Design and Development Tools for the Systems Engineering Experience Accelerator – Part 2
Technical Report No. SERC 2016-TR-110

May 9, 2016

Principal Investigator:
Dr. Jon Wade, Stevens Institute of Technology

Co-Principal Investigator:
Dr. Douglas Bodner, Georgia Institute of Technology

Research Team
Dr. Richard Turner, Stevens Institute of Technology
John Hinkel, Georgia Institute of Technology
Peizhu Zhang, Stevens Institute of Technology
Table of Contents

Table of Contents .............................................................................................................................................. iii
Tables ................................................................................................................................................................ iv
Figures ................................................................................................................................................................. iv
Preface ................................................................................................................................................................. v

Abstract.............................................................................................................................................................. v
Project Documentation ......................................................................................................................................... v

1 Introduction ....................................................................................................................................................... 1
  1.1 Educational and Training Challenges ........................................................................................................ 1
  1.2 Experience Accelerator Research Program ............................................................................................ 2
    1.2.1 Summary of Original Prototype Requirements and Functions ....................................................... 2
    1.2.2 Research Objectives .......................................................................................................................... 5
  1.3 EA Tools ..................................................................................................................................................... 6
    1.3.1 Identification of Tools Needed for Future Experience Design and Development ...................... 6
    1.3.2 Research Approach for this Task ....................................................................................................... 9

2 Simulation Tools .............................................................................................................................................. 11
  2.1 Success Metrics ........................................................................................................................................ 12
  2.2 Critical Use Cases ................................................................................................................................ 13
  2.3 Requirements ......................................................................................................................................... 16
  2.4 Integrated Interface ................................................................................................................................. 17
  2.5 Sim Builder ............................................................................................................................................. 17
  2.6 Sim Tuner ............................................................................................................................................... 18
  2.7 Chart Designer ..................................................................................................................................... 22
  2.8 Simulator Execution Code .................................................................................................................... 23
  2.9 Evaluation .............................................................................................................................................. 24

3 Experience Building Tools ................................................................................................................................ 24
  3.1 Tools overview ...................................................................................................................................... 24
  3.2 Integrated Toolset ................................................................................................................................ 25
  3.3 Functionalities ....................................................................................................................................... 26
  3.4 Evaluation .............................................................................................................................................. 29
    3.4.1 Internal Evaluation ............................................................................................................................ 29
    3.4.2 External Evaluation .......................................................................................................................... 29

4 Learning Assessment Tool Prototype ............................................................................................................ 30
  4.1 assessment of systems thinking ............................................................................................................. 31
  4.2 assessment of systems engineering ..................................................................................................... 32
  4.3 Learning Assessor Prototype ............................................................................................................... 32
  4.4 Evaluation .............................................................................................................................................. 34

5 Infrastructure Modification for Supporting Tools .......................................................................................... 34

6 Conclusions and Future Research .................................................................................................................. 35
  6.1 Conclusions ........................................................................................................................................... 35
  6.2 Future Research ..................................................................................................................................... 38

References ............................................................................................................................................................. 40
Tables

Table 1. Limitations and solutions ........................................................................................................................................ 7
Table 2. Tool development in three parts ......................................................................................................................... 35
Table 3. Part 3 work plan .................................................................................................................................................. 39

Figures

Figure 1. Experience Accelerator architecture .................................................................................................................. 3
Figure 2. Relationship of tools to EA architecture elements ............................................................................................. 7
Figure 3. Navigation among tools ....................................................................................................................................... 17
Figure 4. Sim Builder with shared nodes ............................................................................................................................ 18
Figure 5. Sim Tuner at end of Part 1 .................................................................................................................................. 19
Figure 6. Execute interface of Sim Tuner ........................................................................................................................... 19
Figure 7. Sub-model graph .................................................................................................................................................. 20
Figure 8. Graph with learner-decidable variables ........................................................................................................... 21
Figure 9. Chart depicting possible outcomes of decisions impacting weight ................................................................. 21
Figure 10. Example range chart ......................................................................................................................................... 23
Figure 11. Chart Designer interface .................................................................................................................................. 23
Figure 12 Overview of Tools and Aspects of the EA Experience ....................................................................................... 24
Figure 13 User Interface of Experience Editor ................................................................................................................ 25
Figure 14 Phase Editor User Interface ............................................................................................................................ 26
Figure 15 Event Editor User Interface .............................................................................................................................. 27
Figure 16 Artifact Integrator User Interface ...................................................................................................................... 28
Figure 17 CDR Criteria Editor User Interface .................................................................................................................. 28
Figure 18 Learning Assessor Interface with Project Status ............................................................................................. 33
Figure 19 Learning Assessor Interface with Recommendation Data .................................................................................. 33
Figure 20 Learning Assessor Performance Comparison Interface .................................................................................... 34
Preface

ABSTRACT

This document is a summary of the work that was completed in Part 2 of the SERC Research Topic DO1/TTO2/0123 “Design and Development Tools for the Systems Engineering Experience Accelerator (SEEA)” supported by Strategic Initiatives, Office of the DASD (Systems Engineering). The tools development efforts fall into four major categories – simulation tools for building and testing simulation models that mimic the behavior and results of acquisition programs that focus on system design and development, experience building tools that provide the structure for such system engineering experiences and the events that occur in them, learning assessment tools to measure the efficacy of the experience, and EA infrastructure changes to support this work.

The simulation tool development project achieved its Part 2 objectives with the completion of the requirements and use cases for simulation tools, a review of existing tools to leverage, the development of a Prototype Sim Builder with GUI for model building and capability to manipulate sub-models for purposes of modularity and model archiving/curation and the development of a Prototype Sim Tuner that allows testing of model behavior drilled down to variables of interest. The Experience building tools development project achieved its Part 2 objectives with the completion of the requirements, a review of existing tools to leverage, and the use cases and prototypes for the Phase Editor, Event Editor and Artifact Integrator. Progress was achieved in the learning tools area with a literature search, the exploration of an initial concept and the identification of data to be captured by the Experience Accelerator.

A successful demonstration of the tools has been given to the sponsor. The tools are now at the stage where they are ready to be evaluated by external users for their use in Experience and Simulation development. An iterative development approach was quite successful at providing incremental functionality that was reviewed with its potential users throughout the research effort, prioritizing the most important features and delivering working prototypes throughout the effort. This approach will be continued in Part 3 of this research program.

PROJECT DOCUMENTATION

The following are the documents that were produced by this research and may be referenced in this document:

RT146 Design and Development Tools for the SEEA Project documents:

- RT146 Technical and Management Work Plan (A009)
- RT146 Bi-Monthly Status Reports (A008)
• RT1XX Design and Development Tools for the Systems Engineering Experience Accelerator, Part 3 proposal

Publications and Reports:
1 INTRODUCTION

1.1 EDUCATIONAL AND TRAINING CHALLENGES

Systems engineering is a multidisciplinary practice and is as much of an art as it is a science. While a waterfall model of education can provide a background of domain-centric knowledge, it is not until this knowledge is put into practice in an integrated, real world environment that a systems engineer can develop the necessary insights and wisdom to become proficient. In the workplace, these learning events are often distributed sparsely over time such that an engineer may only see a complete system life cycle over a period of several years. As a result, the maturation time from completion of formal studies to becoming seasoned systems engineer is unacceptably long, particularly when contrasted with the clock speeds of today’s society in which career change is the norm rather than the exception, particularly among the young.

Clearly, there is a critical need to promote rapid skill development of systems engineers, in particular senior systems engineers, across the Department of Defense (DoD) and government workforces, as a large cohort of personnel is nearing retirement age. In addition, new systems engineering skills are needed to address important societal needs in national security, homeland security, airspace management and disaster recovery. These domains involve large-scale, systems-oriented solutions with increasingly limited budgets. Educational technologies hold the promise of providing customized learning exercises based on real-world situations to reduce the reliance on extensive on-the-job training that is the hallmark of current workforce development.

