BROOKHAVEN
NATIONAL LABORATORY

Using sequencing to improve operational efficiency and reliability

T. D'Ottavio, J. Niedziela

Collider-Accelerator Department

Brookhaven National Laboratory
P.O. Box 5000
Upton, NY 11973-5000
www.bnl.gov

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party's use or the results of such use of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

USING SEQUENCING TO IMPROVE OPERATIONAL EFFICIENCY AND RELIABILITY*

Ted D'Ottavio[#], Jennifer Niedziela, BNL, Upton, NY 11973, U.S.A

Abstract

Operation of an accelerator requires the efficient and reproducible execution of many different types of procedures. Some procedures, like beam acceleration, magnet quench recovery, and species switching can be quite complex. To improve accelerator reliability and efficiency, automated execution of procedures is required. Creation of a single robust sequencing application permits the streamlining of this process and offers many benefits in sequence creation, editing, and control. In this paper, we present key features of a sequencer application commissioned at the Collider-Accelerator Department of Brookhaven National Laboratory during the 2007 run. Included is a categorization of the different types of sequences in use, a discussion of the features considered desirable in a good sequencer, and a description of the tools created to aid in sequence construction and diagnosis. Finally, highlights from our operational experience are presented, with emphasis on Operations control of the sequencer, and the alignment of sequence construction with existing operational paradigms.

BACKGROUND

Operation of a control system typically involves the execution of many procedures. These procedures usually begin their life during commissioning periods when experts determine how to best operate their systems. Once satisfied, they attempt to transfer their knowledge into procedures that can be used by those responsible for routine operations.

Sequencing, in the context of this paper, is the process of using computers to automate the execution of procedures. To do this, software tasks are created that can perform specific functions. These tasks are then linked together into a sequence that, when run, executes the tasks sequentially in a manner similar to executing each line of a written procedure. A set of reusable tasks permits the creation of sequences that can automate many different types of procedures.

High-level sequencing has been used for accelerator operations at BNL since the commissioning of the Relativistic Heavy Ion Collider (RHIC) in the year 2000. A description of the sequencing software in place at that time was described in 2001 [1]. That system was heavily influenced by the experience at Fermilab with sequencing and collider operations [2].

A major consolidation and upgrade of the sequencing software at BNL took place in 2006. Some components of the original system were completely eliminated. Others went through substantial upgrades. A description of the current system and its use during the first six months of 2007 is reported here.

SYSTEM DESCRIPTION

The components of the sequencing system, as seen from a users perspective, are shown in Figure 1. The heart of the system is the sequencer, which is referred to as tape (Tool for Automated Procedure Execution). This is where a user selects and runs existing sequences. The sequencer executes its sequences via software that communicates with control system devices, servers, applications and electronic logbooks.
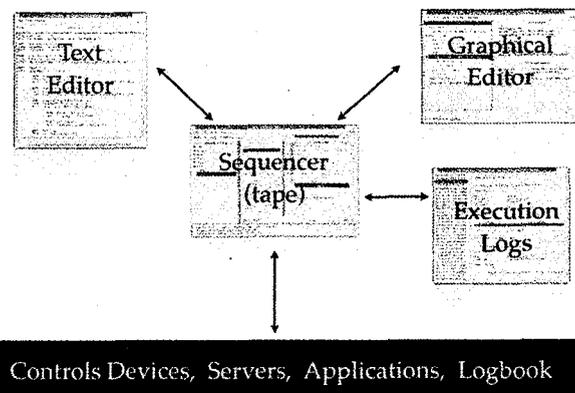

Figure 1: Components of the BNL sequencer system.

A selected sequence can also be edited. Here the user has a choice of either editing with a standard text editor or using a graphical editor that steps the user through the editing process with a custom user interface. Minor edits of a sequence can also be made from within the sequencer itself. If a user is interested in seeing the results from previous runs of a particular sequence, a listing of execution logs is available.

A more detailed view of the sequencer can be seen in Figure 2. The left panel on the display shows the sequence tree, which gives the user a hierarchical view of the available sequence areas. The organization of the sequence tree is under the control of the Operations group. Selecting a node in the sequence tree loads a list of sequences available within the selected area (middle panel). Selecting a sequence then loads the sequence display area (right panel).

