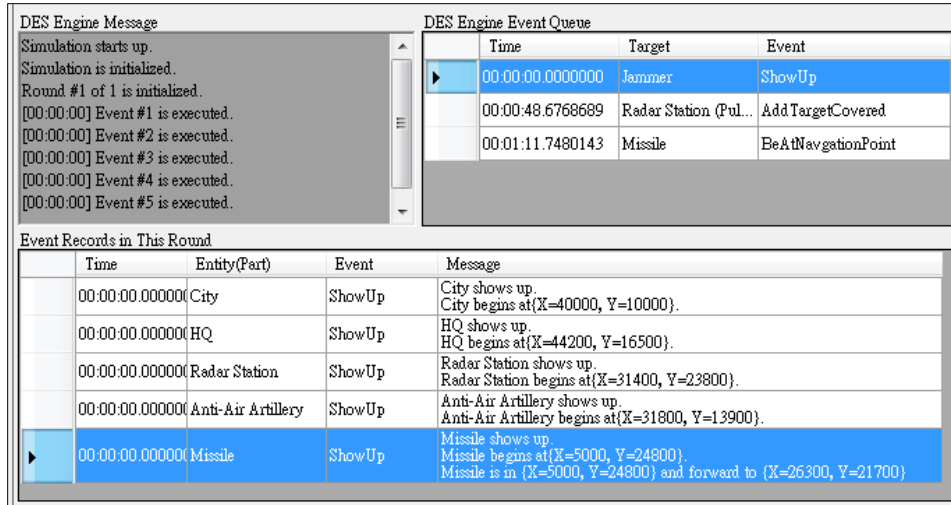


Figure 112. Messages Displayed in Step-by-Step Mode

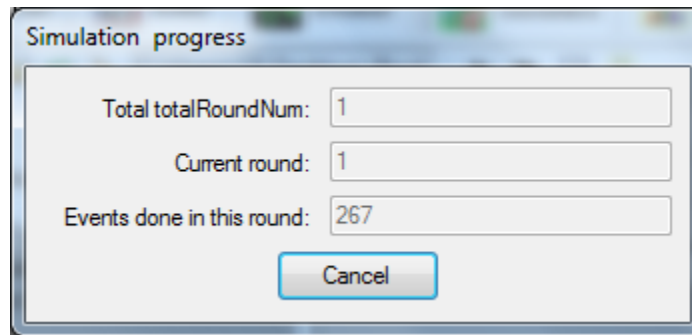


Figure 113. Dialog for Run-All-At-Once Mode

*c. Save and Bring to Result Viewer Tab*

Click save button in the main toolbar to save the result then click the latest button to bring this result to the Result Viewer Tab.

**5. Review the Result**

*a. Result Viewer Tab*

Switch to result side (Figure 114). The information of the scenario is shown in the left hand side, round and event lists locate in the middle, and event message and related state map re in the right hand. Note that the result differs from settings.

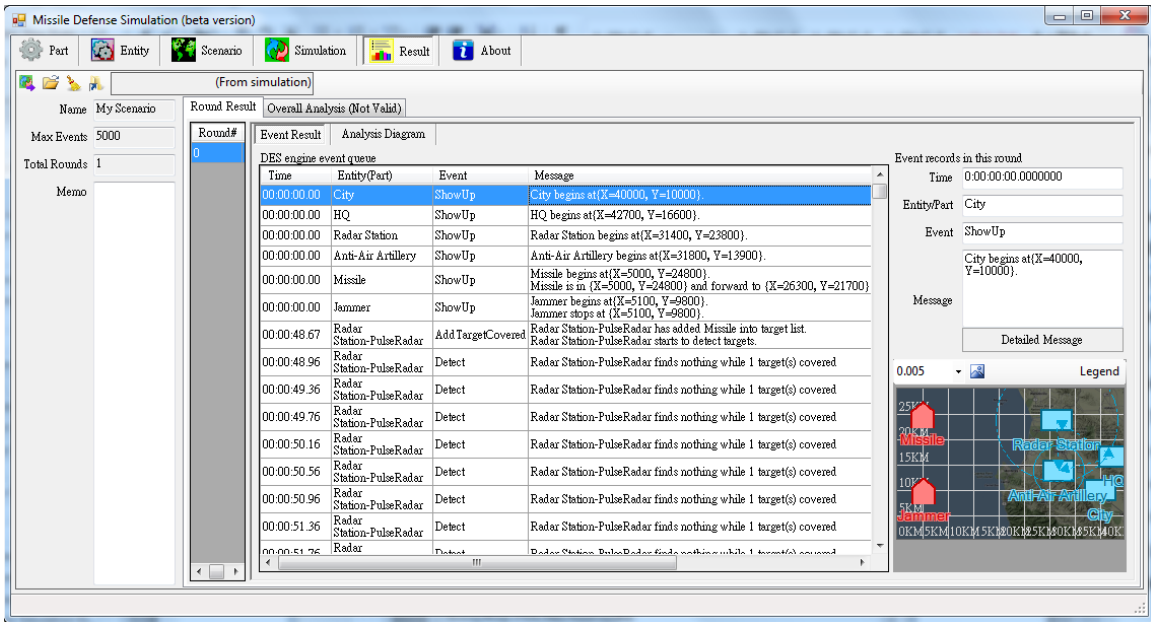


Figure 114. Result Tab of MDSIM

**b. Detail Event Message**

Some events may contain much information. Select a detection event and click the detail button below the message textbox to review the detail event messages (Figure 115).

