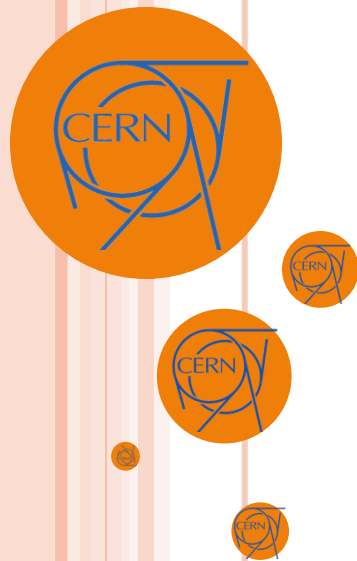


UNICOS: UNIFIED INDUSTRIAL CONTROL SYSTEM CPC (CONTINUOUS PROCESS CONTROL)

**BASIC COURSE
SESSION 0:**

UCPC FROM SPECS TO IMPLEMENTATION



UCPC 6
UNICOS-Continuous Process Control

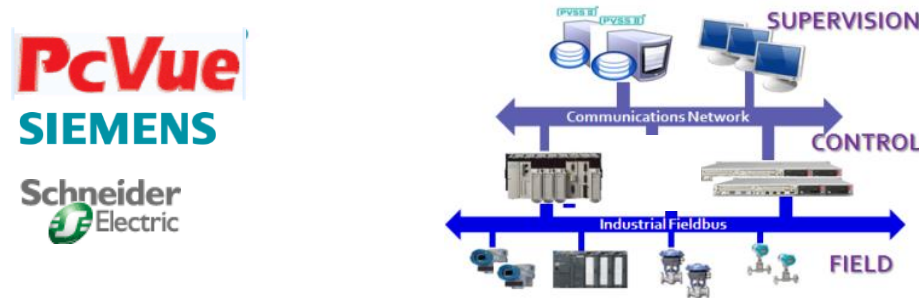


CERN EN/ICE group

- **UNICOS vs. UNICOS-CPC**
- Objects main functionality and connectivity
- From specs to implementation: Overview
 - Life cycle
 - Generation Tools

A LOOK TO THE PAST

- [1998] UNICOS (UNified Industrial Control System) was born at CERN as a need to develop the LHC cryogenics control system. The **goal** was to create an industrial control system covering the three layers of the typical automation pyramid.



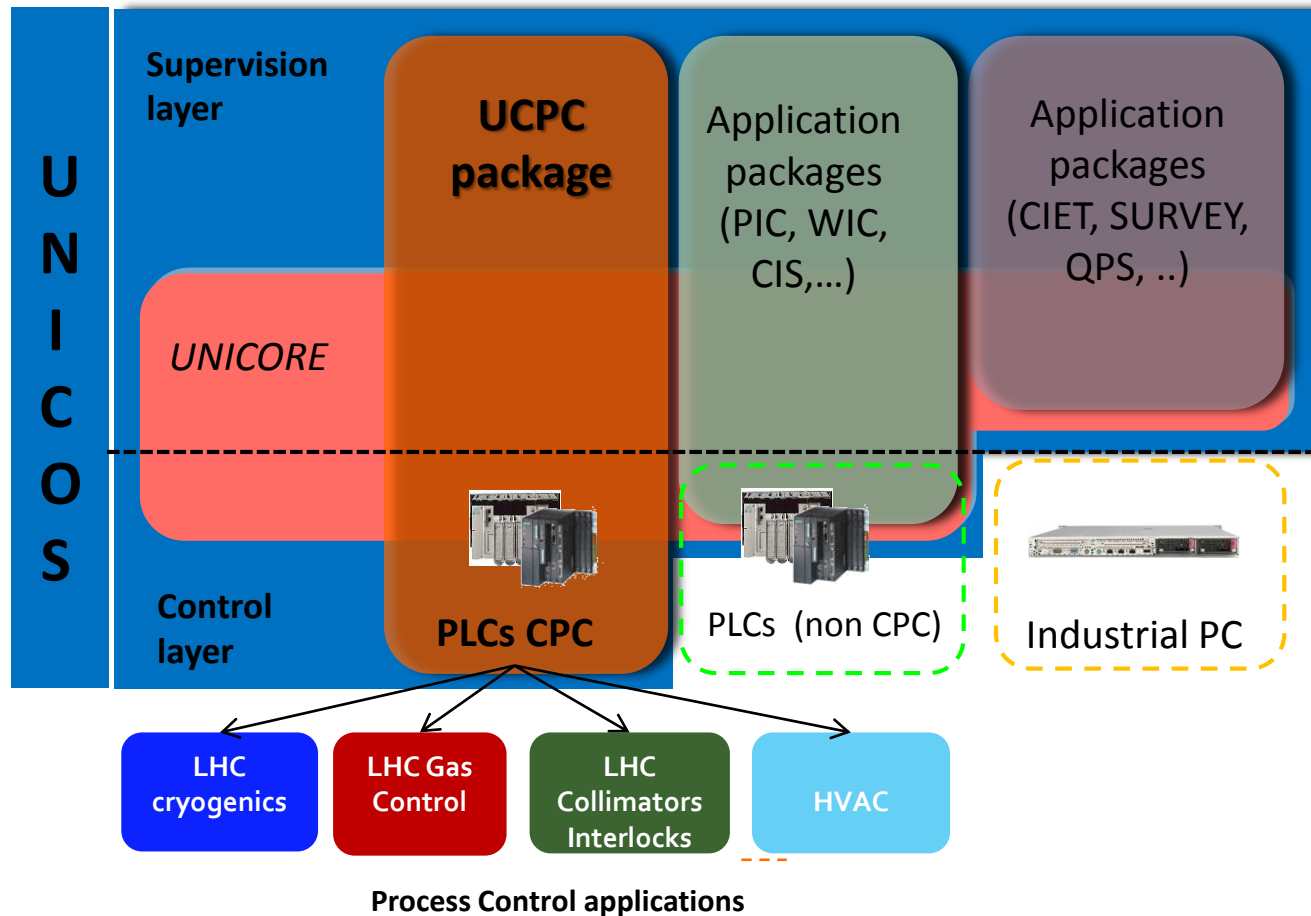
- [2002] UNICOS replace the PcVue SCADA and adopt ETM's PVSS under the CERN recommendation.
- [2004] UNICOS offered the choice of creating applications based on SIEMENS S7 PLCs
- [2009] UNICOS turned into *de facto* **standard framework** to develop industrial control applications at CERN:
Cryogenics, Cooling, HVAC, Vacuum, Interlocks,...
- [2010] The UNICOS framework has been extended to other kinds of applications (i.e.: supervisory: [QPS,SURVEY], monitoring, ...)
- [2012] **Re-engineering** process

A FLAVOR OF APPLICATIONS

- Many UNICOS CPC applications done:
 - LHC Cryogenics
 - Detector and Test facilities cryogenics
 - Magnet Control System
 - Vacuum installations : ATLAS, CMS, ISOLDE
 - LHC collimators: Environmental temperatures
 - ATLAS Big wheels (motion)
 - AMS servomotors control
 - Detector gas control systems
 - Cooling and HVAC installations
 - Winding machines: HTS cable (hybrid with a Safety system)
 -

UNICOS AND UCPC

- UNICOS is a **framework** to create control applications
 - UCPC**: A basic package (**C**ontinuous **P**rocess **C**ontrol) to develop integrated process control applications.



USE CASE: SURVEY (No PLC)

- LHC Inner Triplet magnet alignment (tolerance ~ 100 micron)
- Front End: FESA devices (DOMS, WPS, HLS, Steiner, ZTS, ESTOP, ...)