As can be seen from Figure 2, a sequence is constructed and displayed as a tree structure. Execution of the sequence occurs from top to bottom. Branches of the tree are expanded and are executed from top to bottom as that point is reached in the sequence. Only highlighted items (white background) are executed. Clicking on a line in the sequence causes its highlighting (and the highlighting of all its children) to change. This makes skipping a section or running part of a sequence very easy to do.
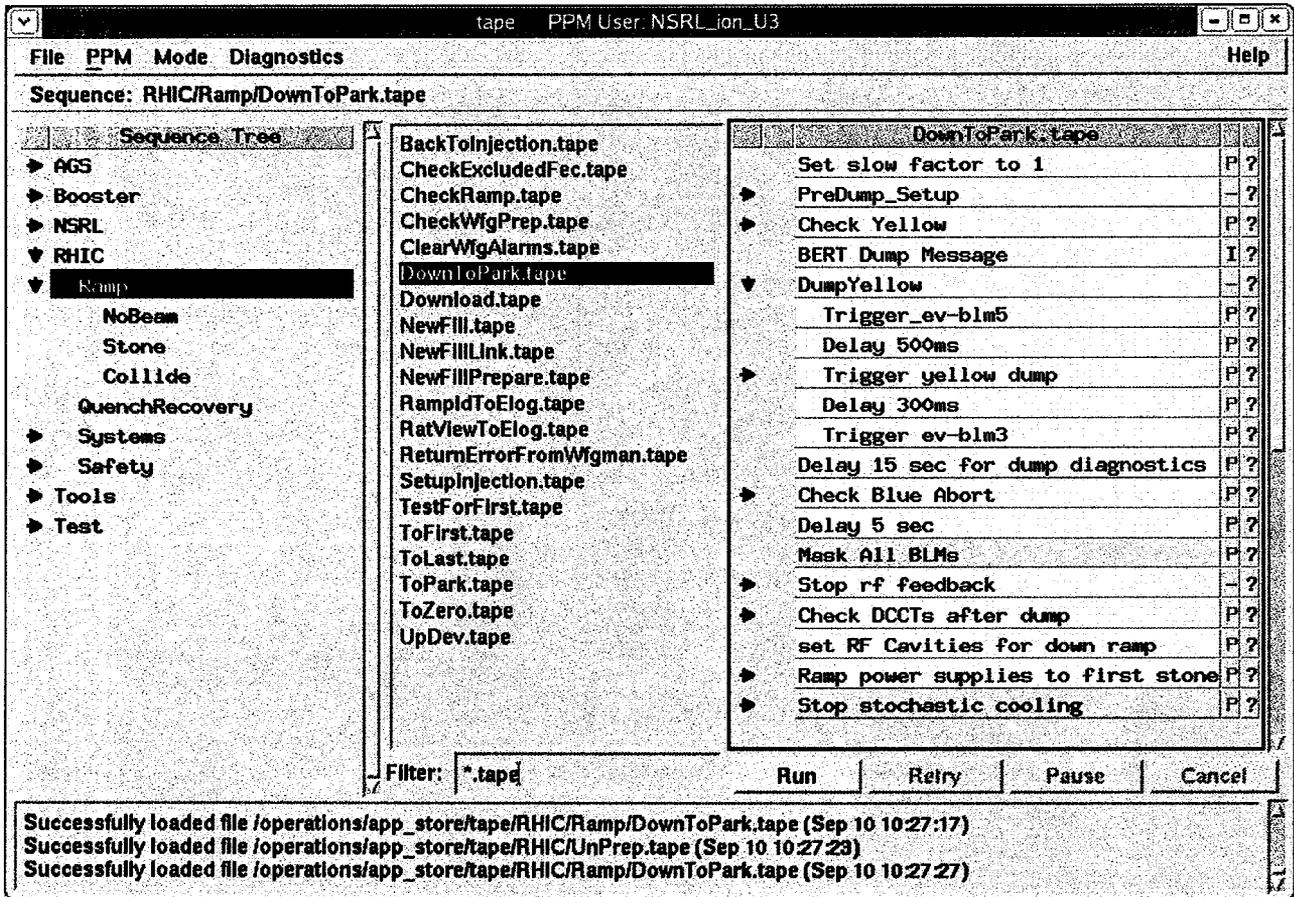
Figure 2: The BNL sequencer (tape) with a sequence loaded and ready to be run.

Clicking the Run button causes the sequence to execute. When running, the line executing is shown in reverse video and the Pause and Cancel buttons become available. When the sequence is paused, either manually or due to an error, the buttons allow the user to Retry, Resume or Cancel. A window listing the details of executing each step in the sequence can also be shown. These execution logs are stored permanently so that they can be viewed at a later date.

RESULTS AND DISCUSSION

Sequencing Use at BNL

The Collider-Accelerator Department at BNL has been providing particle beams for the last several years to two major facilities: 1) experiments run at the Relativistic Heavy Ion Collider (RHIC), and 2) experiments run at the NASA Space Radiation Laboratory (NSRL). The RHIC collider provides particle stores that typically last 4-5 hours, while NSRL takes beam for its cell exposures for periods lasting from a few minutes to several hours.

Sequencing is heavily involved in coordinating this daily activity. Beam for RHIC is injected, accelerated, optimized and dumped for each store. Many smaller procedures are involved for each of these activities. NSRL often requires frequent changes in particle species requiring rapid switching of the injector facilities. Other sequences are available for failure recovery, system-specific setup and diagnostics, safety (moving in or out of controlled access states), and documentation. Table 1 provides a summary of the sequence use at BNL for the six month period starting January 2007.

