*Proceedings of the 2013 Winter Simulation Conference*
*R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds*

# RECONFIGURABLE C3 SIMULATION FRAMEWORK : INTEROPERATION BETWEEN C2 AND COMMUNICATION SIMULATORS

Bong Gu Kang
Tag Gon Kim

Dept. Electrical Engineering
KAIST
Daejeon, 305-701, Republic of Korea

## ABSTRACT

This paper presents a reconfigurable military simulation framework that reflects behaviors of a C3 system, consisting of command & control (C2) and communication (C). To achieve this goal, this paper identifies models in the C3 system and defines the interfaces between the models. In detail, we dispart a C2 simulation framework as three components: the C2, the military unit, and the communication agent model. We also suggest a new metamodel that can represent the dynamic property of the communication simulator. With the enhanced modularity of the C2 system and the new metamodel, users can assess various structures of the C2 system and rapidly calculate the diverse effects of communication. From case studies, we can test the new structure for the C2 system and gain improved performance with regard to speed by using the metamodel. Finally, we expect that this work will help decision makers evaluate various scenarios in a short time.

## 1 INTRODUCTION

Network-centric warfare (NCW) is a military doctrine or theory of war pioneered by the United States Department of Defense in the 1990s. The aim of NCW is translating an information advantage, enabled in part by information technology, into a competitive advantage. NCW integrates command & control (C2), communications, computers, intelligence, surveillance, and reconnaissance (C4ISR Architecture Working Group 1997). Figure 1 shows the battlefield of NCW based on this C4ISR structure. When the enemy emerges in the battle field, a sensor, such as an unmanned aerial vehicle (UAV), detects it and transmits the information to C2 through communication equipment. Then, C2 recognizes the situation from the detected information and assigns weapons and sends a fire request to a shooter, such as a fighter plane, using communication equipment. Last, the shooter attacks the enemy after receiving the fire order. We call this process the Sensor – C2 – Shooter cycle. In this field, the result of war can be changed depending on how fast this cycle rotates (Walsh et al. 2005), and the importance of C2 and communication in the rotation is increasing. For those reasons, to describe the field of NCW realistically, we need to model and simulate C3, which is made up communication and C2.

In the C3 system, the communication equipment takes charge of delivering information between a pair of C2s. Under this circumstance, the success, failure, and delay of transmissions can be changed depending upon the communication equipment and parameters; therefore, we should consider such effects of communication when constructing models and simulations. In the C2 system, the existing order structure has a latency problem due to many stages from uppermost to lowest C2 and vice versa. To solve those issues, the Korean government has suggested that new structures for C2 be developed (Ahn 2012).

Such structure can be tested without economical cost if using modeling and simulation. For such reasons, a method able to simulate C3 is needed that comprises communication and C2.

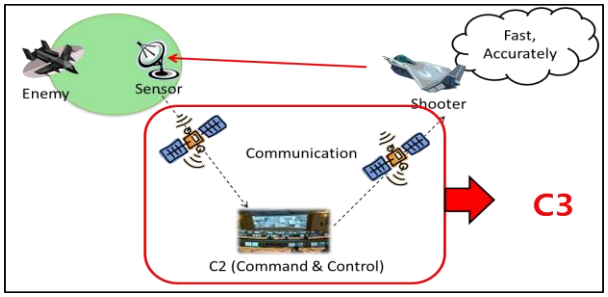


Figure 1: General engagement cycle : Sensor – command & control – shooter.

This paper therefore proposes a C3 simulation framework that responds to the above requirements, as diagrammed in Figure 2. Through this framework, users can test various C2 structures and analyze the various effects of communication upon the C2 system. Users can acquire precise results of simulation with the communication simulator, and they can also quickly obtain output with the metamodel instead of the communication simulator. Or, users can choose one of them according to the objective of the simulation.

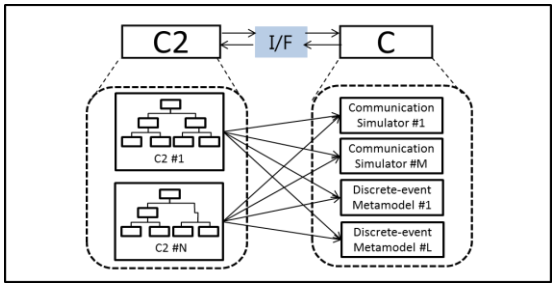


Figure 2: Overview of the proposed C3 simulation framework.

In the next section, we present related works about C3 simulation. Then, we will specifically argue the proposed C3 simulation framework in section 3 and show how the C3 simulation framework can be applied to various case studies in section 4. Finally, we summarize the points of this paper and ponder the application of this framework in various realm.

## 2 RELATED WORK

Some studies related to simulation of C3 already exist. Figure 3(a) is a schematic of Die Neue Framework Simulation (DNS) developed in Germany (Eberhard 2001). This framework uses HORUS, or training analysis in ground combat; FIT, or the C3 model; and ORIS, or ISR, models. But communication is expressed in FIT as a concise model, which limits the ability of DNS to express details regarding the effectiveness of the communication. To overcome this limitation, new methods have been suggested, as in Figure 3(b). This method was developed to take into account the interoperation between communication effects servers (CES) or communication simulators based on OPNET in the United Kingdom (Paz 2008), and One Semi-Automated Forces (OneSAF), which is a mission-level wargame simulator developed in the United States. This study uses a communication simulator as the communication model, and through this approach, precise communication effects can be calculated. However, only the C2 structures that are supported by OneSAF can be tested experimentally. To address these problems, a modifiable, C2 system

wargame simulator has been invented, as shown in Figure 3(c) (Kim et al. 2011). This approach takes into account the interoperation between the communication simulator and the modifiable wargame simulator. In case of communication, detailed communication effects can be represented by the simulator, and also, by separating from the war simulator, the communication simulator can be reused.

