THE ROLE OF THE CEBAF ELEMENT DATABASE IN COMMISSIONING THE 12GEV ACCELERATOR UPGRADE

Authors

T. Larrieu, M.Keesee, M. Joyce, C. Slominski, D. Turner

Jefferson Lab

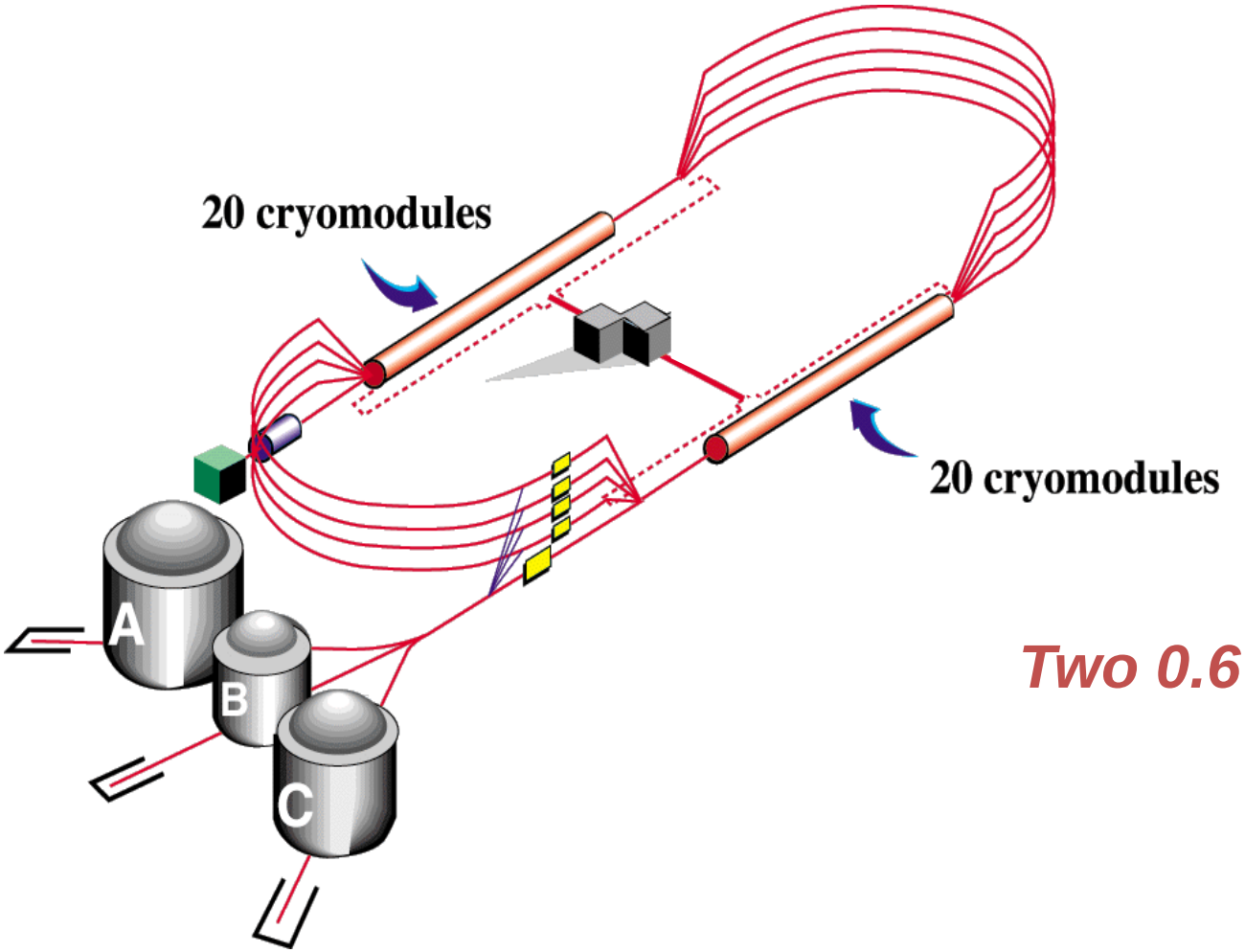
Outline

- Introduction
- Overview of the CEBAF Element Database (CED)
  - Design
  - Implementation
- The CED during commissioning
- The role of the CED during Operations

