THROUGH THE VIRTUAL BARRIER: VIRTUAL PROTOTYPING AND ANALYSIS WITH MODEL-BASED SYSTEMS ENGINEERING

Omar Valverde
The MITRE Corporation
202 Burlington Road, Bedford, MA, USA
ovalverde@mitre.org

Larry Sun
The MITRE Corporation
202 Burlington Road, Bedford, MA USA
larrys@mitre.org

ABSTRACT

The International Council on Systems Engineering (INCOSE) Vision 2025 states that advancing Model-Based System Engineering (MBSE) beyond its current state of practice should include full integration with other engineering models, representation of different characteristics of a system with different models, and utilization of immersive technologies for efficient shared human understanding of systems. Accomplishing these goals would address frequent weak communication links between system users and system developers and facilitate the development of systems that are adaptable to changing operational environments. A literature review performed identified video game engines as a technology that can enable the accomplishment of INCOSE's goals, and a framework is proposed that augments the functionality and capabilities of model-based systems engineering with this technology. The initial framework prototype developed for a ground vehicle is described. This research is directly applicable to many issues that have been present in system development programs across the United States government.

Keywords: Model-Based Engineering, Virtual Prototyping, Video Game Technology

1 INTRODUCTION

In 2014 the International Council on Systems Engineering (INCOSE) published the Systems Engineering Vision 2025 where goals were put forth for advancing the state of practice of Systems Engineering. Specifically, SE Vision 2025 laid out the following three goals in order to advance Model-Based Systems Engineering (INCOSE, 2014):

1. Formal systems modeling is standard practice for specifying, analyzing, designing, and verifying systems, and is fully integrated with other engineering models.
2. System models are adapted to the application domain, and include a broad spectrum of models for representing all aspects of systems.
3. The use of internet-driven knowledge representation and immersive technologies enable highly efficient and shared human understanding of systems in a virtual environment that span the full life cycle from concept through development, manufacturing, operations, and support.

The aim behind model-based systems engineering is to migrate from a document centric practice to a model based practice. System models are generally descriptive models that are represented in graphical modeling languages such as the Unified Modeling Language (UML) or its extension, the Systems Modeling Language (SysML). Sanford Friedenthal illustrated in his book, *A Practical Guide to SysML*, a figure which is shown here as Figure 1 (Friedenthal et al. 2015).

This figure shows how a system model can serve as a consistent source for engineering truth with traceability between the following:

- system specifications and documentation
- results from analytic models
- engineering domain analysis and tools

Exact use of the system model is dependent on the MBSE approach taken.

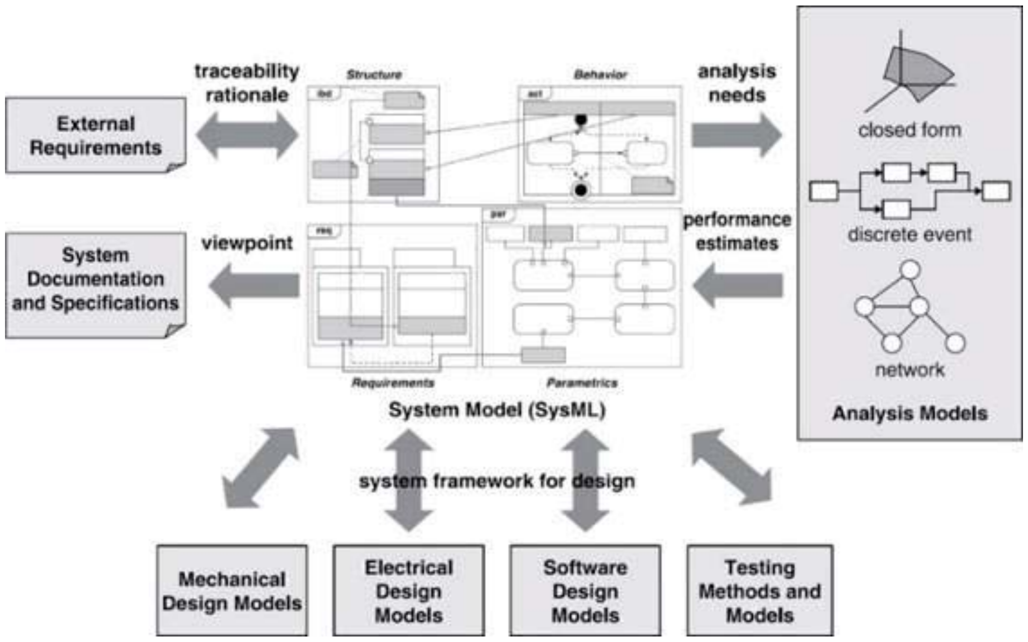

Figure 1: Framework for Traceability and Analysis.