However, there are many challenges in developing effective educational technologies. Our prior work resulted in the development of the Systems Engineering Experience Accelerator (SEEA or EA); a first-generation educational technology set that provides experiential learning for systems engineers seeking technical leadership roles. This work also resulted in a learning experience based on design and development of an unmanned aerial vehicle (UAV) system for Department of Defense applications. What is of current interest is expanding the set of learning experiences to include different types of systems and development efforts. To do this successfully, we need to provide tools for the design and development of such experiences. Part 1 of this project initiated this effort and yielded a set of prototype tools for experience design and development, simulation model design and testing, and learning assessment. This report addresses Part 2 of the project, which has further developed the tools developed in part 1, initiated new tools such as a designer for simulation output charts, and focused on updating the Experience Accelerator application so that it is compatible with these tools.

The rest of the report is organized as follows. Section 1 discusses the research approach and the tools being developed. Section 2 presents tools used for simulation model development and testing, plus simulation output design. Section 3 focuses on tools for designing and developing experiences. Section 4 discusses tools for assessing learning. Section 5 concludes and provides a summary of future work.
1.2 EXPERIENCE ACCELERATOR RESEARCH PROGRAM

The RT-16 Systems Engineering Experience Accelerator (SEEA) project created a new approach to developing the systems engineering workforce which augments traditional, in-class education methods with educational technologies aimed at accelerating skills and experience with immersive simulated learning situations that engage learners with problems to be solved. Although educational technology is used in a variety of domains to support learning, the SEEA is one of the few such technologies that support development of the systems engineering workforce.

1.2.1 SUMMARY OF ORIGINAL PROTOTYPE REQUIREMENTS AND FUNCTIONS

The SEEA was developed to support a single-person role-playing experience in a digital environment, as well as a specific learning exercise in which a learner plays the role of a lead systems engineer for a DoD program developing a new unmanned aerial system. This exercise is based on the notion of experiential learning, and thus will be referenced as an experiential learning module. The learner engages with the experience (i.e., simulated world), makes decisions to solve problems, sees that results of those decisions, abstracts lessons learned from what was successful and what was unsuccessful, and then repeats the process in a series of cycles, simulating the evolution of the program over time.

The SEEA technology provides a graphical user interface allowing the learner to see the program status, interact with non-player characters to gain additional program information, and make technical decisions to correct problems. It also provides capability to simulate the program into the future, based on these learner decisions, so that outcomes can be shown to the learner. This cycle of decision and simulation-into-the-future supports the Kolb cycle of experiential learning; the Experience Accelerator uses multiple such cycles operating through the lifecycle of the program. In particular, this approach allows illustration of the effect of upstream decisions on downstream outcomes in the system lifecycle. The SEEA can support a wide variety of systems domains and areas of expertise through changes to the experience. Recently, additional multi-player technology is being developed to allow live player support for team-based learning, as well as for a mentor to provide advice and feedback.

The current set of Experience Accelerator technologies is shown in Figure 1, which illustrates the architecture of these technologies. In designing the Experience Accelerator, two principles are relevant. First, the architecture is highly modular to allow substitution of new different technologies, to accommodate new and better technologies as they become available. Second, there is a generic set of technologies that are used to support any experience module. The specifics of a particular experience module are housed in libraries and other data files separate from the generic EA technologies.
The modules currently included in the Experience Accelerator are:

- **Presentation engine** – Provides a user interface to the learner. This interface displays the status of the simulated program via a dashboard and status charts, and it lets the learner interact with a simulated program office. This interaction includes exploration of archived documentation on the program, querying of non-player characters about issues of interest regarding the program, and entering decisions about the program to correct perceived problems.

- **Experience master** – Maintains current value of state variables related to program status and performance and manages the invocation of other modules as needed. The experience master controls the flow of events and provides updated information from the other modules to the presentation engine for display. It also takes the learner inputs from the presentation engine and invokes internal functions or other modules as appropriate.

- **Non-player character engine** – Provides a number of non-player characters (NPCs) and associated dialog sessions based on the state of the program. The non-player characters have roles (e.g., program manager), and the learner is tasked with querying them about...
various aspects of program performance to determine the true state of the program and its underlying problems.

- Simulation engine – Maintains the detailed state of the program and advances this state via simulation models, incorporating learner decisions about the program. The simulation engine takes as input a base state of the program and then advances this state using any changes the learner may have entered. It also generates artifact charts detailing the program status and performance over time (e.g., costs incurred, schedule and progress indicators, system performance estimates relative to targets, etc.).

- Challenge control engine – Maintains learner history and skill levels and provides parameter tuning to increase or decrease the difficulty of the experience based on learner knowledge and skills, and his/her success.

Building an experience, however, can be a time-consuming task. The intent of this project is to automate much of the experience-building process by designing and developing a set of tools that an experience designer can use.

In developing a new experiential learning module using the current EA, the following steps would typically be followed:

Experience conceptual design:

- Determine the specific competencies, ‘aha’s’ and targeted difficulty levels to be supported.
- Select/design a particular scenario (e.g., a particular DoD program) for the experience that supports the desired learning and develop a storyboard for the desired experience elements.
- Define the program’s high-level performance measures, each measure’s limits that constitute success and failure, and the direction of “better.”
- Define phases and cycles within phases.
- Define challenges and landmines consistent with competencies, ‘aha’s’ and difficulty levels.
- Define clues that lead to recognition of problems and possible solutions.
- Identify the means of discovering the clues (artifacts, events, dialog).
- Specify important non-player characters/roles, plus any live player roles.
- Specify desired simulated program behavior, status outputs, inter-relations between elements, and learner control points.
- Specify desired artifacts (e.g., program background material, learner decision/recommendation forms, etc.).
- Specify the means provided to the learner for providing recommendations and decisions.
- Specify the types of feedback to be provided to the learner based on decisions made during the experience.
Experience development, testing and enhancement:

- Specify program status variables and targets that operate at a more detailed level than high-level performance measures.
- Implement phase and cycle behavior manually.
- Develop simulation models and datasets to ensure desired simulated program behavior and inter-relations of program elements (via testing/tuning).
- Develop artifacts to be populated with simulation output to provide learner insight into current and previous program status.
- Develop non-player characters and state-based dialog whereby the learner can query the NPCs to discover additional information (or be distracted by inconsequential minutia).
- Embed challenges and landmines into simulation models and NPC dialog.
- Develop/write desired artifacts (e.g., program background material, learner decision/recommendation forms, etc.).
- Write scripted feedback to learner based on alternate learner decisions, linked to program outcomes.
- Integrate artifacts, simulation models, NPC dialogs, and learner feedback into Experience Accelerator via a manual process involving re-linking or recompilation.
- Test consistency of experience.

In Part 1 of this project, these steps were prioritized in terms of which ones were most important for automation and which also were most amenable. Prototypes were developed for this tool set. Part 2 continues the development of these tools into more mature implementations.

1.2.2 Research Objectives

In Part 2, we continue with the same fundamental project statement, hypothesis, outcomes and goals as in Part 1.

Problem Statement: Traditional systems engineering education is not adequate to meet the emerging challenges faced by the Department of Defense and others in system acquisition and sustainment. New educational technologies such as the Systems Engineering Experience Accelerator hold the promise of facilitating a rapid skill and experience accumulation for the workforce to meet these challenges. However, to have scalable effect, such technologies cannot rely on extensive programming and low-level development to create a rich set of experiential learning modules needed to accelerate systems engineering workforce development.

Hypothesis: The Experience Accelerator technology will scale to support a community of developers engaged in creating modules for their organizations’ use if tools are developed that allow educators and other non-programmers to create, maintain and evolve experiential learning modules.
Measurable Outcomes: The outcomes from this research will be measured in two main ways. First, educators and others interested in creating experiential learning modules will provide qualitative feedback on the effectiveness and efficiency of the Experience Accelerator toolset in creating experiential learning modules based on their use of the design and development tools. Second, the number of such developers who commit to create modules for their organizations’ use will be tracked, as well as the number of organizations and variety of different application areas.

Research Goal: Validate the hypothesis through the creation of design and development tools for experiential learning modules that maximally leverage the current Experience Accelerator research, technology and content.

