Tcl at the NSCL: a 30 (15?) year retrospective

Ron Fox and the Staff and Students of the National Superconducting Cyclotron Lab
Michigan State University

***Abstract*—**The National Superconducting Cyclotron Laboratory (NSCL) is an NSF funded laboratory that performs basic nuclear physics research on nucleus-nucleus collisions innvolving systems that are far from stability. The operation of the NSCL has been funded by the National Science Foundation since 1980.

The NSCL has developed and used several Tcl based applications and tool. These tools are used by a broad community of researchers and accelerator technologiest This retrospective will examine the impact of presenting the NSCL staff with Tcl based tools and toolkits. A speculative look forward at the role of Tcl within the NSCL as it constructs the DOE funded Facility for Rare Isotope Research (FRIB)

I. Introduction

The National Superconducting Cyclotron Laboratory (NSCL) is an National Science Foundation (NSF) funded laboratory that conducts basic research in Nuclear Physics. Software based on and using Tcl have been used at the NSCL for a number of years. The purpose of this paper is to describe the ways in which Tcl has been and is now used at the NSCL. Tcl application case studies will also be provided where appropriate.

In December 2008, the Department of Energy (DOE) selected Michigan State University and the NSCL as the location of a new laboratory; the Facility for Rare Isotope Research (FRIB). FRIB is scheduled to begin operation around 2018. The potential application areas and barriers to the use of Tcl will be discussed as well.

The remainder of the paper will be organized as follows:

- The NSCL will be described with a layman's introduction to the motivation behind the research this done here.
- A brief overview of the FRIB project, its purpose, schedule and remaining administrative hurdles will be given.
- A historical perspective of the introduction of Tcl to the NSCL will then be described. Some speculative work in progress will be described.
- Taxonomy of the use of Tcl at the NSCL will be presented along with case studies illustrating each of the elements in this taxonomy.
- Conclusions about the use of Tcl in the past will be presented along with a bit of crystal ball gazing regarding the role of Tcl in the future of the NSCL/FRIB.

II. The NSCL and our research

What is now the NSCL first started producing accelerated nuclei 1961 when it commissioned the K-50 cyclotron. In 1982 the NSF funded the construction of a K500 (500MeV/A) cyclotron, and later (1989) a K800 cyclotron which outperformed its design specifications and was therefore renamed the K1200. An n NSF grant in 2000 supported running a coupling line between the K500 and K1200 to improve primary beam intensity and to build a fragment separator which started the NSCL on its career as a radioactive beam facility.

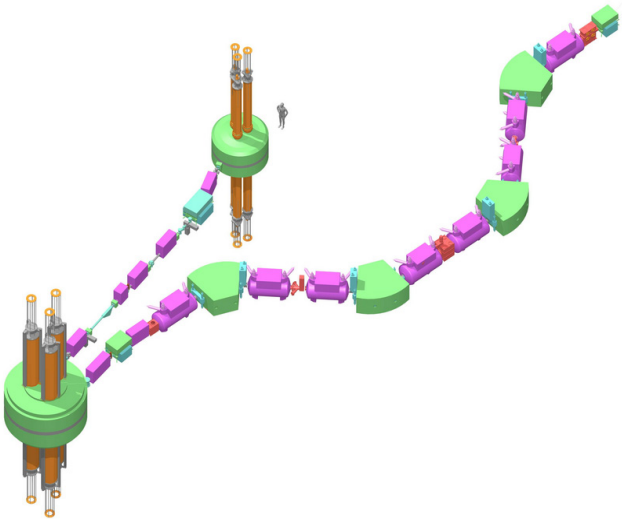

Figure 1 Schematic of the accelerator and separator

Figure 1 above shows a schematic of the beam production facility. An ECR ion source (not shown in the schematic) injects partially stripped ions into the K500 at the top center of the picture (a small grey human figure is provided for scale). Beam extracted from the K500 is transported along a coupling line to the K1200 where it is run through a foil that increases the ionic charge. The more fully stripped ions are injected into the K1200 (lower left). The K1200 beam is then extracted and is transported to a target at the entry of the A1900 fragment separator (running lower left to upper right). The fragment separator selects the desired secondary beam which is then transported to the experimental target.

Figure 2 shows a floor plan of the experimental part of the facility. Each experimental area (to the right of the A1900

fragment separator in the floor plan) has an experimental target as well as detector and electronics packages that are specialized for specific types of experiments and the apparatus in that area.

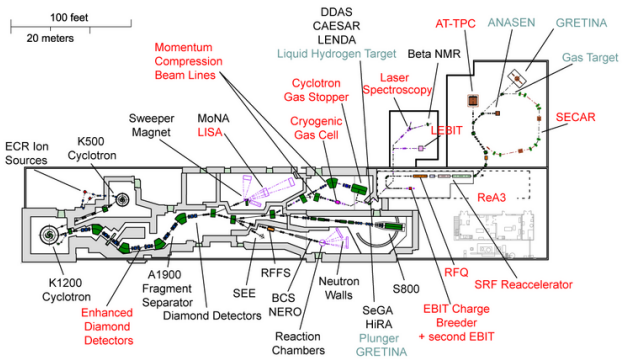

Figure 2 NSCL experimental area floor plan

A. *Why do radioactive beam experiments.*

In this section we present a brief motivation for the research done at the NSCL.

