New Orleans JAZZED about Engineering Education

Paper ID #17230

The Experience Accelerator: Tools for Development and Learning Assessment

Peizhu Zhang, Stevens Institute of Technology

Peizhu Zhang is currently a PhD student in Systems Engineering at Stevens Institute of Technology, having earned a master's degree in Computer Science there in July 2012. His research interest includes systems engineering, competency assessment, software engineering and serious games.

Dr. Douglas A. Bodner, Georgia Institute of Technology

Douglas A. Bodner is a principal research engineer in the Tennenbaum Institute at the Georgia Instituteof Technology. His research focuses on computational analysis and decision support for design, operation and transformation of enterprise systems. His work has spanned a number of industries, including aerospace and defense, automotive, electronics, energy, health care, paper and pulp, semiconductors and telecommunications. Dr. Bodner is a senior member of the Institute of Electrical and Electronics Engineers (IEEE) and the Institute of Industrial Engineers (IIE), and a member of the American Society for Engineering Education (ASEE) and the Institute for Operations Research and Management Science (INFORMS). He is a registered professional engineer in the State of Georgia.

Dr. Richard Glenn Turner, Stevens Institute of Technology

Dr. Richard Turner has forty years of experience in systems, software and acquisition engineering in both private and public sectors. Currently a Distinguished Service Professor and a Principle Investigator for the Systems Engineering Research Center at the Stevens Institute of Technology in Hoboken, New Jersey, Dr. Turner is active in the agile, lean and kanban communities and was a core team author of the IEEE Computer Society/PMI Software Extension for the Guide to the PMBOK. Dr. Turner's current research includes using kanban and service concepts to transform systems engineering and applying lean and complexity concepts to critical system development and acquisition. He is a Golden Core awardee of the IEEE Computer Society, a fellow of the Lean Systems Society, a Senior Member of the IEEE, and co-author of four books: The Incremental Commitment Spiral Model: Principles and Practices for Successful Systems and Software, Balancing Agility and Discipline: A Guide for the Perplexed, CMMI Survival Guide: Just Enough Process Improvement, and CMMI Distilled.

Mr. Ross David Arnold, Stevens Institute of Technology

Mr. Arnold has over 12 years of experience in software development, systems engineering, and technical leadership for the defense industry. He has independently developed and licensed a variety of software products, as well as published over 30 technical papers focused primarily on fire control and mission command systems. Mr. Arnold holds a B.S. in Computer Science from Rutgers University as well as an M.S. in Software Engineering from Stevens Institute of Technology. He is currently a PhD Candidate in Systems Engineering at Stevens, where his research focuses on systems thinking and its assessment.

Prof. Jon Patrick Wade, Stevens Institute of Technology (School of Systems & Enterprises)

Jon Wade is a Distinguished Research Professor in the School of Systems and Enterprises at the Stevens Institute of Technology and currently serves as the Director of the Systems and Software Division and Chief Technology Officer for the Systems Engineering Research Center (SERC) where he is leading research in the use of technology in systems engineering education and complex systems. Previously, Dr. Wade was the Executive Vice President of Engineering at International Game Technology where he managed corporate wide research and development. Dr. Wade spent ten years at Sun Microsystems during which time he managed the development of Enterprise Servers. Prior to this, he led advanced development of supercomputer systems at Thinking Machines Corporation. Dr. Wade received his SB, SM, EE and PhD degrees in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology.

The Experience Accelerator:
Tools for Development and Learning Assessment

abstract

The Systems Engineering Experience Accelerator is a new approach to developing the systems engineering and technical leadership workforce, aimed at accelerating experience assimilation through immersive, simulated learning situations where learners solve realistic problems. A prototype technology infrastructure and experience content has been developed, piloted, and evaluated. While the prototype has proved useful, its ability to support a community of educators and developers is limited by the challenges in creating or changing experiences. This paper discusses a set of tools built to reduce those challenges and enable educators to more easily design, develop, and share experiences. The tools fall into three major categories – *simulation tools* for building and testing the experience environment models, *experience building tools* that support defining effective learning scenarios, learner interactions and events, and *learning assessment tools* to measure the efficacy of the experience. The authors describe the capabilities of the tools and provide an evaluation of their capabilities based on the update of an existing experience, the development of new educational experiences, and the application to learning assessment in a class environment.

1 introduction

Systems engineering and technical leadership (SETL) is a multidisciplinary practice that is as much an art as a science. While a traditional model of education can teach the fundamental body of 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. Systems engineering educators are struggling to meet the growing educational demands for a workforce able to solve problems driven by accelerating technology, rapidly evolving needs, and increasing systems complexity [1-3]. At the same time, there is a widening gap in industry between the need and the availability of systems engineering practitioners with the necessary experience to address these challenges [4].

The Systems Engineering Experience Accelerator (SEEA) project was designed as a response to these critical needs and challenges [5]. The project goals are to:

- assess the feasibility of an immersive, simulated learning approach for accelerating systems engineering competency development
- validate the ability of such an environment to create an experiential, emotional state in the learner
- determine if such an environment, coupled with reflective learning, effectively compresses learning time.

If the above are true, then the SEEA could significantly increase the experiential resources available to a systems engineer (SE) over time, and provide assimilation of the experiences at a higher rate as compared what would occur naturally on the job.

Figure 1 shows how the various concepts developed for the SEEA are related.

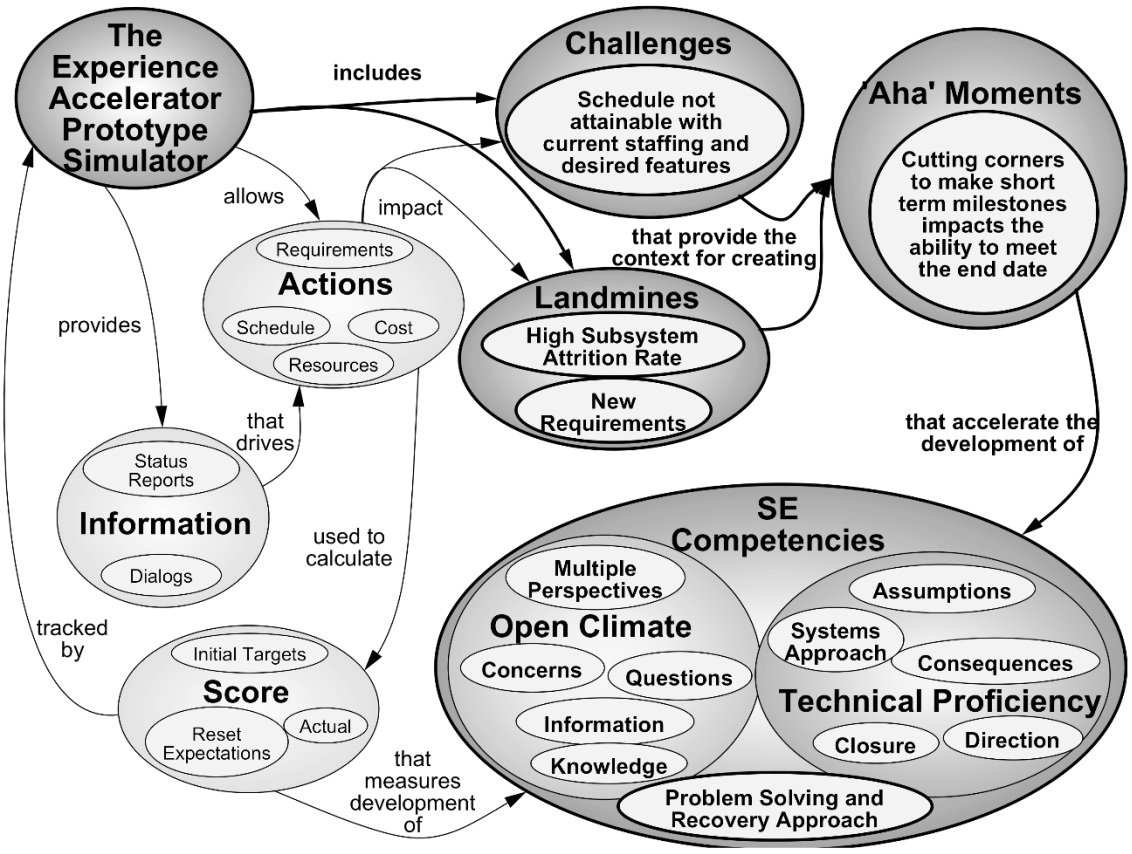

Figure 1: Systemigram of the concepts involved in the SEEA Experience [6, 7]

As shown, the development team had a threefold challenge to balance the development of simulator technology supporting displayed content (shown in light grey), the technology that supports the developed experiential learning concepts (shown in dark grey) and the creation of the actual artifacts and information that make up the experience (shown in white). The goal is to effectively create challenges and landmines that support the learner's experience of the target "Aha" moment. By experiencing the "Aha" moment, the learner transitions to a more advanced level of competency—in this case "Problem Solving and Recovery Approach".

The initial SEEA experience 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 Department of Defense (DoD) program developing a new unmanned aerial system. The exercise is based on the notion of experiential learning, and thus is referenced as an experiential learning module. The learner engages with the experience (i.e., simulated world), makes decisions to solve problems, sees the 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.