1.3 EA Tools

1.3.1 Identification of Tools Needed for Future Experience Design and Development

This research task is focused on developing a set of tools specifically for educators and developers outside the SEEA research and development team, to support their designing and developing learning modules for their use. It concentrates on a subset of possible tools prioritized by the likelihood of having the most impact on facilitating module development:

1. Sim Builder - Simulation model builder using libraries/templates
2. Sim Tuner - Parameter tuner that automates the tuning of parameters to yield desired outputs via batch processing of different combinations of settings
3. Chart Designer – Application to automate the process of specifying output charts from the simulator
4. Phase Editor - GUI-based tool for phase, cycle and event specification, with code generation
5. Event Editor - GUI or text-based tool to specify events and their triggers, with code generation
6. Artifact Integrator - Application that allows designer to take artifact files, such as design documents and enter it into EA application with automatic recompilation and re-linking
7. Learning Assessor - Assessment tool-suite that provides automated performance scoring and decision comparisons against proven baselines

Table 1 shows the limitations of designing and developing new experiences mapped to the tools above.
### Table 1. Limitations and solutions

<table>
<thead>
<tr>
<th>#</th>
<th>Limitation</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Complex simulation models with limited reuse supported</td>
<td><strong>Sim Builder</strong> - Simulation model builder utilizing libraries/templates</td>
</tr>
<tr>
<td>2</td>
<td>Complex simulation outputs dependent on hundreds of variables/parameters</td>
<td><strong>Sim Tuner</strong> - Parameter tuner that automates the tuning of parameters to yield desired outputs via batch processing of different combinations</td>
</tr>
<tr>
<td>3</td>
<td>Manual nature of chart specification</td>
<td><strong>Chart Designer</strong> – Application that automates the specification of charts</td>
</tr>
<tr>
<td>4</td>
<td>Manual nature of phase and cycle development</td>
<td><strong>Phase Editor</strong> - GUI-based tool for phase, cycle and event specification with code generation</td>
</tr>
<tr>
<td>5</td>
<td>Manual nature of specifying events and their triggers in the Experience</td>
<td><strong>Event Editor</strong> - GUI or text-based tool to specify events and their triggers with code generation</td>
</tr>
<tr>
<td>6</td>
<td>Manual nature of artifact integration involving re-linking and recompilation</td>
<td><strong>Artifact Integrator</strong> - Artifact entry application that allows designer to take an artifact file and enter it into EA application with automatic recompilation and re-linking</td>
</tr>
<tr>
<td>7</td>
<td>Manual nature of assessment of learner performance</td>
<td><strong>Learning Assessor</strong> - Assessment tool-suite that provides automated performance scoring, decision comparisons against proven baselines, etc.</td>
</tr>
</tbody>
</table>

Figure 2 shows the relationship of these tools to the various elements that constitute the Experience Accelerator (from Figure 1). Two additional tools (Dialog Editor and Visuals Editor) are shown. These are not addressed in the current project, but are being investigated separately.
It should be noted that the set of design and development tools noted above does the team identify a subset of the possible tools and template, and they are prioritized as the most important ones for inclusion in this work. Others include:

- Experience concept tools (storyboarding, learner profile creation,
- Context tools (project specification linked to learning outcomes, NPC roles/motivations/personalities), and
- Experience module events/flows (automated linkages between challenges/landmines and competencies/’aha’s’ and mitigating actions/effects)

The prioritization methodology and overall results are presented in Appendix A. Below is a short description of each tool.

**Simulation Tools:**
The Sim Builder and Sim Tuner work interactively to allow the construction, test and tuning of systems dynamic models. Both the Sim Builder and Tuner tools have application outside of the Experience Accelerator in the development of system simulation models, particularly those in systems dynamics. The Chart Designer allows an experience developer to specify which output charts will be used for a particular experience.

1. **Sim Builder** – This tool provides the ability for non-technical staff to build systems dynamics models based on existing templates in a GUI environment.

2. **Sim Tuner** – This tool provides the ability to analyze the system, determine the sensitivity of various parameters, and aid in the tuning of the system to achieve desired behaviors.

3. **Chart Designer** – This tool facilitates the specification of output charts from the simulator.

**Experience Building Tools:**
These three tools are used together with Chatmapper¹ to allow non-technical staff the ability to build and modify EA experiences without any programming. Chatmapper is an application that supports dialog creation. In most cases, this work will be accomplished through a GUI and a single workbench.

4. **Phase Editor** – This tool provides the ability to change the finite state machine that changes the phases within an EA experience. For example, the project phases can be customized to new domains and environments and can be constructed to represent state changes that are not affiliated with formal project states.

5. **Event Editor** – This tool provides the capability to create and edit events during an experience and the activities that may trigger them. For example, a phone call from the

¹ www.chatmapper.com
learner’s supervisor can be triggered based on a decision made by the learner or the state of the project.

6. **Artifact Integrator** – This tool allows an experience builder the ability to quickly upload an experience change, be it a new artifact such as a new dialog or report, or a changed phase and or event, and test the results without having to do any programming.

**Learning Analysis Tools:**
This tool has application outside of the Experience Accelerator and can be used as a tool for learning in a number of experiential environments.

7. **Learning Assessor** – This tool will analyze the subject’s activities, decisions, project performance and self-assessments to determine the learning level achieved. This work will involve developing the logging ability to collect the necessary information, and an analysis tool for making the final predictions.

To design and implement the design and development tools for the Systems Engineering Experience Accelerator, work consists of the following major categories:

- Determine requirements for each tool described above through discussions with the user community.
- Review existing tools and technology for potential leverage and reuse.
- Develop and integrate the tools described above.
- Enhance the Experience Accelerator technology as necessary to support these tools.
- Integrate the tools with the Experience Accelerator (e.g., the artifact entry app spans modules including NPC dialog, project background document, simulation models and outputs, etc.).
- Validate the tools efficacy through their use in developing new prototype Experiences.

**1.3.2 Research Approach for this Task**

Traditional systems engineering education is not adequate to meet the emerging challenges faced by the Department of Defense and others in system acquisition and sustainment. New educational technologies such as the Systems Engineering Experience Accelerator hold the promise of facilitating a rapid skill and experience accumulation for the workforce to meet these challenges. However, to have scalable effect, such technologies cannot rely on extensive programming and low-level development to create a rich set of experiential learning modules needed to accelerate systems engineering workforce development.

We are utilizing the following approach to scale the SEEA to include additional experiences.

**Hypothesis:** The Experience Accelerator technology will scale to support a community of developers engaged in creating modules for their organizations’ use if tools are developed that
allow educators and other non-programmers to create, maintain and evolve experiential learning modules.

**Measurable Outcomes:** The outcomes from this research will be measured in two main ways. First, educators and others interested in creating experiential learning modules will provide qualitative feedback on the effectiveness and efficiency of the Experience Accelerator toolset in creating experiential learning modules based on their use of the design and development tools. Second, the number of such developers who commit to create modules for their organizations’ use will be tracked, as well as the number of organizations and variety of different application areas.

**Research Goal:** Validate the hypothesis through the creation of design and development tools for experiential learning modules that maximally leverage the current Experience Accelerator research, technology and content.

**Research Approach:** The tool developments will use an iterative process that will involve some or all of the following research activities in each Part:

**A. Specify:**
1. **Determine project goals & success metrics:** Determine goals and success metrics based on EA research team meetings with sponsors and EA developers based on their future EA design and development needs.

2. **Identify critical use cases:** Identify potential external Experience developers who can be define objectives, use and evaluate the toolset. Create test case scenarios based on Experience design and development objectives from these third parties.

3. **Review existing tools and technology:** Identify relevant existing tools and technologies and determine which to use and/or leverage.

4. **Specify initial tool design:** Specify requirements and features of the each of the desired new tools. Review this with the sponsor and identified external Experience developers.

**B. Develop**
1. **Develop and test tools:** Develop and test tools in an agile development process with period feedback from the potential future users.

2. **Revisit design as needed:** Revisit the design and (re)assign feature priorities based on user input.

3. **Test:** Test the tools by having the potential users create portions of their prototype Experiences per the identified test case scenarios.
C. Evaluate

1. Perform usability analysis: Record and analyze the results of the usage of the tools by the test case users. Determine the strengths and weaknesses of each of the tools.

2. Refine tools and correct defects: Based on the usability analysis, refine the tool set adding feature and correcting defects as required.

