Software Architecture Considerations for Ion Source Control Systems

John W. Sinclair

Oak Ridge National Laboratory
P.O. Box 2008
Oak Ridge, TN 37831-6368

General characteristics of distributed control system software tools are examined from the perspective of ion source control system requirements. Emphasis is placed on strategies for building extensible, distributed systems in which the ion source element is one component of a larger system. Vsystem, a commercial software tool kit from Vista Control Systems was utilized extensively in the control system upgrade of the Holifield Radioactive Ion Beam Facility. Part of the control system is described and the characteristics of Vsystem are examined and compared with those of EPICS, the Experimental Physics and Industrial Control System.
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, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness 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. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.
1. Introduction

An ion source control system is typically a single component of a much larger system. Additional subsystems deal with ion beam acceleration, transport, mass or energy discrimination and diagnostics. Two decades ago, the advantages of distributed control were explored in similarly structured systems built from the ground up [1]. For the past ten years, distributed control topologies have appeared in many forms. Benefits of such topologies include minimal cabling investment, improved signal quality and the capability to build and test complex systems incrementally as a combination of autonomous subsystems [2]. In recent years, sophisticated tools for building real-time, distributed control systems have become readily available to system designers. EPICS [3], the Experimental Physics and Industrial Control System, and its commercial derivatives along with Vsystem [4,5] from Vista Control Systems are two such tool kits that have generated user communities of international scope. The capabilities of both packages have grown steadily and the current substantial number of facilities that have been built with these tools have more than proven the technology.

The following discussion will examine how these tools fit into the software development process and discuss the characteristics and capabilities of systems built with them. In addition, the way in which Vsystem has been utilized at the Holifield Radioactive Ion Facility will be examined. Finally, basic characteristics of the Vsystem tool kit will be examined and compared with those of EPICS.
2. Software development considerations

One might be lead to believe that a product like Vsystem or EPICS is all that is necessary to build a solid, maintainable system without traditional software engineering disciplines. Indeed, systems have been built under this assumption, sometimes resulting in nonexistent specifications, incoherent documentation and a product that requires extensive efforts to enhance and maintain. Moreover, performance can be disappointing, having characteristics that are at best, poorly understood. Certainly, many successful complex systems have been built without the advantage of sophisticated tools. For example, the success of the Triton control system [6], which reportedly meets high standards for flexibility and extensibility, being implemented in Open Network FORTH, is undoubtedly a testimony to sound software engineering practices. The lack of such practices, even in conjunction with the sophistication of tools like Vsystem and EPICS, will almost certainly result in a poor-quality product. On the other hand, when utilized at the proper level of abstraction, these tools can significantly enhance the development process by providing conceptual simplifications, a powerful set of implementation instruments and a number of invaluable testing and diagnostic resources. In addition, a set of application utilities (archival utilities, alarm managers, etc.) is included in these packages that may adequately satisfy the requirements for some installations. Some of the more powerful characteristics of a system built with tools like Vsystem or EPICS are considered below.

2.1. Data accessibility

Data is globally accessible by name throughout a heterogeneous network. Accessibility by name implies the capability to build data-driven applications where expansion is achieved merely
by editing a database. Network-wide access implies the capability to distribute the load arbitrarily by the addition of processing nodes to the system, either to achieve additional features or to enlarge the existing database. In the later case, the addition of the node is accompanied by repartitioning a portion of the database and associated I/O scanning process into the new node. Furthermore, a system comprised of many devices spread out over a large geographical area can be built as if all data were locally accessible to each processor.

2.2. Data monitoring

Data may be monitored asynchronously on change in value. Instead of polling the database (which may require network access) whenever a process is interested in obtaining the current value of some record, a callback routine may be associated with a significant change in the value of the variable. The callback routine receives the current value, various other related data and a user context.

2.3. Multiple process access

Tools like EPICS and Vsystem provide efficient, multiple-process access to shared information. This characteristic permits the database to hold operator interface data, process derived data, diagnostic data as well as raw device data from I/O scanner utilities. All processes that share information may do so without concern for arbitration among each other as the database access component provides this functionality in a transparent manner. This feature, combined with network wide visibility, provides great flexibility in the partitioning of functional system components.
2.4. System prototyping and testing capabilities

Vsystem and EPICS permit the I/O scanner process to be uncoupled from other processes. As a result, the system may easily be assembled with a component that simulates I/O. In this manner all components of the system may undergo integration testing in a controlled environment before the device interfaces are available. I/O load may also be investigated before the hardware has been fully integrated. In addition, diagnostic device support may be developed off-line and later integrated into the system.

