NICI Temperature Monitoring

1. Introduction

This document describes the software modules that control and interface to the cryogenic thermal devices. These devices monitor the internal temperatures at critical points inside the dewar as well as provide a stable and closely monitored thermal environment for the Aladdin III arrays. The software description will address both user level monitoring and the engineering level monitoring.

2. The NICI requirements from section 7.1.1:

The following is from document TEMP2 800-220-00 REDSTAR3 Dual Array Temperature Controller and summarizes the Customer requirements for array temperature control. The hardware is selected to maintain the real-time requirements. The software function will be to supply the Lakeshore Model 332S Controller with the set points for temperature control and continually monitor the values that are sensed by the Lakeshore Model 332S Controller.
Independently control the temperature of each array to a user settable temperature approximately 30-35 degrees Kelvin +/- 0.1 degrees Kelvin.

The temperature may not vary by more than 0.1 degrees over periods of 10 minutes. Gradual drifts of not more than 1 degree are allowed as long as the change in temperature does not exceed 0.1 degrees K/10 minute period.

The temperature controller shall be serially controlled and allow for readout of the temperature and modification of the set point, the servo loop PID parameters and the heater output power.
Temperature sensors positions will be in the following locations:

1. Array #1 mount
2. Array #2 mount
3. Cold Structure Closed Cycle Cooler first stage
4. Cold Structure Closed Cycle Cooler second stage
5. Camera cold structure (farthest point from closed cycle cooler attachment)
6. Radiation shield (farthest point from closed cycle cooler attachment)
7. Radiation Shield cooler first stage
8. APD temperature

The two array mount temperatures and the cold structure temperature will be reported continuously to the Gemini environment. All other temperatures will be available to the engineering interface. All temperature controls will be available to the engineering interface.

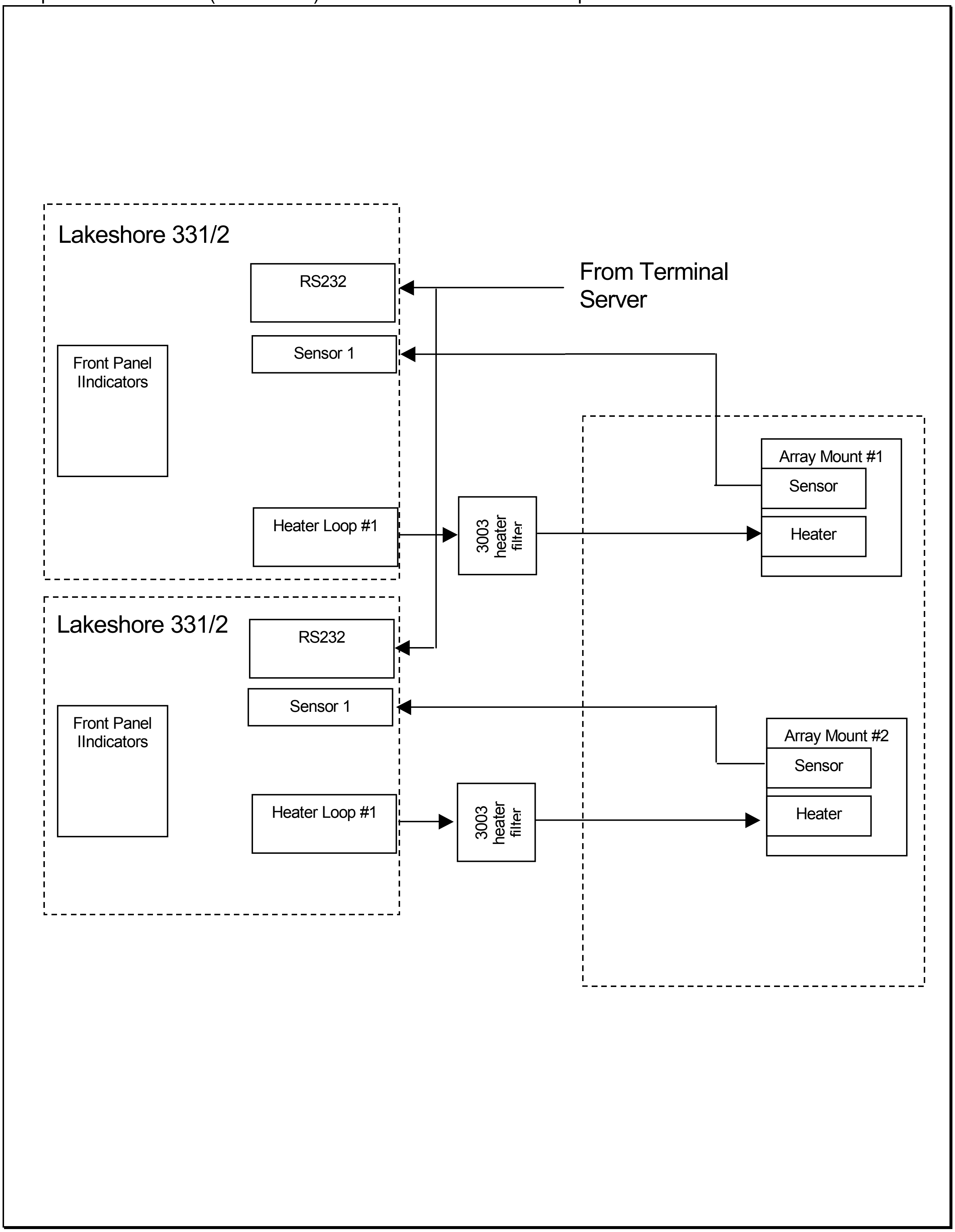
3. How we are going to meet these requirements:

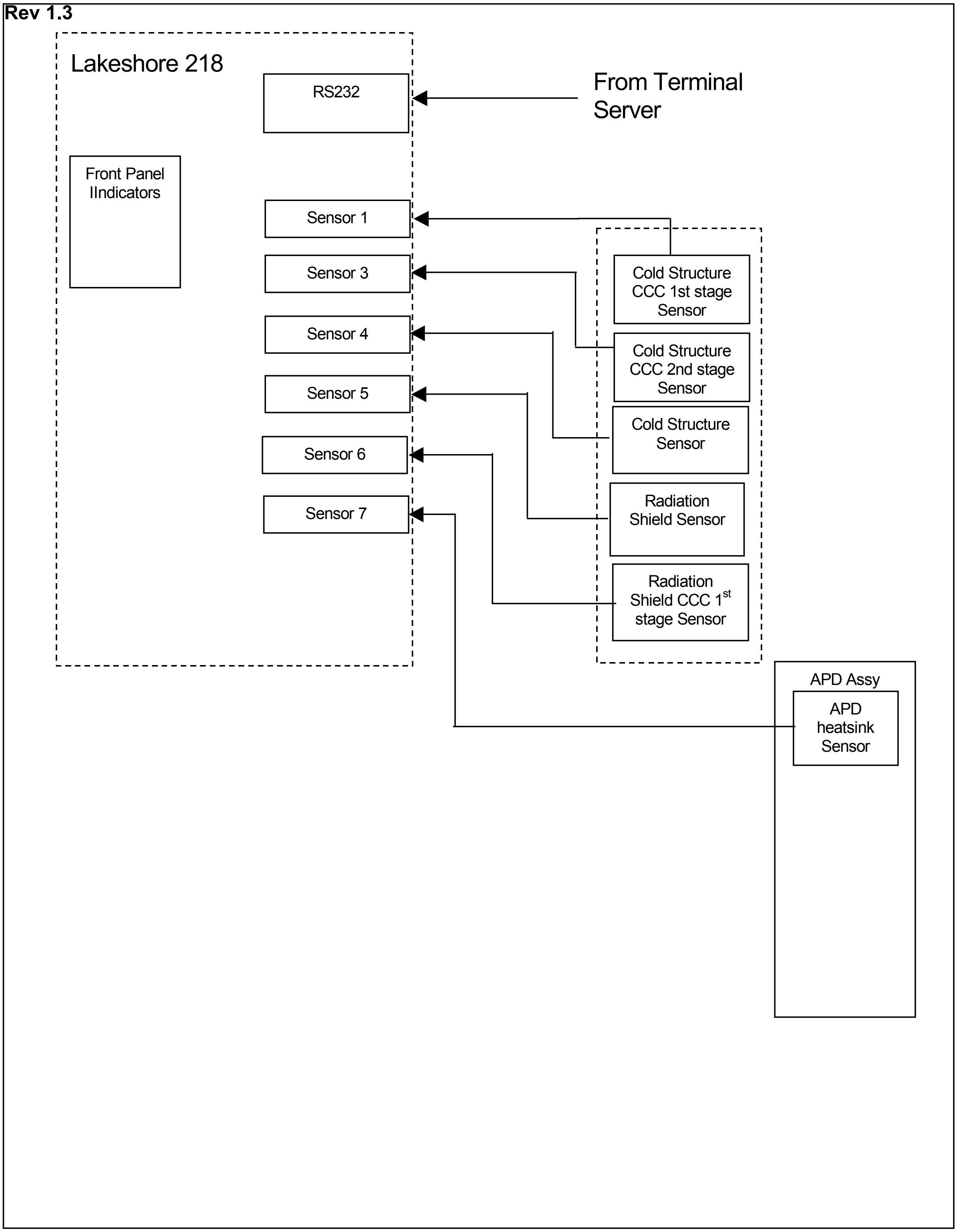
The software access for NICI will be in two domains: user and engineering. This division is to allow the user to access and control those aspects that most directly address the problem domain, i.e. Astronomy and the engineer to access the low level aspects of the physical instrument. Clearly the control of servo loop parameters and set points is not in the user domain, but the user does need to be assured of the proper operation of the NICI instrument and the imaging array.