3. Write final report: The findings from the above research activities will be summarized in a final report. The final report will describe the additional work that will be necessary to fully deploy the system. In addition, opportunities for expanding the use of the technology will be described for further development. These findings will be presented and reviewed with the sponsor and translated into a plan for the work in following year.

2 Simulation Tools

Part 2 of this research has continued development of the Sim Builder and Sim Tuner tools, and it has initiated design and development of the Chart Designer.

In the Experience Accelerator, simulation models are used to represent the state over time of the system being developed in the experience. They track technical performance measures, key performance parameters and quality. They are also used to represent the state over time of programmatic elements of system development, including cost (Earned Value Management), schedules (milestones such as entrance criteria for review meetings) and staffing (including assignment of technical staff to tasks). The simulator executes a model to advance these states over simulated time via behavioral models and relationships. General phenomena modeled include feedback loops, lags and non-linear behavior. Simulator outputs provide the learner with graphical system/program status information for decision support and status data for NPC dialog support. The simulator is implemented in Java™ and XML using the system dynamics formalism.

Simulation models present substantial development and curation challenges. Typically, both expert domain knowledge and simulation modeling knowledge are required. Thus, it is difficult for a novice to create a mode that would apply to an experience of interest to the novice. Additionally, once a model is developed, it must be validated. In the Experience Accelerator, such validation addresses authentic and compelling experience targets and trades them against development time and effort. This requires experimentation with the model to determine that it behaves as intended and in a realistic manner. These issues have been known in simulation and have remained largely unsolved for many years.
At the same time, simulation models have complex parameter and variable relations, making desired output behavior potentially difficult to achieve. In the currently modeled experience involving development of an unmanned aerial vehicle, for instance, there are 185 variables and 172 parameters (constants). The learner can modify a number of variables as decisions entered into the experience to solve problems. There are complex inter-relations among these variables, and these may be complicated by the learner input.

The intent here is to provide a suite of tools to aid an experience developer, with the following features:

- Main interface from which all tools are accessible
- Simulation model building tool (Sim Builder)
  - Graphical user interface (GUI) support for model development
  - Support for model curation and documentation via libraries, templates and sub-models
- Simulation tuning tool (Sim Tuner)
  - Graphical output displays for parameter tuning to get desired behavior (including difficulty levels for different classes of learners)
  - Graphical output displays to facilitate understanding of learner inputs and their effects on the simulation output
- Output chart specification tool (Chart Designer)
  - GUI support for output chart design

### 2.1 SUCCESS METRICS

The following metrics were used to determine successful design and development of the simulator tools.

- The simulation tools will provide similar functionality to existing commercially available tools, with the understanding that functionality will be prioritized based on importance subject to funding availability. This includes intuitive support for model building constructs:
  - Dynamics,
  - Dependency,
  - Feedback, and
  - Lags.

- The simulation tools will provide methods for the following:
  - Model and sub-model library creation and archiving. A limited library of sub-models will be provided.
  - Experimentation with models to verify and validate behavior.
  - Design of chart outputs.
• The simulation tools will conform to generally accepted usability principles.

• The simulation tools will be designed to facilitate model building by experience developers who are not simulation experts.

• The tools will support model integration with the Experience Accelerator application.

2.2 CRITICAL USE CASES

In analyzing the usage of the simulation tools, we determined that critical use cases fall into the following categories.

1. Conceptual model design and development – Here, the developer is tasked with designing and developing the overall simulation model to be used for a cycle or phase of an experience. This model may have many different components, reflecting technical system behavior, cost and schedule, and workforce activities.

2. Sub-model design and development – The overall model typically can be decomposed into sub-models. This is useful for the sake of modularity and also for facilitating the creation of sub-model libraries that can be used in developing new simulation models.

3. Model element interaction – The simulation tools provide a graphical user interface for model building. The developer thus interacts with different model elements in building a model.

4. Model output specification – The developer is interested in outputs from the model to support learner decisions in the experience. The numeric values of these outputs and their behavior over time are important. The developer also must be able to specify the visualization format to be provided to learners.

Within the category of conceptual model design and development, the following use cases were derived:

1. Designing a new model. The developer starts with a verbal description of the simulated behavior of the program and system under development, as well as the learner decisions that can impact program and system behavior and outcomes.
   a. The developer specifies the different sub-models that will be needed and the relationships between them.
   b. The developer specifies the structure of model elements needed to support each learner decision set.
   c. The developer tests the model, verifies its internal consistency and validates its outcomes against the desired experience/scenario.
2. **Evolving an existing model.** The developer starts with an existing model that is to be used in an experience that is modified in some manner from the original experience for which the model was designed.
   
   a. The developer adds sub-models and/or model elements to an existing model.
   
   b. The developer changes existing sub-models or model elements to make an experience either more difficult or less difficult for a learner.
   
   c. The developer tests the model, verifies its internal consistency and validates its outcomes against the desired experience/scenario.

3. **Deriving a model from an existing model.** The developer starts with an existing model and derives a new model that corresponds to a phase in the experience that either precedes or follows the phase to which the existing model corresponds.
   
   a. The developer determines the common variables, sub-models and other model elements, as well as the new features that the new model must support.
   
   b. The developer implements the new model.
   
   c. The developer tests the model, verifies its internal consistency and validates its outcomes against the desired experience/scenario.

Sub-model design and development yielded the following use cases:

1. **Create a sub-model.** The developer creates a sub-model by adding and linking model elements. Specific model elements subject to learner decisions are developed.

2. **Design sub-model interfaces.** The developer specifies the interfaces between sub-models.

3. **Validate a sub-model.** The developer verifies that a sub-model behaves as intended and validates that it accurately reflects the portion of the experience/scenario being modeled.

4. **Archive a sub-model.** The developer saves a sub-model in an archive where it can be retrieved by other developers who may wish to use it (or a modified version of it) in other experiences.

5. **Tune an existing sub-model.** The developer determines the combination of variables and parameters that yield the detailed values over time for sub-model outputs.
   
   a. The developer considers internal parameters for the sub-model’s endogenous behavior.
   
   b. The developer also considers external relations (i.e., exogenous inputs to the sub-model).
6. **Add an archived sub-model to a new model.** The developer selects an existing sub-model from an archive and inserts it into a model being developed. The developer understands the behavior and function of the sub-model from documentation. The developer modifies its internal workings as needed and links it to sub-models in the current model as appropriate.

The use cases for model element interaction focus heavily on usability.

1. **Detailed model and sub-model creation.** The developer creates a model or sub-model using graphical elements from a menu.
   a. The developer places elements on the screen in the desired location.
   b. The developer connects those elements to represent linkages and relationships.
   c. The developer manipulates those elements to add or modify names, change locations, etc.
   d. The developer marks the model elements subject to learner influence.

2. **Equation specification.** The developer specifies math equations for various model elements (rates and auxiliary functions) to model the dynamic behavior and relationships in the simulated program and system.

3. **Use of standard application functions.** The developer uses standard application functions including:
   a. Save
   b. Edit
   c. Cut-copy-paste
   d. Zoom

Finally, the use cases for model output specification include the following:

1. **Chart design.** The developer uses a graphical user interface to design charts by specifying:
   a. Variables
   b. Plans (i.e., how critical program/system metrics should unfold during the experience)
   c. Units
   d. Chart labels
   e. Time horizon (amount of time shown on the chart)

2. **General output consistency checking.** The developer tests the consistency of the actual output with the desired output of the simulation model. The developer adjusts variable and parameter values as needed to get the desired outcomes.

3. **Learner-influenced output consistency checking.** The developer tests the consistency of the actual learner-influenced output vs. the desired learner-influenced output. The
developer tests various values of learner decisions, taking care to study the interaction
effects from multiple decisions. The developer identifies ranges of valid learner decision
inputs (e.g., new values of simulation variables).