The elements within Figure 1 are owned by specific stake holders/decision makers of the system. Each of these stake holders may utilize independent sets of tools to develop and/or asses the system. These tools are brought together to form part of an overall integrated systems environment, shown in Figure 2 (Friedenthal et al. 2015), that enables the framework in Figure 1. The tools within this environment, tightly or loosely coupled, feed into various simulation and analysis scenarios and can be used for verification and validation of the system. Ideally, there is an information/data exchange at each of the boundaries that is seamless and eliminates the need for manual exchange of data. Facilitating this type of integration of tools for information/data exchange has been undertaken by commercial, academic, and independent bodies in order to move systems engineering into a more cohesive environment.

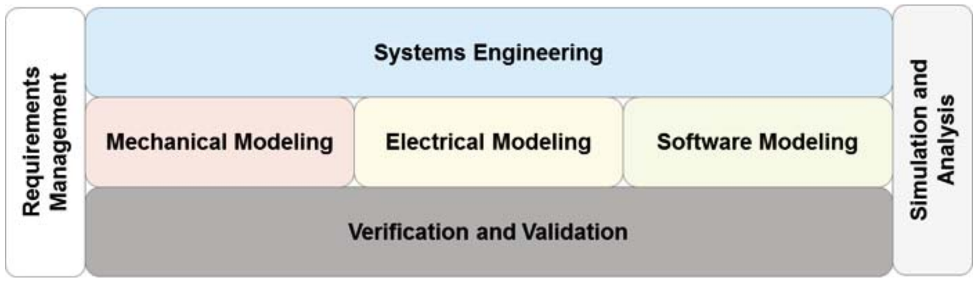

Figure 2: Integrated Design, Development and Analysis Systems Environment.

Verification and validation efforts seek to answer two questions:

- Are we building the right system?
- Are we building the system right?

Verification and validation can be performed with reference to the concept of operations (CONOPS) developed for a system. However, the paper "Model-Based Concept of Operations Development Using Gaming Simulation Preliminary Findings" (Korfiatis et al. 2015) highlights how the document based method of capturing CONOPS leads to misunderstanding between the various stake holders and negligence in updating the document as the system evolves. Thus, the authors establish the need for a shared mental model through a graphical CONOPS. This graphical CONOPS leverages gaming engines to simulate the current and future mission operations of a system. The authors propose this method as a way to bring a mutual understanding of the system across the various stake holders.

A paper published by the Defense Acquisition University (Smith and Vogt, 2014) cited and expanded on Dr. Korfiatis's work. In this paper, the authors developed a notional systems engineering process to develop a ground vehicle for the United States Army. This systems engineering process would utilize video games as a way to allow the end users to "virtually kick the tires" of a new ground vehicle system. In essence, the authors were looking at how to exploit gaming platforms as a way to virtually prototype systems.

The advances made in the physics engines used for video game platforms and the application highlighted by the above papers seeded the idea of leveraging gaming technology in order to extend the capabilities of model-based systems engineering and bring about INCOSE's 2025 Vision. Gaming/virtual engines have evolved from a single isolated game to a networked virtual platform with the capability to run many mathematical calculations in the background. It can be inferred from data by the Entertainment Software Association (2015) that approximately \$1.4 – \$3.8 billion is invested by video game companies in internal research and development efforts. The development of Microsoft's Hololens, the Oculus Rift, and other virtual reality technologies are creating a much more immersive experience for the users.

Successfully augmenting MBSE capabilities with gaming technology would move the systems engineering practice toward INCOSE's 2025 vision by:

- Enabling the traceability of multiple engineering model outputs to system user interaction in an operationally relevant scenario,
- Enabling the broad representation of various models in a virtual environment that can facilitate a shared human understanding of systems, and
- Enabling an immersive environment where various stake holders can interact with the system(s) in consideration.