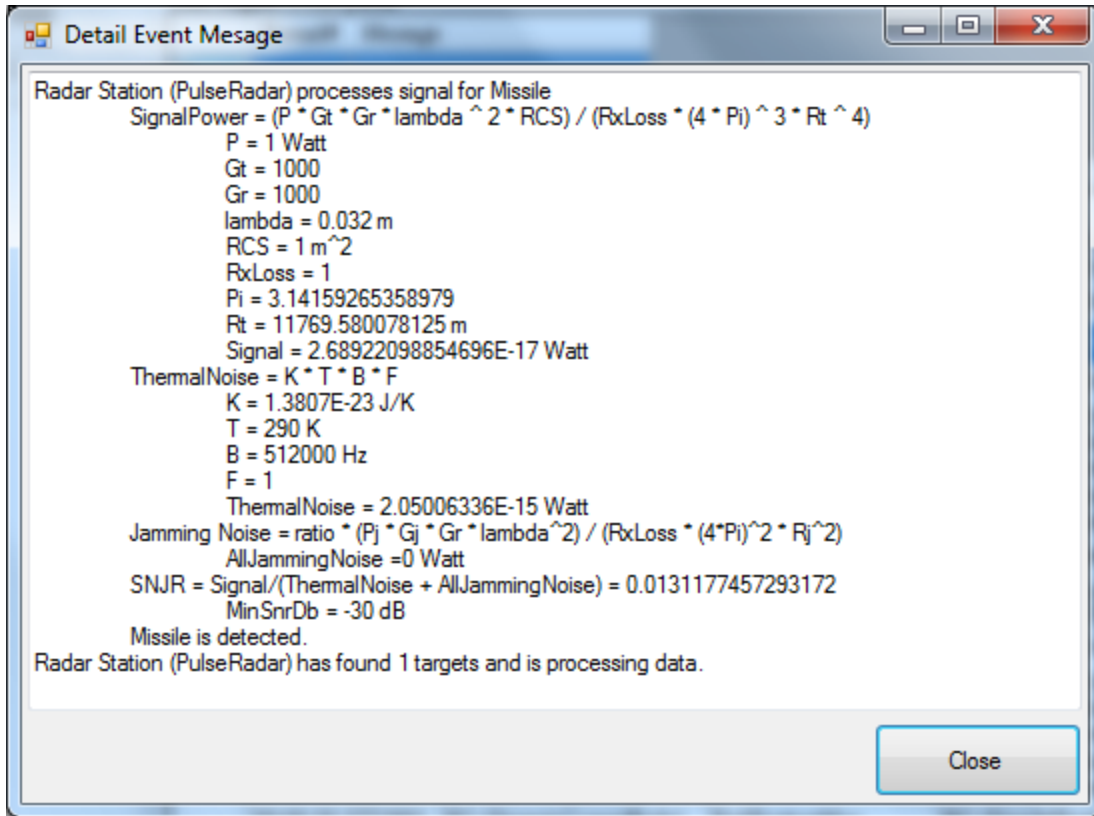


Figure 115. An Example of Event Detail Message

c. *Analysis Data Diagram*

Switch to the “Analysis Diagram” tab (shown in Figure 116). Data collected in the simulation are listed here. Select “Detection SNR (no jamming)”, “Detection SNR (with jamming)”, and “Detection SNR threshold” then click “Generate Diagram” button to show the diagram form (as Figure 117). Due to the extensibility of MDSIM, the types of data collected in simulations cannot be known during the development. As a result, this diagram form is made generic. End users have to adjust parameters to have a desired appearance.

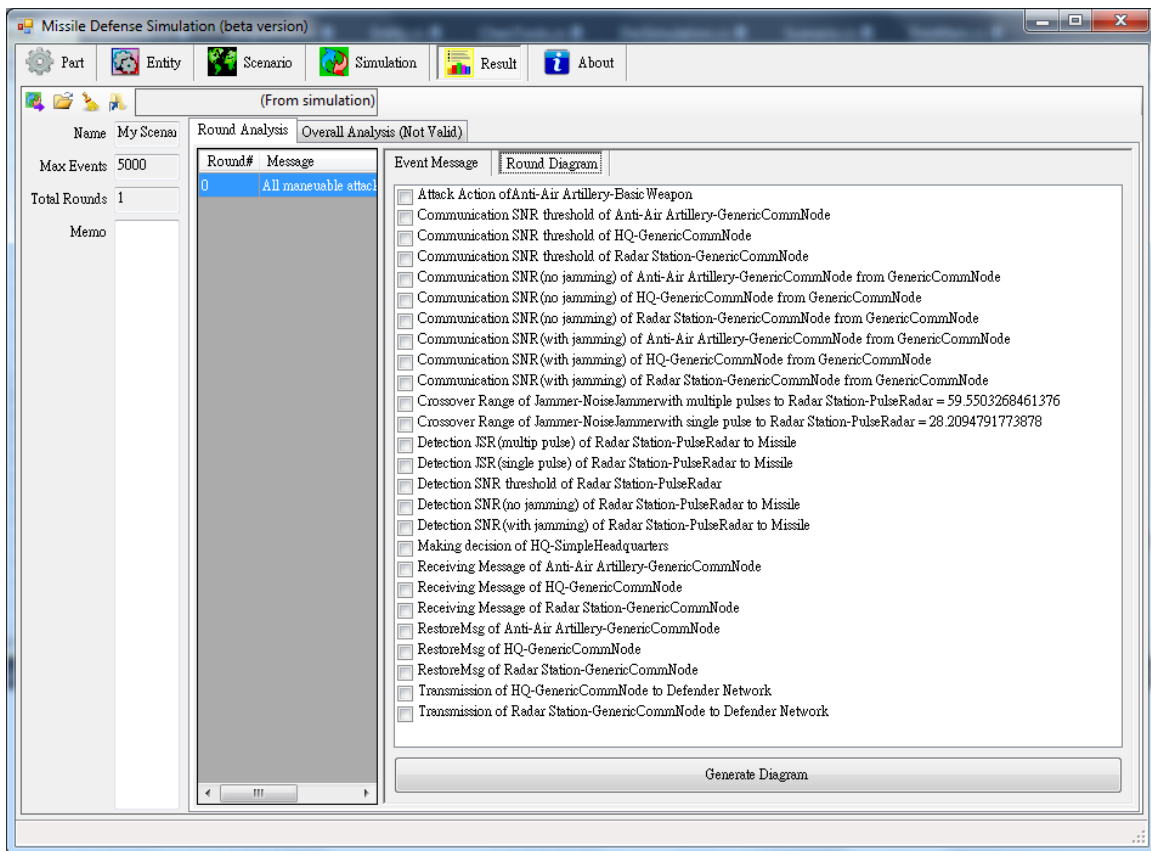


Figure 116. Data collected in simulation can be selected to generate a diagram.