2.3 Requirements

Requirements are divided into three main categories – graphical user interface, sub-models and
parameter tuning. The GUI requirements are the following:

1. There will be a main interface that will provide navigation between the Sim Builder, Sim
Tuner and Chart Designer.
2. The graphical user interface will be similar to Microsoft Visio™.
3. It will support the functionality expected from a graph builder application, including
   a. The ability to create a new graph,
   b. The ability to insert specific nodes,
   c. The ability to specify properties that belong to those nodes,
   d. The ability to zoom the overall graph,
   e. The ability to verify that a graph is correct and to provide feedback if the graph
      has not been created correctly,
   f. Facilities for creating charts from the simulation output, and
   g. The ability to execute all standard program functions (open/save/create new files,
      cut, copy, paste, and changing language)

The sub-model requirements are as follows:

1. The user will be able to section a graph into sub-models.
2. The GUI will show how sub-models are linked.
3. The GUI will allow the manipulation of each sub-model on screen.
4. Sub-models will be able to be saved for use in other models

Finally, requirements related to parameter tuning in the Sim Tuner include the following.

1. The tools will have the ability for the user to change multiple variables and have the
   corresponding graphs automatically updated based on the changes.
2. The graphs must be readable on the screen.
3. The tools will display multiple graphs to illustrate how different variables change over time.

2.4 INTEGRATED INTERFACE

The main interface starts the experience developer in the Sim Builder. It allows navigation to the Sim Tuner and Chart Designer from there. The experience designer must either open an existing model or build a new model using the Sim Builder to be able to use either the Sim Tuner or Chart Designer. Both of these tools require a simulation model as input. Thus, the interface starts in the Sim Builder. The experience designer must save a model before entering either the Sim Tuner or Chart Designer. Once the experience designer is finished either tuning a model or designing its charts, she may return to the Sim Builder. Figure 1 shows the navigation among the tools.

2.5 SIM BUILDER

Part 1 focused primarily on the Sim Builder and implementation of sub-models to allow better management and curation of large-scale models. The Part 2 work extends the sub-model implementation with a variety of shared nodes. A node is shared by reference in that its value is imported into a sub-model from the original node’s home sub-model for use in the new sub-model. In the new sub-model, the shared node can be used as a regular node, but its value cannot be changed. Its value can be changed only in its original sub-model. The following node types can be shared.

- Level nodes (variables or stocks). These variables change over time by various rate functions during a simulation run.
- **Constant nodes.** A constant node contains a value that does not change during a simulation run.

- **Auxiliary node.** An auxiliary node contains an intermediate function for use elsewhere in the model.

This work required modification to the underlying XML syntax to support the shared nodes, as well as the read-write functions for the Sim Builder. In addition, the simulation execution code was modified to support computation of models with shared nodes. Finally, the Sim Builder GUI was enhanced to allow the experience designer to create shared nodes and to represent them on the interface. Figure 2 shows a simple model with two sub-models. The sub-model on the left has a variable node and a constant node. The sub-model on the right has shared nodes for each of these nodes. When the model is executed, the sub-model on the left will modify its variable. The sub-model on the right will import the value of that variable (as well as the value of the constant). The variable’s value will be updated at each time step. Thus, the sub-model on the right will import a new value each time step.

![Figure 4. Sim Builder with shared nodes](image)

2.6 SIM TUNER

At the end of Part 1, the Sim Tuner had capability for only one graph of variables. Clearly, this was a major limitation for a complex model. First, too many variables would be graphed, making the graph unreadable. Second, unrelated variables graphed at different scales would be problematic. This is shown in Figure 3.
The Part 2 effort addressed this limitation by providing multiple graphs to tune different sub-models. Each sub-model has its own graph in the new version of the Sim Tuner. The experience designer can select which variables from the sub-model to include on the graph. Figure 4 shows the execute interface within the Sim Tuner for a model with two sub-models. The experience designer can select the number of time steps and then navigate using tabs to the various sub-model charts to display results via the graphs.

A graph from one of the sub-models is show in Figure 5. These graphs let the experience designer test the behavior of the simulation model at a more detailed level.
In addition to the capability for multiple graphs, with each having variables selected so that the graph is a rational collection of variables, the Sim Tuner now allows the experience designer to change the value of variables and constants in between simulation executions. This allows the designer to experiment in the role of the leaner. In this case, the variables to be changed correspond to those that would be changed by a learner when making recommendations to influence the program. Thus, the experience designer can now test, verify and validate the various recommendations to be made by the learner.

This is enabled by a new attribute for variables and constants – “learner decidable.” When building a model, the experience designer can assign a “true” to this attribute to emulate a variable or constant that the learner can change with a recommendation. Those variables and constants so tagged will show up in the interface via Excel-like text boxes where the experience designer can change their values. For instance, the experience designer can execute the simulation in the Sim Tuner for 30 days. After this execution, she can review the graph of a particular sub-model of concern with selected variables. Those variables and constants that are learner-decidable will have text boxes that can be used by the experience designer to change variable and constant values, emulating learner decisions in changing them for the next execution. This emulates the notion of cycles in the Experience Accelerator, as well as the learner decisions/recommendations between cycles. The interface for a graph with learner-decidable variables and constants is shown in Figure 6. Their text boxes are highlighted in yellow.
As an example, consider the following for the UAV experience. A sudden spike has occurred due to new requirements for electromagnetic interference and compatibility in the actuators. This spike has occurred just prior to the end of a cycle when a learner must examine the program status and make new recommendations. The learner is tasked with deciding on a potential redesign of the actuator system. The learner is reviewing a chart that provides the status and trajectory of weight and weight growth for the airframe and propulsion sub-system. This chart is shown in Figure 7.

The purple lines indicate the potential impacts of decisions on weight. The chart will show the actual impacts after the learner has entered his/her decision. The designer can use the Sim Tuner to experiment interactively with such a chart to see how to model the actuator system to produce the desired effect from potential learner decisions. It should be noted that a multitude of other charts track other technical performance measures. Thus, the learner is actually making a trade
with this decision, since it may improve the weight but degrade other important performance measures, as well as schedule and cost.

2.7 Chart Designer

The Chart Designer is invoked from the common interface, and it accesses variables from the current model that is loaded into the Sim Builder. The experience developer can create any number of charts. The charts will be saved to a chart XML file that corresponds to the model file from which the variables are drawn.

In specifying a chart, the experience developer first specifies the following chart attributes through a series of dialog windows:

- Name (title of chart)
- ID (numeric ID of chart)
- X-axis label
- Y-axis label
- File (graphics file name to be used when chart is generated by Experience Accelerator)
- Global (whether the chart is to graph variables for program life or for current phase in Experience Accelerator)
- PR (which variable is tied to a plan to be used in the chart)

The experience designer then specifies which variables are to be graphed in the chart, along with a label for each.

Each chart file can contain a library of plans. A plan is graphed similar to a variable, and it represents the planned or desired trajectory for that variable. For instance, a range variable may have a plan associated with it. Deviations of the variable (i.e., actual range) from the plan can then be tracked, and the learner can use this information in the Experience Accelerator to make corrections to the technical aspects of the program. Figure X shows a range charts as an example. “Range Reported,” “Threshold/Objective” and “Range Requirement” are variables, and “Range Plan” is the plan associated with “Range Reported.” The other elements of the chart design are displayed, as well (name, x-axis label, etc.).
Each plan in the library can be used in any of the charts. The Chart Designer allows the experience designer to build this library and then to select a particular plan for a chart when that chart is being specified.

The prototype interface for the Chart Designer is shown below. The interface allows editing of multiple charts per chart file.

2.8 SIMULATOR EXECUTION CODE

The Part 2 work resulted in a number of changes to the XML specifications used for simulation model files and chart design files. The code for the tools, obviously, was implemented to use this modified XML specification. The simulation execution code that runs in the Experience Accelerator application was also modified to use the new XML specification. This was facilitated by a common code base for much of the simulation execution code and Sim Tuner tool.
2.9 Evaluation

Evaluation of the Sim Builder, Sim Tuner and Chart Designer has been conducted internally and has identified a number of issues that were addressed. Unit testing was performed on individual functions, and then integration testing was done for the overall tools individually as well as collectively (i.e., navigation between tools using the common interface). In addition, sub-models for range and earned value management were constructed to test the Sim Builder and Sim Tuner.

Future evaluation work will test usability and effectiveness of the tools as potential experience developers use them.

3 Experience Building Tools

Part 2 of this research has continued development of the Experience Editor, with the inclusion of Phase Editor, Event Editor and Artifact Integrator. New functionalities have been added to the toolset. An integrated toolset was developed to combine the three individual tools into a comprehensive toolset for improved usability and code management. Furthermore, help documentation such as user guide was created to provide experience developers an easier hands-on experience.