Continuous
Electron
Beam
Acceleration
Facility

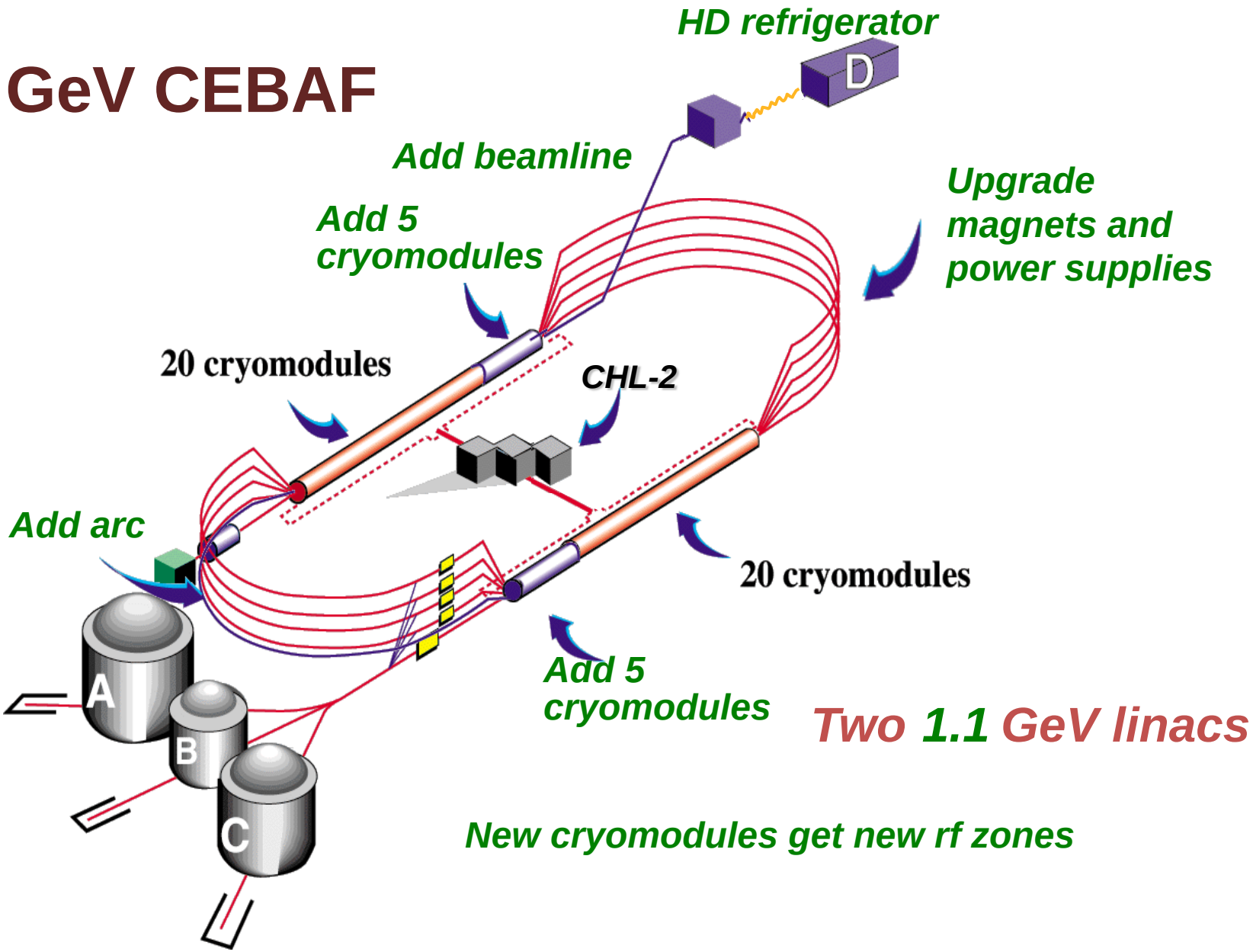
Jefferson Lab's accelerator site

6 GeV CEBAF



Two 0.6 GeV linacs

12 GeV CEBAF



Two 1.1 GeV linacs

New cryomodules get new rf zones

CED and the Upgrade

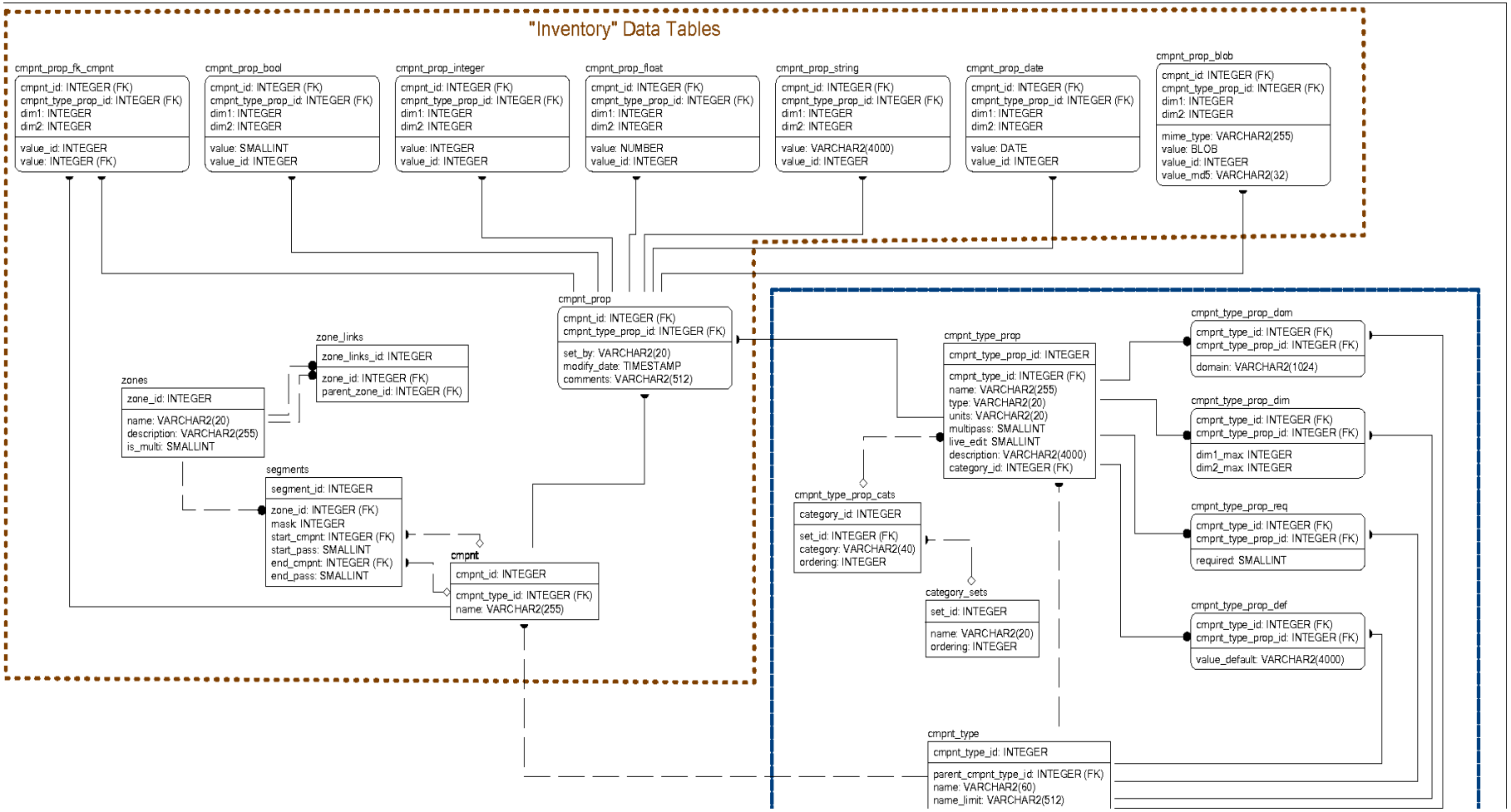
The CEBAF Element Database (CED), designed before the upgrade as a settings management system to support model-driven machine configuration was extended in scope to support the upgrade project.

- *Hot Checkout*
- *Screen Generation*
- *Commissioning*

CEBAF Element Database (CED) Scope

- Inventory and describe the elements needed to operate the CEBAF accelerator.
  - Physical Coordinates (x,y,z,s)
  - Design Parameters (twiss)
  - Control System Attributes (channel, EPICSName, etc.)
- Document Relationships among elements
  - Powered by
  - Housed by
  - Controlled by

The CED's EAV/CR Schema

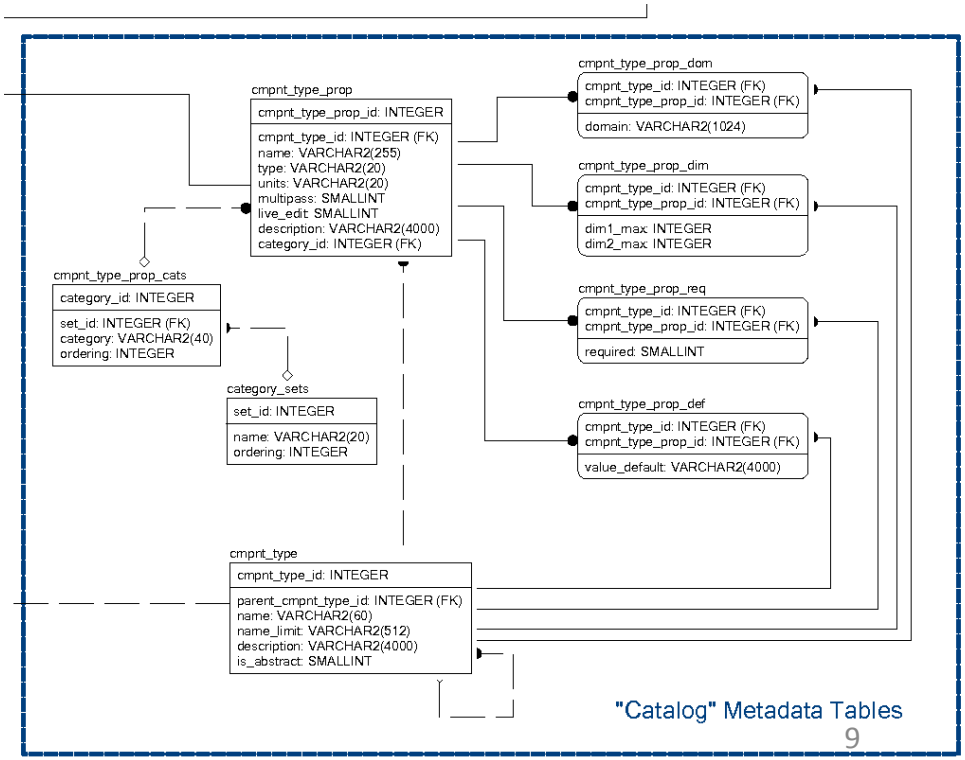


- Catalog tables defines types and their inheritance.
- Inventory tables stores element instance data.

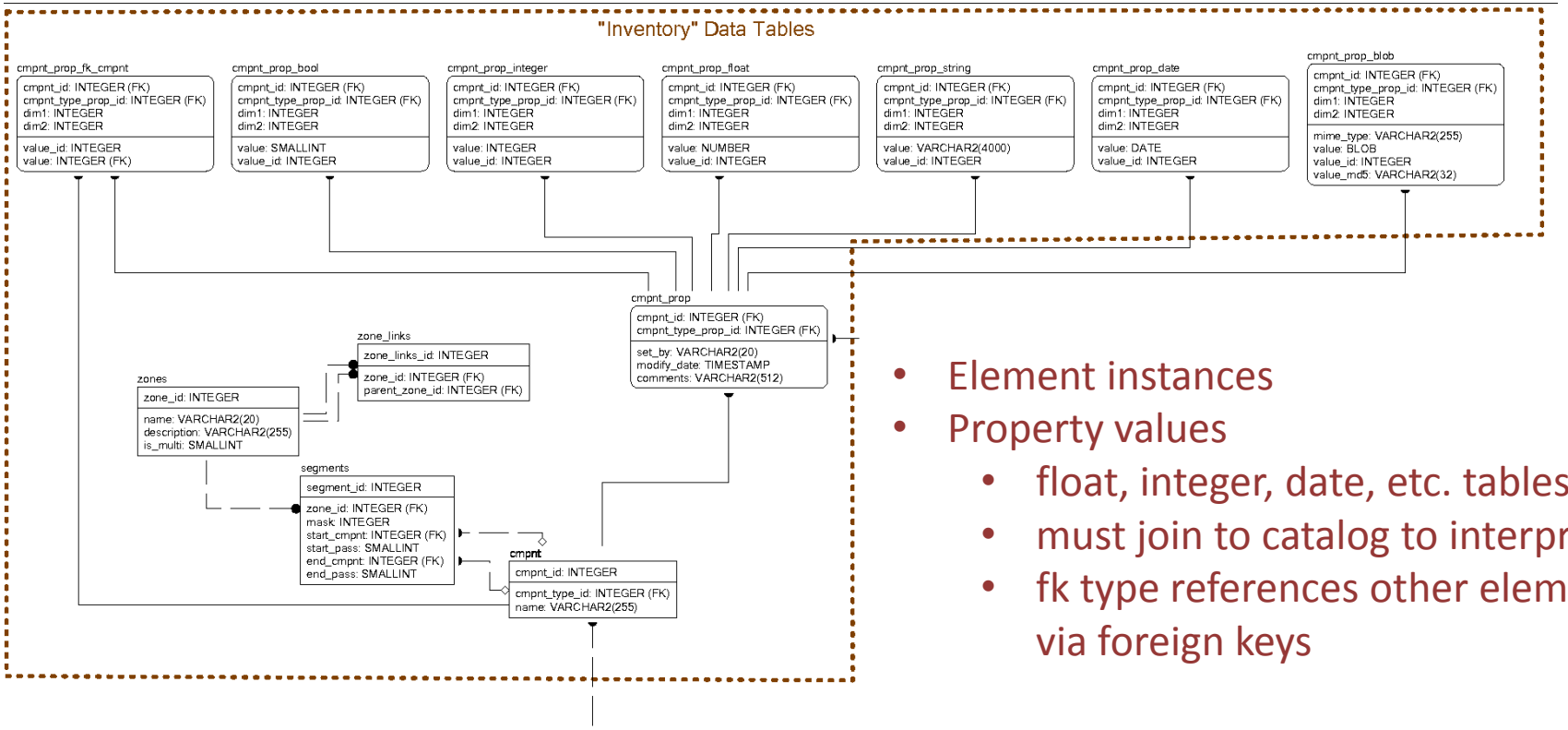
Metadata Catalog Tables Define

- Class Hierarchy
- Class Properties
- Class Property Attributes
  - Type (float, string, blob, fk, bool, date, list)
  - Required/Optional
  - Scalar or Vector Dimension
  - Default Value
  - Domain Constraints
  - Live Editable
  - Units
  - Multipass
- Subclasses inherit properties, but may override
  - Dimension
  - Required
  - Domain
  - Default

AccStyle Inventory (24 items)
Lineage: Elem > LineElem > BeamElem > Diagnostic > Harp > AccStyle