Central
Control Rooms



SCADA Data Servers

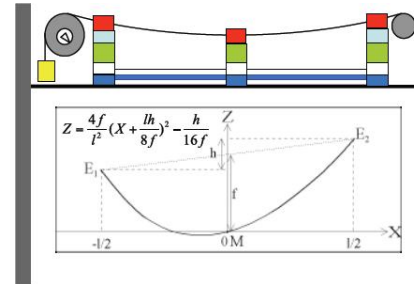


Ethernet (TN)

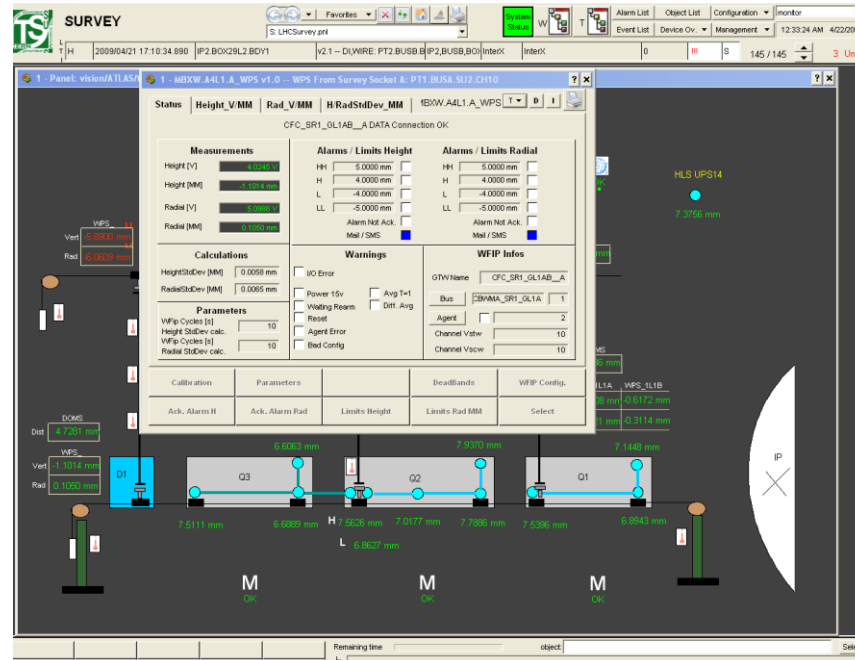


WorldFIP

General Architecture

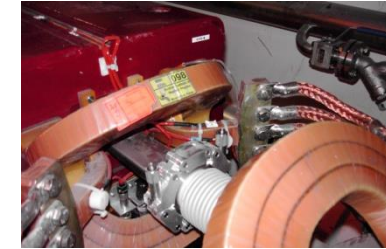


WPS: stretched-wire pos. system



USE CASE: WIC (PLC NON UNICOS)

- Warm Interlock controller: monitors the magnet and safeguards it from overheating
- **PLC is not UNICOS-like.**



Magnet thermo-switches @ 60C

Central Control Rooms

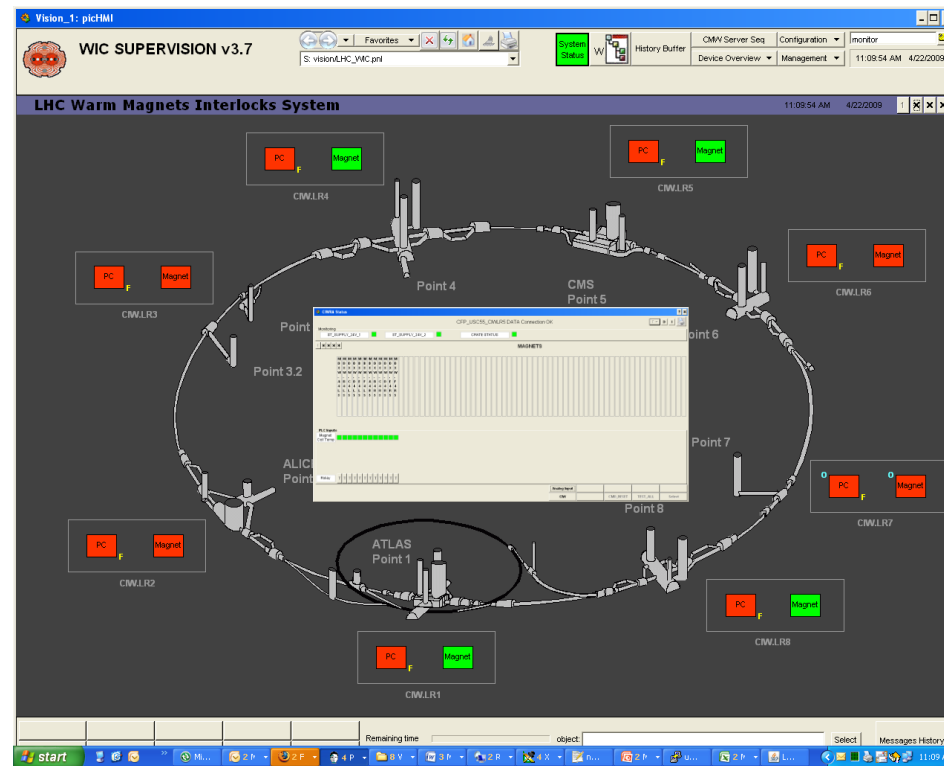


PVSS II



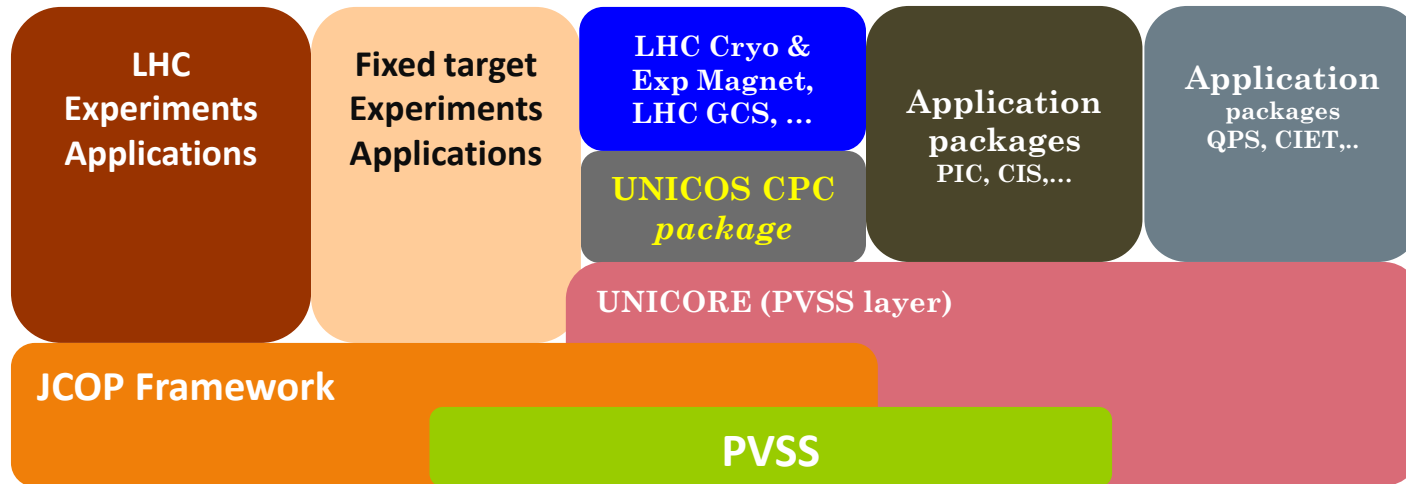
Ethernet (TN)

Profibus



FRAMEWORKS INTEGRATION

- At the supervision level, UNICOS framework reuses some of the JCOP framework components



- **Standards (or *best-practices*)**
ISA-88 / IEC-61512 / IEC-61499 : Batch control / Distributed systems
 - Open architecture for distributed control and automation, applying object oriented techniques to process control
 - Organize objects into a hierarchy (process/equipment hierarchy)
 - Well placed and separated code for the process control logic
- Process control systems design: **Object Orientation**
 - *Modularity*: optimize design, tests and maintainability
 - *Abstraction*: only interfaces seen, reduces complexity
 - *Generic*: Any module can be plug-in without major impact
 - *Re-use*: Minimize engineering effort