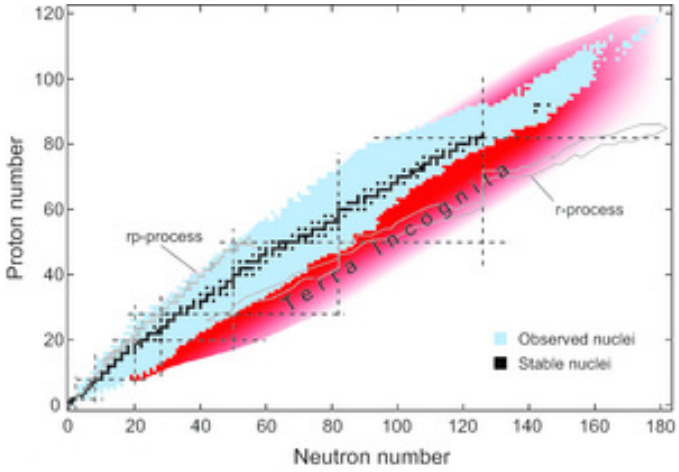

Figure 3 chart of the nuclei

Figure 3 above shows a chart of the nuclei. Each isotope consists of a fixed number of protons (**Z**) (which identify the element) and neutrons. The sum of the neutron and proton count is referred to as **A** which is roughly the nuclear mass. In figure 3 above, stable nuclei are in black. Those which are lighter or darker shades of grey are unstable.

There is a strong belief amongst astrophysicists that most of the heavy elements in the universe have been, and still are being created in nuclear reactions in stars, and that those processes involve decay chains with nuclei far from stability. The nuclei involved in the production of stable heavy elements are shown in Figure3 in the bands labeled *rp-process* and *r-process* as well as a band, not labeled that participate in the *p-process*. An understanding of the rates of these decays and, where several decays are possible, the *branching-ratios* between these decays is critical to an understanding of how the elements we now see were created and what their actual abundances are.

Collisions of heavy ions and unstable neutron rich nuclei create momentary nucleon densities that approach the densities and compositions of supernovae and even neutron stars. The number of nucleons present is already sufficient to help reach an understanding of the liquid-gas phase transition in nuclear matter as it occurs under these stressed conditions.

In short we can imagine the work done at the NSCL as bringing the heavens to earth, allowing us to study what happens in the interiors of stars that are, for now, only observable at a distance.

B. *Stopped and Reaccelerated Beams*

The technique used at the NSCL to create radioactive isotope beams is called *projectile fragmentation*. This is because we select from the remains of the projectile after it has interacted with the A1900 production target. This has the advantage that the secondary beam will have energies that are essentially those of the primary beam. The secondary beam can therefore be easily transported from the separator exit to the experimental target.

Projectile fragmentation requires beams of sufficient minimum energy. This minimum required energy arises, among other things, from the fact that in order to get two positively charged nuclei to interact, we must jam them close enough together that they overcome the electric repulsive force between them and come within the much shorter range of the nuclear strong force. For example with a primary beam of ^{16}O on a production target of ^{5}Be, a very light projectile on a typical production target, this coulomb barrier is already 20MeV. In practice we use much heavier projectiles and consequently we need higher energies to provide sufficient incident energy to create the desired isotopes. This is because the coulomb barrier goes up like the product of the number of protons in the two nuclei.

While the resulting energetic secondary beams are useful for a broad variety of experiments, there are still a large set of interesting experiments for which we would like to have lower secondary beam energies. The NSCL has developed several methods to stop these high energy beams (the most energetic are moving at about ½ the speed of light)! We have just finished commissioning a reaccelerating LINAC which will allow us to study radioactive isotopes at energies from a few hundreds of KeV to 5MeV.

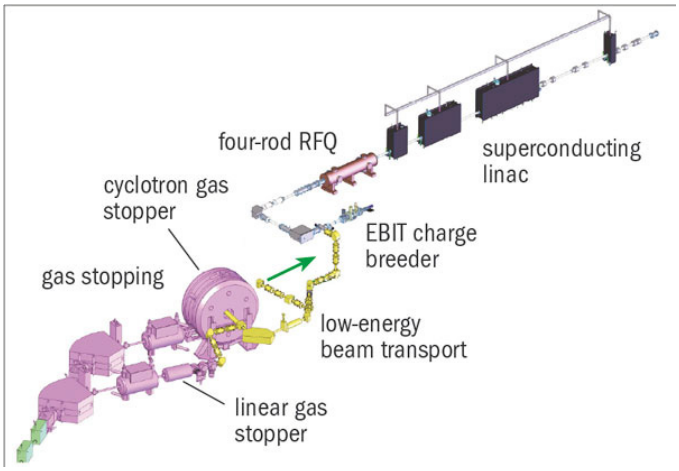

Figure 4 Producing low energy radioactive beams.