2.5 Summary

These characteristics, taken in concert, result in the capability to build seamless, complex systems from a number of independent programs that may be executed anywhere on the network. This capability is provided by the most fundamental components of both Vsystem and EPICS, namely the real-time database and the record access routines. These two components provide the glue that may be used to transform a collection of otherwise unrelated programs into cooperative peers.

An immediate consequence of this is the inclusion of a sophisticated human interface tool in both Vsystem and EPICS that facilitates the construction of complex graphical display screens that possess all the above mentioned characteristics. Consider again, the previously mentioned collection of programs, functioning as cooperative peers. With the introduction of the display tool (which is but another independent program), the collective unit comprised of all these independent programs evolves in complexity without alteration of the essential architectural structure of any given component. Specifically, each program may now obtain input from an
operator through sophisticated graphical input elements (e.g., sliders, editable text blocks, push buttons, etc.) and generate output to an operator through equally sophisticated graphical output elements (e.g., x-y graphs, bar graphs, multi-colored text blocks, stripcharts, multi-state graphical symbols, animation paths, etc.).

The next consequence of equal significance applies to the matter of device integration. As will be shown in a later section, support for device integration and I/O scanning differs widely between EPICS and Vsystem. What remains in common, however, is the notion that an independent group of processes responsible for device I/O can raise the capabilities of the collection of programs previously considered. Just as the display tool gave these programs the capability to interact with the operator by merely interacting with the database, so the I/O scanner processes give them the capability to interact with many types of complex distributed I/O subsystems. From the perspective of these programs, the database has been extended into the graphical displays and I/O devices.

Building an entire system becomes a matter of identifying data to be shared, populating the database accordingly, interactively building the operator interface screens and writing a set of simple custom utilities where each may be designed to perform some specified task without any concern for I/O details or graphical operator interface details. From this perspective, it is clear that the task of building utilities like mass scans, recipe managers, sequencers and so on, is a simple one that perhaps requires a few days for each.
3. The Holifield target ion source

The target ion source controls for the Holifield Radioactive Ion Beam Facility have been built according to the previously examined scheme. A database has been specified and populated. With the assistance of Vista Control Systems personnel, I/O scanners have been written that accommodate serial I/O, VMEbus accessible Allen-Bradley remote I/O and ethernet PLC I/O. Operator interface screens were then interactively created using the Vsystem display tool. At this point, all system functions were accessible through the graphical human interface screens.

Figure 1 shows an example of a screen that provides access to several ion source parameters. Note the small button in this figure labeled “ramp control.” Pressing this button brings up another screen that interacts with an independent application that assists an operator during a cold-startup of the ion source. This utility controls the ramping of three independent output quantities. For each output, the levels and rate of change of three inputs may be simultaneously monitored. During the ramping process, if a monitored quantity exceeds a specified rate of change maximum or falls below a specified rate of change minimum, the output being ramped is held constant until the rate of change returns to an acceptable value. Furthermore, if the magnitude of the monitored quantity falls outside of a specified set of limits, the output begins ramping in the reverse direction (toward some specified percentage of the starting value) until the magnitude returns to an acceptable value. This utility took two days to complete and was fully tested off-line. It is used to ramp the ion source cathode current and the target heater current while monitoring vacuum.
The anticipated activity of the radioactive ion beam source necessitated the installation of a commercial remote handling subsystem. The subsystem, provided by PAR Systems, consists of a crane, conveyor, robot and two control computers. It manages the transport of a metal cask, which contains the ion source assembly, from a storage area to the beamline and back again. The system was initially designed to execute individual CNC files as directed by commands entered from a local computer console. It was later enhanced to receive commands over a network link. On command, a crane lifts a metal cask containing the ion source assembly from a storage area to the conveyor. Another command engages the conveyor system, moving the cask from the storage chamber to the injector platform. The next command activates the robot which removes the cover from the cask, lifts the ion source assembly, places it into position and then replaces the cask cover. The final command returns the cask to the storage chamber. Another set of commands is given to remove the ion source assembly from the beamline and return it to storage. An application was written that interacts with the database to send commands to and receive information from the remote handling subsystem command processing computer. Figure 2 shows a screen through which the remote handling subsystem is accessed. As before, the operator sees the remote handling subsystem as a seamlessly integrated part of the overall system.