The software will monitor the temperatures for user level reporting. It will also allow the user to set valid ranges for safe operation. These ranges will allow the user to be aware of proper versus improper operation of the NICI temperature control system. These temperatures will be able to be placed in the descriptive information of the images (e.g. FITS header.) The software will allow pass-through engineering access to the Lakeshore controllers, exactly as if they were attached by way of a serial connection to the engineering GUI.

4. Block Diagram

The following block diagram from the above mentioned document: TEMP2 800-220-00 REDSTAR3 Dual Array Temperature Controller (Section 7.4) shows the control and sense points.





This indicates two lakeshore controllers each with two sensors and two heaters. It also indicates a lakeshore sensor unit with 4 active sensors.

Rev 1.3

The software control shall consist of the following: Each sensor or controller unit will be addressable as independent IP address/port combinations via the terminal interface controller. Each will have a pass through mode for engineering set-up of the lakeshore controllers. This set-up will be in the form of a sequence of commands (macros) that will set these units into normal operating mode. Specifically, the PID values for the sustained operation of the arrays will be calculated by the engineering staff and supplied as part of the set-up procedures. This includes the ‘set-points’ for temperature controls for each of the heater elements.

For ongoing operations the values of the sensors will be read on an ongoing basis and reported to the higher level User interfaces (engineering GUI and robotic clients). These values will also be made available for inclusion in the system data base for inclusion in the FITS header.

The software will control the lakeshore controllers making them appear as simple data elements of the temperature command object. There will be a range of legal temperature values for all of the sensors. When the value goes outside of the range, the system may be forced into an error mode. This may force an automatic recovery (if the terminal interface controller is down, for example) or may shut down one or both of the arrays and signal a serious temperature situation.

The sensors and controllers will be known as software controlled objects with the following names: LC1, LC2, LS for Lakeshore Controller 1 and 2 and Lakeshore Sensor.

Software Names for Temperature Control and Sensing access -- NICI

Equipment position	CO name	Range list	Error?
Array #1 mount sensor	LC1::S1	LC1::R1 = {lowval, hival}	Array #1
Array #2 mount sensor	LC2::S1	LC2::R1 = {lowval, hival}	Array #2
Cold Structure Sensor	LS::S4	LS::R4 = {lowval, hival}	Entire instrument
Entries below only for engineering mode			
Array #1 Mount heater	LC1::H1		
Array #2 Mount heater	LC2::H1		
Cold Structure CCC 1st stage sensor	LS::S1		
(no connection)	LS::S2		
Cold Structure CCC 2nd stage sensor	LS::S3		
Radiation Shield sensor	LS::S5		
Radiation Shield CCC 1st stage Sensor	LS::S6		
APD heatsink sensor	LS::S7		

The Command Objects LC2, LC2 and LS will have the local variables (attributes) as above. Setting the values of the ranges will be done by the ‘configure’ subcommand. The values for the sensors will be automatically scanned from the Lakeshore instrument controller in the following sequence and pacing:

LC1::S1, LC2::S1, LS::S4 (and again from start). Each value will be read sequentially at intervals of 5 seconds. The total scan will take 15 seconds. Each time the value is updated, it will be compared with the valid range. If the range is exceeded an error will be signaled.