NOT ONLY A BUNCH OF DEVICES

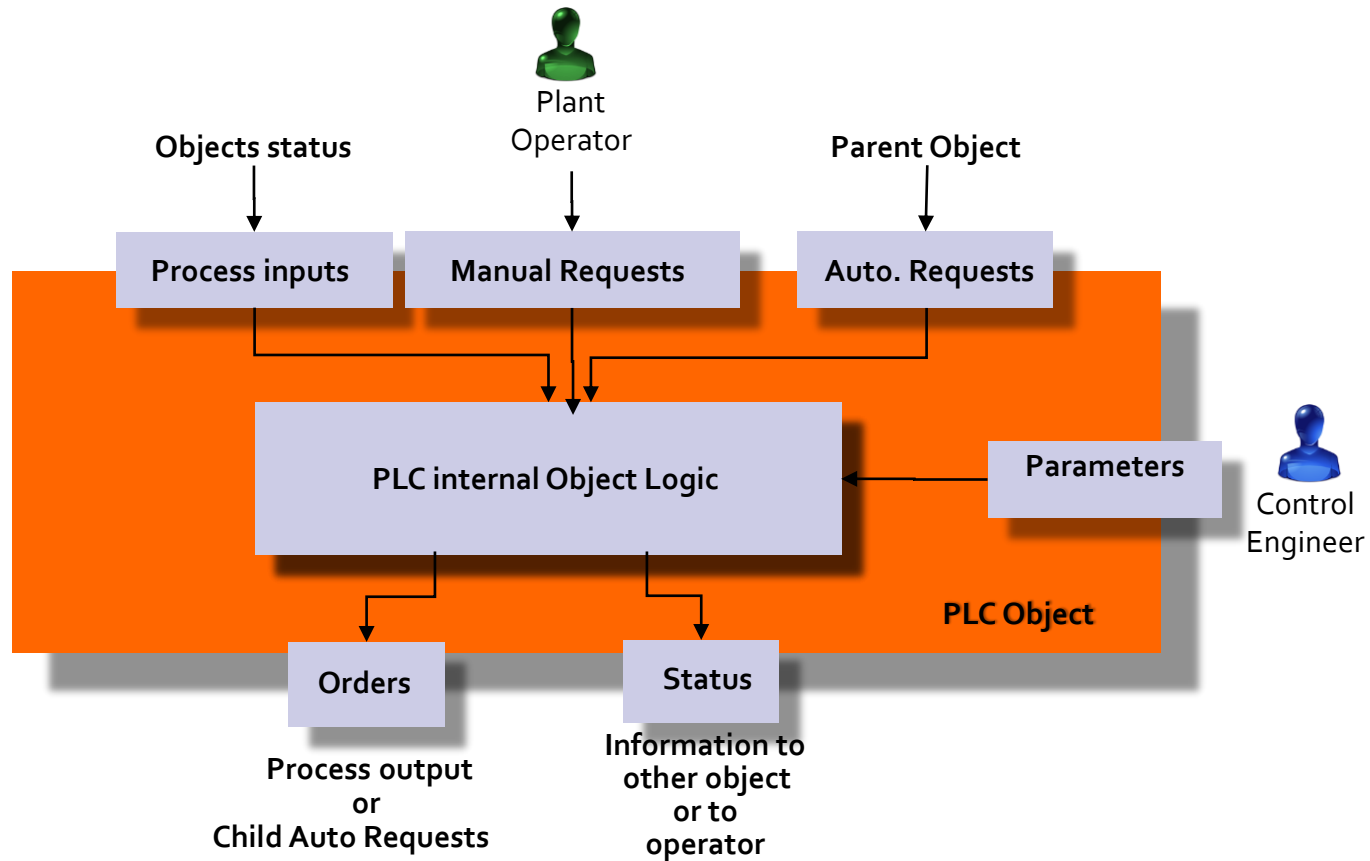
- UNICOS CPC provides libraries (control and supervision layers)
- A well defined set of **standard device types** (objects), modeling most of the equipment and needs of continuous processes and the **relationships** between them.
 - **I/O Objects**
 - ✓ Digital I/O
 - ✓ Analog I/O
 - **Field Objects**
 - ✓ OnOff
 - ✓ Analog
 - ✓ AnalogDigital
 - ✓ Local
 - ✓ AnaDO
 - **Control Objects**
 - ✓ Controller
 - ✓ Alarms
 - ✓ Process Control Object
 - **Interface Objects**
 - ✓ Parameter (Digital, Word, Analog)
 - ✓ Status (Word, Analog)
- A **formalized** way of :
 - Define the **control units** of a process (ISA-88 standard: Batch processes)
 - **Programming the specific process logic** for those units

- UNICOS vs. UNICOS-CPC

- **Objects main functionality and connectivity**
 - **Objects**
 - **Relationships**

- From specs to implementation: Overview
 - Life cycle
 - Generation Tools

UNICOS CPC OBJECT MODEL

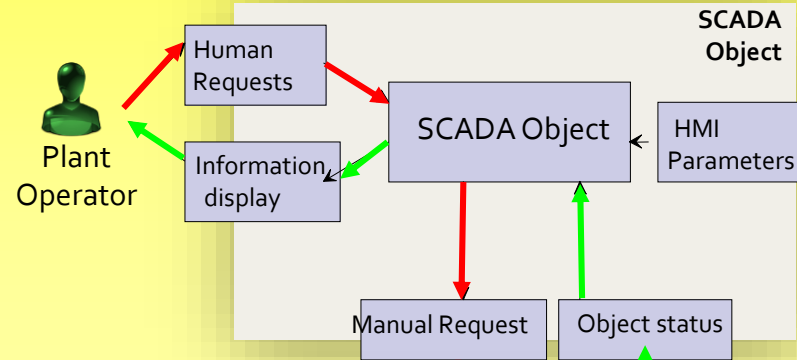


OBJECTS & LAYERS INTEGRATION

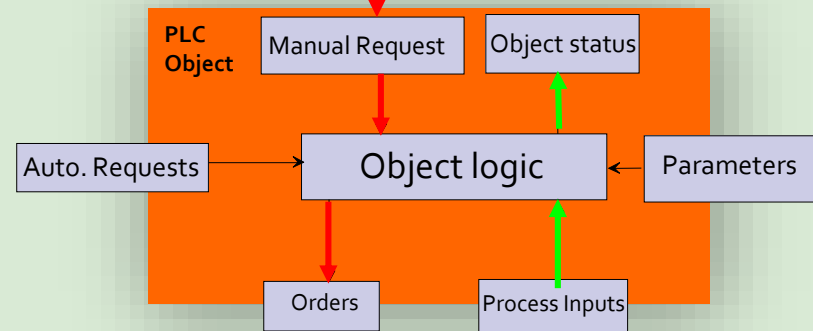
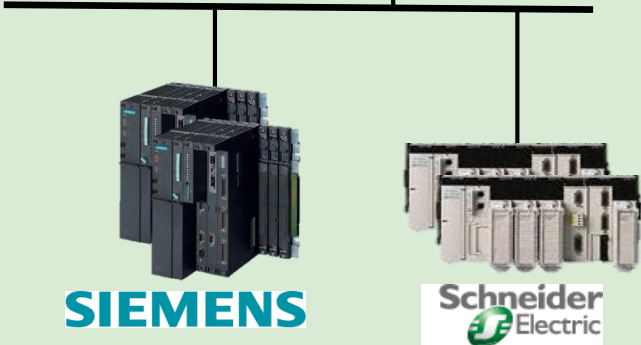
Supervision Layer



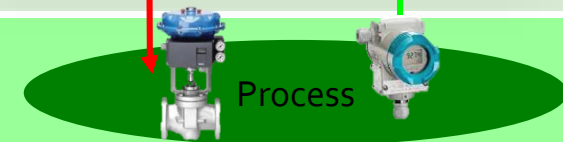
In the Supervision layer the object presents the relevant information to the operator and allow manual commands



Control Layer

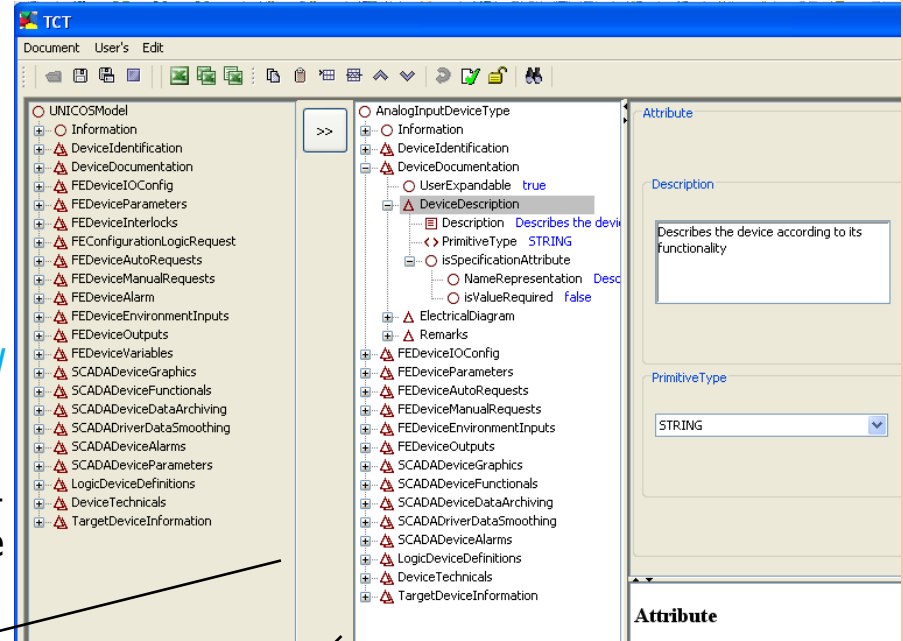
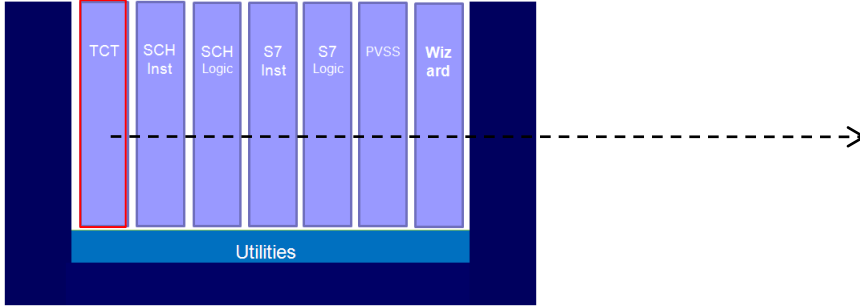