Applied in an academic setting, the SEEA concept provides the possibility for a much broader scope of learning environments than a capstone project or industry internship. These more traditional approaches provide a beneficial learning experience and support integrating the various components of the SE body of knowledge, but are limited by time and domain. The capstone is usually a single project and at most a year in length. If it covers the full lifecycle, then it must be a fairly simple project and most likely represents only one domain. An internship is even more limited, given that few companies would assign a student to a significant role or provide much variation of role or domain. The SEEA envisions the ability to provide learning experiences that involve significant decision making at various levels of authority and drawn from many different domains. Neither a capstone or internship could likely present the same range of specific challenges and "aha" moments that the SEEA can provide. Whether or not the SEEA experience is as effective as a truly *in vivo* experience is part of the research underway, with the results from academic and industrial pilots of the SEEA as the primary means of validating effectiveness.

While the existing technology infrastructure and experience content of the SEEA is useful, it is limited in its ability to support a community of educators and developers. Currently, it is difficult to design and develop educational content without significant knowledge of the SEEA design. The SEEA was developed with a goal of transitioning to an open-source sustainment model which will provide long-term support for a community of educators and learners in creating experiential learning modules to address their specific needs. The tools described in this paper are designed to greatly reduce the effort and skill necessary to create and tailor experiences, thus expanding that potential user community.

2 background

2.1 current SEEA architecture and implementation

The SEEA consists of the following major components as shown in Figure 2: [8]

- Experience Master – Maintains current value of state variables related to experience 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.

- 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.

- Non-player Character Engine – Provides a number of non-player characters (NPCs) and associated dialog sessions that reflect changing state of the experience. The non-player characters have roles (e.g., program manager), and the learner is tasked with querying them about various aspects of the simulated experience performance to determine the true state of the program and its underlying problems.

- Simulation Engine – Maintains the detailed state of the experience and advances this state via simulation models, incorporating learner decisions about the experience. 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.).

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

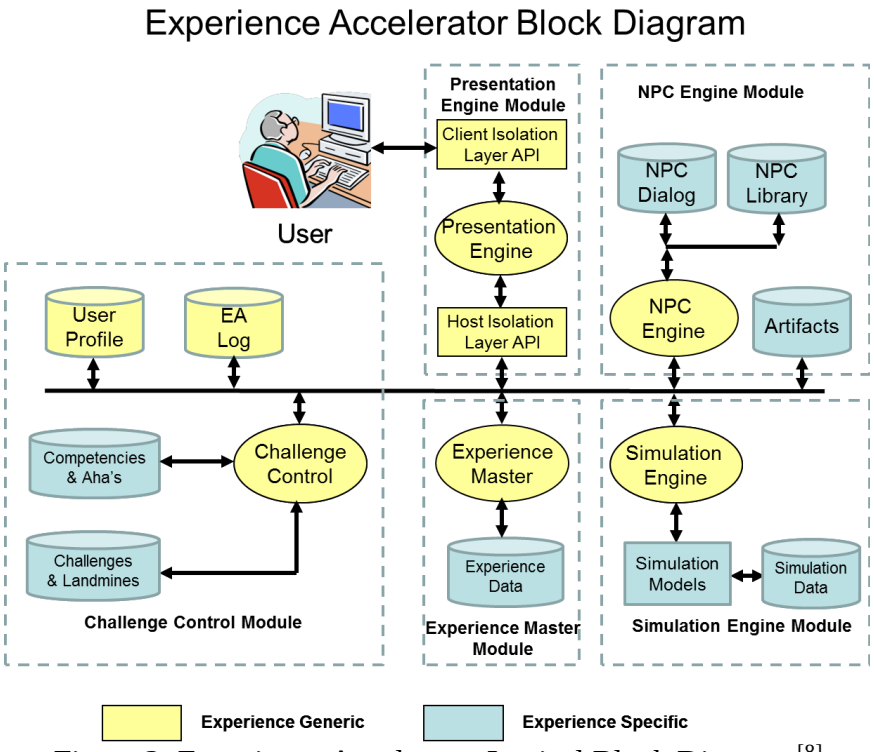

Figure 2: Experience Accelerator Logical Block Diagram [8]

A single internal interface "bus" is provided to share information between these major modules. Isolation layers are provided around the Presentation Engine to facilitate support of changing engine technologies. Finally, Experience generic technology, which can be used to support multiple domains and experiences, and experience-specific blocks are segregated to support an open architecture with the intent to maximize reuse. Except for the simulation models, all of the Experience Specific information is stored in databases within the system. All interactions with the learner take place via the Client Isolation Layer API contained in the Presentation Engine module. The Simulation Engine determines the program state information that will be presented to the learner, and the NPC Engine formats and stores the information and processes the transactions with NPCs, while the Presentation Engine creates the appropriate look and feel for the interaction.

As currently implemented, the Experience Accelerator is a thin client based system that can be supported with a multi-threaded server as shown in Figure 3. A thread is generated every time a learner logs on to the Experience Accelerator program (i.e., when a client connects to the server over a socket); this thread runs until the learner logs out. All threads run in parallel on the server. For the multi-user mode, the server enables intra-thread communication. The client sends the server codes for authentication and file transfers.

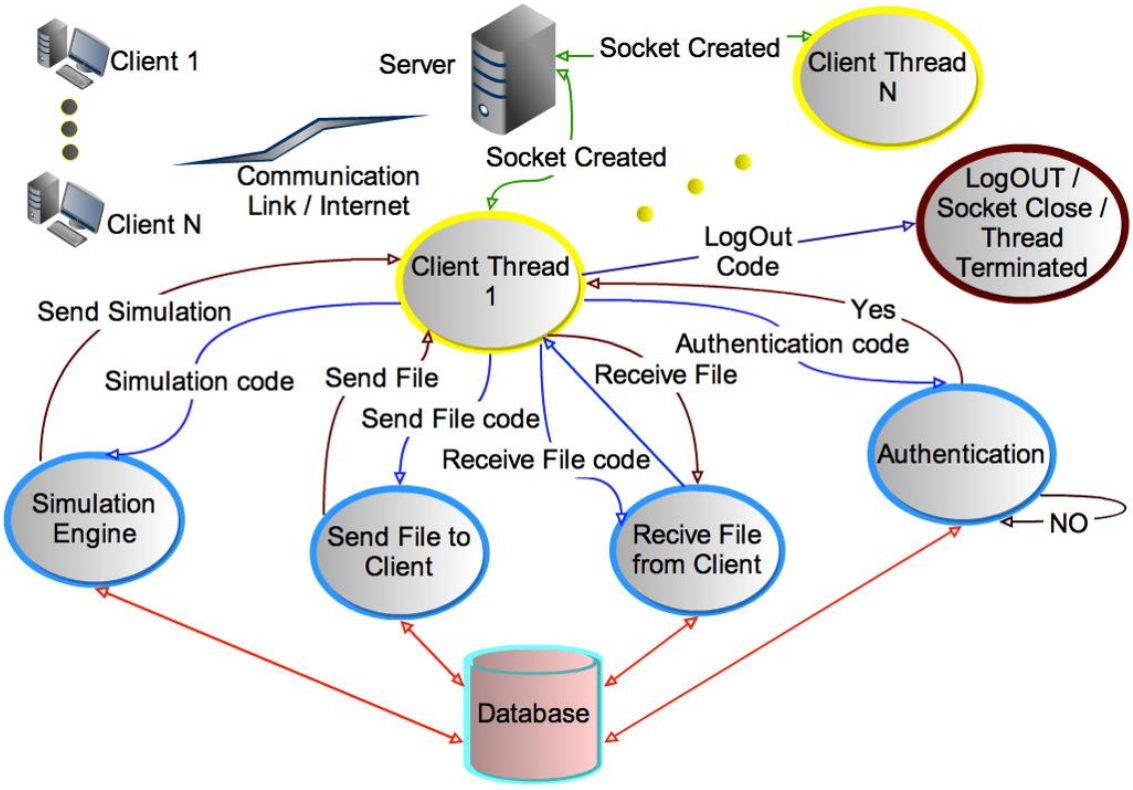

Figure 3: Experience Accelerator Server System Architecture [8]

2.2 learning with the SEEA

The following is a brief description of the Experience Accelerator operation [4]:

The Learner logs into the system, is presented with a control screen as shown in Figure 3, and selects an Experience from those available. The Experience Master loads the selected Experience and the appropriate Phase to present based on the history of Learner activity for this experience. The Experience Master then ensures that each of the other major modules are loaded with the proper contextual information and are sequenced appropriately. The Challenge Control takes as inputs the Learner's profile and EA log, and the Competencies and Aha's that have been targeted by the current Experience. The Challenge Control then selects from the available set of Challenges & Landmines, and Experience Scenarios for use in this Experience. The Experience Master uses this information to update the internal Experience state appropriately.