2 THREE DOMAIN AREAS

The tools that would satisfy the elements in Figure 2 were categorized into three domain areas. Requirements and systems engineering were categorized together as systems engineering tools (MBSE tools). The middle row tools were categorized as engineering analysis tools. Simulation, verification and validation can be done within a gaming environment due to the embedded physics engines that enable virtual prototyping. As such, Figure 3 illustrates the three domain areas and the interfaces that would enable information/data exchange. In the following sections, the state of practice in each area will be addressed. Progress in establishing interfaces for data/information exchange between the tools within each domain area will also be discussed.

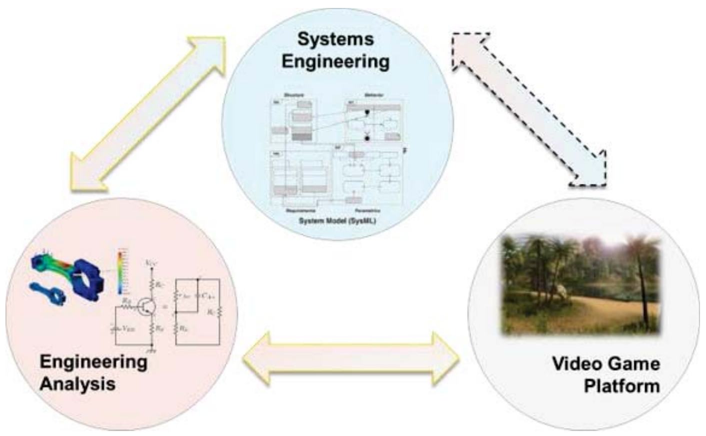

Figure 3: Domain Area and Current Interface Availability.

2.1 Area 1: Systems Engineering

MBSE tools that implement the Systems Modeling Language (SysML) have been developed in order to migrate systems engineering from a document centered practice to a more model based approach. The extension of SysML from the Unified Modeling Language (UML) has provided diagrams that are specific to the requirements for a system. These requirements can be modeled directly in an MBSE tool or many MBSE tools have the capability to integrate to other requirements tools (e.g. IBM DOORS). An example of this would be the integration capability of a DOORS requirement database and IBM's Rational Rhapsody. SysML system models facilitate the traceability between requirements, use case and system components and/or elements.

Many MBSE tools allow for the transformation of the descriptive system model into code (Java, C++, etc.) The code representation of the model allows execution for further verification and validation, specifically the behavioral aspects of the system. MITRE has used this ability extensively, and has leveraged this capability in order to create a systems integration laboratory where the executable model is the backbone that facilitates integration with other component models (Campbell, Garrett, and Wheeler, 2015).

2.2 Area 2: Engineering Analysis

Updates are continually made to various engineering analysis software packages. These updates can range from computation algorithms to implementing academic advancements in the respective tool domain. This cycle of research and implementation in engineering tools is well understood and will not be further elaborated.

2.3 Area 3: Video Game Platform

As stated earlier, the video game industry is continually evolving. Application of gaming technologies for non-entertainment purposes, termed serious gaming, has been implemented for various purposes beyond those stated previously including differing engineering activities (Uskov and Bhuvana, 2014). The human-in-the-loop aspect of gaming technology facilities an ability to capture user impact data (Kosmadoudi et al. 2013). Other applications of serious games include:

- Citizen government participation (Ahmed et al. 2015)
- Democracy and Government Funded Video Games (Losh, 2006)

A more in depth overview of the evolution of gaming technologies beyond entertainment can be read in Zyda (2005).

2.4 Interface 1: Systems Engineering Tools – Engineering Analysis Tools

The idea of integrating engineering analysis tools with MBSE tools is not new and has been undertaken by different commercial and academic bodies. The Object Management Group has developed the Functional Mock-Up Interface (FMI) for model and data exchange between various engineering models (Blochwitz et al. 2011). Third party tools, such as Phoenix Integration's Model Center, have also developed the capability to integrate system architectures via the SysML parametric diagrams with other engineering domain tools. (Min et al. 2011). Integration via executable models and OpenMDAO has also been explored (Balestrini-Robinson et al. 2015). There are also various standards that have been developed for multi-model simulation (e.g. High Level Architecture).

2.5 Interface 2: Engineering Analysis Tools – Video Game Platforms

Though a direct interface between engineering analysis tools and video game platforms is not wide spread, data and information exchange has occurred in various academic and commercial settings. Third party tools have been developed to transform a CAD model to a 3D model compatible with the gaming environments This has also been studied in the academic realm (Kosmadoudi et al. 2013). Gaming technology has been used for data visualization (Lv et al. 2013). Gaming technology has also been used in the manufacturing realm as seen in Duin, et al. (2011) and Stark (2010).

