16th International Conference on Space Operations, Cape Town, South Africa 3 - 5 May 2021.

SpaceOps-2021,2,x1527

Change Management and Verification of Electronic, Automated Procedures

David Kortenkamp[a]*, Khalid Adil[a], Scott Bell[a], Jeffrey Graham[a], Mary Beth Hudson[a], Debra Schreckenghost[a], Neil Woodbury[b]

[a] *TRACLabs, Inc. 16969 N. Texas Ave. Suite #300. Webster, TX USA 77598*
[b] *Retired, NASA Johnson Space Center, Houston TX USA 77058*
* Corresponding Author: korten@traclabs.com

Abstract

Procedures play a large role in successfully operating complex space assets. They provide step-by-step instructions for system check-out and operations and for responding to off-nominal situations. Traditionally, procedures have been developed and distributed as paper documents and PDF files. More recently, organizations have been transitioning electronic procedure systems that represent procedures in formats that are understandable to computers. The International Space Station (ISS), for example, has been using the International Procedure Viewer (IPV) application for many years with procedures represented in an eXtensible Markup Language (XML) format. NASAs Orion vehicle will use electronic procedures that have embedded telemetry and automation capabilities. Commercial space companies such as SpaceX and Sierra Nevada Corporation are also adopting electronic procedure systems with capabilities well beyond what could be done with paper alone, including automatic execution of procedures. However, the procedure verification process for electronic procedures is often the same as was done for paper procedures. In some cases, the electronic procedure is converted to Word in order to go through a paper-based change management and verification process. In this paper, we describe new change management and verification processes and techniques specifically for electronic, automated procedures. These new processes and techniques take advantage of the electronic procedure representation and also acknowledge that electronic procedures, especially those with embedded automation capabilities, become more like software than documents and need to be verified accordingly. To illustrate the points in this paper, we will describe an existing electronic, automated procedure system called PRIDE that is used by both NASA and several commercial space companies and its change management system.

1. Introduction

Electronic operating procedures are becoming the state-of-the-art standard for space operations. From commercial space companies such as SpaceX and Sierra Nevada Corporation [1] to NASA International Space Station (ISS) operations [2], electronic procedures are used for both nominal and off-nominal operations. These electronic procedure platforms replace the older printed checklist procedures used during Space Shuttle operations. They have significant advantages over paper procedures including the ability to bridge the gap between manually executed procedures and automatically executed scripts [3]. As electronic procedure platforms become more prevalent in space operations, the need to safely manage the updating and publication of these procedures is critical. Often the change management systems in place for paper procedures are simply applied to the electronic versions, which fails to take advantage of the synergies and efficiencies that electronic procedure systems provide. In addition, as electronic procedures become a means to automate some operations, they need to be treated more as software and less as documents from a change management perspective. This provides an opportunity to rethink procedure change management. For example, the software industry has specific methodologies and tools for dealing with software change management, verification and validation, and publishing of software releases. In this paper, we describe a change management system that borrows from software engineering approaches and is integrated with an electronic procedure platform. We also describe the advantages of such an approach.

A procedure change management system tracks, reviews, and approves changes to existing procedures before they are published. A typical (simplified) procedure change management process includes the following steps:

- A change is requested for a procedure. This change request (CR) must be documented, tracked, and archived for auditing and quality control purposes.
- An administrator, librarian or other designated individual reviews the CR making sure it has been correctly documented and assigns it to an individual (the author) who will make the change(s) to the procedure.
- Once the procedure changes have been completed, the procedure is submitted for review by relevant experts.
- The reviewers inspect the changes between the previous version of the procedure and the new version with respect to the goals of the CR, overall system safety, and conformance to organizational standards and norms.
- The reviewers provide comments and feedback to the author as necessary and either approve the changes or request that edits be made.
- The new procedure also undergoes verification and validation processes as required by the organization.

- The librarian manages the review process by tracking each reviewers status, identifying issues, and working towards consensus.
- Once all reviewers and the librarian approve, the changed procedure is scheduled for publication.
- The changed procedure is published as a new version and distributed to users.
- All comments, verifications and validations, and approvals are archived for future auditing and process improvement.

Throughout this paper, we will describe the tools and techniques that we have developed to support this typical process for electronic procedures.

2. Background

There are existing approaches to change management for procedures and for software systems. We briefly describe the state-of-the-practice for change management in these two areas in order to provide a point of reference. We also briefly introduce an electronic procedure platform called PRIDE that will be used as an example throughout this paper.

