SYSTEMS ENGINEERING
Research Center

Developing the Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap – Year 2

Technical Report SERC-2012-TR-016-2

October 24, 2012

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

Team Members:

Georgia Institute of Technology: Dr. Douglas Bodner, Pradeep Jawahar, Subbu Ramanathan, Aravind Sundaram,

Purdue University: Dr. William Watson, Dana Ruggiero,

Stevens Institute of Technology: Dr. George Kamberov, Dr. Alice Squires, Brent Cox, Vincent Simonetti,

With Contributions by Subject Matter Experts:

Eric (Rick) Abell, John Griffin, John McKeown

Contract Number: H98230-08-D-0171 Task Order: 0006, DO 001, RT016a

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Office of the Assistant Secretary of Defense for Research and Engineering (ASD(R&E)) under Contract H98230-08-D-0171 (Task Order 0006, DO 001, RT016a).

Executive Summary

This document is a summary of the work that was completed in the first increment year of the SERC Research Topic DO1/TTO2/0016 "Developing Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap" supported by the Defense Acquisition University (DAU). The purpose of the research project is to test the feasibility of a simulated approach for accelerating systems engineering competency development in the learner. The SEEA research project hypothesis is:

By using technology we can create a simulation that will put the learner in an experiential, emotional state and effectively compress time and greatly accelerate the learning of a systems engineer faster than would occur naturally on the job.

The major research activities in Increment 1 are as follows:

1. Update Documentation and Planning
2. Prototype Development:
   - 2.1. Experience Accelerator Technology: Presentation Engine, Experience Master, Challenge Control and Non-Player Character (NPC) Engine
   - 2.2. Simulation Engine
   - 2.3. Content Creation Tools Development
   - 2.4. Experience Design - Artifacts & Dialog
3. Informal Evaluation and Formal Evaluation Development:
   - 3.1. Informal Prototype Evaluation
   - 3.2. Design of evaluation experiment
4. Prototype Evaluation:
   - 4.1. Plan Update
   - 4.2. Learner Identification
   - 4.3. Prototype Evaluation
5. Write Final Report

In addition to the work activities, the following five top program risks were identified and tracked throughout Increment 1 of the program:

1. Project Management - Inability to support known and evolving customer/user feedback with current staff, budget and timeframe.
2. Technology Development - Inability to tradeoff long-term architecture and technology objectives (leading to successful open source support) vs. short-term prototype goals.
3. Experience Development - Inability to produce a prototype that provides a compelling experience, supports the desired learning and is seen to be authentic.
4. Integration - Inability to successfully integrate our many ideas, approaches, requirements and developed technology and design.

5. Evaluation - Inconclusive results due to threats to validity of Experimental design (inability to generalize results), limited availability of suitable subjects and insufficient literature to support development of evaluation instruments.

A set of lessons learned was compiled and categorized as noted below:

1. Competencies, Learning and Content
2. Complexity/Effort vs. Authenticity/Learning
3. Technology
4. R&D Processes

Follow-on work has been defined for Increment 2 that is focused on the following:

- Prototype Evaluation (cont.)
- Pilot System Development
- Pilot System Evaluation
- Open Source Preparation and Deployment
- External Developer Engagement

A final review of Increment 2 was held on September 6, 2012 with the following sponsor representatives: Jim Anthony, Tony Costanza, Darren Dusza, Steven Jones, Scott Lucero, Dave Pearson, and John Snoderly. The following are the highlights of the meeting results:

As the Experience Accelerator program is moving into Increment 2, there need to be increasing efforts made in the areas of pilot and tools support to both validate its effectiveness and improve the efficiency with which new experiences can be created, the latter being essential for open source community development. It was discussed that for piloting, there needs to be pre- and post-experience facilitation. Proper support needs to be put in place to enable this facilitation. It was also determined that DAU students should be used in this validation effort which will require government Institutional Review Board (IRB) support. In this case, we should be able to use an IRB exception in that this is normal educational development. The question arose on how the hypothesized outcome of "effectively compress time and greatly accelerate the learning of a systems engineer faster than would occur naturally on the job" would be measured. While the possibility of a large longitudinal study was discussed, the preferred approach was to compare the results of game traces of SEEA subjects to game traces of systems engineers who are deemed to be experts in selected areas, perhaps with the use of control groups with and/or without alternative equivalent educational instruction, to assess the subject's learning. The development of the capability to capture and analyze the subject's behavior in the Experience will be critical to this effort. Finally, a tools assessment plan needs to be completed to ensure that the Experience Accelerator development environment is adequate to support a thriving open source community.

Table of Contents

List of Figures

Preface

This document is a summary of the work that was completed in the Increment 1 of the SERC Research Topic DO1/TTO2/0016 "Developing Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap" supported by the Defense Acquisition University (DAU). This summary focuses on each of the work items noted in the proposal.

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

Experience Accelerator RT16 Project documents:

- RT16 Project Goals and Success Metrics (A013)
- RT16 Technical and Management Work Plan (A009)
- RT16 Monthly Status Reports (A008)
- Experience Accelerator Concept of Operations (A013)
- Experience Accelerator System Architecture and Design Specification (A013)
- Experience Accelerator Systems Specification (A013)
- Experience Accelerator: Experience Design Document
- Experience Accelerator White Paper
- Developing Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap, Increment 1 proposal
- Developing Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap, Increment 2 proposal
- Developing Logistics Experience Accelerator (EA) Prototype, 5/15/2012.
- Technical Leadership Development Experience Accelerator Prototype, 8/22/2012.

Publications:

- Bodner, D., Wade, J., Squires, A., Reilly, R., Dominick, P., Kamberov, G., Watson, W. (2012), "Simulation-Based Decision Support for Systems Engineering Experience Acceleration", IEEE Systems Conference, Vancouver, BC, Canada, March 19-23.
- Squires, A., Wade, J., Watson, W., Bodner, D., Reilly, R., Dominick, P. (2012), "Year One of the Systems Engineering Experience Accelerator", Proceedings from the Ninth Annual Conference on Systems Engineering Research (CSER), Rolla, Missouri, March 19-22, 2012.
- Wade, J., Kamberov, G., Bodner, D., Squires, A. (2012) "The Architecture of the Systems Engineering Experience Accelerator", International Council on Systems Engineering (INCOSE) 2012 International Symposium/European Conference on Systems Engineering (EUSEC), Rome, Italy, July 9-12.
- Squires, A., Wade, J., Watson, B., Bodner, D., Okutsu, M., Ingold, D., Reilly, R., Dominick, P., Gelosh, D. (2011), "Investigating an Innovative Approach for Developing Systems Engineering Curriculum: The Systems Engineering Experience Accelerator", Proceedings of the 2011 American Society for

Engineering Education (ASEE) Annual Conference and Exposition, Vancouver, BC, Canada, June 26-29, 2011.

- Squires, A., Wade, J., Dominick, P., Gelosh, D. (2011) "Building a Competency Taxonomy to Guide Experience Acceleration of Lead Program Systems Engineers", Proceedings from the Ninth Annual Conference on Systems Engineering Research (CSER), Redondo Beach, CA, April 14-16, 2011.
- Squires, A., Wade, J., Bodner, D., (2011), "Systems Engineering Experience Accelerator Workshop", National Defense Industrial Association (NDIA) 2011 14th Annual Systems Engineering Conference, San Diego, CA, October 24-27. (Presentation)

1 INTRODUCTION

Systems engineering educators are struggling to address workforce development needs required to meet the emerging challenges posed by increasing systems complexity (Bagg, et. al, 2003) and the widening gap in systems engineering expertise in the workforce (Charette, 2008). The Systems Engineering Experience Accelerator (SEEA) research project was conceived as a critical response to these needs and challenges. The project was initiated to validate the use of technology to potentially create an experiential, emotional state in the learner coupled with reflective learning so that time is effectively compressed and the learning process of a systems engineer (SE) is significantly accelerated as compared to the rate at which learning would occur naturally on the job. The purpose of the research project is to test the feasibility of a simulated approach for accelerating systems engineering competency development in the learner. An example of how the various concepts developed for the SEEA are related is shown in Figure 1.

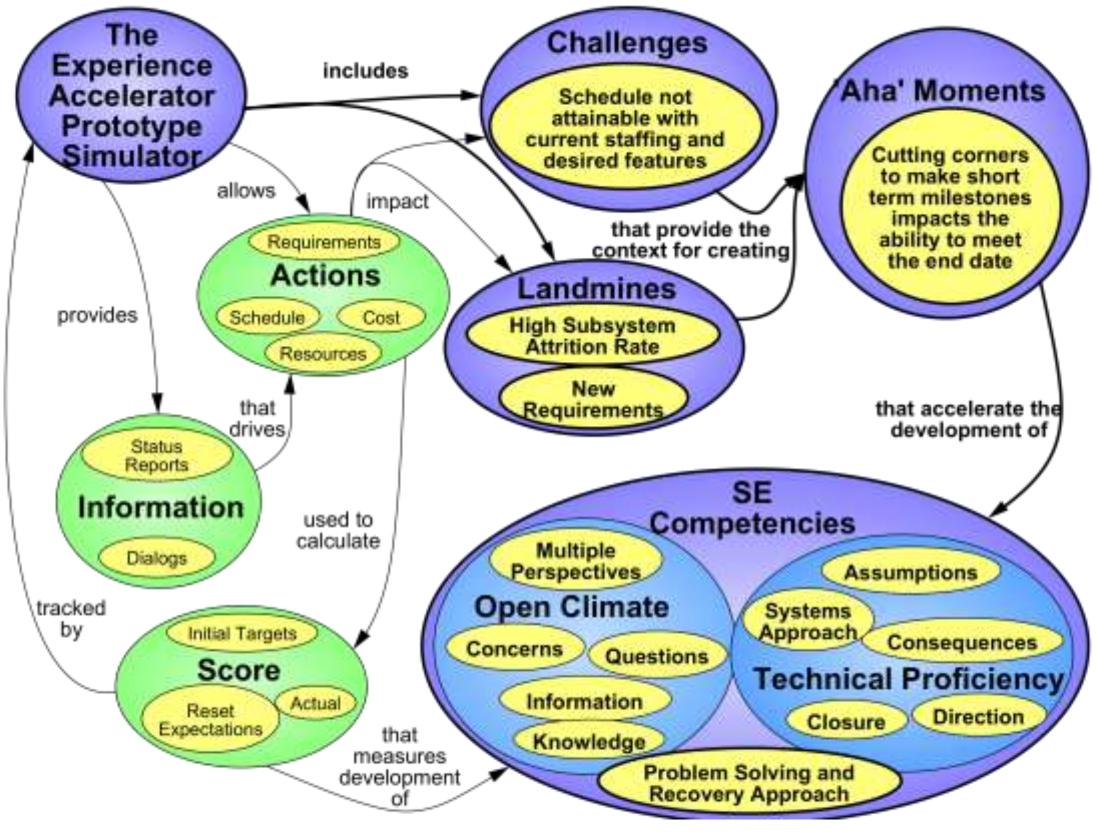


Figure 1: Notional Diagram of the SEEA Prototype Simulator

As shown, the development team had a threefold challenge to balance the development of the simulator technology that supports displayed content (shown in green) that, in turn, supports the developed concepts (shown in purple). The goal was to effectively create challenges and landmines that support the learner's experience of the necessary "Aha" moment. The intent was that by experiencing the "Aha" moment, the learner transitions to a more advanced level of understanding in the targeted competency, in this case "Problem Solving and Recovery Approach".

The testing of the prototype will support evaluation of the theoretical capabilities of the developed system and provide guidance for the continuing development of the SEEA simulator going forward.

1.1 Background and Motivation

In *The Art and Science of Systems Engineering*, Mr. Harold Bell, Director Advanced Planning and Analysis Division, NASA Office of Chief Engineer, is quoted as saying: "A great systems engineer completely understands and applies the art of leadership and has the experience and scar tissue from trying to earn the badge of leader from his or her team" (Ryschkewitsch, et. al, 2009, p. 2). Historically, competent systems engineers have developed their "scar tissue" by gaining the necessary insights and wisdom through both failures and successes, in an integrated real world environment. In the workplace, however, the learning events that result in the development of "scar tissue" are distributed, sometimes sparsely, over time. In addition, a common benchmark time for the development of a competent SE is a minimum of about 10-15 years (Dubey, 2006). Given there is a shortfall of SEs in the global workforce today (NDIA SE Division, 2010) and no readily available source of SEs to replace the top SEs in the retiring baby boomer generation, the time to develop competent SEs needs to be significantly shortened.

The primary goal of the SEEA, once it is developed, is to accelerate the maturation of SEs in the workforce by providing the opportunities to earn "scar tissue" through realistic, engaging simulation. These tailored experiences will allow the learner to feel the consequences of success and failure in a simulated environment so they can gain the necessary insights and wisdoms to mature as a SE, and yet not jeopardize the lives of others or compromise their careers. The initial target audience of the SEEA program is lead program SEs in the acquisition workforce who are required to effectively manage complex systems throughout their lifecycle from an acquisition/acquirer viewpoint in a typical program office. The initial focus is on maturing these leads to prepare them for executive assignments.

1.2 Project Goals and Success Metrics

Based on SEEA research team meetings and feedback provided by the sponsors, the team set specific goals and success metrics as summarized in the following sections.

1.2.1 Purpose

The ultimate purpose of the SEEA is to leverage technology to create an experiential, emotional state in the learner so that time is effectively compressed and the learning process of a systems engineer accelerated as compared to the rate at which learning would occur naturally on the job.

The purpose of this project is to develop a prototype of the SEEA that is focused on a small set of competencies, in order to evaluate the theoretical capabilities of that technology.

1.2.2 Hypothesis

By using technology we can create a simulation that will put the learner in an experiential, emotional state and effectively compress time and greatly accelerate the learning of a systems engineer faster than would occur naturally on the job.