2.6 Interface 3: Systems Engineering Tools – Video Game Platforms

Review of the literature found this interface area to not be currently actively pursued. The notion of virtual prototyping is seen in the literature such as McIntosh (2012) and Kande (2011). However, the virtual environment utilized are purpose built and do not leverage the gaming industry available assets. Developing the interface using the commercially available gaming engines would tap into the capabilities that are already being developed by the commercial industry. This would alleviate upfront costs and labor, and lead to the faster development of analysis environments. In short, successfully implementing an interface here would facilitate three goals of systems engineering research (Mavris and DeLaurentis, 2000):

- Reduce initial costs by leveraging the commercial gaming industry R&D updates
- Increase system knowledge earlier in development by assessing system performance in operational relevant scenarios earlier
- Maintain design freedom by evaluating multiple system options in operational environments

3 INITIAL PROOF-OF-CONCEPT PROTOTYPES

3.1 Single Instantiation Framework

The authors found it beneficial to expedite the production of a proof-of-concept framework prototype. Tools and software packages familiar to the author were selected as a means to mitigate any time delays that would occur due to learning curves. By leveraging these tools, the work focused on the integrating aspects of the framework.

IBM's Rational Rhapsody Developer was selected as the model-based systems engineering tool. Rational Rhapsody has the capability to generate code in various programming languages. For the virtual environment, Unity3D was used. Games and virtual environments developed in Unity 3D are written and compiled in programming languages that differ from the code generation capabilities of Rational Rhapsody. After exploring various options, it was decided that developing a methodology that overcomes the language obstacle would set the path for a framework that could be tool agnostic.

The work and success of MITRE with Model-Based System Engineering and integration with executable models provided a springboard for tackling the programming language obstacle. The authors of "Using model based engineering to own the technical baseline" (Campbell et al. 2015) successfully integrated an executable UML model in Rational Rhapsody with the ActiveMQ pub-sub messaging bus. Binaries online for ActiveMQ included those that were compatible with the language and compiling environment for Unity3D.

The message format used to communicate between the Rhapsody model and Unity was based off simple strings, and the messaging system was used in a publish-subscribe configuration. Though, it was recognized that these are not the most efficient methods of transferring data when trying to achieve a real-time (or near real-time) system execution, this was deemed good enough for the initial evaluation of what would have to be modeled where (e.g. behavioral characteristics in MBSE, kinematics in the virtual environment).

The use of a messaging bus allowed the user-input to be abstracted out from the virtual environment. This enabled future flexibility in the types of human interfaces that could be used (beyond the typical gamepads) by decoupling the virtual environment and human interface and isolating messages that impacted only the architecture. Due to this, normal game scripting conventions were not followed in order to facilitate the integration with the Unity environment. Instead of directly sending the user input to the video game model, the system physical calculations (e.g torque produced by the powertrain) would first be processed by the architecture in the Rhapsody model using the input string message on the message bus, then the kinematic physics were evaluated by Unity once it registered an update on the bus.

Figure 4 illustrates the initial framework that was developed. A ground vehicle was architected in Rational Rhapsody using the SysML modeling language. A corresponding 3-D model was put into a notional virtual environment. A synchronous relationship was developed between the vehicle behavioral states and powertrain characteristics. For example, if an executing architecture indicated that the vehicle was off then the virtual environment would not respond to any user input until the vehicle's state was switched to on.