In addition to the ramp control application and the remote handling interface, two mass scan programs, a utility to provide ganged, multi-gain, assignable knobs, and a save/restore utility have been added to the system. With the exception of the knob utility, all applications have been easily implemented, requiring a few days to complete.
4. Tool kit component characteristics

Lutz and Marsaudon have enumerated the logical components of a modern control system [7]. This analysis may not be complete for all architectures, but it does provide a basis for comparing some characteristics of the Vsystem architecture with those of EPICS. Each component will be discussed below:

4.1. The distributed real-time database and channel access module

As it is difficult in discussing one without the other, the database and channel access module will be considered together.

The real-time database (RTDB) contains a description of the shared process data. It might be described as a network-wide, active global common data store whose elements are accessible by name. This component facilitates developing data-driven software that is dynamically reconfigurable.

For example, consider a real-time thread that executes a simple control loop. It is truly a trivial exercise to provide, on demand, the capability of intermediate value logging and display for diagnostic or tuning purposes without impacting the normal execution characteristics. All similar capabilities of dynamically changing state and operation parameters (e.g., controller gains) may, likewise, be trivially implemented with no requirement to poll for value changes. Instead, all parameters of interest are included in the RTDB. During initialization the values are read and a callback routine is associated with the corresponding value change in each parameter. Arbitration
among all processes that might change a given parameter value is managed transparently by the database access module.

These general characteristics are features provided both by Vsyste and EPICS. There are, however, a number of significant differences between the two, both in overall organization and in low level behavior.

4.1.1. **Database organization**

With regard to database organization, a Vsyste database is comprised of primitive nonextensible record types (in the sense that they may not be further decomposed). Each record possesses a value (or a circular array of values) and a number of attributes. It is thus not reasonable to map one database record to an I/O card for which one I/O operation results in the retrieval on N sensor values. Rather, one record might contain references to N other records; the entire set of records would be required to adequately represent the device. Vsyste currently includes support for integer, single and double precision floating point, character string, binary, and time record types.

EPICS, on the other hand, includes support for user extensible, complex record types. A record type could be added to map to the device described above in a wholly natural manner. The current version of EPICS includes support for a large number of record types.
4.1.2. Database location

The Vsystem implementation permits the database to reside on any supported platform. At the time of writing, this includes VAX and Alpha VMS, Digital UNIX, VxWorks platforms, Solaris platforms, Sun OS platforms and HP UNIX platforms. WindowsNT is forthcoming.

In contrast, the EPICS database may reside only on a VxWorks supported platform. The channel access component, however, runs under many UNIX flavors (including Linux) as well as VMS, WindowsNT and the Macintosh OS.

4.1.3. Hardware integration

The manner in which a database record is connected to a piece of hardware represents a significant difference between the two systems.

Vsystem supports the notion of device handlers that are no more than names of dynamically-linked library routines. When a database get operation is performed on an input record, the routine is invoked and when a put operation is performed on an output record the routine is likewise invoked. There is, however, no way for the database to know about changes in the value of a hardware device without the assistance of an external user-written I/O scanner process. Furthermore, though the handler notion works adequately for simple hardware devices, it cannot be used with devices that are comparatively slow and require interrupt driven, asynchronous completion mechanisms.

The EPICS database hardware integration is built into the record type. The general design easily accommodates simple and complex hardware. Interrupt driven asynchronous I/O
completion and I/O interrupt scanning is an integral part of the database. In addition, the EPICS database includes a built-in multi-priority record scanning facility that can monitor changes in hardware without the aid of an external user-written I/O scanner. The topic of scanning I/O devices will be revisited in another section.

4.1.4. Record Linking

Another major difference between the Vsystem and EPICS database is the manner in which records may be linked together.

A Vsystem record may contain references to other records. These references, however, in no way modify the behavior of database access operations.

In contrast, an EPICS record may be linked to other records through a rich set of configuration features. For example, the following configurations are possible:

Figure 3(a) shows the value fan-out capability of record linking. A put to record R1 would result in puts to records R2, R3 and R4 which might perform a multiply operation resulting in the outputs at B, C and D being multiples of the input at A.