The reacceleration line is shown schematically in Figure 4. As most of the stopping techniques allow the ions to recombine with electrons the EBIT charge breeder shown in Figure 4 is required to restore a high charge state to the ions so that the LINAC can efficiently accelerate the resulting stopped beam. Reaccelerated beam experiments are scheduled to start in 2012.

III. Facility for Rare Isotope Beams

Many of the interesting isotopes shown in figure 3 are labeled as "Terra Incognita". This is because they have not been generated at sufficient intensities to allow experiments with them to be performed. This is unfortunate as the *r-process* is believed to take place in this neutron rich realm. The r-process is believed to have produced many of the heavy elements in the collapsing cores of supernovae. In the r-process, as the nuclear matter are compressed, the inner core becomes neutron rich and the nuclei in the less dense outer core can rapidly capture neutrons (r-process is an abbreviation of rapid neutron capture) resulting in very neutron rich, and short lived nuclei. These nuclei decay by sequential β^- decay which converts neutrons to protons, increasing the atomic number (Z) and moving these unstable nuclei step by step closer to the line of stability.

Once more the rates of these reactions, the half lives of these nuclei are important to an understanding of how stars work and how we wound up with the distribution of elements we have today.

To create these neutron rich elements close to the *neutron drip-line* requires higher intensity and higher energies than can be produced by the accelerator systems at the NSCL. To meet that research need, the Nuclear Science Advisory Council (NSAC), in a report presented to the DOE in August 2007, recommended that "DOE and NSF proceed with solicitation of proposals for a FRIB based on the 200MeV, 400kW superconducting heavy-ion driver linac at the earliest opportunity."[1]. In this passage FRIB is an acronym for a "Facility for Rare Isotope Beams" and is pronounced eff-rib.

As a result of a competitive proposal process, the DOE selected Michigan State University and the NSCL to construct this facility in 2008. "The Facility for Rare Isotope Beams (FRIB) will be a new national user facility for nuclear science, funded by the Department of Energy Office of Science (DOE-SC) Office of Nuclear Physics and operated by Michigan State University (MSU). FRIB will cost approximately $600 million to establish and take about a decade for MSU to design and build." [2]

Figure 5 shows the schedule for the construction of this facility. The milestones labeled CD-*n* are *critical decision* reviews. These are making or break reviews of the project progress. The NSCL has successfully passed the CD-1 review and is actively preparing for CD-2 at the time this paper has been written. CD-3 approves the start of the construction and CD-4 is a pre-startup approval.

Michigan State University as further committed funds to support an early start of conventional construction in 2012 approximately one year ahead of schedule.

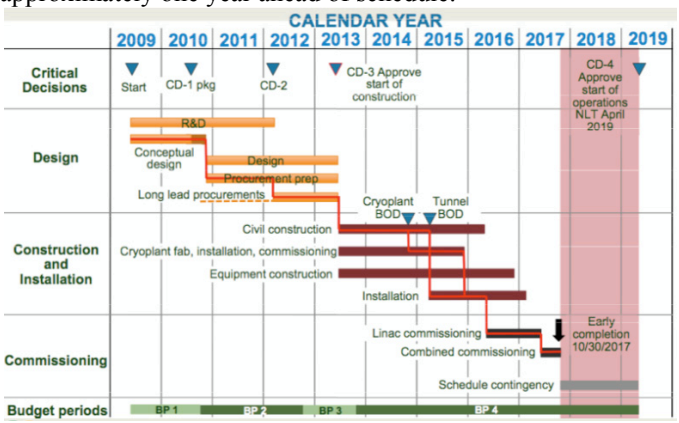

Figure 5 FRIB timeline.

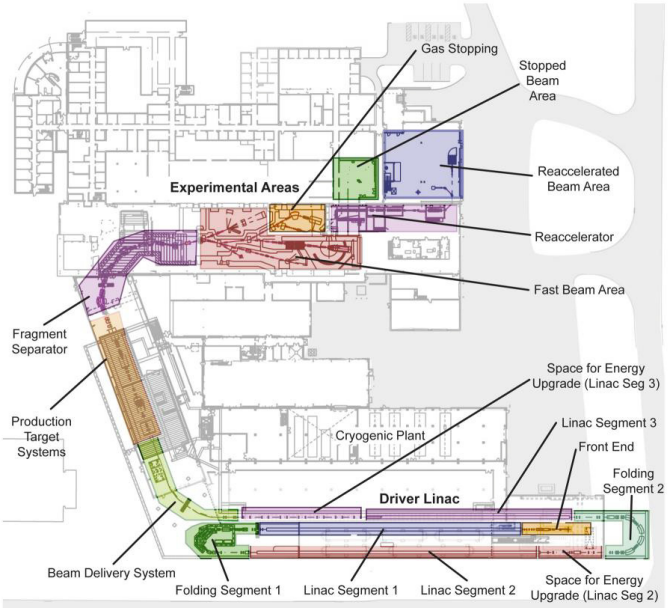

Figure 6 FRIB as planned.