Field Layer



DEVICES CREATION: TCT

TCT: define the UNICOS Types



- The model is supported by a *meta-model* describing the properties of the model.
- TCT (Type Creation Tool) provides a *drag&drop*-based mechanism to build new device type definition described in XML

Specifications.xml



| | A | B | C | D | E | F |
|----|----------------------|-------------------------|--------------------|---------|------------------|-----------------|
| 1 | Name | AnalogInput | | | | |
| 2 | Object Type Family | IOObjectFamily | | | | |
| 3 | Description | 19/Jun/2010 | | | | |
| 4 | Version | 1.0 | | | | |
| 5 | Version Commented | by the development team | | | | |
| 6 | Status | UnderDevelopment | | | | |
| 7 | | | | | | |
| 8 | DeviceIdentification | DeviceDocumentation | | | | |
| 9 | | | | | | |
| 10 | Name | Description | Electrical Diagram | Remarks | FE encoding type | InterfaceParam1 |
| 11 | | | | | | |
| 12 | | | | | | |
| 13 | | | | | | |
| 14 | | | | | | |
| 15 | | | | | | |
| 16 | | | | | | |
| 17 | | | | | | |

UNICOSTypeDefinition.xml



Type documentation



EN/ICE Web page

TCT (TYPE CREATION TOOL)

UNICOS model

- UNICOSModel
 - + ○ Information
 - + △ DeviceIdentification
 - + △ DeviceDocumentation
 - + △ FEDeviceParameters
 - + △ FEDeviceInterlocks
 - + △ FEConfigurationLogicRequest
 - + △ FEDeviceAutoRequests
 - + △ FEDeviceManualRequests
 - UserExpandable true
 - △ ManReg01
 - Meaning Manual Register 1
 - Description Manual Register 1
 - isCommunicated true
 - <> PrimitiveType SHORTINT16
 - + △ MAuMoR
 - + △ MMMoR
 - + △ MFoMoR
 - + △ MOnR
 - + △ MOffR
 - + △ MNewPosR
 - + △ MAIAckR

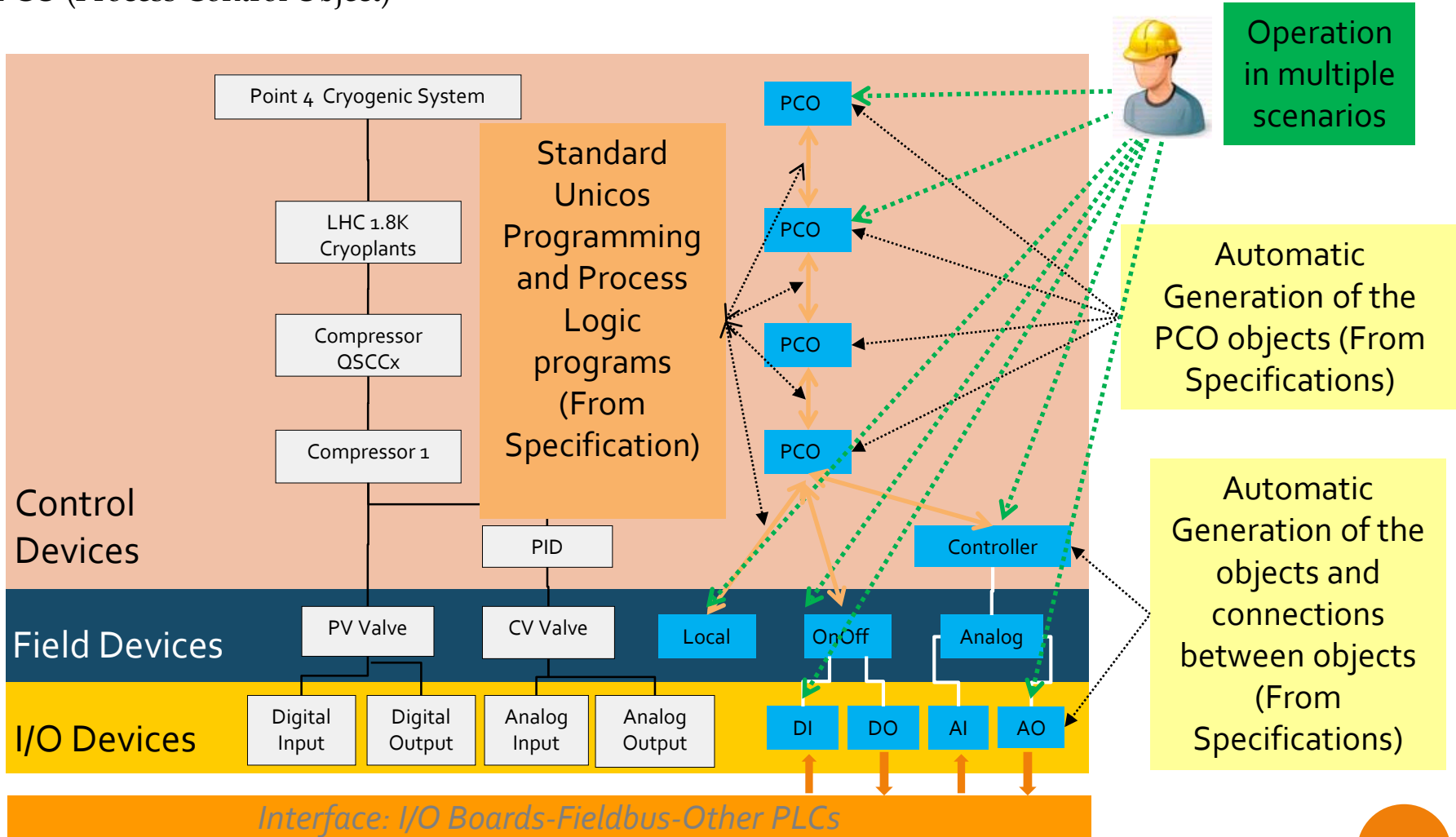
- AnalogDeviceType
 - + ○ Information
 - + △ DeviceIdentification
 - + △ DeviceDocumentation
 - + △ FEDeviceParameters
 - + △ FEDeviceInterlocks
 - + △ FEDeviceAutoRequests
 - △ FEDeviceManualRequests
 - UserExpandable true
 - △ ManReg01
 - Meaning Manual Register 1
 - Description 1st Manual Register
 - isCommunicated true
 - <> PrimitiveType SHORTINT16
 - △ MAuMoR
 - Meaning Manual Auto Mode Request
 - Description The operator requests the Auto
 - <> PrimitiveType BIT1
 - BitPosition 0
 - + △ MMMoR
 - + △ MFoMoR
 - + △ MSoftLDR
 - + △ MOnR
 - + △ MOffR
 - + △ MNewPosR
 - + △ MStpInR
 - + △ MStpDeR