1.2.3 Program Goals

The primary goal of the SEEA is to transform the development of systems engineers by creating a new paradigm capable of significantly reducing the time to mature and sustain a senior systems engineer while providing the skills necessary to address emerging systems challenges in an economically attractive manner.

The approach is to build insights and "wisdom" and hone decision making skills by:

- Creating a "safe", but realistic environment for decision making where decisions have programmatic and technical consequences
- Exposing the participants to job-relevant scenarios and problems
- Providing rapid feedback by accelerating time and experiencing the downstream consequences of the decisions made

Outcomes needed to achieve this goal include:

1. Moving the systems engineer to the next level of proficiency in one or more SE competencies as listed in the Systems Planning, Research Development, and Engineering (SPRDE) Systems Engineering (SE) and Program Systems Engineer (PSE) competency model, known as the SPRDE-SE/PSE.
2. Developing and maturing systems thinking skills.
3. Developing and maturing leadership skills.

1.2.4 Target Audience

The initial focus is on the Systems Engineering Executive Level skills of a DoD Lead Program Systems Engineer necessary to effectively manage complex systems throughout their lifecycle from an acquisition/acquirer viewpoint in a typical Project Management Office (PMO). The skills addressed may well complement or support those taught in senior program management courses. The SEEA targets the entire life-long learning of the Systems Engineer.

1.2.5 Success Metrics

Success of the Experience Accelerator prototype will be indicated with a positive result in the following areas:

- Experienced Lead Program Systems Engineers authenticate the EA and provide useful feedback on areas of improvement.
- Learners express general satisfaction with the learning experience.
- The potential for learners who successfully complete the training to be able to immediately implement lessons learned from the training experience to the job, assuming the culture allows this.

1.3 Management Plan/Technical Overview

The RT16 work plan is summarized in the next section, with the detail in the *Technical and Management Work Plan (A009)*. Program risks, addressed in the following section, were also reported on in the latter half of the project in the Monthly Status Reports (A008).

1.3.1 Research Activities

The major research activities that were completed in Increment 1 are as follows:

1. **Update Documentation and Planning:** This involves updating the Experience Design, Architecture, System Specification and Final Report from the base year of work in response to feedback from the final review. In addition, project plans will be created for the Increment 1 efforts. Deliverables include these following updated documents:
    i. Project Goals & Success Metrics
    ii. Concept of Operations
    iii. High-Level Architecture Specification
    iv. Systems Specification
    v. Experience Design Document
    vi. Final Report
    vii. Management Plan
    viii. Experience Accelerator White Paper

2. **Prototype Development:** This involves the development work necessary to address the gaps noted in the Phase I research. With the knowledge gained from the development of the base year Prototype, the architecture, technology, and content will be revisited and further honed such that it can be tested to evaluate its effectiveness.

a. **Experience Accelerator Technology: Presentation Engine, Experience Master, Challenge Control and Non-Player Character (NPC) Engine -** This activity will focus on the completion and refinement of the Experience Accelerator technology. This includes the virtual desktop system, simulating the user's desktop environment utilized to interface with the EA environment, the Experience Master which serves as the master controller for the experience, the Challenge Control which determines the type and level of challenge, and the NPC Engine which controls dialogue. This work will be done at Stevens with assistance from Purdue University *(Dr. George Kamberov, 2 Stevens and 1 Purdue Research Assistant.)*
b. **Simulation Engine –** This activity will focus on the development of the simulation technology, data files and graphics to provide the Experience Accelerator User with the results of their decisions and actions. This work will be done at Georgia Tech *(Dr. Douglas Bodner and 1 Georgia Tech Research Assistant.)*
c. **Content Creation Tools Development -** This activity will focus on the development of tools that can be used to increase the efficiency of the content development work. Some critical areas involve the creation of dialog, artifacts and events. This work will take place at Stevens, Georgia Tech and Purdue *(Dr. Jon Wade, Stevens, Purdue & Georgia Tech students.)*
d. **Experience Design - Artifacts & Dialog –** This activity will be done at Stevens Institute of Technology and by consultant subject matter experts and will focus on the development of EA content, including artifacts and NPC dialogue *(Dr. Jon Wade, Dr. Alice Squires, and multiple subject matter experts.)*

3. **Informal Evaluation and Formal Evaluation Development:** In the base year of this project, evaluation of the prototype was largely incorporated through sponsor and expert review. This phase will involve an informal evaluation process with actual students and a limited number of practitioners testing the Experience Accelerator during this development process. In addition, planning will be done for the more formal evaluations in the second phase of this Increment 1 (noted below). This activity will be completed at Purdue University and Stevens Institute of Technology *(Dr. Bill Watson and 1 Purdue Research Assistant, Dr. Peter Dominick, Dr. Richard Reilly.)*
   a. **Informal Prototype Evaluation –** This activity will consist of the creation of a short set of survey questions which will be used to provide feedback from Users of the prototype Experience Accelerator as it is refined through this phase. This information will be summarized in a short report.
   b. Design of evaluation experiment – This activity will consist in the development of a set of evaluation metrics and the design of the evaluation experiments. This work will require developing the means of pre- and post-testing the participants to determine not only their perception of the

value of the experience, but also to determine the level at which the targeted lessons were learned. Post-experience competency assessments will also be developed, along with additional pre- and post-test formal assessments to capture user perception of the prototype and measure experience impacts. This information will be summarized in a draft report.

6. **Prototype Evaluation:** In Increment Year 1 Phase 2, assessment will be elevated to involve formal assessment of the impact of the prototype. This involves piloting the prototype with a representative sample of learners (students and practitioners). This needs to be done such that the results are indicative of what would likely be achieved with a finished delivery vehicle on a representative group of participants. This activity will be completed at Purdue University and Stevens Institute of Technology *(Dr. Bill Watson and 1 Purdue Research Assistant, Stevens Sr. Researchers Dr. Peter Dominick, Dr. Richard Reilly, Dr. Alice Squires.)*
   6.1. **Plan Update:** Metrics, associated instruments and evaluation experiments will be refined based on review feedback at the conclusion of Phase 1. The result will be an update in the Plan for Formal Evaluation report.
   6.2. **Learner Identification:** Determination of targeted learners at DAU and SERC collaborating universities will be identified. Dates and logistics will be determined to support the evaluation efforts.
   6.3. **Prototype Evaluation:** This is the actual evaluation of the learners, as determined in the Evaluation Plan. These results will then need to be analyzed to determine the efficacy of the approach and how it will be refined for deployment.

13. 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 Increment Year 2. *(This activity will be completed by the entire project team, based at Stevens Institute of Technology, Purdue University and Georgia Tech.)*

1.3.2 Risk Management

In addition to the work activities, five top program risks were identified and tracked throughout the first year of the program in the following areas:

1. Project Management
2. Technology Development
3. Content Development
4. Integration
5. Evaluation

Mitigation strategies were put in place as outlined in detail in the following sections.

1.3.2.1 Risk 1: Project Management

Risk: Inability to support known and evolving customer/user feedback with current staff, budget and timeframe.
Mitigation: Create project plan with prioritized features list and periodically review and re-prioritize the (possibly evolving) list with stakeholders identifying and resolving potential conflicts as they arise. Incrementally implement and continuously integrate capability in priority order, to ensure that final system incorporates the most important features that can be implemented with the available resources, in the available time. As necessary find additional resources to do development work to fill identified critical gaps. If need be, extend the development timeframe with a zero cost program extension.

1.3.2.2 Risk 2: Technology Development

Risk: Inability to tradeoff long-term architecture and technology objectives (leading to successful open source support) vs. short-term prototype goals.
Mitigation: Identify the areas in which the base year prototype needs to be re-architected and designed to support the long-term objectives. This development work will be prioritized such that it can take place outside of the experience demonstration and evaluation process critical path to ensure appropriate capability is in place by the end of Increment 1.

1.3.2.3 Risk 3: Content Development

Risk: Inability to produce a prototype that provides a compelling experience, supports the desired learning and is seen to be authentic.
Mitigation: Develop and review a design experience document which is used to guide the development process. Create a prioritized list of additional features and capabilities to improve the authenticity and engagement of the experience. Iteratively develop dialogue and feedback used during simulation that is based on inputs from Subject Master Experts (SMEs). Produce dialog and artifact authoring tools that allow these activities to take place independently of the technology development process and provide more immediate feedback. Have subject matter experts (SE and Unmanned Aerial Vehicle (UAV)) and representatives of the target Learners go through the Experience throughout the development process providing continuous input. Include preliminary evaluation and feedback from prototype learner subjects during the iterative development process. Leverage the team's behavioral learning expertise and gaming experience during this process.

1.3.2.4 Risk 4: Integration

Risk: Inability to successfully integrate our many ideas, approaches, requirements and developed technology and design.
Mitigation: Document and review our top-level “story line” on a regular basis to ensure alignment with our development and evaluation efforts. Employ a modular,

loosely-coupled architecture that enables geographically-distributed developers to work independently. Define explicit, arms-length interfaces between modules to simplify integration. In particular, create interfaces and tools (as noted above) that allow the development of content to proceed independently of the technology development. Use continuous integration as part of an agile development process to identify and resolve integration problems early ("fail fast").

1.3.2.5 Risk 5: Evaluation

Risk: Inconclusive results due to threats to validity of Experimental design (inability to generalize results), limited availability of suitable subjects and insufficient literature to support development of evaluation instruments.
Mitigation: Leverage team's knowledge and contacts on research in relevant fields to formulate and evaluate instrumentation and design methodology. Leverage team (Purdue, Georgia Tech and Stevens), sponsor (DAU), and SERC connections (e.g., BKCASE and collaborating institutions), as well as recruitment through project research and presentations to identify diverse available subject populations. Explore development of new research instrumentation by synthesizing relevant literature should no suitable instrumentation be found in the literature.

2 Prototype Development

The following describes the development of the Experience Accelerator prototype in RT16 Increment 1.

2.1 Development Process

At the beginning of the baseline year, given the exploratory nature of the research, activities tended involve communication of the entire research team. While this was effective in providing open communication between all of the team members, it was relatively inefficient and resulted in large parts of the team waiting for the completion of a particular piece of the program. Towards the end of the baseline year, the program migrated to a weekly meeting with the SMEs for content, a weekly team meeting and technology meetings on an as needed basis. In Increment 1, this was further refined to the development of a code and content Release Train in which major code releases were synchronized. The following are the three major code releases for Increment 1:

- **V1: Improve first-year prototype**
  - Stabilize operation
  - Complete implementation of existing features
  - Improve interfaces to facilitate updates and modifications
  - Update to conform with architecture

- **V2: Refine first-year prototype based on evaluation feedback**
  - Desktop usability improvements

  - Improved artifacts and dialog
  - New dialog authoring tools and capabilities

- **V3: Add new features**
  - Earned Value Management (EVM) cost support
  - Variable reflection feedback to learner
  - Program actions feedback to learner
  - User interface additions and improvements
  - Preparation for open source support

It was the intention in the baseline year for the team to develop technology and content using an agile software development process on a single server site at Purdue. However, it was found to be quite difficult to set up an iterative development environment and workflow. A major difficulty was in getting the necessary access for the team at the internal Purdue server site. Another challenge is that academic researchers are not necessarily full-time and it is very difficult to find frequent synchronization points. As a result we had the challenge of not having each of the development teaming working in the same environment with the same releases of code. We also had a rather adhoc version control system which consisted of uploads/downloads to/from DropBox.

In Increment 1, we have established an integrated development work at Stevens. In addition to this we have moved toward the use of a more robust configuration management and defect ticketing system, most of which will be used in an open source environment. The following are the major elements of this system:

- Software:
  - Source control: Subversion
  - Project Development, Management and Tracking: TRAC
  - Hosting: Stevens' server
- Content:
  - Dialog files – Chat Mapper
  - Integration: Dropbox, manual upload to Subversion
  - Software upgrades: versioned release trains with major and minor releases
- Documentation:
  - Move to Wiki-site at version 2.0

While the intention is to move to a wiki-site for all documentation, this migration has not yet taken place due to priorities being on the experience design and technology development.

Synchronization between the teams is still provided during weekly status meetings, and one to one conversations such that everyone has the opportunity to provide feedback including an assessment of how those changes might impact the work in their area going forward -- before the decision to adopt the change is made.

Another lesson learned in the baseline year is that the various pieces of the Experience

Accelerator are closely linked and it is difficult to separate them. Significant effort was spent in Increment 1 to improve these interfaces such that artifacts and dialog can be created without the involvement of developers. This has allowed the teams to work rather independently. In addition, some tools have been created or procured that allows the content creators to test portions of what they are developing, e.g. dialog, as it is being developed.