Figure 6 shows the current plan for FRIB. The plan allows for a re-use of the experimental areas and much of the fragment separator, by placing a stacked multistage LINAC driver in a tunnel to the south of the current building. The plan also provides for a later upgrade to the LINAC energiesw by adding space for extensions to two of the planned LINAC segments.

The future looks bright for making the early completion date of late 2017 paving the way for physics runs to start in 2018.

IV. TCL AT THE NSCL

A. *History of the First Adoption*

The first use of Tcl/Tk at the NSCL traces back to the commissioning of the S800 spectrograph. The S800 is used by over 50% of the experiments at the NSCL. The spectrograph is shown in figure 7 below:

Figure 7 S800 Spectrograph

For scale, note the three experimenters at the base of the spectrograph.

The S800 is usually run with two detector packages. The white box at the top of the S800 is the focal plane of the spectrograph and contains 2-d position sensitive detectors as well as particle Id detectors, and instrumentation to provide time of flight information through the spectrograph. The experiment target is located at the base of the spectrograph and is often surrounded by an experiment specific detector package.

In 1996 when the S800 was commissioned, the readout systems associated with the detector packages were not powerful enough to handle both packages while maintaining a reasonable dead time. Therefore it was decided to use a readout system for each of the detector packages and to do event building via a reflective memory system that connected the readout nodes.

The readout computers at that time were controlled by RS-232 ports that were connected to terminal servers. We needed a simple method to provide a control interface to users while sending duplicate commands to both systems.

In the previous year, the NSCL had hosted the IEEE 9'Th Biennial conference on Real-time Computer Applications in Nuclear, Particle and Plasma Physics (RT-95). At that conference, Gene Oleynik et al. presented a paper describing the run control system of the FNAL DART data acquisition system, a far more distributed system than required by NSCL experiments.

The DART team chose Tcl as the basis of an implementation of a group communication protocol inspired by the ISIS Distributed Toolkit [3]. They also chose to build user interfaces from Tk. From Oleynik's paper: "We chose TCL because of its extensible interpretive procedures. For graphics, we chose TK…our experience has been that interfaces can be built more quickly with TK than from X…or Motif…The ocp GUI…took on the order of ½-1 hour...We feel this is a big success of the TCL/TK approach."[4] (Capitalization of Tcl and Tk from that paper).

Based on this endorsement of Tcl/Tk and a similarity between the applications (the Readout systems could be thought of as a group containing two members and communication with them implemented as a group communication problem), the S800 run control software was implemented completely in Tcl/Tk. A low level group communication mechanism was built on top of the [socket] command, it was possible to specify an arbitrary number of target system for the group (S800 focal plane only experiments could then use the same software). A simple state machine was built to manage the system state diagram. On top of all of this Tk was used to build a GUI with which the experimenters interacted.

Our experience with using Tcl/Tk for this project was similar to that of the Fermilab group. The entire system came together in a matter of a day or so, including the time required to learn the few bits of the Tcl/Tk language needed to implement the software.

B. *Coupled Cyclotron Facility and adoption of Tcl/Tk.*

Wide-spread use of Tcl/Tk at the NSCL did not occur until the software development group was tasked with creating a new data acquisition and data analysis tools for the coupled cyclotron facility (proposed in 1994 funded in 1996 and commissioned in 2001).

The functional goals of this development project included:

- Breaking the NSCL's dependency on proprietary software (specifically VMS and Tru64).
- Providing better accessibility to the software in the readout computers (which up until now had been embedded computing systems with a very minimal operating system).
- Providing near turnkey online analysis solutions with a high degree of flexibility with a low accessibility threshold to researchers that were not trained computer professionals.
- Provide a high degree of extensibility and customizability for all these systems.

We had as an additional goal to introduce the researchers at the NSCL to modern (at the time) programming techniques.

The data Acquisition system was largely implemented in C++, introducing object oriented techniques to the researchers which, at the time, were largely a FORTRAN speaking community. Each piece of software that required user interaction embedded a Tcl interpreter with an extended set of commands to control the functions of that program. This philosophy is in keeping with Ousterhout's original motivation for developing Tcl as described in the Preface to [5].

A block diagram of the data acquisition system as it is typically used is shown below in figure 8. Components that embed a Tcl interpreter or that are entirely written in Tcl are indicated.

