Developing Authoring Tools for Skill Models that Enable Adaptive Game-Based Maintenance Training

Sean Guarino
Charles River Analytics Inc.
Cambridge, MA
sguarino@cra.com

Peter Weyhrauch
Charles River Analytics Inc.
Cambridge, MA
pweyhrauch@cra.com

James Niehaus
Charles River Analytics Inc.
Cambridge, MA
jniehaus@cra.com

ABSTRACT

While game-based maintenance training provides a powerful, personalized approach to address individual training needs, it can be costly to update immersive game engines to address new training objectives. Challenges lie not only in the incorporation of new technology that must be trained, but also in the construction of surrounding training materials—curriculums, performance metrics, and optimal training methods—to address procedures for the new technology. In ongoing work with the Air Force Research Laboratories (AFRL), the authors are developing a modeling framework and editing tools for subject matter experts to translate new technology and Technical Orders (TOs) into training objectives, scenarios, and content for existing virtual game-based trainers. The Methodology for Annotated Skill Trees (MAST) provides a formalism that organizes training goals and associated performance metrics, skill decay models, scaffolding models, and effective training methods. This paper discusses the application of this modeling framework to maintenance training for the F-15E aircraft, and the associated development of editing tools to adapt content both in MAST and in the immersive game engine. This paper also describes an approach to improving training by adapting training objectives to support focused repetition of maintenance procedures and review with instructors. Finally, this paper summarizes initial feedback from active duty instructors, and next steps for improving these tools.

ABOUT THE AUTHORS

Mr. Sean Guarino is a Principal Scientist at Charles River Analytics, and the Principal Investigator for this research. Mr. Guarino's research interests include game-based training, including virtual game environments and microgames, computational models of human behavior and skills, and intuitive authoring of computational models and games. Mr. Guarino's has led the development of game-based training for maintenance, information assurance, perception, and adaptive reasoning. Recently, Mr. Guarino has been exploring new applications of game-based technologies to motivating and optimizing crowdsourcing applications. Mr. Guarino earned his B.S. in Computer Science at Cornell University and is currently pursuing his Masters in Psychology at Harvard University.

Dr. Peter Weyhrauch is a Principal Scientist at Charles River Analytics. He leads the design and development of Charles River's methodology for annotating skill trees (MAST), and has been supporting its application in this research. His primary research interests include simulation-based training, artificial intelligence, models of human behavior, and computational narrative technology, a field which he helped to pioneer 20 years ago. Recently, he has worked with ONR to develop a model of surgical skills and scenario-based surgical training and refresher courses. Dr. Weyhrauch has a B.S. in Computer Science from MIT and an M.S. and Ph.D. in Computer Science from Carnegie Mellon U.

Dr. James Niehaus is a Senior Scientist at Charles River Analytics. His areas of expertise include artificial intelligence, training systems, and health technology, and he has been an instrumental contributor to the design and development of MAST, which he used to characterize and model critical skills for training surgical and medical skills. Dr. Niehaus has a B.S. in computer science from the College of Charleston and a Ph.D. in computer science from North Carolina State University.

Authoring Skill Models to Enable Adaptive Game-Based Maintenance Training

Sean Guarino
Charles River Analytics Inc.
Cambridge, MA
sguarino@cra.com

Peter Weyhrauch
Charles River Analytics Inc.
Cambridge, MA
pweyhrauch@cra.com

James Niehaus
Charles River Analytics Inc.
Cambridge, MA
jniehaus@cra.com

INTRODUCTION

Currently, live exercises and classroom training are the central methods for training USAF aircraft maintainers. However, these training methods require costly expert instructors and equipment, and are limited in their ability to demonstrate realistic—and potentially dangerous—maintenance failures. While Partial Task Trainers (PTTs) provide a cost-savings improvement over live training [1], they are brittle systems that also require hands-on instruction, with limited ability to adapt to new training objectives [2; 3]. Adaptive game-based training provides a powerful, personalized approach to address individual training requirements, often designed to react to the trainee and provide additional content based on his success in performing particular actions [4-6]. Effective games motivate participation by being realistic, accessible, and engaging, leading to more independent training that allows instructors to manage more simultaneous students. However, without development and adaptation tools that are accessible to course designers and instructors, updating game engines to address new training requirements remains costly. To manage the development costs of such trainers—and ultimately harness the full power of game-based training for USAF maintenance proficiency—tools are needed to make these games more readily adaptable, both before and during training, by instructors and course designers.