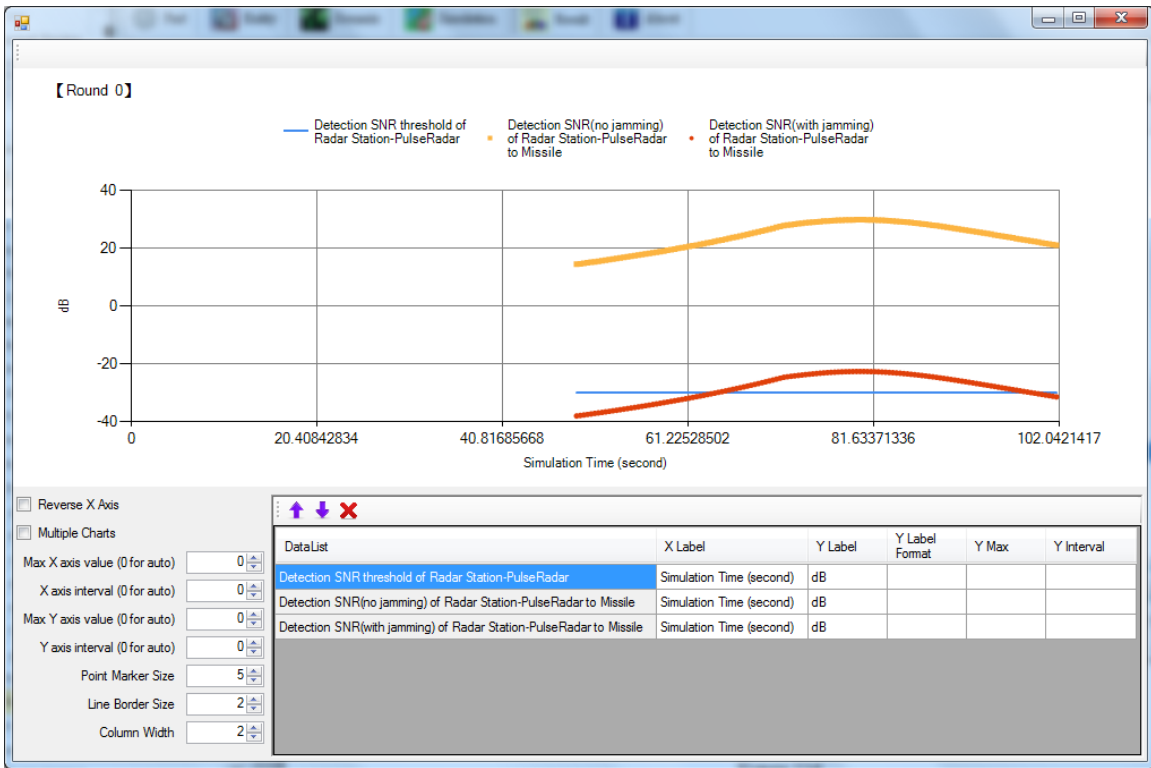


Figure 117. The generic diagram form can show data collected in a simulation.

## APPENDIX B: USER MANUAL OF THE $k$ -COVERED RATE CALCULATION PROGRAM

### A. INTRODUCTION

This document is dedicated to introduce the  $k$ -covered rate calculation program used in this dissertation. Readers should understand manipulate it after reading this document. The concept is introduced in next section, followed by a tutorial showing how to use it..

### B. WHAT THIS PROGRAM IS

#### 1. Background

This program is developed by You-Quan Chen to support his Ph.D. dissertation in the Naval Postgraduate School in 2013. This research provides an improved algorithm to accommodate binary, probabilistic, omnidirectional, and sectorial sensor models.

#### 2. File Structure

The file structure of the calculation program is shown in Table 28.

Table 28. File structure of the calculation program

| Folder\File Name  | Description  |
|-------------------|--|
| DetectionModels   | Files define the detection probability distribution. So far only binary and linear probability models are provided. User can define new models to extend program capabilities. |
| starter.m         | Access point   |
| main.m & main.fig | Main user interface  |
| UML.vsd           | Design support document  |
| Other m files     | Class files  |



### 3. Installation and Execution

- MATLAB 2012b is needed before execution.
- Execute the MATLAB
- Switch to program folder
- Add the “DetectionModels” to path
- Run the “starter.m” file

### C. TUTORIAL: FROM DETECTION TO FIRE

The initial program interface is shown in Figure 118. The region properties are set in the top-left area that contains a table for sensor setting. Attributes in the table are explained in Table 29.