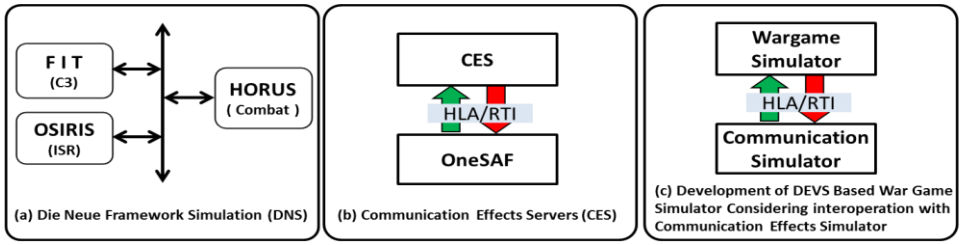


Figure 3: Comparison of related works for C3.

However, from the C2 structure perspective, the C2 model is incorporated into each command center model, as shown in Figure 4(a). Therefore, when we change the C2 structure, we should modify the structure of each command center, which is not efficient. Also, from the communication viewpoint, whereas the accuracy of communication can be increased, a long execution time is necessary for complex computations. As a result, these drawbacks constrain the types of cases that can be tested experimentally. To resolve these problems, a framework must be developed that satisfies two requirements: the framework must be applicable to various C2 structures, and it must enable quick computation. In this paper, we propose a C3 simulation framework that meets these requirements.

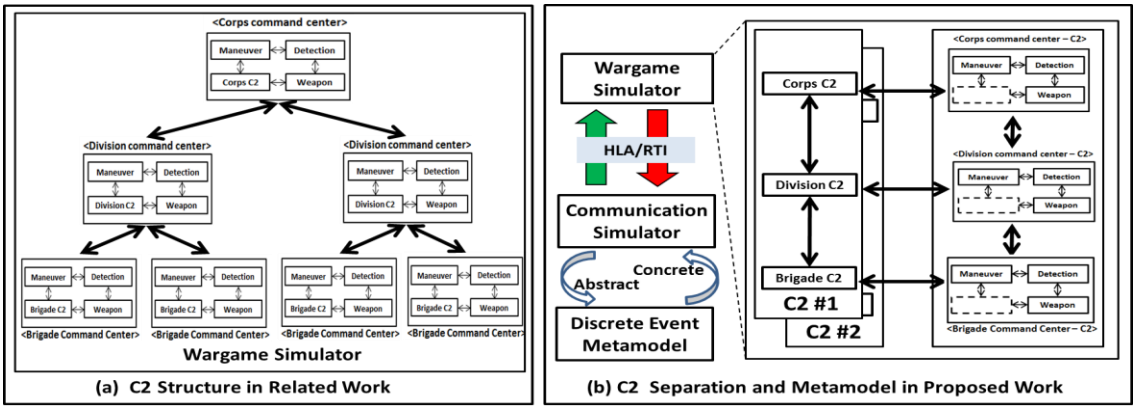


Figure 4: Difference between existing and proposed work.

In C2 structures, we separated C2 from the command center model as in Figure 4(b); in turn, we ensured modularity by constructing C2 models with components that can be changed as needed. Within this framework, users can experiment just by changing the inner structure. Meanwhile, a drawback in the communication simulator is that it requires a long execution time. To save time, we constructed a metamodel as shown in Figure 4(b). In the process of making a metamodel, we found that existing metamodel methods were limited in their ability to express the property of the simulator, because those methods assumed a static system (Kleijnen et al. 2000), even though the simulator is a dynamic system. For this system, we proposed a new metamodeling method called discrete event metamodeling. This paper will deal with the metamodel in section 3. Through these solutions, this study meets the two requirements outlined above.

## 3 C3 SIMULATION FRAMEWORK

The objective of the C3 simulation framework is to make it easy for users to analyze the C2 structure and communication effects that they want to test. To ensure the ease of testing various scenarios, we enhanced the modularity of the framework, thus making it reconfigurable. As a result, users can simulate a C3 system without adjusting the C2 or communication models. To be specific, users can analyze the structure of a C2 system through a C2 model and determine the effects of communication on wargame simulations with a communication model. Such a C3 simulation framework consists of C2, the C framework, and an interface, as shown in Figure 5. In the C simulation framework, there is a communication model with a communication simulator or discrete event metamodel as the inner model. If the user needs high accuracy, the simulator is suitable. If the user needs fast simulation execution time (e.g., to understand a tendency), the metamodel is appropriate. These results of communication are transferred through the interface. In this section, each framework and the interface between them will be described.

