Teaching Modeling to Engineers in an Undergraduate Simulation Course

Vikram Mittal, Robert Kewley, Brett Lindberg
Department of Systems Engineering, United States Military Academy
West Point, NY
vikram.mittal@usma.edu, robert.kewley@usma.edu, brett.lindberg@usma.edu

ABSTRACT

A significant challenge in teaching simulation to undergraduate students is to find a way to allow them to model a real world referent system within time and student skill constraints. Several research sources highlight not only the important challenge of model development (Garcia and Ceneno, 2009, Tako, 2011) but also the increased need for model development instruction among engineers (Grasas et. al., 2013, Saltzman and Roeder, 2013). One approach to this challenge is to use a general purpose discrete event simulation software package within the course, but this presents two challenges. Teaching the package to the students takes significant time, and the package introduces limitations which may restrict their ability to model certain real-world referents, particularly in the engineering domain. A conceptual approach to solving this problem is to use a model development paradigm that abstracts away the interface to the simulation infrastructure while still allowing the students to use the full expressive nature of a programming language. Two undergraduate courses at the United States Military Academy employed this strategy via the Discrete Events Specification System – Distributed Modeling Framework (DEVS-DMF) (Kewley et. al, 2016). The DEVS abstraction allowed students to think about their model as a simple state change function with defined inputs and outputs, and DMF allowed them to program in a cloud-based Jupyter Notebook using the Python language. Students in a combat modeling course employed a variety of models to understand drone jamming, and students in an engineering capstone project employed models to account for human factors in rifle marksmanship. The effectiveness of this approach was assessed through student grades, exit-interviews, and course-end surveys. These assessments showed an increased understanding of the model development process, and students also reported greater ownership of their models. However, this experiment also highlighted some weaknesses in their understanding of underlying methodologies and programming skills.

ABOUT THE AUTHORS

Dr. Vikram Mittal is an Assistant Professor in the West Point Department of Systems Engineering. He holds a BS from Caltech, an MS from Oxford, and a PhD from MIT in Mechanical Engineering. Prior to West Point, he worked at Draper Laboratory. He has broad M&S research and teaching experience ranging from computational fluid dynamics, solid mechanics, chemical kinetics, and combat modeling. He currently works as part of the Models Based Systems Engineering team for the TALOS program (SOCOM). Additionally, he is building a simulation of the mobilization process for the Army G3/5/7. He also works on developing human factor models for the Soldier System Enterprise Architecture program, which are integrated through a DEVS-DMF. In his teaching capacity, he is the course director for the Combat Modeling course at West Point.

COL Rob Kewley, Professor of Systems Engineering, is currently the head of the West Point Department of Systems Engineering. He holds an MS in Industrial Engineering and a PhD in Decision Science and Engineering Systems from RPI. His operational background includes service as an Armor officer and operations research analyst. He has over 15 years of experience in teaching M&S to undergraduate students. His research background is in the development of simulation methodologies and engineering processes to analyze complex problems. The Army Research Lab supported his development of DEVS-DMF to enable modular and cloud-based simulation models. He currently serves as the co-chair of the SISO Cloud Based Modeling and Simulation Study Group. He also was lead author for NATO's M&S as a Service Engineering Process.

LTC Brett Lindberg, M&S Research Scientist, is currently the head of CEMA Synthetic Environments research at the Army Cyber Institute. He holds a MS in Modeling, Virtual Environments and Simulation from the Naval Postgraduate School. His operational background includes service as an Armor officer, from which he transitioned to Functional Area 57 (Simulation Operations). He has acquired over 10 years of M&S experience, including a deployment to Afghanistan and 6 Networked Integration Evaluations. His research background is in tactical applications of Augmented Reality technology, and he has also served as the chair of the Army M&S Office's Cyber and Electronic Warfare M&S Working Group.

Teaching Modeling to Engineers in an Undergraduate Simulation Course

Vikram Mittal, Robert Kewley, Brett Lindberg
Department of Systems Engineering, United States Military Academy
West Point, NY
vikram.mittal@usma.edu, robert.kewley@usma.edu, brett.lindberg@usma.edu

INTRODUCTION

Modeling and Simulation (M&S) is a critical methodology course taught in numerous systems engineering and operations research programs, as well as most engineering disciplines. Though somewhat domain specific, these courses are intended to teach students how to apply M&S to engineering design, as well as the underlying methodologies and good modeling practices. Though a multitude of different software platforms are available for teaching M&S, the courses are primarily taught with academic versions of commercial software packages. These software packages usually already have the underlying methodologies integrated into the code. Therefore, upon learning to use the software, students can run simulations and analyze the outputs. However, additional benefits can be realized by having students code the underlying methodologies. Not only does it reinforce the underlying principles and mathematics, but it also allows the course to not be limited to the constraints of the software.