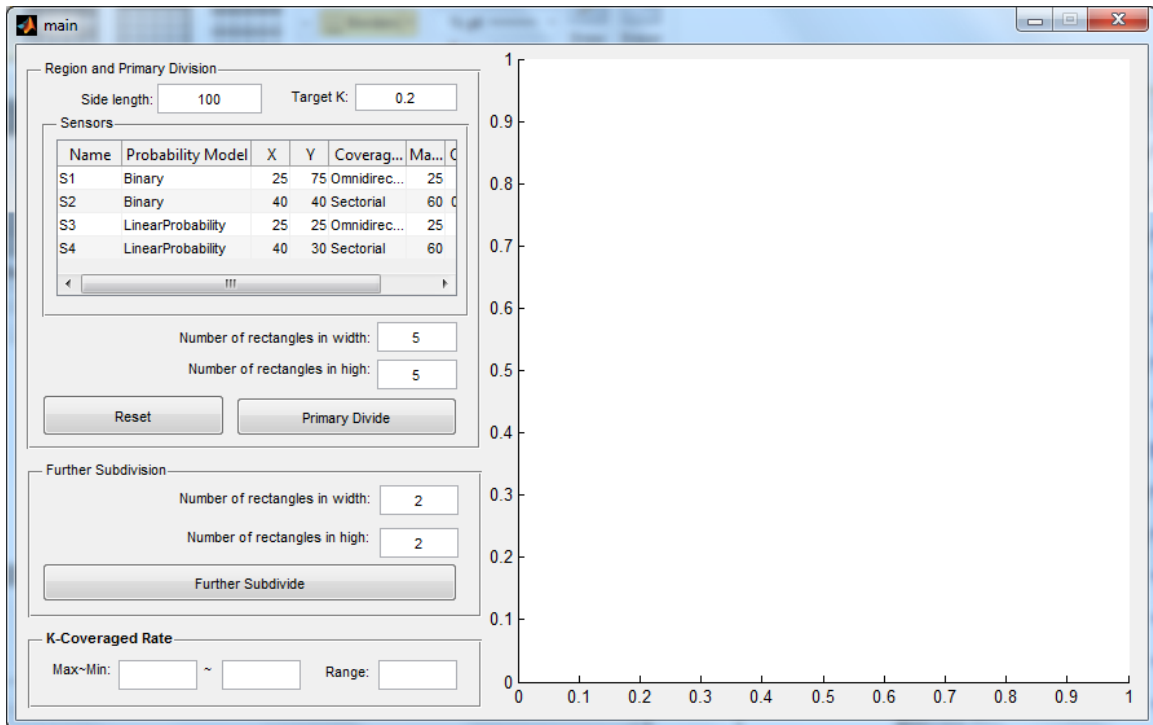


Figure 118. The initial program interface

Table 29. Attributes of sensors

| Sensor Attributes          | Description   |
|----------------------------|---|
| Name                       | Name of sensors   |
| Probability Model          | Name of detection models that stored in the “DetectionModels” files   |
| X, Y                       | Location of sensors   |
| Coverage Type              | Coverage type of sensors, either omnidirectional or sectorial.  |
| Max Range                  | Max ranges of sensors   |
| Orientation (rad)          | orientation   |
| Sectorial Side Angle (rad) | The angle of a sectorial coverage beam from axis to one side. Only available when the “Coverage Type” is sectorial. |

Click “Primary Divide” button to perform an initial division. The result of grid scan is shown in the right hand side (shown in Figure 119). Note that the estimation value is shown in the left-bottom area.

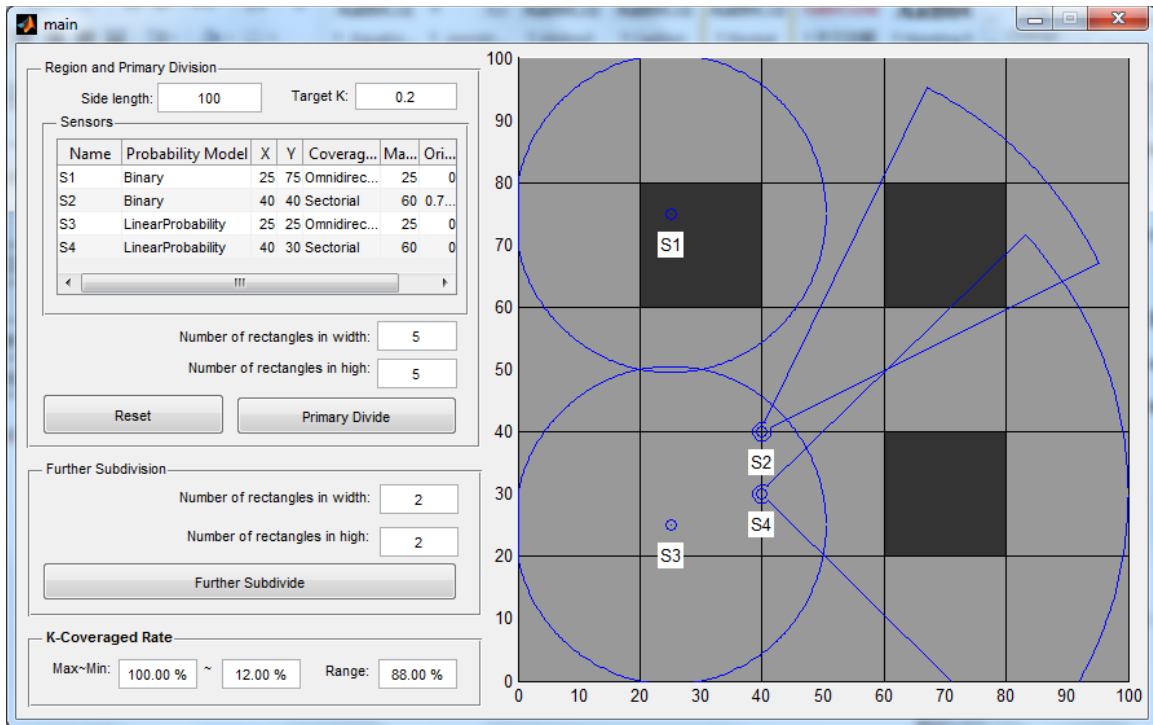


Figure 119. The result of grid scan is shown in the right hand side.

Click “Further Subdivide” for a second division. The scan result and estimation result are then updated as well. If more precise estimation is desired, more performing further subdivisions are allowed.

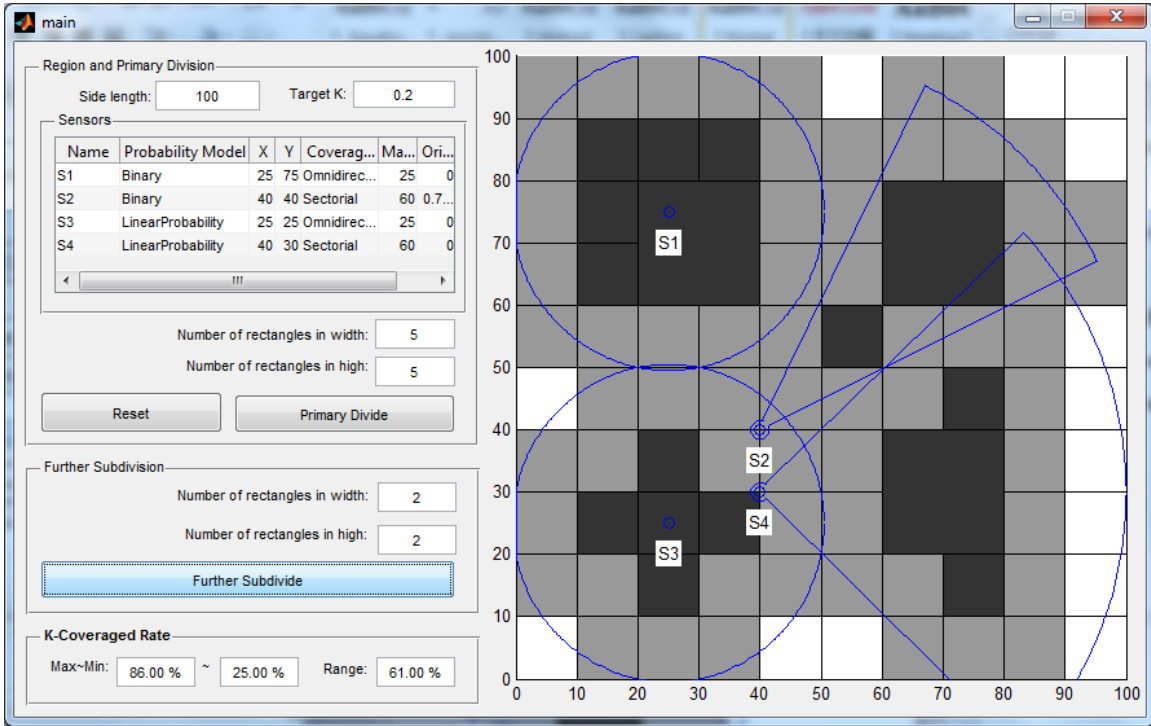


Figure 120. The scan result and estimation result are updated after a further division.

