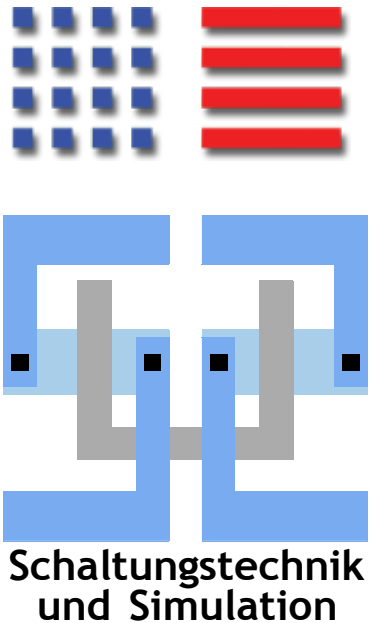




# VXD Slow-Control



Michael Ritzert

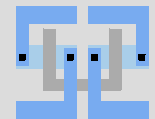
[michael.ritzert@ziti.uni-heidelberg.de](mailto:michael.ritzert@ziti.uni-heidelberg.de)

7<sup>th</sup> Belle II VXD Workshop

Prague

21.01.2015

- PXD Slow-Control coordinated by [Michael Ritzert](#).
- SVD Slow-Control coordinated by [Szymon Bacher](#).
  
- Shared systems
  - IBelle CO<sub>2</sub> cooling plant
  - FOS environmental sensors
  
- Coordinated efforts in all areas of the slow-control.
  - EPICS, CSS
  - Databases
  - PV naming / GUI conventions
  - Alarm & interlock systems



- DESY testbeam
  - Big success with all deployed SC systems.
- UNICOS
  - Ready to work with the actual IBelle UNICOS project.
- EPICS and CSS binary repositories
- SC unification
  - “The common GUI for Belle II is CSS.”
- Interface to SuperKEKB EPICS established.
- Productive use of the LMU PS
- Interlock
  - Prevents dangerous settings from reaching the PS.
- PXD power-up sequence
  - Replaces tedious manual operation.

# Configuration Database

Configuration visible as PVs to display and modify database entries.

The screenshot shows a configuration database interface. On the left, a tree view displays the hierarchy: Config-DB > configurati... > PowerSu... > unit0... > DC... > dcd-amplow. The right pane shows a table of values:

Name	Type	Value	Un
stage1	double	400.0	m/
stage2	double	600.0	m/
<b>voltage</b>	double	350.0	m/

Below the table, a control panel shows various parameters with sliders and labels: THERMAL, UPS, Disconnected, DB Voltage, DB Current S1/S2, and various component labels like sw-sub, sw-dvdd, sw-refin, dcd-amplow, dcd-avdd, dcd-refin, dhp-core, and dhp-io.

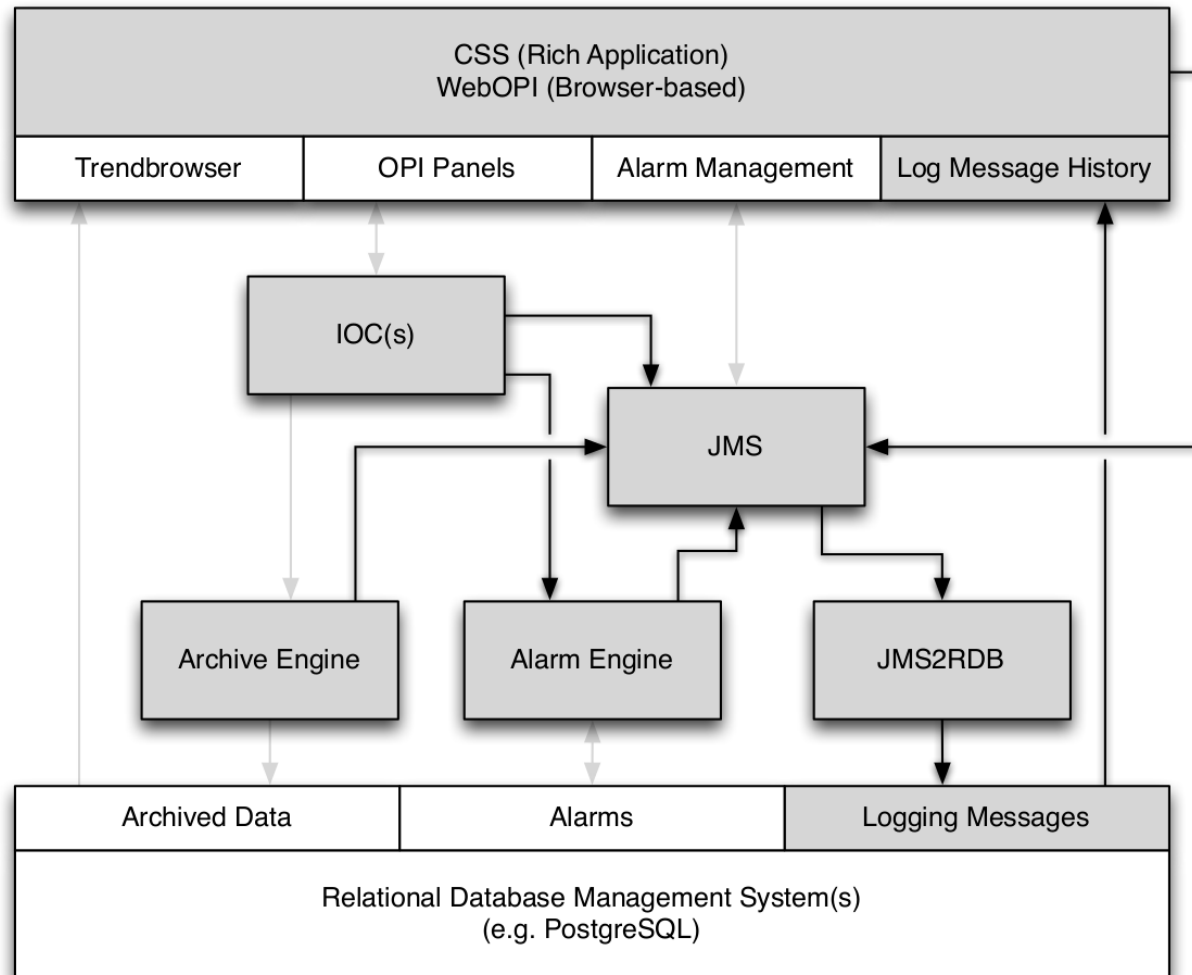
Implementation note: Use “Linking Containers” with macros for repetitive GUI items.