This paper describes ongoing research with the Air Force Research Laboratory (AFRL) to construct tools to support the rapid development and adaptation of game-based training materials. This research focuses on developing and integrating three components to enable game-based maintenance training: (1) a graphical Procedure Authoring Tool that enables course designers to construct training scenarios and performance metrics, leveraging the MAST (Methodology for Annotated Skill Trees) approach to adaptive training; (2) an Instruction Management Tool that enables instructors to observe class-based training exercises in real time, provide assistance as needed within those exercises, and edit scenarios to adapt training to individual needs; and (3) a Game-Based Virtual Trainer that enables teams of trainees to perform maintenance procedures on virtual aircraft. This paper focuses on the first two components, addressing how these tools enable the configuration and management of training exercises. First, the paper describes the MAST training methodology [7-10] and specific requirements for maintenance training. Second, it describes the Procedure Authoring Tool and Instruction Management Tool in more detail. Third, the paper discusses initial evaluation by the USAF's Instructional Technology Unit (ITU) at Sheppard AFB. Finally, the paper presents some initial conclusions on the tools discussed herein, and plans for future research and development.

BACKGROUND

Methodology for Annotated Skill Trees (MAST)

Our approach to developing training scenarios has been focused on adapting the Methodology for Annotated Skill Trees (MAST) to maintenance training needs. Originating from research on developing skill trees that organize medical training objectives [8; 10], MAST, at its base level, models the critical skills that must be trained in a hierarchical skill tree that organizes training objectives for complex skills and procedures. This skill tree can be used to guide a variety of organization paradigms, including scaffolding processes to build from simple base mechanics to complex skills and capabilities, as well as a more ordinal approach to drive the execution of specific multi-step procedures. MAST's novelty and power comes from its flexible computational annotations to these skill trees, which provide modeling components to enable performance assessment, skill level and decay tracking, and instructional management of training content and methodologies. These annotations enable users to link a variety of meta-information to the model, including simple performance attributes (e.g., time constraints for performing a particular step in a procedure), more complex metrics (e.g., computational methods for analyzing immediate performance and skill decay), and models for recommending performance improvements and additional training (e.g., expert performance models that specify expected behavior in executing a given skill once it has been mastered). Using

these annotations, training systems can compute performance on individual skills and aggregated procedures and can assess, identify, and adapt individual learning objectives based on that performance to drive future training content and methodologies.

Figure 1 shows a visual representation of a MAST skill tree. The "skeleton" of the skill tree is the procedure or task model that organizes the procedure into constituent steps, tasks, and subtasks. On top of this skeleton are annotations (shown as colored boxes in the tree and in greater detail on the right) that set MAST apart as a method to represent complex tasks. These annotations are implemented using a hybrid modeling approach that enables the incorporation of a variety of computational elements, ranging from simple computational metrics to complex behavior models constructed using cognitive architectures. Consider an example task of powering on an aircraft. The MAST skill tree would be used to represent this task, breaking it down into checking safety measures, connecting the power device, powering on the aircraft, and testing key capabilities. Each of those subtasks may be further refined to fully define the task. Attached to the subtasks are skills and metrics, which allow performance to be defined and measured. For example, the task of checking the fire extinguisher is a decision-making skill which can be characterized by accuracy (the decision that the fire extinguisher is operational is correct) and speed (how long does it take to make the decision).