- DigitalAlarmDeviceType
 - + ○ Information
 - + △ DeviceIdentification
 - + △ DeviceDocumentation
 - + △ FEDeviceParameters
 - + △ FEDeviceAutoRequests
 - △ FEDeviceManualRequests
 - UserExpandable true
 - △ ManReg01
 - Meaning Manual Register 1
 - Description 1st Manual Register
 - isCommunicated true
 - <> PrimitiveType SHORTINT16
 - + △ MAIBSetRst
 - + △ MAIAckR
 - + △ FEDeviceAlarm
 - + △ FEDeviceEnvironmentInputs
 - + △ FEDeviceOutputs
 - + △ SCADADeviceGraphics
 - + △ SCADADeviceFunctionals
 - + △ SCADADeviceDataArchiving
 - + △ SCADADeviceAlarms
 - + △ LogicDeviceDefinitions

| DeviceIdentification | DeviceDocumentation | | | FEDeviceIOConfig | | FE | |
|----------------------|-----------------------------------------|--------------------|---------|------------------|-----------------|-----------|-----------|
| Name | Description | Electrical Diagram | Remarks | FE encoding type | InterfaceParam1 | Range Min | Range Max |
| QSDN_4_1TT4001 | Vessel 1- Heater section1-Temp. control | A11.0 | | | %IW1.1.0 | 80 | 350 |
| QSDN_4_AI1 | SPARE | A11.1 | | | %IW1.1.1 | 0 | 100 |
| QSDN_4_1TT4002 | Vessel 1- Heater section2-Temp. control | A11.2 | | | %IW1.1.2 | 80 | 350 |
| QSDN_4_1TT4003 | Vessel 1- Heater section3-Temp. control | A11.3 | | | %IW1.1.3 | 80 | 350 |
| QSDN_4_1LE400 | Vessel 1- LN2 Level | A11.4 | | | %IW1.1.4 | 0 | 1350 |
| QSDN_4_1PT400 | Vessel 1- LN2 Vessel Pressure | A11.5 | | | %IW1.1.5 | 0 | 4.0 |

- **I/O Objects**
- **Field Objects**
- **Control Objects**
- **Interface Objects**

- Each control module or equipment module is a device
- Equipment modules and Units are embedded in a unique object class: PCO (Process Control Object)



○ **Functionality**

- Base components
- PLC Periphery interface and/or internal memory variables

○ **Types**

- AI, AIR: Analog Input or Analog Input Real
(*e.g. temperature transmitter*)
- DI: Digital Input
(*e.g. end contact*)
- AO, AOR: Analog Output or Analog Output Real
(*e.g. control valve position order*)
- DO: Digital Output
(*e.g. on/off valve position order*)

Widget examples



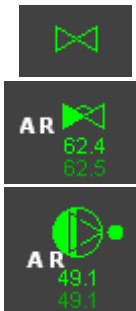
○ Functionality

- Model the real field equipments (e.g. pumps, valves...)
- As a general rule, the field objects are connected to the I/O Objects. No direct connection to the PLC periphery.

○ Types

- *OnOff*: Binary Objects
(e.g. on/off valve, motor, pump)
- *Analog*: Analog objects
(e.g. control valve, heater)
- *Anadig*: Analog inputs and Digital outputs objects
(e.g. valves/heaters controlled by on/off pulses)
- *AnaDO* : Similar functionality of an OnOff + Analog object
(Motor with VFD, Thyristor, Heater, etc.)
- *Local*: Field localized objects :
(e.g. manual valve)

Widget examples



- **Functionality**
 - Main objects holding the control logic
 - Feedback controllers
 - Handle the abnormal situations: Alarms and interlocks

- **Types**
 - *PCO*: Process Control Objects/Unit. It implements the control logic (e.g. *Compressor Station*)

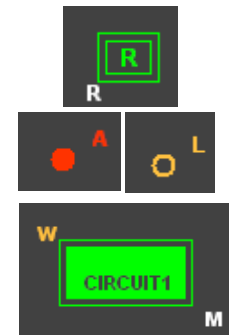
 - *Controller*: feedback control objects (e.g. *PID controller*)

 - *AA, DA*: Analog/Digital Alarm Objects. It models alarms and interlocks. Analog alarms include alarm and warning thresholds (e.g. *Temperature Too High*)

Options for an AA:

- *Explicitthreshold*: Initialized in PLC and then modified from SCADA
- *Logic*: Set by control logic in the PLC
- *APAR* : Linked object APAR sets the value

Widget examples



○ Functionality

- Parameterization and status
- Can be connected to the periphery
- Light objects

○ Types

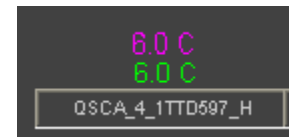
- *DigitalParameter, WordParameter, AnalogParameter*: Parameters
(e.g. *Threshold*)

Can be set by an operator (SCADA -> PLC)

- *WordStatus, AnalogStatus*: Status
(e.g. *stepper position, PA valve feedback*)
PLC ->SCADA

Widget examples

- FailSafe Pos
- Req Local Oper
- Dev oper locally
- Manual mode
- motion deviation
- Stop r (~open)
- Stop r (~closed)
- compressed air
- cmd open sent
- cmd close sent
- param change
- simulation
- N/A
- Ctrl Fault
- control inactive
- self test



- **Operation Modes**
- **Alarms & Interlocks**
- **Controllers**
- **Recipes**

- **Auto Mode**
 - The object is driven by the control logic of a higher object of the hierarchy.
 - Interlocks apply to the request
- **Manual Mode** (requested by operators via the OWS)
 - The automatic return to the auto mode is possible by the control logic.
 - Interlocks apply to the request
- **Forced Mode** (requested by operators via the OWS)
 - The automatic return to the auto mode is impossible by the control logic.
 - Interlocks apply to the requests.
- **Local mode**
 - **Hardware Local Mode**
 - The object is driven locally by the process field (activated via a DI)
 - E.g. maintenance purposes
 - **Software Local Mode** (requested by operators via the Local panels)
 - The Local software is writing directly in the manual requests of the objects
 - **Priority** over "Auto" and "Manual" mode. The "forced mode" setup by the normal SCADA can override the software local mode.
 - Interlocks apply to the requests.

ALARMS & INTERLOCKS (1)

- **Interlock:** Asynchronous condition carrying an actuator or a unit to its safety position (or preventing from starting). An interlock must not be used for normal operation but for abnormal behaviour. UCPC (soft) interlocks are not guarantying functional safety !
- The possible interlocks for a complete unit or for an actuator are:
 - **Full Stop Interlock (FS):** Stop the unit/actuator (all dependent units/actuators are set to their fail-safe position) and wait manual acknowledgement before restarting.
 - **Temporary Stop Interlock (TS):** Stop the unit/actuator (all dependent units/actuators are set to their fail-safe position) and restart automatically when the interlock disappears.
 - **Start Interlock (SI):** Prevent the unit from starting (all dependent units/actuators stay in their fail-safe position).
- **Alarm (AL):** It is an indication of a potential problem to aware operator in SCADA.
- *Alarm and Interlock are only one object with different functionality*

ALARMS & INTERLOCKS (2)

○ Acknowledgement

- Can be done in advance while the alarm is still active
- Can be directly performed to a unit/actuator, then, all dependent alarms of this unit/actuator are automatically acknowledged by propagation.

○ Mask/Block

- Operators can **mask** alarms in the UI. The alarm is still active in the PLC !
- Operators can **block** alarms in the UI. The alarm is then blocked in the PLC, then no subsequent actions due to that alarm (if any) !!!!

○ Resume Full stop on units/actuators.

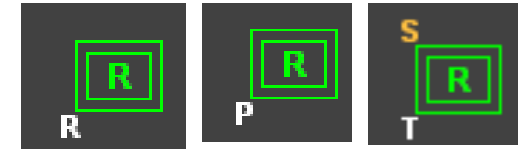
The operator have two options to resume the unit (the unit will resume only once the interlock is gone):

- By ACK the interlock
- By ACK and “**allow restart**” action explicitly. This can be done even while the interlock is active (if configured in the specifications)

- UNICOS Mode management as other field objects
 - **Auto/Manual/Forced** with same meanings



- Working States
 - **Regulation**
 - **Positioning**
 - **Tracking**



- PID features:
 - 1st order filter on Measured Value
 - PID Cycle time
 - Scaling mode (No Scaling / Input Scaling / IO Scaling)

- PID cascade automatically generated

| DeviceId | FEDeviceParameters | | |
|-----------------------|--------------------|---------------|----------------------|
| Controller Parameters | | | |
| Name | MV Filter Time (s) | PID Cycle (s) | Scaling Method |
| HRM_876_TC0001 | 0.1 | 0.1 | No Scaling |
| HRM_876_PID1 | 0.02 | 0.05 | Input Scaling |
| HRM_876_PID2 | 0.1 | 0.1 | Input/Output Scaling |

- Allows bulk parameterization and/or multiple setting in one shot
- Established list of recipe-able objects (e.g. PID parameters, Xpar for Thresholds,...)