3.1 Tools Overview

To provide a viable solution for the experience designer, the tools need to work together. SEEA experience construction requires implementation of three essential aspects of the experience, experience flow, dynamic experience components, and contents and experience integration as shown in Figure 12.

![Figure 12 Overview of Tools and Aspects of the EA Experience](image)

Experience flow is the part of experience design which dictates the sequence of the phase, sub-phase and cycles. By providing logical and realistic experience flow, experience designer can
create an experience that simulates a real world scenario. Dynamic experience components are contents that provide the learner unexpected happenings throughout the experience either through the use of scripted events or the outputs from the simulation engine. Dynamic experience components are the key for learning evaluation in which the learners improve through the experiential learning instead of recognizing patterns of the system. Contents and experience integration represents the learner accessible resources and how they are represented throughout the experience. By opening or closing learners’ access to time sensitive materials, experience designer can create a realistic and deeply customized experience.

Experience building tools including the Phase Editor, Event Editor and Artifact Integrator are focusing on implementation of the experience design. Simulation tools including the Sim Builder, Sim Tuner and Chart Designer are designed to help with simulation engine implementation. Phase Editor allows experience designers to customize the experience flow. Event Editor provides experience designer the ability to create scripted events. Sim Builder, Sim Tuner and Chart Designer helps experience designer to create, tune and customize a simulation engine for the experience. And Artifact Integrator supports the integration of new materials to an experience with permission control and content creation functionalities.

### 3.2 INTEGRATED TOOLSET

During part 2 of this research, an integrated toolset has been developed to reduce the code load of the tools, increase the performance, improve the usability and to provide the easy addition of features in the future. The integrated toolset is shown in Figure 13.

![Figure 13 User Interface of Experience Editor](image-url)
3.3 FUNCTIONALITIES

Experience building tools provide multiple options for experience designers to alter different aspects of the experience. The tools utilize the SEEA architecture design and allows experience designers to change the built-in experience. They also provide functionalities to create new experience from scratch.

The target user of the experience building tools are educators and experience designers without significant knowledge of the SEEA design. There is no requirement for them to have programming skills. Therefore, the experience building tools utilized user friendly design with graphical user interface and simple actions like drag and drop.

**Phase Editor** – This tool provides the ability to change the finite state machine that changes the phases within a SEEA experience. For example, the project phases can be customized to new domains and environments and can be constructed to represent state changes that are not affiliated with formal project states. For existing experience modification, the Phase Editor can be used to change the available events, available NPCs and number of cycles for a specific phase. Figure 14 shows the graphical user interface of the Phase Editor.

Phase Editor presents the experience designer a canvas with support of drag and drop to easily create experience flows with phase and sub-phases. For each phase and sub-phase, experience designer can setup name, starting time, available NPCs, events that may be triggered and number of cycles learner will go through. By connecting phases and sub-phases on the canvas, experience designer could create a phase sequence for an experience which can be saved and exported for later use.

Phase Editor also supports modification of existing phase/sub-phase. Importing an XML formatted Phase Data File will bring in the configuration of an existing phase with name, time, NPC and events data for modification.

![Figure 14 Phase Editor User Interface](image-url)
**Event Editor** – This tool provides the capability to create and edit events during an experience and the activities that may trigger them. For example, a phone call from the learner’s supervisor can be triggered based on a decision made by the learner or the state of the project. Figure 15 shows the screenshot of Event Editor.

By using the Event Editor, experience designer can create/modify events using the condition and action panel. Inside the condition panel, event ID, condition type and properties related to the selected event type are needed. Condition type indicates the type of the triggers that will be fired. The action panel provides the options of action type, action properties.

![Figure 15 Event Editor User Interface](image)

**Artifact Integrator** – This tool allows an experience designer the ability to quickly upload an experience change, be it a new artifact such as a document, report, or a change phase and or event, and test the results without having to do any programming. Figure 16 shows the graphic user interface of the Artifact Integrator.

The functions of Artifact Integrator are separated into four groups as follows:

- **Learner File System by Phase**
  This function allows experience designer to set the availabilities of textual and graphical materials by phase. It is also possible to create permission settings that grants learners the access to certain materials only when certain criteria are met.

- **Emails**
  Artifact Integrator provides functions to create, edit and remove emails in a SEEA experience. Experience designers can create new emails by adding email ID, sender/contact, subject and email body. Email body supports the use of variable which allows the creation of dynamic content based on experience status.
Voicemails
Voicemail creation is similar to the emails. However, voicemails can only be sent by an NPC character where emails can be sent by other learners as well.

PDF File Conversion and Integration
Since the technology used for client-side graphical user interface does not directly support PDF formatted files, Artifact Integrator embedded PDF file conversion function to reduce the efforts needed to add new learner accessible materials to the experience.

CDR Criteria
CDR criteria editor provide the capability to change difficulties of the UAV experience. The CDR Criteria editor user interface is shown in Figure 17.
3.4 Evaluation

3.4.1 Internal Evaluation

Evaluation of the Experience Editor Toolset has been conducted internally and has identified a number of issues that were addressed. Unit testing was performed on individual functions, and then integration testing was done for the overall tools individually as well as collectively.

Future evaluation work will test usability and effectiveness of the tools as potential experience developers use them.

3.4.2 External Evaluation

To evaluate the capabilities of the tools, a scenario for a completely new experience is developed in collaboration with a team of UK researchers. The new experience focuses on a submarine maintenance scenario where the learner is responsible for finding issues with the submarine by communicating with other personnel in the team. Failing to communicate with peers in a timely fashion results in a catastrophic outcome. Since this experience design does not require a simulation engine, therefore experience creation requires only the experience building tools. Although the experience creation is still a work-in-progress, it demonstrates how the tools will be used to help the creation.

The experience is divided into four phases:

- Setup phase – Welcoming the learner and provides the learner background information about the situation.
- Diagnosis Phase – Learner investigates the potential issues by communicating with peers in a timely and appropriate manner.
- Scheduling phase – Providing learner the option to schedule maintenance period for the submarines.
- Debriefing Phase – Depending on the learner action, different situation will happen and learner receives feedbacks about his/her experience performance.

For the setup phase, an email is sent to the learner requiring him/her to read through five technical documents and proceed. Artifact Integrator is used to convert and integrate the PDF formatted documents. The email is composed in the Artifact Integrator and then referenced from Event Editor where the email event was created. One limitation of the tools is that they currently do not support creation of user interface elements, therefore some coding efforts will be needed for creating a proceed button on the user interface (UI).

For the diagnosis phase, Chat Mapper is used to create phone call dialogs between the learner and NPC. In addition to the phone calls, office meetings are also needed for this experience. In the current state of SEEA, no such function is implemented. The underlying infrastructure uses the same parsing engine for phone call dialogs, but new UI functionality is needed to make such conversations possible.
The creation of the scheduling phase involves composing emails, creating decision making forms and a task schedule panel. Emails can be created using Artifact Integrator together with the Event Editor. However, decision making forms and the task schedule panel are UI elements which require some coding efforts to create the UI needed for this phase.

The debriefing phase requires the use of Artifact Integrator and Event Editor to create feedback messages based on user performance.

After the creation of events and artifacts, Phase Editor will be used to connect the phases together and assign events and available NPCs. Phase Editor allows experience designer to configure each phase and connect them together using the canvas and drag-and-drop composing.

Through the creation of a completely new experience, several limitations of the tools are discovered. Creating a new experience usually involves new user interface design. Current SEEA infrastructure still lacks the foundation for easy UI creation with tools. A major improvement of SEEA will likely involves an infrastructure update to enable dynamic UI loading. At the same time, a new tool will be needed for easy code-free UI creation. There are still challenges with integrating the Chat Mapper dialogues with the simulation, but work is ongoing to simplify and streamline the SEEA/Chat Mapper interface and reduce the duplication of functionality.

4 Learning Assessment Tool Prototype

The accurate assessment of learning is a critical element of the Experience Accelerator’s capabilities. To date, much of the focus has been in developing an experience that has the potential to facilitate effective learning. The evolution of this experience has been directed by feedback from subject domain experts, informal and formal evaluations as described below. However, the Experience Accelerator (EA) has now reached the point of maturity at which more precise tools must be developed to measure learning.