Figure 4: Experience Accelerator Control Screen [8]

The Simulation Engine uses the Simulation Models and Data along with the Experience state that resides in the Experience Master to create an Experience Scenario optimized to meet a specific set of project goals relating to schedule, cost, capabilities and quality. The Scenario contains an objective utility function of the project metrics that the Experience Master uses to determine a score for the overall success of the experience.

Selected Challenges and Landmines are used to create deviations in the optimized scenario to reflect the challenges. For example, the required schedule and/or budget can be reduced beyond what is achievable, resources can be misallocated, or desired features can be removed. The Simulation Engine is thus able to determine the experience's level of success if the Learner makes no decisions that affect the experience's outcome. The degree to which the learner is able to increase this score is a measure of success. If the score falls below this level, it is a sign of counterproductive actions by the Learner. Landmines can be set up to be tripped when certain parameters in the Experience go beyond a certain set limit. For example, if the Experience is not sufficiently challenging for the Learner, a new stakeholder can be introduced who has new requirements, or productivity in a team can be reduced due to personnel being pulled to other projects.

Once the project challenge has been incorporated via the deviations, the Simulation Engine updates the utility function to create the "gold standard" view of the Experience. This means the Learner can enter into the experience to identify and solve the problems.

Information that provides clues to the Learner are revealed passively through dashboard displays, status reports, graphs, and emails, and actively through conversations with Non-Player Characters (NPCs). Based on their ability to discover the challenge issues, the Learner needs to take corrective action. These actions generally take the form of recommendations or actions that either affect the inputs to the Simulation Engine or reset the expectations for the program's results. In the case where the Learner is a technical leader, the corrective actions of the role focus on identification and mitigation of technical problems and risks. Significant communication, facilitation and coordination skills are required, especially for programmatic changes that must be agreed upon by various program personnel and stakeholders and approved by the program manager.

In general, the Learner makes corrective changes through written recommendations to the relevant decision-maker or interacting with the decision-maker NPC. There is generally some ambiguity in communication with these NPCs, and so the recommended changes may or may not be made. Again, the Challenge Control determines the number and amount of deviation that is input to the Simulator. Note that the Learner also has access to their profile, the log, artifacts that were created on previous experiences and has the ability to restart previous experience simulations. This can be used to help the Learner understand the connection between the recommended actions and their impact on the responses from the simulation.

2.3 the current SEEA experience

The current public SEEA experience was designed in a defense acquisition program context [9] where an Unmanned Aerial Vehicle (UAV) acquisition program is underway (Figure 5). The learner assumes the role of lead program systems engineer (LSE) just after the preliminary design review, replacing the previous lead program systems engineer. The learner is tasked immediately with identifying any existing problems with the program. The learner progresses through a set of phases corresponding to those of a real UAV acquisition program, each phase ending with an important program milestone [10]. The UAV Experience developed in the baseline year focused on developing the systems thinking, and problem solving and recovery skills of a DoD Lead Systems Engineer [11].

The UAV system under development consists of three major subsystems: The Airframe and Propulsion is primarily electro-mechanical, the Command and Control system is mainly software, and the Ground Support system is mainly human based. The key performance measures (KPMs) are schedule, quality, range and cost. Each of the learner's sessions in the Experience represent a single day in the program and are estimated to take approximately one hour to complete, although the learner is free to login and out any number of times during a session.

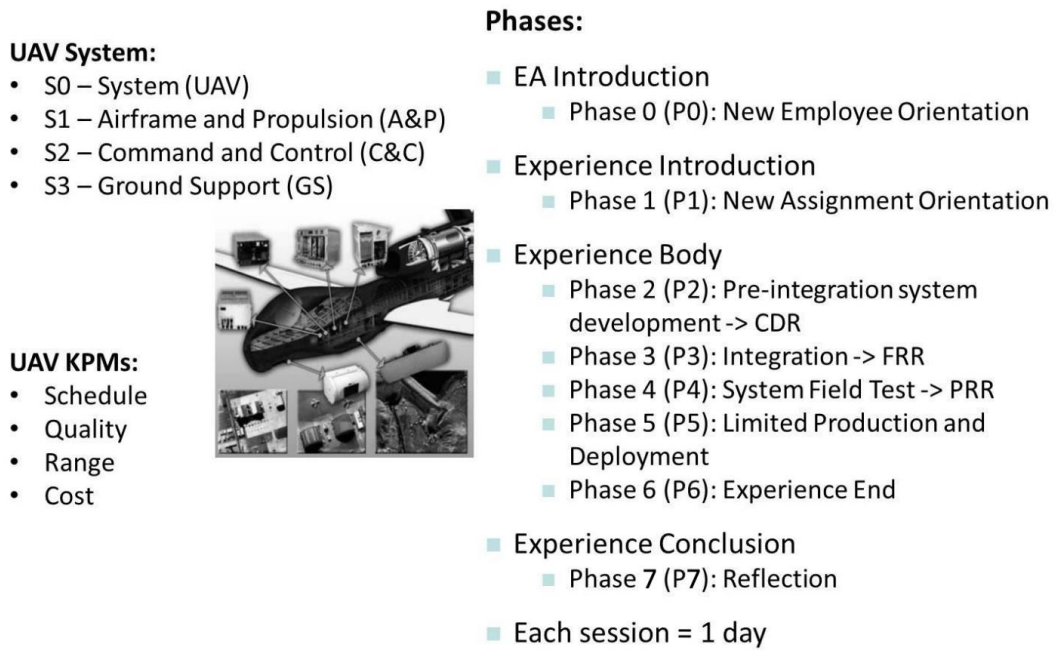

Figure 5: Context for the UAV Experience [11]

Phases represent the different stages of an experience. Within each phase, the learner can interact with the personnel involved in the program, review documents and reports together with project status, and then make decisions to prevent or recover from problems. Inside each phase, there are sub-phases that contain cycles representing smaller portions of the detailed project lifecycle experience. The complete UAV experience is composed of 8 phases as shown in Table 1 [12].

Table 1: Phases for the UAV experience [9]

Phase	*Phase Description*		
	Phase Activity Focus	*Ending Event*	*Activities*
0	SEEA Introduction	Survey completion	The learner is introduced to the SEEA
1	Assignment to UAV Program	Submission of likely problems and actions	Introduction to the experience
2	System Pre-integration	Critical Design Review	Acts as LSE
3	System Integration	Flight Readiness Review	Acts as LSE
4	Flight Test	Production Readiness Review	Acts as LSE
5	Limited Production	Integrated System Review	Acts as LSE
6	End of Project	Success or Failure	Results are presented.

7	Reflection	End of experience	Receive information about their decisions and reflect on learning objectives.

The architecture of SEEA experience is shown in Figure 6 in UAV project context. Learner decisions are made based on the information gathered from documents and the NPC interaction. Once decisions are recommended, a systems dynamics model of a typical large development simulation and tuned to the UAV experience is executed to advance the time in the experience. The learner is then presented with the results of their decisions in the form of project status updates that arrive in email and are displayed on the project dashboard.

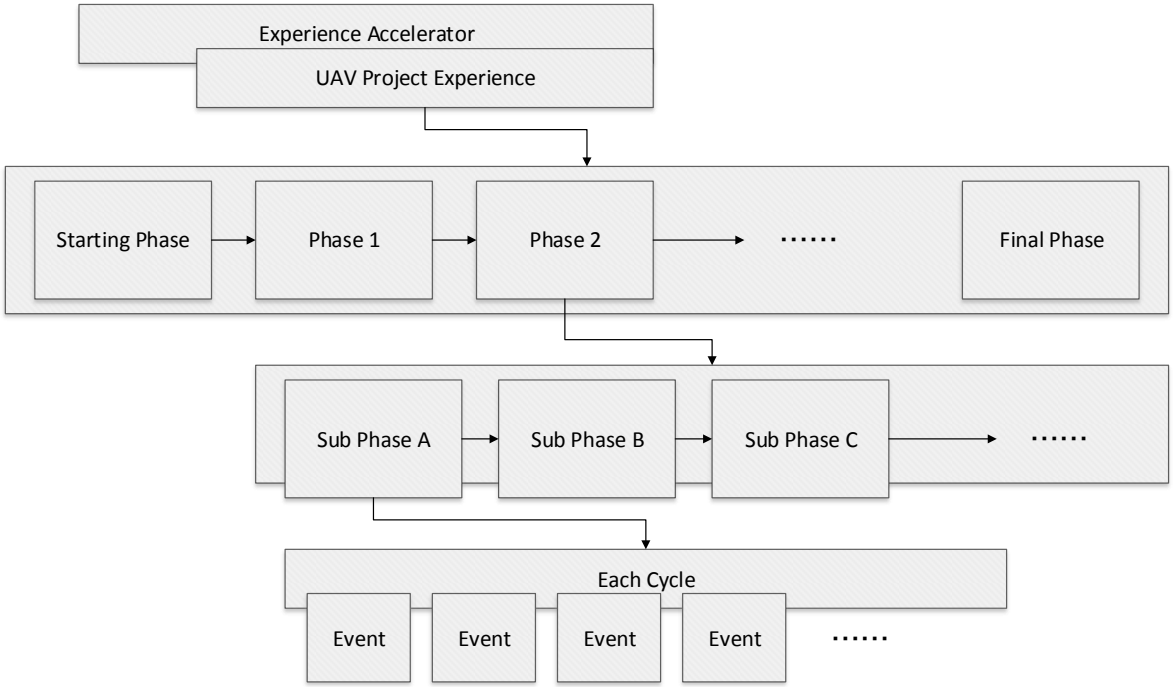

Figure 6: Typical SEEA Experience Structure Design

Within each phase/sub-phase/cycle, there are multiple events to simulate the situation of a real world scenario. Events can be triggered throughout the experience, and provide experience designers the ability to create a more dynamic experience. Each event contains the conditions necessary to trigger the event and the actions that take place during the event. Conditions that can trigger events include [13]:

- Experience time
- Leaner opening documents
- Learner submitting decision
- Value changes of certain project status variables

The following are some of the actions that can happen as a result of an event triggering:

- Receiving an email from an NPC
- Receiving a voicemail from an NPC
- Receiving a phone call from the NPC
- Sending pop-up messages to the learner about experience status

- Transitioning to another phase/sub-phase/cycle
- Changing values of state variables used by other parts of the SEEA

Throughout the UAV experience, learners encounter multiple events each phase/cycle. For example, in the introduction phase – Phase 0, learners receive a voicemail from the HR manager welcoming the learner to the team, as shown in Figure 7. As time progresses, an email from the assistant of the HR manager reminds the learner to complete an experience survey. Once the learner submits the survey, the experience moves to the next phase as shown in Figure 8.

Figure 7: A Voicemail Event in Phase 0

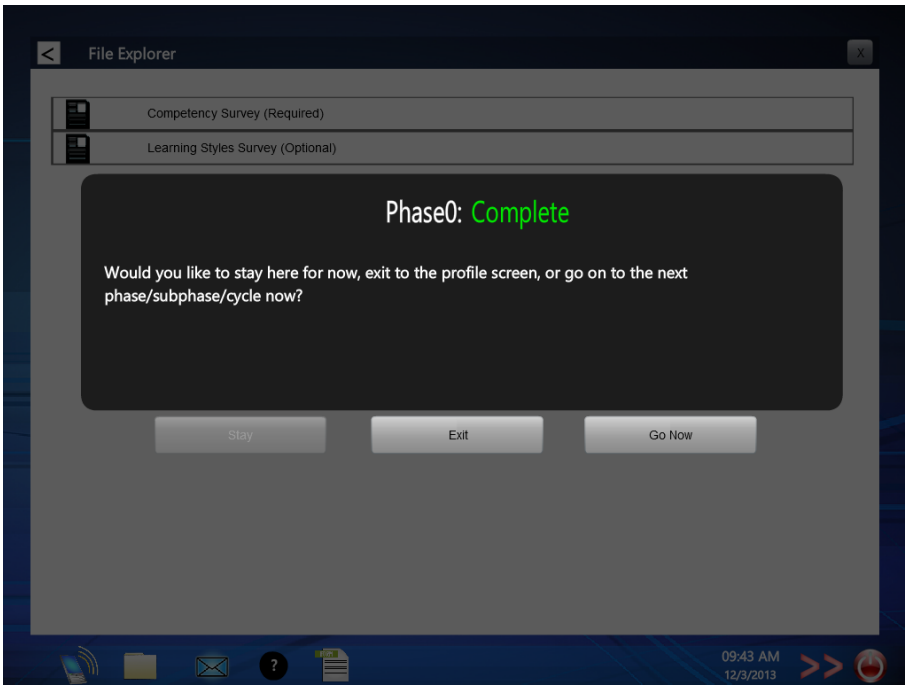

Figure 8: Phase 0 Completion Notification

Learners have access to many artifacts in the experience. There are two kinds of artifacts in the SEEA server file system storage: static artifacts and configuration artifacts [12]. Static artifacts are the textual and graphical materials made available to the learners in the experience. The UAV experience uses the following type of static artifacts:

- Email contents
- Voicemail contents

- NPC dialogs
- PDF format documents and charts, Figure 9 shows a PDF document presented in the UAV experience

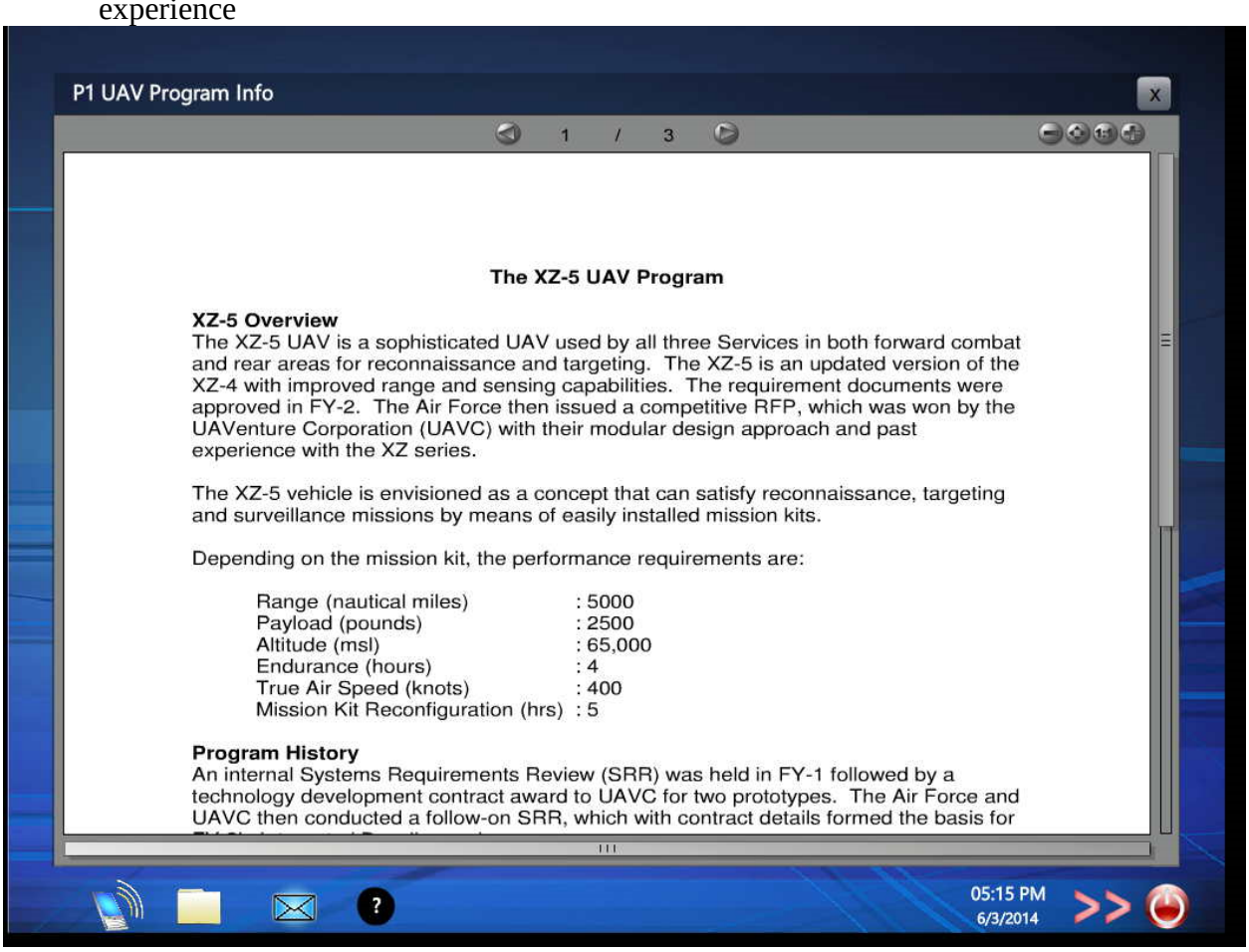

Figure 9: Static Artifact in the UAV Experience

Configuration artifacts are files residing on the SEEA server that store configuration information of the experience. For example, the following XML formatted files are considered to be configuration artifacts:

- Learner File System Data
- Phase Data
- Email Index
- Event Configuration

Most configuration artifacts are interconnected following the design of the SEEA system. This design provides a certain degree of flexibility and modularity to the system if experience modification is desired, however it increases the learning curve for experience designers to create or modify experiences.

2.4 SEEA experience construction without tools

Before the development of the tools, the SEEA experience construction required not only programming skills but also significant knowledge of the architecture and underlying infrastructure of the SEEA. Depending on the complexity of the experience, the amount of effort varied significantly. In the UAV experience, many components of the system structure were implemented specifically for the UAV experience, two software developers were collaborating in the development team in an effort to update the experience. Updating the experience flow and phase transition required modification of both client-side and server-side source code as well as interaction with complex dialogs being developed in a stand-alone tool. Updating the simulation engine also required programming efforts. Furthermore, to make changes to the experience master, experience designers need to first be familiar with the underlying infrastructure of the SEEA. The majority of the experience data reside on the server file system which requires multiple I/O operations.