The screenshot shows a graphical user interface with two tabs: device\_config.opi and config-line.opi. Below the tabs is a horizontal slider control with a scale from 0 to 800. The slider is currently positioned at approximately 350. There are also some small icons and a double-headed arrow next to the slider.

1. The run type is set in the Master RC.
2. We look up the configuration id to load for this run type.
  - Previously configured as run type  $\Rightarrow$  configuration id pairs.
3. The id is stored in the conditions database.
4. The configuration is loaded and used by the power-up sequence.
5. Any deviation from the configured values is raised as an alarm.

# Message Log

- C++ library that connects to the ActiveMQ service in the same way as CSS applications.
  - Can be used from IOCs.
    - ⇒ All log messages in one common database accessible from CSS.

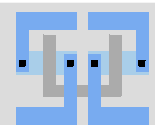


# Message Log II

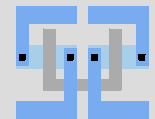
- Showing mixed C++ and Java log messages from the database:

TIME	TEXT	NAME	SEVERITY	CREATETIME	USER	APPLICATION-ID	CLASS
2015/01/05	Internal browser is not e	logging	WARNING	2015-01-05 20:06:39.04	ritzert	CSS	org.csstudio.logging.PluginLogListener
2015/01/05	WARNING: Prevented re	logging	SEVERE	2015-01-05 16:04:41.88	ritzert	CSS	org.csstudio.logging.PluginLogListener
2015/01/05	Operation details	logging	SEVERE	2015-01-05 15:53:53.50	ritzert	CSS	org.csstudio.logging.PluginLogListener
2015/01/05	Operation details	logging	SEVERE	2015-01-05 15:53:53.50	ritzert	CSS	org.csstudio.logging.PluginLogListener
2015/01/05	SIGNAL 11 received.	static void SuS::lo	FATAL	2015-01-05 15:50:59.37	fecmess	app	logger
2015/01/05	final	int main()	WARNING	2015-01-05 15:50:59.37	fecmess	app	example
2015/01/05	info: 9	int main()	INFO	2015-01-05 15:50:59.37	fecmess	app	example
2015/01/05	9	int main()	DEBUG	2015-01-05 15:50:59.37	fecmess	app	example

- Features of the C++ logging classes:
  - Output to stdout, log files (automatically rotated), STOMP (JMS protocol)
  - Multi-threaded: application thread continues while the messages are sent.
  - Automatic retry when log server is unavailable.
  - runtime configurable from EPICS IOC shell
  - Logs backtrace on crashes.



- Reminder: We need to implement the supervision layer (operator interface). The actual plant control logic needs not be changed.
- UNICOS can now create output for EPICS
  - EPICS database for the modbus IOC. (4298 PVs for IBBelle Plant A.)
  - Template OPI with all devices.
- Implementation of the “faceplates” in CSS has progressed well.
- Requires rigorous testing and comparisons with WinCC OA (“PVSS”) behavior!
  - The UNICOS documentation is incomplete in parts.
- Ready to run with the actual hardware!
- Still to do:
  - generate Archiver configuration.
  - generate device tree views
  - same approach as for EPICS database and OPI template output can be used.