When the scan is taking place, if the last value requested has not been sent by the time the 5 second polling interval has expired, the Terminal interface controller will be marked as suspect. If this happens twice, the terminal interface controller will be sent the ‘recover’ command. Scanning will continue. If the terminal interface controller is down for an entire scan (15 seconds), the system will report an error. If the range of valid temperatures is set to the empty set, the comparison will not take place and the value, however weird will not cause an instrument error condition.

The information from the scan will be similar to the user interface issuing the command: `<CO> cget -<variable>`. For example:

```
LS cget –S4 32.05
```

Would be a valid message indicating that LS::S4 currently reads at 32.05 degrees (the trailing command accept/reject word is not present, since this message is NOT generated as a direct result of a user command)

5. Summary:

The syntax of commands across the thin-client—NICI interface to configure temperature ranges or to read out important values would follow the standard Command Object syntax. The specific examples for the temperature subsystem are for setting ranges or reading values as follows.

To set the range values for the command objects, use the following syntax:

```
LC1 configure –R1 { lowval, hival}
LC2 configure –R1 {lowval, hival}

LS configure –R4 {lowval, hival}
```

To disable the value range use the following example syntax:

```
LS configure –R1 {}
```

The values read back will look like:

```
LC1 cget –S1 <value>

LC2 cget -S1 <value>

LS cget –S4 <value>
```

6. Diagrams

The following diagram indicates the static interconnections of the temperature sensing subsystem. That is, how the objects and classes of the TCL program are syntactically and semantically interconnected.

