ICE: An Instrument Control Environment for the New Millennium

Nick Maliszewskyj[1], Mathieu Doucet[2], Steve Pheiffer[1]

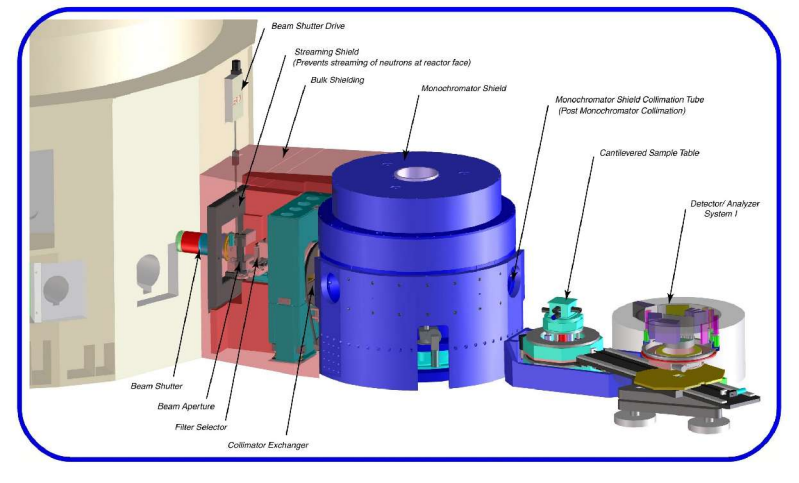
[1]*Center for Neutron Research, NIST, Gaithersburg MD, USA*

[2]*University of Maryland, College Park MD, USA*

Statement of the Problem

- Existing instrument control software (ICP) is mature but showing its age
- Numerous issues with maintenance
  - Adding new devices is major undertaking
  - Platform obsolescence and OS-specific extensions
  - Increasing demands of computer security
- Accommodations need to be made for new users
  - Not everybody is an expert…
- New instruments will require more sophistication
  - Not every instrument is a triple axis…

An opportunity for Change



BT7 Thermal Triple Axis

MACS (Multi Analyzer Crystal Spectrometer)