## **APPENDIX C: SOFTWARE AVAILABILITY OF EXAMPLE SIMULATION PROGRAM**

### **A. ACCESS ELIGIBILITY**

This software is used to demonstrate the improved DES algorithm (AEMF-DES) proposed in this dissertation. Request for access can be sent to:

- Dr. Don Brutzman (brutzman@nps.navy.mil)
- Dr. Phillip Pace (pepace@nps.edu).

### **B. SOFTWARE LICENSE AND DISCLAIMER**

Copyright (c) 2012-2013 held by the author. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted to all faculty staff and students of Naval Postgraduate School with that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the names of the Naval Postgraduate School (NPS) Modeling Virtual Environments and Simulation (MOVES) Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,

INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Reference information about the rationale for this software license can be found in the Savage developer Guide under licenses

(<https://savage.nps.edu/Savage/developers.html>)

### **C. AVAILABLE AT SOURCE ARCHIVE**

<https://savage.nps.edu/theses/chen>

### **D. LIST OF ASSETS**

- Source Code
- Executable Files
- Dissertation
- Dissertation Defense Slides (contains example experiment videos)

### **E. VIDEO DESCRIPTION**

The three videos included in the assets are attachment of the slides. Each one is design for different purposes:

- *Experiment 1.avi*: This video demonstrates the feasibility of AEMF-DES. It first builds few simple custom entities and then defines a scenario that is not predefined in MDSIM.
- *Experiment 2.avi*: This experiment illustrates how C4ISR processes are

simulated in MDSIM. It builds a basic scenario containing all types of parts to perform an integrated C4ISR simulation.

- *Experiment 3.avi*: This experiment shows electronic warfare (EW) issues. A jammer is added to the missile that can jam the defender's radar.

THIS PAGE INTENTIONALLY LEFT BLANK

## LIST OF REFERENCES

- Al Rowaei, A. A., Buss, A. H., & Lieberman, S. (2011). The effects of time advance mechanism on simple agent behaviors in combat simulations. *Proceedings of the 2011 Winter Simulation Conference*, 2426–2437.  
doi:10.1109/WSC.2011.6147952
- Asatani, M., Sugimoto, S., & Okutomi, M. (2011). Real-time step edge estimation using stereo images for biped robot. *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4463–4468.  
doi:10.1109/IROS.2011.6095092
- Bolkcom, C. (2005). *CRS report for congress military suppression of enemy air defenses (SEAD): Assessing future needs* ( No. RS21141). Congressional Research Service Reports, Washington, DC.
- Bozlu, B., & Demirörs, O. (2008). A conceptual modeling methodology: From conceptual model to design. *Proceedings of the 2008 Summer Computer Simulation Conference*, 1–11.
- Brooks, C. A., Chesir, A., Lindy, E., McConnell, J. A., & Farah-Stapleton, M. (2006). Design approach to implement implicit traffic in a simulation environment. *Proceedings of the 2006 Winter Simulation Conference*, 1234–1239.  
doi:10.1109/WSC.2006.323218
- Buss, A. H., & Al Rowaei, A. (2010). A comparison of the accuracy of discrete event and discrete time. *Proceedings of the 2010 Winter Simulation Conference (WSC)*, 1468–1477. doi:10.1109/WSC.2010.5679045
- Buss, A. H., & Sanchez, P. J. (2002). Building complex models with LEGOs (listener event graph objects). *Proceedings of the 2002 Winter Simulation Conference*, , 1 732–737 vol.1. doi:10.1109/WSC.2002.1172954
- Buss, A. H., & Sánchez, P. J. (2005). Simple movement and detection in discrete event simulation. *Proceedings of the 37th Conference on Winter Simulation*, 992–1000.
- Cassandras, C. G. (1993). *Discrete event sys modeling & performance*. Hillsboro, OR.: CRC Press.
- Cellier, F., & Kofman, E. (2006). *Continuous system simulation*. New York: Springer Science.
- Chang, T. G. (2005). Exploring C4ISR employment methods. *Proceedings of the 37th Conference on Winter Simulation*, 1205–1212.



- Chasteen, L. (2012). Use system engineering, not politics, for missile defense discussions. *2012 IEEE International Technology Management Conference (ITMC)*, 73–77. doi:10.1109/ITMC.2012.6306385
- Chen, Y., & Pace, P. E. (2008). Simulation of network-enabled radar systems to assess the value of jamming in a general radar topology. *2008 IEEE International Conference on System of Systems Engineering*, 1–5.
- Chu Wei. (2011). Efficiency evaluation method of simulation system based on BP neural network. *2011 Second International Conference on Networking and Distributed Computing (ICNDC)*, 41–44. doi:10.1109/ICNDC.2011.16
- Clive Wood, Patricio Jiménez López, Heliodoro Ruipérez Garcia, & Jan van Geest. (2008). Developing a federation to demonstrate the NATO live, virtual and constructive concept. *Proceedings of the 2008 Summer Computer Simulation Conference*,
- COMSOL. (2013). Multiphysics modeling, finite element analysis, and engineering simulation software. Retrieved July 25, 2013, from <http://www.comsol.com/products/multiphysics/>
- Davis, P. K., Bigelow, J. H., & McEver, J. (2000). Informing and calibrating a multiresolution exploratory analysis model with high resolution simulation: The interdiction problem as a case history. *Proceedings of 2000 Winter Simulation Conference*, 1 316–325 vol.1. doi:10.1109/WSC.2000.899734
- Delaney, W., & Vaccari, E. (1989). *Dynamic models and discrete event simulation* New York and Basel: Marcel Dekker, Inc.
- Devore, J. (2004). *Probability and statistics* (6<sup>th</sup> ed.) Brooks/Cole, Thomson Learning Inc.
- Difranco, J. V., & Kaiteris, C. (1981). Radar performance review in clear and jamming environments. *IEEE Transactions on Aerospace and Electronic Systems*, 17(5), 701–710.
- Dimarogonas, J. (2004). A theoretical approach to C4ISR architectures. *2004 IEEE Military Communications Conference*, 1 28–33 Vol. 1. doi:10.1109/MILCOM.2004.1493242
- Donnell, C. (2008). Defining, conceptualizing, and measuring fidelity of implementation and its relationship to outcomes in K–12 curriculum intervention research. *Review of Educational Research*, 78(1), 33–84. doi:10.3102/0034654307313793
- Donnelly, R. E. (2009). Bridging live and simulated domains with a common integration approach. *Proceedings of the 2009 Spring Simulation Multiconference*, 1–10.