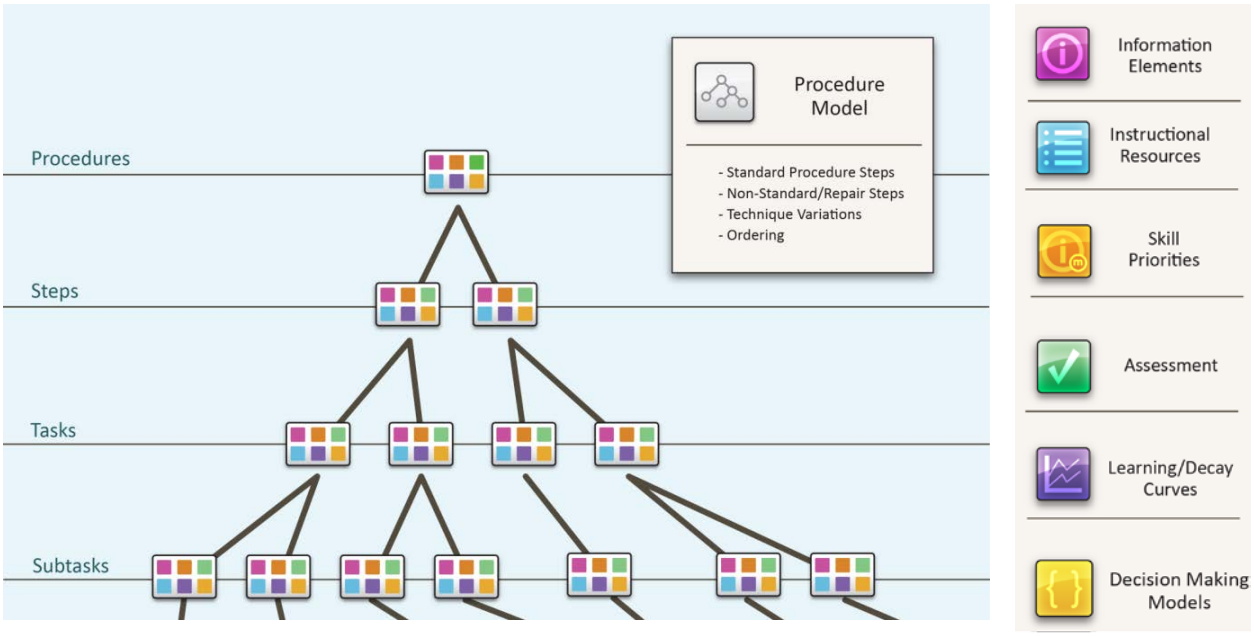

Figure 1: MAST skill tree and annotations

The specifics of the task descriptions are contained in the annotations. These include the following:

- Information Elements: information needed by the task performer, such as the current goals and situation awareness.
- Instructional Resources: these are annotations listing sources such as Field Manuals and data resources to explain or clarify the task or skill.
- Skill Priorities: ratings of difficulty and criticality of the skill so that analysts can prioritize introducing these skills or tasks into a constructive simulation.
- Assessments: methods of measuring and assessing skill performance so that the constructive simulation can determine performance degradation.
- Learning/Decay Curves: functions of how the skills changes over time parameterized by training, use, and other factors so that the system can understand how abilities to perform tasks degrade over time, and how they can be best refreshed.
- Decision-Making Models: computational models to understand and assess the impact of higher order cognitive skills on the procedure.

Challenges in Maintenance Training

An effective game-based maintenance trainer must address several key challenges. At a base level, a primary focus for maintenance training is to ensure that trainees are able to perform specified procedures correctly, understanding and correctly interpreting the Technical Orders (TOs) that are provided to them. Currently, our work is focused on the basic application of static maintenance procedures. These procedures involve executing a series of steps, with few—if any—decision points within them. Trainees are expected to perform the procedure as specified in the TO, and stop that execution if a fault is found. Rather than provide branching points based on faults in the procedure, the fault resolution instead becomes a separate procedure that must be executed. As we move towards fault resolution and troubleshooting maintenance efforts, we expect these models to become more complex, and will likely need to extend our solutions accordingly.

Because of our current focus on the application of static maintenance procedures, there is little nuance available to drive performance assessment. Trainees either pass by performing all steps correctly, and correctly stopping the procedure should they reach a fault; or they fail because they make some error. Trainees may be allowed to make some number of requests for instructor assistance when they are learning a new procedure, but there is ultimately little room for errors that would ultimately be costly in actual procedure execution. Because of this limitation, generating an effective competitive scoring mechanism within a game-based trainer has been challenging. We have largely addressed this issue through the consideration of time constraints on particular steps in the procedure, rewarding trainees for rapid, but accurate, performance. We have also begun to explore the introduction of fault detection exercises that ensure trainees understand when parts of the aircraft have issues. Below, we describe specific authoring and tracking tools that we have built to support adaptive game-based maintenance training.

PROCEDURE AUTHORING TOOL FOR DEVELOPING MAINTENANCE PROCEDURES AND SCENARIOS

A key focus of this work has been to support course designers in offline development of computational training models of maintenance procedures and scenarios to drive training objectives, and to support instructors in customizing these models to address specific training needs, such as, specific active faults to make the scenario more complex for advanced trainees. For maintenance training, these computational training models consist of simple MAST models that are focused on capturing the details and performance metrics for specific procedures. We developed a Procedure Authoring Tool (PAT) that adapts MAST to the maintenance training domain, enabling course designers with limited experience developing these models to rapidly construct them by simply executing the procedure within the virtual trainer. To achieve this, the PAT is built directly on the game environment (DiSTI's GLStudio maintenance trainer). The PAT tracks procedural events within the maintenance trainer, and generates a procedural hierarchy of associated tasks that must be completed (see Figure 2). Course designers can organize these events into groups of steps that represent specific tasks and skills, and can specify key meta-information to drive specific training scenarios and performance analysis functionality. To enable course designers to more readily read these procedures, they can be separated into a multi-screen view, with the procedure model in one window and the aircraft simulation in another (see Figure 3).