A Discrete Event Specification Distributed Modeling Framework (DEVS-DMF) is an M&S framework that allows for the integration of multiple stateless models into a discrete event simulation. This framework is a useful tool for teaching M&S. In particular, it avoids the constraints of typical academic software, while allowing students to develop methodologies, build models, work as a team, and builds upon their previous mathematical and coding skills. This paper provides guidance on how to build the framework for teaching purposes, the advantages of teaching M&S through the use of a DEVS-DMF, and two case studies where this technique was implemented.

SKILLS NECESSARY TO TEACH IN INTRODUCTORY SIMULATION COURSES

A survey of relevant textbooks, syllabi, and course websites found that most M&S courses following a structure similar to what is shown in Figure 1. The overall goal of these courses is to introduce M&S concepts, reinforce learning from previous courses, and teach students to appreciate the use of M&S in real-world engineering design; additionally, most courses teach its students at least one M&S software package (Hoad and Kunc, 2018).

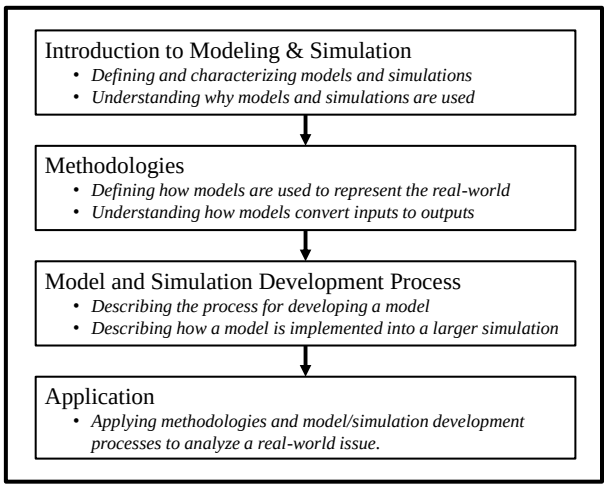

Figure 1: Framework for standard Modeling and Simulation Courses

The courses start by introducing the students to M&S, including definition of key terminology and uses of models. This foundational knowledge is inherently necessary for the students to be able to understand the role of M&S in engineering design (Feurzeig and Roberts, 1999).

Courses then tend to teach the underlying methodologies that drive the simulation, focusing on both the high level structure of the model and the underlying mathematics (Stern, et. al, 2006). These methodologies can range from the Navier-Stokes equations for a fluid dynamic modeling course to Monte Carlo processes for a stochastic modeling course. Additionally, this block of classes teaches the statistical methods needed for analyzing the model outputs. These methodologies are fairly complex; as such, the M&S course must rely heavily on prerequisite courses to teach the necessary material. The M&S course must meanwhile focus on how to implement these methodologies in a modeling environment (Kress, et. al, 2010).

Once the students understand the underlying methodologies, they can then learn how to build models. This process is critical for any M&S course (Garcia and Ceneno, 2009, Tako, 2011). In an ideal case, model development would follow a traditional systems engineering design process. This process would require the students to understand the methodologies, determine the model inputs and outputs, build a conceptual design, build a preliminary design, build the model, and verify and validate the model. However, many classes take a more direct approach where students are given an assignment and they struggle through the software until their model works (Grasas et. al., 2013). Once developed, the model can then be used for output analysis.

Most courses conclude with an application-based project that is tied to real-world issues. These issues must be picked such that they can readily be modeled in the course software while still being relevant. These projects allow students to apply their knowledge of methodologies and model development processes to actually build a model. The output from the model is then analyzed to develop conclusions. The overall goal of these projects is to allow students to understand the model development process while appreciating the use of M&S for solving real world problems (Saltzman and Roeder, 2013).

OVERVIEW OF DISCRETE EVENT SPECIFICATION DISTRIBUTED MODELING FRAMEWORK

The DEVS-DMF (Kewley, Kester, and McDonnell, 2016) is an implementation of DEVS for integrating simulation models in parallel and distributed architectures typically found in cloud environments. It is based on the Parallel DEVS framework of (Chow and Ziegler, 1994), but enables asynchronous execution of distributed models via the actor model of computation (Agha, 1985). Designed for interoperability of models and flexibility of deployment, the DEVS-DMF framework simplifies simulation development in two ways. First, it encapsulates internal state and synchronizes simulation execution to free the developer to focus on development of state transitions and their associate messages. An additional advantage of this framework is location transparency. Each DEVS actor may be deployed locally in the same Java Virtual Machine or elsewhere in the cloud on another machine.