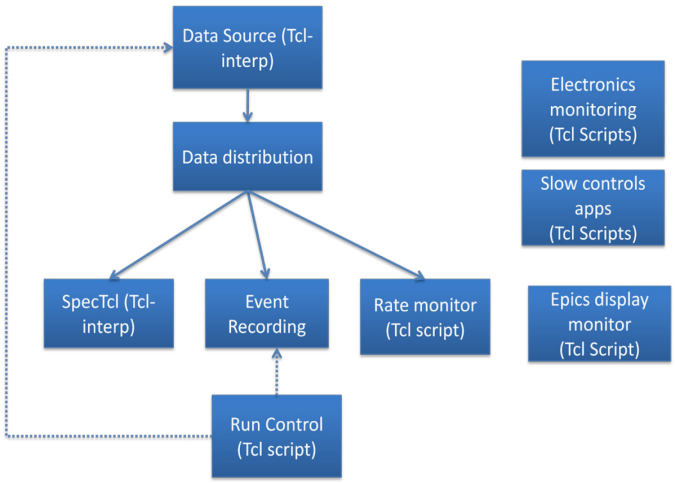

Figure 8 Structure of NSCLDAQ.

The solid arrows represent the flow of event data while the dotted lines represent control flow. Tcl is involved in all but two of the nine boxes in figure 8, and in the case of the boxes to the right of the figure, each box may represent more than one program used by the experiment.

The system was ready for use two years ahead of schedule, in 1999 as evidenced by a description of the data acquisition system and the analysis program SpecTcl in two NSCL 1999 Annual report articles. The gain from using Tcl is best described by a quote from one of those articles:
"Components we provide are often used in ways we did not anticipate. This is a good thing. We intend to use the Tcl/Tk scripting language as a base command language for all components of the system. This allows us to support run-time extensions of the functionality of the software and its user interface via Tcl/Tk scripting. It also allows support for compile time extensions of the command set via C++ wrapper classes around the Tcl command registration procedures. Tcl/Tk scripting provides a common basis for automating tasks within the data acquisition system. The Tk component provides powerful GUI creation and modification tools available to all interactive components" [6].

V. How Tcl and Tk are used at the NSCL.

Tcl and Tk are used in the following ways at the NSCL:

- An embedded command language for applications.
- To provide application specific languages and configuration languages.
- To provide enabling components on which pure Tcl/Tk scripts can be built.
- As a scripting language for applications.

The remainder of this section will provide case studies and references to the uses of Tcl/Tk described above.

A. Tcl/Tk as an embedded command language.

Embedding Tcl/Tk and application specific extensions as the command language for an application was the original intent of Tcl. Using Tcl in this way provides several free benefits:

- Common flavor of command language across all applications.
- Ability of application users to automate commonly performed operations as Tcl scripts and [proc]s.
- Ability, via the Tk package facility to provide a GUI front end to the application and for the users of the application to either extend or replace this GUI with one more suited to their use of the application.
- Ability via a well defined internal API and the [package require] command to provide a plug-in architecture that provides for extensions to the application base functionality, and the ability to selectively add these plug-ins at run-time.

The flagship Tcl/Tk application at the NSCL is nsclSpecTcl [8] the online/offline event analysis/histogramming application. Users have extended it in many ways that were not originally foreseen in the design including the replacement of its visualization package with a Tcl/Tk client called SpecTk [9]. Both SpecTcl and SpecTk were described in earlier Tcl conferences.

B. Application specific languages and configuration

Applications that operate in this way use Tcl and extensions to steer the way they operate. The normal pattern of usage is that sometime during the execution of a program, a Tcl interpreter

is created and possibly extended. A script is sourced into the interpreter and used to build data structures that define how the program will operate.

The readout software for the focal plane of the A1900 fragment separator uses this technique in its simplest form. A configuration file that consist of a bunch of Tcl [set] commands provide values to Tcl variables that are examined by the C++ level software and used to instantiate readout objects for the various detector packages that can live in the A1900 focal plane.

Taking this to its logical extension, [10] describes using Tcl as a basis for a domain specific language that describes and configures the digitizer devices used in a nuclear physics experiment. The Readout software uses scripts in this language to initialize and configure the described modules and to construct the operations required to read out those modules in response to an event trigger.

The experiment configuration script is also processed NSCLSpecTcl selecting the set of event processors required to process raw events into parameters, and to turn those parameters into an initial set of raw spectra. This technique brings Tcl's high level of abstraction into the domain of defining an experiment leading to what the experimenter believes to be 'programming free' experimental setups.

Figure 9 shows an actual segment of a configuration script used to describe the readout of the Particles And Non-Destructive Analysis (PANDA) detection system used by the Finish nuclear safety organization (STUK)[20]:

```
madc create dsssd1.x  -base 0x40000000 -id 4 -ipl 0
madc config dsssd1.x  -gatemode common -gategenerator
disabled
madc config dsssd1.x  -inputrange 8v
madc config dsssd1.x  -timestamp on -timingsource vme \
                      –timingdivisor  $madcTimeDivisor
madc config dsssd1.x  -thresholds $thresholds(dsssd1.x)
stack create event
stack config event -trigger nim1
stack config event -modules [list  fadc
stack config event -delay 40
set       adcChannels(dsssd1.x) $xstrips
lappend adcChannels(dsssd1.x) timestamp
```

Figure 9 Sample Experiment configuration

C. *Enabling components and their applications*

An enabling component usually takes the form of a Tcl loadable package. The package is normally written by the software development group and provides access to some facility that is not easily accessed by Tcl itself. Researchers use these packages to write pure Tcl scripts to perform operations that they would otherwise find difficult.