Figure 3(b) illustrates how record linking might be utilized to synchronize the update of a group of records. In this case, putting a value to R1 (perhaps done by an operator by clicking a button) would result in R2, R3 and R4 taking an input sample.

Figure 3(c) gives an example of matching dissimilar scan requirements. Here, a process, perhaps doing closed-loop control, is accessing R1 locally and another process is monitoring the value of R2 remotely. The output at B might be changing 100 times per second but the output at
C would change at most once per second. Network traffic would thus be constrained by the scan rate of R2. The input link attribute specified as NPP tells EPICS that R1 should not be processed (which would result in a hardware access) when R2 is processed.

These few examples illustrate only a small fraction of the vast number of configuration possibilities that can be achieved through with EPICS record linking.

4.1.5. Distribution granularity

In principle, a database may be split across two nodes in both Vsystem and EPICS based systems. There is, however, a fundamental difference in the way in which records and databases are specified by the two products that have profound implications.

Vsystem supports the notion of many distinct databases per node each containing a number of records (called channels in the Vsystem documentation). A record is fully specified by the form

<db name>::<channel name>

where <db name> gives the name of the database and <channel name> the name of the channel. The database name must be unique to a given process but the same channel name can appear in many databases. This has the advantage of allowing the specification of a process group private database in which two or more groups of processes all reference the same logical database name but each individual group maps the logical name to a distinct physical database. At the same time, the <db name>::<channel name> notation also adversely impacts the granularity of information distribution. Consider the following case. To reference a database channel, say with the Vsystem
display tool, requires the use of this `<db name>::<channel name>` form. If, for load balancing issues, the database need be split across two nodes, all references in all database access tools would have to be manually changed. This would result in an unacceptable maintenance cost. Therefore, a group of databases may be redistributed over several nodes but channels within databases may not. This places an undesirable artificial constraint as to how the shared data may be distributed.

EPICS supports no such notion of multiple, distinct databases. Instead, a node may be thought of as containing within one database a number of records (called process variables in the EPICS documentation). Process variables may therefore be freely distributed anywhere on the network.

4.1.6. Loss of connection management

The final issue to be considered here is network link disconnection and reconnection. EPICS solves this problem elegantly [8]. When an application makes a call to get a process variable id by name, it may also specify a callback routine to be invoked whenever the network link state changes. When this routine is called it receives, among other things, a user defined context variable and a parameter that indicates whether the connection has broken or has been restored. When the connection breaks, the application needs to take appropriate action (perhaps, for an interactive application, informing an operator of connection loss) and stop using the channel id. When the connection is restored, the application may carry merrily on, continuing to use the channel id as before.
Vsystem provides only one callback per database and the callback is invoked only when the connection breaks. The application not only needs to do all the work in determining when the connection may be restored, but it also must reconnect to the database, find all the channel indices and fix them, and even reestablish all monitor points so that it will continue to receive events when values change.

4.2. The I/O software module

As previously discussed, EPICS provides a great deal of hardware integration support whereas Vsystem provides very little. Vista Control Systems, on the other hand, has written several general purpose sophisticated I/O scanners which come with the distribution in source-code form. It is often a simple matter to add support for additional hardware devices.

As can be easily imagined, some system requirements exceed the capabilities provided by EPICS making it necessary to build custom external I/O scanners. One portion of the HRIBF control system upgrade is an example of just this case. The old tandem accelerator control system was designed to accommodate selectable high speed scanning of analog inputs and outputs. All inputs and outputs are scanned twice per second. Outputs assigned to control knobs and inputs assigned to meters or special purpose DACs, however, are scanned 50 times per second. High speed scanning of analog data assigned to knobs and meters makes possible the practice of spinning the knobs and watching for local maxima or minima on the meters during beamline tuning procedures. The special DAC channels are used to correlate beam jumps and instabilities observed on beam monitors with corresponding power supply transient instabilities. The new control system, based on Vsystem, provides the same dual scan rate capabilities, although the
fast scanning is done at 20 Hz. Scanning all analog channels at 20 Hz would have placed a heavy
load on the CPU, especially with the generation of network events to multiple display tool
processes monitoring many channels for changes.