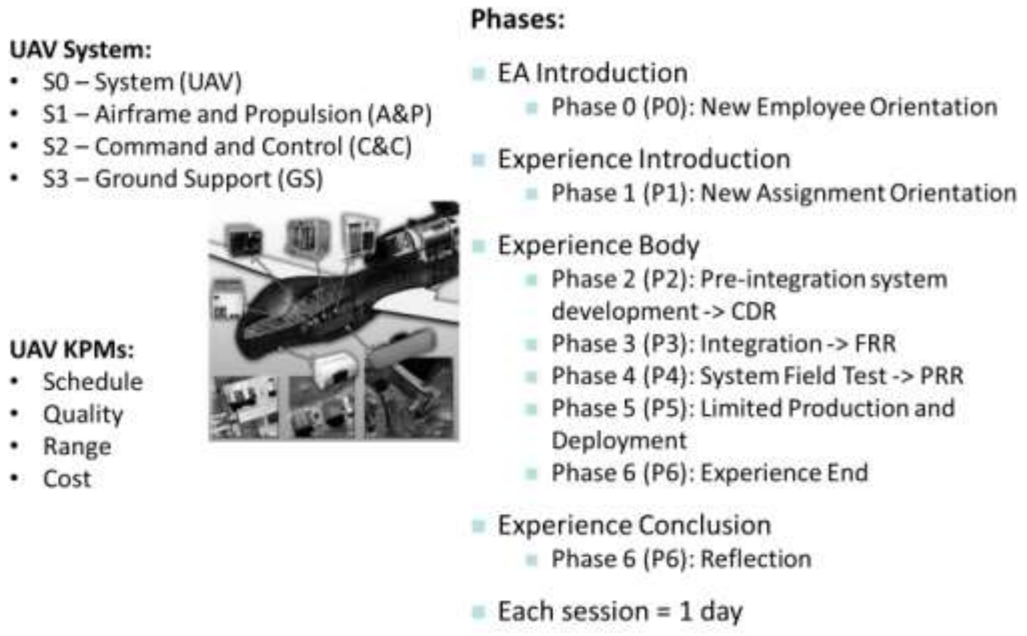
2.2 EXPERIENCE DESIGN

The following provides an overview of the experience design and the work that was accomplished in Increment 1. (For a detailed description of this work, see *The Experience Accelerator: Experience Design Document*.)

2.2.1 OVERVIEW

It is believed that accelerating the learning and maturation of Systems Engineers requires:

- Viewing a program through the entire lifecycle
- Seeing the relationships between elements of the system, and the system developing the system
- Encountering the challenges faced in a complex system development
- Being able to navigate through the "gray" zone
- Creating mental templates which can be applied to similar future situations



UAV System:
- S0 – System (UAV)
- S1 – Airframe and Propulsion (A&P)
- S2 – Command and Control (C&C)
- S3 – Ground Support (GS)

UAV KPMs:
- Schedule
- Quality
- Range
- Cost

Phases:
- EA Introduction
  - Phase 0 (P0): New Employee Orientation
- Experience Introduction
  - Phase 1 (P1): New Assignment Orientation
- Experience Body
  - Phase 2 (P2): Pre-integration system development -> CDR
  - Phase 3 (P3): Integration -> FRR
  - Phase 4 (P4): System Field Test -> PRR
  - Phase 5 (P5): Limited Production and Deployment
  - Phase 6 (P6): Experience End
- Experience Conclusion
  - Phase 6 (P6): Reflection
- Each session = 1 day

Figure 2: A Day in the Life of a PSE

The SE Experience initially developed in the baseline year focused on developing the systems thinking, and problem solving and recovery skills of a DoD Lead Program Systems Engineer. As shown in Figure 2, the SE Experience is designed to provide this learning by simulating the lifecycle of a UAV in which the learner is brought into the program after the Preliminary Design Review and is responsible for discovering the issues in the program and making the appropriate recommendations to correct the situation. The UAV system consists of three major subsystems of which 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 major key performance measures (KPMs) are schedule, quality, range and cost, with cost being newly added in Increment 1. 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. The difficulty of the experience is determined by the learner's self-evaluation of their competencies and their past history in the experience.

2.2.2 New Features and Capabilities

A number of new features and capabilities have been suggested by the sponsors of this program, by the subject matter experts and by the RT16 design and development team. While the fundamental basis of the experience described above has not changed, the key performance measure of cost has been added such that the basic contract structure is one of cost plus development vs. fixed cost. This is a significant change which adds a critical dimension to the entire experience.

The following reflects some the changes necessary to support this new feature of EVM:

- EVM related dialog, email and other correspondence to learner
- Learner recommendation forms and background information
- Determination of budget overruns due to schedule slips
- "Gold card" reference documentation
- UAV Program background document updates for budget information for each phase
- Development of cost models
- Cost model simulation

In addition to this new feature, there has been significant work done in completing and refining the existing experience. Some of these additions and refinements to the experience include the following:

- Feedback to the learner in the reflection phase based upon the learner's actions taken during the experience
- Feedback to learner on the actions that were taken based on their recommendations

- Program context information made available to the learner on the relative importance of each of the program KPMs

There were also a number of improvements in the User Interface including:

- Program dashboard on desktop for high-level system status
- Usability information/instruction
- Status report formats and presentation
- Emails and status reports were made to be persistent between sessions

Finally, there were improvements and updates made in number of artifact areas including:

- UAV Background – subsystems need to be updated
- UAV Status Charts explanation document
- Dialogs for various scenarios
- Extensible Mark-Up Language (XML) syntax for automated simulation graph generation

It is expected that the Experience will continue to be refined such that it is at the quality level necessary or pilot use by the end of the Increment 2.

2.3 Technology Development

2.3.1 Overview

The EA game engine has two components: the runtime engine and the tools suite. The tools suite includes the Experience Development Tools described in Section 2.5 and Simulation Engine-related tools described in the Section 2.4.

In this section we will describe the EA Run Time Engine component (EARTE). As shown in Figure 3 the EARTE has a layered architecture client-server incorporating the following modules:

- **Experience Master:** contains the overall Experience state and provides control and sequencing for the other major EA modules.

- **Challenge Control:** contains the Learner profiles and Experience history logs and leverages these in conjunction with the competency taxonomy and 'Aha' moments to determine the appropriate challenges and landmines for each Learner.

- **Simulation Engine:** determines the future state of the system and outputs to be presented to the Learner.

- **Non-Player Characters (NPC) Engine:** represents other non-player characters in the simulation, and creates and assembles the content for Learner interactions.

- **Presentation Engine:** accepts inputs from the Learner and provides the presentation of the Experience interface to the Learner.

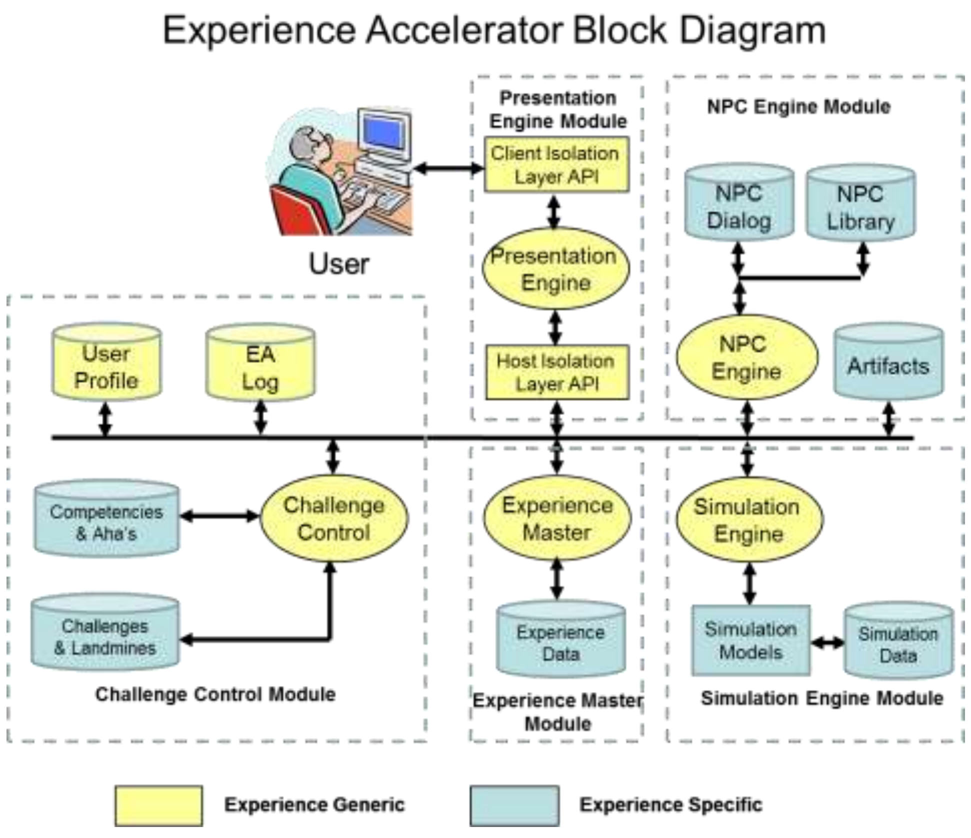


Figure 3: Experience Accelerator Logical Block Diagram

The EARTE is a multiuser architecture for internet gaming. It has light clients (currently implemented in FLASH) and a Java server which runs multiple game instances concurrently. In order to support as wide as possible range of EA games and scenarios the EARTE does not incorporate a simulation engine, but rather the NPC engine provides a framework to interface with 3d-party simulation engines. For more on the simulation see Section 2.4 Simulation Engine.

During Increment 1 the technology development team rebuilt the EARTE to achieve architecture adherence and improved runtime performance stability. In addition a number of new features and capabilities have been implemented in the last year. The following section provides an overview of the work that arose from suggested enhancements, fixes, and moving to a larger-server/thinner-client architecture.

2.3.2 NEW FEATURE AND CAPABILITIES

2.3.2.1 Multi-User

The server now runs a separate experience master instance for each client. In addition, each learner stores simulator related files in their own folders. These additions allow multiple clients to now use the Experience Accelerator concurrently.

2.3.2.2 Simulator Integration

Instead of a single hard-coded play through, the simulator is now used by the system and learner actions to affect the experience. This happens in the form of:

- Simulator Input – the learner's selections in recommendation forms are sent to the server
- Simulator Output – Charts and state information (schedule, budget, defects, etc.) are received from the simulator and used to determine a learner's progress through an experience

2.3.2.3 Dialog System

ChatMapper output files can now be parsed and used by the system for NPC dialog.

2.3.2.4 Persistence Support

Learner progress is more thoroughly recorded and persisted allowing a learner to continue where they left off after logging out and returning at a later date. Persisted information includes:

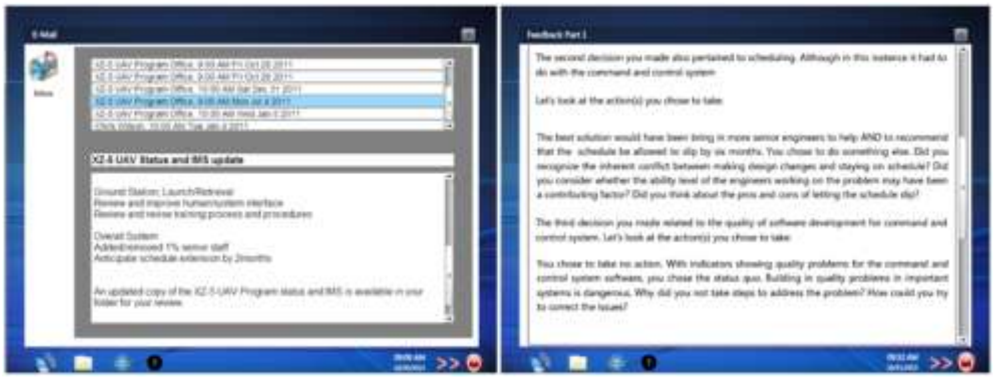
- State information (schedule, budget, quality, etc.)
- Events such as emails and voicemails
- Learner input in forms
- Feedback to the learner in status update emails, and reflection documents

2.3.2.5 Feedback

The learner can now receive feedback from the system in relation to how his/her actions have affected his/her progress through the experience. Mechanisms for feedback include:

- Visual dashboard on the desktop displays current status for quality, budget, schedule, and range

- Status update emails containing a recap of the actions chosen by the learner in the previous recommendation form (as shown in Figure 4)
- Reflection documents in Phase 7 give textual feedback based on actions taken by the learner during an entire experience



XZ-5 UAV Program Office, 9:00 AM Fri Oct 28 2011
XZ-5 UAV Program Office, 9:00 AM Fri Oct 28 2011
XZ-5 UAV Program Office, 10:00 AM Sat Dec 31 2011
XZ-5 UAV Program Office, 9:00 AM Mon Jul 4 2011
XZ-5 UAV Program Office, 10:30 AM Wed Jan 5 2011
Chris Wilson, 10:00 AM Tue Jan 4 2011

XZ-5 UAV Status and IMS update

Ground Station: Launch/Retrieval
Review and improve human/system interface
Review and revise training process and procedures

Overall System
Added/removed 1% senior staff
Anticipate schedule extension by 2months

An updated copy of the XZ-5-UAV Program status and IMS is available in your folder for your review.

The second decision you made also pertained to scheduling. Although in this instance it had to do with the command and control system.

Let's look at the action(s) you chose to take.

The best solution would have been bring in more senior engineers to help AND to recommend that the schedule be allowed to slip by six months. You chose to do something else. Did you recognize the inherent conflict between making design changes and staying on schedule? Did you consider whether the ability level of the engineers working on the problem may have been a contributing factor? Did you think about the pros and cons of letting the schedule slip?

The third decision you made related to the quality of software development for command and control system. Let's look at the action(s) you chose to take.

You chose to take no action. With indicators showing quality problems for the command and control system software, you chose the status quo. Building in quality problems in important systems is dangerous. Why did you not take steps to address the problem? How could you try to correct the issues?

Figure 4: Feedback via Status Update E-mail

2.3.2.6 Network Improvements

Several network problems needed to be addressed throughout the year. Due to poor connections, clients could crash when attempting to read un-arrived data from the server or be unable to reconnect after disconnecting due to a timeout or crash. Data sent between the server and client is now packaged in discrete messages using a small header and read/written as a whole for stability. Clients are also able to reconnect to the server after having an unexpected disconnect. One issue remaining is the inability to use computers that block ports used by the client. We are currently investigating administrative as well as technical solutions to this problem.

2.4 Simulation Engine

The following provides an overview of the simulator and the work that was accomplished in Increment 1. (For a detailed description of this work, see *The Experience Accelerator: Architecture Specification* and *The Experience Accelerator: Experience Design Document*.)