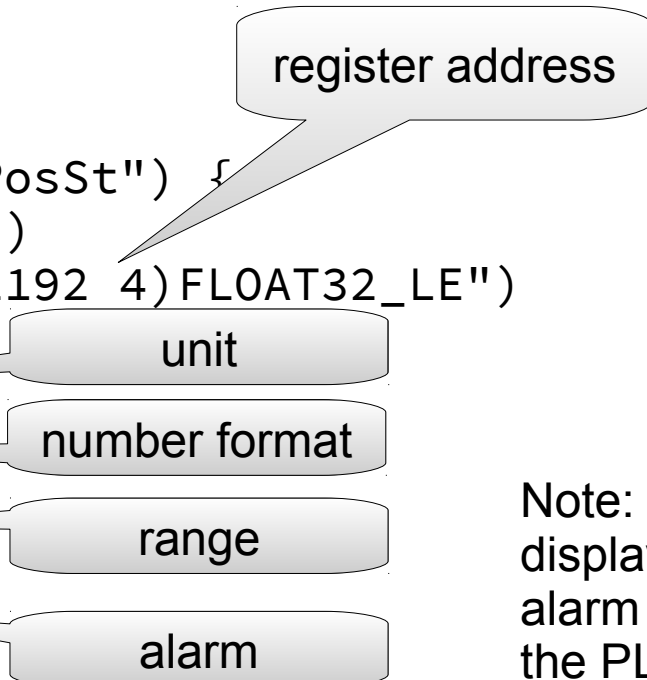




- Python script started (one time) by the UNICOS converter GUI.
- Outputs an EPICS database with hardware addresses to be used by the modbus IOC.
- Tries to convert all metadata
  - Alarm levels
  - Units
  - Number formats

- **Sample output:**

```
record(ai, "CSMB_WP1_PT142:PosSt") {  
  field(DTYP, "asynFloat64")  
  field(INP, "@asyn(block1192 4)FLOAT32_LE")  
  field(SCAN, "I/O Intr")  
  field(EGU, "bar")  
  field(PREC, "1")  
  field(LOPR, "0.0")  
  field(HOPR, "110.0")  
  field(HIGH, "55.0")  
}
```



Note: The alarm limit is for display purposes only. The alarm is generated within the PLC.

Generated OPI template: All widgets in one OPI to be copied.

The screenshot displays a grid of process control widgets in a dark-themed interface. Each widget is represented by a symbol and a label. Some widgets show numerical values in green, while others show 'no value'. An 'Outline' window is open in the bottom right corner, listing the widgets in a tree structure.

Widget Label	Value / Status
EV208	10.0
FIAA_FSDIC	10.0
FIAA_WDIC	10.0
FL104_dp	10.0
FL144_dp	10.0
FT106_calc	10.0
FT106_dens	10.0
FT106_temp	10.0
Heaters_OK	10.0
HL_OFF_det	10.0
HL_ON_det	10.0
HX206	10.0
HX207	10.0
JBoxA_FSDIC	10.0
JBoxA_WDIC	10.0
B_24V_CB_C	10.0
LP101	10.0
LT142_Mass	10.0
LT142_RL	10.0
LT142	10.0
MV041_GH	10.0
MV041_GL	10.0
MV041	10.0, no value
MV050_GH	10.0
MV058_GL	10.0
MV058	10.0, no value
MV106_GH	10.0
MV106_GL	10.0
MV106	10.0, no value
MV110_GH	10.0
MV110_GL	10.0
PresA_WDIC	10.0
PS501	10.0
PT101	10.0
PT102	10.0
PT103	10.0
PT104_Tsat	10.0
PT104	10.0
PT224	10.0
PT234_Tsat	10.0
PT234	10.0
PT244_Tsat	10.0
PT244	10.0
PT248_Tsat	10.0
PT248	10.0

**Outline Window:**

- Local
  - LocSpare08
    - DI** LP101\_CB\_OK
    - DI** LP101\_DI
    - OnOff
      - LP101
    - AI** 123.4 LT142\_Mass

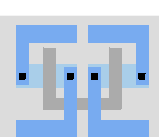
# UNICOS Faceplates

- Dialogs for seeing details about and manipulating the devices.
- Linked from each device widget.