This notion of a dual scan rate requirement is related to another feature of Vsystem, namely
database channel interest count. When a process, for example the display tool, places a value
change monitor on a database channel, that process is said to have expressed interest in that
channel (more precisely, in value changes for the channel). Another process, for example an I/O
scanner, can then monitor changes in interest for all dual scan rate database channels. The I/O
scanner may then scan all channels at some minimum rate for alarm management and then, for
each channel, when interest count goes from zero to one, the scan rate may be adjusted to the
higher value. Conversely, when the interest count goes from one to zero, the lower scan rate may
be restored.

No such notion of interest count exists in the EPICS database. This is a significant limitation.

4.3. The alarm module

The current Vsystem alarm manager, Valarm, operates only on individual database records
(channels). It should be noted, however, that Vista is currently working on a new, more
sophisticated version.

The EPICS alarm manager includes a rich feature set and operates on a hierarchy of user
configurable alarm groups, where a group consists of subgroups and database records (process
variables).
4.4. The logging and printing module

The Vsystem data logger, Vlog, logs data either on change or at a specified time interval. The stored data may be replayed into a specified database, plotted with a utility named Vtrend or listed in ASCII form as the result of an SQL-like query. Data is written in a documented format but there is no Vista maintained API for data retrieval.

The EPICS data logger stores data in a self-defining data set (SDDS) format developed at Argonne National Laboratory [9]. The stored data may be filtered through a query language and then displayed or plotted. A general purpose API exists for accessing the stored data and, as a result, a good number of data manipulation tools have been interfaced to EPICS log files.

4.5. The display module

Vdraw, the Vsystem display tool, and MEDM, the EPICS display tool, are both excellent interactive human interface builders. A comparison of these components is beyond the scope of the current discussion.

4.6. The sequencer module

Vsystem includes a scripting language named Vscript. The EPICS distribution includes the state notation language sequencer. Both scripting tools provide transparent manipulation of database records.

4.7. Additional modules

The specifics of the test module, simulation module and custom user programs are mostly application dependent and have not been considered in this discussion.
5. Documentation and Installation

Vsystem documentation is well organized and quite complete. Product installation is straightforward and cross platform feature support is highly consistent.

Some amount of novice level information is missing in the EPICS documentation. Several simple concepts became clear only after discussion with experienced users. Training is available and is probably necessary for the inexperienced user. Product installation is straightforward but assumes some experience with UNIX system administration. Cross platform consistency is uncertain and seems to be dependent on specific individuals.

6. Support

Customer support is an important consideration in assessing the viability of software tools.

Vista Control Systems is committed to the highest standards of customer support. Staff members on many occasions have worked long into the night to find and fix subtle problems that sometimes turn up when new versions are initially released. In addition, Vista engineers are always on hand to assist customers in solving needs specific to their applications and environment. A strong sense of partnership has always existed between Vista and its customers.

The EPICS collaboration, by definition, can not provide the same level of commitment and support. There is, however, a remarkable willingness among members of the EPICS community to provide assistance and guidance to newcomers.
In 1993, the Vsystem product was chosen for the Holifield Radioactive Ion Beam Facility control system upgrade. Utilization of the Vsystem tools has resulted in a highly flexible system with a uniform interface. Third party subsystems have been easily integrated and a number of useful utilities have been written without a great deal of effort.

There are some similarities in the higher level characteristics of Vsystem and EPICS. Low level behavior is, however, quite different. The higher level components of both products are powerful tools capable of greatly boosting productivity. They are, however, built on top of core components and therefore inherit all the strengths and weaknesses contained therein. Clearly, there are significant differences in the core components of Vsystem and EPICS. The Vsystem based HRIBF control system contains a considerable amount of custom written code to deal with the loss of connection issue due to lack of such capability in Vaccess. The distribution granularity problem has required a couple of sessions of manually finding database/channel name pairs in order to change the database component. On the other hand, the capability to locate databases on multiple platforms and the notion of process group private databases have been extremely useful. EPICS provides a significant advantage in hardware integration support assuming the supported model meets all system requirements. For example, the lack of an interest mechanism in EPICS could result in the need for custom written code. The emphasis placed on exportability of the EPICS data logs has provided a strong basis for sharing applications between many sites. Customer support, installation assistance and documentation provided by Vista Control Systems
are definitely more suited to the inexperienced user. Some care needs be taken in choosing the implementation platform for a system built with EPICS.