While several packages have been written that could be classified as enabling components (including plug-in for nsclSpecTcl), this section will focus on the capabilities and application of two of them, Vme and epics.

1) *Vme package*

Many hardware components in experiments run at the NSCL are VME cards. VME bus started out as a multi-master computer bus and is now an ANSI/IEEE standard (ANSI/IEEE 1014-1987). As used at the NSCL, however, this bus is largely an instrumentation bus, providing power and data transfer to a host system for experimental electronics.

The Vme package provides access to this backplane from Tcl scripts. The package itself was described in [11]. It provides a mechanism for declaring interest in address windows within the VME and performing simple pokes and peek operations within those windows.

Researchers typically use this package to build graphical user interfaces to control devices that are not in the primary event data flow. Figure 10 below is a screen shot from one of these applications, the discriminator control program for the CAEsium iodide Detector Array (CAESAR) [12]:

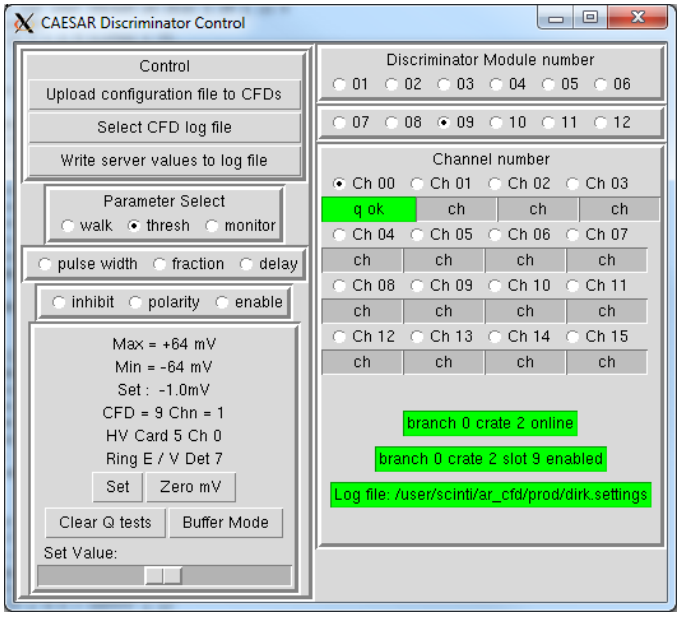

Figure 10 CAESAR discriminator control panel.

This application was written by Andrew Ratkiewicz and NSCL nuclear physics graduate student.

2) *Epics Tcl package*

The Experimental Physics and Industrial Control System [13] (EPICS) is a control system in common use at accelerator labs. EPICS is used to control accelerators and also to provide control over some experimental devices. For example, the S800 magnets are all controlled via EPICS.

For some experiments it is critical to be able to know the state of the beam line leading up to the experiment or the state of the experimental devices themselves. Furthermore, accelerators tend to be one-of-a-kind devices and when commissioning them it is not always clear what human operator interface is actually required. The Epics Tcl package was built to address these needs. It enables physicists accelerator physicists and operators to rapidly build monitor and control interfaces via Tcl/Tk as well as via snit epics specialized mega widgets that are provided with the package.

The package itself was presented at Tcl 2007[14]. It provides mechanisms to access EPICS channels (called Process Variables in EPICS nomenclature), to bind them to variables and to bind traces to them. A feature of the EPICS package that supports programming in the large is the ability for a one-to-many binding of process variable to Tcl variables, along with application wide process variable coalescence. This allows the programmer to specify an Epics channel, and link variables to it without being concerned about whether the execution trace of the program has already linked to the same process variable elsewhere. Changes in the underlying process variable update all linked variables. Changes in any one linked variable set the corresponding Epics process variable eventually triggering and update of all process variables.

The Epics package played a key role in the debugging and commissioning of the ReA3 re-accelerator. Two accelerator operators build the entire control and monitoring console for ReA3 as a set of Tk applications build on the Epics package.

Figure 11 below shows a screen shot the ReA3 beam line monitor application.