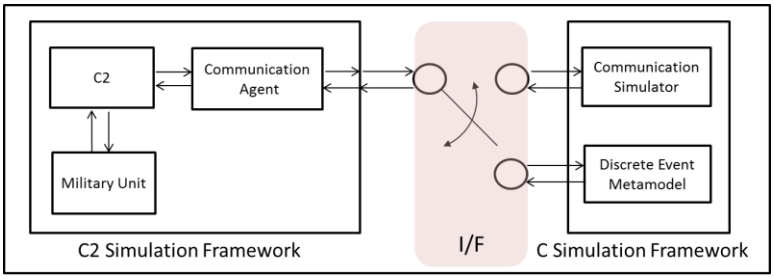


Figure 5: Proposed C3 simulation framework.

### 3.1 C2 Simulation Framework

#### 3.1.1 Model Description

The C2 simulation framework consists of C2, a military unit, and a communication agent model. To describe the behavior of precise communication, we separate this communication model from other models. Then, to achieve reconfigurability, we separate the C2 model from the military unit. By separating each component, we can enhance modularity.

The C2 model is set of function of C2 in each command center in the existing research. We represent each C2 model as a coupled model based on Discrete EVent System Specification (DEVS) because this formalism can easily express hierarchical structure (Zeigler et al. 2000). In this coupled model, there is a C2 atomic model, which is formalized to an atomic model of DEVS. This atomic model describes the general C2 process—that is, target identification, confirmation of target information and battle plans, and allocation of weapons against targets, as in Figure 6(b). These functions are expressed in one C2 atomic model, such as corps C2 and division C2. As depicted in Figure 6(b), C2 models can be different, depending on how C2s are connected. As a result, if users want to change the structure of the C2 framework, then they only need to change the inner C2 model, depending on the C2 structure scenario.

The military unit is a model that accepts the C2 model in the command center. This model is also based on DEVS formalism because it is suitable to the modularity. Just as in a real battlefield, each battle troop is composed of detection troops and engagement weapons, such as air defense weapons and artillery and so on. Also, they consist of functions such as maneuvering, detection, and combat, as diagrammed in Figure 6(b). By expressing troops as a coupled model and functions as an atomic model, the modularity of the model can be well portrayed and each troop can be reused. In the C3 simulation framework, the detection of troops plays a role in the reporting of detection information to the C2 model, and engagement weapons conduct attacks against targets based on fire orders from C2.

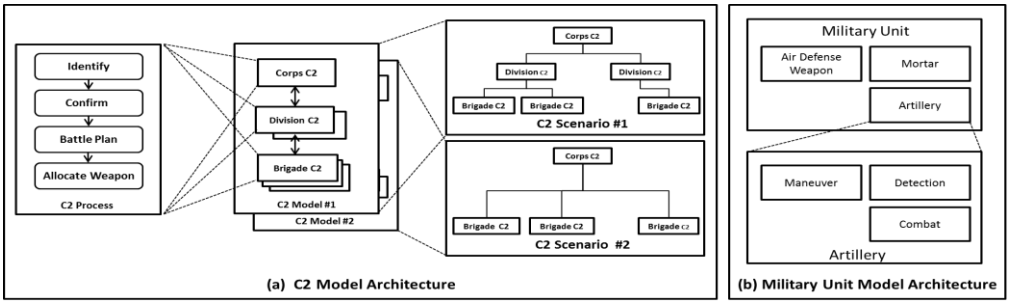


Figure 6: Architecture of models in the C2 simulation framework.

In a real battlefield, C3 comprises C2 and communication between the pair of C2s. By functionally dividing each other, they are separated as the C2 model and the communication agent model, which acts for communication as illustrated in Figure 7. If users utilize a disjoined architecture, as in the framework proposed by this paper, the modularity of the C2 model can be sustained. Furthermore, when changing the logic of the communication model, users can just modify the agent model. This agent model has two functions: to reflect communication effects on C2 and to retransmit if there is no response. The former means to apply that effect to an atomic model of the C2 model. For instance, in the case of transmission of messages from battalion to infantry, the communication agent receives the message and transmits it to the communication model, and then when the message from the communication model arrives, it applies that message to communication effects and delivery to the infantry C2. The latter function means to conduct retransmissions after C2 sends a message to the communication model if there is no acknowledgement from the communication model.

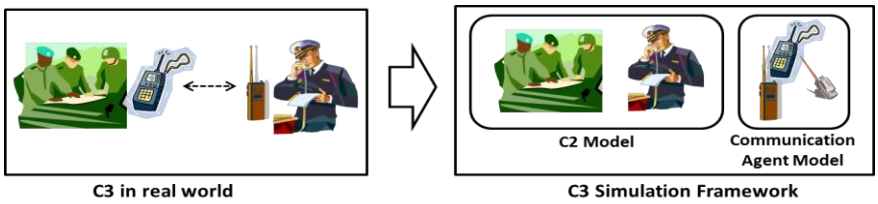


Figure 7: Separation of communication agent model.

To be more concrete, in a real battleground, if there is no response after a C2 sends a message to another C2, the C2 will repeat the transmission. To simulate these real-world events, the communication agent model checks the response time based on an inner clock. If a message returns to the agent from the communication model before the time set by the inner clock, it regards the transmission as successful and the agent sends a message with the calculated communication effects to another C2. If, however, the response time is over the time set by the inner clock, the transmission is considered a failure and a retransmission occurs based on a military policy. Because the real world has the property of resending, this framework also considers this function.