HOW TO BUILD A DEVS-DMF MODEL TO ALIGN WITH TEACHING REQUIREMENTS

The key element of modularity in DEVS-DMF is the state transition function. The stateless nature of the state transition function allows it to take an external message or the current state of the model as an input, producing a new state and possibly an output message. This feature allows one to isolate the state transition function from the state management and time management infrastructure of the model. These parameters can simply be passed into the function. Therefore, instructors can assign students to the development of a specific state transition function in the model. These functions can then be integrated into a larger, pre-established model, potentially built by the instructor for the purpose of the class. Alternatively, the larger model can have been previously developed as part of a larger modeling effort. Regardless, it is necessary that the model architecture be properly developed and the appropriate model infrastructure be built prior to the students commencing work.

It is critical that these state transition functions be defined clearly enough for students to understand the inputs, outputs, and mathematical processes associated with it. In particular, students must be able to clearly identify the model's properties, values which are simulation inputs that change from run to run, but not during the execution of an iteration. Properties can be received once as input, but state variables must be taken in with each execution of the function. One example would be determining the power received by a radio receiver given a power output from a transmitter a certain distance away. In this case, the student would initialize the model at runtime with the properties of the radio receiver. The power of the transmitter, the frequency transmission, and the distance between the radios would be inputs to the transition function. The student could then do research and build a state transition function based on a free space loss propagation model to determine whether the transmitted message is received by the receiving radio.

BENEFITS OF USING DEVS-DMF IN TEACHING SIMULATION

Methodologies

As discussed in the previous section, most M&S courses are centered on teaching students the underlying methodologies that drive simulation packages. However, the students then run models on software platforms with pre-programmed methodologies that often do not allow students to alter the methodologies.

The use of software with pre-programmed methodologies poses two problems. First, it forces the instructor to focus the course material on those methodologies used by that software package. However, there might be other methodologies that are relevant to the field of study. Second, the students only receive a surface-level understanding of the methodologies, since they are not required to actually build these methodologies into the model. For example, many courses teach the mathematics behind random number generation without ever having students build a realistic random number generator, since that capacity is pre-programmed in the M&S software package.

The use of DEVS-DMF avoids these issue by allowing instructors to select the relevant methodologies and requiring the students to actually build them into the model. Since many of these methodologies are somewhat complex, the distributed nature of DEVS-DMF further allows for a large model to be decomposed into smaller modules that can be spread out amongst the instructors and students.

Model Design Practices

Many M&S courses teach modeling by introducing the students to the software and then have students build models in that software. This strategy often follows a "brute force" approach where students will learn software through building and troubleshooting their model.

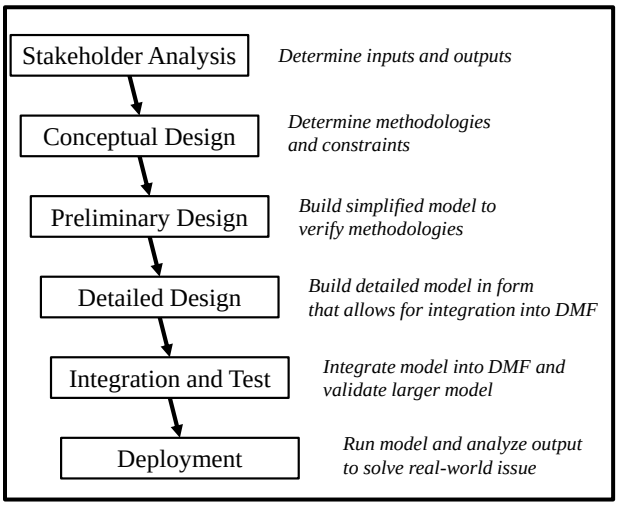

Figure 2: Model Design Process

However, the usage of DEVS-DMF allows students to follow a standard Systems Engineering design process, as shown in Figure 2. It starts with the identification of the stakeholder needs which are related to the inputs and the outputs to the models. The second step involves a conceptual design which identifies the methodologies necessary for calculating the outputs. Often the conceptual design is illustrated as an IDEF-0 diagram. After the conceptual design is determined, the students can create a preliminary model. This model can take the form of pseudocode or an Excel-based spreadsheet model. The preliminary model undergoes a certain amount of verification to ensure that the methodologies accurately calculate the outputs relative to the inputs. After the preliminary model is verified, a detailed model can be constructed. The design model is then coded in a programming language such as Python or Java. The detailed model must also be designed to integrate with other models that have been developed. For example, the outputs from one model should be in the units required for the inputs for another model. The detailed models can then be integrated and the larger model can undergo verification and validation. By having students implement the full model design process, they will be better equipped for developing models throughout their professional careers across a wide spectrum of domains.