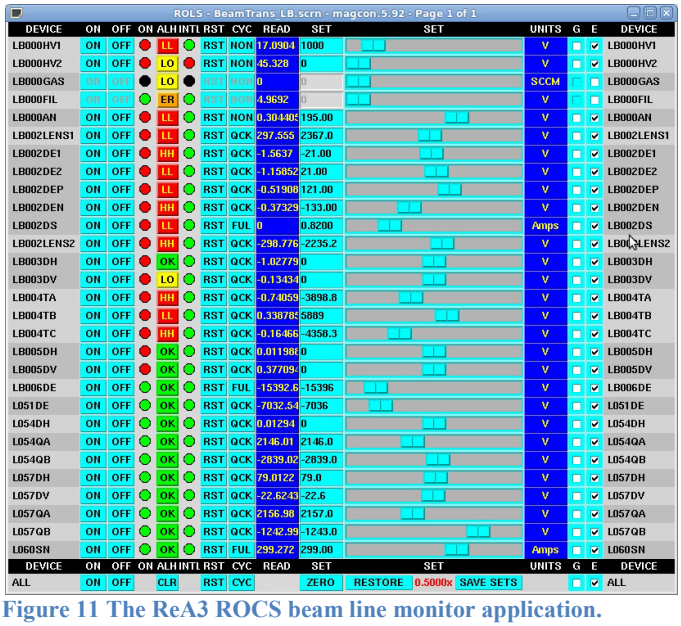

Figure 11 The ReA3 ROCS beam line monitor application.

3) SpecTcl

SpecTcl itself can be thought of as both an enabling technology and an application. Daniel Bazin has implemented a commonly used graphical user interface front end on top of SpecTcl. This front end is shown below in Figure 12:

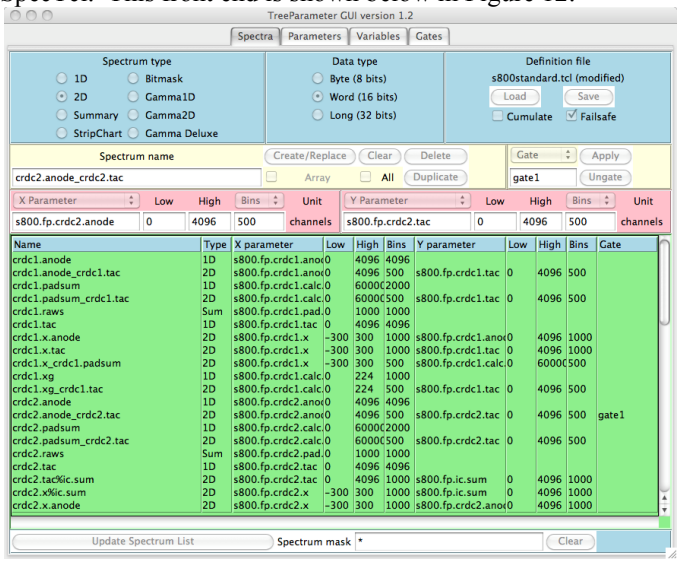

Figure 12 SpecTcl GUI front end

Many other experimental groups have leveraged SpecTcl, and Tk to produce control panels of their own that select data sources or steer the analysis performed by their experiment dependent code.

D. Pure Tcl uses

Tcl and especially Tk are also used as a language to write complete applications. One very successful application is an access controlled 'TclServer'. This is simply a Tcl script that accepts connections from a well defined set of client and accepts Tcl commands over a socket from them. The server is often used in conjunction with a Tcl script that manages a pool of server ports and serves as a directory for those ports enabling clients to discover the ports on which various applications are listening for connections.

VI. Conclusions and a look forward

To date, it is safe to say that Tcl/Tk have removed a great deal of the programming load from the software development group at the NSCL. That load has been transferred to end user community by a mixture of tool and application building. An educational program to teach the basics of Tcl to the first generation of graduate students was also useful as knowledge tends to be passed down from one generation of graduate students to the next.

The transfer of programming load from a software development group to the user community is only possible in a community that has a relatively high technical level. The NSCL research staff fit that profile. In our community the end users were actually grateful for the empowerment that Tcl/Tk and the tools we wrote provided. It allowed them to quickly iterate between versions of user interfaces to see what worked best for their application needs. If we had been involved in each iteration of every application, I can only imagine the frustration that would set in. In the end it is likely that model f

development would have led to a willingness to settle for sub-optimal solutions.

This empowerment has some cost as well:

- Bad code can be written in any language and physicists are renowned for their ability to demonstrate this fact. This has led to a number of Tcl applications that are essentially un-maintainable even by the group that wrote it. This also results from the rapid cycling of generations of graduate students who are often tasked to develop support code for research groups.
- In addition to knowledge being passed from graduate student to graduate student, folklore is passed as well. This folklore is usually based on a poorly understood solution to a problem that was not well understood in the first place. It can take a great deal of effort to dispel the folklore and associated rituals that spring up around it.
- While the users generally develop user interfaces that meet their needs, they do so by learning the minimum needed to do this. This means that:
    - Interfaces might benefit from the use of widgets the users are not familiar with.
    - There are no user interface standards between or even within groups. That results in having to learn each application from scratch rather than being able to start with knowledge gained from the use of other applications.