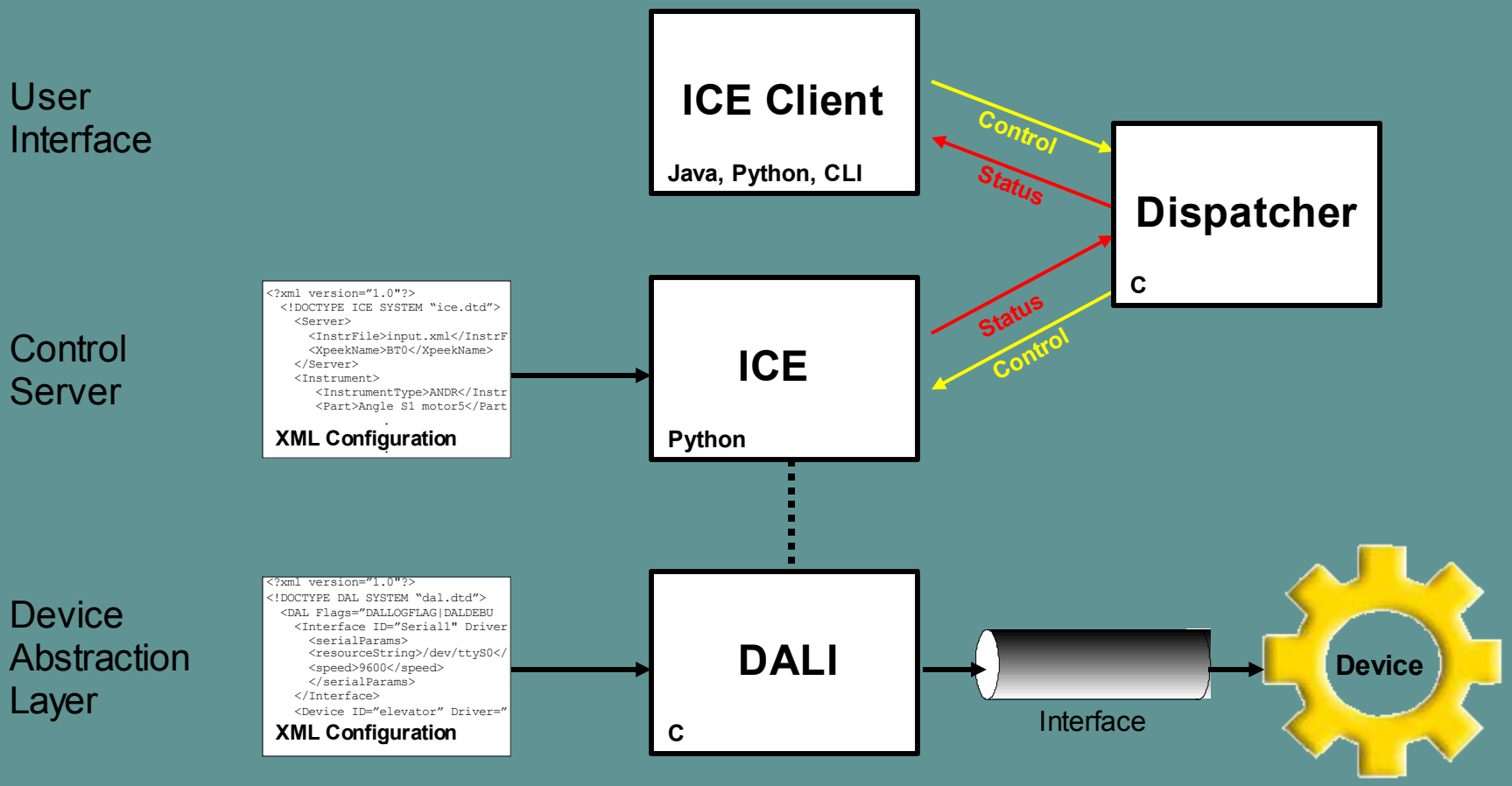
NIST 10M SANS

Advanced Neutron Diffractometer/Reflectometer

Design Goals

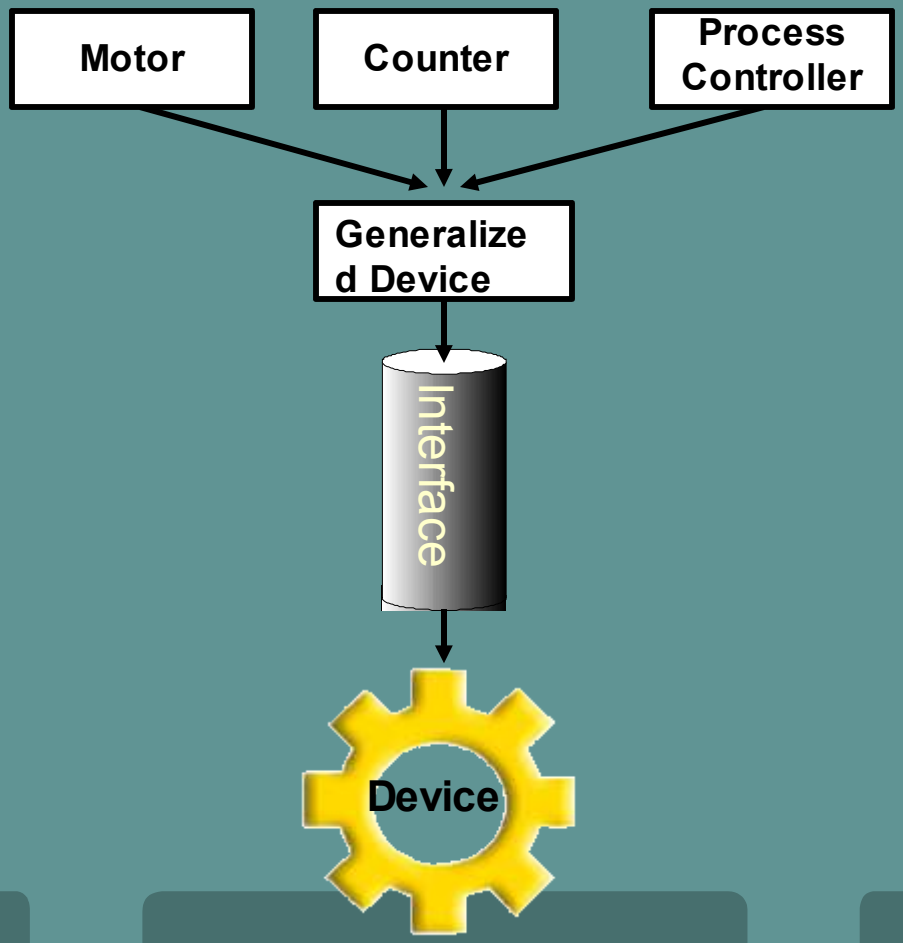
- Extensibility
  - Incorporate new devices and subsystems without requiring a paradigm shift or resulting in spaghetti code
- Flexibility
  - Permit measurements that fall outside routine operations
- Adaptability
  - Reuse infrastructure throughout our facility
- Modularity
  - Easily isolate/exchange bits that need “refactoring”
- Portability
  - Easily migrate systems to new platforms