The PAT captures the steps taken by the course designer, allowing the course designer to organize groups of steps within a tree. When executing the procedure, students must perform these steps in order, from top to bottom, and then left to right, to successfully complete the procedure. Student actions are matched against actions taken by the course designer during scenario development, and identified as correct when matching, or incorrect otherwise. Many procedural steps are discrete, requiring simply moving a switch or performing an action, and therefore require a direct match. Other steps, such as moving a joystick, are more continuous and more easily incorporate user error in execution. In these cases, we allow the course designer to define error thresholds that allow for some degree of noise in performing the action (e.g., not moving a joystick in a straight line due to limitations in the virtual interface). The PAT captures each of these behaviors automatically, and enables the course designer to then edit limits and settings within the procedural structure to correct and clarify errors made in their own execution. As our trainer is extended to address more complex troubleshooting or fault resolution procedures, the tree will also be extended to support branches in procedural execution based on the aircraft state.

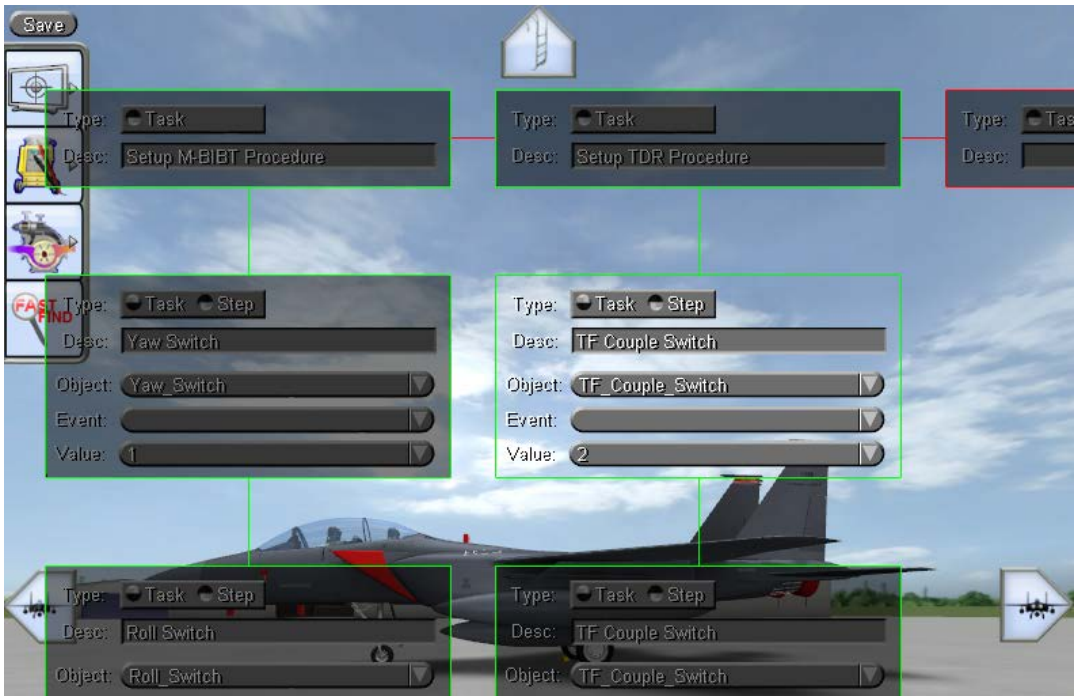

Figure 2: Authoring tool for constructing maintenance training procedures by performing tasks within the game-based simulation