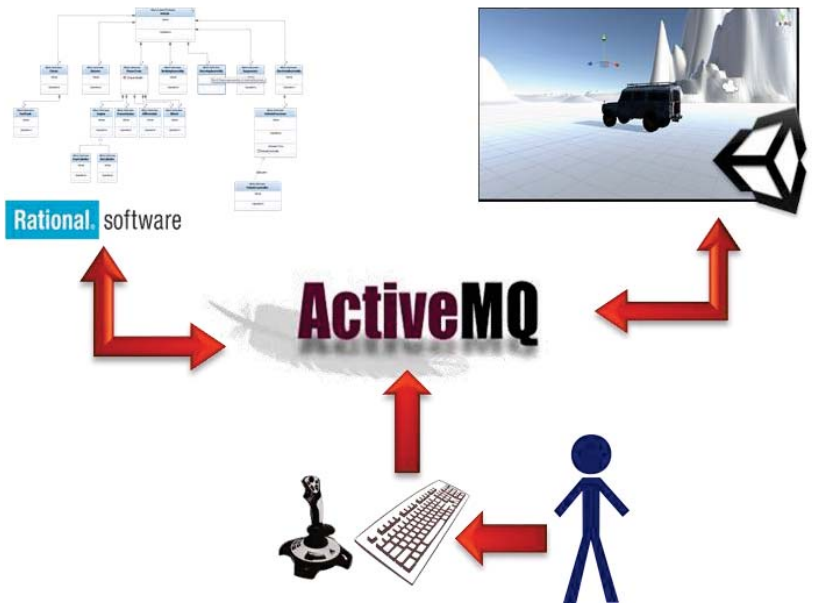

Figure 4: Initial Single Instantiation Framework.

The powertrain components' (engine, transmission, and differential) performance characteristics were modeled in the Rational Rhapsody with the resulting torque being sent via the messaging bus to the virtual environment. The virtual environment would then apply the resultant torque on the vehicle 3-D model and render the resultant reaction. As a simple example, in the equation force equals mass times acceleration, Rational Rhapsody calculated the force while the virtual environment would output the corresponding acceleration.

Engine horsepower verses torque curves, transmission gear ratios, and differential ratio data were retrieved from the Chevrolet website. By extracting the powertrain characteristics into Rational Rhapsody, various powertrain configurations could be evaluated in the operational scenario by simply swapping out the corresponding module in the architecture (i.e. swap out engine A for engine B).

3.2 Networked Framework Instantiation

Once a single architecture was successfully integrated with a virtual environment, work was conducted to leverage the multiplayer network capabilities of virtual environments. Leveraging this capability would support the goal of having multiple systems with their respective architectures interact in a single virtual environment. This could then facilitate evaluating system of systems' concept of operations. This type of framework was termed a networked framework.

Figure 5 illustrates the initial networked instantiation. Each system representation within the networked framework required its own respective architecture, human interface, and a client representation of the virtual environment. A single messaging bus was utilized and it was kept on the same server where the virtual environment would be hosted. A successful case study was conducted where two different ground vehicles interacted in the same virtual environment. Each ground vehicle architecture had a different powertrain instantiation (i.e. ground vehicle A had one engine and ground vehicle B had a different engine) and a different human interface.

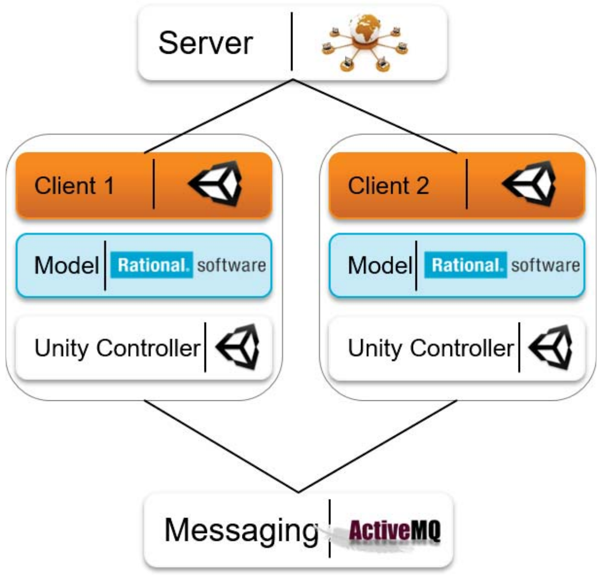

Figure 5: Networked Framework Instantiation.

3.3 Further work

Issues that arose in the instantiated frameworks that will need to be addressed are the latency that can occur due to the large amounts of messages being sent across the messaging bus. The sending, reception, interpretation, and execution of each message has to be nearly real time. Due to this, system response began to lag and thus affected the user immersion into evaluating the system. A temporary solution was implemented that limited the amount of messages going across the message bus. However, the temporary solutions cannot serve the long term needs nor the scalability of the framework. Work has already been undertaken to address this issue and will be published subsequently.