### 3.1.2 Interface Description

In the C2 simulation framework, the C2 model is connected with a communication agent and a military unit model. The four kinds of messages used among them are as follows in Table 1. A detection message includes information about a detected enemy, while a fire command message includes weapons allocated from C2 and the position of the target. A move order message is used to indicate maneuvering of machinery, and an update order message is used to report the present position to C2 when a military unit arrives at its destination. The messages passed between C2 and the communication agent model are detection and fire command messages. The former is used to report detection information to upper C2, while the latter

is needed for requesting fire support from upper C2 to another C2, which includes available weapons to attack. In case of messages between C2 and the communication model, move order and update messages are needed besides the above two types of messages. A move order is needed to move machinery from C2, and an update message is required to give a weapons arrival report to C2. These above messages are transmitted via the interface, as shown in Figure 8.

Table 1: Identification messages in C2.

| Message Type | Contents |
|---|---|
| Detection info | Enemy Information |
| Fire command | Weapon ID, Target Position |
| Move order | Destination Position |
| Update position | Weapon Position |

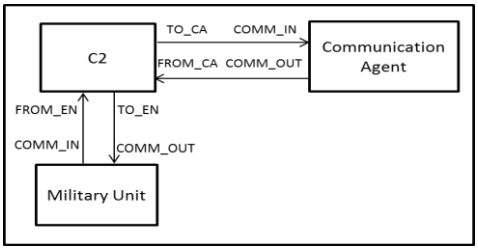


Figure 8: Interfaces between models in C2.

Because models in the C2 simulation framework are constructed as DEVS models, the user only makes couplings among them if he or she wants to connect them. In the case of the relationship between C2 and the agent model, the output of C2 is connected to the input of the agent model, and for this route, messages wanted to be reflected effect are sent to the agent model. Conversely, the output of the agent model is connected with the input of the C2 model, and through this interface, messages reflecting communication effects are delivered from the agent to the C2 model. Meanwhile, in the case of C2 and the military unit model, the output of C2 is linked to the input of the military model, and by this means, order can be delivered to the unit from C2. Conversely, the output of the military unit is attached to the input of the C2 model, and with them, target information from engagement or detection troops can be sent. To sum up, through this interface, if users just satisfy the interface among models, they can simulate diverse situations easily.

### 3.2 C Simulation Framework

#### 3.2.1 Model Description

The C simulation framework consists of the communication model, which plays a role in reflecting communication effects (e.g., delay, success or failure of transmission, etc.) on the C2 model. Such effects can vary with the parameters of the communication equipment. In this paper, the communication model can be a communication simulator or discrete event metamodel. The user chooses one of two alternatives according to the objective of the simulation. For example, in a circumstance requiring accurate results of simulation, the communication simulator is suitable, but it requires a long execution time. To save this time, the metamodel of the communication simulator can be utilized. For this case, the existing metamodels cannot describe this simulator because they can only be applied to static systems. However, this simulator is a dynamic system because output is only not determined by current input. To tackle this problem, a new metamodel called the discrete event metamodel is proposed.

#### 3.2.2 Discrete Event Metamodel

The discrete event metamodel we propose addresses the problem of the lengthy computation time required by extant metamodels. As shown in Figure 9, the communication simulator determines the internal property, such as the delay or success of a transmission, by using the parameter of the simulator as an input. Upon receiving a message from the wargame simulator, the communication simulator resends an updated message to that simulator after reflecting effects based on delay time and success rate. Therefore, in

order to construct the metamodel of that simulator, three functions should be satisfied. The first is to determine communication effectiveness based on communication parameters. Second is to describe behavior, such as the success of transmission. Last is to reflect delay time. Of the three functions, the second can be represented by static metamodeling, but the final cannot be expressed by existing methodology since this simulator is a dynamic system—that is, sequences of input messages can be different from the order of output.

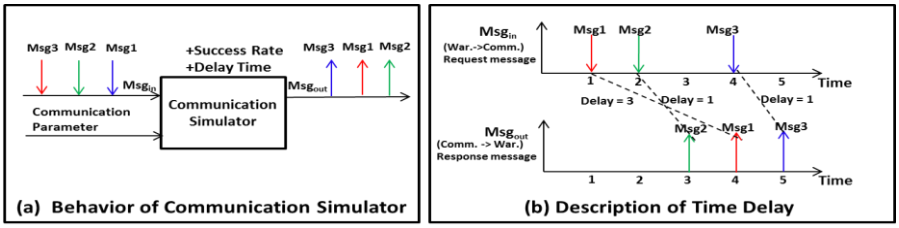


Figure 9: Input and output of communication simulator.

To make the metamodel satisfy the property of the above system, we suggest a new metamodel based on a discrete event system. To fulfill the characteristic of delay, we make a queue that manages the remaining time inside the metamodel, and when input or output events occur, the status of the queue is updated. This discrete event metamodel is expressed with DEVS formalism, as shown in Figure 10. The interesting thing is that inner status changes according to success rate, like $\delta_{int}^{P}$. Also, the queue varies whenever an input or output event occurs. For better understanding, we give a detailed account using a diagram of DEVS in Figure 10, which includes a description about functions used in the diagram. First, Update() is used to renew the success rate using input parameters, and CheckTransmissionSuccess() is used to determine either success or failure of transmission against the input message. InsertMessageInQueue()/PopMessage() is used to insert or pop the message from the queue, CheckQueueLength() is used to return the length of the queue, and UpdateReviseTA() is used to calculate the time when the next message is going out and update the queue accordingly.