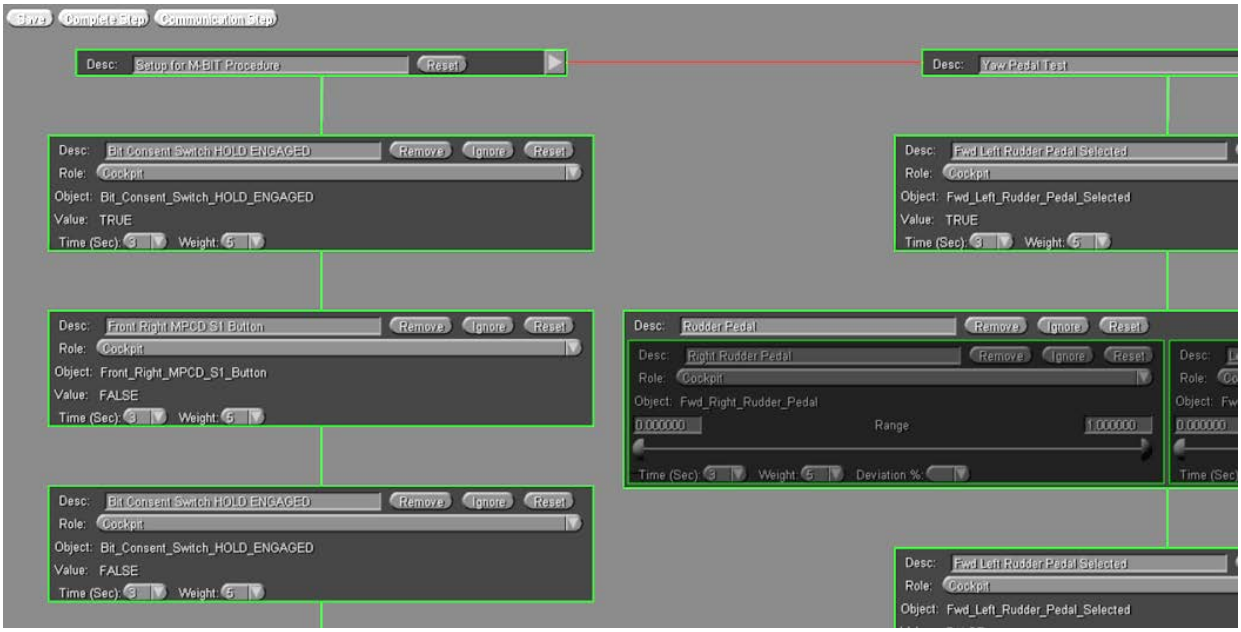

Figure 3: Procedure editor broken into a separate screen for easier editing and aircraft manipulation

The PAT includes an extensible framework for specifying meta-information on procedural tasks. Currently, the PAT collects time constraints, step weights, and team responsibilities. The time constraints and step weights are used to support performance analysis, while the team responsibilities are used to task assignments to teammates in team-based training. The time constraint specifies the expected length of each step; if trainees perform within that length, they receive a score bonus proportional to their temporal performance; if they perform far outside of that boundary, the instructor is informed that they may need assistance. The step weight identifies the relative importance of the task, allowing more performance weight to be placed on task failures that cause expensive or dangerous outcomes

than mistakes that have no lasting impact on the aircraft. Based on feedback from instructors, the system currently does not use catastrophic outcomes for performance assessment, as any mistake is considered to be equally indicative of failure. While the PAT currently collects limited meta-information, the information collected is extensible, allowing additional information to be readily incorporated across procedural steps, both at a global level, providing additional fields that must be specified for all tasks, and a task-specific level, providing specific meta-information that impact only the given task.

Beyond supporting these basic procedural execution use cases, on-going work extends the PAT to specify faults within the aircraft. Specifically, by providing meta-information fields that specify the error state of particular components (e.g., broken flight surfaces or equipment), SMEs can create scenarios that better challenge trainees to detect potential faults. Importantly, error states should not be preset into the procedure, but rather should be manipulated by the instructor when beginning to run a training exercise, based on the expertise level of the trainees. For example, instructors will likely wish to focus on correct execution of the procedure with novice trainees, but may wish to focus on faulty equipment for more advanced trainees. As PAT better tracks fault-tracking meta-information during procedures, PAT will ultimately enable the capability to adapt training to individual students.

INSTRUCTION MANAGEMENT TOOL FOR CONFIGURING AND MANAGING TRAINING EXERCISES

The Instruction Management Tool is designed to enable instructors to observe, lead, and configure training exercises. Instructors need to be able to track overall performance across a number of trainees who are often performing complex procedural tasks as a team. They need to be efficiently informed of those trainees who need immediate assistance, and need a ready way to gather a more detailed assessment of the individual's correct and incorrect actions. Once they understand the trainee's issues, they need to be able to provide instruction to address those issues in an efficient manner. Finally, instructors need to be able to customize the training experience to address trainee needs and gaps, increasing the challenge for those trainees who need a more engaging training experience and introducing equipment challenges to those trainees who fail to address all safety requirements.