Status	Operation Modes	Alarms / Limits	
Position <input type="text" value="#####"/>	Auto <input type="checkbox"/> Inh <input type="checkbox"/> On <input checked="" type="checkbox"/>	HH <input type="text" value="#####"/> H <input type="text" value="#####"/> L <input type="text" value="#####"/> LL <input type="text" value="#####"/>	
Range Max. <input type="text" value="#####"/> Range Min. <input type="text" value="#####"/>	Forced <input type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<b>Warnings</b> I/O Error <input checked="" type="checkbox"/> I/O Simulated <input checked="" type="checkbox"/> Forced <> Process <input checked="" type="checkbox"/> I/O Error Blocked <input checked="" type="checkbox"/>	
<input type="button" value="Set Value..."/>			
<input type="button" value="Auto Mode"/>		<input type="button" value="Forced Mode"/>	
<input type="button" value="Ack. Alarm"/>		<input type="button" value="Select"/>	

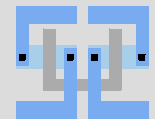
Status	Operation Modes	Alarms	
On <input checked="" type="checkbox"/> Position <input type="text" value="#####"/> Limit On <input type="text" value="#####"/> Limit Off <input type="text" value="#####"/> Failsave On / Opened <input checked="" type="checkbox"/>	Auto <input type="checkbox"/> Inh <input type="checkbox"/> On <input checked="" type="checkbox"/> Manual <input type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> Forced <input type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> Local <input type="checkbox"/> <input checked="" type="checkbox"/> Hardware Local <input type="checkbox"/> <input checked="" type="checkbox"/>	Full Stop Interlock <input checked="" type="checkbox"/> Temporary Stop Interlock <input checked="" type="checkbox"/> Start Interlock <input checked="" type="checkbox"/> Alarm <input checked="" type="checkbox"/> Alarm Not Ack. <input checked="" type="checkbox"/>	
<b>Ranges</b> Range Max. <input type="text" value="#####"/> Range Min. <input type="text" value="#####"/>	<b>Requests</b> Auto <input type="text" value="#####"/> On <input checked="" type="checkbox"/> Off <input checked="" type="checkbox"/> Manual <input type="text" value="#####"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> Active <input type="text" value="#####"/> <input checked="" type="checkbox"/>	<b>Warnings</b> Allow restart needed <input checked="" type="checkbox"/> Alarm blocked <input checked="" type="checkbox"/> I/O Error <input checked="" type="checkbox"/> I/O Simulated <input checked="" type="checkbox"/> Manual <> Auto <input checked="" type="checkbox"/> Position Warning <input checked="" type="checkbox"/>	
<input type="button" value="On"/> <input type="button" value="Off"/> <input type="button" value="Set Value..."/> <input type="button" value="Inc."/> <input type="button" value="Dec."/> <input type="button" value="Allow Restart"/>			
<input type="button" value="Auto Mode"/>		<input type="button" value="Manual Mode"/>	
<input type="button" value="Forced Mode"/>		<input type="button" value="Ack. Alarm"/>	
<input type="button" value="Select"/>			

Status	Operation Modes	Alarms	
On / Opened <input checked="" type="checkbox"/> Off / Closed <input checked="" type="checkbox"/> Failsave On / Opened <input type="checkbox"/> Pulse Activation <input type="checkbox"/>	Auto <input type="checkbox"/> Inh <input type="checkbox"/> On <input checked="" type="checkbox"/> Manual <input type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> Forced <input type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> Local <input type="checkbox"/> <input checked="" type="checkbox"/> Hardware Local <input type="checkbox"/> <input checked="" type="checkbox"/>	Full Stop Interlock <input checked="" type="checkbox"/> Temporary Stop Interlock <input checked="" type="checkbox"/> Start Interlock <input checked="" type="checkbox"/> Alarm <input checked="" type="checkbox"/> Alarm Not Ack. <input checked="" type="checkbox"/>	
	<b>Requests</b> On <input checked="" type="checkbox"/> Auto <input checked="" type="checkbox"/> Off <input checked="" type="checkbox"/> Manual <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> Output On <input checked="" type="checkbox"/> Output Off <input checked="" type="checkbox"/>	<b>Warnings</b> Allow restart needed <input checked="" type="checkbox"/> Alarm blocked <input checked="" type="checkbox"/> I/O Error <input checked="" type="checkbox"/> I/O Simulated <input checked="" type="checkbox"/> Manual <> Auto <input checked="" type="checkbox"/> Position Warning <input checked="" type="checkbox"/>	
<input type="button" value="On"/> <input type="button" value="Off"/> <input type="button" value="Allow Restart"/>			
<input type="button" value="Auto Mode"/>		<input type="button" value="Manual Mode"/>	
<input type="button" value="Forced Mode"/>		<input type="button" value="Ack. Alarm"/>	
<input type="button" value="Select"/>			