Figure 1

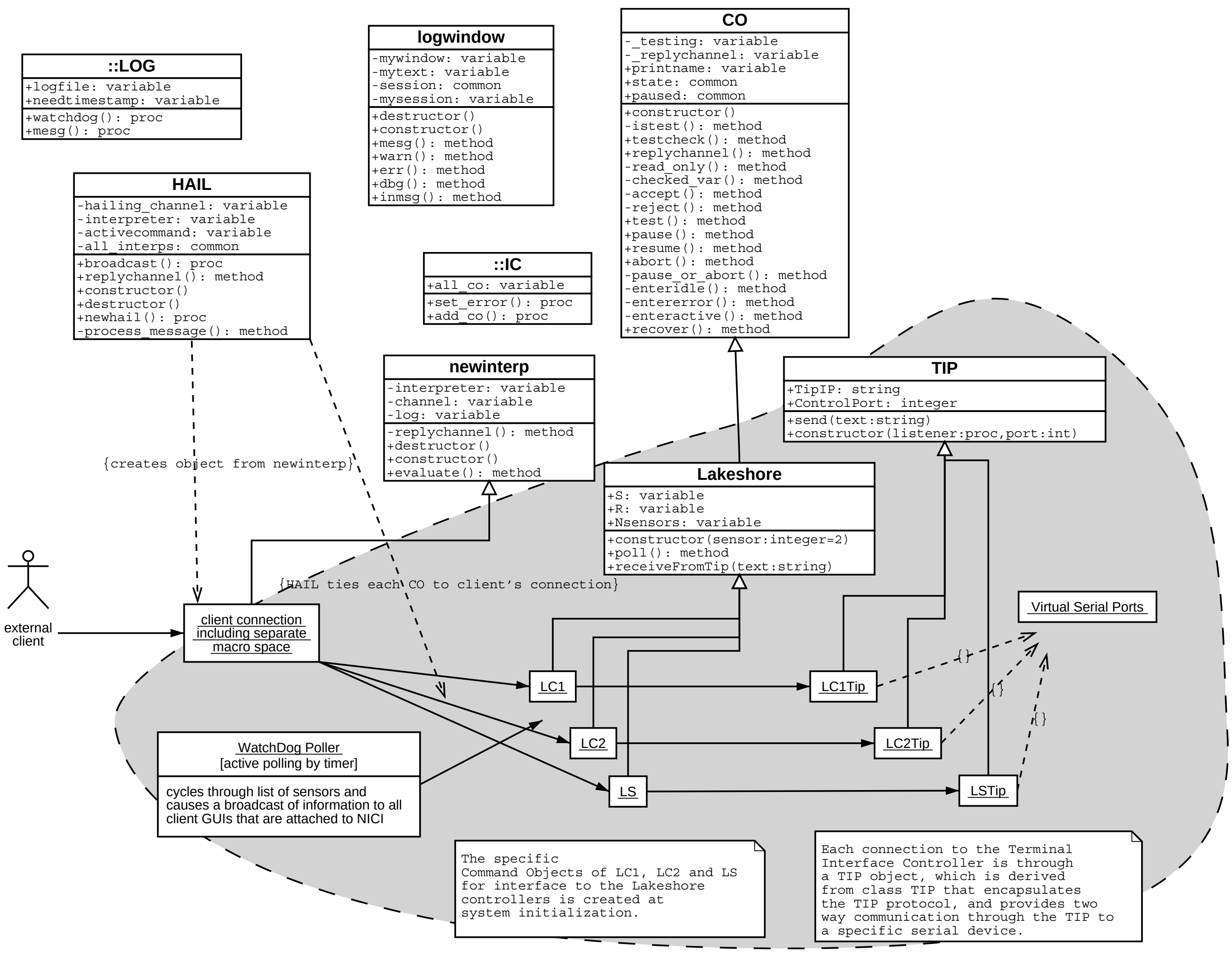
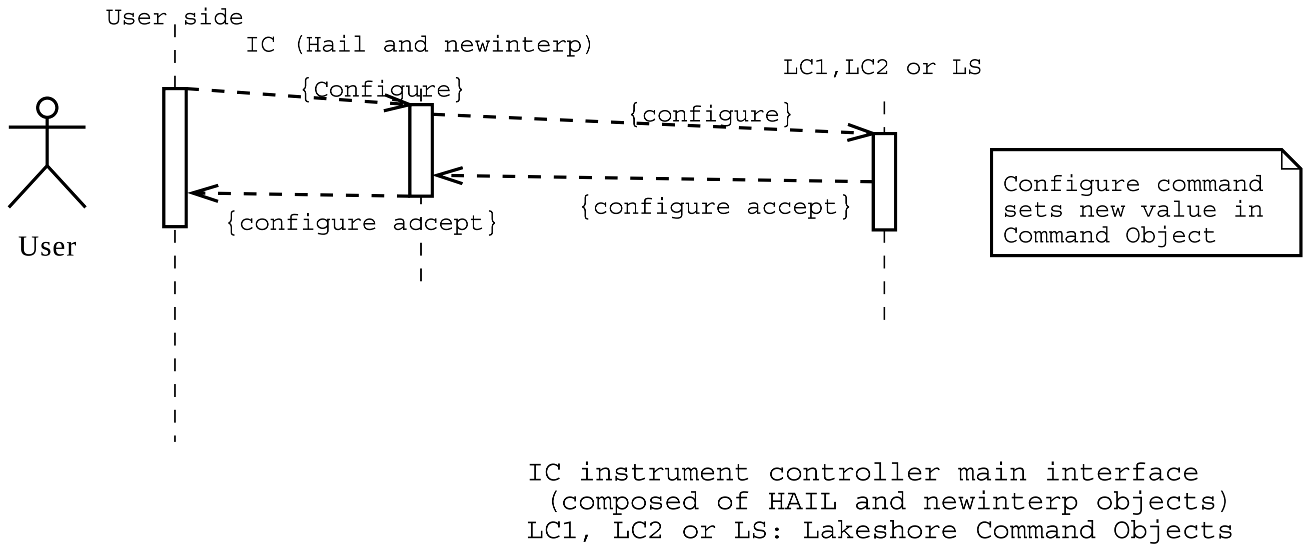