The use of Tcl in the nuclear physics community has been largely driven by the widespread adoption of NSCLSpecTcl by the NSCL user community. As such it is appropriate to look in to the future to try to understand what the data acquisition and analysis environment might be at FRIB.

As users have become more comfortable with object oriented techniques, they have also adopted object oriented tools.

- Root[15], developed by R. Brun at al. at CERN for LHC experiments is gaining increasing popularity for late stage data analysis amongst all users in the nuclear physics community.
- Python [16] is also gaining in importance as a scripting language in the community.
- Finally with the advent of good Java implementations of the Abstract Interfaces for Data Analysis (AIDA) [17], physicists are also increasingly turning to Java and its large (though sometimes cumbersome) set of libraries.

If Tcl/Tk is to compete it must meet several challenges:

- One or more OO toolkits must be sold effectively to break the impression that Tcl is only an imperative language.
- Software groups that support nuclear physicists must be encouraged to forge interfaces between Tcl and existing software such as Root and AIDA based applications such as the Java Analysis Studio (JAS) [18], or the Python based Hippo Draw [19]. Jacl and Swank may be of some use in the AIDA front and a set of effective Tcl bindings to Root would help there.
- The benefits of the simplicity of the Tcl language and the speed with which that simplicity enables development must be actively sold.
- The fact that Tcl is an 'old' language needs to be placed in context. C is still a highly used language, however it dates from 1969-1973 while Tcl originally emerged in 1988.

In conclusion, I believe that Tcl has provided a great deal of benefit to the nuclear physics community. If, however it is to continue to be of use to that community there are several significant challenges and hurdles that must be overcome.

VII. References

[1] **Report to the NSAC of the Rare-Isotope Beam Task Force** August 20, 2007 available online at http://science.energy.gov/~/media/np/nsac/pdf/docs/nsacrib_finalreport082007_dj.pdf

[2] http://frib.msu.edu

[3] Reliable Distributed Computing with the ISIS Toolkit Birman, VanRenesse Wiley 1994 ISBN: 978-0-8186-5342-1

[4] **Fermilab DART Run Control** G. Oleynik et al. *IEEE Trans Nucl. Sci* NS43 No. 1 February 1996 pp 20-24.

[5] Tcl and the Tk Toolkit J. Ousterhout Addison-Wesley 1994 ISBN 0-201-63337-X pg xvii

[6] *Development status and deployment of the next generation NSCL Data Acquisition System* R. Fox, E. Kasten NSCL 1999 Annual report available online at http://groups.nscl.msu.edu/nscl_library/pub/annual_reports/1999/fox_deployment.pdf

[7] *Status of the SpecTcl Data Analysis Package* R. Fox, C. Bolen, J. Rickard NSCL 1999 Annual report available online at http://groups.nscl.msu.edu/nscl_library/pub/annual_reports/1999/fox_spectcl.pdf

[8] *NSCLSpecTcl Meeting the Needs of Preliminary Nuclear Physics Data Analysis* R. Fox, C. Bolen, K. Orji, J. Venema Presented at Tcl 2004 available online at : http://www.tcl.tk/community/tcl2004/Papers/RonFox/fox.pdf

[9] *SpecTk: a displayer for SpecTcl – or how even a physicist can build a high level application with Tcl/Tk* D. Bazin Presented at Tcl 2005 available online at: http://www.tcl.tk/community/tcl2005/abstracts/scienceandTech/SpecTk.pdf

[10] *A Domain Specific Language for defining Nuclear Physics Experiments* Ron Fox 15'Th Annual Tcl Association Conference Proceedings October 2008 pp105-111

[11] *The Vme Package at the NSCL; Large leverage from a Small Extension* R. Fox presented at Tcl 2007 and available online at http://www.tcl.tk/community/tcl2007/papers/Ron_Fox/vmepackage.pdf

[12] *CAESAR – A high-efficiency CsI(Na) scintillator array for in-beam γ-ray spectroscopy with fast rare-isotope beams* D Weisshaar, A Gade, T Glasmacher, G F Grinyer, D Bazin, P Adrich, T Baugher, J M Cook, C A Diget, S McDaniel, A Ratkiewicz, K P Siwek, K A Walsh **Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment** (2010) Volume: 624, Issue: 3, Pages: 615-623

[13]*Experimental Physics and Industrial Control System* ANL introduction at http://www.aps.anl.gov/epics/about.php

[14] *Tcl/Tk Tools for EPICS Control Systems* R. Fox presented at Tcl 2007 available on line at: http://www.tcl.tk/community/tcl2007/proceedings/Gui/epics.pdf
[15] http://root.cern.ch
[16] http://www.python.org
[17] http://aida.freehep.org
[18] http://jas.freehep.org/jas3
[19] http://www.slac.stanford.edu/grp/ek/hippodraw/index.html
[20]PANDA – A novel instrument for non-destructive sample analysis J. Turunen, K. Parajarvi, R. Pollanen, H. Toivonen **Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment** **V** 613, No. 1, 21 January 2010, Pages 177-183