2.4.1 Overview

The simulator module maintains the quantitative state of the program and advances this state in between learner cycles, based on changes that the learner might make to program parameters and variables. The simulator uses the well-known paradigm of system dynamics (Sterman, 2000), which allows the modeling of system behaviors over time featuring flows, lags, feedback loops and other non-linear phenomena. System dynamics has been employed in modeling and simulating systems engineering processes, most notably those involved in software development (Madachy, 2008). The simulator module contains the following components:

- The *execution engine* advances the program state via simulation and records values of updated program variables over time. It returns these values to the experience master, which can use them for scoring. It also generates output charts for the learner reflecting updated program status and progress. It maintains state variables in between simulation cycles and phases via various files, and it also reads from files provided by the experience master that correspond to learner decisions about which program parameters or variables should be changed. It consists of Java-based, open source code based on pre-existing products.

- Various *simulation models* specify program parameters, state variables, their behavior over time, and their interactions. These are implemented as XML files and are executed by the execution engine.
- *Output charts* provide graphs of program state variables over time reflecting program status, behavior and indicators. These charts are in the form of graphics files fed to the experience master based on an XML chart specification. The specification defines the chart format, which is then filled in by values of program variables over time.

A simulation model specifies the behavior over time of a system of interest in terms of:

- Variables – quantities that change over time during model execution,
- Parameters – quantities that do not change over time during model execution (but that can be changed via user action),
- Rates – the rates at which a variable changes (which may in turn change over time), and
- Auxiliary relationships – equations used to model the relationships between parameters, variables and rates.

These are expressed via an XML-based specification. A simulation model is provided for each of the different phases of the experience. Each model has a variety of sub-models that represent important elements of the acquisition program in terms of the engineering workforce and its functional areas (e.g., design versus quality assurance), cost accruals and budgets, defects (resolved, unresolved and unidentified), UAV system performance estimates, and progress toward entrance criteria for critical program reviews at the end of each phase.

Of course, there is significant interaction between the sub-models. For instance, the workforce accrues labor cost over time. This may be at the scheduled accrual rate, or it may be at a higher accrual rate, especially if the schedule is falling behind. The productivity of the workforce impacts the progress toward meeting entrance criteria for reviews. Performance measure estimates for the UAV system often degrade over time, unless workforce resources are called to address problems. Below is a summary of the various sub-models in each phase.

- Phase-2 – Pre-integration
  - The UAV sub-model tracks estimated range plus underlying estimates of technical performance measures such as drag, propulsion efficiency, overall weight and sub-system weight.

  - The design workforce sub-model converts full-time equivalent engineering headcounts (at two levels of experience) to progress on various tasks using productivity, training and communication overhead effects. There is a design workforce for each of the sub-system contractors plus the prime contractor.
  - The entrance criteria sub-model represents the progress over time for the various entrance criteria needed for Critical Design Review (CDR). The labor sub-model results feed this progress. Each contractor's workforce provides a contribution to progress on each criterion, depending on its involvement. Thus, each workforce typically splits its results among entrance criteria.
  - The quality sub-model tracks the number of defects encountered over time plus the workforce that addresses these defects. Defects must first be identified and the resolved. Design reviews can help identify defects. It should be noted that only software quality for the command-and-control system is currently modeled.
  - The EVM sub-model is used to track the accrual of costs over time and compares these to the expected costs over time. Thus, it can be determined where the program is relative to schedule and budget. There is an EVM sub-model for each contractor, including the prime contractor, and these roll up into an overall EVM sub-model.

- Phase-3 – System integration
  - The UAV sub-model operates quite similarly to Phase-2.
  - The design workforce sub-model is similar to that of Phase-2 except that the prime contractor is more heavily involved due to the integration nature of the work.
  - The entrance criteria sub-model is similar to that of Phase-2, except that the specific criteria are for Flight Readiness Review (FRR), and the contributions of each workforce are consequently different.
  - The quality sub-model operates similarly to that of Phase-2. Once again, only software quality for the command-and-control system is currently modeled.
  - Finally, the EVM sub-model operates similarly to the sub-model of Phase-2. The budget and cost parameters are, of course, different.

- Phase-4 – Flight test
  - The UAV sub-model operates quite similarly to Phase-2 and Phase-3.
  - The workforce sub-model continues to operate similarly to that of Phase-2 and Phase-3, except that it is addressing test rather than design.
  - The entrance criteria sub-model is similar to that of Phase-2 and Phase-3, except that the specific criteria are for Production Readiness Review (PRR), and the contributions of each workforce are consequently different.
  - The quality sub-model operates similarly to those of Phase-2 and Phase-3. Of course, during flight test, it is expected that there are fewer quality issues than before.

  - Finally, the EVM sub-model operates similarly to the sub-models of Phase-2 and Phase-3. The budget and cost parameters are, of course, different.

- Phase-5 – Low rate initial production (LRIP)
  - The UAV sub-model operates quite similarly to previous phases.
  - Instead of a workforce model, there is a production model. Incremental progress on production of each air vehicle sub-system (discretized into release of finished sub-systems). These are then available for integration into an air vehicle. Thus, it is desired that the production rates of each air vehicle sub-system (airframe and command-and-control) are roughly equal over time. Ground stations are produced separately. Finished air vehicles and ground station units are then compared against the production target for each in this LRIP.
  - There is no entrance criteria sub-model.
  - There is no quality sub-model, although one could be introduced for production defect issues.
  - Finally, the earned EVM sub-model operates similarly to the sub-model of previous phases. The budget and cost parameters are, of course, different.

The simulator provides output artifact charts so that the leaner can see the results of his/her decisions integrated with the continued progress of the program. These charts fall into the following categories.

- <u>Acquisition program baseline (APB) metrics</u>. These metrics track program performance related to baseline plans as the program moves from design and development to production.
  - Key performance parameters and technical performance measures (KPPs/TPMs). These charts track estimates and actual for metrics related to performance on program requirements.
  - Other important system attributes. These metrics track non-requirements performance, typically relate to production cost per unit, maintenance costs, training costs, operational personnel required. These are tracked as the program moves from design and development to production.
- <u>Quality</u>. These charts track key indicators of quality apart from program requirements.
- <u>Review meeting entrance criteria</u>. These charts track progress on meeting the various entrance criteria associated with review meetings. They serve as indicators of program schedule.
- <u>Cost and budget</u>. These charts track cost and schedule performance relative to budget using the concept of EVM. EVM utilizes a number of metrics over time to track performance.
- <u>Initial operating capability (IOC) metrics</u>. These charts track system production and delivery status in terms of meeting IOC (number of systems to be delivered by scheduled delivery date).

3 As shown in

3.1 Lessons Learned

The following is a summary of the lessons learned from the RT16 team. The lessons are divided into the following four categories:

1. Competencies, Learning and Content
2. Complexity/Effort vs. Authenticity/Learning
3. Technology
4. R&D Processes & Tools

Lessons learned in each of these areas are described below along with an approach to mitigate negative impacts in Increment 2. These lessons learned have impacted the nature of the future work, the processes used, and the identified risk factors moving forward.

3.1.1 Competencies, Learning and Content

LL1.1: ***Systems Thinking Evaluation*** - It is very difficult to evaluate capabilities in systems thinking. After an extensive literature search, very little was found in how to test system thinking and technical problem identification and resolution skills. Additional research will need to be done to develop a means of testing these capabilities. Our approach has been to use a Delphi approach in which subject matter experts review behavior and grade them with respect to competency levels.

LL1.2: ***Learning and Concept Capability Evaluation*** – It is difficult to determine if the learner has actually learned and can apply the identified concepts. Even if the learner goes through the Experience more than once and shows improvement, it is not clear whether they have just learned how to best this particular experience or if they have learned the concept and how to apply it in future situations. It would be desired to develop and exam in a different medium which could be used for pre- and post-testing to assess the results.

3.1.2 Technology

LL2.1: ***Client Graphic Technology Migration*** - While Flash is currently has the most productive environments in which to develop graphical content and is free on the client, the development licensing can be expensive (approximately $200 for individual educational licenses, and $700 for individual commercial licenses) and Flash does not work on iPads which represent a major client technology base. Once the open competing technology, HTML5, has development environments which provide the same level of productivity of Flash, it would be advantageous to migrate to the new technology.

LL2.2: ***Client/Server Interface Reliability*** - We had a number of problems getting the EA client/server interface reliably in the presence of an unreliable network. While the system now works, getting it to this state required a significant amount of debug and redesign work. In Increment 2, we will need to review the design and update it and the implementation to provide a cleaner, more supportable system.

LL2.3: ***UI Look and Feel*** - Creating a professional look and feel for a virtual desktop continues to be a non-trivial undertaking. We received feedback from the subject matter experts on the difficulties that they faced in using the browser and virtual desktop. We have added a text based support online, but we may need to create an overall experience training video as well. We have made and will continue to make changes to address this.

3.1.3 COMPLEXITY/EFFORT VS. AUTHENTICITY/LEARNING

Defense acquisition is a very complex enterprise, with many processes, actors and organizations. Selecting a subset of these to represent in the Experience Accelerator involves numerous design trade-offs to address the interests of (i) the learner community, that wants a realistic but not overwhelming experience, (ii) the education community, that wants realism in support of learning objectives, (iii) the acquisition community, that wants its various aspects represented faithfully, and (iv) the developer community, that wants to provide a useful product while managing complexity, schedule and cost. One of the biggest challenges was to create an authentic, learning experience while managing complexity and the amount of content that needed to be created. During this past year we have continued to learn in these areas.

LL3.1: ***Challenges & Landmines*** - There are an almost infinite number of ways in which a program can fail; combinatorial explosion is a major challenge. This is not so much of a challenge for the simulator, but this continues to be a major issue for the creation of artifacts and dialog which can support these to allow the Learner to make sense of the situation. While we created a catalog of a large number of frequently encountered Challenges and Landmines, for the prototype we implemented just a few of the most likely ones in the areas of aviation hardware and software. During this past year we have added challenges in the area of budget management.

LL3.2: ***KPMs*** - KPMs drive the amount of information that needs to be simulated, the amount of artifacts for background information, dialog and Learner recommendations. During Increment 1 we have added cost to the existing KPMs of schedule, quality and capabilities (range) which has required the development of a plausible cost model and supporting artifacts and dialog. While we have implemented a relatively simple EVM system, it is clear that developing one similar to what would be found in a DoD program office would be very challenging. We will have to see if what we developed has sufficient authenticity to provide the desired learning.

LL3.3: ***Feedback to/from Learner*** – In Increment 2 we increased the amount of feedback to the learner in the form of a confirmation of the actions which were taken as a result of the recommendations that the learner made throughout the experience, and in the feedback that the learner received at the end of the experience. While the recommendation form of feedback was fairly straightforward in its implementation as it represented objective fact, the reflection feedback was more complicated in that it was both subjective in nature and the multiple actions that the learner has taken are inter-related. The process that was used involved noted the decisions that were deemed appropriate, inappropriate and neutral and feedback on each decision was made independently. Understanding the interactions between these is far more complex. One possibility for future work is to record how the learner's behave and determine if there are specific patterns of behavior which can be identified and feedback given to the learner which is most appropriate to the pattern of behavior.

3.1.4 R&D Processes & Tools

At the outset of this project, given the exploratory nature of the research, open communication was established between all of the team members through the use of a Wiki site. However, it was discovered that the overhead in learning how to use and navigate through the site outweighed its advantages so we migrated to the use of a simple DropBox technology for documents and development code, and Webex for group meetings which were held once a week. As the team more than doubled in size and became more specialized, this mode of communication became a bit cumbersome. We eventually migrated to a weekly meeting with the SMEs for content, a weekly team meeting and technology meetings on an as needed basis. While the intention was for the team to develop technology and content using an agile software development process on a single server site at Purdue, it was more difficult than expected to set up an iterative development environment and workflow. During the Increment 1, we moved the development to Stevens and used a set of technologies for bug tracking and version control that are standard in the open source software world. The following are some of the lessons that were learned in Increment 1 after moving to this new environment.

LL4.1: ***Change Management*** - In Increment 1 we started using a formalized ticketing system and process. Due to the instability of the design, we quickly created a huge number of tickets for the developers. Due to this large backlog, we slowly moved away from this system, and started contacting the developers directly on our needs. In retrospect, this was a mistake as it created challenges in tracking the work in queue, in process and completed. Now that the design and implementation has stabilized, we intend to move back to the more formal approach that should prepare for open source distribution and should help us with our challenges in configuration management (see LL4.2).

LL4.2: ***Configuration Management*** - Due to the large number of design files, artifacts, simulation parameters and the like, configuration management has become a great challenge. One of the issues is that the Experience Design document is overly generalized and it is sometimes difficult to know exactly what has been implemented. Another issue is that we do not have a centralized place that is a source for all of the files. The issue goes beyond the software build and reflects all of the artifacts and code. To address this issue we will be creating a single design document that will be used with a work tracking tool to provide configuration management for the program. Reconciliation and updates in these two sources of data will be done on a weekly basis to ensure that control is maintained over the design.

LL4.3: ***Verification and Validation*** - The system that we are developing has become very complex making it quite difficult to test. In addition, resources are quite limited such that the testing that we do is ad hoc and carried out by the person who is implementing the code. One possible solution to this problem is to develop an automated tool for verification. There are simulation packages available that have the ability to automate tests that provide the ability to run a suite of simulations and then check all of the results in parallel. In addition, we should create a pool of people who are outside the implementation team who can test the overall Experience Accelerator system.

LL4.4: ***Configuration Dependency Testing*** - There are a number of implementation issues that are configuration dependent, yet may not be discovered through our normal testing procedures which are limited in their configuration variations. Addressing this will require automated test beds and the prerequisite software and hardware support.