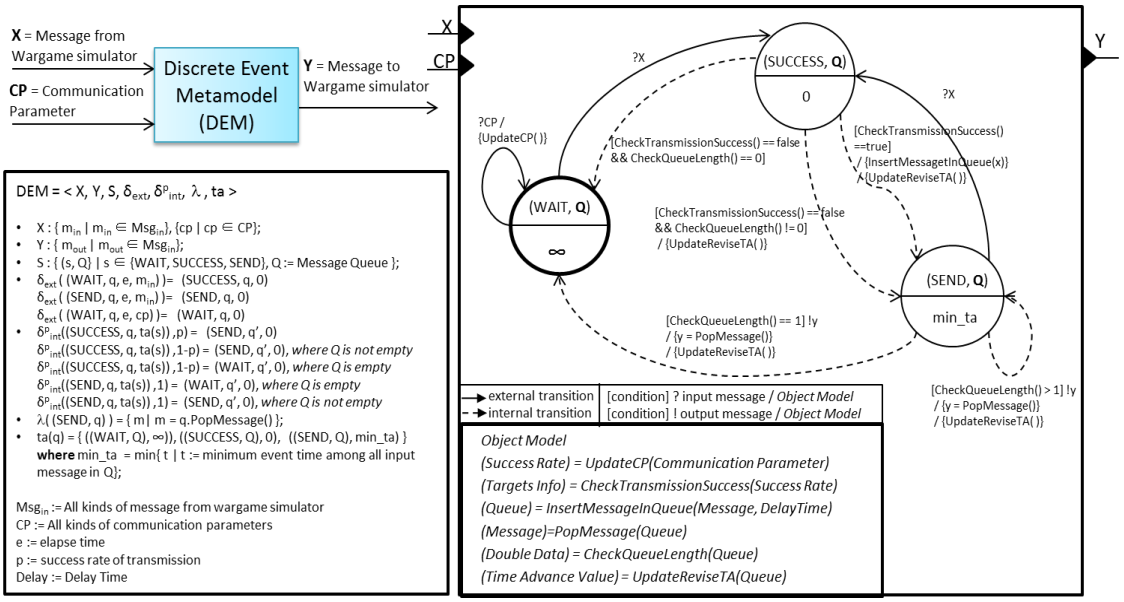


Figure 10: Discrete event metamodel representation: DEVS formalism and diagram.

As shown in Figure 10, this model starts with WAIT status and then, after receiving a parameter, calculates the success rate of transmission. In WAIT status, when entering a message, the status transfers to SUCCESS. In this status, it calls the CheckTransmissionSuccess() function; after that, if the transmission is successful, the status changes to SEND, and if unsuccessful, to WAIT. On the way to SEND, an input message is inserted into the queue with the InsertMessageInQueue() function. Calling the UpdateReviseTA() function updates the remaining time of messages in the queue. And then, remaining time value of message which is going out at first in the queue applies to min_ta. In SEND status, if there is no input event after min_ta time, the model sends output with the PopMessage() function and checks the length of the queue using CheckQueueLength(). If the queue is empty, the status changes to WAIT; if it is not empty, it changes to SEND. While in SEND status, if an external event occurs, the status changes to SUCCESS, and then the same process is recursive as before during simulation. Through this process, the discrete event metamodel can similarly express the behavior of the communication simulator.

### 3.3 Interface Description

The communication model is connected with the communication agent model in the C2 simulation framework through an interface. Messages between the agent and the C2 model are represented as in Table 2. There are three types of messages: detection info_C2, fire command_C2, and status_C2. Detection info is used to send and receive tracing information between the pair of C2s. Fire command includes information about firing weapons and a status message is used to transmit the situation of C2. In this table, the important thing is that the messages' sending or receiving time and Sender C2_ID and Receiver C2_ID, and these contents are used for reflecting the effects of communication. For instance, we assume a situation in which C2 (A, id=1) sends messages to another C2 (B, id=2) when the current time is 0 sec. First, C2(A) sends the message with Sender C2_ID=1, Receiver C2_ID=2, and Sending time=0 to the agent model, and then the agent model sends the message to the communication model. In the communication model, delay time will be calculated and a response message is sent to the agent model. At this moment, the agent model has a message with C2_ID=1, Receiver C2_ID=2, and Receiving time=0+delay time that is from the communication model. Then, this agent model will send the message to its destination on the basis of Receiver C2_ID=2. In this case, C2 will receive the message because the ID of that is 2. In this way, communication effects are reflected.

Table 2: Messages between agent and communication model.