Figure 2

The next diagram shows the handling of setting the valid operational range

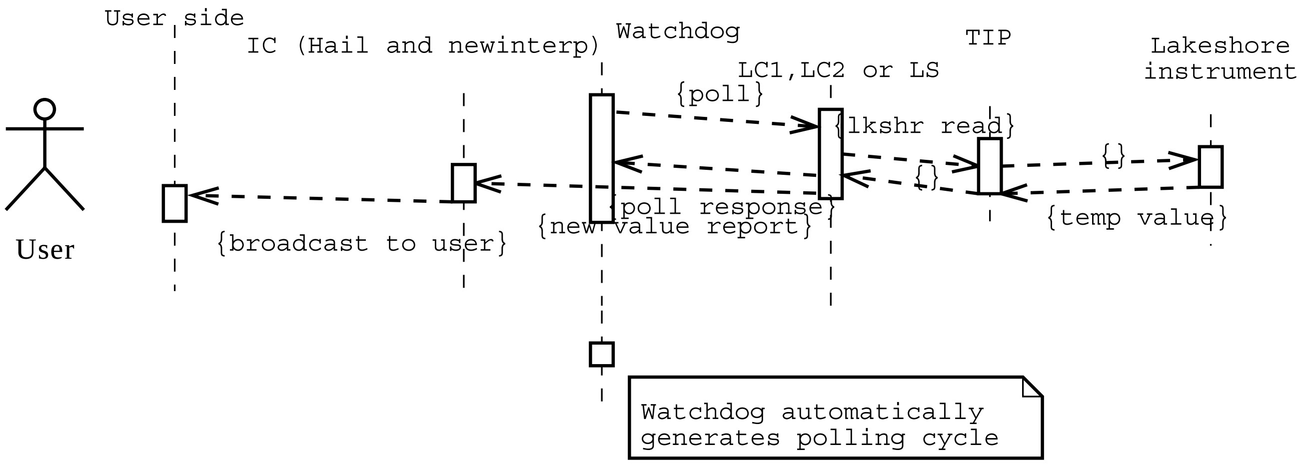


User initiates CONFIGURE command to set varaibles internal to a Lakeshore command object. This is a standard example of how such data access operations are done for any command object.

Rev 1.3

Figure 3

Normal polling of temperature values. (out of range temperature values are considered non-fatal errors, but are flagged for user)



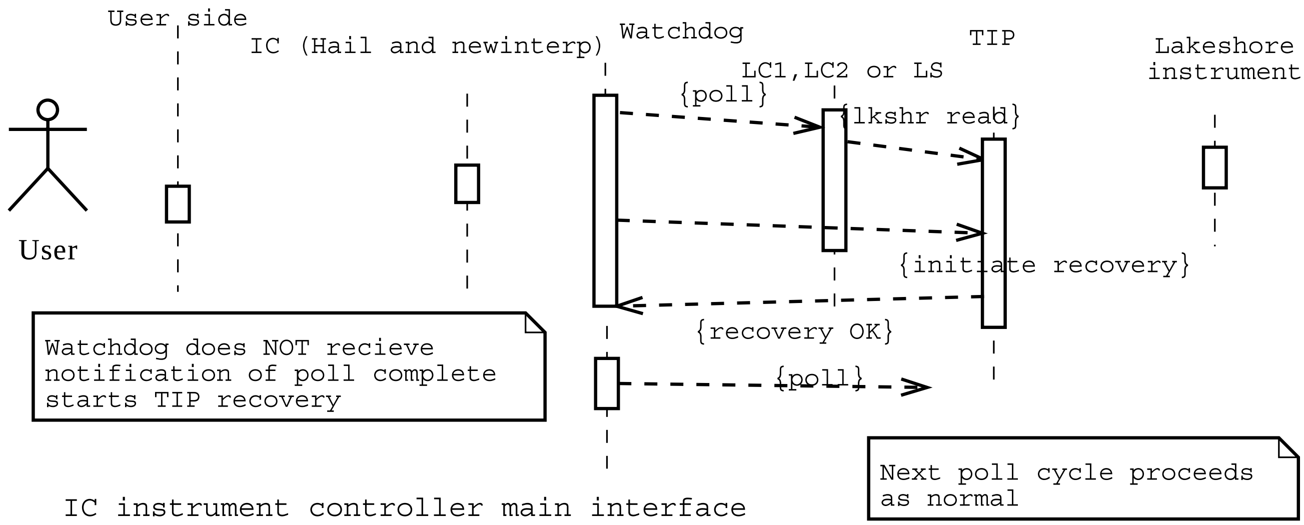
IC instrument controller main interface
(composed of HAIL and newinterp objects)
LC1, LC2 or LS: Lakeshore Command Objects
Watchdog: timer driven object for polling
TIP:Terminal interface controller
Lakeshore: Lakeshore controller via TIP

User receives continual updates about internal temperature by way of a watchdog timer that polls the Lakeshore controllers and BROADCASTS the information back to active user interfaces

Rev 1.3

Figure 4

Abnormal polling. Recovery attempted and possible error flagged.



IC instrument controller main interface
(composed of HAIL and newinterp objects)
LC1, LC2 or LS: Lakeshore Command Objects
Watchdog: timer driven object for polling
TIP:Terminal interface controller
Lakeshore: Lakeshore controller via TIP

In this diagram, the TIP needs recovery, and responds properly, operation proceeds normally. However, If the TIP could not clear it's error condition, or the poll fails twice in a row the error flag is set on the TIP and an error message is broadcast to the user interfaces. A failing TIP is an instrument level error (except in engineering mode) and no functions can be done until cleared and the recover command is sent.

Figure 5 -- engineering screen