Flexibility to Adapt Simulation to Real World Challenges

Academic M&S software are set up to follow set tutorials or address a constrained problem set. This often arises from the necessity that the academic software must have less capability than its commercial variants, lest engineers buy the academic version for commercial purposes. These constraints result in a limited scope of real-world issues that the model can address. In turn, instructors must constrain their course projects to the limits of the software, and students do not get a full understanding as to the power of M&S in analyzing real-world issues.

The use of a DEVS-DMF model provides endless possibilities for the real-world issues that can be analyzed. Although the instructor must scope the project to ensure that it is feasible, they can set up the framework to be relevant to almost any real-world issue. By doing so, students are able to learn about real-world challenges and develop an appreciation for the use of M&S in solving those challenges.

Avoids "Fighting the Software"

DEVS-DMF models can be built in standard computer coding software, such as Python or Java. Most engineering courses require that their students complete an introductory course that teaches them a language. This programming knowledge can be leveraged when teaching M&S in a DEVS-DMF. By doing so, students reinforce their programming knowledge and get the opportunity to apply their programming skills to real world challenges.

Additionally, a large portion of M&S courses currently involve teaching the software. Since the software packages are often an academic variant of a commercial M&S package, the user interfaces are often tailored for specific commercial application. As such, the software can end up not being user-friendly or conducive to learning, resulting in a significant portion of class time being spent learning how to use the software, rather than how to build and implement models.

In other situations, the software packages are tailored towards teaching. These software packages are tailored to very specific applications, requiring that only those methodologies be taught. Additionally, these software packages will often have undergone limited test and evaluation, resulting in glitches that are frustrating for students.

CASE STUDY 1: SYSTEMS ENGINEERING 485 – COMBAT MODELING

Overview of SE485

Systems Engineering 485 (SE485): Combat Modeling is an M&S course at the United States Military Academy (USMA). The course introduces senior-level undergraduates to the theoretical and practical issues in combat modeling and simulation. It covers a range of fundamental M&S topics to include: the role of random numbers; probabilistic underpinnings of simulation; verification, validation, and accreditation; methods of designing and conducting simulation experiments; modeling techniques; and analysis of results. The students learn and apply algorithms specific to combat M&S, such as target detection, shot delivery accuracy, and casualty assessment.

Traditionally, the course has utilized the Infantry Warrior Simulations (IWARS) computer simulation environment; IWARS is an entity-based multi-sided simulation program that focuses on small-unit military operations. The underlying methodologies are powered by a detailed database that captures the associated parameters. IWARS was found to have an intuitive user-interface, such that after 10-15 hours of lab exercises, students became proficient in it.

IWARS was a useful software package for developing a limited set of combat simulations and to generate data for output analysis. However, since the methodologies are built into the software, students had limited visibility into them outside of being able to change the parameters in the database. Additionally, IWARS was limited to small units, such as infantry squads and platoons, performing doctrinal infantry missions, such as a movement to contact. New methodologies cannot be added into the software to account for new tactics or enemy threats. Moreover, the software has a number of glitches and requires substantial troubleshooting to get models to run robustly. As such, the course deviated from IWARS in Academic Year 2018 to a DEVS-DMF based course project to provide students the ability to model emerging threats.

Overview of SE485 Project

The DEVS-DMF paradigm was implemented as part of a SE485 course project in Academic Year 2018. Students worked in groups of 3 to build models relevant to the neutralization of enemy remote controlled (RC) aircraft. In particular, insurgents in Iraq had started using commercial RC aircraft to drop grenades on Coalition forces. Multiple research groups in the US Army were looking at methods of handling this problem; however, no solution existed aside from trying to shoot down the RC aircraft. As such, modeling this problem provided students with the opportunity to tackle a relevant project faced by the US Army.