Recipe Class: RcpClass2 Recipe: Instance1

Recipe class: RcpClass2

Class: RcpClass2
Last Activated: RcpClass2/Instance1
Status: [Active]
Class Desc.: Rcp Class 2 Description
Rcp. Desc.: Example of RcpClass2 instance

Creator: admin Creation time: 2011.11.07 14:26:53.112
Last modifier: admin Last modification time: 2011.11.09 18:09:37.036
Last activator: admin Last activation time: 2011.11.15 09:30:28.011

| Index | Alias | Description | Value | Unit | Range |
|-------|--------------|--------------------------------|----------|------|----------------------|
| 1 | DEMON_1_AA1 | Analog Alarm 2 : HH | 500.00 | | |
| 2 | DEMON_1_AA2 | Analog Alarm 2 : L | 30.00 | | |
| 3 | DEMON_1_AA2 | Analog Alarm 2 : LL | 20.00 | | |
| 4 | DEMON_1_AA2 | Analog Alarm 2 : LL | 20.00 | | |
| 5 | DEMON_1_AP26 | Analog Parameter 26 : MPosR | 3.300 e1 | cm | [1.500 e1, 3.500 e1] |
| 6 | DEMON_1_Ctr2 | Controller 2 : MKc | 10.580 | | |
| 7 | DEMON_1_Ctr2 | Controller 2 : MOutH | 100 | cm | |
| 8 | DEMON_1_Ctr2 | Controller 2 : MOutL | 0 | cm | |
| 9 | DEMON_1_Ctr2 | Controller 2 : MSP | 3.300 e1 | °C | [0.000, 1.800 e2] |
| 10 | DEMON_1_Ctr2 | Controller 2 : MSPH | 3.500 e1 | °C | |
| 11 | DEMON_1_Ctr2 | Controller 2 : MSPL | 1.500 e1 | °C | |
| 12 | DEMON_1_Ctr2 | Controller 2 : MTd | 0.000 | | |
| 13 | DEMON_1_Ctr2 | Controller 2 : MTds | 0.000 | | |
| 14 | DEMON_1_Ctr2 | Controller 2 : MTi | 40.000 | | |
| 15 | DEMON_1_DP2 | Digital Parameter 2 : ManReg01 | TRUE | | [FALSE, TRUE] |

Activation Timeout: Selection Timeout:

[2011.11.15 09:30:21] Activating recipe: RcpClass2 / Instance1
[2011.11.15 09:30:21] Sending recipe data to the PLC dist_1:_unPlc_CFP_PLC1
[2011.11.15 09:30:23] All recipe data sent to the PLC dist_1:_unPlc_CFP_PLC1
[2011.11.15 09:30:28] Recipe activation completed in PLC dist_1:_unPlc_CFP_PLC1

Buttons: Activate, Edit, New Instance, Duplicate, Delete, Online Values, Save, Cancel, Deselect

| | initialRecipe | deviceType | deviceAlias | dpe | value | unit | range |
|--|---------------|------------------|--------------|----------|-------|------|----------|
| | TRUE | - | | | | | |
| | | DigitalParameter | DEMON_1_DP2 | ManReg01 | TRUE | | |
| | | WordParameter | DEMON_1_WP2 | MPosR | 25 | bar | [0, 100] |
| | | AnalogParameter | DEMON_1_AP26 | MPosR | 33 | cm | [15, 35] |
| | | AnalogAlarm | DEMON_1_AA2 | HH | 500 | °C | [0, 220] |
| | | | | H | 450 | °C | [0, 220] |
| | | | | L | 30 | °C | [0, 220] |
| | | | | LL | 20 | °C | [0, 220] |
| | | Controller | DEMON_1_Ctr2 | MSP | 33 | | |
| | | | | MSPH | 35 | | |
| | | | | MSPL | 15 | | |
| | | | | MOutH | 100 | | |
| | | | | MOutL | 0 | | |
| | | | | MKc | 10.58 | | |
| | | | | MTi | 40 | | |
| | | | | MTd | 0 | | |
| | | | | MTds | 0 | | |

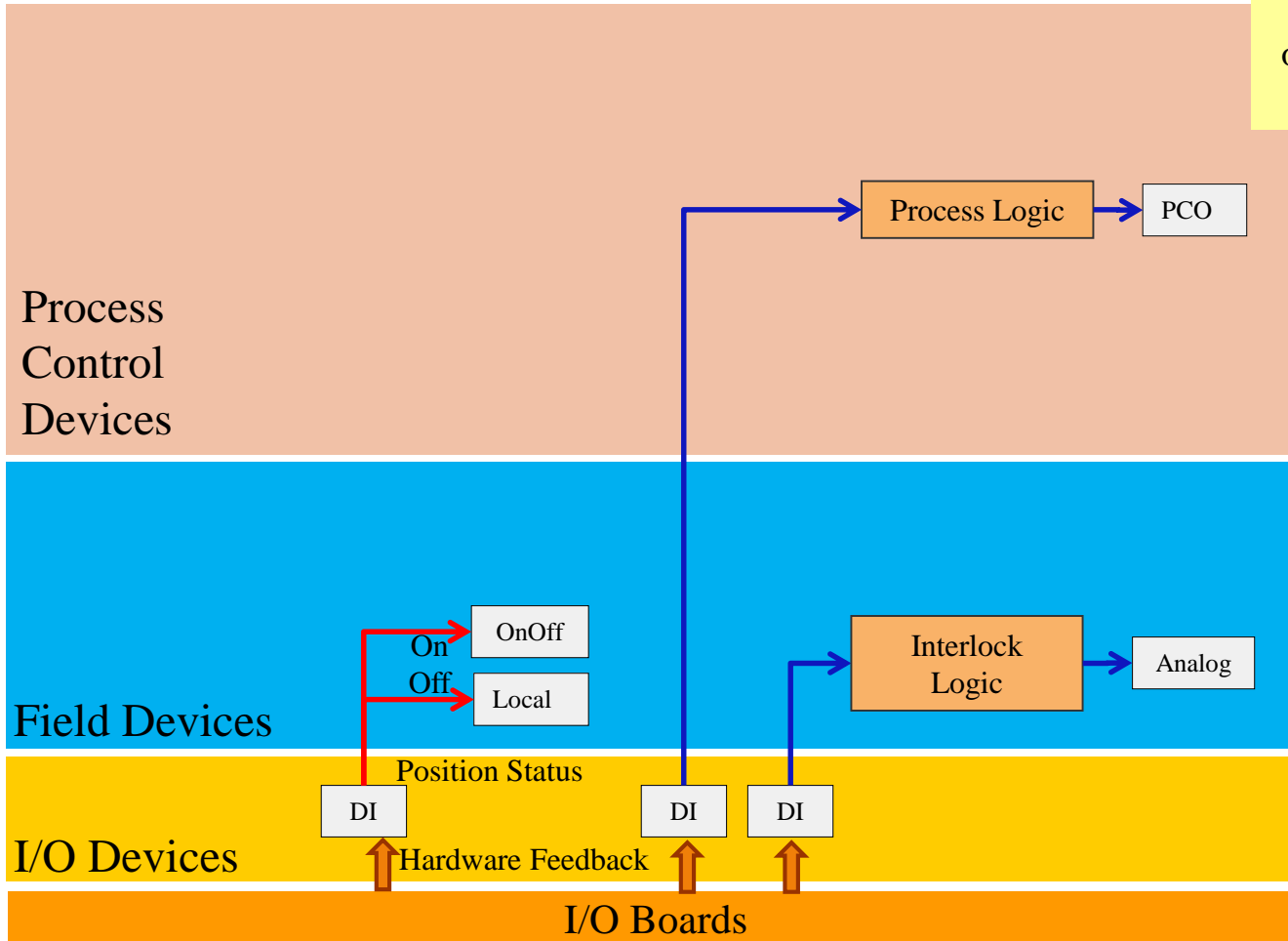
Example of a recipe instance

Example of a HMI recipe tool (creation, activation, edit,...)