During part 2, researches have been conducted on learning assessment techniques and tools. Through the research, a high level design of learning assessor tool, specification of learning analysis tools and preliminary prototype have been developed. This section describes the work completed to date, and the current concept for learning assessment tools development.

The Learning Assessment Tools measure the efficacy of the experiences by analyzing the data recorded throughout the learners’ participation. Traditionally, learning assessment has been done through examinations and experts’ reviews and opinions on students’ work. However, most approaches emphasize comparing learners’ performance against those of the experts’ and less about the evaluating the actual learning performance of individuals. There has been much research in the domain of systems engineering education attempting to find the best way to assess students’ understanding and learning about systems engineering. Though simulation has been widely adapted by systems engineering learning, it has yet to be used to assess learner competencies and learnings performance in systems engineering and technical leadership learning.

Learning Assessor will analyze the subject’s activities, decisions, project performance and self-assessments to determine the learning level achieved. This work involves developing the logging
ability to collect the necessary information, and an analysis tool for perform the final analysis. The SEEA infrastructure has been updated to perform the data logging and collection task. Learner Assessor is currently under development, the prototype is capable of importing the recorded learner data and perform visualization tasks to help experience designers and instructors to understand learners’ performance. The next phase of the prototype development will involve developing a performance assessment engine, which evaluates learners’ competency by comparing their performance in the experience to an experts’ one. And by comparing a learner’s performance data with his/her history data, the tool can also measure the efficacy of the current experience thus helps experience designers to improve the experience design. The design of Learning Assessor has application outside of the SEEA where it can be used as a tool for assessing learning in a number of experiential environments.

4.1 ASSESSMENT OF SYSTEMS THINKING

Systems thinking is a set of synergistic analytic skills used to improve the capability of identifying and understanding systems, predicting their behaviors, and devising modifications to them in order to produce desired effects. These skills work together as a system, feeding into and depending upon one other to produce a greater net effect. Of the various systems engineering skills, systems thinking is a skill that has its own breadth and depth, reaching beyond the boundaries of Systems Engineering and across many other fields. This scope results in additional assessment challenges.

Measuring systemic thinking and knowledge structure is often considered a difficult task. Systems thinking is an abstract concept, and represents a way of thinking rather than a specific set of acquired knowledge. Though tests to measure abstract concepts have been devised with varying levels of success, this type of measurement more closely resembles the demonstration of a skill rather a test of knowledge. At the core of the challenge lies the fact that systems thinking is a bundle of abilities that cannot be described and investigated like a single ability.

Research is underway to measure systems thinking ability, also called Systems Literacy or Systems Thinking Maturity, by applying an appropriately designed set of assessment tools, including one or multiple systems simulations. Using simulation, a system can theoretically be simplified without losing its essence as a system. A sufficient simulation could also be designed in a way that is meaningful to educators outside of the system dynamics community. This is a highly desired quality in systems thinking assessment.

As part of the research to assess systems thinking ability, a standardized definition of systems thinking has been proposed and a set of systems thinking skills has been devised. The development of a set of assessment tools is underway. The tool set will likely consist of two simulations. The first will be an adaptation of a sub-Saharan African scenario utilized in several systems thinking educational efforts. The second will be a context-free simulation designed to capture the essence of systems thinking apart from the knowledge of any particular domain. Together, these two simulations are intended to cover the entirety of the systems thinking skillset. The simulations will be used by experts in the systems field in order to gather their usage patterns. These patterns will then be analyzed and used as an initial standard by which to assess systems thinking. This method of recording expert usage patterns of specially designed systems simulations will advance the state of the art in this field.
The results of the systems thinking assessment research will be included in future SE educational programs. In particular, the context-free simulation will be added to the SEEA as a means through which to assess the improvement of SEEA learners’ systems thinking abilities.

### 4.2 ASSESSMENT OF SYSTEMS ENGINEERING

To accurately assess systems engineering learning performance, assessment of systems engineering (SE) competencies is crucial. For experiential and accelerated learning of systems engineering, assessment of SE competencies provides learners, instructors and experience designer insights about the capability of the learner, the performance of the experience and the improvements can be done for learners to approaching industry professional level.

Existing assessment techniques and methodologies not only heavily rely on written statements and recommendations which are subjective, but they also lack the ability to accurately assess and differentiate candidates who don’t have significant industry experiences. Traditional techniques to assess competencies of systems engineering does not provide a solution to accurately assess learners who learned through experiential and accelerated learning experiences. This ongoing research focuses on incorporating a new systems engineering assessment model which doesn’t rely on industry experience, together with learners’ performance in experiential learning experience to assess learners’ systems engineering competencies.

### 4.3 LEARNING ASSESSOR PROTOTYPE

During part 2 of this research, a proof of concept Learning Assessor prototype has been developed. The following is a list of the top priority tool features that are completed within the part 2 work period.

1. Gather experience data and process to database

   The learning assessor tool will read experience data form the EA logs and process it into the database.

2. Visualize project status data

   The learning assessor tool will display the project status data from the learners’ experiences in the form of line chart. Figure 18 shows the user interface of Learning Assessor with visualized project status data.
3. Visualize learners’ recommendation data

The learning assessor tool will display the learners’ recommendation data from the experiences in the form of highlighted text section and bar chart. Figure 19 shows the user interface of Learning Assessor with visualized recommendation data.

4. Display the user actions and general performance score

The learning assessor tool will display the learners’ actions through the experience, indicates if the experience is completed and display the general score.

5. Provide visual comparison between two learning experiences

The learning assessor tool will provide visual comparison between two learning experiences. These experiences can be from the same learner for analysis of performance improvements, they can be from two experiences of two different learners for a horizontal comparison. This feature can also be used to compare one learner’s experience with one from the experts. The interface of visual comparison is shown in Figure 20.
4.4 EVALUATION

Evaluation of the Learning Assessor prototype has been conducted internally and has identified a number of issues that were addressed. Unit testing was performed on individual functions, and then integration testing was done for the overall tool.

5 INFRASTRUCTURE MODIFICATION FOR SUPPORTING TOOLS

To make code-free experience creation and learning assessment possible, the infrastructure of SEEA had been modified and updated to accommodate the new requirement. During the initial implementation, the SEEA was developed tailoring the UAV experience thus lacking the flexibility needed for customization and new experience creation. The infrastructure modification efforts updated the experience master code base with changes that allowed the customization of experience flow, challenges design and difficulty control.

Data recording and events logging are the most important features needed for learning assessment since they provide invaluable data necessary for analyzing learners’ performance. Former implementation of the SEEA infrastructure did not support data recording as well as customization of the feedbacks provided to the learners. The updated SEEA infrastructure will record the following experience data during learners’ participation in the experience:
• Participant Identification:
  o Experience information
  o Learner’s information including name and email address

• Experience Session Information:
  o Number of past attempts at the save experience
  o Number of “resets” and phase/sub-phase/cycle of “resets”
  o Date/time of experience start and end

• Learner Experience Input & Actions:
  o Learner’s Self-assessment data
  o Learner decision making in the form of recommendations inputs

• Simulation Output and Project Status:
  o Last phase/sub-phase/cycle completed
  o Results of simulation outputs from each phase/sub-phase/cycle
  o Project Status

6 CONCLUSIONS AND FUTURE RESEARCH

This section summarizes the results of Part 2 and describes the future work in Part 3 of the EA Tools Development program.

6.1 CONCLUSIONS

This report has presented results from the first part of a three-part project to create tools to support design, development and measurement of learning of experiences in the Experience Accelerator as shown in Table 5. Each Part builds on the successful completion of its previous part. The tools development efforts fall into four major categories – simulation tools for building and testing simulation models that mimic the behavior and results of acquisition programs that focus on system design and development, experience building tools that provide the structure for such system engineering experiences and the events that occur in them, learning assessment tools to measure the efficacy of the experience and EA infrastructure changes to support this work.