ICE Overview



DALI: Device Abstraction Layer for Instrumentation

Top level Classes



Devices are registered as belonging to one of the major classes Generalized methods include get/set parameters and action

Generalized interfaces

Interfaces are just conduits for communication and control

Physical devices

DALI Features

- Built on concept of modular device drivers
- Device drivers can be loaded/unloaded dynamically
- Major classes: motors, counters, process controllers
- Devices and interfaces can be managed independently
- Configuration described by XML file
- Written in ISO C and ported to POSIX / Windows

ICE Control Server

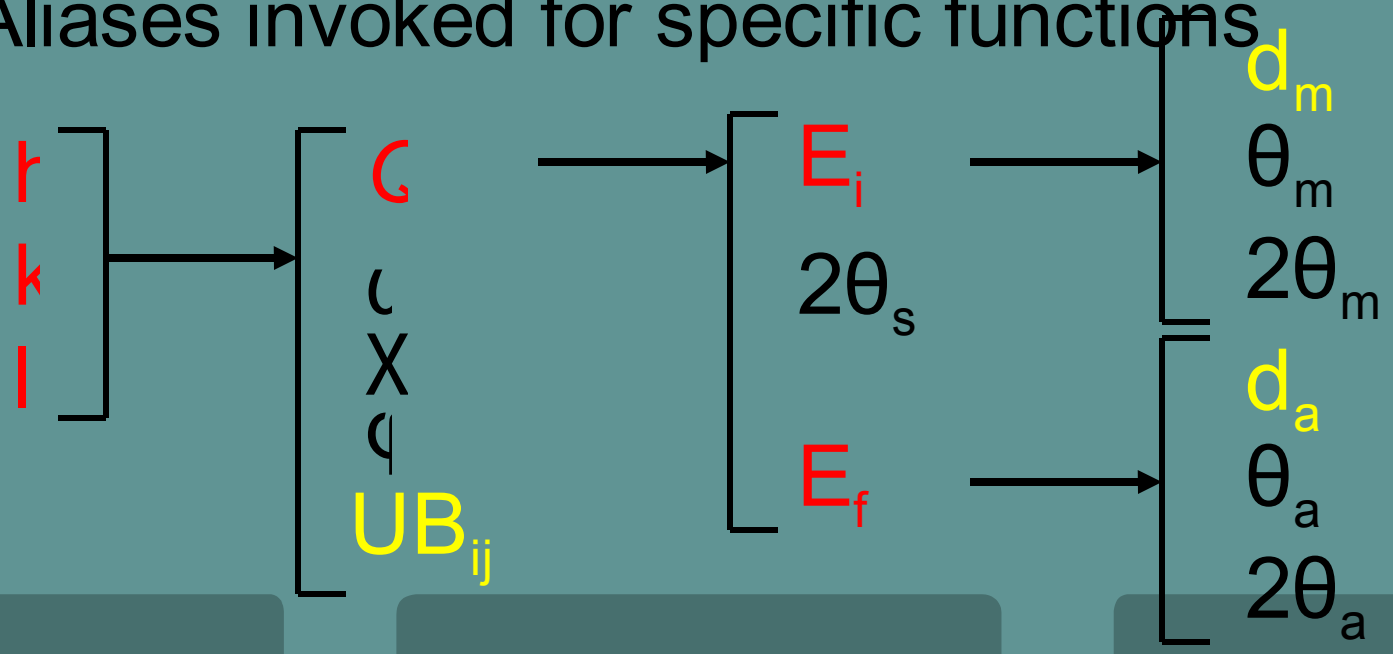
- Runs autonomously as an operating system service
- Implemented as a state machine
- Scans are really considered as a sequence of states
- Server maintains a FIFO to execute commands sequentially
- Commands principally textual
- Tiered permission scheme for remote command execution (local token for full rights)