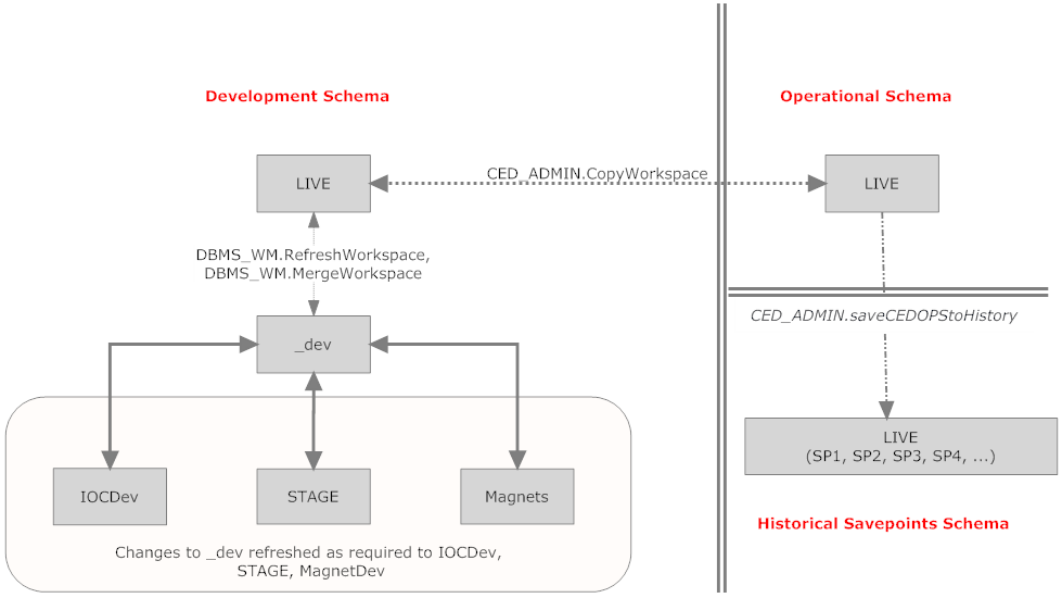
Inventory Tables Store



- Element instances
- Property values
  - float, integer, date, etc. tables
  - must join to catalog to interpret
  - fk type references other elements via foreign keys

CED Implementation

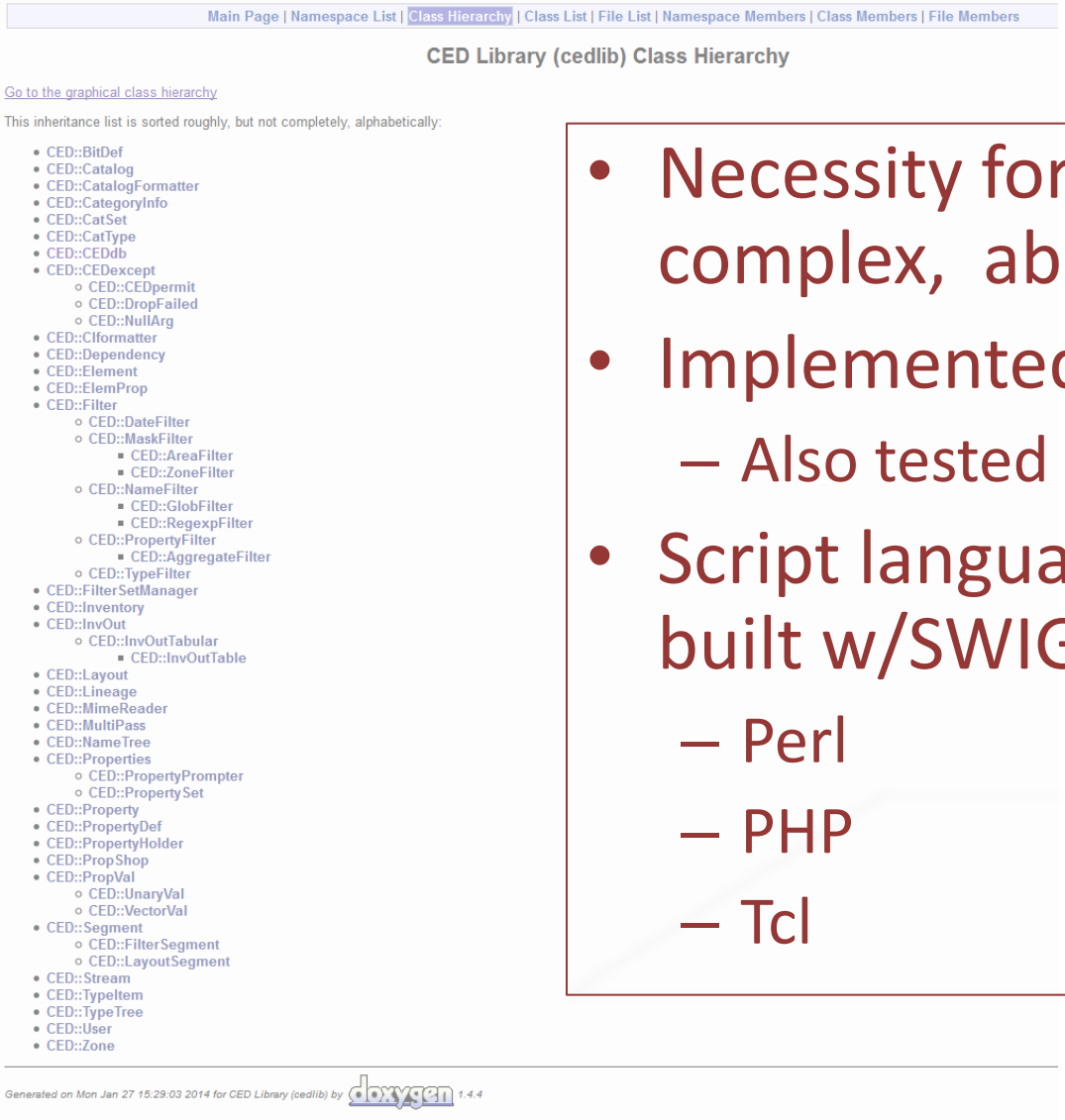
- Protected OPS Schema
  - Strict limits on “Live Edits”
- DEV Schema with Workspaces (Branches)
  - Branches where changes can be made and then merged.
  - Safely import data and test “what if” scenarios
  - Tools for merging, refreshing, and resolving conflicts.
- Read only History Schema
  - Automatic and Named Savepoints
  - Efficiently stores only row differences.



Software Tools can seamlessly utilize any schema/workspace/savepoint:

ced –wrkspc OPS –inventory –t Harps
ced –wrkspc STAGE –inventory –t Harps
ced –wrkspc AutoSP93 –inventory –t Harps

CED API



Main Page | Namespace List | Class Hierarchy | Class List | File List | Namespace Members | Class Members | File Members

CED Library (cedlib) Class Hierarchy

Go to the graphical class hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- CED::BitDef
- CED::Catalog
- CED::CatalogFormatter
- CED::CategoryInfo
- CED::CatSet
- CED::CatType
- CED::CEDdb
- CED::CEDexcept
  - CED::CEDpermit
  - CED::DropFailed
  - CED::NullArg
- CED::CIformatter
- CED::Dependency
- CED::Element
- CED::ElemProp
- CED::Filter
  - CED::DateFilter
  - CED::MaskFilter
    - CED::AreaFilter
    - CED::ZoneFilter
  - CED::NameFilter
    - CED::GlobFilter
    - CED::RegexpFilter
  - CED::PropertyFilter
    - CED::AggregateFilter
  - CED::TypeFilter
- CED::FilterSetManager
- CED::Inventory
- CED::InvOut
  - CED::InvOutTabular
    - CED::InvOutTable
- CED::Layout
- CED::Lineage
- CED::MimeReader
- CED::MultiPass
- CED::NameTree
- CED::Properties
  - CED::PropertyPrompter
  - CED::PropertySet
- CED::Property
- CED::PropertyDef
- CED::PropertyHolder
- CED::PropShop
- CED::PropVal
  - CED::UnaryVal
  - CED::VectorVal
- CED::Segment
  - CED::FilterSegment
  - CED::LayoutSegment
- CED::Stream
- CED::TypeItem
- CED::TypeTree
- CED::User
- CED::Zone

Generated on Mon Jan 27 15:29:03 2014 for CED Library (cedlib) by doxygen 1.4.4

- Necessity for dealing with complex, abstract schema.
- Implemented once in C++.
  - Also tested and debugged once!
- Script language library wrappers built w/SWIG for:
  - Perl
  - PHP
  - Tcl