Each group was tasked with building 1 of the following 6 models:

- audible detection of RC aircraft based on size of aircraft, distance to soldier, and ambient noise levels;
- effectiveness of a direction finding radio system in determining the azimuth and elevation of aircraft based on the strength of the radio signal and location of the drone;
- location and error of the RC aircraft position displayed on a Helmet Mounted Display (HMD);

- effectiveness of electronic warfare equipment in jamming the signals to/from the drone based on the power of the jammer, transceiver properties of the drone, and the location of the drone;
- effectiveness of a radio in picking up a cyber-hacking signal by a receiver radio based on characteristics of the receiver radio, the signal strength of the emitted signal, and environmental conditions;
- point of impact of a grenade dropped from the RC aircraft at a target soldier based on the munition properties, location of drone, and environmental conditions.

These models were then integrated together as shown in Figure 3.

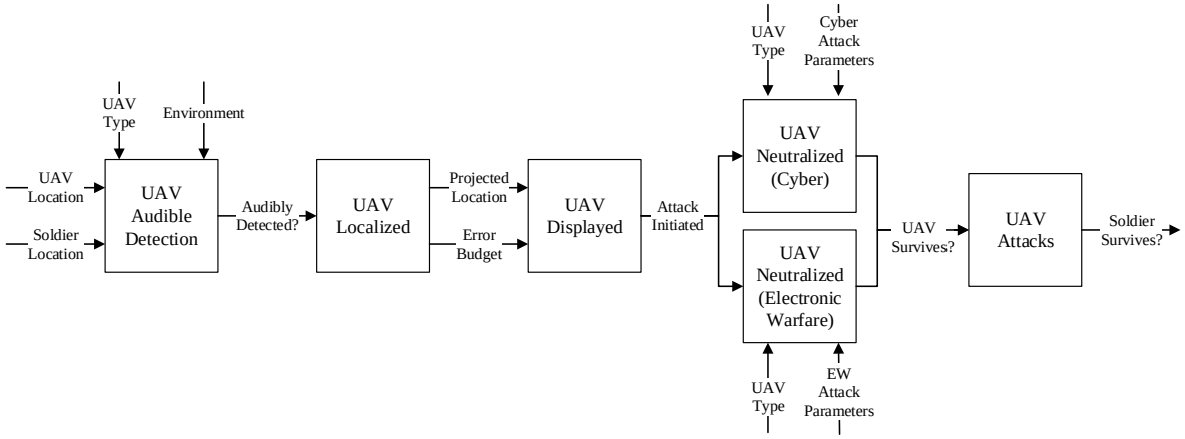

Figure 3: Integration of the 6 stateless models developed as SE485 on neutralization of commercial drones

The students had to first perform a literature review and find relevant studies to develop methodologies for their respective models. They then built mathematical models in Excel, including the development of a random number generator. After the Excel model had been debugged, the code was converted over to Python. The students coded the model in a cloud-based Jupyter notebook that allowed for it be implemented in a DEVS-DMF. A deliverable was required for each stage of development. Though this project had a substantial amount of out-of-class work, 5 class periods were dedicated as project workshops to give the students the opportunity to receive help on each phase of the project. The entire time frame for the project was approximately 3 weeks.

Implementation and Assessment

After the first day of the assignment, the instructors concluded that the students did not have the confidence or experience necessary to perform an adequate literature review. Therefore, the instructors provided sources to the students to help them better develop their model methodologies. Most of the students were able to develop the methodology and build the subsequent Excel model.

However, students encountered significant difficulty in coding the model in Python on the provided Jupyter Notebook. Although the students were taught to code in Python in an introductory computer science course, the course itself only covered the basics of the language. Additionally, the majority of the students had taken that course in their first year and had forgotten a substantial amount of the material. Therefore, many of the groups submitted codes in other languages, pseudocode, or codes that did not compile. Even when given extra time and help on the code, many students opted to take a lower grade than finish their models. Only three of the twelve groups submitted models that could be integrated into the DEVS-DMF. These models were

Table 1: Grades for SE485 Project

Project Section	Average Grade	Standard Deviation
Methodology Development (Write-Up)	91%	8%
Basic Model Development (Excel)	94%	10%
Model Integration (Python)	81%	14%
Output Analysis (Write-Up)	59%	28%

integrated to show the effects of jamming on a UAS as it moves towards a squad of soldiers.

Table 1 displays the grade data for the project which reflect the trends previously identified. The grades show that students understood and completed the methodology development and building the model in Excel. However, the students received low scores on the final model development in Python.

An informal exit interview was conducted with students following the end of the project. The students reported the following sustains for the project:

- the underlying problem was interesting in that it modeled a real-world issue;
- the project allowed students to go through the full model development process;
- the project reflected real-world problem solving techniques;
- the Excel model development process was straightforward and allowed students to apply methodologies learned in class;
- the bulk of the project was done during class time with instructor supervision.