2.1 State-of-the-practice: Space operations procedure change management

Crew Procedures at NASA are very tightly configuration controlled. Every change is documented and approved prior to implementation. For late Shuttle, and International Space Station crew procedures, NASA uses an electronic procedure viewer called IPV[2]. The NASA change management process is run by a specific organization and uses an in-house developed change management system called CR Workflow. This is a software application, where the user inputs the required change information, and the application then routes the change through the defined process. In NASAs case, the crew procedure change is routed between the responsible system representatives, safety, Crew Office, Flight Directors, operations experts, etc. The reviewers can log into CR Workflow and get a tailored list of Change Requests (CRs) awaiting their review. The submitter typically submits the change as a Word document. The attached word document has the track changes feature enabled for the duration of the CR review. Reviewer comments can be entered on the Word document or on the CR. The Procedures Management organization monitors the progress of the CR and ensures a timely response by reviewers and, if required, routes the CR to a formal Procedures Control Board, if the change justifies it. Once the CR is approved, it typically needs to be re-typed into the correct IPV format. This is done in-house by trained editors. The result then runs through a separate, careful, deliberate QA process to ensure that the CR has been incorporated as approved. Basic procedure validation is done at the editor level. The editor auto-checks basic structure and format requirements at authoring time. System experts validate the procedure via review as part of their CR review. Full verification occurs via various methods including everything from desktop reviews, full fidelity simulator use, to use in orbit.

2.2 State-of-the-practice: Software systems change management

Software change management is controlled at most organizations through a combination of version control software [4], tool-based code reviews [5], and issue tracking systems [6]. A typical workflow starts with an issue ticket describing the feature or defect in the software to be addressed. A ticket contains high level discussion of what is to be done, the release the change should be applied to, and what part of the software will be affected. Once consensus has been reached that the ticket is ready for development, a developer signals through the ticket that they have begun work. The developer commits their changes through a version control system, e.g., Git. The version control system keeps a history of every change made to the software, and can easily be reverted or inspected. After the developer has deemed that the software development is finished, the developer marks in the ticket that the code is under review, and a code review is conducted. Tool based code reviews are done by creating a review for the proposed change and adding reviewers. The reviewers can see the proposed change through a comparison view between the old and new lines of code (called a diff view), and discuss the changes through a threaded discussion provided by the tool. Often automatic verification tools scan the proposed change for styling, possible errors, and integration tests, and add their results to the code review. Until consensus is reached, a feedback cycle of changes through code commits and discussion continues. Code reviewers use the code review process for education of norms, protection of architectural decisions, and accident prevention. After consensus is reached, reviewers mark their approval in the code review, and the software changes are approved. The ticket is then marked finished by the developer and the workflow continues.

2.3 The PRIDE Electronic Procedure Platform

The PRIDE electronic procedure platform is a suite of tools for the development, deployment, and execution of standard operating, checklist procedures. An authoring tool called PRIDE Author [7], [8] is used to create or modify procedures in a drag-and-drop fashion. PRIDE Author allows the author to concentrate on the procedure content without needing to worry about formatting or display issues. PRIDE Author produces an eXtensible Markup Language (XML) file that describes the procedure in a well-defined, structured format [9]. PRIDE Author integrates with the PRIDE Version Control system that maintains an audit trail of all procedure changes and versions and differentiates between procedures under development and procedures that have been approved for release.

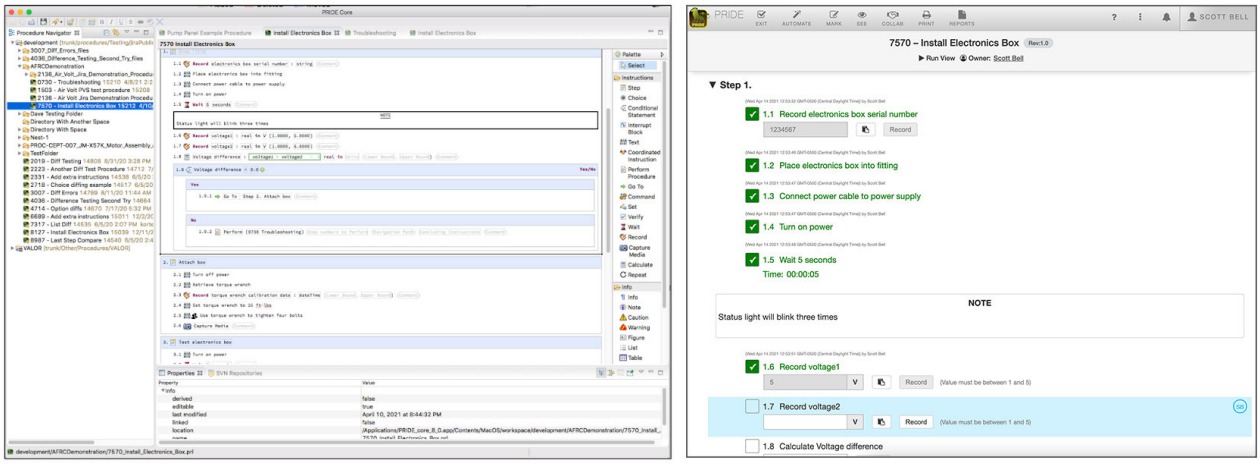

Figure 1: (left) The PRIDE Author tool. (right) The PRIDE View procedure display

The procedure XML file is then translated into an HTML5 document and made available to the operator via the PRIDE View Server. The PRIDE View Server is a web application that browsers connect to for procedure execution. Multiple operators can be working on parts of the same procedure and immediately see the status of all parts of the procedure. All operator interactions with the procedure are stored in the PRIDE View Database, which provides an as-run audit trail of all procedure activity.