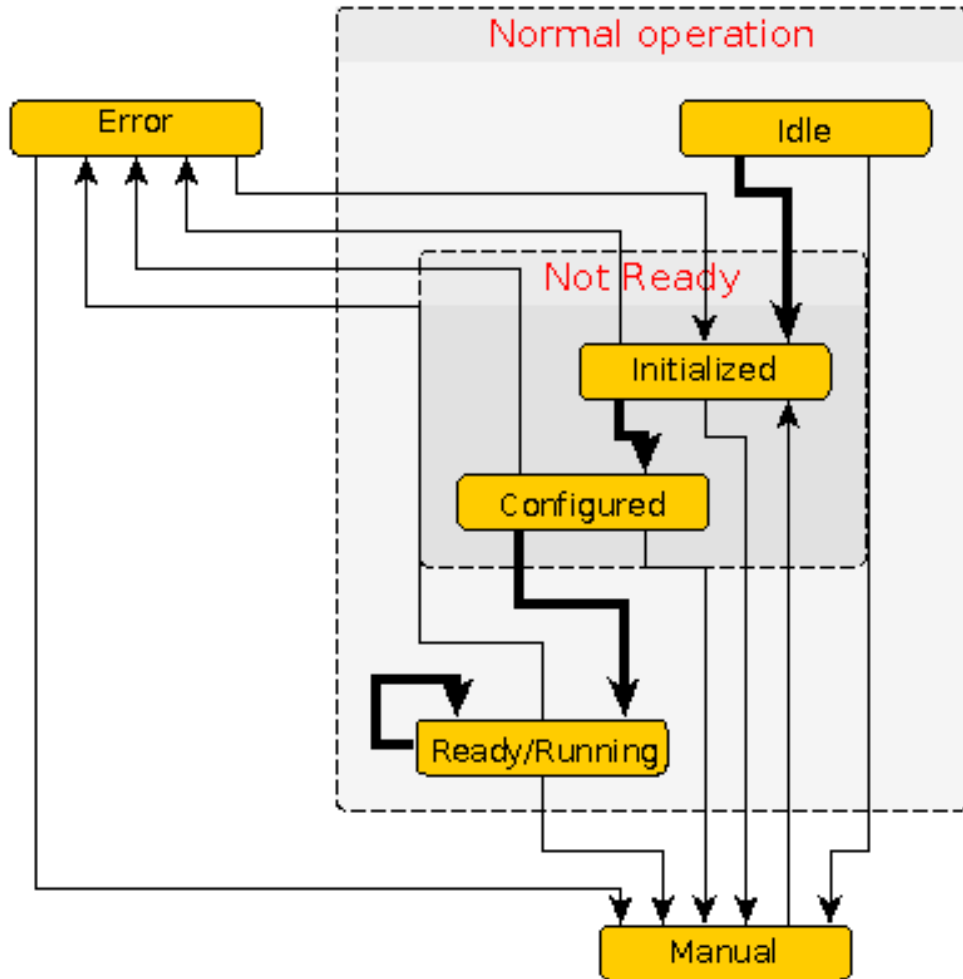


# SVD Slow-Control Recent Activities

- State machines: First version of state machines is being tested.
- Power supplies: IOC for PS has been prepared, and is being documented.
- FADC IOC: C++/EPICS interface on the way. First version of C wrapper was examined.
- Some more data can be found on Slow Control Twiki, and during SVD Software panel.



# State Machine Concept



Startup steps can be divided into two groups:

- Things to do before low voltage is on
- Things to do after low voltage is on

eg: setting up APV must happen after ramping LV.

because of that, Nakao-san's "NotReady" step is subdivided into two steps:

- Initialized
- Configured

Condition for state transition Init->Config is "Power Supply State" is at least "Standby". If "Off" is issued afterwards, we go to "Error"