The students also reported the following improves:

- the Python portion of the code was overly difficult and relied on material not covered in the course;
- the assignment did not leverage IWARS or Virtual Battle Space 3, which a large portion of the course had previously covered;
- several of the model topics required a deep understanding of electromagnetism; students did not have enough time to fully grasp these concepts to understand the methodologies.

Additionally, course end surveys were used to determine how well the project met its objectives. The overall course objectives that the students were intended to learn from the project were:

- design combat models in a computerized simulation environment;
- build models based on analyzing data collected through experimentation;
- describe the state of technology for modeling and simulation for the Army;
- solve real-world problems through quantitative analysis.

In the Course-End survey, students were asked to rank between 1 and 5 as to how well they would be able to complete these objectives (5 = very well, 1 = poorly).

The results are shown in Table 2. Though the project has significant room for improvement, it did meet these objectives. The course-end survey results were compared to previous years. Though the average assessment values only had a statistically insignificant change from past course iterations, the spread of results were larger, with more students giving low scores. This change is likely due to the frustration that some students felt with their final deliverable.

Table 2: Course End Survey Results

Objective	Responses (Percentage)					Average
	5	4	3	2	1	
Design combat models in a computerized simulation environment	34%	52%	14%	0%	0%	4.2
Build models based on analyzing data collected through experimentation	31%	59%	10%	0%	0%	4.2
Describe the state of technology for modeling and simulation for the Army	21%	55%	17%	3%	3%	3.9
Solve real-world problems through quantitative analysis	24%	52%	21%	0%	3%	3.9

CASE STUDY 2: SYSTEMS ENGINEERING 403: SENIOR CAPSTONE

Overview of SE403

Systems Engineering 403 (SE403) is a year-long senior level Capstone course, where students work in groups of four to apply their knowledge to solving a real-world challenge. The goal of the Capstone course is to reinforce their foundational knowledge and bridge the gap between academia and the real-world. These real-world challenges are brought to USMA from primarily military clients. These challenges can range from developing test plans, performing value modeling to assess solutions, and model development.

Overview of Capstone Project

In Academic Years 2018, the Natick Soldier Research, Development, and Engineering Center (NSRDEC) sponsored multiple Capstone groups to support model development for the Soldier System Enterprise Architecture. The project scope involved looking at the existing weapon model space and account for human factors in capturing soldier lethality and survivability. The existing weapon model used for this Capstone project was the Weapons Lethality Service (WLS) which was built on a DEVS-DMF in the Scala language. The WLS divides the shooting process into target acquisition, delivery accuracy, and casualty assessment. The target acquisition models are based on sensor detection algorithms, the delivery accuracy model uses weapon data, and casualty assessment uses biomedical data. The model architecture has federated models for each of these processes that are interconnected to capture the full shooting event.

The Capstone team was tasked with building additional models into the WLS to create a more holistic model to capture soldier performance metrics. In particular, they wanted to look at the effects of training and weapon design on the overall lethality and survivability of the soldier. The project was left open for the students to determine which models were necessary to build and incorporate into the WLS.

Implementation and Assessment

The Capstone team performed a failure mode analysis on the shooting process and found several models that should be incorporated into the WLS. These models included weapon instability associated with added weight, weapon jamming, indecision in target identification, shooter heartbeat, and shooter sleep levels. For a semester, the team conducted the requisite research to develop the methodologies for each of these models. These methodologies were framed as IDEF-0 diagrams, which show each models inputs, outputs, constraints, and mechanisms. These IDEF-0 diagrams were then combined with the WLS to determine how to integrate the models, as shown in Figure 4. They then built and tested each model independently in Microsoft Excel.

The following semester, the team familiarized themselves with the WLS. The Capstone team was given access to the existing WLS Scala code, which allowed them to understand the coding language such that they could build their models. They then built each aforementioned model into the WLS. The WLS was then used to explore the trade-space and better understand weapon lethality.