LL4.5: ***Content Creation*** - It is quite difficult to create content, particularly dialog, without actually going through the experience to see what it is like. Unfortunately, this was not possible during much of Increment 1 due to instabilities in the code base and the recent development of the EVM features. In addition, as a content creator you need the ability to see the effects of the experience from the beginning to the end. It would be quite useful to have a developer mode in which one could quickly see the projected results of an entire Experience based on some scripted learner behaviors. These behaviors could be based on recorded behaviors of actual learners. This information could also be useful in the identification of classes of behavior types, how to identify them, and how to improve their performance.

LL4.6: ***Architectural Conformance*** - There was a lack of conformance between the implementation, heavy client, and the architecture, thin client, that was developed in the base line year. As a result, there was a good deal of rework that had to be completed by the Stevens development team. It was clear that we needed someone with a computer science and game design background to manage the development team. This action was taken and the issues have been remedied.

LL4.7: ***Academic Software Development*** - Software is difficult in an academic setting as the workforce is largely composed of students who quickly come and go. In addition, since the development team is so small, consisting of 2-3 member, transitions are particularly difficult as the loss of a single member is a large fraction of the team, there is little overlap with other members, and it is not easy to quickly recruit and bring up to speed new members. It is clear that there needs to be at least one member of the team who understands the entire design and is there long term to provide continuity to the team. This has been accomplished on the technology, simulation and experience design teams through continuity of technically capable faculty researchers. In addition, longer term support is also in place with a graduate student in technology development.

LL4.8: ***Automation of Repetitive Tasks*** - There are a number of repetitive tasks that consume a great deal of our developers' time and effort. One example is the conversion of documents from one format to another which then need to be placed in the appropriate design file. There is a strong need for the development of tools to automate these tedious processes. There is a need for tools to automate tedious processes. A conceptual design has been created for this tool.

LL4.9: ***System Interfaces and Partitioning*** - Another lesson learned in the baseline year and repeated in Increment 1 is the importance of partitioning in the system and creating interfaces such that artifacts and dialog can be created without the involvement of developers. This was increasingly successful in Increment 1 as tools were created to allow the simulation team to create graphs and charts that could be automatically incorporated into the system

3.2 FUTURE WORK

The plan is to preserve most of the EA team going forward with the potential inclusion of researchers from Carnegie-Mellon University (CMU) or University of Southern California (USC) in developing multi-learner technology. The organization chart for RT16 is shown in **Error! Reference source not found.**. The team plans to leverage SMEs and other resources to test and validate the system through a pilot program.

Follow-on work has been defined for Increment 2 that is focused on the following:

- Prototype Evaluation (cont.)
- Pilot System Development
- Pilot System Evaluation
- Open Source Preparation and Deployment
- Multi-learner Technology Development
- External Developer Engagement

For more detail on the follow-on work plan, see *Developing Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap, Increment 2 Technical and Management Work Plan.*

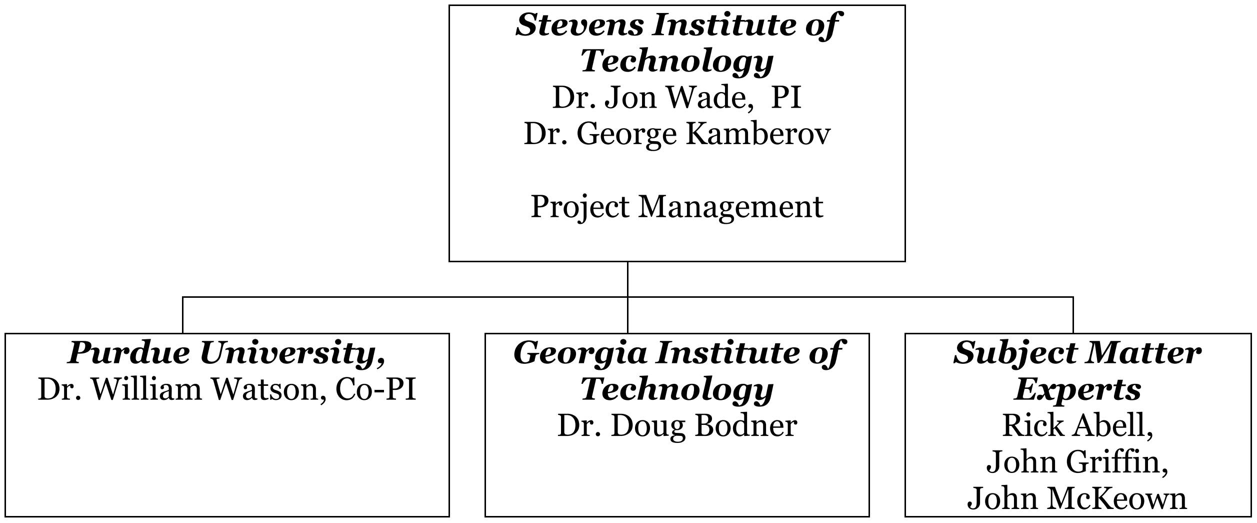


Figure 10: SERC RT16 Organization Chart

, a particular chart typically has values over time of:

- One or more metrics of interest,
- A target or historical benchmark for each metric, and
- A plan for where each metric is expected to be given expected program progress and performance.

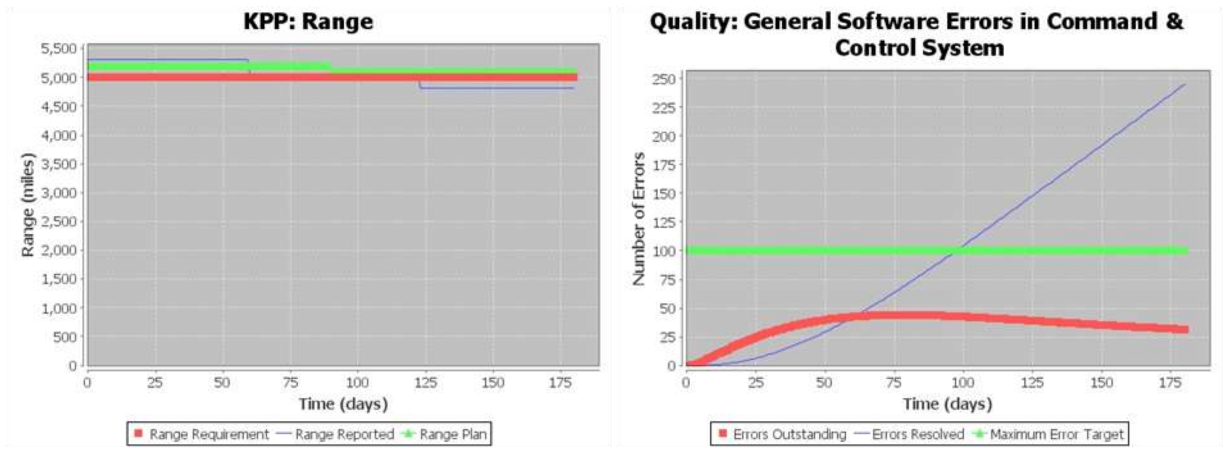


Figure 5: Example Charts

3.2.1 New Features and Capabilities

A number of new simulator features and capabilities were implemented this past year. These resulted from continued progress, internal needs plus input from subject matter experts on program behavior and output chart realism. This section summarizes this work by simulator module component. Note that many improvements have interdependencies between components (e.g., changes to chart format require changes to execution code needed to generate new formats).

3.2.1.1 Execution Engine

The execution engine was extended in a number of areas, mostly to facilitate desired model and chart features.

- Functions were added to provide discrete variables and outputs from models. The existing open-source product used did not have this feature. The system dynamics paradigm is fundamentally based on a continuous variable representation.
- Improvements were made to the way values of global simulation variables were transferred between experience phases. Since each phase uses a different simulation model, it is important that variables that span more than one model have their values transferred between the models.
- The chart generation code was reconfigured to read from an XML syntax for the chart specification (described in further detail below). This included updates to support scaling of the axes of charts for readability purposes.

3.2.1.2 Simulation Models

Each of the existing simulation models were enhanced with further detail.

- A simulation model was provided to provide a beginning status and set of output charts for the user in Phase-1 of the experience (i.e., program orientation). This model is executed prior to the orientation so that the learner can see the program status since Preliminary Design Review (PDR), before making any decisions in Phase-2. Previously, this had been a static display of charts. Now, models with different parameters can be executed to provide learners with different starting points in the experience.
- The range sub-model was enhanced with a discrete variable reporting the current range to the user, as it was recommended by subject matter experts that range is reported only in intervals (e.g., of approximately a month).
- The entrance criteria sub-models were linked more extensively with the workforce sub-models.
- The Phase-5 model was enhanced with a supply chain model that reflects the need to have all sub-systems produced and ready to assemble into an air vehicle, rather than the previous model of air vehicle production at the system level.

3.2.1.3 Simulation Output Specification

One important task for this past year was the creation of a way to specify the output charts so that they were code-independent (i.e., charts could be modified and new charts added easily). This is done via development an XML syntax that allows specification of the following:

- Chart title,
- X-axis and y-axis labels,
- Simulation variables to be graphed (metrics and targets),
- Planned values for metrics to be graphed (pre-specified), and
- Whether the chart should be presented such that the y-axis is scaled to include only those values contained in the range of data (e.g., for data in the range of 5,000 to 5,200, the y-axis would start at 5,000 and go to 5,200)

The last point is included based on feedback that it is difficult to distinguish graphs of different variables that are clustered together when the y-axis might be scaled starting at zero.

Currently, the chart specification for a particular simulation model resides in a companion file to that model's XML file.

3.2.1.4 Output Chart Artifacts

A number of new charts and chart enhancements were added this past year. A summary is included below.

- Technical performance measures for drag, overall air vehicle weight, air vehicle sub-system weight and propulsion efficiency,
- EVM metrics reflecting cost and budget for sub-systems and overall program, as well as cost and schedule variances, and
- Sub-system production charts for Phase-5 that include production targets along with sub-systems produced over time.

3.2.1.5 Earned Value Management

A relatively simple model of EVM was implemented in each experience phase. It should be noted that this is not an EVM system, nor does the model contain the level of detail that would be required to simulate a full EVM system, as that would be outside the scope of this project. EVM relies on three fundamental quantities that are tracked over time.

- Budgeted Cost of Work Scheduled (BCWS) – Expectation of cost incurred according to budgeted cost rates and pre-specified schedule.
- Budgeted Cost of Work Performed (BCWP) – Expectation of cost incurred according to budgeted rates and actual schedule performance.

- Actual Cost of Work Performed (ACWP) – Actual costs incurred according to actual cost rates and actual schedule performance.

An example chart with these values is shown below in **Error! Reference source not found.**.

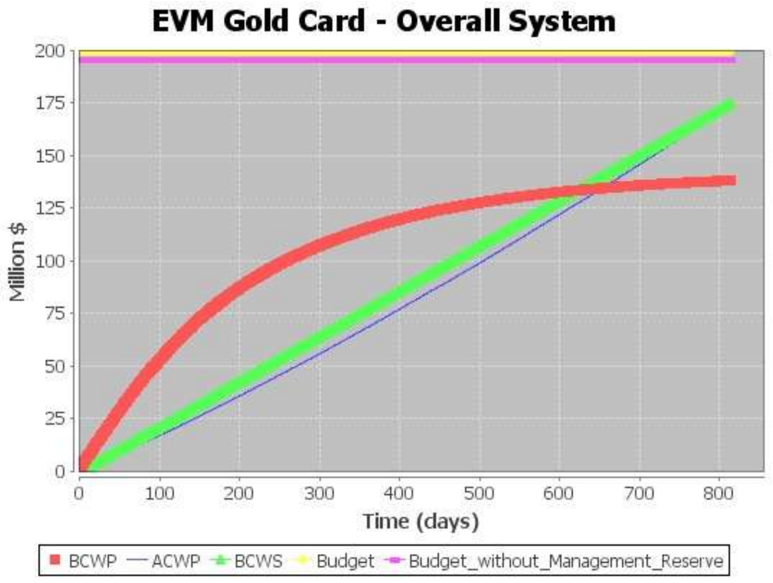


Figure 6: EVM Gold Card

Deviations in schedule and cost are then captured by various metrics that compare these three quantities. Two metrics capture the magnitudes of deviations.

- Schedule Variance (SV) – The difference between BCWP and BCWS, this quantity detects how well the program is doing relative to schedule. If BCWP > BCWS, this implies that the work performed is ahead of schedule. Thus, SV > 0 is generally a positive indicator of program performance.
- Cost Variance (CV) – The difference between BCWP and ACWP, this quantity detects how well the program is doing relative to cost. If BCWP > ACWP, then the actual cost is less than the budgeted expectation. Thus, CV > 0 is generally a positive indicator of program performance.

Two additional metrics capture the relative percentages of deviations.

- Schedule Performance Index (SPI) – The ratio of BCWP to BCWS. SPI > 0 is generally a positive indicator of program performance.
- Cost Performance Index (CPI) – The ratio of BCWP to ACWP. SPI > 0 is generally a positive indicator of program performance.

The EVM sub-model tracks schedule and cost separately for each sub-contractor (i.e., each sub-system). The prime contractor's schedule and cost are tracked separately, as well. Then the overall program cost and schedule consist of the combined costs and schedules of the prime contractor and sub-contractors. This is done by summing the values of BCWS, BCWP and ACWP across the prime contractor and sub-contractors.

Similarly, schedule variance, cost variance, schedule performance index and cost performance index are tracked separately for each sub-contractor and for the prime contractor using the respective values of BCWS, BCWP and ACWP for each.

3.3 EXPERIENCE DEVELOPMENT TOOLS

3.3.1 OVERVIEW

One of the major objectives for this program is to provide the means whereby new experiences can be efficiently created in a cost effective method. It is believed that this capability is essential to the development of a thriving open source foundation community. (These needs will be assessed in Increment 2.)