Virtual Devices

Users will work in experimentally convenient quantities

Transformations handled by instrument-specific modules

Aliases invoked for specific functions

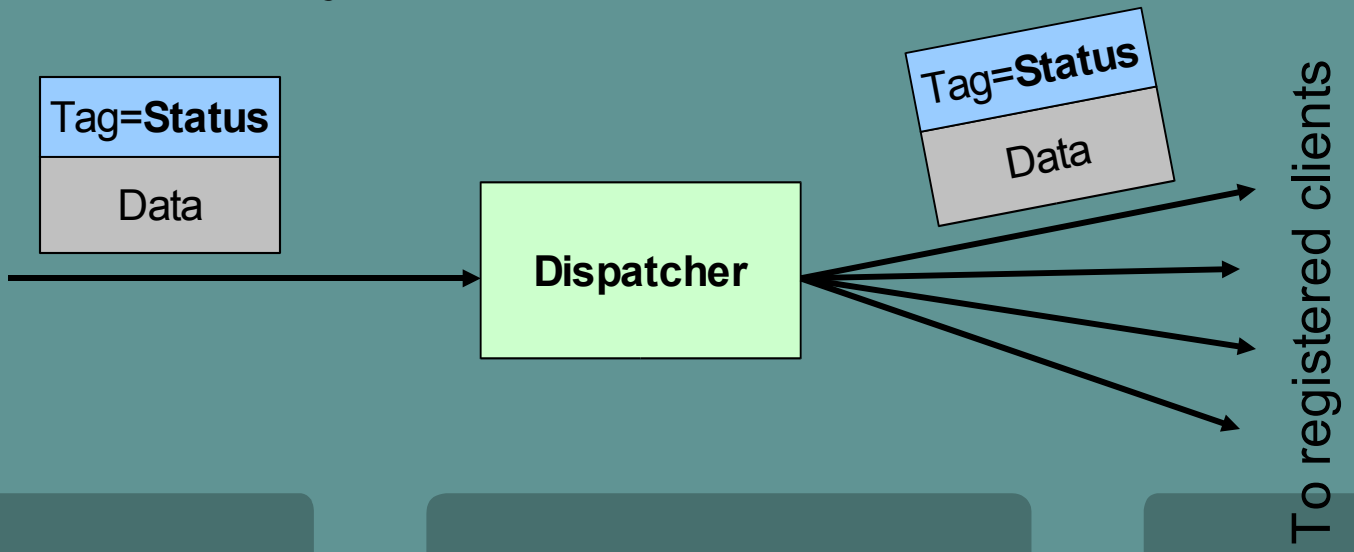


Dispatcher

Independent process that relays messages between server and subscribed clients

Messages are two part: tag and data

Clients only receive messages with tags to which they have subscribed



Remote Command Execution

Local

Immediate Commands: executed immediately

Stacked Commands: processed sequentially in order received

Remote

Remote commands are a subset of local commands

They have the same permissions / restrictions on execution

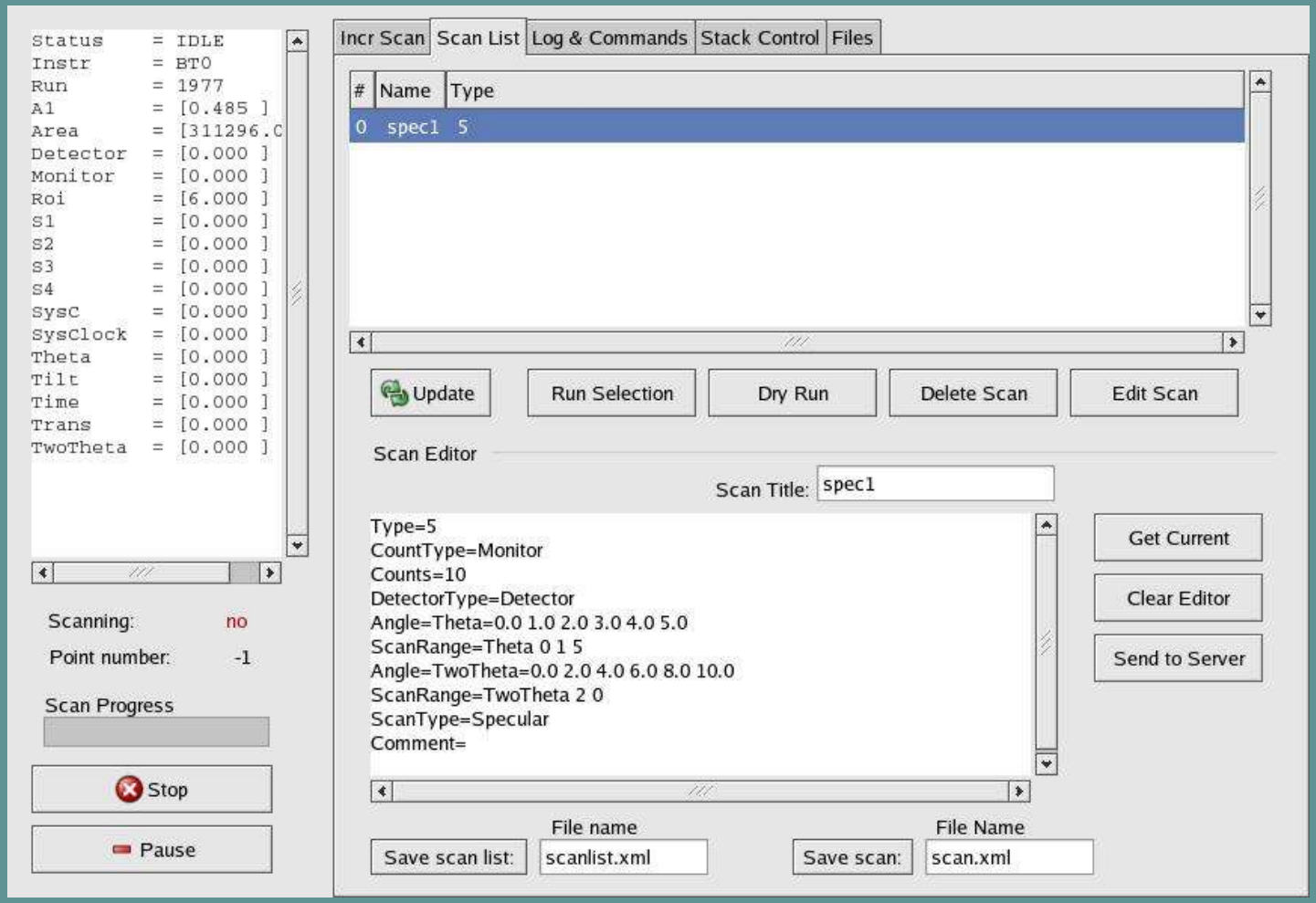
Administrator

A client registered as an administrator can execute any command as if it was immediate