PRIDE also has the ability to integrate with a variety of external systems to pull data into the procedure for in-line display to operators. This can include external databases or processes and can also include real-time system telemetry for operations. PRIDE has an optional ability to embed live telemetry and even automate select procedures under careful operator supervision and control [3], [10]. Figure 1 shows (left) the PRIDE Author tool and (right) the PRIDE View procedure display.

3. Procedure Change Management

We have developed and integrated a procedure change management system with the PRIDE electronic procedure platform that includes features of both procedure and software change management systems. This system consists of two components: 1) a change tracking system; and 2) a procedure review system. We describe each of these in turn.

3.1 Change Tracking System

Any change to a procedure needs to be tracked from the first request for a change to its ultimate disposition. That allows for browsing of the change history and for tracking how the procedure evolved over time. Each change request is assigned a unique identifier and all interactions saved to a database. In particular, a change tracking system provides an organization a mechanism for:

1) Initiating procedure change requests
2) Commenting on change requests
3) Assigning those requests to an author
4) Tracking the procedure change through a workflow
5) Resolving the change request (e.g., publish or close)
6) Maintaining a record of all activities with respect to procedure changes

In the next section we describe how a commercial change tracking system was integrated with the PRIDE procedure platform to provide the capabilities outlined above.

3.1.1 Jira

Jira[1] is a commercial issue tracking system that provides a customizable workflow. Issues (or tickets) are entered via a configurable web-based form. All tickets are assigned a number and saved to a database. Tickets can be assigned to individuals and the status of the ticket can be tracked through the workflow. Figure 2 shows an example Jira workflow that would be used for procedure change tracking. The workflow includes transitions between different states. This workflow is editable by the organization to add or change different states and transitions. Jira provides a variety of reporting mechanisms to monitor the status of all tickets. Jira is widely used in commercial software development. It has a robust application programming interface (API) that we used to integrate Jira with PRIDE.

[1] https://www.atlassian.com/software/jira

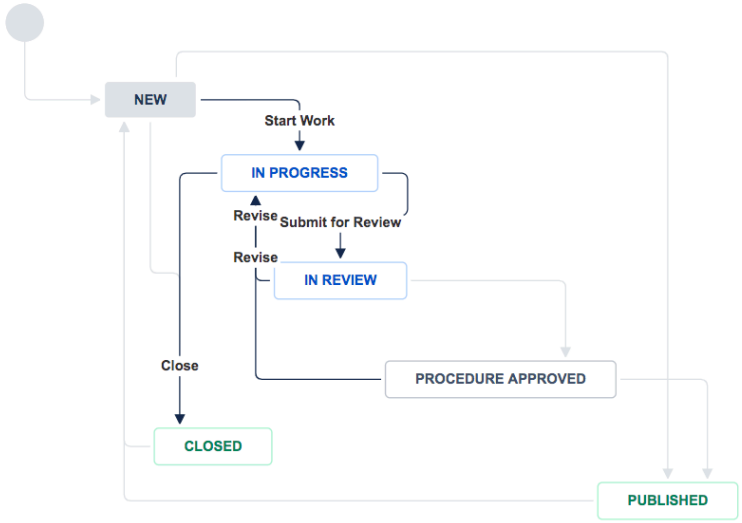

Figure 2: An example change workflow in Jira.

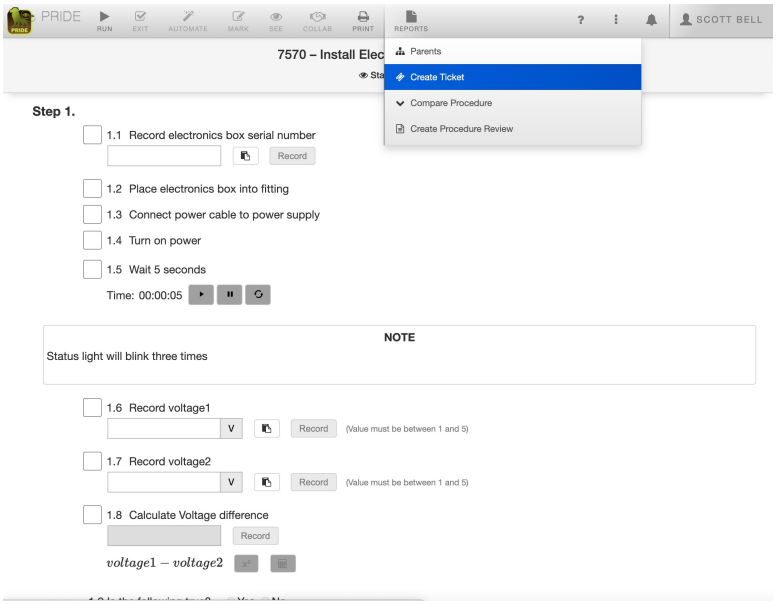

Figure 3: Creating a Jira ticket from within a PRIDE procedure.