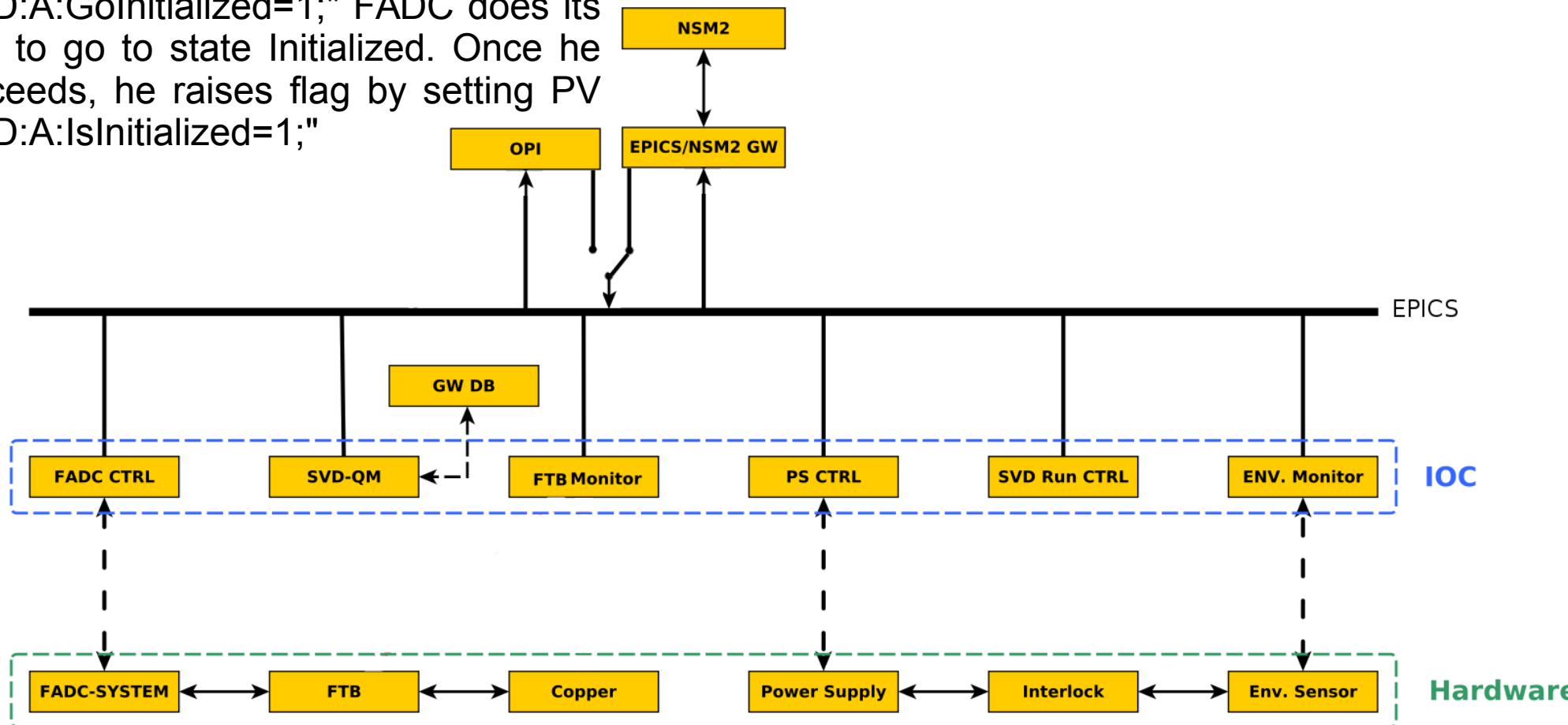
# Hierarchical Approach

All IOCs have their own state machine, with identical states.

For IOCs other than SVDRunCTRL state transitions are dependent of flags raised by SVDRunCtrl.