- Dufour, C., Andrade, C., & Bélanger, J. (2010). Real-time simulation technologies in education: A link to modern engineering methods and practices. *Proceedings of the 11th International Conference on Engineering and Technology Education*, Bahia, Brazil.
- EM Software & Systems. (2013). FEKO - EM simulation software. Retrieved 7/25, 2013, from <http://www.feko.info/>
- Erdemli, M. G. (2009). *General use of UAS in EW environment--EW concepts and tactics for single or multiple UAS over the net-centric battlefield*. Naval Postgraduate School.
- Fanfan Yao, Renjie Xu, Qiang Liang, & Lanjuan Tong. (2012). Some key issues on modeling and simulation of military communication network. *2012 International Conference on Computer Science and Electronics Engineering (ICCSEE)*, 1 532–535. doi:10.1109/ICCSEE.2012.385
- Ferenci, S. L., Myung Choi, Evans, J., Fujimoto, R. M., Alspaugh, C., & Legaspi, A. K. (2004). Experiences integrating NETWARS with the naval simulation system using the high level architecture. *2004 IEEE Military Communications Conference*, 3 1395–1401 Vol. 3. doi:10.1109/MILCOM.2004.1495146
- Fishman, G. (2001). *Discrete-event simulation: Modeling, programming and analysis* New York: Springer–Verlag.
- Fishwick, P. (2007). *Handbook of dynamic system modeling* Boca Raton, FL: Chapman & Hall/CRC.
- Fusano, A., Sato, H., & Namatame, A. (2011). Study of multi-agent based combat simulation for grouped OODA loop. *2011 Proceedings of SICE Annual Conference (SICE)*, 131–136.
- Giampapa, J. A., Sycara, K. P., Owens, S. R., Grinton, R. T. E., Seo, Y. -, Yu, B., . . . Lewis, C. M. (2005). An agent-based C4ISR testbed. *2005 8th International Conference on Information Fusion*, 2 8 pp. doi:10.1109/ICIF.2005.1592030
- Gross, D. C. (1999). *Fidelity implementation study group report*. ( No. SISO-REF-002–1999). Simulation Interoperability Standards Organization (SISO).
- Haitao Yang, Xinmin Wang, & Hongli Zhao. (2006). The modeling, simulation and effectiveness evaluation for communication networks of multi-layer satellites constellation. *2006 International Conference on Computational Intelligence and Security*, 2 1055–1060. doi:10.1109/ICCIAS.2006.295424
- Hock-koon, A., & Oussalah, M. (2010). Defining metrics for loose coupling evaluation in service composition. *2010 IEEE International Conference on Services Computing (SCC)*, 362–369. doi:10.1109/SCC.2010.17

- Jiang Jin-long, Zhou Xian-zhong, & Sun yong-cheng. (2004). The improvement of multi-federations architecture for C4ISR simulation system. *2004 8th Control, Automation, Robotics and Vision Conference*, 2 1384–1387 Vol. 2. doi:10.1109/ICARCV.2004.1469049
- Jing Liu, Ai-Min Luo, & Xue-Shan Luo. (2009). A new military decision-making model from cognitive perspective. *2009 International Conference on Machine Learning and Cybernetics*, 2 881–884. doi:10.1109/ICMLC.2009.5212390
- Jun Wang, Xiaozhe Zhao, Yinhan Zhang, & Buyun Wang. (2011). A model of cooperative air-defense system of system for surface warship formation based on immune multi-agent. *2011 Fourth International Workshop on Advanced Computational Intelligence (IWACI)*, 473–476. doi:10.1109/IWACI.2011.6160053
- Katopodis, P., Katsis, G., Walker, O., Tummala, M., & Michael, J. B. (2007). A hybrid, large-scale wireless sensor network for missile defense. *2007 IEEE International Conference on System of Systems Engineering*, 1–5. doi:10.1109/SYSOSE.2007.4304261
- Kotenko, I. (2007). Multi-agent modelling and simulation of cyber-attacks and cyber-defense for homeland security. *2007 4th IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, 614–619. doi:10.1109/IDAACS.2007.4488494
- Kuhl, F., Weatherly, R., & Dahmann, J. (1999). *Creating computer simulation systems-an introduction to the high level architecture*. Upper Saddle River: Prentice Hall PTR.
- Lambeth, B. S. (Summer 2002). Kosovo and the continuing SEAD challenge. *Aerospace Power Journal*, 16(2), 8–21.
- Larocque, G. R., & Lipoff, S. J. (1996). Application of discrete event simulation to network protocol modeling. *1996 5th IEEE International Conference on Universal Personal Communications, 1996. Record*, 2 508–512 vol.2. doi:10.1109/ICUPC.1996.562625
- Law, A. M., & Kelton, W. D. (2000). *Simulation modeling and analysis* (3<sup>rd</sup> ed.) New York: McGraw-Hill.
- Lei, Z., & Hongzhou, J. (2012). Variable step euler method for real-time simulation. *2012 2nd International Conference on Computer Science and Network Technology (ICCSNT)*, 2006–2010. doi:10.1109/ICCSNT.2012.6526312
- Leskiw, D. M., Pflug, D. R., & Sisti, A. F. (1999). *Extreme simulation of C4ISR communications and networking*. ( No. ADA458044).U.S. Department of Commerce.