3.1.2 Jira Integration

PRIDE has been integrated with Jira using the Jira API and Jira automation capabilities. We will illustrate this integration through a simple example that we will carry through the paper to describe the change management system. In this example, a user requests a change to an existing procedure. They do this by bringing up the procedure in the PrideView application and choosing Create Ticket from a menu at the top of the procedure (Figure 3). This generates a pop-up that asks the user to confirm and then automatically creates a Jira ticket (Figure 4). The user can then click on the link to be taken to the Jira ticket, where they can fill in information about what change needs to be made to the procedure (Figure 5). In this case, the change is that the first two instructions of the procedure need to be swapped. This ticket has a current state of New as shown in the upper right of Figure 5. Once the ticket has been reviewed and assigned to an author its state would change to In Progress. Once the author has completed the requested changes, the state would be changed to In Review and the change management process moves to the Procedure Review System described in the next section.

3.2 Procedure Review System

Any change to a procedure needs to be verified that it is safe and effective. This means a combination of peer review and automated verification and validation. The procedure review system provides an organization a mechanism for ensuring changes to a procedure are safe and meet the requirements in the change request. This includes capabilities for:

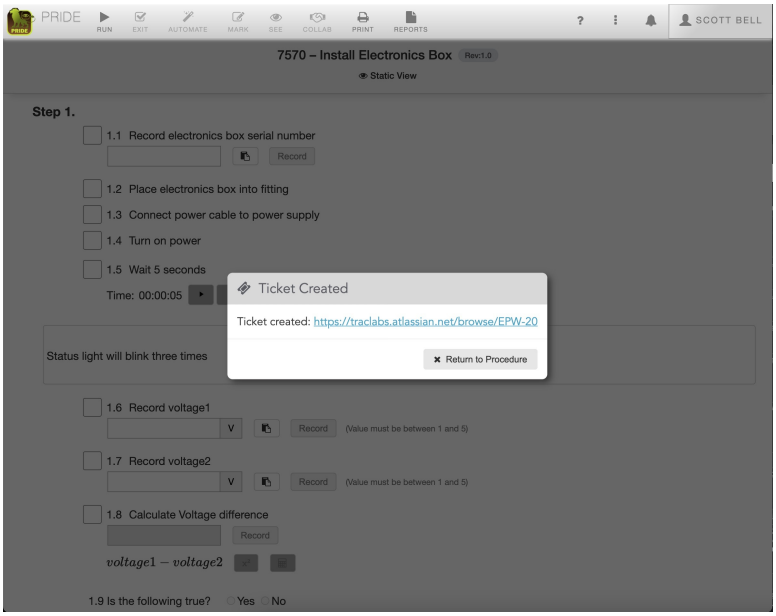

Figure 4: Acknowledging the creation of a Jira ticket.

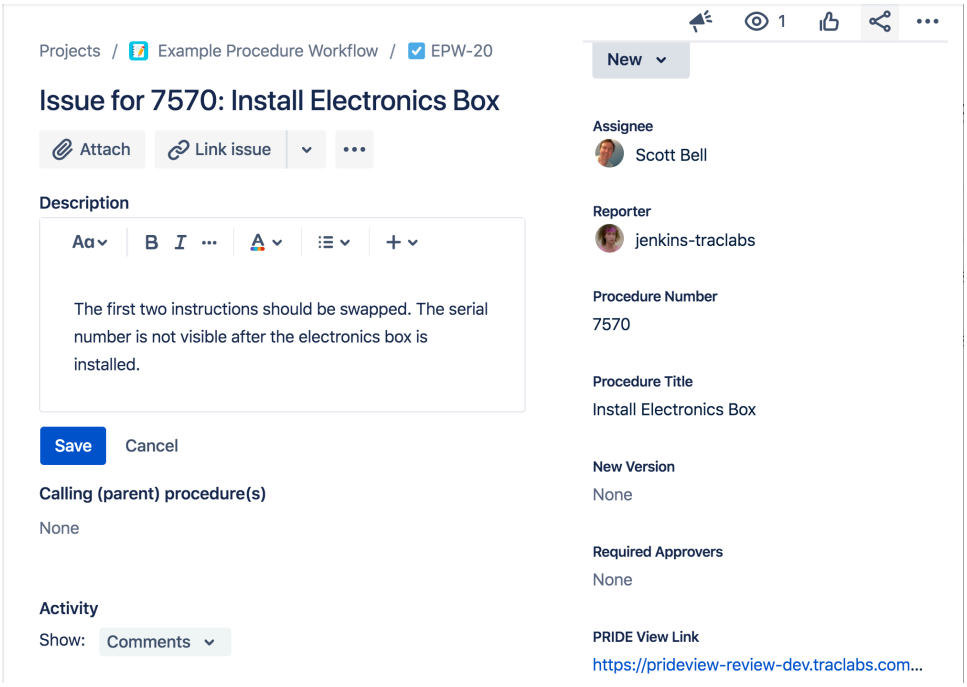

Figure 5: The resulting ticket in the Jira change management system.

1) Assigning individuals to review procedure changes
2) Providing a user-friendly display of the differences between two versions of a procedure
3) Providing for the ability to comment on procedure changes
4) Tracking reviewer approval or non-approval of procedure changes
5) Running automated verification and validation algorithms on a procedure and reporting their results
6) Providing for an administrative ability to approve or close a procedure review
7) Maintaining a record of all activities with respect to procedure reviews

In this section, we describe a procedure review system that is integrated with the PRIDE electronic procedure platform. We continue with the example introduced in the previous section.