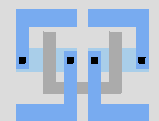
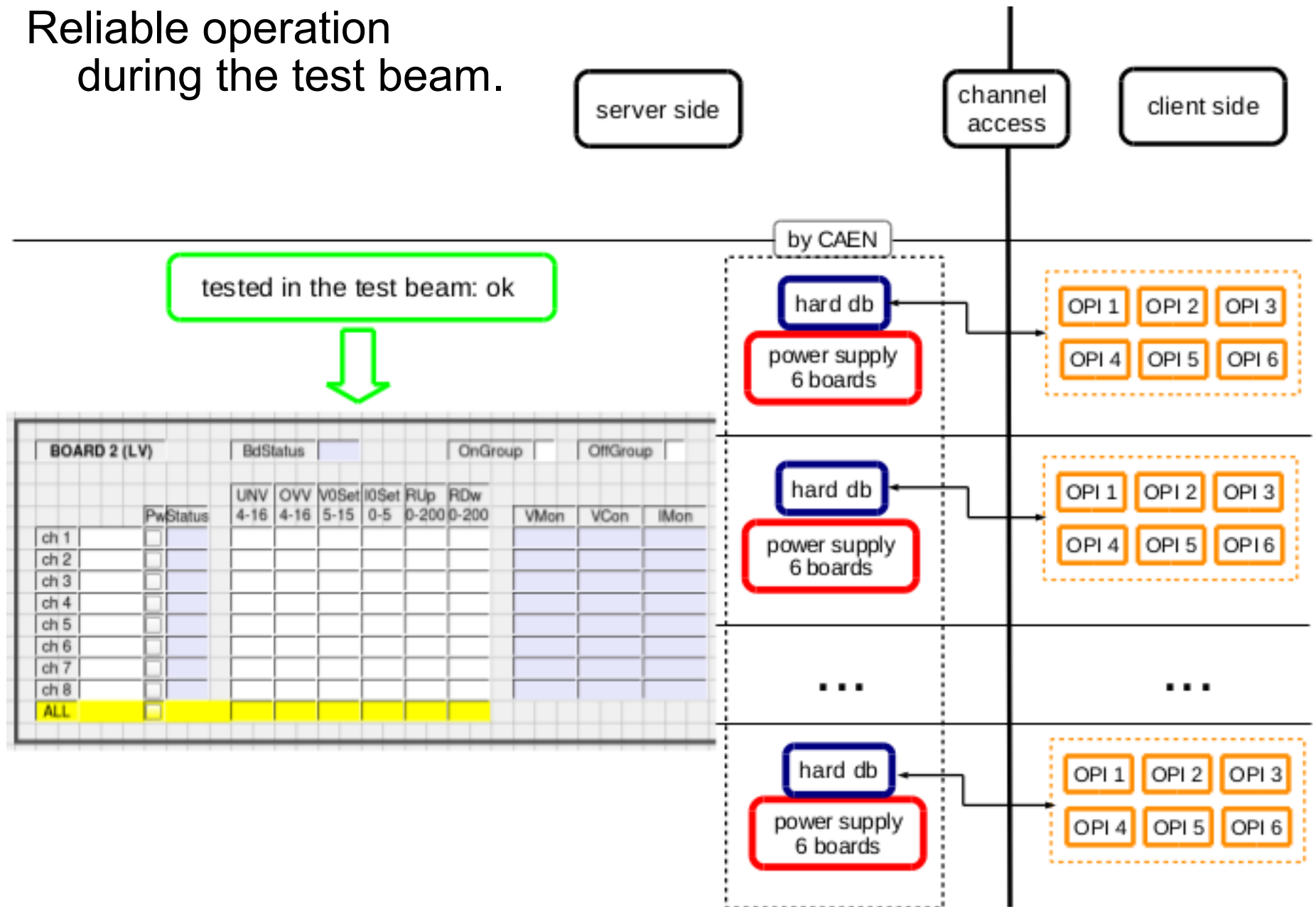
eg: when SVDRunCtrl does "SVD:A:GoInitialized=1;" FADC does its best to go to state Initialized. Once he succeeds, he raises flag by setting PV "SVD:A:IsInitialized=1;"