To make this possible, it is our objective to eliminate the need for content creators to do programming. An analysis was made of each of the major areas of design and development work to develop new experiences which are shown below. (Note: key areas that are believed to provide the best cost/benefit are noted in preceded by '*', noted with a number and shown in italics.)

- 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
  - **1 Experience Phases & Time specification*
    - *Conceptual design completed*
  - **2 Challenges, Landmines and Levels specification & triggering*
    - *Conceptual design completed*
  - Challenge Control -> simulation parameter setting
  - Evidence of Challenges and Landmines
  - Mitigating Actions & Effects
  - Relationships between Competencies/Aha's, Challenges/Landmines, Mitigating Actions & Effects
- Reflection

    - Feedback format
    - Scoring
- Artifacts
    - Background information
    - **3 Project reviews*
        - *Templates created*
    - **4 Learner recommendation forms*
        - *Templates created*
    - **5 Simulation status reports*
        - *XML syntax created for automated graph generation*
    - Email
    - **6 Dialog*
        - *External tool integrated into system*
    - Mentoring
    - Evaluation feedback
- Simulation
    - **7 Models construction*
        - *Research topic*
    - **8 Parameter setting*
        - *Research topic*
- Overall
    - **9 Process and tools documentation*
    - **10 Artifact entry*
        - *Conceptual design completed*
    - **11 Experience learning evaluation*
        - *Research topic*

Of the 11 top areas, Dialog creation has been supported with tools, Project reviews, Learner recommendation and Simulation status forms have been addressed with templates. Conceptual designs have been created for Experience Phases & Time specification, Challenges, Landmines and Levels specification & triggering, and Artifact entry, while Model construction and Parameter setting remain research topics. All of these are described in more detail below.

3.3.2 Dialog Support

One of the major challenges in the single-learner Experience is the generation of dialog for all of the machine supported human roles with whom the learner interacts. Thus, of the identified areas, the challenge of generating and managing dialog was one of the greatest. The dialog engine that was created for the Experience Accelerator was designed to use XML as dialog input. To support this, a parser was developed to take text as an input and create XML in the proper format. However, it was discovered that the text based development of dialog was challenging, and a Graphic User Interface (GUI) based tool would be much more efficient.

The desired tool would provide a non-technical writer the capability to create, test and manage non-linear dialog with the following characteristics:

- Has a user-friendly GUI
- Is easy to construct visualized branching graphs
- Is easy to define control conditions
- Provides the capability to quickly simulate inputs
- Provides custom exporters including XML (for EA Dialog Engine)
- Is free or low-cost

After a researching the commercially available toolsets, one was identified, *ChatMapper*, which satisfies all of these requirements. A screen shot of the ChatMapper development environment is shown in **Error! Reference source not found.**. To support this tool, a translator was developed to convert ChatMapper output to the XML format required by the EA Dialog Engine. In addition, a native Dialog Engine simulator was created. The Dialog Engine was also updated to support the new capabilities. Finally, all of the existing dialog was input into the ChatMapper so that the source code is now present in an easily manipulated form.

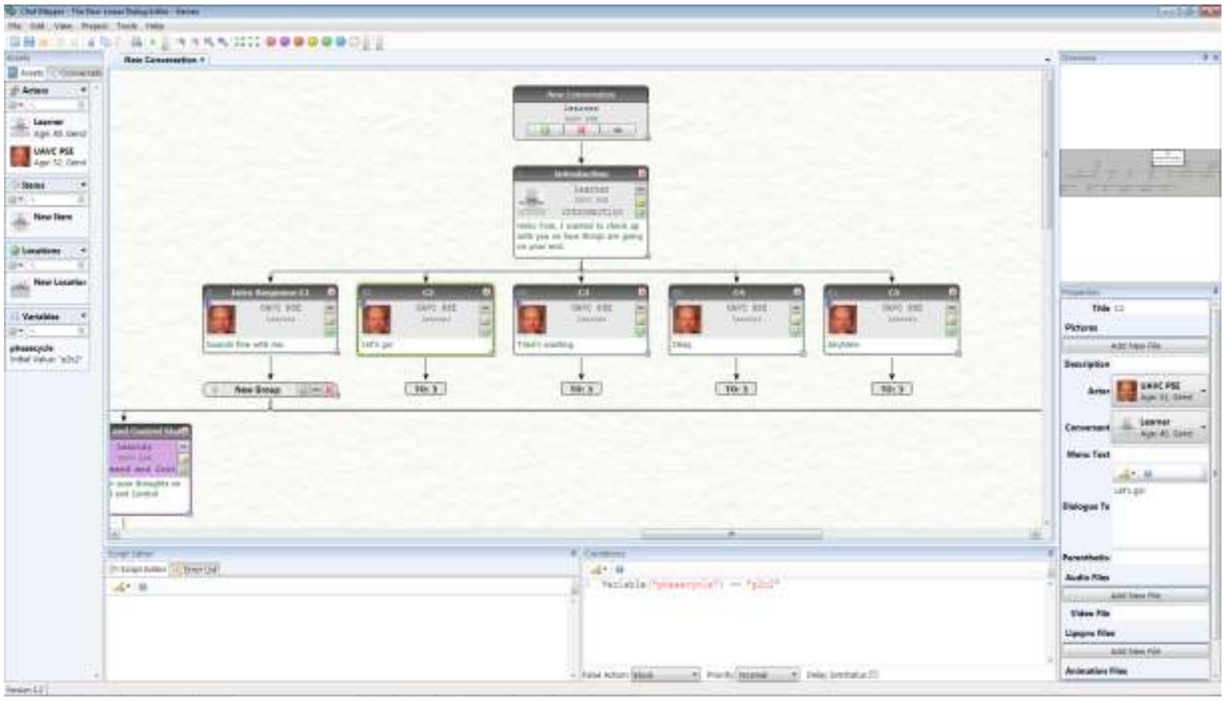


Figure 7: Chat Mapper Dialog Development Environment

3.3.3 TEMPLATES

There are a number of content artifacts that have changed continuously throughout the development of the Experience. These documents include project reviews, learner recommendation forms and simulation status reports. In the base line year, each time these documents changed, the developers were required to hand edit a number of coded documents to provide the information in the correct format to the learner. To avoid this unnecessary work which greatly reduced technical developer productivity and slowed the program, templates were created for the project reviews, learner recommendation forms, as shown in **Error! Reference source not found.**, and a number of other frequently changed artifacts. Even more significantly a means of automatically generating simulation status graphs was developed through the use of an XML syntax to remove developers entirely from this effort. As development continues, additional candidates for templates and auto-generation will be identified and similar development will be done to reduce repetitive work.

	Airframe and Propulsion	Command and Control	Ground Station, Launch/Retrieval	Overall System
Schedule:				
Confidence Level to Achieve Program Schedule Goals	<L, M, H>	<L, M, H>	<L, M, H>	<L, M, H>
Actions to address issues:				
Nothing Required	○	○	○	○
Add/Remove senior/junior staff (%)	Sr<xx>/Jr<xx>	Sr<xx>/Jr<xx>	Sr<xx>/Jr<xx>	Sr<xx>/Jr<xx>
Anticipate schedule extension by xx months	-	-	-	<xx>
Quality:				
Confidence Level to Achieve Program Quality Goals	<L, M, H>	<L, M, H>	<L, M, H>	<L, M, H>
Actions to address issues:				
Nothing Required	○	○	○	○
Add Design & Test Plan Reviews	○	○	○	○
Add/Remove senior/junior test staff (%)	Sr<xx>/Jr<xx>	Sr<xx>/Jr<xx>	Sr<xx>/Jr<xx>	Sr<xx>/Jr<xx>
Add/Remove test systems (n)	○	○	○	○
Review and improve human/system interface	-	-	○	-
Review and revise training process and procedures	-	-	○	-
Capabilities:				
Confidence Level to Achieve Program Capability Goals	<L, M, H>	<L, M, H>	<L, M, H>	<L, M, H>
Actions to address issues:				
Nothing Required	○	○	○	○
Increase/Decrease weight allocation (lbs)	<xx>	<xx>	-	-
Increase/Decrease power allocation (kw)	<xx>	<xx>	-	-
Increase/Decrease volume allocation (ft^3))	<xx>	<xx>	-	-
Increase propulsion efficiency (%)	<xx>	-	-	-
Decrease aerodynamic drag (%)	<xx>	-	-	-
Renegotiate range to xx (nautical miles)	-	-	-	<xx>

Figure 8: Learn Recommendation Template

3.3.4 Future Tools

As noted earlier, there are several additional candidates for tool development that could have a very significant impact on content creation productivity.

3.3.4.1 Phase and Event Specification

Currently the specification of Experience phases and events is one where the content creator specifies this is text, uses a simple format as noted in the Experience Design document. Once this is completed, the technical developers must read this information, interpret it unambiguously and then change the appropriate source code in the Experience Master modules. As one may imagine, this is a tedious process that is error prone and difficult to debug during actual Experience operation. An alternative to this approach has been designed in concept that involves the creation of a graphical interface based tool that allows the content creator to specify the desired Experience phases and events in a natural way similar to the diagram shown in **Error! Reference source not found.**.

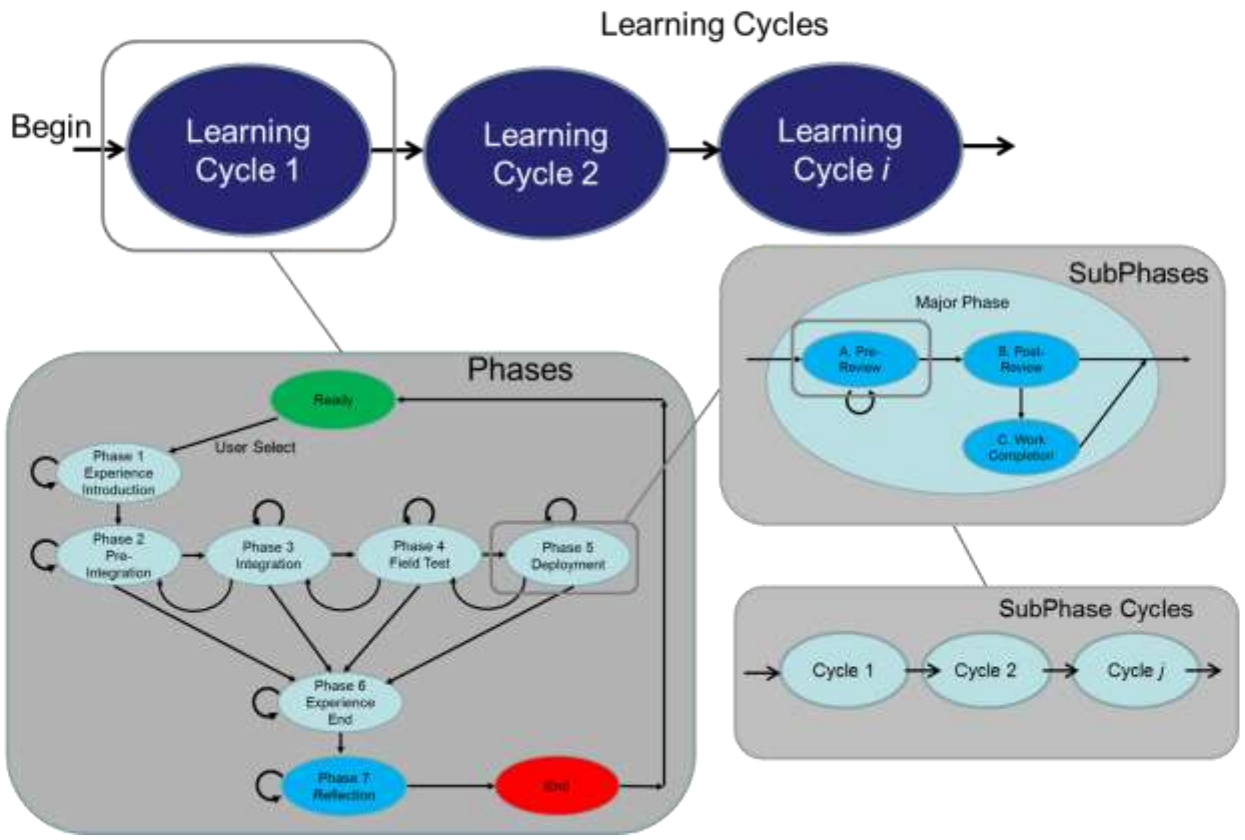


Figure 9: Experience Learning Cycle and Phases

3.3.4.2 Artifact Entry

Another tedious, cumbersome and error-prone process is that of artifact entry. In the current Experience, there are hundreds of artifacts that may be updated within a development cycle. As noted below, currently there are half a dozen steps in the process that involve both the content designer and the technical developer. Requiring the technical developers to get involved in the process greatly slows down the content development work, particularly in an academic or open source environment as there is not full-time staff standing by to assist the content developer. A conceptual design has been done of the development of an application that the content designer could use to eliminate the interaction with the technical developer to reduce the total effort, reduce the time delay in making changes and finally reduce the errors in the process.

- Today:
  - Designer saves file in DropBox
  - Designer tells technical staff to load it into the design
  - Technical staff moves file to the correct location or handcodes changes
  - Technical staff recompiles or links the new code
  - Designer is notified of the change
  - Designer tests changes
- Future:
  - Designer opens artifact entry client
  - Designer saves file into system sandbox
  - Designer tests changes

3.3.4.3 Simulation Model Construction and Parameter Setting

One of the most challenging and technical areas of the Experience content development work is in the construction of the simulation models that are used to provide a simulated experience to the learner. Effort is necessary both to develop the models (in the case of the current Experience these are case systems dynamic models) and to tune the parameters properly to ensure that their output is both plausible and provides the desired experience. Currently, existing models can be used as templates, and a library of such models can be constructed. Longer term, however, it is believed that there is an opportunity to development tools that can assist in the design and tuning process.