Research at the Oak Ridge National Laboratory is supported by the U.S. Department of Energy under contract DE-AC05-96OR22464 with Lockheed Martin Energy Research Corp.
References


<table>
<thead>
<tr>
<th>Anode</th>
<th>Volts</th>
<th>Amps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.02</td>
<td>0.000</td>
</tr>
</tbody>
</table>

**Anode: Voltage Control**

- **Volts**: 0.00
- **Amps**: 0.00

**Ramp Control**

- **Source Leads**: Mark Restore

<table>
<thead>
<tr>
<th>Cathode</th>
<th>Volts</th>
<th>Amps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.00</td>
<td>-0.04</td>
</tr>
</tbody>
</table>

**Cathode: Current Control**

- **Volts**: 0.00
- **Amps**: 0.00

**Ramp Control**

- **Source Leads**: Mark Restore

<table>
<thead>
<tr>
<th>Tgt Heater</th>
<th>Volts</th>
<th>Amps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.00</td>
<td>-0.06</td>
</tr>
</tbody>
</table>

**Target Heater: Current Control**

- **Volts**: 0.00
- **Amps**: 0.00

**Ramp Control**

- **Source Leads**: Mark Restore

**Source Leads**

- FS1: FLOW OK
- FS2: FLOW OK

**Ramp Control**

- FS1: FLOW OK
- FS2: FLOW OK

---

Figure 1. Sinclair
<table>
<thead>
<tr>
<th></th>
<th>Volts</th>
<th>Amps</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Anode</strong></td>
<td>0.02</td>
<td>0.000</td>
</tr>
<tr>
<td><strong>Cathode</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tgt Heater</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Anode:Voltage Control</strong></th>
<th>0.00 Volts</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mark Restore</td>
</tr>
<tr>
<td></td>
<td>300.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Cathode:Current Control</strong></th>
<th>0.00 Amps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mark Restore</td>
</tr>
<tr>
<td></td>
<td>600.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Tgt Heater:Current Control</strong></th>
<th>0.00 Amps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mark Restore</td>
</tr>
<tr>
<td></td>
<td>1000.00</td>
</tr>
</tbody>
</table>

**Source**

**Leads**

FS1: FLOW OK
FS2: FLOW OK

Figure 1. Sinclair
Target Ion Source Installation

1. Move CCB containing TIS from Cask1 to conveyor
2. Move CCB from C-110 to C-111S along the conveyor
3. If platform and source table are ready for TIS, initiate robotic installation of TIS

Arm Power Status | OFF
MCS Execution Mode | 3
Fault Number | -1
Execution Error String | 19
Scan Plan File Executing
Current Record
Emergency Stop Status | Cleared

Operation Status: OK

4. Initiate coupling sequence

Can Valve | Open
Can Bellows | Engaged
Shutters | In-Transition
Swing Clamp | Engaged
Pusher | Engaged
Target Ion Source Installation

1. Move CCB containing TIS from C-1 to conveyor.
2. Move CCB from C-110 to C-1115 along the conveyor.
3. If platforms and source table are ready for TIS, initiate robotic installation of TIS.

<table>
<thead>
<tr>
<th>CCB pundits - conveyor</th>
<th>Conveyor 019-011</th>
<th>Results TIS personnel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shuttle Retracted</td>
<td>Document Retracted</td>
<td>CCB Levered</td>
</tr>
<tr>
<td>Lid OK</td>
<td>Temp OK</td>
<td>Air On</td>
</tr>
</tbody>
</table>

Arm Power Status: OFF
MCS Execution Mode: 3
Fault Number: -1
Execution Error String: 19
Scan Plan File Executing: Cleared
Current Record: Cleared
Emergency Stop Status: Cleared

Operation Status: OK

4. Initiate coupling sequence:

- Can Valve: Open
- Can Covers: Engaged
- Shutters: In-Transition
- Swing Clamps: Engaged
- Pusher: Engaged
(a) Value fan out

(b) Group update

(c) Dissimilar sample rates

Figure 3, Sinclair
Figure Captions

Figure 1. Ion source human interface example.

Figure 2. Remote handling system human interface example.

Figure 3. EPICS record linking examples.