New experiences require a complete design and development process with simulation engine and experience flow implementation. For experience flow development, server side code needs to be reconfigured to reflect to the new phase/sub-phase sequence. As an example, the creation of email events that can be triggered requires the following steps:

- Create the artifacts needed for the event, in this case an email file in plain text format.
- Add a new entry in the email index file with the sender, subject and email body reference information and link the new email entry to the email file created.
- Navigate to the events data file. Add the path of the email index file to the event data file.
- Navigate to the phase data file. Add a new entry in the file with information about event type, state type, session type, cycle, action type, action, external reference, and the id for the new email event.

The process not only requires understanding the infrastructure of the EA, but also specific knowledge about server files structure and the skills needed for composing and interpreting XML files.

3 experience accelerator tools development

3.1 design process and tool identification

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.

These efforts can be categorized into the following areas, with the most needed and highest potential for tool development ranked by number:

- Objectives & Experience Concept Development
  - Learner Profile creation
  - Competencies and Aha’s identification
  - Experience story boarding and conceptualization
- Context
  - Project specification
  - Project state and thresholds
  - Roles, motivations personality factors and character types
  - Review types and result options

- Experience Events and Flow
  - *3 Experience Phases & Time specification
  - *4 Event specification
  - Challenge Control -> simulation parameter setting, event triggering
  - Evidence of Challenges and Landmines
  - Mitigating Actions & Effects
  - Relationships between Competencies/Aha's, Challenges/Landmines, Mitigating Actions & Effects
- Reflection & Evaluation
  - Feedback format
  - Scoring
  - *6 Learning evaluation
- Artifacts
  - Background information
  - *7 Project reviews
  - *8 Learner recommendation forms
  - *9 Simulation status reports
  - emails
  - *10 Dialog
  - Mentoring
  - Evaluation feedback
- Simulation
  - *1 Models construction
  - *2 Parameter setting
- Overall
  - *5 Artifact entry
  - *11 Process and tools documentation

The critical limitations and proposed tools for addressing them are shown below in Table 2.

Table 2: Limitations and Tools Solution

#	**Limitation**	**Solution**
1	Complex simulation models with limited reuse supported	**Sim Builder** - Simulation model builder utilizing libraries/templates
2	Complex simulation outputs dependent on hundreds of variables/parameters	**Sim Tuner** - Parameter tuner that automates the tuning of parameters to yield desired outputs via batch processing of different combinations
3	Manual nature of phase and cycle development	**Phase Editor** - GUI-based tool for phase, cycle and event specification with code generation
4	Manual nature of specifying events and their triggers in the Experience	**Event Editor** - GUI or text-based tool to specify events and their triggers with code generation
5	Manual nature of artifact integration involving re-linking and recompilation	**Artifact Integrator** - Artifact entry application that allows designer to take an artifact file and enter it into EA application with automatic recompilation and re-linking

6	Manual nature of assessment of learner performance	**Learning Assessor** - Assessment tool-suite that provides automated performance scoring, decision comparisons against proven baselines, etc.

The design and development tools noted above is only a subset of the possible tools and template identified by the team, and were identified as the most important ones for inclusion at this time. Others include:

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

3.2 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 10.

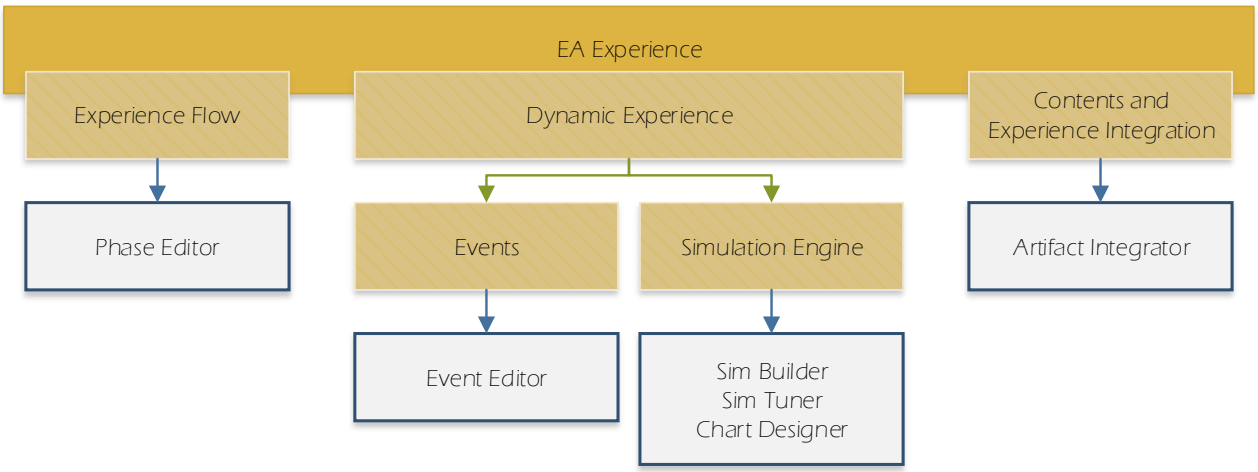

Figure 10: Overview of Tools and Aspects of the EA Experience

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.

3.3 experience building tools

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 [12]. 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 11 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.

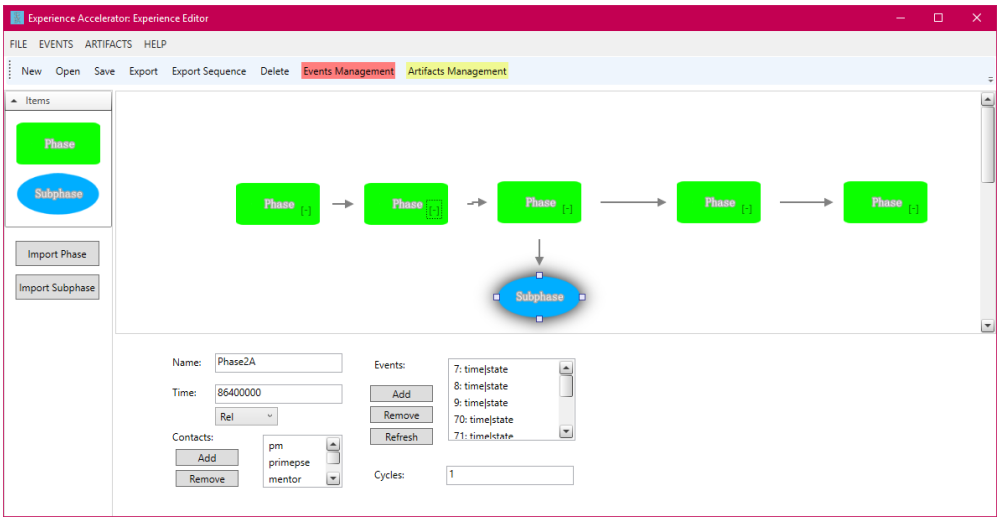

Figure 11: Phase Editor User Interface

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 12 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.

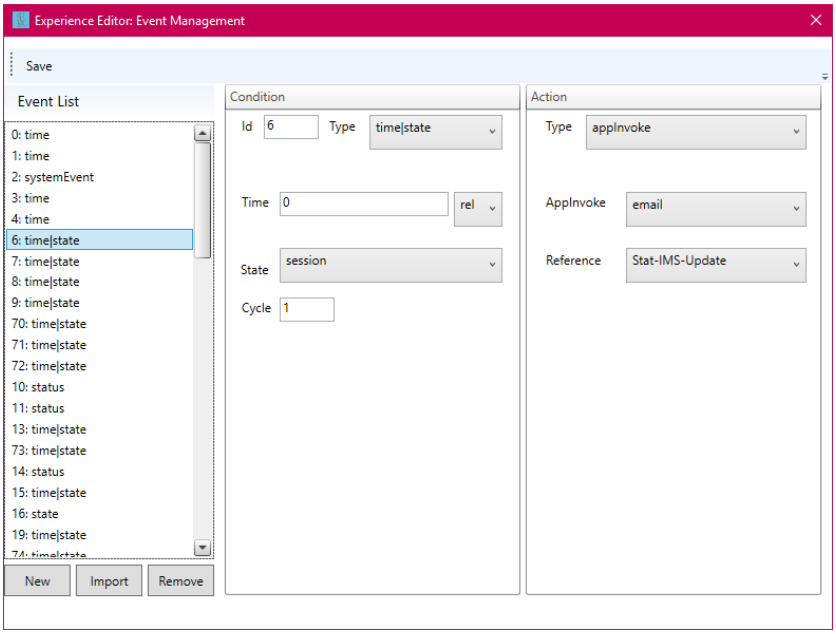

Figure 12: Event Editor User Interface

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 13 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.