Table 1: Use of the BNL Sequencer (Jan – Jun 2007)

Number of Sequences Available	492
Sequence Executions	13,992
% of Sequences Run by Operations	85%

The sequence execution rate shown in Table 1 (about 75 sequences run per day) gives an indication of how important sequencing has become to the operation of the particle accelerator complex at BNL. Most of the sequences were created by, and almost all are maintained by, the Operations group at BNL.

Sequence Construction and Editing

A sequence is constructed as a collection of predefined tasks, organized into a tree structure. Sequences are stored as files, with each file showing a listing of the tasks to be run in the order they will be executed. The indenting level of a task indicates its proper place in the sequence tree hierarchy. Sequences can be nested through references to other sequences.

For the purpose of sequence construction, a full task definition for a particular node in the sequence tree consists of three parts:

- The node text displayed to the user (NAME)
- The specific task to be executed (TASK)
- Input data needed by the task (TAGS)

For example, a task in a sequence tree that is setup to change the current going to a particular power supply will look something like:

NAME Move DH2 power supply to injection current
TASK Device::Set
TAGS -Device PS.DH2 -Property currS –Value 7.8

Over 100 different task types (like Device::Set) are available to sequence developers. Some perform very specific functions and are used infrequently, while others are completely generic and get more frequent use. Some of the more heavily used tasks are shown below:

- Device – Set, Get, Wait, Compare, Ramp
- Save/Restore – SaveArchive, RestoreArchive
- GUI – ShowMessage, GetText, GetSelection
- Other – SequenceReference, Timer, RunCommand
- Variable – Set, SetFromFile, Compare, Dump
- Loop – Begin, End, Break, ForEach
- Conditional – If, ElseIf, Else, EndIf

A user constructing or editing a sequence can use either a text editor or a graphical editor. In practice, both methods are used and considered essential. The graphical method is favored when a user wants to be stepped through the editing process with listings of the available TASK and TAG options. The text editor tends to be used when the advantages of a text editor (cut/paste, search/replace) are apparent.

Version control is handled by the RCS revision control system, which we have built into both editing methods. A graphical overlay on top of the RCS command-line tools was constructed at BNL so that users can readily view the revision dates, editors, and comments held by the RCS system. A graphical display of differences between revisions is also available and reverting to earlier revisions is supported within the sequencer.

Sequencing Pros

Operation of a large machine often involves many people using a set of operations procedures. Well-written procedures have a number of benefits. They improve efficiency, allow non-experts to operate complex equipment, and significantly reduce error rates.

Written procedures, however, also have some well-known problems. A good procedure is hard to write and sometimes doesn't have the required level of detail. Often written by experts, a procedure makes assumptions about user knowledge that can cause different users to run a procedure in different ways. In addition, procedures are often not kept up to date leading to errors or lost time due to attempted workarounds.

Sequencing automates procedure execution. The most obvious benefit of automation is that procedures are executed more efficiently and time is saved. In a collider,

reducing the time it takes to inject, accelerate and optimize beam leads to more time available for physics.

Another significant benefit of sequencing is that procedures are executed in a consistent way leading to consistent results. Sequence problems are reproducible and, therefore, are found and fixed quickly leading to improvements in reliability. And since it is easy to log the results from a sequence execution, questions about any particular sequence run can be quickly answered.

At BNL, the Operations group has embraced the use of sequencing for many reasons. Sequences can be built and owned by Operations personnel leading to pride and personal responsibility for the developed sequences. Operator-driven sequence development can lead to an enhanced knowledge of the control system. Operations personnel are more productive since they can often do other things while a sequence is running. And since the sequencer is now handling a lot of monotonous tasks, the quality of life for Operations staff is improved.

Sequencing Cons

A few downsides to using sequencing vs. written procedures are notable. The most significant is that a failure of the sequencer itself (a lock-up or crash) can cause confusion about what was executed and how to recover. We have tried to mitigate this as much as possible by building the writing of the execution logs into the execution of the sequence tasks so that virtually all information about what was executed exists in the logs.

Perhaps more so than with written procedures, the knowledge that went into building a sequence that has been run unaltered over a long period of time can quickly deteriorate, though the sequence details provide some built-in documentation. Finally, the ease with which a sequence can be run (a button click) can also lead to potentially harmful sequences being run inadvertently.

CONCLUSIONS

A major upgrade to the sequencing system at BNL was successfully completed during 2006 and commissioned during 2007. The upgrade included a more powerful and flexible sequencing language, improvements to sequence editing capabilities, and an improved sequencer.

The sequencing system is now heavily used at BNL. It has been essential for the smooth running of the RHIC and NSRL experimental facilities. While the Controls group is responsible for the sequencing software, the Operations group maintains responsibility for creating, maintaining and running almost all of the sequences.

REFERENCES

[1] T. D'Ottavio, B. Frak, J. Morris, T. Satogata, and J. Van Zeijts, "Description of the RHIC Sequencer System," ICALEPCS 2001.

[2] J. Annala, "The Fermilab Sequencer – Use in Collider Operations," ICALEPCS 1995.