Figure 4 shows the first generation Instruction Management Tool, which addresses several of these capabilities. To track trainee performance, this tool provides a direct video feed of the trainees, allowing instructors to directly observe the actions of any trainee participating in the class. The instructor can also toggle an overlay that provides a graph of the procedure that the trainee is performing, with visual cues for the status of that procedure. Specifically, this map shows the successfully completed steps with a green outline, the next targeted step with a yellow outline, and any errors with a red outline. As the tool is refined, this overlay will be placed into a separate screen, as initial feedback from instructors affirmed that the overlay approach is difficult to read. In addition, ongoing work includes implementing a review capability that allows instructors to not only observe real-time performance, but to go back and review previous activities by the student.

When the trainee is having difficulty and requests assistance (or appears to need assistance), the instructor has several methods to provide feedback, beyond the basic automated game-based feedback mechanisms that include performance and error reports. To answer questions, instructors can use a voice interaction channel or text channel. Should more direct assistance be needed, the instructor can take control of the trainee's interface to show them the next step to perform. In this case, the instructor takes on the role of the observed trainee, and the trainee observes the actions of the instructor. Ultimately, this enables the "assists" that instructors are allowed to provide to inexperienced trainees. In future implementations, this tool will support larger class sizes integrating underlying analytical tools that help guide the instructor to students who may need help (even if they have not asked for it yet) and provide automated assistance to students where appropriate (e.g., to provide recommendations in response to simple requests for aid).

Finally, the Instruction Management Tool enables configuration of the training experience in several ways. Currently, the focus of this configuration is on elements of the training interface, including feedback mechanisms (i.e., whether the trainee is provided immediate scoring feedback), time constraints (i.e., whether a timer is presented to the trainee), and error tracking (i.e., whether the system tracks errors and aid requests for the trainee). The instructor can control, through a series of toggles, which information will be shown to the trainee. This enables the instructor to configure a range of modes, from simple practice modes—where trainees are provided information about the next steps they should perform, and are allowed to make numerous errors or requests for assistance, to

testing modes, where trainees are not allowed any errors and are only informed of errors upon failure. Future iterations of these modes include pre-set combinations of features that provide established maintenance training modes, rather than requiring instructors to configure each of these features. Ongoing work includes extending the system to enable instructors to introduce faults into a procedure before or during execution. As described above, each step in the procedure may have meta-information associated with it that specifies the status of the related equipment. This meta-information can be manipulated at the start of the procedure to activate particular faults, or can be manipulated during the procedure to activate a fault that a student may have missed (e.g., making a missed step that would have otherwise had no impact on the success of the procedure impactful to provide the associated training lesson of that potential impact).