The following are three major areas of potential tool development:

- Model Builder: construct models based on templates
- Parameter Tuner: highlight parameters with greatest impact over selected time scales
- Simulation Master: ability to run batches of simulations to accelerate tuning and validation of models.

3.3.4.4 Process and Tools Documentation

Currently, there is no document outside of the flow presented in the Experience Design document on how one would develop a new experience. Documentation will be created in Increment 2 that provides this information in the detail necessary to create a new experience. In addition, external developers will be engaged in Increment 2 to validate these processes and also to provide feedback necessary to improve and update these tools and environments.

3.3.4.5 Experience Learning Evaluation

Development of Learner experience behavior trace capture and analysis tools such that the behaviors of learners can be compared with the behaviors of acknowledged experts in the areas identified to be stress in this experience. This effort can leverage the work that is being done at Stevens in other serious game environments.

4 EVALUATION

4.1 INFORMAL EVALUATION

The following describes the design and results from the information evaluation of the Experience Accelerator.

4.1.1 DESIGN OF INFORMAL EVALUATION

Two forms of informal evaluation were designed in order to provide formative feedback on the EA simulation. The initial formative evaluation of the EA was a survey designed for distribution at the NDIA conference where the EA would be demonstrated. While audience members would not be able to utilize the EA themselves, they would observe as an EA team member walked them through the EA scenario, working through the experience, while explaining the goals and background of the EA as well as future plans. This evaluation focused on three areas: the audience's perception of the usefulness of the EA, their perception of the importance of different components, and their perception of how well the EA supported various features. Finally, the survey sought to recruit those interested in future testing of the EA or contributing to its development. The survey utilized a seven point Likert scale, ranging from strongly agree to strongly disagree, to gather audience perceptions of the EA.

4.1.2 INFORMAL EVALUATION RESULTS

The results of the NDIA conference demonstration provided feedback from eleven audience members. The first section asked if the EA appeared as though it would be useful to their peers, students, or employees, and also asked if they liked the look and feel of the interface. Responses to the Likert questions demonstrated a strong belief that the EA would be useful and that most agreed that the look and feel was appealing. Written comments further supported that the EA was viewed as an important project for the field. Given that the EA was originally envisioned as a three dimensional environment, and was ultimately developed as a two dimensional simulated desktop environment, where the learner views emails, documents, and computer-to-computer online voice conversations, it was encouraging that the surveyed responses did not indicate that the simple visual look was a concern.

The second section of questions asked audience members to indicate the importance of components of the system to the project's success based on the description of the goals of the project. Supported competencies, the EA's architecture, the design of the experience, tools for designing experiences, supported systems dynamics, and assessment and evaluation were listed, with audience members again rating their importance on a seven point Likert scale with seven as very important and one as very trivial to the project's success. All components of the project were rated as largely important, so the audience members indicated the scope of the project was on target.

The third section asked audience members to rate their perception of how well the EA currently supported the targeted features of competencies, architecture, experience design, design tools, systems dynamics, and assessment and evaluation. Ultimately, feedback showed that overall, audience members, without having hands-on experience with the EA, perceived that the targeted features were largely supported. We also were able to identify future participants interested in evaluating or contributing to the project.

The second evaluation placed the prototype into the hands of three experienced systems engineer subject matter experts with decades of experience and asked them to evaluate the EA. While a survey was designed and provided to the subject matter experts, it was decided that the best approach to gathering their insights would be to interview them regarding their experience as they were primarily at evaluating the interface as their difficulties navigating the EA largely prevented them from evaluating other aspects of the EA in significant detail. The key result of this written and oral feedback was expressing frustration with the usability of the interface. The experts struggled to navigate through the project, and this struggle overshadowed their ability to assess the level of accuracy or the effectiveness at promoting learning of the EA.

While the team had scripted guidelines for introducing users to the interface, they had not yet been developed for the interface itself, and therefore, navigating the interface proved difficult and highlighted the importance of first having these guides in place before conducting future evaluations.

4.2 FORMAL EVALUATION DEVELOPMENT

In order to evaluate the efficacy of the EA, a component of the Increment 1 EA activities focused on discovering how to evaluate the impact of the EA on learners' systems thinking competencies. In order to evaluate this impact, literature on assessing systems thinking competencies was reviewed from a variety of fields and example assessments were sought that would be well suited to a pre – post evaluation, measuring learners' systems thinking competency prior to introducing the EA and following their use of the EA. Unfortunately, nothing was found in the literature that was well suited to a quality evaluation of the EA.

4.2.1 Design of Evaluation Experiment

Given the lack of appropriate evaluation tools in the literature, an additional research component of the EA project will be defining how to evaluate the effective learning of systems thinking competencies. A number of potential approaches have been identified for possible implementation in Increment 2.

Given that the EA is seeking to accelerate the experience of novice systems engineers, one approach to evaluating its efficacy could be comparing the choices that novices make in the EA to the choices made by expert system engineers. Furthermore, we could compare the choices the novices make in their first use of the EA to the choices they make in subsequent cycles of use. We can therefore hypothesize that if the EA is effectively accelerating their gaining of experience, or learning of system thinking competencies, then the choices of the novice systems engineers should grow to more closely match those of the expert systems engineers.

Another approach could be developing a separate evaluation tool for systems thinking competencies. This could be a test, developed by examining current assessments in use in DAU and other University's courses that incorporate systems thinking competencies as part of the evaluated learning outcomes. Pre and post test scores could then be compared to see if learners' scores improve after using the EA. If an adequate test is difficult to develop, a text-based case incorporating the usage of similar systems thinking competencies as the EA could be developed as well as a rubric for evaluating it based on expert systems engineer's choices in the case. The choices of a group of subjects who complete the EA could then be compared to the choices of a control group who receive "traditional" systems engineering instruction, such as a course, or the reading of a book, to see how the groups' scores compare. Alternatively, the case and rubric could be used simply with the EA group, utilizing a pre/post format to look for gains in their approach to solving systems problems.

An additional approach could be to do a qualitative analysis of novice systems engineers' experiences using the EA. Video capture could be taken of each learner's interaction with the EA and a think-aloud interview conducted while the learner completes the EA. This could be done for initial and subsequent user interactions with the EA to look for qualitative evidence of learning of systems competencies.

While we will explore a number of options in Increment 2 to determine which evaluation approach holds the most promise, our preliminary thoughts point towards evaluating the choices of novices in the EA compared with those of experts.

4.2.2 Learner Identification

Learners who will be utilized to evaluate the EA will come from several sources. Ideally, DAU students could be utilized as learners in order to best match target user groups for the EA. Additionally, engineering students at the universities involved in the EA project could be recruited for the evaluation. Finally, a number of interested audience members at the conferences where the EA has been presented have indicated their interest in evaluating the EA, either as individuals or as instructors implementing the EA in their own courses.

4.2.3 PLAN

In Increment 2, the evaluation of the Experience Accelerator will initially focus on an additional round of formative research on the EA. The survey developed for the SMEs, detailed above, will be implemented to generate a last round of formative data to help guide any additional improvements to be implemented in the EA prior to its formal evaluation.

Next, the evaluation approach will focus on determining the best approach or approaches for formally evaluating the learning efficacy. Courses currently teaching similar systems thinking competencies will be reviewed for suitable assessments. Likewise, some additional literature review work will be conducted to see how similar competencies, such as problem-solving and project management skills have been evaluated.

The results of this review will guide the targeting of one or more evaluation approaches. Target learners will then be identified and recruited. The targeted evaluation approach will then be implemented and a report on the formal evaluation will be written and presented at a targeted conference.

5 FORWARD PLAN

5.1 LESSONS LEARNED

The following is a summary of the lessons learned from the RT16 team. The lessons are divided into the following four categories:

5. Competencies, Learning and Content
6. Complexity/Effort vs. Authenticity/Learning
7. Technology
8. R&D Processes & Tools

Lessons learned in each of these areas are described below along with an approach to mitigate negative impacts in Increment 2. These lessons learned have impacted the nature of the future work, the processes used, and the identified risk factors moving forward.

5.1.1 Competencies, Learning and Content

LL1.1: ***Systems Thinking Evaluation*** - It is very difficult to evaluate capabilities in systems thinking. After an extensive literature search, very little was found in how to test system thinking and technical problem identification and resolution skills. Additional research will need to be done to develop a means of testing these capabilities. Our approach has been to use a Delphi approach in which subject matter experts review behavior and grade them with respect to competency levels.

LL1.2: ***Learning and Concept Capability Evaluation*** – It is difficult to determine if the learner has actually learned and can apply the identified concepts. Even if the learner goes through the Experience more than once and shows improvement, it is not clear whether they have just learned how to best this particular experience or if they have learned the concept and how to apply it in future situations. It would be desired to develop and exam in a different medium which could be used for pre- and post-testing to assess the results.

5.1.2 Technology

LL2.1: ***Client Graphic Technology Migration*** - While Flash is currently has the most productive environments in which to develop graphical content and is free on the client, the development licensing can be expensive (approximately $200 for individual educational licenses, and $700 for individual commercial licenses) and Flash does not work on iPads which represent a major client technology base. Once the open competing technology, HTML5, has development environments which provide the same level of productivity of Flash, it would be advantageous to migrate to the new technology.

LL2.2: ***Client/Server Interface Reliability*** - We had a number of problems getting the EA client/server interface reliably in the presence of an unreliable network. While the system now works, getting it to this state required a significant amount of debug and redesign work. In Increment 2, we will need to review the design and update it and the implementation to provide a cleaner, more supportable system.

LL2.3: ***UI Look and Feel*** - Creating a professional look and feel for a virtual desktop continues to be a non-trivial undertaking. We received feedback from the subject matter experts on the difficulties that they faced in using the browser and virtual desktop. We have added a text based support online, but we may need to create an overall experience training video as well. We have made and will continue to make changes to address this.

5.1.3 Complexity/Effort vs. Authenticity/Learning

Defense acquisition is a very complex enterprise, with many processes, actors and organizations. Selecting a subset of these to represent in the Experience Accelerator involves numerous design trade-offs to address the interests of (i) the learner community, that wants a realistic but not overwhelming experience, (ii) the education community, that wants realism in support of learning objectives, (iii) the acquisition community, that wants its various aspects represented faithfully, and (iv) the developer community, that wants to provide a useful product while managing complexity, schedule and cost. One of the biggest challenges was to create an authentic, learning experience while managing complexity and the amount of content that needed to be created. During this past year we have continued to learn in these areas.

LL3.1: ***Challenges & Landmines*** - There are an almost infinite number of ways in which a program can fail; combinatorial explosion is a major challenge. This is not so much of a challenge for the simulator, but this continues to be a major issue for the creation of artifacts and dialog which can support these to allow the Learner to make sense of the situation. While we created a catalog of a large number of frequently encountered Challenges and Landmines, for the prototype we implemented just a few of the most likely ones in the areas of aviation hardware and software. During this past year we have added challenges in the area of budget management.

LL3.2: ***KPMs*** - KPMs drive the amount of information that needs to be simulated, the amount of artifacts for background information, dialog and Learner recommendations. During Increment 1 we have added cost to the existing KPMs of schedule, quality and capabilities (range) which has required the development of a plausible cost model and supporting artifacts and dialog. While we have implemented a relatively simple EVM system, it is clear that developing one similar to what would be found in a DoD program office would be very challenging. We will have to see if what we developed has sufficient authenticity to provide the desired learning.

LL3.3: ***Feedback to/from Learner*** – In Increment 2 we increased the amount of feedback to the learner in the form of a confirmation of the actions which were taken as a result of the recommendations that the learner made throughout the experience, and in the feedback that the learner received at the end of the experience. While the recommendation form of feedback was fairly straightforward in its implementation as it represented objective fact, the reflection feedback was more complicated in that it was both subjective in nature and the multiple actions that the learner has taken are inter-related. The process that was used involved noted the decisions that were deemed appropriate, inappropriate and neutral and feedback on each decision was made independently. Understanding the interactions between these is far more complex. One possibility for future work is to record how the learner's behave and determine if there are specific patterns of behavior which can be identified and feedback given to the learner which is most appropriate to the pattern of behavior.

5.1.4 R&D Processes & Tools

At the outset of this project, given the exploratory nature of the research, open communication was established between all of the team members through the use of a Wiki site. However, it was discovered that the overhead in learning how to use and navigate through the site outweighed its advantages so we migrated to the use of a simple DropBox technology for documents and development code, and Webex for group meetings which were held once a week. As the team more than doubled in size and became more specialized, this mode of communication became a bit cumbersome. We eventually migrated to a weekly meeting with the SMEs for content, a weekly team meeting and technology meetings on an as needed basis. While the intention was for the team to develop technology and content using an agile software development process on a single server site at Purdue, it was more difficult than expected to set up an iterative development environment and workflow. During the Increment 1, we moved the development to Stevens and used a set of technologies for bug tracking and version control that are standard in the open source software world. The following are some of the lessons that were learned in Increment 1 after moving to this new environment.

LL4.1: ***Change Management*** - In Increment 1 we started using a formalized ticketing system and process. Due to the instability of the design, we quickly created a huge number of tickets for the developers. Due to this large backlog, we slowly moved away from this system, and started contacting the developers directly on our needs. In retrospect, this was a mistake as it created challenges in tracking the work in queue, in process and completed. Now that the design and implementation has stabilized, we intend to move back to the more formal approach that should prepare for open source distribution and should help us with our challenges in configuration management (see LL4.2).