Long term scalability for the framework would necessitate that the physics modeling be further extracted from Rhapsody and be calculated utilizing tools that were designed for those purposes. As stated earlier, work has already been done in integrating and interfacing engineering domain tools with model-based systems engineering tools. Work will be performed such that the next iteration of the framework will complete the triangle illustrated in Figure 3 and Figure 6. In other words, the tools from the three domains operating together in a single integrated systems engineering development and analysis environment.

In terms of networking, further work needs to be done to improve the execution of the simulations in a networked environment. Current network implementations can result in a noticeable amount of movement latency and inaccuracy when more than two clients are operating within the same simulation. Work will be done to enhance the performance of the multiplayer experience to allow for a more realistic environment in which to test scenarios. Further work must also be done to test the performance of the framework under a heavier load with the addition of more clients to allow for a wider range of scenarios and configurations.

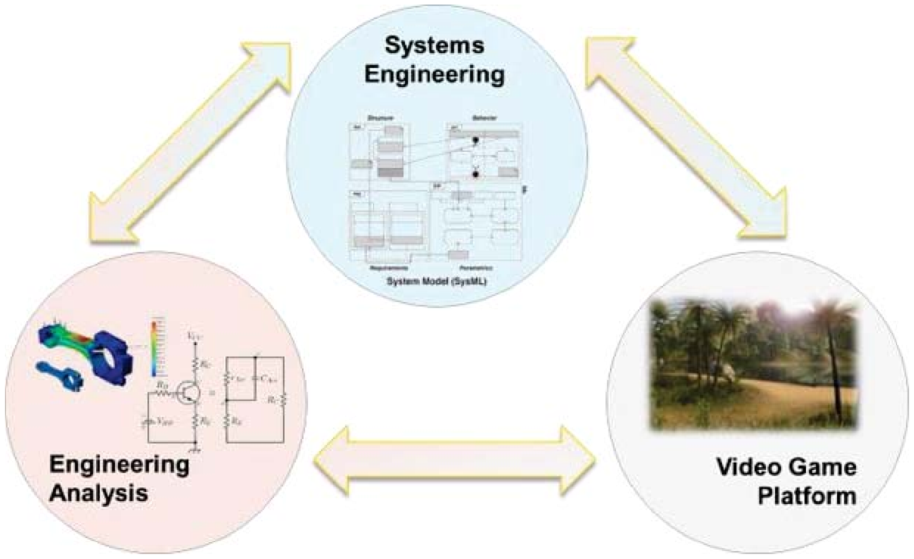

Figure 6: Interfaces Between the Three Domain Areas.

A methodology for evaluation and assessment of architectures in an operational relevant environment will also be incorporated. This effort will leverage the work presented in Garcia and Tolk (2015).

4 CONCLUSION

The augmentation of model-based system engineering with virtual engines is a step in the direction of helping realize INCOSE’s Vision 2025 for systems engineering. The executable aspects of an architecture and representation in a virtual environment facilitates shared human understanding via virtual prototyping. The work is ongoing in order to address the issues stated in the previous section. Also, case studies are being designed and will be executed in order to assess how the use of this type of framework impacts the overall decision-making process for system stakeholders.

ACKNOWLEDGMENTS

The authors would like to acknowledge Michael Dinsmore, Brian Marshall, Randall Hudson, and Jeremy Lassetter from MITRE for their contributions and assistance throughout the research project. Michael Dinsmore and Brian Marshall served as advisors to the project while the rest served as individual contributors. Their technical talent allowed for this first year of the project to be a success. Also, we would like to acknowledge Diana Carlson Valverde for her contributions on edits.

REFERENCES

Ahmed, Alsanossi M., Qasim H. Mehdi, Robert Moreton, and Adel Elmaghraby. "Serious games providing opportunities to empower citizen engagement and participation in e-government services." In *Computer Games: AI, Animation, Mobile, Multimedia, Educational and Serious Games (CGAMES)*, 2015, pp. 138-142. IEEE, 2015.

Blochwitz, Torsten, Martin Otter, Martin Arnold, Constanze Bausch, H. Elmqvist, A. Junghanns, J. Mauß et al. "The functional mockup interface for tool independent exchange of simulation models." In *Proceedings of the 8th International Modelica Conference*; March 20th-22nd; Technical Univeristy; Dresden; Germany, no. 063, pp. 105-114. Linköping University Electronic Press, 2011.

Balestrini-Robinson, Santiago, Dane F. Freeman, and Daniel C. Browne. "An Object-oriented and Executable SysML Framework for Rapid Model Development." *Procedia Computer Science 44* (2015): 423-432.