CED Tools – Command Line Utility

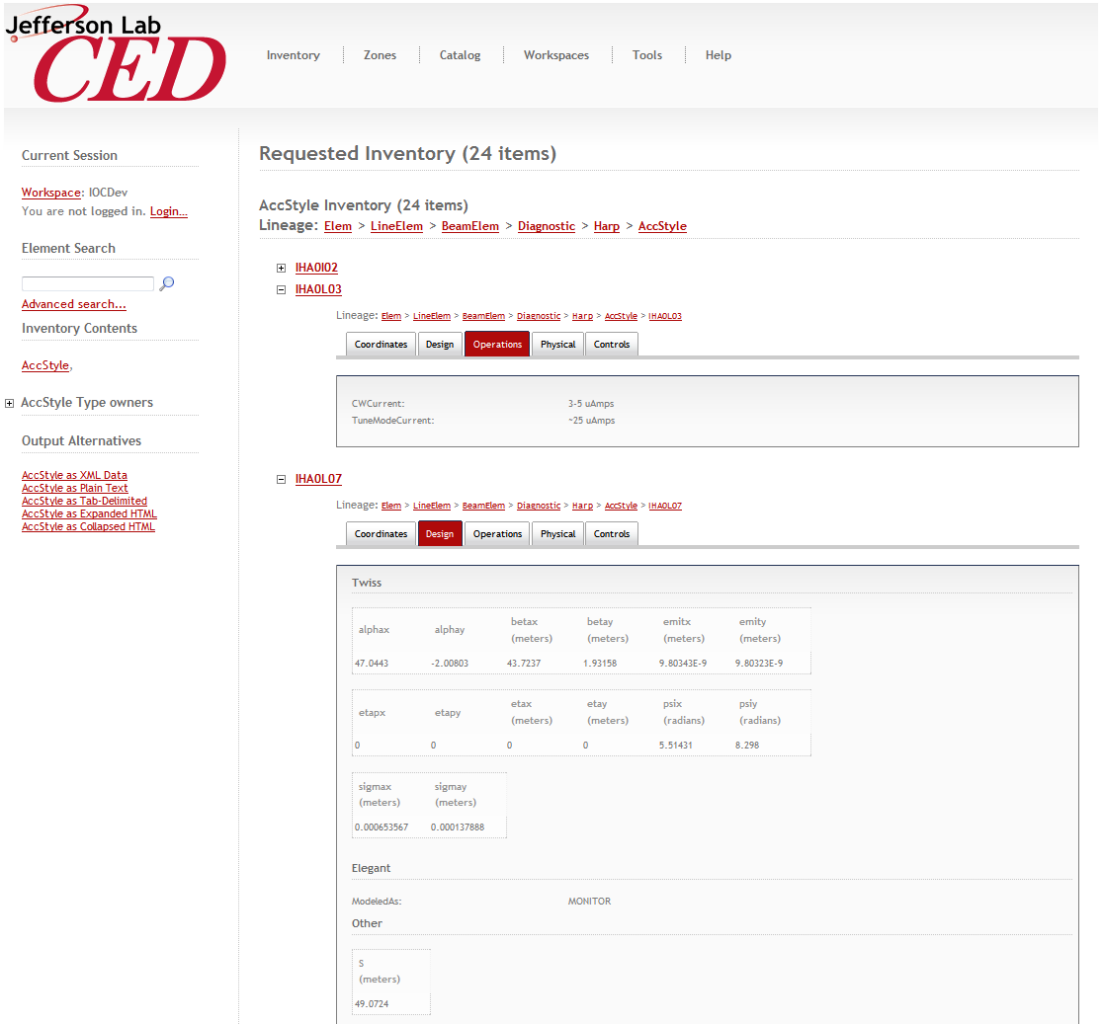
```
ced -h inventory

Inventory of elements.
Usage: -inventory [-e<opt>...<opt>] [-p[<opt>...<opt>]] [-C<opt>]
[-t<opt>...<opt>] [-z<opt>] [-Ex<opt>...<opt>] [-Ea<opt>...<opt>]
[-cs] [-nx<opt>] [-ng<opt>] [-a<opt>] [-d<opt>] [-s<opt>] [-r] [-
f<opt>] [-x<opt>] [-ced<opt>] [-wrkspc<opt>]
e; Elements of interest
p; Properties desired
C; Propeties desired by category
t; Filter by element type
z; Filter by geographical zone
Ex; Filter by property value expression
Ea; Filter by aggregate expression
cs; Case sensitive string property expression
nx; Filter by name pattern (regexp)
ng; Filter by name pattern (glob)
a; Filter by area
d; Filter by date of change
s; Sort property
r; Repeat multi-pass elements
f; Format of output; default = 0
x; Exclude from output
ced; CED login key
wrkspc; Workspace

Note: no elements specified means all (possibly filtered).
Note: filters ignored when individual elements specified.
Note: '-cs' ignored unless a string property expression is
supplied.
Note: if '-p' is provided, '-C' is ignored.
Note: '-d' takes absolute (2011-03-15) or relative (-1w; a week
ago) dates
      and a property can be specified (-5d:Length).
Note: '-f'; 0. Basic  1. Aligned table  2. Spread sheet
Note: '-x' takes a single character for each item to be excluded;
      t-element type l-labels u-units c-element count.
```

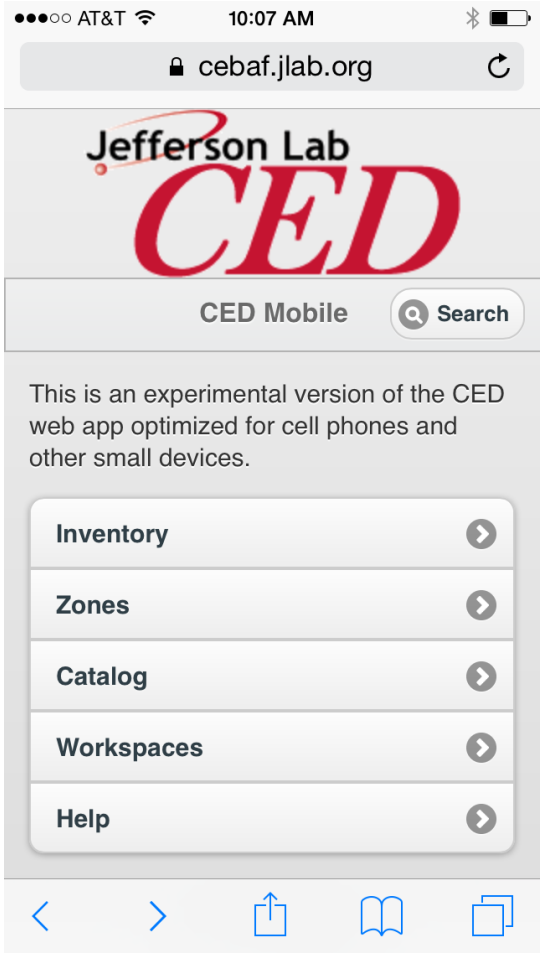
- Manage Catalog
  - Create new classes
  - Add/Edit attributes
  - Define Zones
- Inventory Elements
  - Many filters
  - Easy-to-parse output formats
- Access any workspace
  - -wrkspc switch
- Excellent for shell scripts

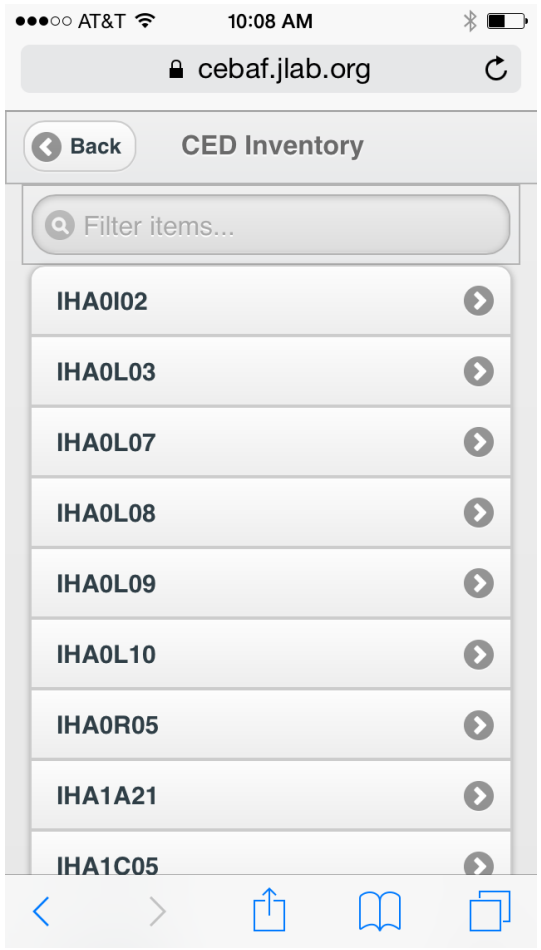
CED Tools – Web Interface

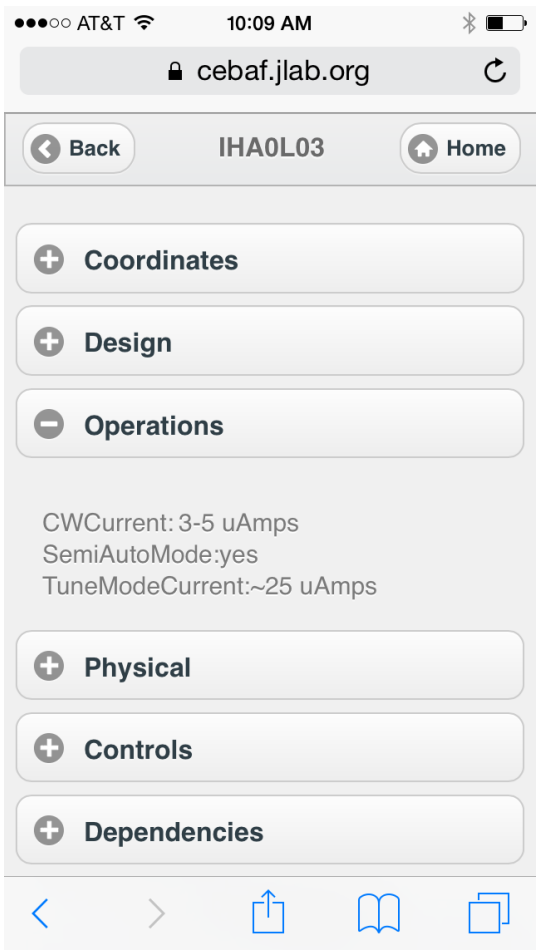


- Built atop API in PHP
  - HTML pages and forms for humans
  - REST interface (json, xml, plain text)
- Auto-builds GUI entirely from metadata catalog
  - New classes immediately accessible.