(a) Message from Agent to Communication Model

| Message Type | Contents |
|---|---|
| Detection info_C2 | Enemy Info., Sender C2_ID, Receiver_C2_ID, Sending Time |
| Fire command_C2 | Weapon ID, Target Position Sender C2_ID, Receiver C2_ID Sending Time |
| Status_C2 | C2_ID, C2_Position,C2_Status |

(b) Message from Agent to Communication Model

| Message Type | Contents |
|---|---|
| Detection info_C2 | Enemy Info., Sender C2_ID, Receiver C2_ID, Receiving Time |
| Fire command_C2 | Weapon ID, Target Position Sender C2_ID, Receiver C2_ID Receiving Time |
| Status_C2 | C2_ID, C2_Position, C2_Status |

## 4 CASE STUDY

In this section, with the proposed C3 simulation framework, we will conduct three experiments about analysis on communication effects and C2 structure. Through the first and second experiments, we will evaluate the accuracy and performance of the discrete event metamodel. In the last experiment, we will apply the new C2 structure to the C3 simulation framework. For the experiment, we plot the defense interoperation (J. S. Dahaman et al. 1988; IEEE Computer Society 2000) simulation of an infantry corps level abstracted by using a self-developed wargame simulator and the NetSPIN communication simulator developed by the Agency for Defense Development (ADD) in South Korea based on OPNET as the communication model. This NetSPIN describes a spider net (ADD. 1998), which is a real-world military

communication environment. The entire scenario of the simulation is as follows. When an enemy emerges, a detection troop detects it and sends its information to the corps command center, or the top level in the structure. Then, the C2 allocates weapons to use for attacking and transmits a fire request to another C2 including available weapons, and this low-level C2 attacks the target with the requested machinery. During the transmission between the pair of C2s, communication effects are calculated and reflected. As a simulation environment, in the case of the wargame simulator, CPU: I7-3770 3.4GHz, RAM: 12Gb, DEVSim++ v.3.1, and KHLAAdaptor1516 are used. The DEVSim++ is a toolset for modeling and simulation in a discrete event system and KHLAAdaptor is also a tool set for interoperation (T. G. Kim et al. 2011). The environment of the communication simulator is CPU: I7-920 2.67GHz, RAM: 12Gb, and NetSPIN v.0.1 based on OPNET (OPNET Technologies, Inc. 1986) v.17.1. These are interoperated by MAK RTI v.4.1.1.

### 4.1 Experiment 1: Identification Communication Parameter

The objective of this experiment was to discriminate parameters affecting the result of the wargame simulator among diverse parameters, and the parameters were used to construct the metamodel. In order to do that, we measured the survivability of an enemy by changing the parameters. Communication equipment and parameters for input were as below in Table 3. A description of each parameter and equipment is in the table. We chose 64 experiment points based on a 2-level full factorial against 6 parameters and ran repeat simulations 30 times per one experiment point. An effective index was set up as survivability at simulation time = 1300 sec, because this time is when the survivability was saturated.

Table 3: Communication parameters and descriptions.

| | Equipment Name / Description | Value |
|---|---|---|
| TDU (IP Router Model) | Data forwarding rate / Forwarding speed of IP routers | 5,000 bit/s, 500,000 packets/s |
| | Router memory size / Memory size to process packets | 8 MB, 256 MB |
| TMR (Wireless Transmission Model) | Data speed / Speed of data in wireless | 256 Kbps, 8192 Kbps |
| | Processing delay time / Delay time to processing packets | 0 Sec, 3 Sec |
| | Tx power / Power to transmit tx channel | 0.32 W, 15.8 W |
| | Valid rx power / Threshold to receive in rx channel | 78 dBm, 90 dBm |

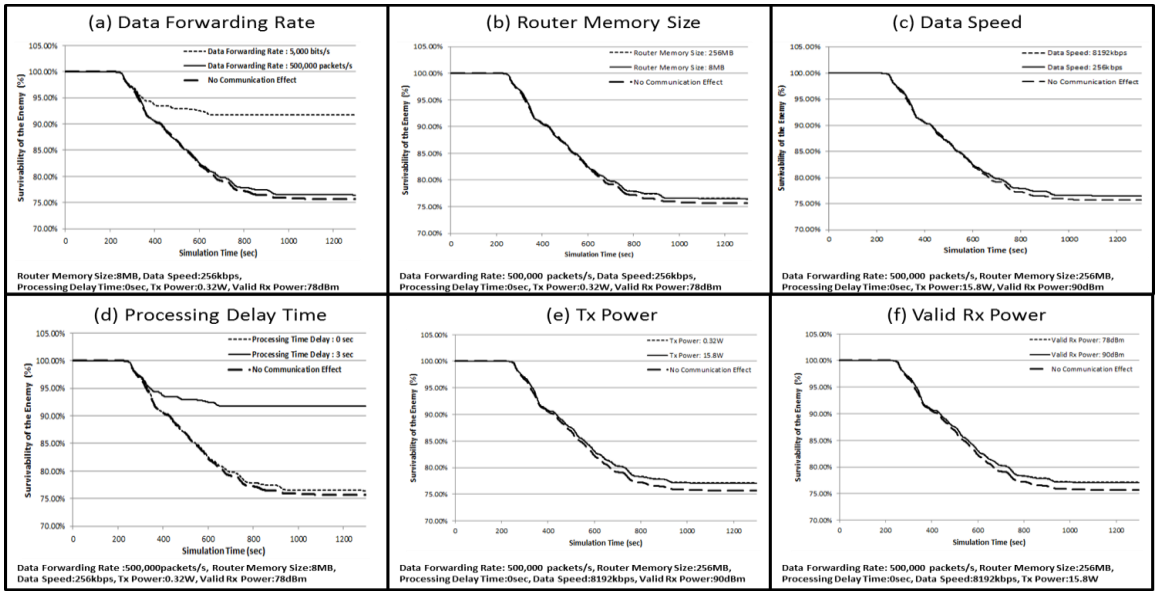


Figure 11: Simulation results for changing communication parameters.