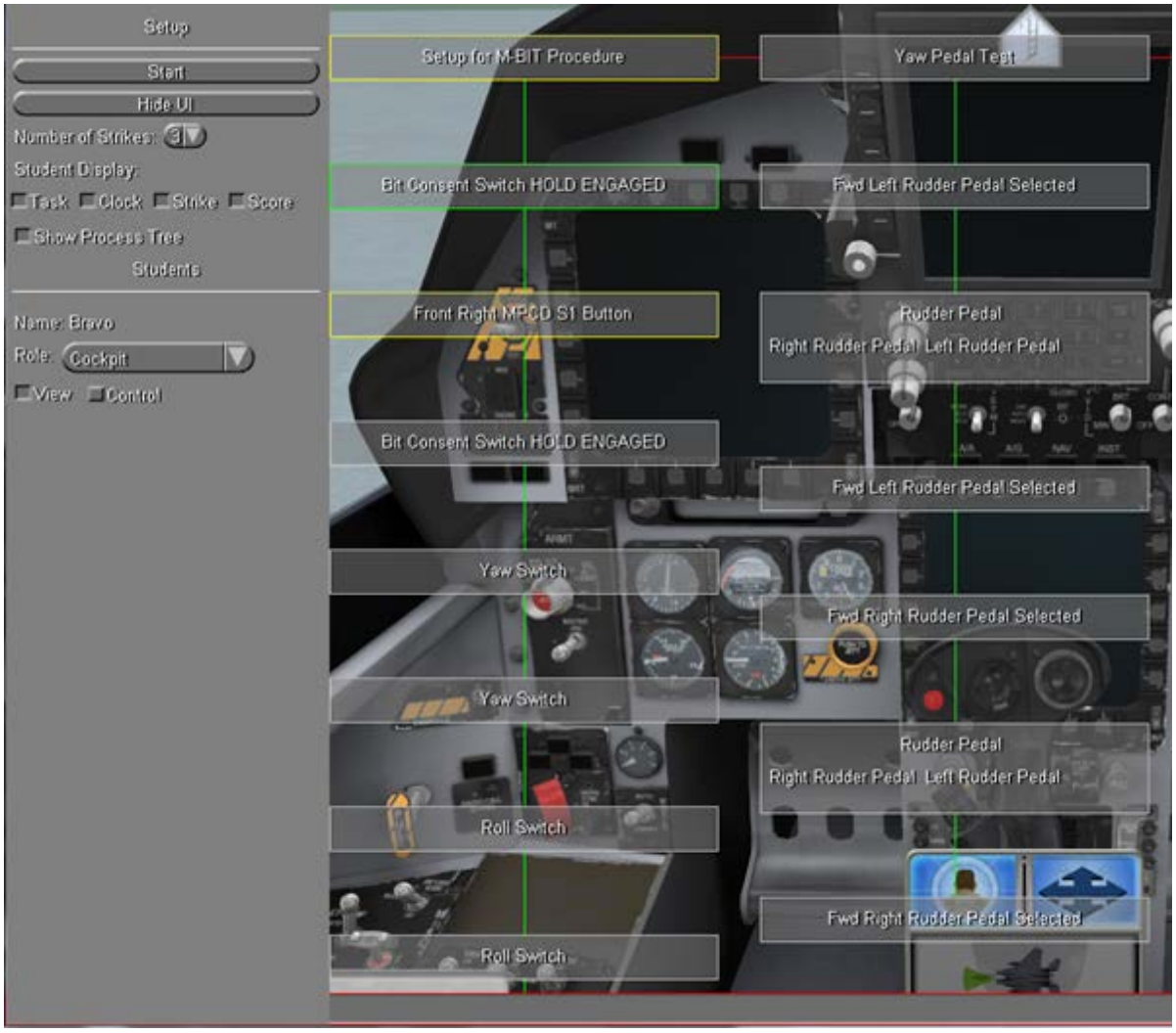

Figure 4: Instruction Management Tool enables instructors to track, assist, and configure team-based training exercises

DEMONSTRATION AND EVALUATION

In November of 2014, the SMEs at the Instructional Technology Unit (ITU) at Sheppard AFB evaluated the usability and usefulness of these tools and provided feedback to drive future development. This was an informal evaluation, in which the ITU members observed the use of the system, and provided verbal feedback and recommendations. Overall, the ITU provided positive feedback, particularly enjoying the capabilities for scenario construction through performance of the scenario, and for tracking trainee performance through the instructor interface. In addition to this positive feedback, the ITU identified a number of key areas for improvement:

- **Larger Class Sizes**: The instructors want to be able to manage class sizes of at least twelve trainees. To support such class sizes, the Instruction Management Tool will need improved interfaces for managing trainees and

teams within the class, as well as analytical tools to direct the instructor's attention and prevent information overload.

- **Improved User Interface**: The instructors identified several key improvements for usability across both tools, particularly focusing on making the procedures more readable and accessible to users. Specifically, the overlay of procedures directly on aircraft views made it difficult to read the procedures; in the future, we will split these views into separate screens, making both the PAT and the Instruction Management Tool multi-window tools.
- **Improved Reporting Capabilities**: The instructors requested improved feedback mechanisms to inform instructors and students of performance details, particularly when managing larger classes.
- **Effective Management of Assists and Errors**: Instructors identified the need to formally track assists and errors that occur across different modes of training.
- **Equipment State Tracking**: Instructors would like to track the state of the equipment, and configure it as faulty. Authoring tools need to be updated to enable that configuration.

Based on this evaluation, the next generation of tools are being refined. A second evaluation is planned in October 2015. This second evaluation will incorporate instructors *and* students in a simulated classroom experience, to gain more complete feedback on the effectiveness of these tools and the training experience. We expect results from this formative evaluation to be available for our presentation at I/ITSEC in December 2015.

CONCLUSIONS AND FUTURE DIRECTIONS

In performing this research, the authors have learned several key lessons that have helped to guide the continued development of tools to support virtual, game-based training for USAF maintenance personnel. The first generation of authoring and instruction management tools were created, and evaluated by USAF SMEs. These instructors have verified the need for strong authoring tools to enable simple construction of training materials and scenarios and instruction management tools to enable instructors to observe, aid, and configure training exercises.