CED Mobile Web (beta)







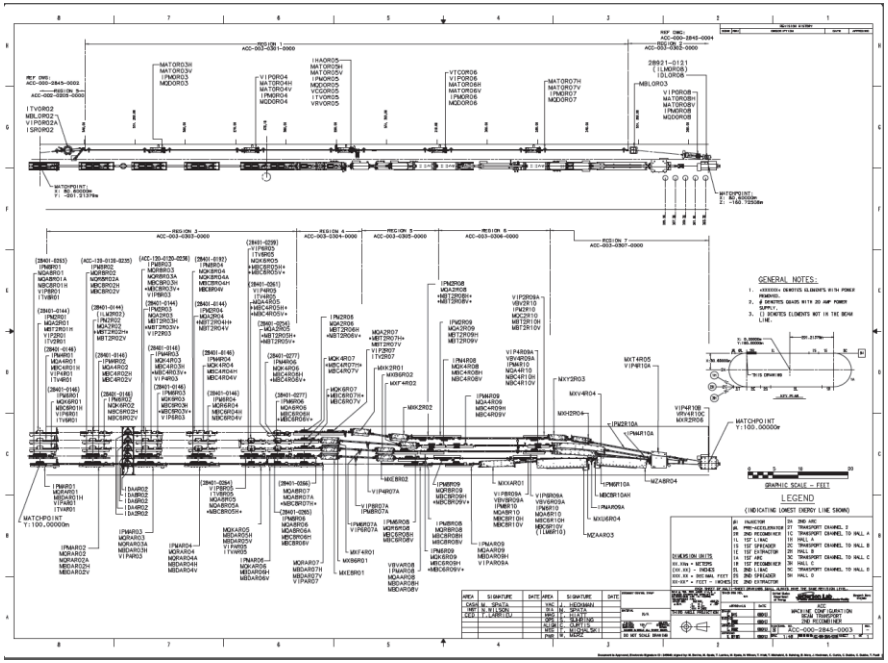
CED Event Server

- Database trigger pushes CED change notifications to event server.

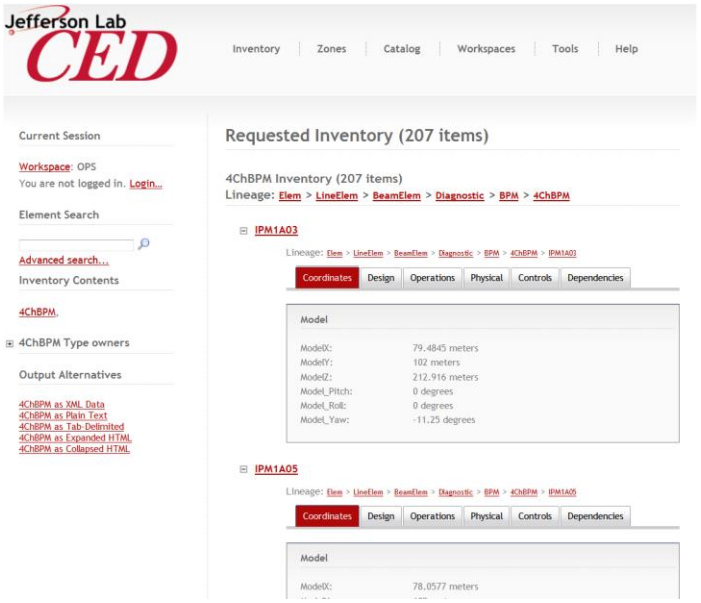
- Based on its rules, CED Event Server can:
  - Post message to OPS display wall.
  - Tell daemon processes to reload configuration.
  - Call ced2Epics to set process variable values.
  - Send email notification.
  - Make a logbook entry.

The CED During Commissioning

- Field Verification of CED, Drawings, Tunnel.
- Hot Checkout (HCO) Element Inventory.
- Back-end for QR codes on hardware labels.
- Support creation of EPICS control screens (EDM).
- Generate Alarm Handler configuration files.

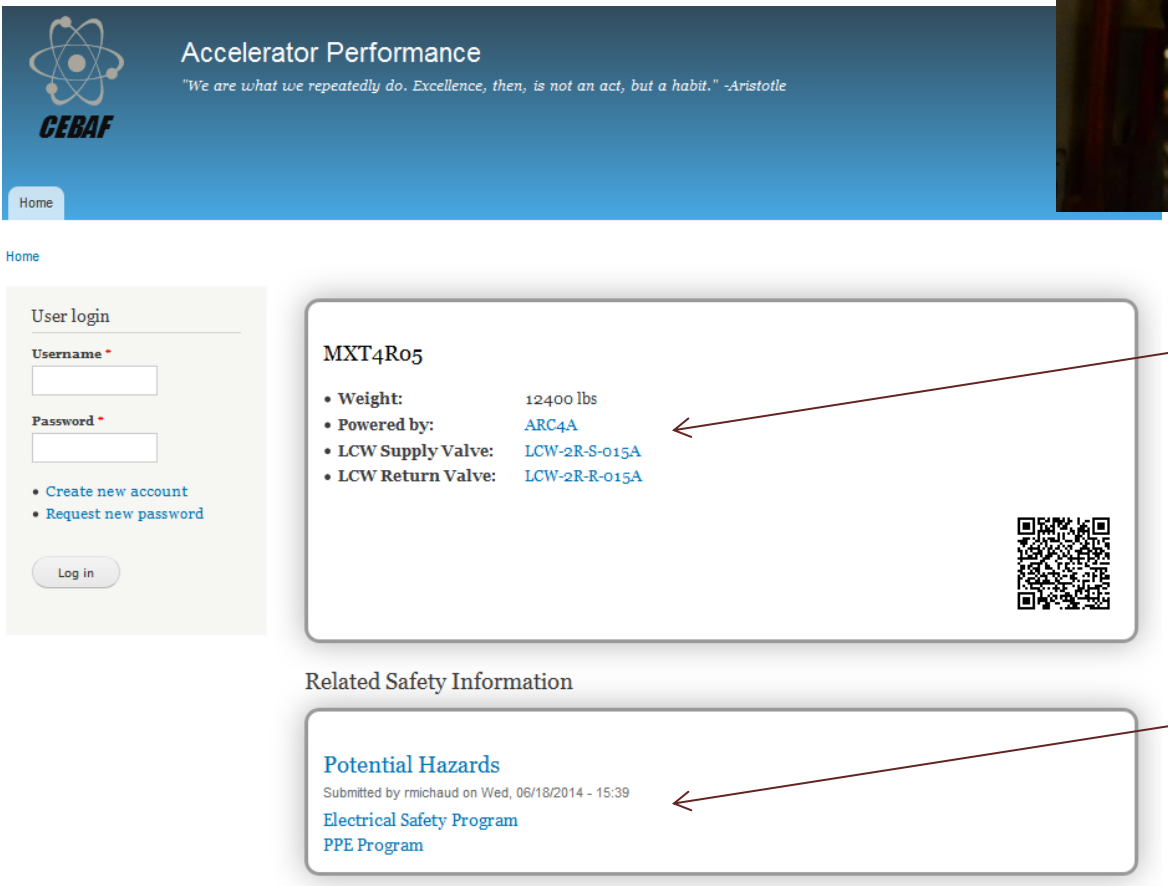


During the shutdown Region Coordinators were assigned to verify that the drawings, database, and tunnel hardware all match.



QR Code labels on tunnel hardware

The QR Code can be scanned in the field with camera-equipped tablet

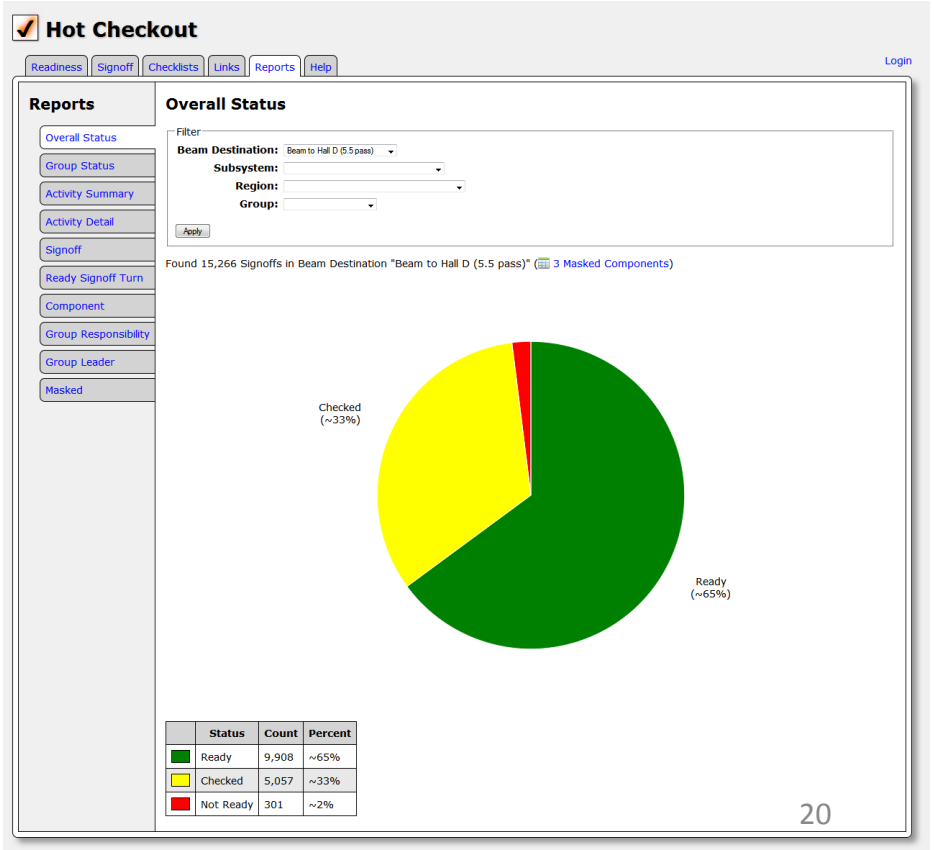


Links on the web page take the user to CED to find out what other elements share the same valves or power supply.