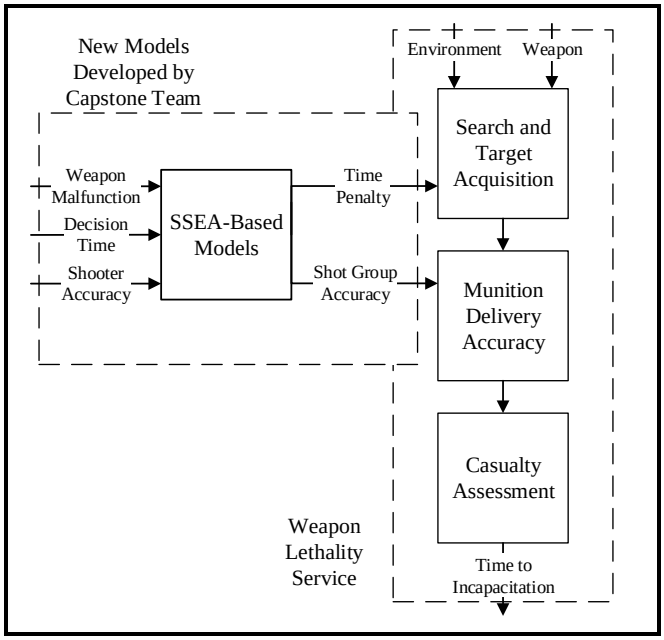

Figure 4: Integration of Weapon Lethality Service models and the models developed by the Capstone Team

Table 3 shows the grade distribution for this project. The group struggled initially with the literature review and the methodology development. In particular, the initial methodologies were overly complex to the point that it would be difficult to build a model. The team then ended up over-simplifying the methodologies when building the Excel-based mathematical models. After these models were corrected, the team had no issues integrating the models into the WLS or performing output analysis.

Table 3: Grades for SE403 Project

Project Section	Average Grade	Standard Deviation
Methodology Development (IDEF-0)	89%	5%
Basic Model Development (Excel)	92%	6%
Model Integration (Scala)	100%	-
Output Analysis (Technical Paper)	100%	-

Informal exit reviews with the students backed the above assessments. The students struggled initially with scope and methodology development. However, once they started building their models in Excel, they understood the methodologies substantially better. Additionally, the students felt a strong sense of ownership for the full model-development process and the final product.

Note that the formal end-of-course surveys for SE403 are overly general to cover the large range of diverse Capstone projects. However, since the SE403 Capstone project was a continuation of a previous year's Capstone effort, the informal exit interviews can be used to qualitatively assess the effectiveness of using a DEVS-DMF for developing models. The previous team attempted to build human factor models and implement them into IWARS; however, the team was never able to adequately modify IWARS' methodologies and finished the course with only the Excel models. The exit interviews showed that the students were unsatisfied with their Capstone experience, having spent the bulk of the year "fighting the software." Since, they required significant help on manipulating IWARS, they also felt that they never had full ownership of the model. Additionally, they felt that their research deliverables lacked utility to the client.

LESSONS LEARNED

These projects were intended to introduce the students to distributed modeling, reinforce methodology development, and allow students to apply mathematical models to a real-world challenge. Overall the students in both SE485 and SE403 were able to meet these objectives based on the grades, exit interviews, and course end surveys. Though the projects met their objectives, they still revealed several lessons that can be incorporated for future variants of this type of project.

First, time is a critical factor, especially in methodology development. In SE485, the entire project spanned 3 weeks, with one week being given to literature review and methodology development. This timeframe resulted in students developing faulty methodologies which led to trivial or overly complicated models. Meanwhile, the SE403 course was able to spend a month simply on developing the methodologies. Though they still needed significant supervision and guidance for performing literature reviews, they were able to develop methodologies that accurately represented real-world situations while still being feasible. However, a normal semester-long M&S course would not be able to dedicate a month to methodology development. As such, the time factor can be handled by having the instructors perform the literature review for the students and providing them with the appropriate references for building their methodologies.

Second, the instructors took it for granted that the students would be able to read a technical paper, extract the relevant information, and use that information to construct a mathematical model. In particular, students were not comfortable making and documenting their assumptions. A discussion on assumptions would be necessary in the classes on model development or verification and validation.

Third, the modeling interface caused substantial issues. The students in SE485 and SE403 had little issue building their models in Excel, a software package that is used in every course in this department. However, the SE485 students had substantial issues converting the model into Python so that it could be integrated into the DEVS-DMF. Though the students had learned Python in a previous class, many students did not feel comfortable programming in it. The SE403 did manage to write their code in Scala, a language that none of them were familiar with. The students managed

to accomplish this by looking at examples that contained all the necessary syntax. In future iterations, the model might be integrated into the DEVS-DMF directly through Excel or coding examples will be given to the students so that they do not have to re-learn the syntax. Alternatively, the instructors could coordinate with the pre-requisite course to ensure that the students have adequate programming skills.