Based on USAF SME feedback, the research team is developing a second generation of improved tools and an improved virtual trainer. A second formative evaluation exercise is scheduled for October 2015. The usability of the Procedure Authoring Tool (PAT) is being enhanced by incorporating multiple windows, with one display tracking the structure and meta-information for the procedure, and a separate screen providing the interface to the aircraft. The authoring tool is being extended to include more annotations and meta-information with additional information necessary to configure scenarios, focusing on enabling course designers to specify equipment states and potential faults that can drive imperfect outcomes in the execution of the maintenance procedures. These faults are currently implemented in our virtual trainer, and simply need to be made available in the authoring tool to activate them as part of the scenario configuration. The usability of the Instruction Management Tool is also being improved by incorporating multiple windows, in this case enabling class and procedure tracking in one screen, and observation and interaction with specific trainees in the second screen. The class management tools are being improved to enable more complete tracking of large classes, with underlying automation to guide instructors to those students who need assistance. The observation tools will enable playback of procedures, and more effective interactions with the trainees. Overall, these changes are being provided based on direct USAF SME guidance. The authors expect these improvements to improve the usability and robustness of the training suite.

Beyond developing a second generation of the tools and simulation, there are several avenues for future research extending these systems. First, adding additional procedures into our system to illustrate the extensibility of this training system to a variety of maintenance objectives. Second, extending the virtual game-based trainer to better support and train physical interactions, developing methodologies for expressing and evaluating physical actions required to understand concepts such as the torque required to manipulate a bolt. Finally, there is parallel work that leverages these training tools and skill models to address more complex maintenance training objectives associated with troubleshooting, fault detection, and fault resolution, addressing not just prescribed procedures, but a more general ability to test for and resolve a variety of potential aircraft issues.

ACKNOWLEDGEMENT

This work is supported by the Air Force Research Laboratory (AFRL) under contract FA8650-14-C-6506. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be

interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ONR or the U.S. Government.

The authors would like to specifically thank Dr. Winston Bennett, Dr. Chantale Wilson, and 1st Lt. Sheri Lamb for their guidance and support throughout this effort. In addition, the authors thank Ms. Tracey Cain and the Instructional Technology Unit (ITU) at Sheppard AFB for their subject matter expertise support, and for providing feedback on early versions of our tools to help guide our development to better address current and future maintenance training objectives. Finally, the authors thank DiSTI Corporation for their participation and support in this effort in building the underlying virtual training software that we are extending with instruction management and authoring capabilities.

REFERENCES

[1] Orlansky, J. (1981). *Cost-Effectiveness of Maintenance Simulators for Military Training*. DTIC Document.
[2] van Leeuwen, M. (2011). *Bell Boeing Receives Order for New and Upgraded CV-22 Training Devices*.
[3] Vogelaar, R. (2010). *Boeing Recieves US Air Force Contract for C-17 Training Devices*.
[4] Annetta, L. A. (2008). *Serious Educational Games: From Theory to Practice*. The Netherlands: Sense Publishers.
[5] Guarino, S., Ho, R., Stair, D., Pfautz, J., and Connell, M. (2012). *Evolvable Microgames for Information Assurance Training*. AFRL.
[6] Niehaus, J. and Riedl, M. O. (2009). Scenario Adaptation: An Approach to Customizing Computer-Based Training Games and Simulations. *AIED 2009 Workshop on Intelligent Educational Games*.
[7] Cao, C. G., MacKenzie, C. L., Ibbotson, J. A., and et, a. l. (1999). Hierarchical Decomposition of Laparoscopic Procedures. *The Convergence of Physical and Informational Technologies: Options for a New Era in Healthcare. MMVR: 7*.
[8] Grosdemouge, C., Weyhrauch, P., Niehaus, J., Schwaitzberg, S., and Cao, C. G. (2012). Design of Training Protocol for Perceptual and Technical Skills in a Minimally Invasive Surgery. *ASME 2012 11th Biennial Conference on Engineering Systems Design and Analysis*, 855-860.
[9] Perez, R. S., Skinner, A., Weyhrauch, P., Niehaus, J., Lathan, C., Schwaitzberg, S. D., and Cao, C. G. (2013). Prevention of Surgical Skill Decay. *Military medicine*, 178, 76-86.
[10] Weyhrauch, P. W., Niehaus, J., Metzger, M., Laufer, S., Kwan, C., and Pugh, C. (2013). Tourniquet Master Training for Junctional and Inguinal Hemorrhage Control (TMT). *Studies in health technology and informatics*, 196, 457-461.
[11] Niehaus, J., Romero, V., Koelle, D., Palmon, N., Bracken, B., Pfautz, J., Reilly, S. N., Weyhrauch, P., Finlayson, M. A., Meister, J. C., and others (2014). A Flexible Framework for the Creation of Narrative-Centered Tools. *5th Workshop on Computational Models of Narrative*, 130.