The Safety Group can provide additional information related to the element.

CED inventory as basis of Hot Checkout.

- 8,252 HCO components from CED
- 245 checklists
- >18,000 readiness sign-offs



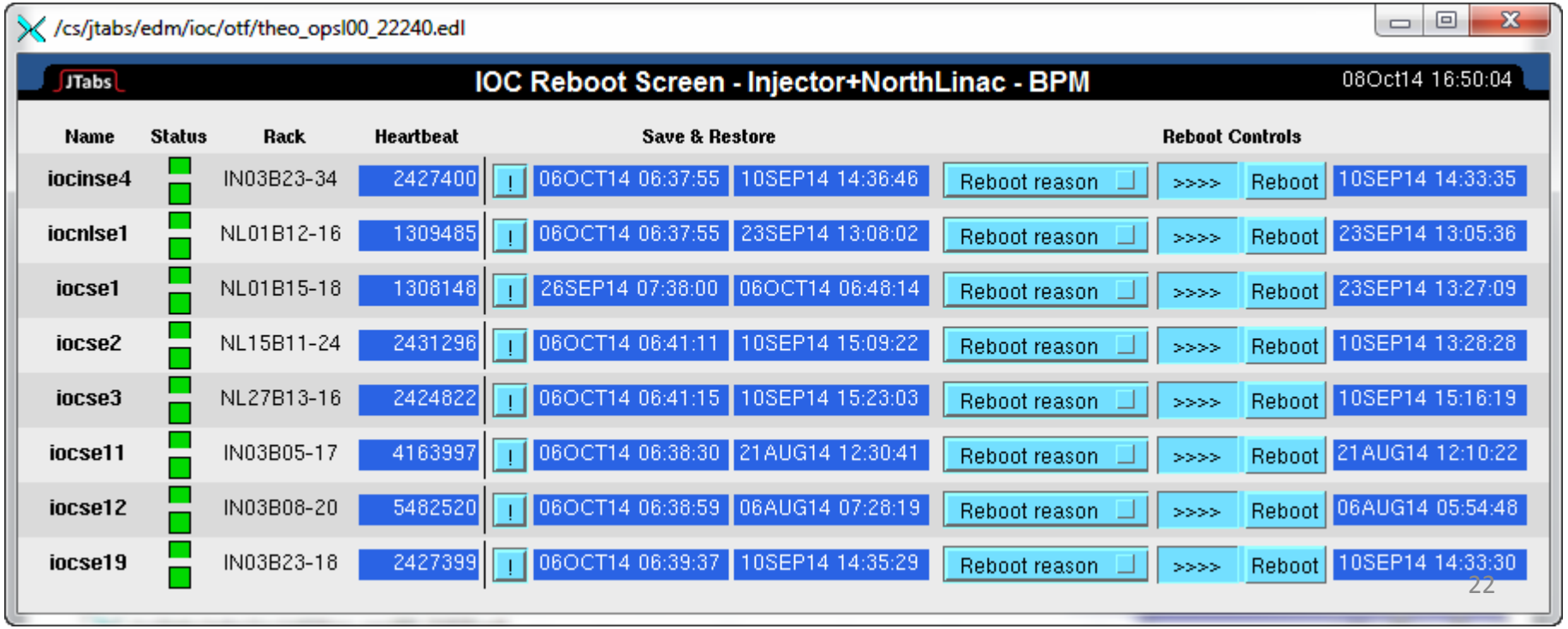
Screen Generation

- Impractical to update the ~6000 pre-upgrade .edl files by hand.
- Solution was to generate screen files programmatically using information in CED.
  - Created edllib to give programmers a qt-style library of widget and layout managers.
  - Built OTFLauncher to build screens flexibly based on CED queries

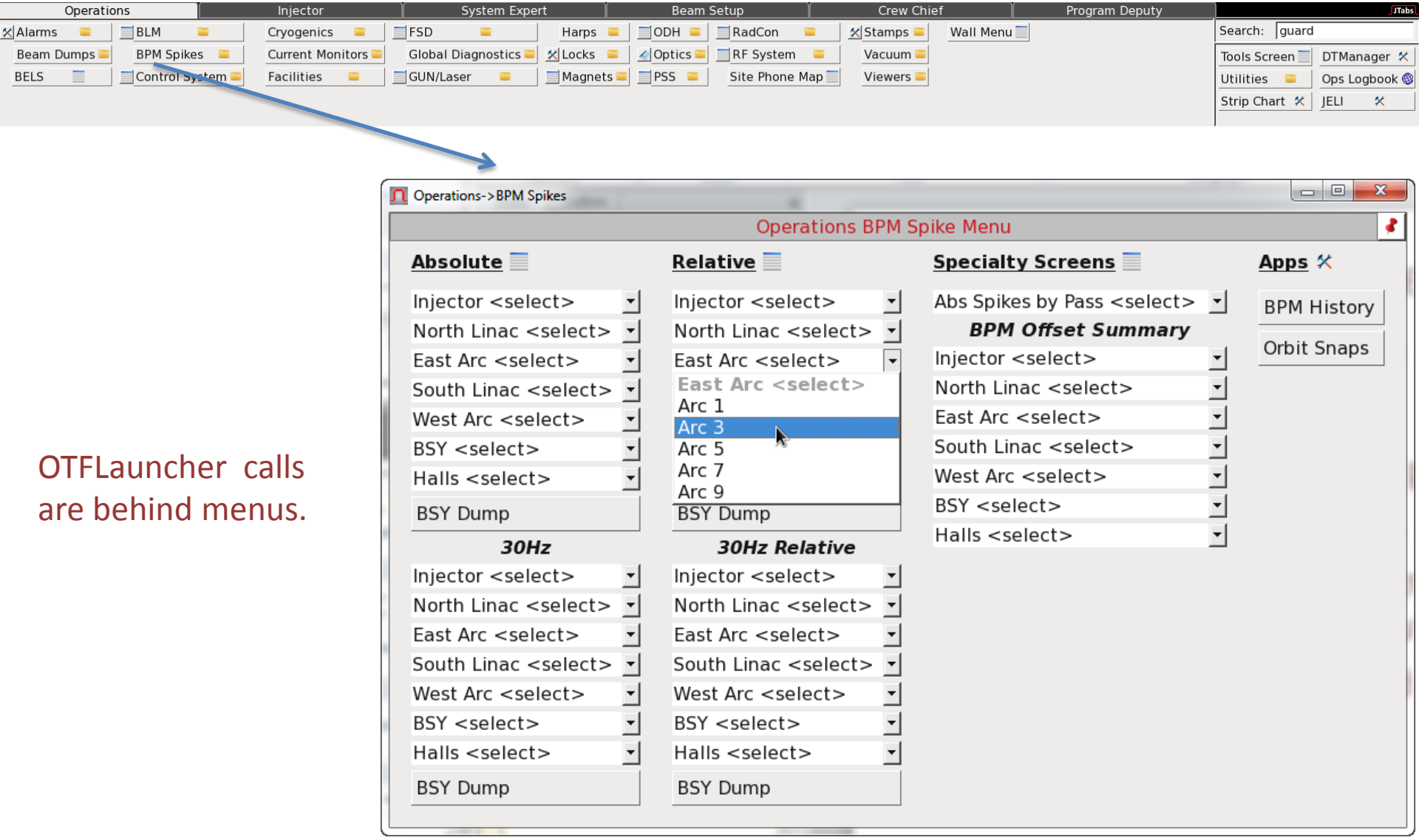
OTFLauncher

- **On-The-Fly Screen Launcher**

```
OTFLauncher ioc iocreboot -system BPM -area
A_Injector+A_NorthLinac
```