3.2.1 Initiating a Procedure Review

When the procedure author has finished the changes they were asked to make to the procedure, they check the changes into the PRIDE Version Control system and then make a Procedure Review Request (PRR). This is done via the PRIDE system as shown in Figure 6. The author types in the change request number (from Jira), the title and

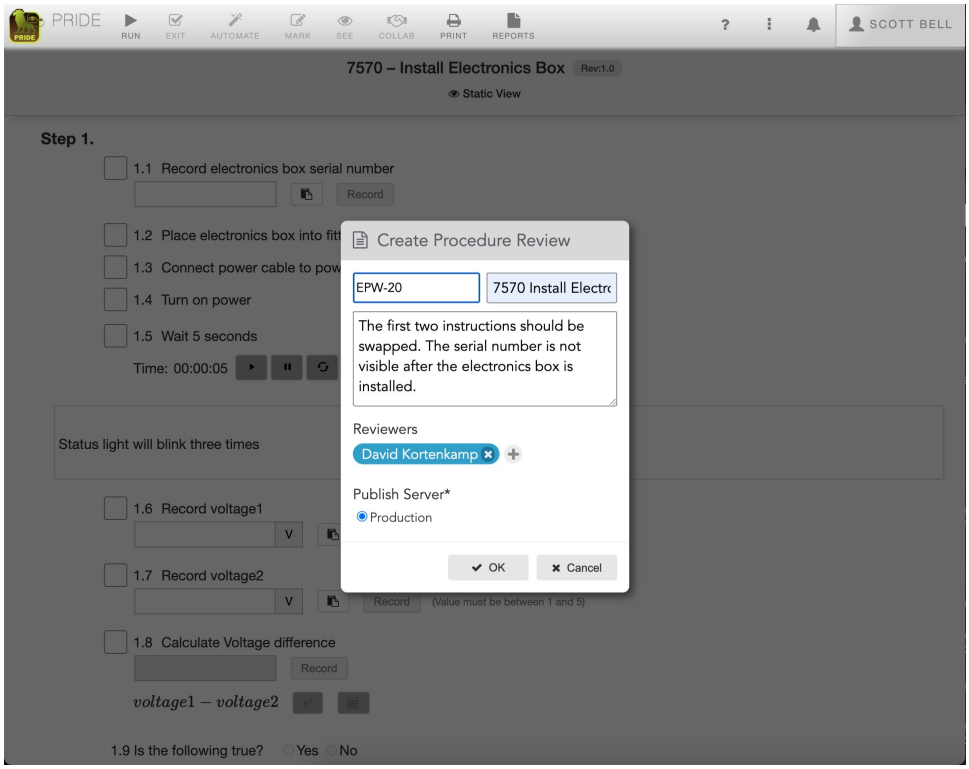

Figure 6: Creating a Procedure Review Request from a PRIDE procedure.

PR Name	PR Number	Procedure Title	Status	Author	Date Started	Activity	Reviewers
Swap First Two Instructions	EPW-20	7570 Install Electronics Box		sbell	04/10/2021		DK

Figure 7: The Procedure Review Request shows up in the Procedure Review Dashboard.

description of the change (also from Jira), and then can assign reviewers from a list of approved PRIDE users. In the future, it would be useful to be able to add groups that must review the procedure or have default reviewers assigned automatically. A unique PRR is stored in the PRIDE View database. A dashboard of all PRRs, their status, and their reviewers status is available (Figure 7). When a PRR is initiated, a set of verification algorithms is automatically run. These are described in the next section.

3.2.2 Verifying Procedures

Paper procedures must rely on manual verification usually against a simulation. Electronic procedures allow for a much wider range of verification options, especially when procedure actions can be automated. Electronic procedure verification can be split into two, complementary approaches: *static* and *dynamic*. Static verification examines the procedure representation looking for errors, inconsistencies, suspicious constructs, or stylistic errors. An example would be flagging a procedure that calls a non-existent child procedure. This type of verification has parallels in software engineering with static program analysis, which can be as simple as automated checking programs (e.g., the "lint" tool) or as complicated as formal methods that mathematically prove program correctness. Dynamic verification automatically executes the procedure against a system simulation of some type to determine whether or not the procedure produces the intended effect.