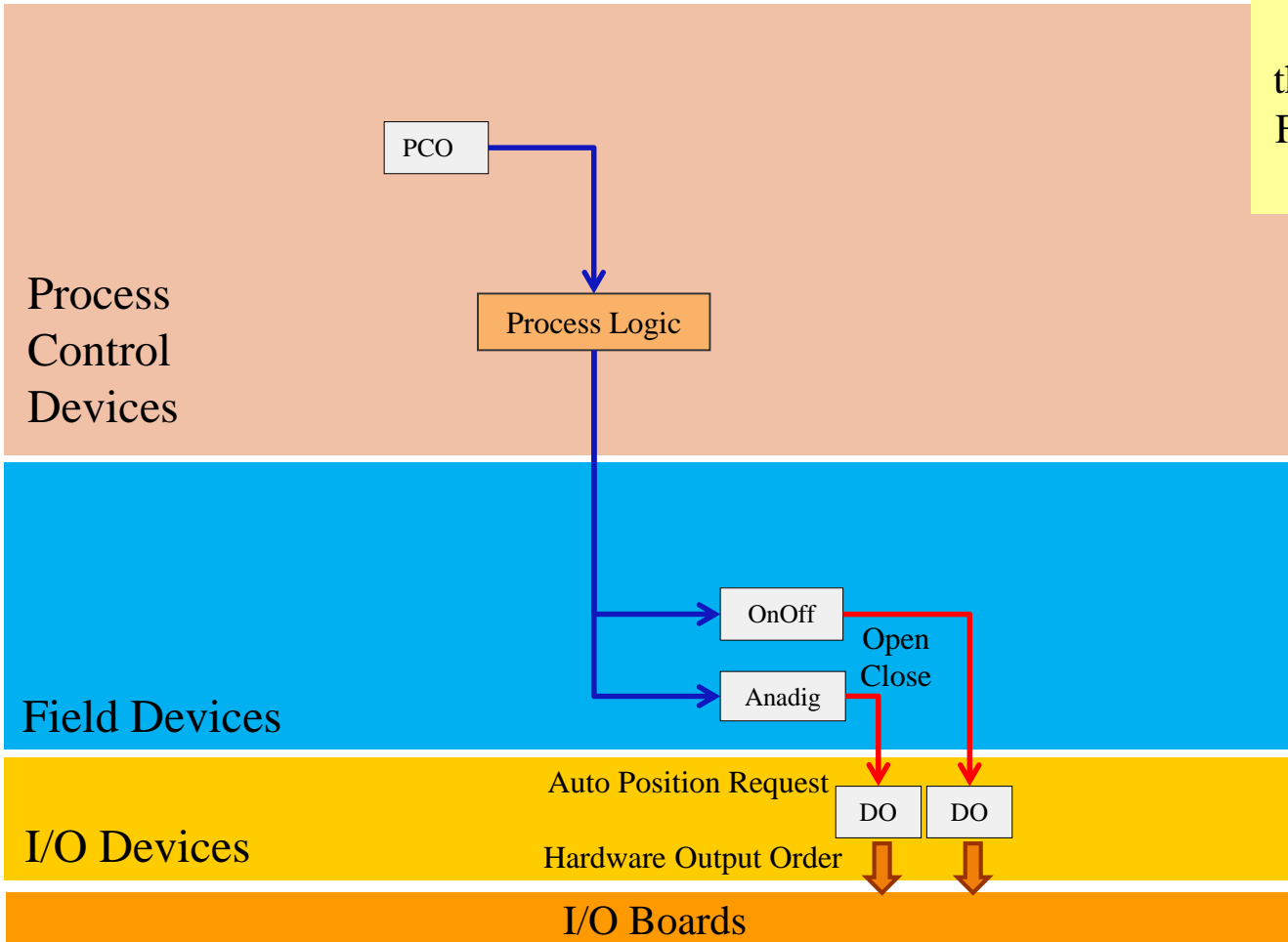
- **Connectivity between objects**
- **Relationships when programming**



Status of DI Objects may be used by all objects anywhere in the program.