“Jtabs” Menus

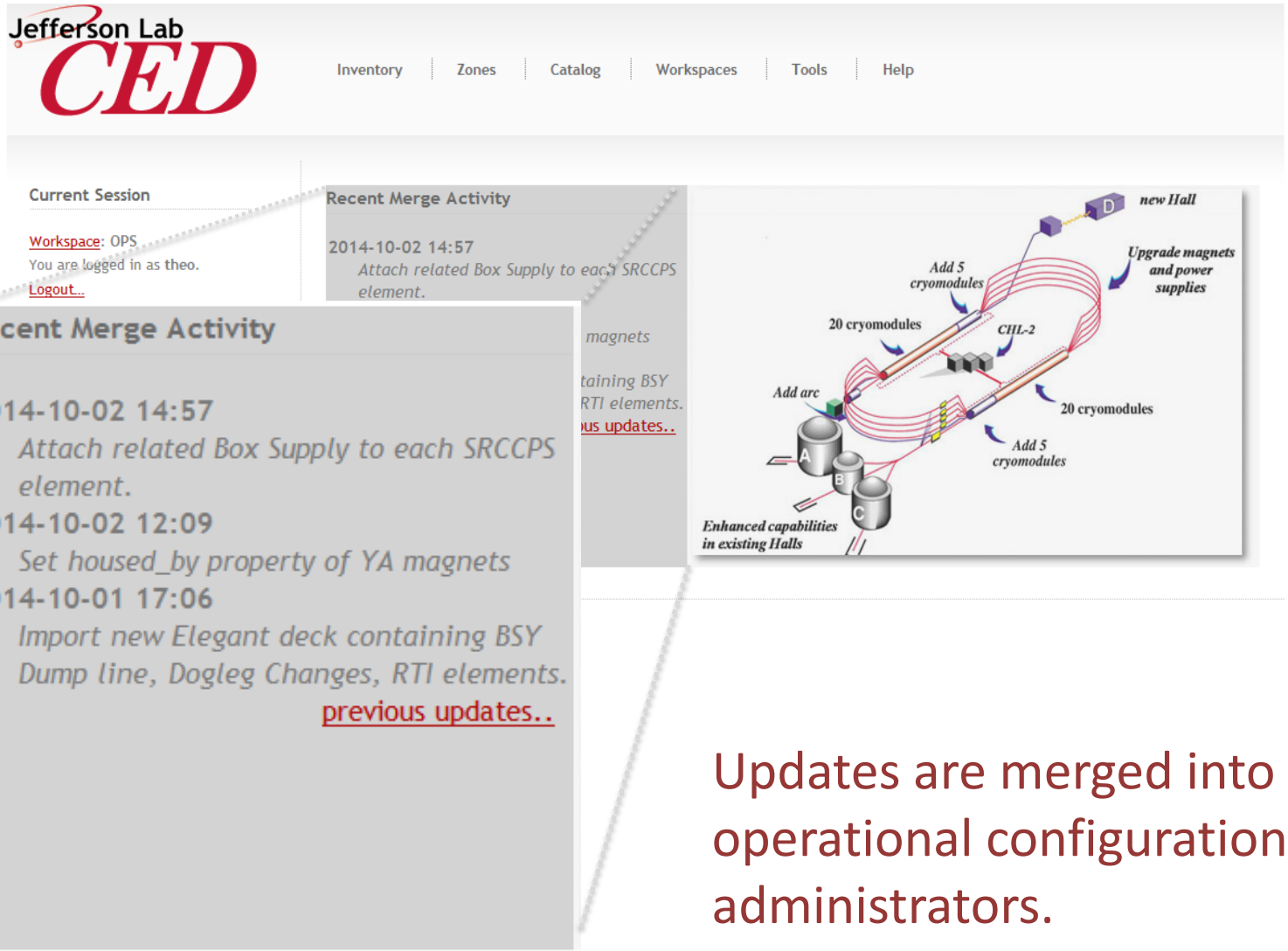


OTFLauncher calls are behind menus.

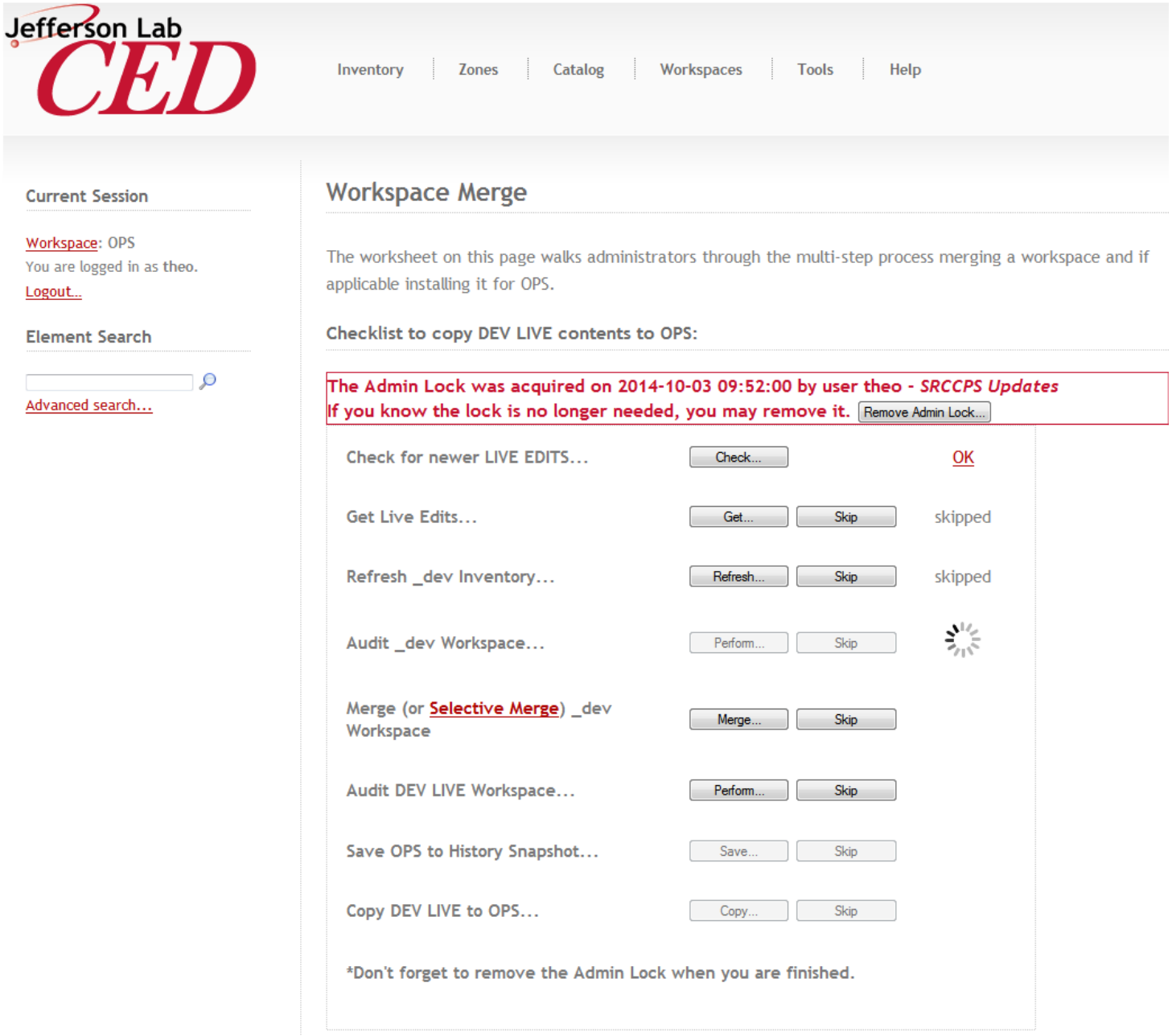
The CED During Operations

- Provide Configuration Control.
- Generate Accelerator models (elegant lattice files) for setup.
- Keep EPICS control screens (EDM) current
- Provide configuration information for High Level Applications (energy locks, orbit locks, Courant-Snyder tuning)

Configuration Control



Updates are merged into the operational configuration by CED administrators.