3.2.2.1 Static verification

We have implemented a number of static verification algorithms for PRIDE. Some of these find errors (i.e., the procedure will not run correctly until these are addressed), some issue warnings (i.e., the procedure potentially has an issue), and some identify stylistic issues (i.e., the procedure violates the organization's stylistic guidelines). Here are some examples of static verification algorithms that have been implemented for PRIDE:

1) All child procedures are present in the procedure library
2) Telemetry and commands referenced in the procedure exist in the system data dictionary
3) Goto location exists in the procedure

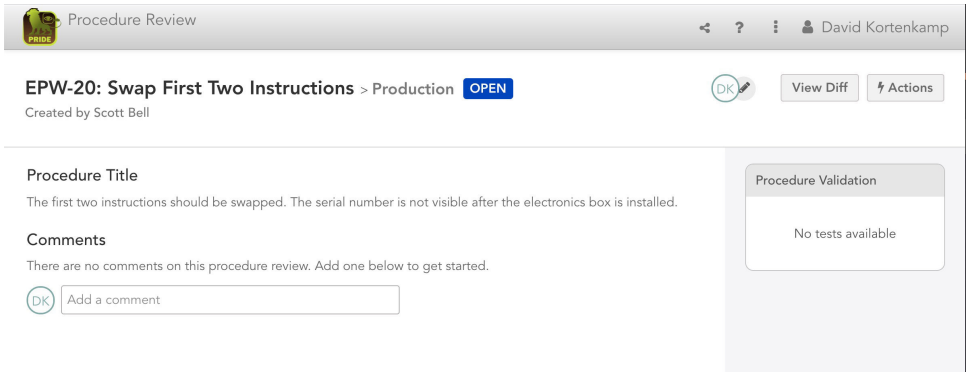

Figure 8: The Procedure Review page.

4) Steps and substeps are not empty and have titles
5) Figure references exist
6) Calculate instructions have valid formulas and references

These static verification algorithms are automatically executed when a PRR is issued and a summary of results is displayed to the reviewers. Authors can also run procedure verification algorithms before creating a PRR to check their procedure. Each issue discovered by the static verification algorithms is also inserted as a comment at the appropriate place in the procedure difference display described in Section 3.2.5.

3.2.2.2 Dynamic verification

The PRIDE Automate component can automatically execute many procedure actions, including sending commands (with arguments) to a system, evaluating telemetry coming from the system, checking calculations using this telemetry, and verifying that telemetered conditions initiate appropriate actions (in If-Then and Choice statements). These automated procedures can be started by external processes using an application programmers interface (API). This means that automation systems (such as Jenkins) can be used to run procedures against system simulations for verification purposes. Execution continues until the procedure is finished or an error is encountered. This is similar to how software projects use continuous testing and integration to ensure changes to software do not introduce errors. This can augment manual testing of procedures and allow for significantly broadened testing and verification.

Procedures can also be manually validated against a system model or simulation. In this case, the as-run validation can be stored in the PRIDE database and a link to that instance be stored in the PRR.

3.2.3 Reviewing Changes

Reviewers are added to the PRR either by the author when creating the PRR or by an administrator or librarian at a later time. Currently, reviewers must be added individually. However, we hope to extend this capability to adding groups of reviewers and automatically adding mandatory reviewers.

Each reviewer needs to log into the PrideView server, navigate to the Procedure Review dashboard, and click on the PRR to which they are assigned as a reviewer. This will bring up a Procedure Review page (Figure 8). This page will show at-a-glance the results of any verifications, any comments on the procedure review itself, and, most importantly, access to a view of the procedure differences from the last version. This last capability is described in the next section.

3.2.4 Displaying Changes

Procedure reviewers need an easy way to see exactly what has changed between the procedure they are reviewing and the current version of that procedure. Because electronic procedures are stored in a computer-friendly representation such as XML or JSON, it is hard to understand the changes that have been made from one version of a procedure to the next. We have developed a user friendly interface for displaying procedure differences that shows the differences between the procedure with the proposed changes and the procedure that is currently published. Figure 9 shows this display for our example change. In this case, the first instruction was moved to become the second instruction. Also, the title for Step 1 was deleted (which was an error on the part of the editor and was not part of the change request). Both of these differences between the new, changed procedure and the existing procedure are easily identified in this procedure differences view. This view essentially serves the same purpose as the Microsoft Word track changes functionality, but for electronic procedures.

3.2.5 Approving Changes

After reviewers examine the changed procedure, they have the option of either approving all of the proposed changes to the procedure or requesting additional changes. When requesting changes, the reviewer can place comments in the procedure differences view or on the Procedure Review page to describe what needs to be changed (see Figure 10). These comments can also have attached images or documents if necessary. As Figure 10 shows, the verification algorithms can also insert their own comments into the procedure differences view. In this case, the verification algorithm noted that the step has no title. Comments can be replied to allowing for a threaded conversation between reviewers and the author. All comments are saved in the database for future browsing.