If any IOC is stuck as some step (eg. PS waits for "Standby", whole system will wait. Also, every IOC can issue "GoToError" for others.



# SVD Power Supply Slow-Control

- Reliable operation during the test beam.

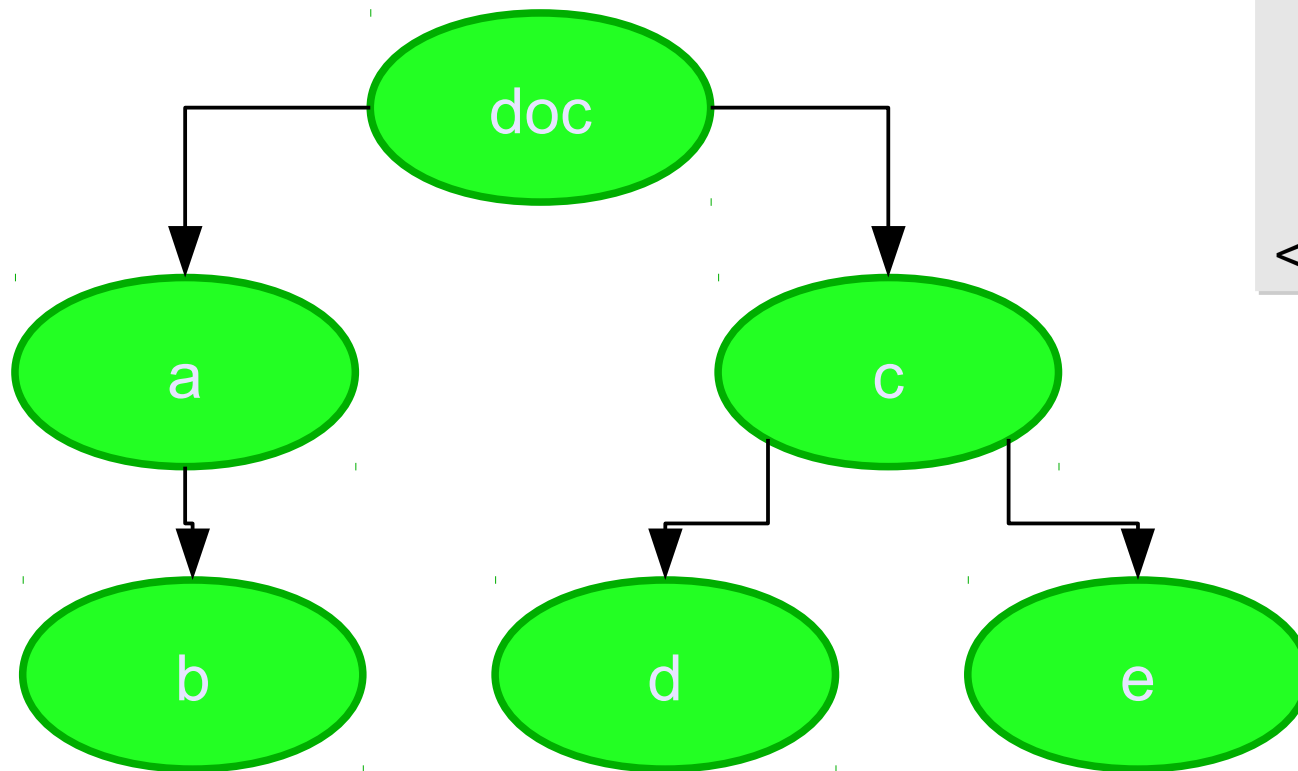


- Include all environment sensors.
  - Update FOS IOC for the readout with multiplexer. To be tested at DESY for mockup tests.
  - NTC and humidity sensor readout.
  - Check what sensors can provide input to environmental interlocks.
- Working group for GUI (aka OPI) guidelines.
- EPICS on ONSSEN and DATCON
  - FPGA: IOC on embedded PowerPC (or IPBus for DATCON)
  - Hardware: IPMI
- Investigate IPMI support of COPPER crates.
- DQM (from ONSSEN data).
- Prepare the control rooms (@KEK + remote).



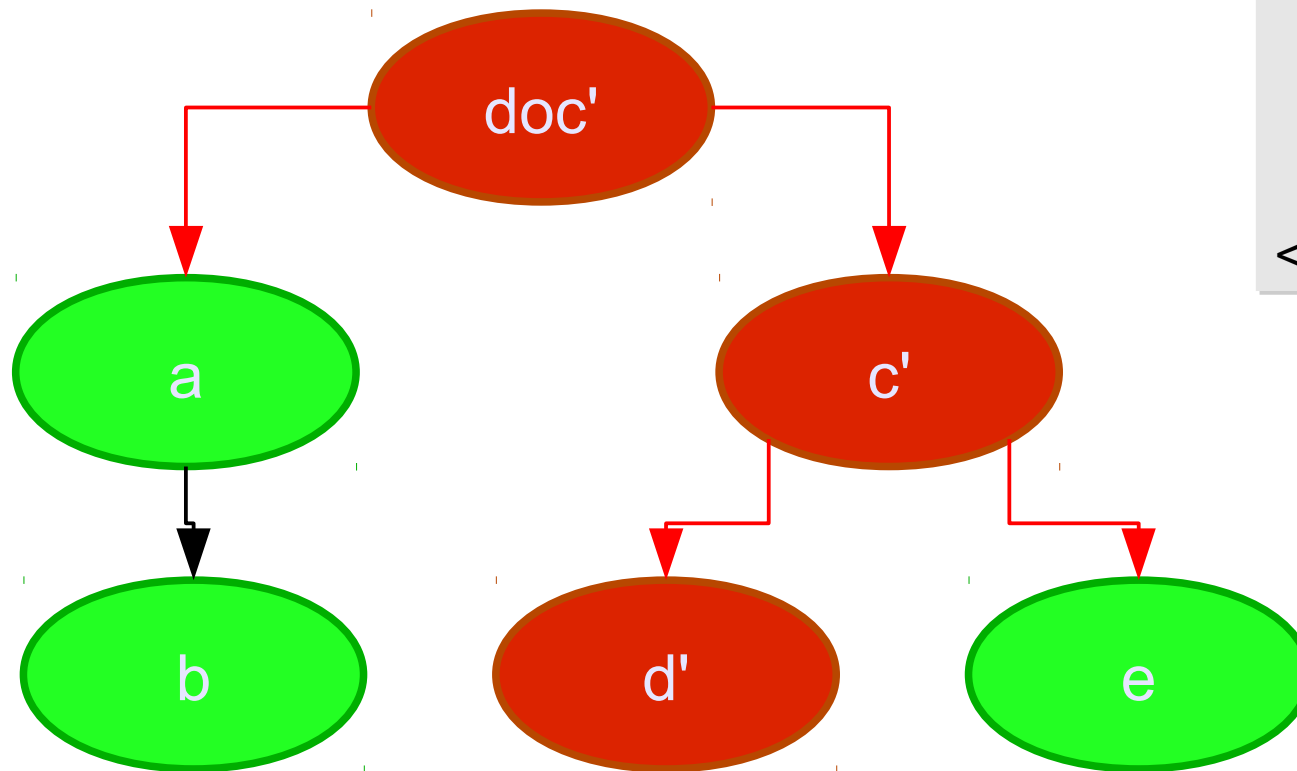
Thank you!

# Data Structure in the Database



```
<doc>  
  <a>  
    <b>12</b>  
  </a>  
  <c>  
    <d>42</d>  
    <e>hello</e>  
  </c>  
</doc>
```

# Updates



```
<doc>  
  <a>  
    <b>12</b>  
  </a>  
  <c>  
    <d>44</d>  
    <e>hello</e>  
  </c>  
</doc>
```

updated data