Figure 11 shows that, of the six parameters, only data forwarding rate and processing delay time affected the effective index. Figure 11(a) shows data for three cases. One was 5,000 bits/s, another was 500,000 packets, and the last did not consider communication effects—that is, there was no delay and always success. In this case, the enemy recorded low survivability because the condition of communication in the friendly forces was perfect. When comparing 5,000 bit/s with 500,000 packets/s, the latter had a lower record because communication traffic was more in that case, and, as a result, the enemy had more damage than in the previous case. In the processing delay time case, when applying 3 sec processing delay time, higher survivability was achieved due to the extended delay time. By contrast, in the other cases, router memory size had no impact because the used memory size was very small. Data speed also had no effects since delay time varied within several milliseconds. In the case of tx power and valid rx power, the results did not change because the communication model used in this experiment is a fixed network. In summary, data forwarding rate and processing delay time were the influential parameters.

### 4.2 Experiment 2: Discrete Event Metamodel

Generally, the procedure for making a metamodel consists of four stages: design of the experiment, data collection, metamodel construction, and model validation. In order to make two metamodels with each success rate and delay time as an output due to having an impact on combat results, we conducted an experiment according to these procedures. In the first stage, we extracted 31 experiment points with a Face-centered Central Composite and 9 points with a Latin Hypercube against influential parameter acquired in experiment 1, or data forwarding rate and processing delay time. Then we substituted these points to the communication simulator and conducted the simulation 30 times per one experiment point. On the basis of the collected data, we constructed a second-order response surface metamodel (RSM) against success rate as shown in Table 4, and the adjusted r-square error was measured as 0.779. On the other hand, in the case of the metamodel against process delay time, the parameter was interpreted as no impact on the result, so we made a probability model with normal distribution where the mean = 0.081sec and standard deviation = 0.0037sec to substitute RSM. By using the RSM against the success rate and the probability model against delay time, we made the discrete event metamodel. To validate this metamodel, we again extracted 10 experiment points aside from the existing 40 points and used these new points as input for the communication simulator and discrete event metamodel, as shown in Figure 5. We assessed the accuracy and performance of the simulator and metamodel, and the results are shown in Figure 12. For the analysis of accuracy, we calculated the root mean square error (RMSE) of enemy survivability, which was measured as 0.0031. This means that when we use this metamodel, we can predict the result within 0.0031 approximate error. Also, for the performance, we measured execution time. In the case of the discrete event metamodel, 150.1 sec was recorded, in comparison to 15197.4 sec for the simulator case. As can be seen in Figure 12, these two models have a big gap against the x-axis, or simulation execution time. In other words, the discrete event metamodel has 100 times performance, and through this, we can simulate diverse scenarios within a short time by tolerating small errors.

Table 4: Metamodel against success rate

| | Unstandardized Coefficients(B) | p-value |
|---|---|---|
| Constant | 0.462 | 0.000 |
| Data Forwarding Rate | 0.014 | 0.011 |
| Processing Delay Time | -0.423 | 0.000 |
| Data Forwarding Rate$^2$ | -9.302E-5 | 0.060 |
| Processing Delay Time$^2$ | 0.120 | 0.000 |
| Data Forwarding Rate × Processing Delay Time | -0.002 | 0.003 |

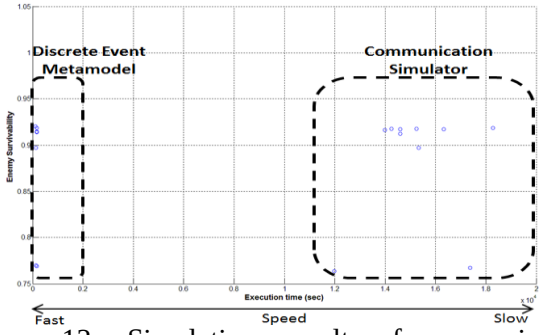


Figure 12: Simulation result of comparing simulator and discrete event metamodel.

### 4.3 Experiment 3 : C2 Structure

The purpose of the final experiment was to make a comparison between the new and existing C2 structure in terms of the effectiveness of combat. In the previous structure, orders are delivered step by step from lowest to uppermost C2. So, this structure has a significant delay time due to many intermediate stages. In order to eliminate this flaw, a direct connection structure was proposed, and we simulated the new structure by comparing it to the existing one. The result of the experiment is shown below. The required time from detection to attack was shorter by about 15 sec in the new structure, which would allow the first attack to occur quicker. The reason is that the smaller number of steps in the new structure causes a fast response. This difference of response time leads to a 60sec gap based on time when enemy survival rate is saturated. Through this time gap, the chance to be damaged declines, and forces can consequently gain a competitive advantage on the battlefield. In short, with the proposed C3 simulation framework, we can simulate the new C2 structure without increasing economic or manpower costs.

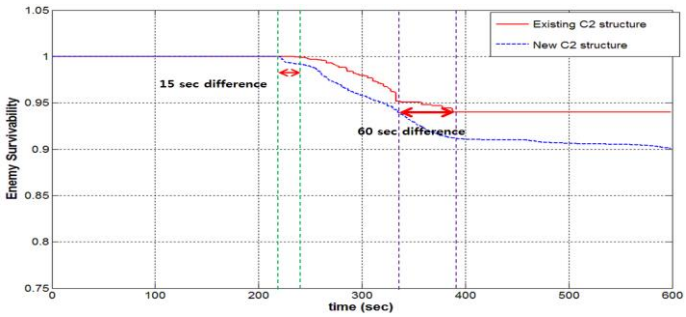


Figure 13: Simulation result for changing C2 structure.