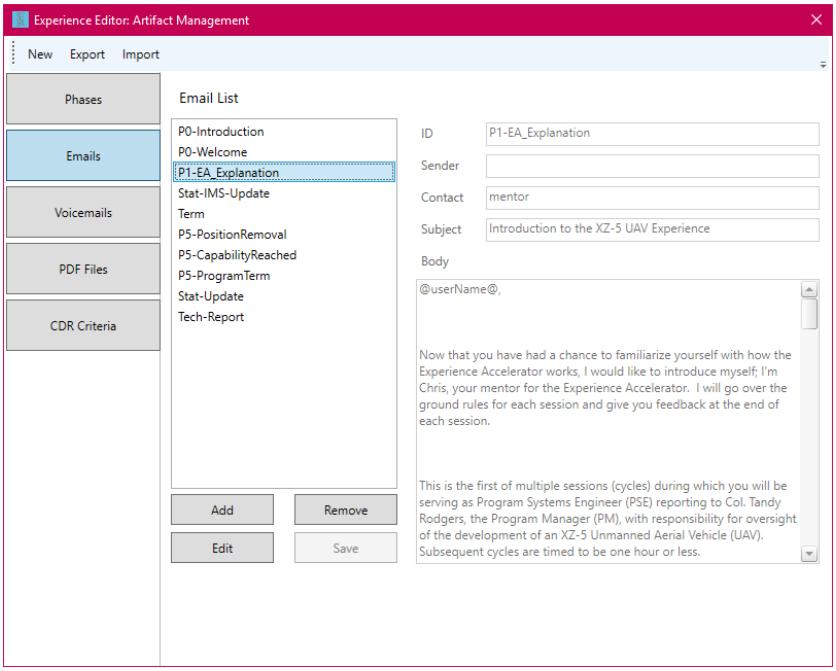

Figure 13: Artifact Integrator User Interface

3.4 simulation tools

The simulation tools comprise a set of tools for an experience designer to build and test simulation models that mimic the behavior and results of programs that focus on system design and development. They also provide functionality to design simulation outputs in terms of charts that allow a learner to visualize program status and issues.

These tools 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.

Sim Builder – This tool provides the ability for experience designers without deep technical knowledge in simulation to build systems dynamics models based on existing templates in a GUI environment.

Sim Tuner – This application provides the ability for an experience designer to analyze the system, determine the sensitivity of various parameters, and aid in the tuning of the model to achieve desired behaviors. This works interactively so that the designer can specify desired model behavior.

Chart Designer – This tool provides the ability for experience designers to create simulation output charts for experience learners that visualize the current status of a program, plus potential future trajectories. Such status can include cost, schedule and technical performance metrics. These charts are used in the SEEA, while the output graphs in the Sim Tuner are used by the designer outside the SEEA to provide desired behavior.

The current toolset is based on an open-source, Java-based simulation toolset developed by Melcher [15]. This original toolset provides basic functionality to create and execute system dynamics models. It was extended substantially to support integration in the Experience Accelerator. This included additional mathematics functionality, support for multi-cycle and multi-phase simulation models with carry-over of variable values, support for learner changes of variables in between cycle (i.e., learner decisions for program improvement and problem-solving), customization for chart outputs used by the SEEA, and customization for simulator variable outputs integrated into the SEEA. These features were used to design and implement the simulation models that support the UAV program experience in the SEEA [16].

The existing toolset also had a GUI for building simulation models and visualizing outputs. However, this toolset had several significant limitations in supporting complex models with hundreds of variables and relationships such as those developed to support the SEEA.

- Cut-and-paste are not support for model elements.
- Model modularity is not supported. This makes the visual system dynamics graphs difficult to understand, maintain, and evolve.

- Along with lack of modularity, there is of course no function for archiving and retrieving sub-model building blocks that could prove beneficial in model reuse.
- Only one interactive output graph for simulation variables is supported (although, there is supported for multiple y-axes for different variables. For models with hundreds of variables, this is a somewhat cumbersome representation.
- Since the output chart design function was not originally supported, there is also no method to design output charts for the SEEA application.

We have enhanced this original toolset extensively in creating the Sim Builder, Sim Tuner and Chart Designer. In addition to supporting cut-and-paste, the Sim Builder addresses the issue of model modularity by implementing system dynamics sub-models. These serve as building blocks for more complex models. The Sim Builder also implements new sub-model archive and retrieve functions to allow an experience designer to reuse sub-models. For instance, the UAV program experience in the SEEA includes simulation models that represent the following program elements:

- Sub-systems of the UAV under design and development (e.g., airframe and propulsion, command and control, and ground station), along with the sub-contractors responsible for designing and developing each;
- Key performance parameters and technical performance measures of the UAV under design and development (e.g., range, weight, drag, etc.), broken down by overall system and sub-system;
- Sub-contractor technical workforces, experience levels and productivities;
- Cost and budget using earned value management, broken down by overall program and sub-contractor;
- Entrance criteria for critical design review and progress toward these objectives, broken down by criteria and responsible sub-contractor.

Figure 14 shows the overall simulation model (without sub-models) on the left, with a sub-model implemented specifically for range, weight, drag, and propulsion efficiency. Clearly, the model on the left is complex and difficult to understand and maintain. The sub-model is developed using the Sim Builder with its sub-model archiving function. The idea is to divide the overall model elements above into sub-models for improved modularity and reuse through a sub-model library with appropriate documentation.

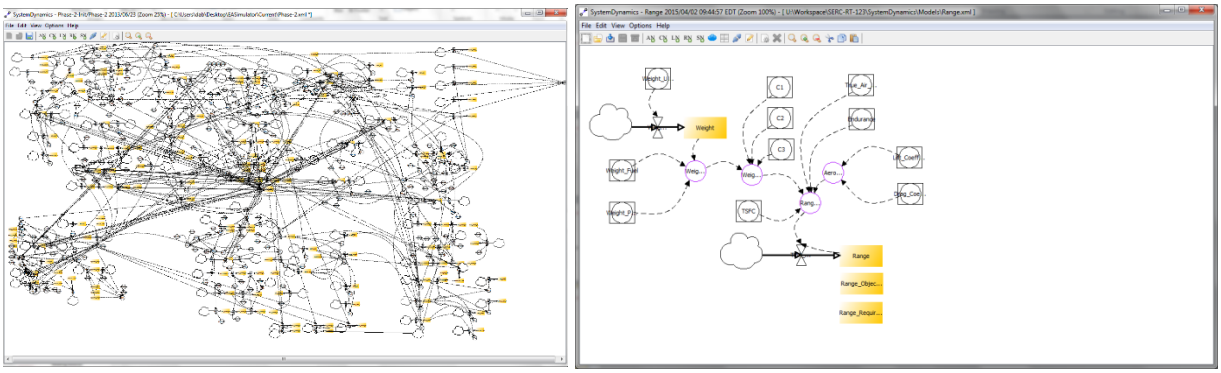

Figure 14: Simulation models for the UAV program – overall model vs a sub-model

The Sim Tuner now incorporates the interactive ability to graph variables in each sub-model separately. In addition, work is underway to support the changing of variable and parameter values by the designer in between "cycles," so that a designer can interactively experiment with the changes that a learner might make while using the SEEA. This is shown in Figure 15, with a separate tab being used to display output graphs from the different sub-models. The designer can execute multiple consecutive runs of the simulation to see how it behaves over consecutive cycles in the experience. In addition, the designer can iterate back and forth between the Sim Builder and Sim Tuner via a common integrated toolset interface so that a model can be modified when testing its behavior.

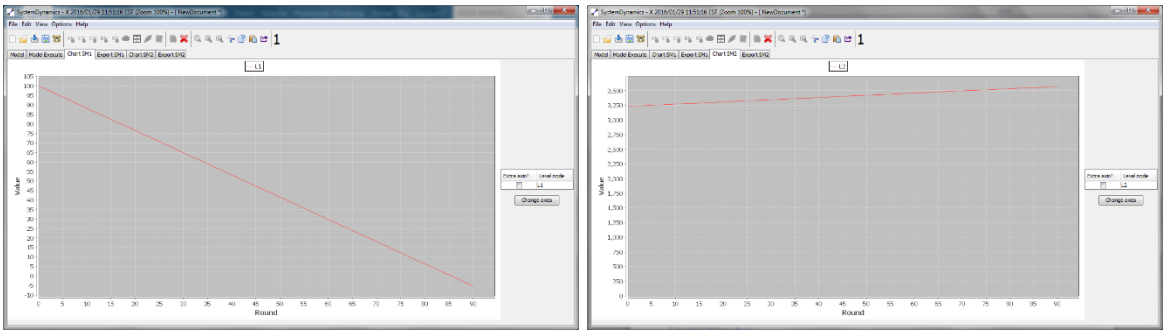

Figure 15: Multi-graph interactive output

The Chart Designer is currently being designed and developed. It will be part of the common toolset accessible from the common interface. It will allow the design of the SEEA output charts. These are similar to the interactive graphs used by the designer, except that these are designed specifically for the learner in an experience as decision support on program status. One such chart is shown in Figure 16. This chart provides the status and trajectory of weight and weight growth for the airframe and propulsion sub-system.