LL4.2: ***Configuration Management*** - Due to the large number of design files, artifacts, simulation parameters and the like, configuration management has become a great challenge. One of the issues is that the Experience Design document is overly generalized and it is sometimes difficult to know exactly what has been implemented. Another issue is that we do not have a centralized place that is a source for all of the files. The issue goes beyond the software build and reflects all of the artifacts and code. To address this issue we will be creating a single design document that will be used with a work tracking tool to provide configuration management for the program. Reconciliation and updates in these two sources of data will be done on a weekly basis to ensure that control is maintained over the design.

LL4.3: ***Verification and Validation*** - The system that we are developing has become very complex making it quite difficult to test. In addition, resources are quite limited such that the testing that we do is ad hoc and carried out by the person who is implementing the code. One possible solution to this problem is to develop an automated tool for verification. There are simulation packages available that have the ability to automate tests that provide the ability to run a suite of simulations and then check all of the results in parallel. In addition, we should create a pool of people who are outside the implementation team who can test the overall Experience Accelerator system.

LL4.4: ***Configuration Dependency Testing*** - There are a number of implementation issues that are configuration dependent, yet may not be discovered through our normal testing procedures which are limited in their configuration variations. Addressing this will require automated test beds and the prerequisite software and hardware support.

LL4.5: ***Content Creation*** - It is quite difficult to create content, particularly dialog, without actually going through the experience to see what it is like. Unfortunately, this was not possible during much of Increment 1 due to instabilities in the code base and the recent development of the EVM features. In addition, as a content creator you need the ability to see the effects of the experience from the beginning to the end. It would be quite useful to have a developer mode in which one could quickly see the projected results of an entire Experience based on some scripted learner behaviors. These behaviors could be based on recorded behaviors of actual learners. This information could also be useful in the identification of classes of behavior types, how to identify them, and how to improve their performance.

LL4.6: ***Architectural Conformance*** - There was a lack of conformance between the implementation, heavy client, and the architecture, thin client, that was developed in the base line year. As a result, there was a good deal of rework that had to be completed by the Stevens development team. It was clear that we needed someone with a computer science and game design background to manage the development team. This action was taken and the issues have been remedied.

LL4.7: ***Academic Software Development*** - Software is difficult in an academic setting as the workforce is largely composed of students who quickly come and go. In addition, since the development team is so small, consisting of 2-3 member, transitions are particularly difficult as the loss of a single member is a large fraction of the team, there is little overlap with other members, and it is not easy to quickly recruit and bring up to speed new members. It is clear that there needs to be at least one member of the team who understands the entire design and is there long term to provide continuity to the team. This has been accomplished on the technology, simulation and experience design teams through continuity of technically capable faculty researchers. In addition, longer term support is also in place with a graduate student in technology development.

LL4.8: ***Automation of Repetitive Tasks*** - There are a number of repetitive tasks that consume a great deal of our developers' time and effort. One example is the conversion of documents from one format to another which then need to be placed in the appropriate design file. There is a strong need for the development of tools to automate these tedious processes. There is a need for tools to automate tedious processes. A conceptual design has been created for this tool.

LL4.9: ***System Interfaces and Partitioning*** - Another lesson learned in the baseline year and repeated in Increment 1 is the importance of partitioning in the system and creating interfaces such that artifacts and dialog can be created without the involvement of developers. This was increasingly successful in Increment 1 as tools were created to allow the simulation team to create graphs and charts that could be automatically incorporated into the system

5.2 Future Work

The plan is to preserve most of the EA team going forward with the potential inclusion of researchers from Carnegie-Mellon University (CMU) or University of Southern California (USC) in developing multi-learner technology. The organization chart for RT16 is shown in **Error! Reference source not found.**. The team plans to leverage SMEs and other resources to test and validate the system through a pilot program.

Follow-on work has been defined for Increment 2 that is focused on the following:

- Prototype Evaluation (cont.)
- Pilot System Development
- Pilot System Evaluation
- Open Source Preparation and Deployment
- Multi-learner Technology Development
- External Developer Engagement

For more detail on the follow-on work plan, see *Developing Systems Engineering Experience Accelerator (SEEA) Prototype and Roadmap, Increment 2 Technical and Management Work Plan.*

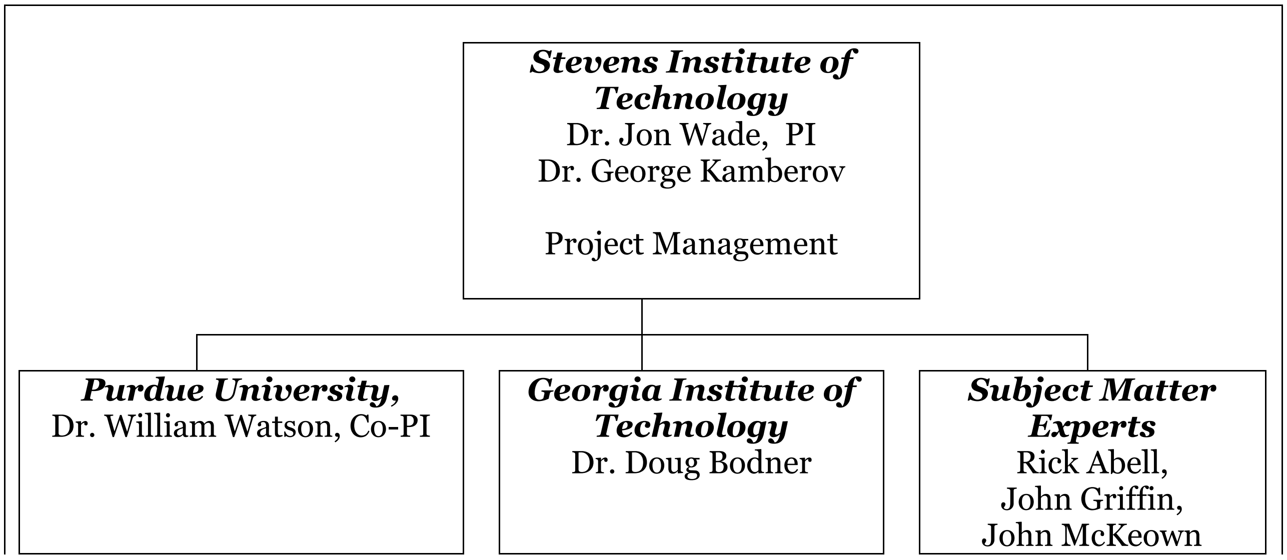


Figure 10: SERC RT16 Organization Chart

In addition to the Increment 2 funding, significant outreach efforts have been made during Increment 1 to find additional sources of funding for the current research team, and opportunities for joint research and external research and development that is necessary to support and sustain an open source community for the Experience Accelerator technology and content. This research falls into two major categories, the development of new capabilities that Increment 2 supports in further refinement of the current experience and multi-learner technology and new experiences. Some of the opportunities that are being explored are noted below:

- Extended Capabilities
  - SERC: Content Creation Tools funding
- New Experiences
  - DAU: Logistics Experience, proposal submitted
  - ONR: Team experience, white paper submitted
  - NSF: Learning in Formal and Informal Settings, 1/14/13 submission
  - NRO: Spacecraft experience, will pilot SEEA
  - MITRE: Team experience, discussions
  - BAE: early exploration
  - Sponsored doctoral research: 2-3 Stevens students

In Increment 2, efforts will be increased to expand the research efforts both internally and externally so that significant development is taking place at multiple research sites to enable our long-term objective of a sustainable open source Experience Accelerator community.

6

6.1 Risk Management

The following is an updated version of the Risk Management plan for Increment 2.

6.1.1 Risk 1: Project Management

Risk: Inability to support known and evolving customer/user feedback with current staff, budget and timeframe.
Mitigation: No significant new EA features are targeted for Increment 2, rather new capabilities will be restricted to those that address the feedback that we receive from learner evaluations of the Experience Accelerator. The work will be targeted at improving the current system to make it ready for Piloting.

6.1.2 RISK 2: CONFIGURATION MANAGEMENT

Risk: Inability to successfully manage the large number of files, configuration variables, present in the Experience Accelerator. (See LL4.2)
Mitigation: A more formalized approach will be used to provide assurance that the implemented configuration is in compliance with the desired specifications. This will be accomplished by creating a single design document that will be used with a work tracking tool to provide configuration management for the program. Reconciliation and updates in these two sources of data will be done on a weekly basis to ensure that control is maintained over the design.

6.1.3 RISK 3: TECHNOLOGY DEVELOPMENT

Risk: Inability to tradeoff long-term architecture and technology objectives (leading to successful open source support) vs. short-term prototype goals. One long term issue is the reliance on Flash which is a proprietary standard not being supported in some Apple client devices (e.g., iPad and iPhone). The likely standard to be supported in the future is HTML5. The time to make the transition will be dependent on when the toolkits are available to make it a productive environment and when it supported on browsers in use (the DoD used some old IE versions which do not support it).
Mitigation: Supporting this migration will require additional funding, with the timing of the migration likely to be post Increment 2. In the meantime, we will keep track of the evolving standards and attempt to reduce the impact of the eventual migration.

6.1.4 RISK 4: CONTENT DEVELOPMENT

Risk: Inability to produce a prototype that provides a compelling experience, supports the desired learning and is seen to be authentic.
Mitigation: Develop and review a design experience document which is used to guide the development process. This experience document will be improved to ensure that it contains the specific information necessary to facilitate configuration management. Unfortunately, due to the instability of the implementation in Increment 1, it was difficult to iteratively develop dialogue and feedback. However, the Experience Accelerator is now sufficiently robust that this iterative approach can be taken. Additional tools will be explored that could improve this situation by providing the ability to quickly see the ramifications of specific learner behaviors.

6.1.5 RISK 5: EVALUATION

Risk: Inconclusive results due to threats to validity of Experimental design (inability to generalize results), limited availability of suitable subjects and insufficient literature to support development of evaluation instruments. The critical challenge is to determine how to measure success in systems thinking and problem identification and resolution.

Mitigation: Additional work will be done to synthesize the published results in the literature. Explore development of new research instrumentation by synthesizing relevant literature should no suitable instrumentation be found in the literature. Create the capability to collect and analyze learner behavior traces, and compare pre- and post-experience traces of learners versus those of acknowledged experts. Possibly utilize Delphi sessions with SMEs will be explored as a means to develop a set of tests that can be used for pre- and post-Experience evaluation in these areas.

References

Bagg, T. C., Granata, R. L., Brumfield, M. D., Casey, C. A., & Jamison, D. E. (2003). "Systems engineering education development (SEED) case study." In *Proceedings of the 13th annual international symposium, INCOSE 2003, Crystal City, Virginia, USA, June 29 - July 3, 2003*.

Bodner, D., Wade, J., Squires, A., Reilly, R., Dominick, P., Kamberov, G., Watson, W. (2012), "Simulation-Based Decision Support for Systems Engineering Experience Acceleration", IEEE Systems Conference, Vancouver, BC, Canada, March 19-23.

Charette, R. (2008). "What's wrong with weapons acquisitions?" *IEEE Spectrum*, November.

Defense Acquisition Program Support (DAPS) Methodology. (2009). *Defense Acquisition Program Support (DAPS) Methodology* (2nd ed. p. 442). Washington, DC: Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering. Retrieved from http://www.acq.osd.mil/se/docs/DAPS_V2.0_Methodology.pdf

Dubey, R. (2006) "Study and Analysis of Best Practices for the Development of Systems Engineers at a Multi-National Organization." *Thesis*, MIT, Boston, MA, USA.

Gavito, V., Verma, D., Dominick, P., Pennotti, M., Giffin, R., Barrese, T., et al. (2010, December 13). "Technical leadership development program: Final technical report SERC-2010-TR-013." Stevens Institute of Technology, Hoboken, NJ.

Kolb, D. (1984). *Experiential Learning: Experience as the Source of Learning and Development*. Prentice-Hall, Englewood Cliffs, NJ.

Madachy, R. (2008). *Software Process Dynamics*. Washington, DC: Wiley-IEEE Press.

NDIA SE Division (2010, July). "Top systems engineering issues in department of defense and defense industry (Final 9a-7/15/10)."

Ryschkewitsch, M., Schaible, D., & Larson, W. (2009, January 20). "The art and science of systems engineering: Long version." *NASA*. Retrieved March 1, 2009, from http://www.nasa.gov/pdf/311198main_Art_and_Sci_of_SE_LONG_1_20_09.pdf

Squires, A. (2007). "Qualifying Exam: Systems Thinking and K12 Education." *Stevens Institute of Technology,* Hoboken, NJ.

Squires, A., Wade, J., Watson, W., Bodner, D., Reilly, R., Dominick, P. (2012), "Year One of the Systems Engineering Experience Accelerator", Proceedings from the Ninth Annual Conference on Systems Engineering Research (CSER), Rolla, Missouri, March 19-22, 2012.

Squires, A., Wade, J., Watson, B., Bodner, D., Okutsu, M., Ingold, D., Reilly, R., Dominick, P., Gelosh, D. (2011), "Investigating an Innovative Approach for Developing Systems Engineering Curriculum: The Systems Engineering Experience Accelerator", Proceedings of the 2011 American Society for Engineering Education (ASEE) Annual Conference and Exposition, Vancouver, BC, Canada, June 26-29, 2011.

Squires, A., Wade, J., Dominick, P., Gelosh, D. (2011) "Building a Competency Taxonomy to Guide Experience Acceleration of Lead Program Systems Engineers", Proceedings from the Ninth Annual Conference on Systems Engineering Research (CSER), Redondo Beach, CA, April 14-16, 2011.

Sterman, J. D. (2000). *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Boston: McGraw-Hill.

Wade, J., Kamberov, G., Bodner, D., Squires, A. (2012) "The Architecture of the Systems Engineering Experience Accelerator", International Council on Systems Engineering (INCOSE) 2012 International Symposium/European Conference on Systems Engineering (EUSEC), Rome, Italy, July 9-12.