## 5 CONCLUSION

The result of combat in the NCW battlefield depends on how fast the Sensor–C2–Shooter cycle rotates. Therefore, it is important to enhance C2 and communication, which control the velocity of the cycle. In short, in order to describe the NCW battleground, we need modeling and simulation about the effects of communication and C2 structure. In the case of communication, a description of diverse effects is needed. In the C2 structure, various structures of order transmissions should be reflected because the situation of war can vary. In spite of the need for C3 simulation, or the variety of C2 and communication models, existing research has not developed a model that allows them to be analyzed effectively. To solve this problem, this paper proposed a C3 simulation framework that comprises C2 and C models. In the case of the C2 framework, we enhanced modularity by separating C2 from the command center model in the existing structure; as a result, we can change the C2 structure without revising the other models. In the case of the communication model, the previous form used a communication simulator as the communication model. However, this requires a long execution time. To overcome this problem, we constructed a new metamodel. The existing metamodels, which can only describe a static system, cannot represent the communication simulator, which is a dynamic system. So, we suggested a new metamodel called the discrete event metamodel. With this model, we can express the characteristics of the simulator, such as delay and success rate of transmission of orders between two C2s. So as to show the implementation of the C3 simulation framework, we conducted three case studies. The discrete event metamodel provided improved performance in communication in respect to time within small approximate error, and, with regard to C2, the new structure can be simulated without modifying other models. In future work, we will construct the discrete event metamodel against more diverse parameters and equipment of communication, since this paper is limited to two kinds of equipment. Inversely, if constructing the metamodel against a wargame simulator, it can be used to analyze node traffic. Also, it would be valuable to apply this metamodel to other domains.

## ACKNOWLEDGMENTS

This work was partially supported by Defense Acquisition Program Administration and Agency for Defense Development under the contract. (UD110006MD)

## REFERENCES

Agency for Defense Development, Inc. 1998. *Spider Network.* http://www.add.re.kr. [Accessed November 19, 2012].

Benjamin, P. 2008. "CES Integration with OneSAF for Mission Level Simulation." In Proceedings of the *2008 Fall Simulation Interoperability Workshop,* 271-276, Florida.

C4ISR Architecture Working Group. 1997. *C4ISR Architecture Framework Version 2.0*. Washington DC: Department of Defense. http://www.fas.org/irp/program/core/fw.pdf. [Accessed February 17, 2013]

Dahmann, J. S., R. M. Fujimoto, and R. M. Weatherly. 1998. "The DoD High Level Architecture: An Update." In Proceedings of the *1998 Winter Simulation Conference,* edited by D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, 797-804. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

IEEE Computer Society. 2000. IEEE Standard for Modeling and Simulation High Level Architecture: Framework and Rules. *IEEE Std. 1516-2000*: 1-22.

Kleijnen, J. P. C. and R. G. Sargent. 2000. "A Methodology for Fitting and Validating Metamodels in Simulation." *European Journal of Operational Research 120 (1)*: 14-29.

Walsh, J., J. Roberts, and W. Thompson. 2005. "Ncw End-To-End Model for Future C2 Architecture Assessment." In Proceedings of the *2005 International Command and Control Research and Technology Symposium,* Virginia.

Kim, D. S., J. W. Bae, and T. G. Kim. 2011. "Development of DEVS based War Game Simulator considering Interoperation with Communication Effects Simulator." In Proceedings of the *2011 Korea Institute of Military Science and Technology Conference.*

Kim, T. G., C. H. Sung, S. Y. Hong, J. H. Hong, C. B. Choi, J. H. Kim, K. M. Seo, and J. W. Bae. 2011. "DEVSim++ Toolset for Defense Modeling and Simulation and Interoperation." *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology 8 (3)*: 129-142.

Manfred, E. 2001. "The German C3-Model FIT." In Proceedings of the *2001 Fall Simulation Interoperability Workshop,* Florida.

OPNET Technologies, Inc. 1986. *Opnet Manual.* http://www.opnet.com. [Accessed December 8, 2012]

Ram, A. N. 2012. "Front Line Emergency, Directly Report to Chairman of the Joint Chiefs of Staff." *The Hankook Ilbo.* http://news.hankooki.com/lpage/society/201211/h2012112702355021950.htm. [Accessed February 10, 2013]

Zeigler, B. P., P. Herbert, and T. G. Kim. 2000. *Theory of Modeling and Simulation.* 2nd ed. New York: Academic Press.

## AUTHOR BIOGRAPHIES

**BONG GU KANG** is a Ph.D candidate in the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST). His research interests include discrete event system modeling in the domain of the defense field. His email address is bgkang@smslab.kaist.ac.kr.

**TAG GON KIM** is a Professor in the Department of Electrical Engineering, KAIST. He was the Editor-in-Chief for *Simulation: Transactions of the Society for Computer Modeling and Simulation International* (SCS). He is a co-author of the textbook *Theory of Modeling and Simulation*. He has published about 200 papers in M&S theory and practice in international journals and conference proceedings. He is very active in defense modeling and simulation in Korea. His e-mail address is tkim@ee.kaist.ac.kr.