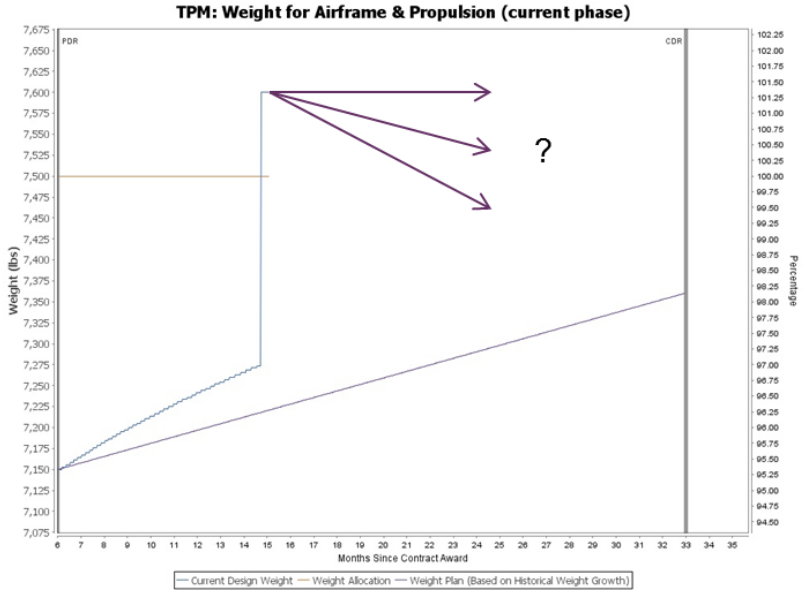

Figure 16: Output chart for airframe and propulsion weight

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 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.

3.5 learning assessment tools

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 [17-30]. 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 assessment is a critical component of accelerated learning [11]. It is imperative to understand individual learning and the efficacy of the various learning experiences. This is critical both in determining the capabilities of the learner, but also to continually improve the capabilities of the learning experience.

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 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.

3.5.1 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 [33]. These skills work together as a system, feeding into and depending upon one other to produce a greater net effect [33]. 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 [34]. 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 [35].

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 [36].

As part of the research to assess systems thinking ability, a standardized definition of systems thinking has been proposed [33] 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 [37, 38]. 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.

3.5.2 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.

3.6 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 mentioned in section 2.3 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:
    - Experience information
    - Learner's information including name and email address
- Experience Session Information:
    - Number of past attempts at the save experience
    - Number of "resets" and phase/sub-phase/cycle of "resets"
    - Date/time of experience start and end
- Learner Experience Input & Actions:
    - Learner's Self-assessment data
    - Learner decision making in the form of recommendations inputs
- Simulation Output and Project Status:
    - Last phase/sub-phase/cycle completed
    - Results of simulation outputs from each phase/sub-phase/cycle
    - Project Status

4 application and validation of tools

4.1 modifying existing experiences using the tools

Although the implemented tools can greatly reduce the efforts needed for creating and modifying experience, they still have their limitations. To validate the capability of the tools in the scenario of updating an existing experience and to tests the limitations of the tools, a trade study session will be added to the existing UAV experience using the tools.

One of the most crucial skills for a seasoned systems engineer is the ability to make well-reasoned technical decisions, balancing often conflicting criteria to arrive at an optimal solution. In a typical tradeoff analysis, multiple competing design alternatives are assessed against defined criteria, often using weightings established through stakeholder consensus. Given the importance of this skill, its inclusion in the SEEA was considered to be the next priority in the learning experience improvements [32].

As mentioned in section 3.2, modification of existing experience involves three aspects of the experience: experience flow, dynamic experience, contents and experience integration. The addition of this trade study session mainly involves the latter two aspects as it does not impact the experience flow directly. Figure 17 shows the flow of trade study introduction.

From the design of the trade study session, tools can be used to greatly reduce the efforts needed. The major phases of the trade study session were defined as follows with the possible use of the tools [32]:

- Trigger the trade experience – Event Editor and Phase Editor
- Setup actuator alternatives and weighting criteria, via NPC dialogues –Event Editor and Chat Mapper
- Provide Learner with required data, via NPC dialogues and artifacts – Chat Mapper and Artifact Integrator
- Provide a trade recommendation form, including rationale, for communication to PM – currently missing
- Update the simulation based on the Learner's recommendation – Sim Builder and Sim Tuner
- Provide experience feedback – Artifact Integrator

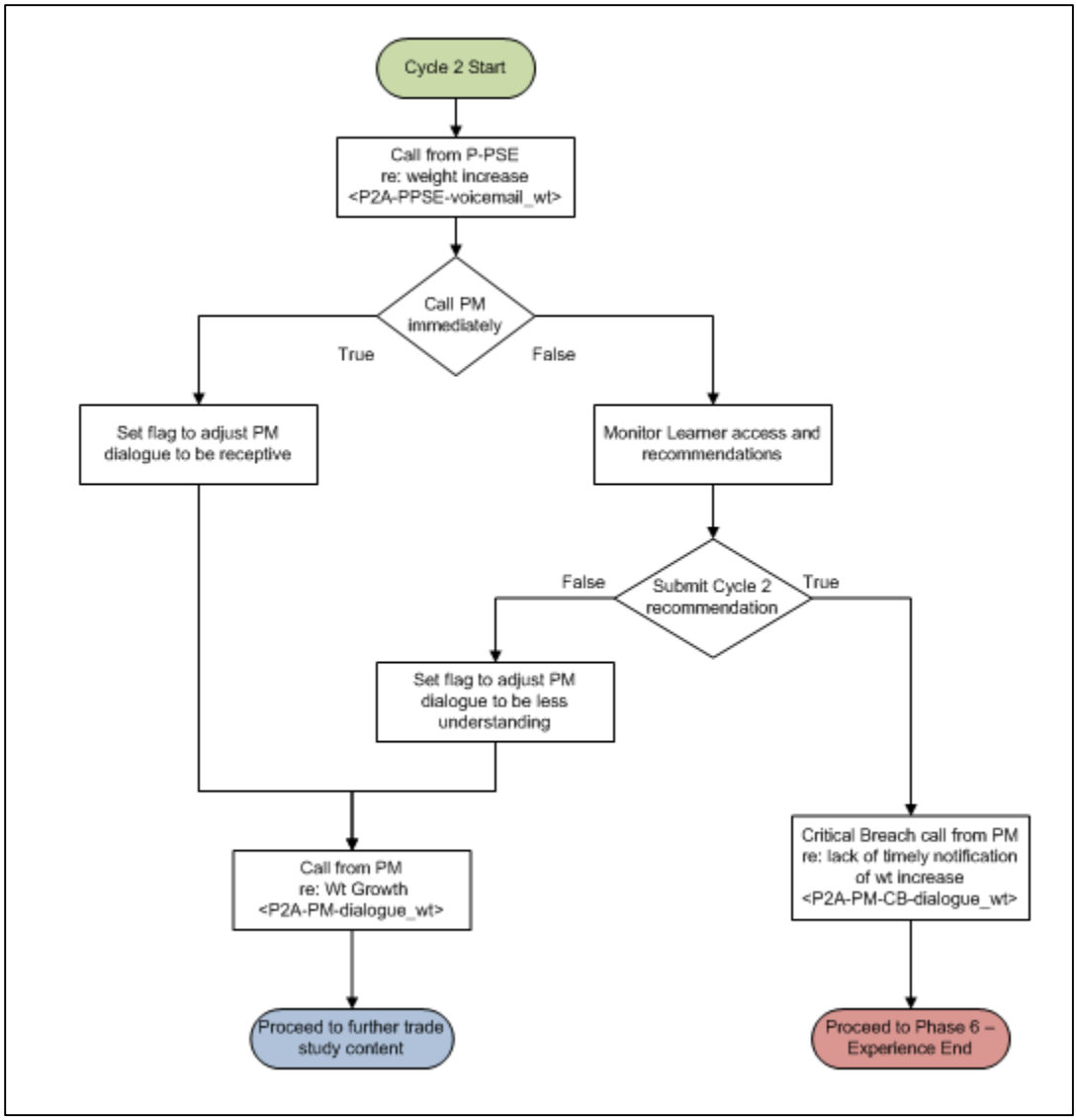

Figure 17: Trade Study Introduction Flowchart [32]

The trade study session is designed to be started at the beginning of phase 2 sub-phase A cycle 2. Experience designer can create the main events in this trade study using Event Editor and Artifact Integrator. Voicemails, and emails can be created using the Artifact Integrator to provide learner information and feedbacks regarding the experience. The phone call dialogs can be created using the third-party tool Chat Mapper. The experience designer can then use the phase editor to incorporate the created events to the experience using Phase Editor. For the simulation modification, Sim Builder and Sim Tuner will be used to adjust the simulation model with the learner's input in the trade study session. However, due to the limitation of the client design of SEEA, the creation of a customized recommendation form on the graphical user interface (GUI) of SEEA still requires some coding efforts of ActionScript in Adobe Flash.

In this experience tailoring scenario, Phase Editor, Event Editor and Artifact Integrator provided the composition capability of experience flow, event and artifacts. Sim Builder and Sim Tuner provided the essential feature to make changes to the simulation engine without programming

efforts. Overall, the functions of the tools covered the majority efforts needed to modify an existing experience with the exception of GUI modification, and eliminated major coding efforts for the experience designers.

4.2 creation of new experiences using the tools

Building new experiences is primarily a challenging creative activity. Even with the new tools the steps described in Section 3.1 are essentially the same, and are dependent on the knowledge and imagination of the experience designer. However, the tools provide a systematic, integrated infrastructure that simplifies developing experience components by taking care of the integration and coordination between them.