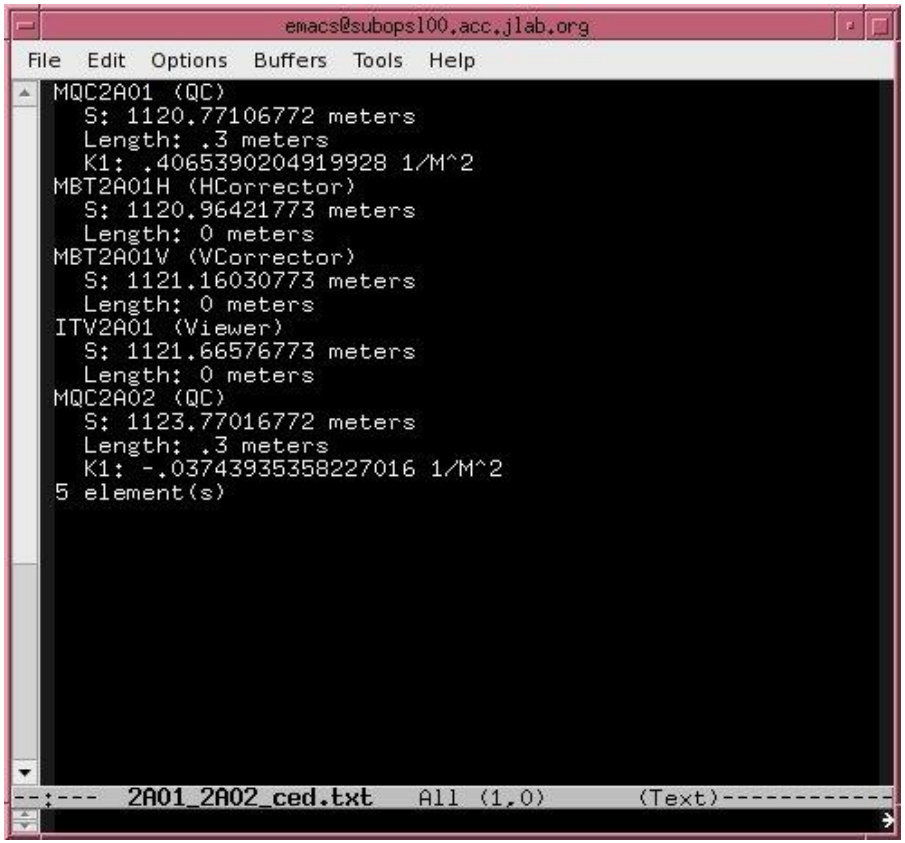


Web-based merge tool for administrators

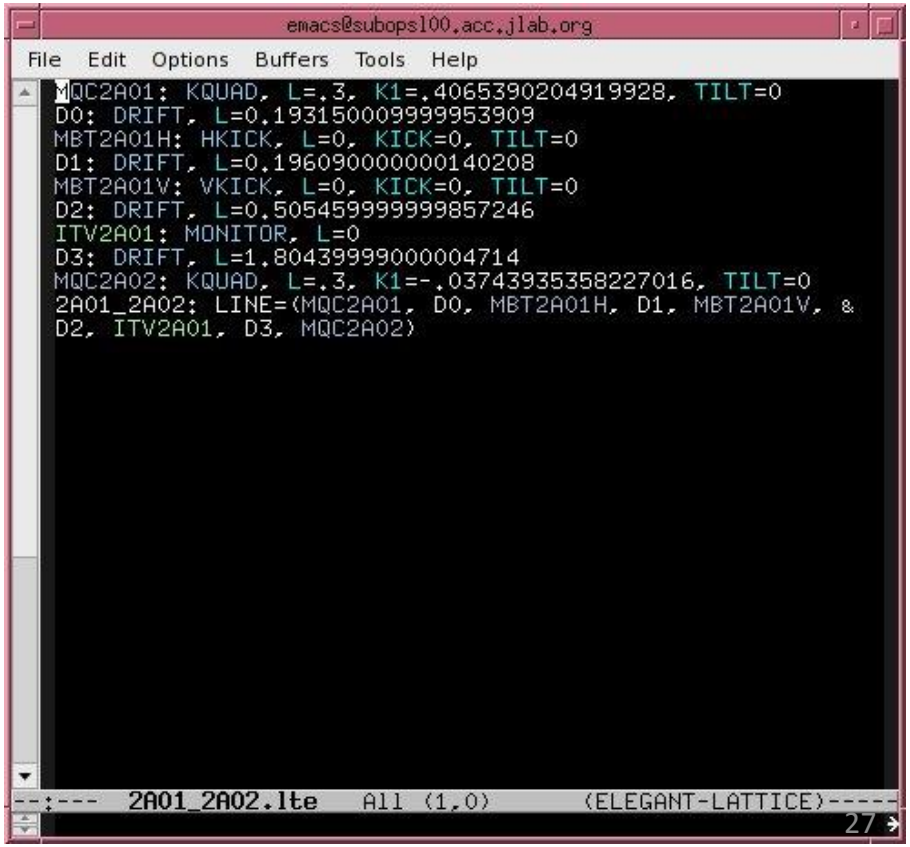
Using CED, ced2elegant, and eDT

- The ced2elegant tool generates an ELEGANT lattice from elements in the CED. The user specifies a start and end element and the pass for each. The tool retrieves all elements from CED that exist between the two given elements, inclusive and builds an ELEGANT lattice file.

CED Query

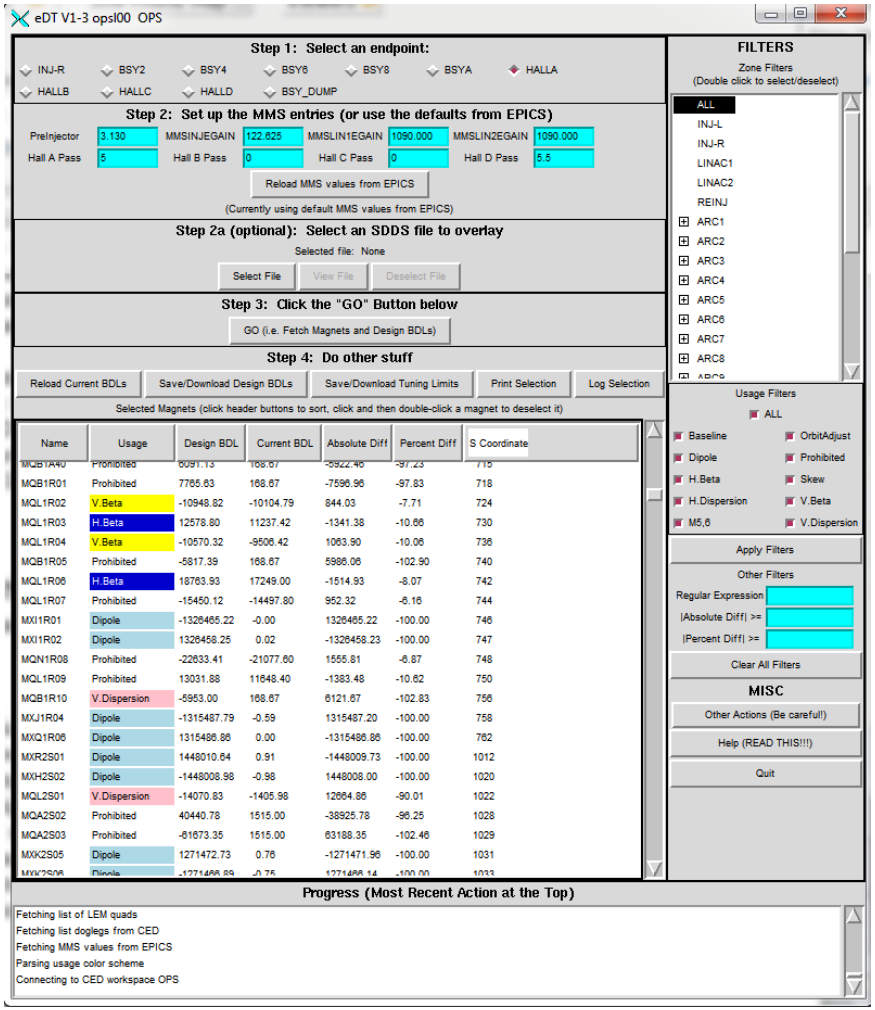


ELEGANT Lattice



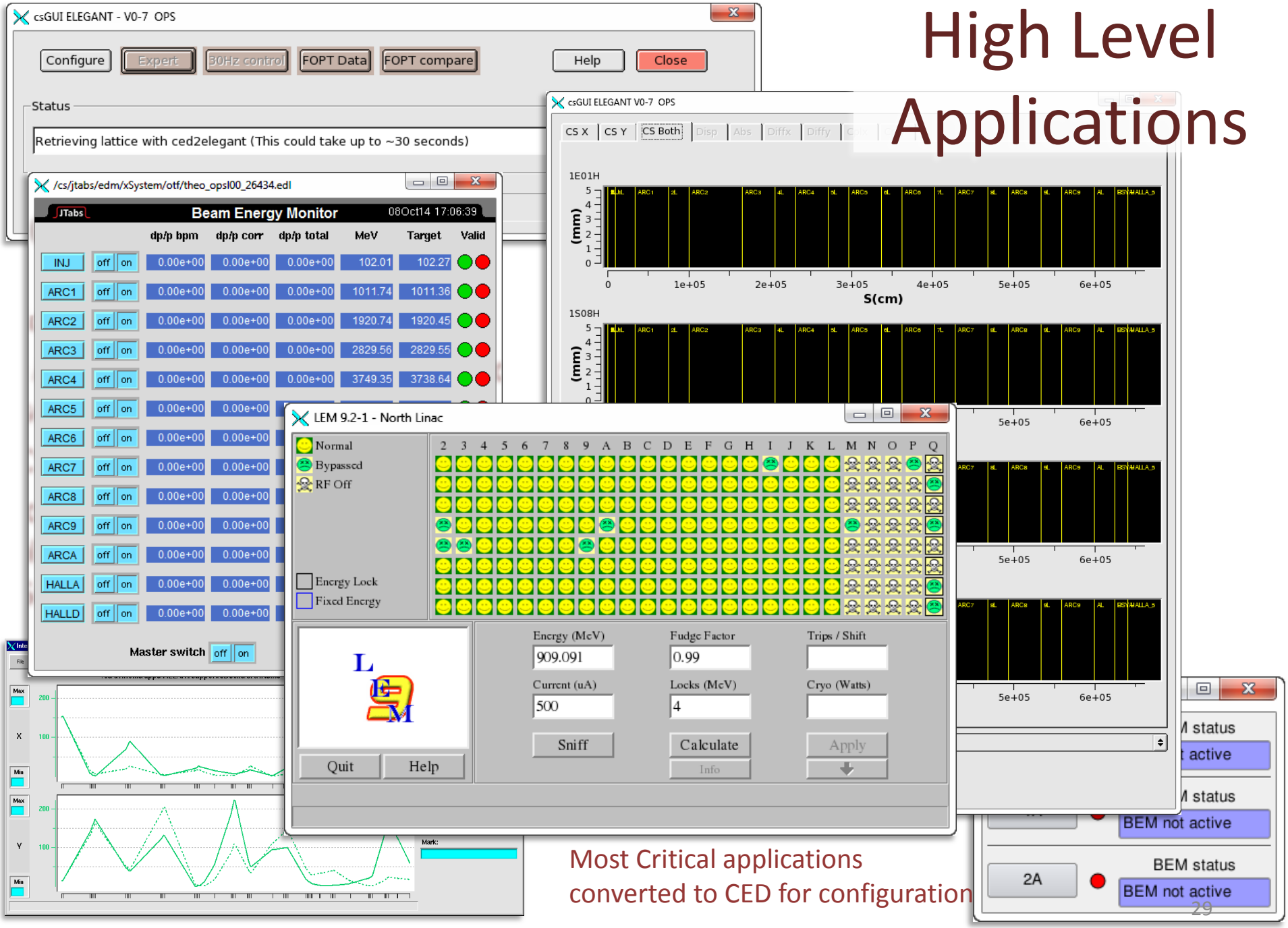
Using CED, ced2elegant, and eDT

- Accepts lattice from ced2elegant as input.
- Calls LEM to obtain Linac gradient distribution and calculated Linac quad settings and applies them to the lattice.
- Uses ELEGANT to reoptimize TWISS parameters.
- Outputs a snap file that can be loaded onto the control system using standard tools such as burt.



eDT GUI

High Level Applications



Most Critical applications converted to CED for configuration

Conclusion

- The Flexible EAV/CR design allows the CED content to evolve without changes to the database schema or API.
- Availability of an accurate database led to thorough hot checkout and meeting commissioning goals safely and on-time.
- The centralization and standardization on the CED yields new capabilities and efficiencies in screen and application development.

CED Hardware

- Oracle 11gR2 (Standard Edition)
  - Dedicated Instance
  - Oracle Workspace Manager
- Redhat Enterprise Linux 6

EAV/CR Trade-offs

- SQL Queries w/o API are complex
- Relies on API to enforce Domain constraints
- Performance challenges
  - Many tables must be joined for most queries