Campbell, Dave, Garrett Wampole, and Tom Wheeler. 2015. "Using model based engineering to own the technical baseline." *Procedia Manufacturing 3*. 1995-2002.

Duin, Heiko, and Klaus-Dieter Thoben. "Serious gaming for sustainable manufacturing: A requirements analysis." In *Concurrent Enterprising (ICE)*, 2011 17th International Conference on, pp. 1-8. IEEE, 2011.

Entertainment Software Association. "*Essential facts about the computer and video game industry*." (2015).

Friedenthal, Sanford, Alan Moore, and Rick Steiner. 2015. *A Practical Guide to SysML: The Systems Modeling Lannguage*. Waltham: Elsevier.

Garcia, Johnny J., and Andreas Tolk. "Executable architectures in executable context enabling fit-for-purpose and portfolio assessment." *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology 12, no. 2* (2015): 91-107.

INCOSE. 2014. *A World in Motion: Systems Engineering Vision 2025*. San Diengo: INCOSE.

Losh, Elizabeth. "Making things public: democracy and government-funded videogames and virtual reality simulations." In *Proceedings of the 2006 ACM SIGGRAPH symposium on Videogames*, pp. 123-132. ACM, 2006.

Lv, Zhihan, Alex Tek, Franck Da Silva, Charly Empereur-Mot, Matthieu Chavent, and Marc Baaden. "Game on, science-how video game technology may help biologists tackle visualization challenges." *PloS one* 8, no. 3 (2013): e57990.

Kande, Akshay. "Integration of model-based systems engineering and virtual engineering tools for detailed design." (2011).

Korfiatis, Peter, Robert Cloutier, and Teresa Zigh. 2015. "Model-Based Concept of Operations Development Using Gaming Simulation Preliminary Findings." *Simulation & Gaming*. SAGE Publications.

Kosmadoudi, Zoe, Theodore Lim, James Ritchie, Sandy Louchart, Ying Liu, and Raymond Sung. "Engineering design using game-enhanced CAD: The potential to augment the user experience with game elements." *Computer-Aided Design* 45, no. 3 (2013): 777-795.

Min, Byung I., Aleksandr A. Kerzhner, and Christiaan JJ Paredis. "Process integration and design optimization for model-based systems engineering with SysML." In *ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 1361-1369. American Society of Mechanical Engineers, 2011.

Mavris, Dimitri N., DeLaurentis, Daniel A. 2000. "A probabilistic approach for examining aircraft concept feasibility and viability." *Aircraft Design* (Elsevier) 3 (2): 79-101.

McIntosh, Paul, Aleksandar Subic, Ka Wai Lee, Patrick Clifton, Pavel Trivailo, and Martin Leary. "An adaptable virtual engineering platform for distributed design based on open source game technology." *Advances in Engineering Software* 43, no. 1 (2012): 71-86.

Smith, Robert E., and Brian D. Vogt. *A Proposed 2025 Ground Systems, Systems Engineering Process*. DEFENSE ACQUISITION UNIV FT BELVOIR VA, 2014.

Stark, Rainer, F-L. Krause, Christian Kind, Uwe Rothenburg, Patrick Müller, H. Hayka, and H. Stöckert. "Competing in engineering design—The role of Virtual Product Creation." *CIRP Journal of Manufacturing Science and Technology* 3, no. 3 (2010): 175-184.

Uskov, Alexander, and Bhuvana Sekar. "Serious games, gamification and game engines to support framework activities in engineering: Case studies, analysis, classifications and outcomes." In *IEEE International Conference on Electro/Information Technology*, pp. 618-623. IEEE, 2014.

Zyda, Michael. "From visual simulation to virtual reality to games." *Computer* 38, no. 9 (2005): 25-32.

AUTHOR BIOGRAPHIES

OMAR VALVERDE is a Lead Systems Engineering in The MITRE Corporation within the Emerging Systems Engineering Technologies Department. He holds a Master's degree in Aerospace Engineering from Georgia Institute of Technology. His e-mail address is ovalverde@mitre.org.

LARRY SUN is an Emerging Technologies Software Engineer in The MITRE Corporation within the Emerging Systems Engineering Technologies Department. He holds a Bachelor's degree in Computer Engineering from Boston University. His e-mail address is larrys@mitre.org.