To evaluate the capabilities of the tools, a scenario for a completely new experience is underway. The new experience focuses on a submarine maintenance scenario where the leaner 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 leaner and provides the leaner 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 leaner 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.

5 conclusions and future work

This research developed a set of tools specifically for educators and experience designers outside the SEEA research and developing team, to make developing experiences easier. Targeting different aspect of experience development, the research task developed three types of tools and essential SEEA infrastructure modifications. Experience Building Tools including the Phase Editor, Event Editor and Artifact Integrator were developed to address the manual nature of previous experience creation process. Simulation Tools including Sim Builder, Sim Tuner and Chart Designer were built to provide code-free simulation engine development. Learning Assessment Tools gather recorded data from learners' experience and analyze it to provide assessment of learners' performance through the experience. The capabilities of the Experience Building Tools and Simulation Tools are being evaluated through the creation and modification of experiences.

For the SEEA to reach its full potential, there needs to be an active community of users, supporters and developers [11]. Having an accessible application, experience, tools and supporting documentation are critical to this. However, additional efforts are necessary to create the community, including website development, distribution support, community building, user group activities and licensing support.

As mentioned in section 4.1, there are still limitations related to the technologies used by the prototype SEEA. To address those, a new presentation engine may be needed to provide code-free GUI design for new experiences. HTML5 and JavaScript are the preferred technologies to replace the aging Adobe Flash used in the current SEEA. This migration will enhance the reliability, flexibility of the SEEA and improve the quality of experiences running on it. Expanding the simulation component beyond systems dynamics models to include agent–based simulations can extend the range and richness of new experiences.

To further validate the tools as they improve, additional activities will be performed including the revision and completion of the existing UAV experience using the tools.

Learning assessment and evaluation are crucial to the experiential and accelerated learning. Next phase for learning assessment tools development will include prototype improvements, assessment engine development and learning data gathering through pilot uses of the SEEA.

acknowledgements

This material is based upon work supported, in whole or in part, by the Systems Engineering Research Center (SERC). SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology. We are grateful to all the members of the Experience Accelerator team including members of participating universities, sponsor members, and subject matter experts.

references

[1] T. C. Bagg, "Systems engineering education development (SEED) case study," 2003.
[2] H. L. Davidz and D. J. Nightingale, "Enabling systems thinking to accelerate the development of senior systems engineers," *Systems Engineering*, vol. 11, pp. 1-14, 2008.
[3] A. Squires, J. Wade, P. Dominick, and D. Gelosh, "Building a competency taxonomy to guide experience acceleration of lead program systems engineers," DTIC Document2011.
[4] R. N. Charette, "What's wrong with weapons acquisitions?," *IEEE Spectrum*, vol. 11, pp. 33-39, 2008.
[5] J. Wade, W. Watson, D. Bodner, G. Kamberov, R. Turner, B. Cox, *et al.*, "Developing the Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap," DTIC Document2013.
[6] A. F. Squires, J. Wade, D. A. Bodner, M. Okutsu, D. Ingold, P. G. Dominick, *et al.*, "Investigating an innovative approach for developing systems engineering curriculum: The Systems Engineering Experience Accelerator," in *American Society for Engineering Education*, 2011.
[7] J. W. A. F. Squires, D. Bodner, M. Okutsu, D. Ingold, P. Dominick, R. Reilly, W. Watson, D. Gelosh, "Investigating an innovative approach for developing systems engineering curriculum: The Systems Engineering Experience Accelerator," presented at the ASEE Annual Conf, 2011.
[8] J. P. Wade, G. Kamberov, D. A. Bodner, and A. F. Squires, "The Architecture of the Systems Engineering Experience Accelerator," in *INCOSE International Symposium*, 2012, pp. 1806-1820.
[9] D. Bodner, J. P. Wade, A. F. Squires, R. R. Reilly, P. G. Dominick, G. Kamberov, *et al.*, "Simulation-based decision support for systems engineering experience acceleration," in *Systems Conference (SysCon), 2012 IEEE International*, 2012, pp. 1-6.
[10] A. Squires, J. Wade, B. Watson, D. Bodner, R. Reilly, and P. Dominick, "Year one of the systems engineering experience accelerator," *Procedia Computer Science*, vol. 8, pp. 267-272, 2012.
[11] J. Wade, R. Turner, D. Bodner, W. Watson, P. Zhang, H. Kang, *et al.*, "Developing the Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap," Final Technical Report SERC-2015-TR-016-42015.
[12] K. Hao, "Design and Development Tools for the Experience Accelerator," Thesis of Computer Science Engineering Degree, Department of Computer Science, 2015.
[13] B. Cox, "Towards A Better Experience Accelerator," Thesis of Computer Science Engineering Degree, Department of Computer Science, 2014.
[14] D. A. Bodner, J. P. Wade, W. R. Watson, and G. I. Kamberov, "Designing an Experiential Learning Environment for Logistics and Systems Engineering," *Procedia Computer Science*, vol. 16, pp. 1082-1091, 2013.
[15] J. Melcher, "SystemDynamics. [Online]," ed, 2009.
[16] D. A. Bodner and J. P. Wade, "Multi-criteria simulation of program outcomes," in *Systems Conference (SysCon), 2013 IEEE International*, 2013, pp. 215-221.
[17] J. Armstrong and J. Wade, "Learning Systems Engineering by Teaching It," in *INCOSE International Symposium*, 2015, pp. 211-220.

[18] M. Hopper and K. A. Stave, "Assessing the effectiveness of systems thinking interventions in the classroom," in *Proceedings of the 26th International Conference of*, 2008.
[19] B. Kopainsky, S. M. Alessi, M. Pedercini, and P. I. Davidsen, "Exploratory strategies for simulation-based learning about national development," in *27th International Conference of the System Dynamics Society, Albuquerque, NM*, 2009.
[20] B. Kopainsky, P. Pirnay‐Dummer, and S. M. Alessi, "Automated assessment of learners' understanding in complex dynamic systems," *System Dynamics Review*, vol. 28, pp. 131-156, 2012.
[21] B. Kopainsky and M. Saldarriaga, "Assessing understanding and learning about dynamic systems," in *30th International Conference of the System Dynamics Society, St. Gallen, Switzerland*, 2012.
[22] B. Kopainsky and A. Sawicka, "Simulator‐supported descriptions of complex dynamic problems: experimental results on task performance and system understanding," *System Dynamics Review*, vol. 27, pp. 142-172, 2011.
[23] E. McGrath, "Research on Building Education & Workforce Capacity in Systems Engineering," DTIC Document2011.
[24] V. Rimiene, "Assessing and developing students' critical thinking," *Psychology Learning & Teaching*, vol. 2, pp. 17-22, 2002.
[25] H. Skaza and K. Stave, "A test of the relative effectiveness of using systems simulations to increase student understanding of environmental issues," in *Proceedings of the 27th International Conference of the System Dynamics Society*, 2009.
[26] H. Skaza and K. A. Stave, "Assessing the effect of systems simulations on systems understanding in undergraduate environmental science courses," in *Proceedings of the 28th International Conference of the System Dynamics Society*, 2010.
[27] H. J. Skaza, "Assessing the effect of simulation models on systems learning in an introductory environmental science course," 2010.
[28] K. A. Stave, A. Beck, and C. Galvan, "Improving Learners' Understanding of Environmental Accumulations through Simulation," *Simulation & Gaming*, p. 1046878114531764, 2014.
[29] K. A. Stave, H. Skaza, and B. Jurand, "Using Simulations for Discovery Learning about Environmental Accumulations," in *International Conference of the System Dynamics Society*, 2011.
[30] J. D. Sterman, "Does formal system dynamics training improve people's understanding of accumulation?," *System Dynamics Review*, vol. 26, pp. 316-334, 2010.
[31] J. Wade, "RT16 Experience Accelerator Overview," 2014.
[32] D. Fontaine, "Addition of Trade Study Competency to the Systems Engineering Experience Accelerator Prototype," Student Thesis, Stevens Institute of Technology. Hoboken, NJ., 2015.
[33] R. Arnold and J. Wade, "A Definition of Systems Thinking: A Systems Approach," *Procedia Computer Science*, vol. 44, pp. 669-678, 2015.
[34] S. Burandt, "Effects of an Educational Scenario Exercise on Participants' Competenecies of Systemic Thinking," *Journal of Social Sciences*, vol. 7, pp. 54-65, 2011.
[35] D. Dörner, A. Wearing, "Complex Problem Solving: Toward a (Computersimulated) Theory," *Problem Solving: The European Perspective*, 1995.
[36] R. Plate, "Assessing individuals' understanding of nonlinear causal structures in complex systems," *System Dynamics Review*, vol. 26, pp. 19-33, 2010.
[37] E. Klieme, U. Maichle, "Evaluation of a Model Building System in the Classroom," 1991.
[38] G. Ossimitz, "Teaching Systems Dynamics and Systems Thinking in Austria and Germany," in *The 18th International Conference of the System Dynamics Society*, 2000.