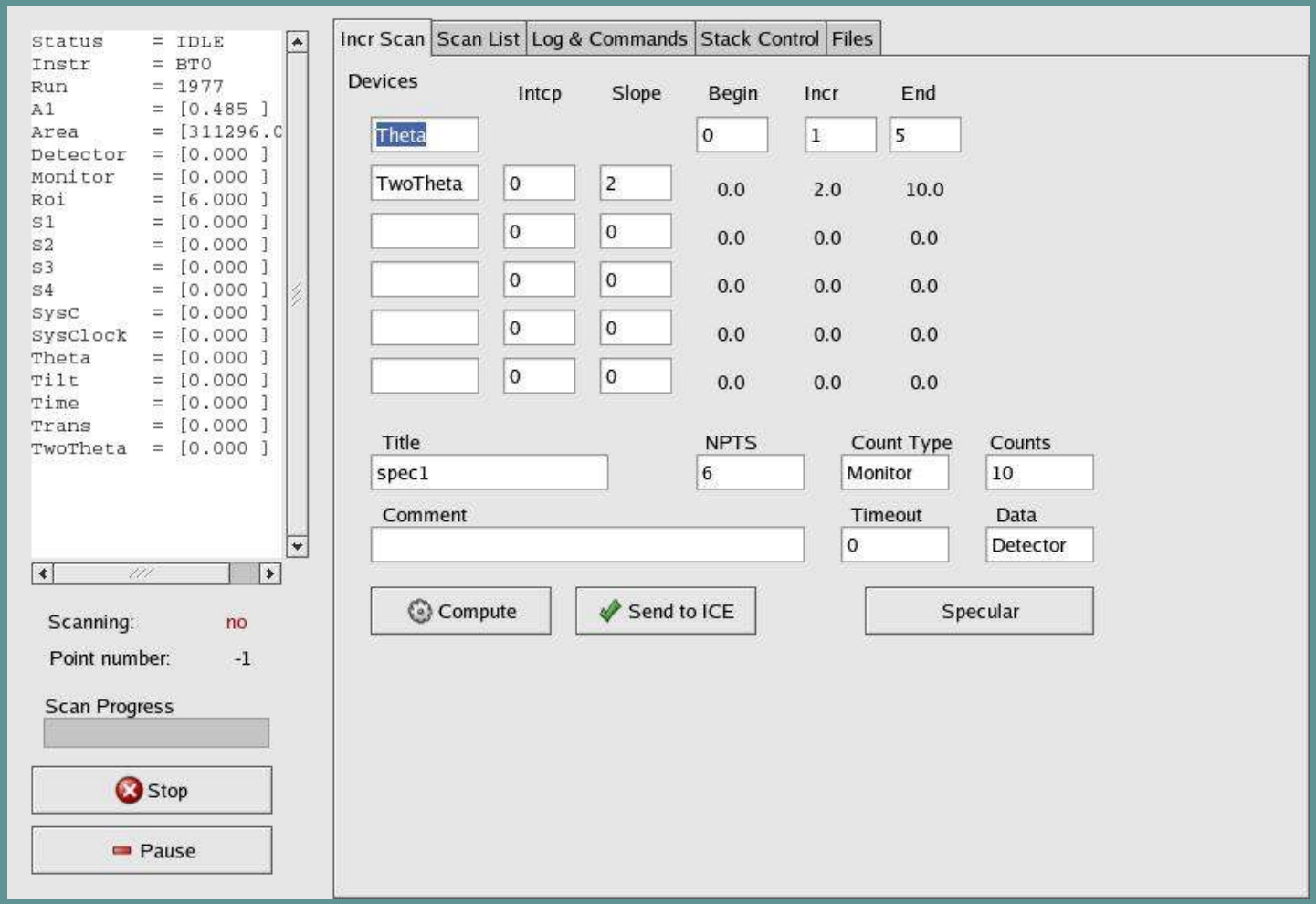
User Interface

- Any dispatcher client can be an ICE client
- Clients can attach from the instrument control computer or elsewhere
- Clients register themselves with server identifying experimenter
- Clients issue textual commands to server and can receive status/data messages from it.