- Likuan Xie, Wei Yang, Shuangyan Hu, & Chengjian Li. (2009). Complex system simulation based on petri net combined with HLA. *2009 First International Workshop on Education Technology and Computer Science*, 3 205–208. doi:10.1109/ETCS.2009.572
- Lozoya, C., Velasco, M., & Marti, P. (2008). The one-shot task model for robust real-time embedded control systems. *IEEE Transactions on Industrial Informatics*, 4(3), 164–174. doi:10.1109/TII.2008.2002702
- Lucas, T. W. (2000). The stochastic versus deterministic argument for combat simulations: Tales of when the average won't do. *Military Operations Research*, 5(3), 9–28.
- Ma Wei-bing, & Zhu Yi-fan. (2011). Interoperability of the simulation-based training support environment with C4ISR system. *2011 International Conference on Electric Information and Control Engineering (ICEICE)*, 1432–1437. doi:10.1109/ICEICE.2011.5777852
- Mei Dan, Xuesong Wang, & Shun-ping Xiao. (2008). Simulation and evaluation of phased array radar's ballistic missile defense capability. *2008 International Conference on Information and Automation*, 911–915. doi:10.1109/ICINFA.2008.4608129
- Moen, D. M., & Pullen, J. M. (2005). Modeling real-time distributed simulation message flow in an open network. *2005 Ninth IEEE International Symposium on Distributed Simulation and Real-Time Applications*, 97–104. doi:10.1109/DISTRA.2005.29
- Moffat, J. (2007). Modelling human decision-making in simulation models of conflict. *International C2 Journal*, 1(1), 31–60.
- Mofrad, R. F., & Sadeghzadeh, R. A. (2010). Scenario modeling and simulation for performance prediction of a modern radar in electronics warfare environment. *2010 11th International Radar Symposium (IRS)*, 1–5.
- National Research Council. (2006). *C4ISR for future naval strike groups* The National Academies Press.
- Nguyen, B., & Yip, H. (2010). A stochastic model for layered defense: Ballistic missile defense and harbor protection. *2010 International Waterside Security Conference (WSS)*, 1–8.
- Niazi, M., & Hussain, A. (2011). Agent-based computing from multi-agent systems to agent-based models: A visual survey. *Scientometrics*, 89(2), 479–499.
- Ong, A., Wong, A., Darken, C., & Buss, A. (2010). Plans validation using DES and agent-based simulation.

- Ouellette, D. S., Wierckx, R. P., & McLaren, P. G. (2008). Using a multi-threaded time-step to model a multi-function relay in a real time digital simulator. *2008. IET 9th International Conference on Developments in Power System Protection*, 162–167.
- Pace, P. E. (2009). *Detecting and classifying low probability of intercept radar* (2<sup>nd</sup> ed.). Boston: Artech House.
- Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000). A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 30, 286–297.
- Qi, X., Sun, Y., Xiong, Z., Yu, L., & Li, X. (2008). Research on simulation of searching radars' intelligence detection in complicated electromagnetic environment. *2008 Asia Simulation Conference - 7th International Conference on System Simulation and Scientific Computing*, 918–923.
- Rouche, P. J. (2009). *Fundamentals of verification and validation* Hermosa Publishing.
- Sanchez, S. M., & Lucas, T. W. (2002). Exploring the world of agent-based simulations: Simple models, complex analyses. *Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers*, 116–126.
- Santiago, J. (2009). Stimulation of live C4ISR experimentation environments by using HPC simulation. *2009 DoD High Performance Computing Modernization Program Users Group Conference (HPCMP-UGC)*, 332–336. doi:10.1109/HPCMP-UGC.2009.54
- Sari, S. C., Kuspriyanto, & Prihatmanto, A. S. (2012). Decision system for robosoccer agent based on OODA loop. *2012 International Conference on System Engineering and Technology (ICSET)*, 1–7. doi:10.1109/ICSEngT.2012.6339299
- Seo, K., Song, H. S., Kwon, S. J., & Kim, T. G. (2012). Measurement of effectiveness for an anti-torpedo combat system using a discrete event systems specification-based under water simulator. *Journal of Defense Modeling and Simulation*, 8, 157–171.
- Shaojie Mao, Guangyu Bao, & ZhenQi Ju. (2008). A study of simulation and evaluation experimental method of C4ISR system. *2008 Asia Simulation Conference - 7th International Conference on System Simulation and Scientific Computing*, 793–796. doi:10.1109/ASC-ICSC.2008.4675469
- Shaulov, G., & Patel, J. (2007). Simulation-based performance robustness studies for optical gigabit LAN in military applications. *2007 IEEE Military Communications Conference*, 1–7. doi:10.1109/MILCOM.2007.4455247
- Shen Tong-yun, Ding Jian-jiang, Ding Yuan, & Shi Jian-gui. (2011). A method of detection performance modeling in jamming condition based on radar network system. *2011 IEEE CIE International Conference on Radar (Radar)*, 2 1366–1369.

- Shen, X., Chen, J., & Sun, Y. (2006). Grid scan: A simple and effective approach for coverage issue in wireless sensor networks. *2006 IEEE International Conference on Communications*, 8 3480–3484.
- Sheridan, T. B. (2002). *Humans and automation: System design and research issues* John Wiley & Sons, Inc.
- Sheu, J., Chang, G., & Chen, Y. (2008). A novel approach for k-coverage rate evaluation and re-deployment in wireless sensor networks. *2008 IEEE Global Telecommunications Conference*, 1–5.
- Sklar, B. (2001). *Digital communications : Fundamentals and applications* (2<sup>nd</sup> ed.) Prentice Hall.
- Skolnik, M. (2001). *Introduction to radar systems* (3<sup>rd</sup> ed.) New York: McGraw-Hill.
- Stein, F., Garska, J., & McIndoo, P. L. (2000). Network-centric warfare: Impact on army operations. *EUROCOMM 2000. Information Systems for Enhanced Public Safety and Security. IEEE/AFCEA*, 288–295. doi:10.1109/EURCOM.2000.874819
- Sun Li-yang, Mao Shao-jie, Liu Zhong, & Deng Ke-bo. (2010). Research on the runtime support platform for the net-centric simulation. *2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, 1 V1–253-V1–257. doi:10.1109/ICACTE.2010.5579021
- Sung-Ho Jee, & Montague, P. (1999). A secure DES implementation for real-time embedded applications. *1999 Third International Conference Knowledge-Based Intelligent Information Engineering Systems*, 496–500. doi:10.1109/KES.1999.820231
- Tajwer, K., & Shamsi, J. (2010). Counter-terrorism simulation framework. *2010 International Conference on Information and Emerging Technologies (ICIET)*, 1–6.
- Tavakoli, S., Mousavi, A., & Komashie, A. (2008). A generic framework for real-time discrete event simulation (DES) modelling. *2008 Winter Simulation Conference*, 1931–1938. doi:10.1109/WSC.2008.4736285
- Teng Fei, & Wang Chang Hong. (2002). Modeling and real time control of Internet-distributed control system. *IEEE 2002 28th Annual Conference of the Industrial Electronics Society*, 3 2392–2396 vol.3. doi:10.1109/IECON.2002.1185347
- U.S. Department of Defense. (2001). *RTI 1.3-next generation programmer's guide version 4* U.S. Defense Modeling and Simulation Office.
- U.S. Department of Defense. (2008). *Bradley stinger fighting vehicle platoon and squad operations* U.S. Army.