Table 2. Tool development in three parts

<table>
<thead>
<tr>
<th>Time</th>
<th>Simulation Tools</th>
<th>Experience Builder Tools</th>
<th>Learning Analysis Tools</th>
<th>EA Infrastructure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part 1</td>
<td>Develop prototype Sim Builder and Tuner tools</td>
<td>Develop prototype Phase and Event Editor, and/or Artifact Integrator tools</td>
<td>Research and create high level design of Learning Assessor tool</td>
<td>Update EA infrastructure to support tools</td>
</tr>
<tr>
<td>Part 2</td>
<td>Extended functionality of Sim Builder and Tuner tools, make available to</td>
<td>Refine Phase and Event Editor, and/or Artifact Integrator tools, make</td>
<td>Develop prototype Learning Assessor prototype</td>
<td>Update EA infrastructure</td>
</tr>
</tbody>
</table>
broader community. Determine new functionality as needed
available to broader community. Open source. Determine new functionality as needed
to support tools

| Part 3       | Extend functionality of Sim Builder and Tuner tools, make available open source. Determine new functionality as needed. Support existing tools, Specify new tools | Extend functionality of tools, make available open source. Support existing tools, Specify new related tools | Develop Learning Assessor tool, make available to for external evaluation. Specify future functionality | Update EA infrastructure to support tools |

An iterative, agile approach was used for the Part 1 work as described earlier in this report. Each of the tools required different amounts of time and effort for each of these phases depending upon their maturity. For example, the Experience Building Tools required the shortest amount of time before they can be specified, as there have already been some preliminary investigations for their use. The Learning Analysis tools required substantially more research in their specification such that their implementation was not expected to take place during this first phase of the EA tools program. The overall results, as anticipated, are prototypes that require various levels of additional development and engineering for deployment. The status of each of these developments is described below.

The simulation tool development project achieved its Part 1 objectives with the completion of the following:

- Requirements and use cases for simulation tools
- Review of existing tools to leverage
- Prototype Sim Builder with GUI for model building and capability to manipulate sub-models for purposes of modularity and model archiving/curation
- Prototype Sim Tuner that allows testing of model behavior drilled down to variables of interest

The Experience building tools development achieved success in its Part 1 objectives, including the development of the Artifact Integrator, which was an optional item, with the following items:

- Requirements and use cases for Phase Editor, Event Editor and Artifact Integrator
- Review of existing tools to leverage
- Prototype Phase Editor with GUI for phase property and sequence editing
- Prototype Event Editor with GUI for event property editing
- Prototype Artifact Integrator for integrating files created with Phase Editor and Event Editor into the Experience Accelerator

Progress was achieved in the learning tools area as noted below:

- Literature review
- Initial concept exploration
- Captured data identification
However, the infrastructure development work necessary to support the Experience building tools diverted resources from this effort.

The following work was completed in the infrastructure development:

- Review of current infrastructure design
- Server-side implementation updated to support Phase Editor
- Review and evaluate client-side implementation for Artifact Integrator support

The risk identification and mitigation strategy noted below was largely effective in addressing the program risks.

- **Risk**: Ensuring that there is graduate student staff with the knowledge and capabilities required for effective, efficient tool development despite gaps in funding.
  - **Mitigation strategy**: Keep current graduate students engaged as much as possible with stopgap funding from other sources. Determine other sources of students or software research developers.

- **Risk**: Rapidly determining the success metrics and requirements for Experience Design and Development Tools, given the diverse set of users and user needs.
  - **Mitigation strategy**: Engage a representative set of users early and work to ensure proper alignment.

- **Risk**: Providing design and development tools that meet the needs of a set of educators and developers with potentially diverse application areas of interest.
  - **Mitigation strategy**: Examine application areas and needs of user base early and identify generic tool features to support multiple applications/needs. Eliminate “outlier” application areas of necessary.

- **Risk**: The design and development tools in some case span different modules of the EA technology breakdown structure, thus necessitating multi-perspective development.
  - **Mitigation strategy**: Identify dependencies and engage team members to integrate tools across modules as appropriate.

- **Risk**: The likelihood of feature creep that prevents the completion of an adequate set of tools for to provide a complete environment for Experience creation.
  - **Mitigation strategy**: Utilize agile develop practices to ensure that the highest value features are being developed at all times.

A successful demonstration of the tools has been given to the sponsor. The tools are now at the stage where they are ready to be evaluated by external users for their use in Experience and Simulation development. The iterative approach was quite successful at providing incremental functionality that was reviewed with its potential users throughout the development effort, prioritizing the most important features and delivering working prototypes throughout the effort.
This approach will be continued in Part 2. It is also noted that updating the EA infrastructure to remove its hard coded limitations required more work than was initially anticipated.

### 6.2 FUTURE RESEARCH

Part 3 of the EA Tools Development program will build upon the success of Part 2. This phase will provide the ability for non-EA researchers to begin to use the tools and provide feedback that will be used to prioritize the development. The following is a brief description of the future work in each of the four major tools areas.

Future research for the simulation tools includes the following items:

- **Testing**
  - DAU personnel
  - Academic partners (GTRI, NPS, UT Arlington)
  - System Dynamics Society
- **Sim Builder**
  - Populate sub-model library
  - Add story documentation to sub-models
- **Sim Tuner**
  - Intelligence for changing values of variables and effects elsewhere
  - Making effects of changes obvious to experience developer
- **Chart Designer**
  - Improve first-generation interface
  - Add color palettes for charts
  - Modify XML syntax
- **Transition tools for use**

Future work for the Experience development tools includes the following items:

- **Testing**
  - DAU personnel
  - Academic partners (UAH)
  - Make available to broader community
- **Experience Editor**
  - Improve usability with interface improvements
  - Add and refine functionalities based on reviewer feedback
    - Variable Editor
    - NPC Editor
  - Prepare tools for the HTML5 migration of EA

Future research for the Learning Assessment tools includes the following items:

- **Testing**
  - Through pilot program
  - Academic partners (UAH, Stevens)
  - Make available to broader community
• Learning Assessor
  o Improve the user interface
  o Complete the development of learning assessor from prototype to production

Additional development work is necessary in the EA infrastructure.
• Update
  o Update the client and server infrastructure to enable the integration of tools and EA.
  o Update the infrastructure for tools and HTML5 support

Table 3 shows the Part 3 work plan for each of the three tool sets and infrastructure. This has been updated from the original RT-146 proposal based on results and findings of Part 2. As in Part 2, an iterative, agile approach will be used in the development, so that the results will be time-boxed to complete with the Part 3 timeframe. However, the specific features and functions have not been determined for these deliverables, as they will be determined in the “specify” phase outlined below. The completion of Part 3 will expand the research conducted in Part 2, while continuing to address the proposed research objectives described above.

Table 3. Part 3 work plan

<table>
<thead>
<tr>
<th>Phase</th>
<th>Simulation Tools</th>
<th>Experience Builder Tools</th>
<th>Learning Analysis Tools</th>
<th>EA Infrastructure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify</td>
<td>25%: Specify sub-model library components to be developed; specify sim tuner feature details; specify chart designer feature details</td>
<td>25%: Update specification of the Experience Builder based on reviewer feedback and pilot usage</td>
<td>25%: Update specification of learning analysis tools; based on reviewer feedback and pilot usage</td>
<td>30% Review needs for HTML5 support and other infrastructure to support tools. Create specification for longer term infrastructure</td>
</tr>
<tr>
<td>Develop</td>
<td>65%: Refine Sim Builder and Tuner tools; enhance chart designer prototype; complete tool prototype. Make available as open source.</td>
<td>65%: Refine Experience Builder tools, prepare documentation and make available to broader community for review</td>
<td>65%: Continue development of Learning Analysis tools, enhance functionalities and enhance GUI design with more variants of data visualization</td>
<td>60% Update infrastructure as necessary for tools and HTML5 support.</td>
</tr>
<tr>
<td>Evaluate</td>
<td>10%: The developed tool(s) will be evaluated iteratively through its development culminating in this Part 3 evaluation.</td>
<td>10%: The developed tool(s) will be evaluated iteratively through its development culminating in this final Part 3 evaluation.</td>
<td>10%: The developed tool(s) will be evaluated iteratively through its development culminating in this final Part 3 evaluation.</td>
<td>10%: Evaluate results through tool evaluations</td>
</tr>
</tbody>
</table>
References

Experience Accelerator Research Publications


Journal Articles and Reports