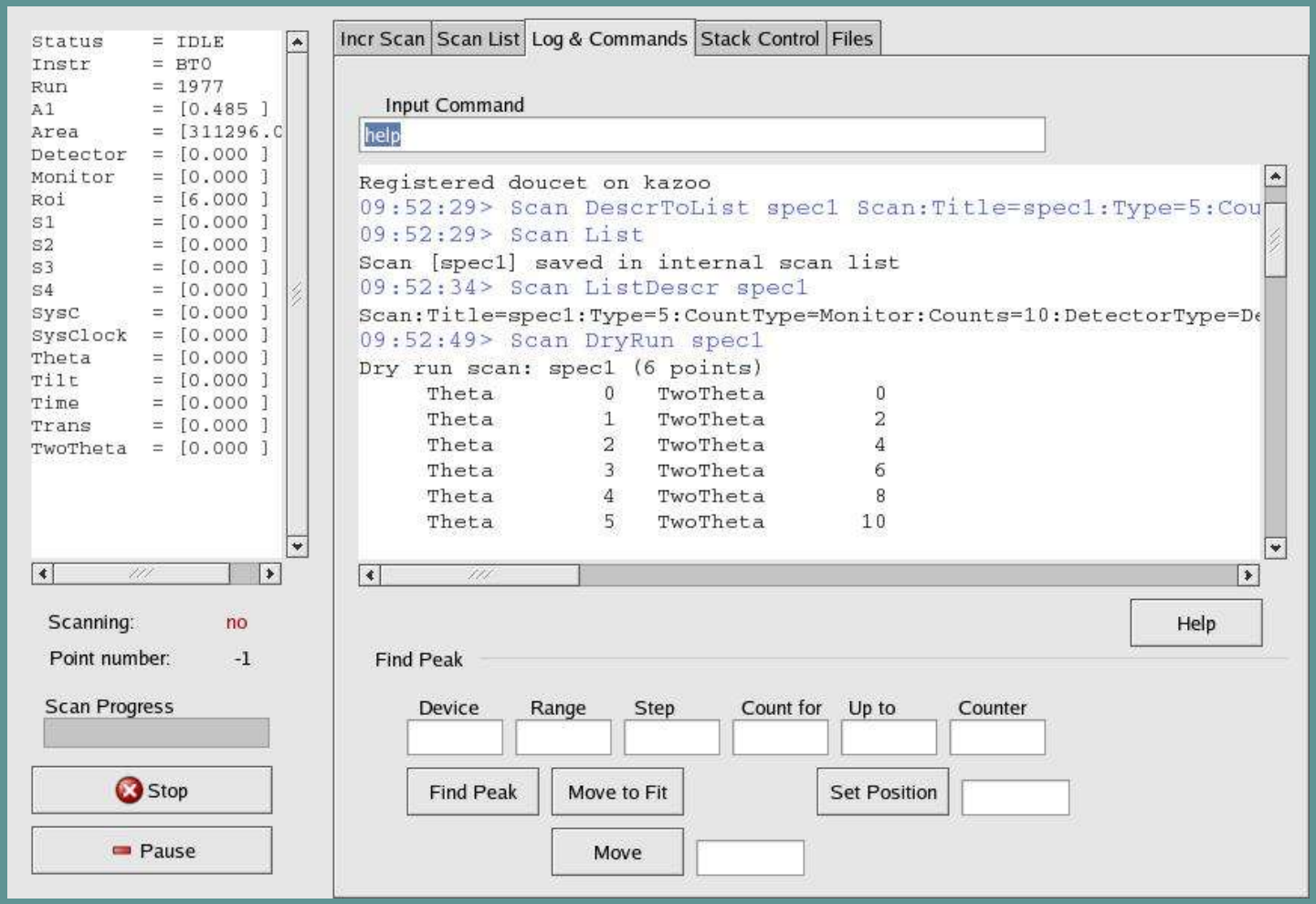
Scan Setup Window



Scan Template Window

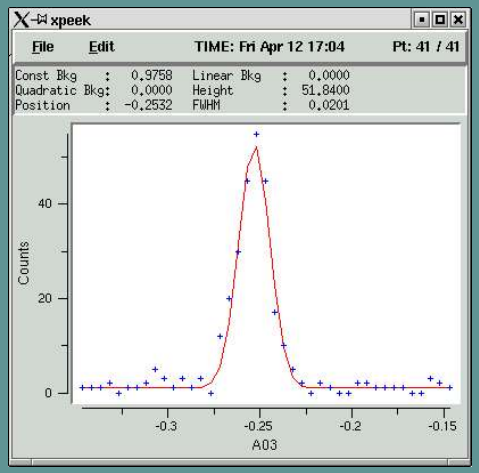


Command Entry View



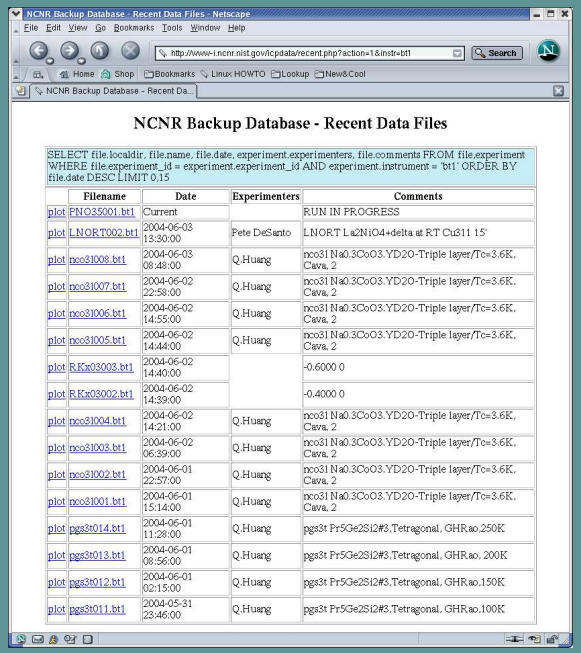
Interoperation with Existing Utilities

Live Data Visualization



Local Spectrometer Cameras

Facility Data Repository



Current Status

DALI Device Abstraction Layer

- Mature support for motors and counters
- Developmental support for process controllers
- Mature Python wrapper

ICE Instrument Control Server

- Major structural elements complete
- Scan infrastructure functional
- ICE Configured to interoperate with existing NCNR tools
- Many specialized features remain to be ported from ICP

User Interface

- Currently using provisional Python/gtk client
- Goal is to standardize interface for instruments of similar types