- U.S. Department of Defense. (2010). *Air force program elements: EW development*. ( No. 0604270f).U.S. Air Force.
- U.S. Department of Defense. (2011). *Modeling and simulation (M&S) verification, validation, and accreditation (VV&A) recommended practices guide (RPG)*. Washington, DC: Author.
- U.S. Department of Defense. (2013). *Dictionary of military and associated terms, department of defense*. Washington, DC: Author.
- Wang Chang-chun, Chen Chao, Chen Jun-liang, & Zhang Ying-xin. (2010). Analysis and simulation of combat systems paralysis based on complex network. *2010 International Conference on Information Networking and Automation (ICINA)*, 2 V2–286-V2–290. doi:10.1109/ICINA.2010.5636507
- Wang Cong, Wang Xiao-guo, Yu Zhen-wei, Liu Jun, & Wang Zhi-xue. (2010). UML-based C4ISR capability requirement analysis. *2010 International Conference on Computer Application and System Modeling (ICCSM)*, 11 V11–445-V11–448. doi:10.1109/ICCSM.2010.5623168
- Wang, H., & Chung, W. (2012). The generalized k-coverage under probabilistic sensing model in sensor networks. *2012 IEEE Wireless Communications and Networking Conference (WCNC)*, 1737–1742.
- Wei Su, Xue-Shan Luo, & Yao-Hong Zhang. (2006). Research on simulation based C4ISR system integrated design technology. *2006 International Conference on Machine Learning and Cybernetics*, 885–889. doi:10.1109/ICMLC.2006.258491
- Weijun Zhang, Zhicheng Wan, Jianjun Yuan, & Zhixia Tang. (2007). An improved rate-monotonic scheduler and a real-time open control platform for robotic control algorithm verification. *2007 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 318–323. doi:10.1109/ROBIO.2007.4522181
- Werrell, K. P. (2005). *Archie to SAM* (2<sup>nd</sup> ed.) Air University Press, Maxwell Air Force Base, Alabama.
- Westermann, E. B. (2001). Edward westerman, “flak: German anti-aircraft defenses. *University Press of Kansas*, 1914–1945.
- Wiedemann, M. (2010). *Thesis: Robust parameter design for agent-based simulation models with application in a culture geography model*. Monterey, U.S.: Naval Postgraduate School.
- Wijesekera, D., Michael, J. B., & Nerode, A. (2005). An agent-based framework for assessing missile defense doctrine and policy. *2005 Sixth IEEE International Workshop on Policies for Distributed Systems and Networks*, 115–118. doi:10.1109/POLICY.2005.4

- Xiao-Hui Bai. (2008). An application with UML object-based petri nets for C4ISR architecture simulation validation. *2008 International Conference on Machine Learning and Cybernetics*, 4 2257–2263. doi:10.1109/ICMLC.2008.4620781
- Xiaohui Bai. (2008). Study of C4ISR architecture simulation validation with UML and object-based petri nets. *2008 IEEE 8th International Conference on Computer and Information Technology Workshops*, 571–576. doi:10.1109/CIT.2008.Workshops.61
- Xin Jiang, Zhi-hui Yin, Jun-liang Chen, & Xin-jian Liu. (2010). Analyzing of effect factors of fleet collaborative decision-making based on simulation experiments. *2010 International Conference on E-Product E-Service and E-Entertainment (ICEEE)*, 1–4. doi:10.1109/ICEEE.2010.5660595
- Yang Juniang, Du Jia, & Li Zhanwei. (2009). Research on indexes system for evaluating C<sup>4</sup>ISR system for NCW. *2009 International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, 1 459–462. doi:10.1109/IHMSC.2009.122
- Yang Qing-wen, Dong Fei, Li Hui-xiang, & Yan Yan. (2009). Design and realization of rocket battalion's communication network simulation based on UML. *2009 International Conference on Computational Intelligence and Software Engineering*, 1–5. doi:10.1109/CISE.2009.5362983
- Yi Deng, Hua Lin, Phadke, A. G., Shukla, S., Thorp, J. S., & Mili, L. (2012). Communication network modeling and simulation for wide area measurement applications. *2012 IEEE PES Innovative Smart Grid Technologies (ISGT)*, 1–6. doi:10.1109/ISGT.2012.6175664
- Yuanzheng Ge, Xiaogang Qiu, & Kedi Huang. (2010). Conceptual interoperability model of NCW simulation. *2010 IEEE International Conference on Information Theory and Information Security (ICITIS)*, 911–914. doi:10.1109/ICITIS.2010.5689791
- Zarchan, P. (2002). *Tactical and strategic missile guidance* (4<sup>th</sup> ed.) American Institute of Aeronautics and Astronautics, Inc.
- Zeigler, B., Kim, T. G., & Praehofer, H. (2000). *Theory of modeling and simulation* (2<sup>nd</sup> ed.). New York: Academic Press.
- Zhang Chengbin, Luan Liqiu, & Zhang Hongzhou. (2010). Multi-agent technology based design of hierarchy model of groups/teams in air defense system. *2010 International Conference on Audio Language and Image Processing (ICALIP)*, 1403–1406.
- Zhang Ying-chao, Ye Feng, Yu Qin-zhang, & Chen Xin. (2010). Research on multi-agent based C4ISR effectiveness simulation and evaluation. *2010 Second International Conference on Computer Modeling and Simulation*, 1 561–568. doi:10.1109/ICCMS.2010.262



THIS PAGE INTENTIONALLY LEFT BLANK

## **INITIAL DISTRIBUTION LIST**

1. Defense Technical Information Center  
Ft. Belvoir, Virginia
2. Dudley Knox Library  
Naval Postgraduate School  
Monterey, California