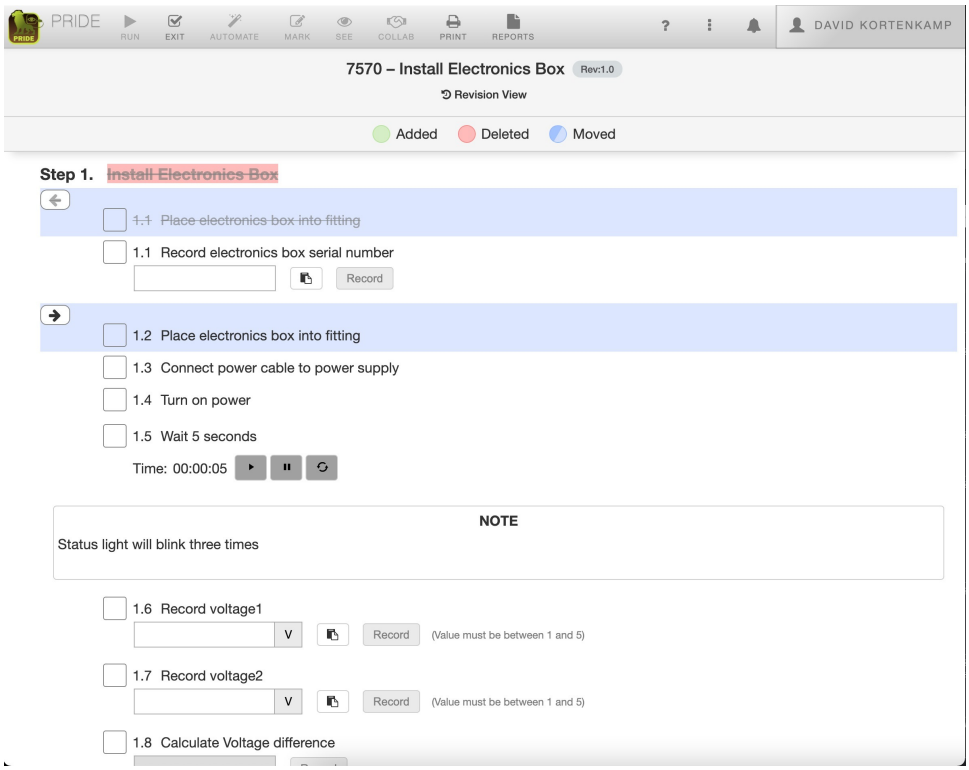

Figure 9: Displaying the differences between two versions of a procedure.

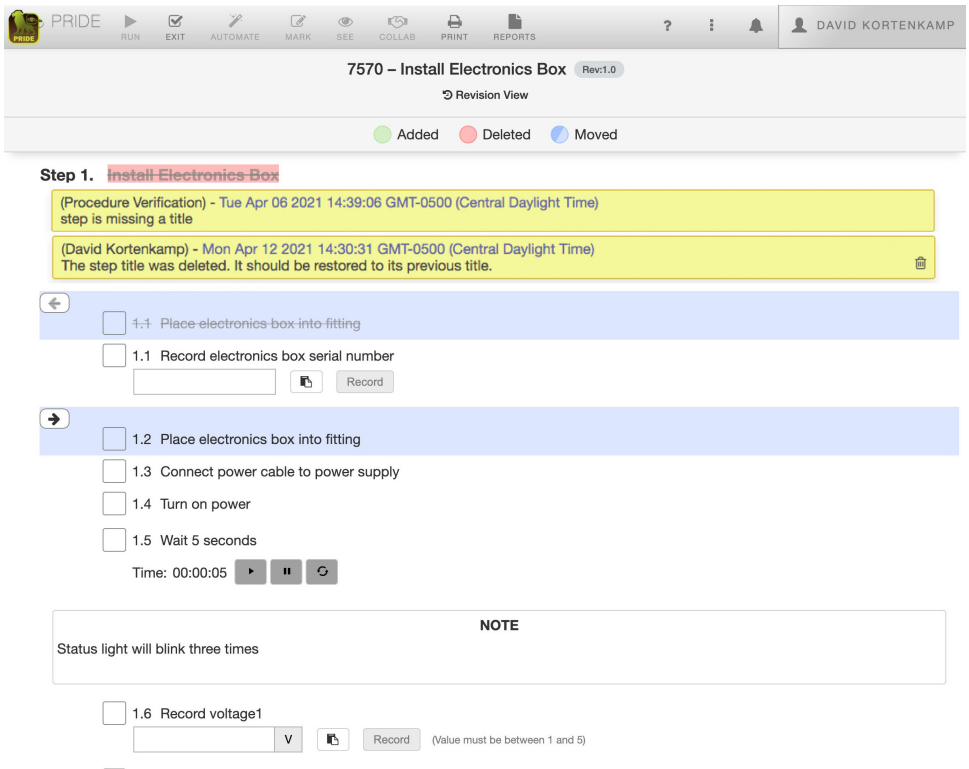

Figure 10: Adding comments to a procedure difference display, including comments generated by procedure verification algorithms.

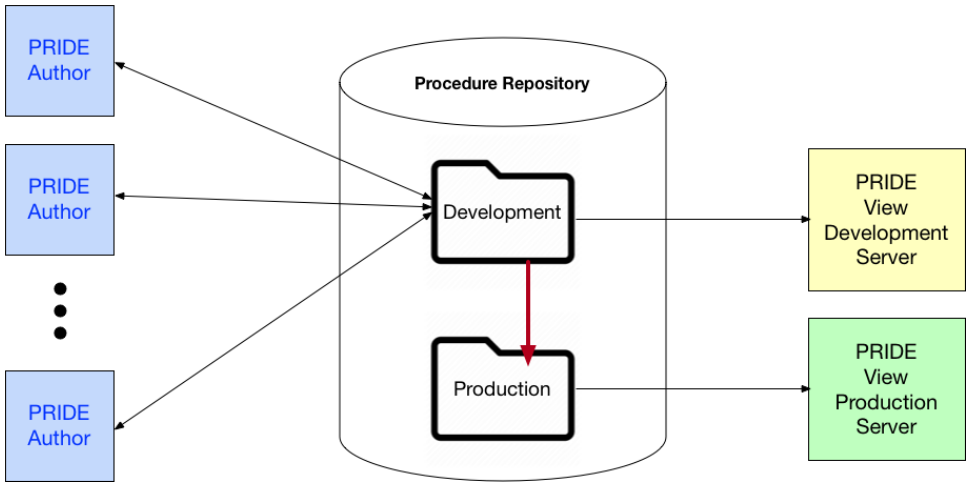

Figure 11: Managing multiple versions of procedures.