Fourth, these projects offer engineering students one of the few opportunities where they would be able to complete the full Systems Engineering Vee from defining the need to validation. Most engineering courses only cover a portion of the Vee, typically stopping at the designing of the system. However, by including the full process, students can understand how the integration and testing ties back into the model design. The SE485 project stopped prior to the testing portion; however, the testing could have been performed through other simulation packages, such as Virtual Battle Space, or through field experiments. The SE403 incorporated validation through tests on the Engagement Skills Trainer, which simulates a rifle range.

CONCLUSIONS

Engineering M&S courses have historically used academic versions of commercial M&S software, which often have pre-programmed, non-editable methodologies. Though these packages are useful for students to appreciate the scope of M&S, they do not get to perform the full process of model development. Additionally, the instructors must tailor their coursework to align with the capabilities of the M&S package. Students end up spending the course "fighting the software" without learning about the proper methodologies or the developing an appreciation for the use of M&S to solve real-world challenges.

These issues can be resolved through the use of a DEVS-DMF structure to teach M&S. A DEVS-DMF is an M&S framework that allows for the integration of multiple stateless models into a discrete event simulation. This framework is a useful tool for teaching M&S. In particular, it avoids the constraints of typical academic software, allowing students to develop methodologies, build models, work as a team, and builds upon their previous mathematical and coding skills. Additionally, the DEVS-DMF offers increased flexibility, allowing the students to conduct a meaningful analysis on relevant issues.

The DEVS-DMF was implemented in two courses at the United States Military Academy. The use of the DEVS-DMF resulted in an increased understanding of methodologies and model development. Additionally, these projects identified shortcomings in the students' abilities to generate their own methodologies and write computer code. Ultimately, these courses provided the cadets with the opportunity to develop a full analytic model to analyze current military issues facing the United States. This experience provided them with a skill set that could help them in their future military careers where they can apply M&S to solve other challenging problems.

REFERENCES

Agha, G. A. (1985). *Actors: A Model of Concurrent Computation in Distributed Systems*. MIT Artificial Intelligence Lab.

Chow, A. C. H., and Zeigler, B. P. (1994). Parallel DEVS: A Parallel, Hierarchical, Modular, Modeling Formalism. In M. Manivannan (Ed.) *Proceedings of the 26th Conference on Winter Simulation*, pp. 716-722.

Feurzeig, W., Roberts, N., (1999). *Modeling and Simulation in Science and Mathematics Education. Springer-Velag, New York, NY, pp. 166-168.*

Garcia, H. and Centeno, M. A. (2009). S.U.C.C.E.S.S.F.U.L.: a Framework for Designing Discrete Event Simulation Courses. In *Proceedings of the 2009 Winter Simulation Conference*, pp. 289-298.

Grasas, A., Ramalhinho, H., Juan, A. A. (2013). Operation Research and Simulation in Master's Degrees: A Case Study Regarding Different Universities in Spain. In *Proceedings of the 2013 Winter Simulation Conference*, pp. 3609-3619.

Hoad,K., Kunc, M., (2018). "Teaching System Dyanmics and Discrete Event Simulation Together: a Case Study." *Journal of the Operational Research Society*, 69 (4), pp. 517-527.

Kewley, R., Kester, N., and McDonnell, J., (2016). DEVS Distributed Modeling Framework - A Parallel DEVS Implementation via Microservices. In F. Barros (Ed.) *Proceedings of the 2016 Symposium on Theory of Modeling and Simulation.* Pasadena, CA.

Klug, M. and Hausberger, P. (2009). Motivation of Students for Further Education in Simulation by an Applied Example in a Related Other Course in Engineering Education – a Case Study. In *Proceedings of the 2009 Winter Simulation Conference*, pp. 248-254.

Kress, R., Cemerlic, A., Kress, J. Varghese, J., (2010). "Discrete Event Simulation Class for Engineering Graduate Students." In *Proceedings of the 2010 Winter Simulation Conference,* pp. 344-352.

Saltzman, R. M. and Roeder, T. M. (2013). Perspectives on Teaching Simulation in a College of Business. In *Proceedings of the 2013 Winter Simulation Conference,* pp. 3620-3629.

Stern, F., Xing, T., Muste, M., Yarbrough, D., Rothmeyer, A., Rajagopalan, G., Caughey, D., Bhaskaran, R., Smith, S., Hutchings, B., MOeyken, S., (2006). "Integration of Simulation Technology into Undergraduate Engineering Courses and Laboratories." *International Journal of Learning Technology,* Vol 2., No. 1.

Taco, A. A. (2011). Model Development in Discrete-Event Simulation-Insights From Six Expert Modelers. In *Proceedings of the 2011 Winter Simulation Conference,* pp. 3928-3939.