4. Publishing Procedures

After the procedure review and editing process, an administrator (or other responsible individual) can change the state of the change request to Procedure Approved. The change request state can also be changed to Closed if the change is no longer needed or if the process needs to be restarted. Changing the status to Procedure Approved does not automatically send the new procedure out to all users. Most organizations that use electronic procedures have at least two separate directories in the procedure repository (Figure 11). A development folder is used for developing new procedures and for editing existing procedures. The development folder is connected to a PRIDE View Server that is used for those purposes. A second folder, often called production, is connected to a separate PRIDE View Server that the users access when performing their work. This folder and server contain the latest, approved versions of all procedures. Moving procedures from development to production is controlled by the change management process described in this paper. Once a procedures state is Procedure Approved, there may still be reasons to delay publication of the procedure. Perhaps the organization only publishes new procedures at certain times (like during the night) or the changes to the procedure are in anticipation of system changes that still need to happen.

We have created a process whereby when the administrator (or other responsible person) changes the state of the change request to Published, the Jira system will automatically move the procedure file from the development folder in the repository to the production folder. Jira will also send a signal to the PRIDE View Server connected to the production folder to update its procedures to the latest versions. At this point, the procedure has made it through the change management process and is published.

5. Conclusions

An efficient and accountable change management system is a requirement for operating procedures in space environments. Many existing change management system were not designed for electronic procedures. We are implementing a change management and procedure verification system as part of the PRIDE electronic procedure platform that leverages the potential of computer-understandable procedure representations and ideas from modern software engineering methodologies. In this paper, we identified the key components of a procedure change management system. An extended example showed how those components work together to track, review, and publish changed procedures. The tools and techniques we described are applicable to a wide range of electronic procedure systems and can play a central role in safe space operations.

Acknowledgements

This work was supported by NASA contract number 80NSSC19C0101 “Electronic procedure verification and workflow management system.” The authors would like to thank Sean Clarke and Matt Redifer of NASA Armstrong Flight Research Center, who provided feedback on the work described in this paper. The authors would also like to thank Jeremy Owen and Jason Gabbert of Sierra Nevada Corporation who are using a beta version of some of these capabilities and have provided feedback on their implementation.

References

[1] J. Gabbert, M. Devereaux, J. Owen, D. Kortenkamp, S. Bell, and G. Kbidy, “Integrated mission operation concepts for the dream chaser cargo system,” in *Proceedings of the 2020 Ground Systems Architecture Workshop (GSAW)*. [Online]. Available: https://gsaw.org/wp-content/uploads/2020/03/2020s09_kbidy.pdf

[2] M. Martignano, M. Wolff, U. Brauer, and P. Kiernan, “Software assisted authoring and viewing of iss crew procedures,” *Acta Astronautica*, vol. 61, no. 11-12, pp. 1053–1060, 2007.

[3] D. Schreckenghost, S. Bell, D. Kortenkamp, and J. Kramer, “Procedure automation: Sharing work with users,” in *AAAI Spring Symposium on Designing the User Experience of Artificial Intelligence*, 2018.

[4] R. Potvin and J. Levenberg, "Why google stores billions of lines of code in a single repository," *Communications of the ACM*, vol. 59, pp. 78–87, 06 2016.
[5] C. Sadowski, E. Söderberg, L. Church, M. Sipko, and A. Bacchelli, "Modern code review: A case study at google," ser. ICSE-SEIP '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3183519.3183525
[6] D. Bertram, A. Voida, S. Greenberg, and R. Walker, "Communication, collaboration, and bugs: The social nature of issue tracking in small, collocated teams," 01 2010, pp. 291–300.
[7] M. Izygon, D. Kortenkamp, and A. Molin, "A procedure integrated development environment for future spacecraft and habitats," in *Proceedings of the Space Technology and Applications International Forum (STAIF 2008), available as American Institute of Physics Conference Proceedings Volume 969*, 2008.
[8] M. B. Hudson, A. Molin, and D. Kortenkamp, "Electronic procedures for medical operations in space," in *Proceedings of the AIAA Space 2011 Conference and Exposition*, 2011.
[9] D. Kortenkamp, R. P. Bonasso, and D. Schreckenghost, "A procedure representation language for human spaceflight operations," in *Proceedings of the 9th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS-08)*, 2008.
[10] S. Bell and D. Kortenkamp, "Embedding procedure assistance into mission control tools," in *Proceedings of the IJCAI Workshop on AI in Space*, 2011.