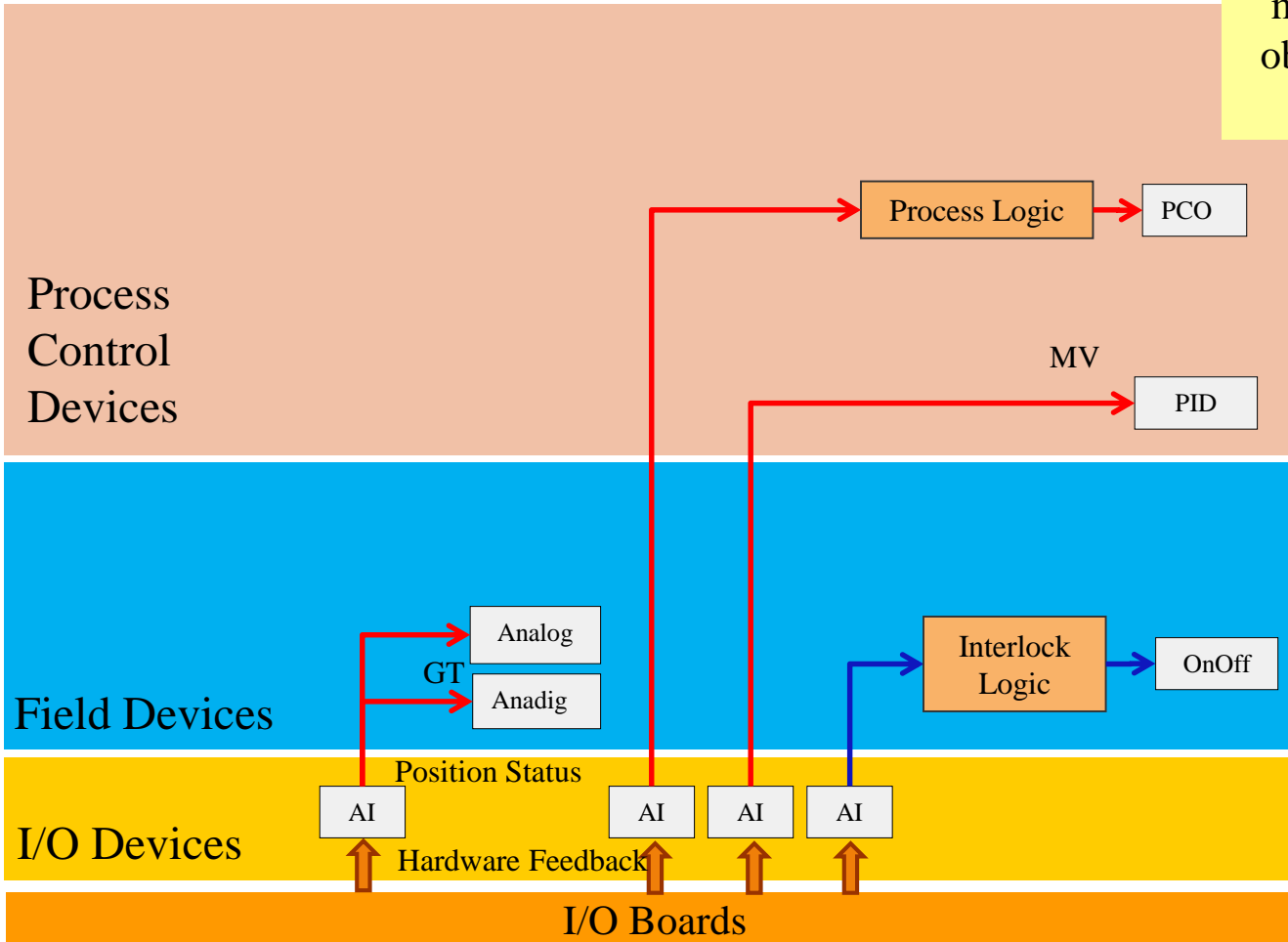
DO OBJECT CONNECTIVITY

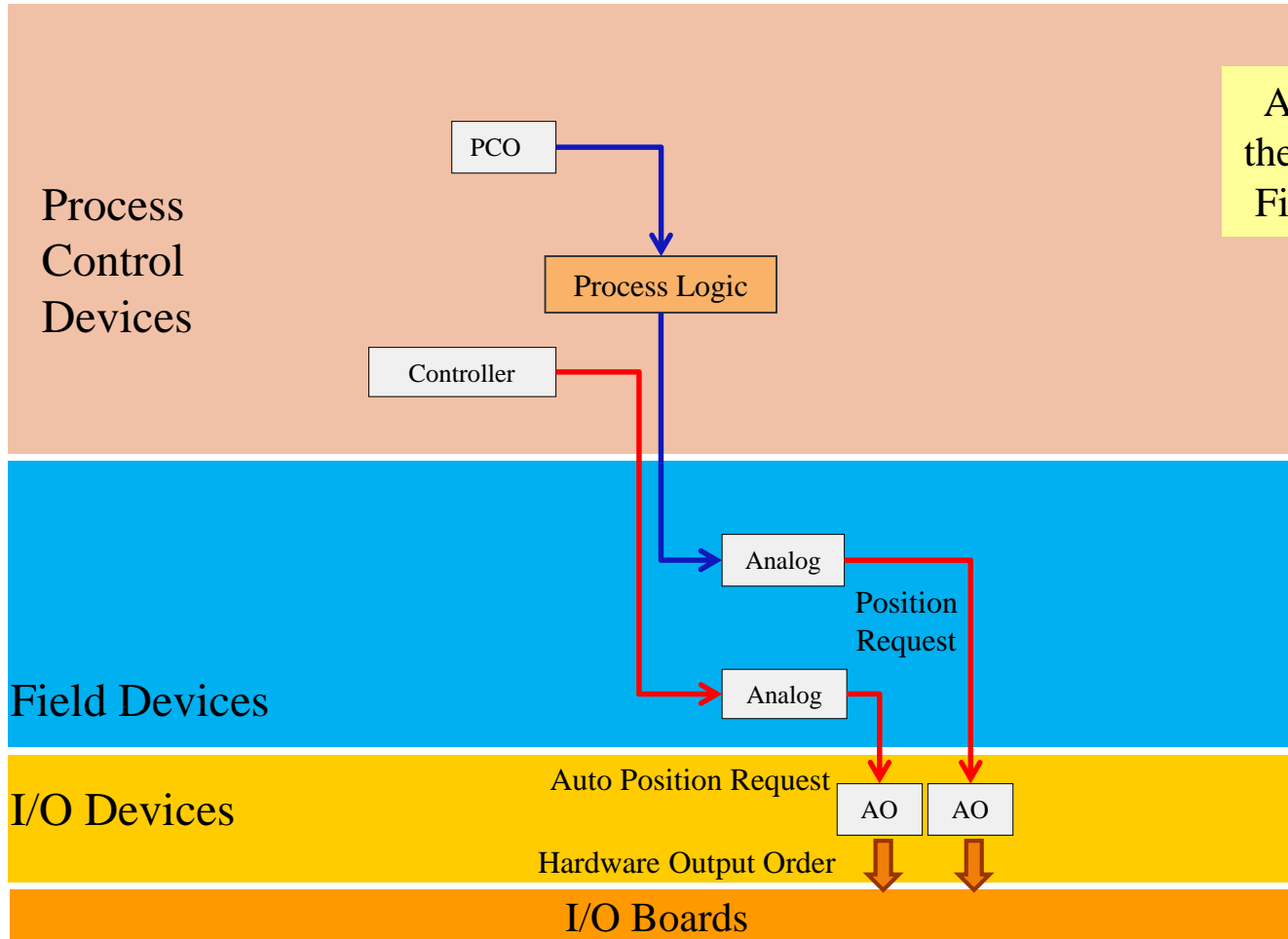


DO Objects receive their orders only from Field Devices (OnOff & Anadig, AnaDO)

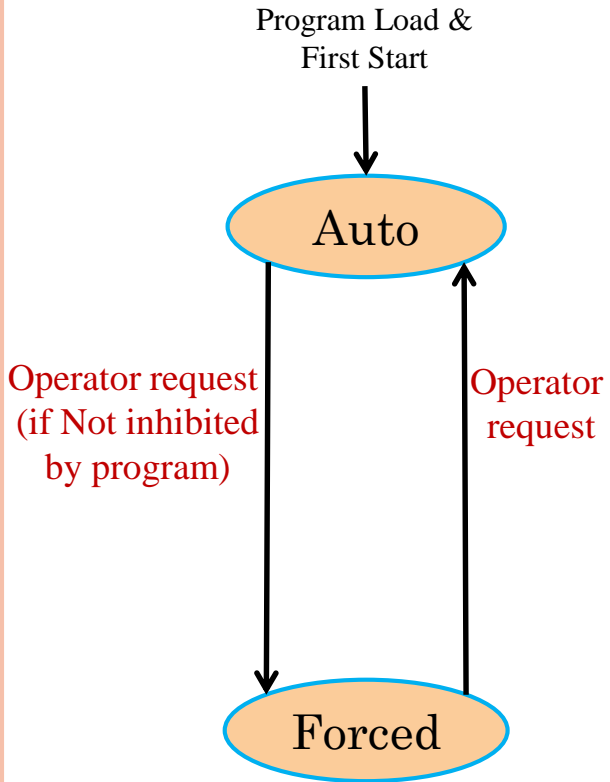
AI OBJECT CONNECTIVITY

Status of AI Objects may be used by all objects anywhere in the program.





AO Objects receive their orders only from Field Device Analog

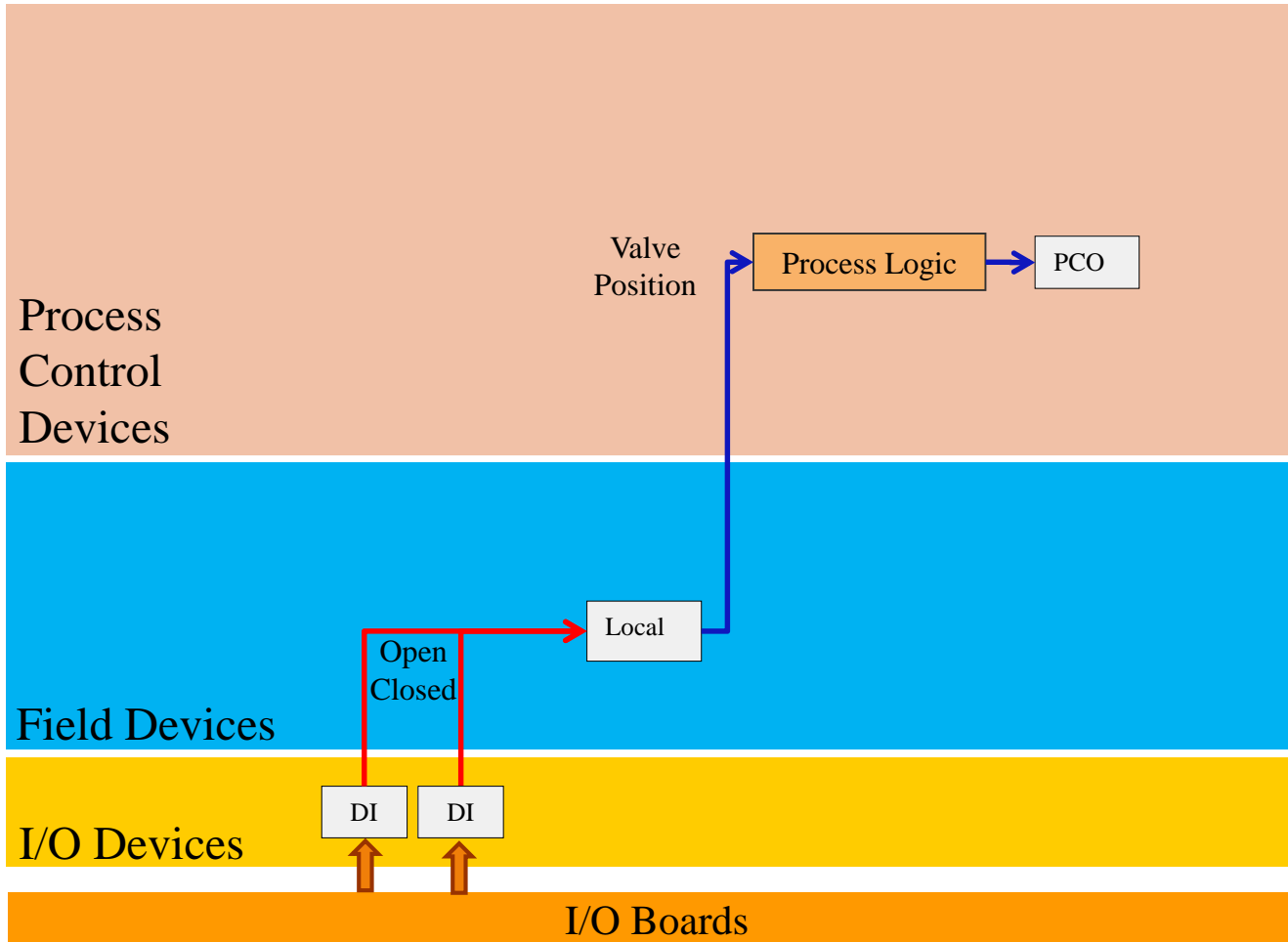


◆ Forced Mode for operators (“**expert**” privilege):

- For AI & DI Objects, when the operator forces the value, the Position Status may become different from the Hardware Feedback.

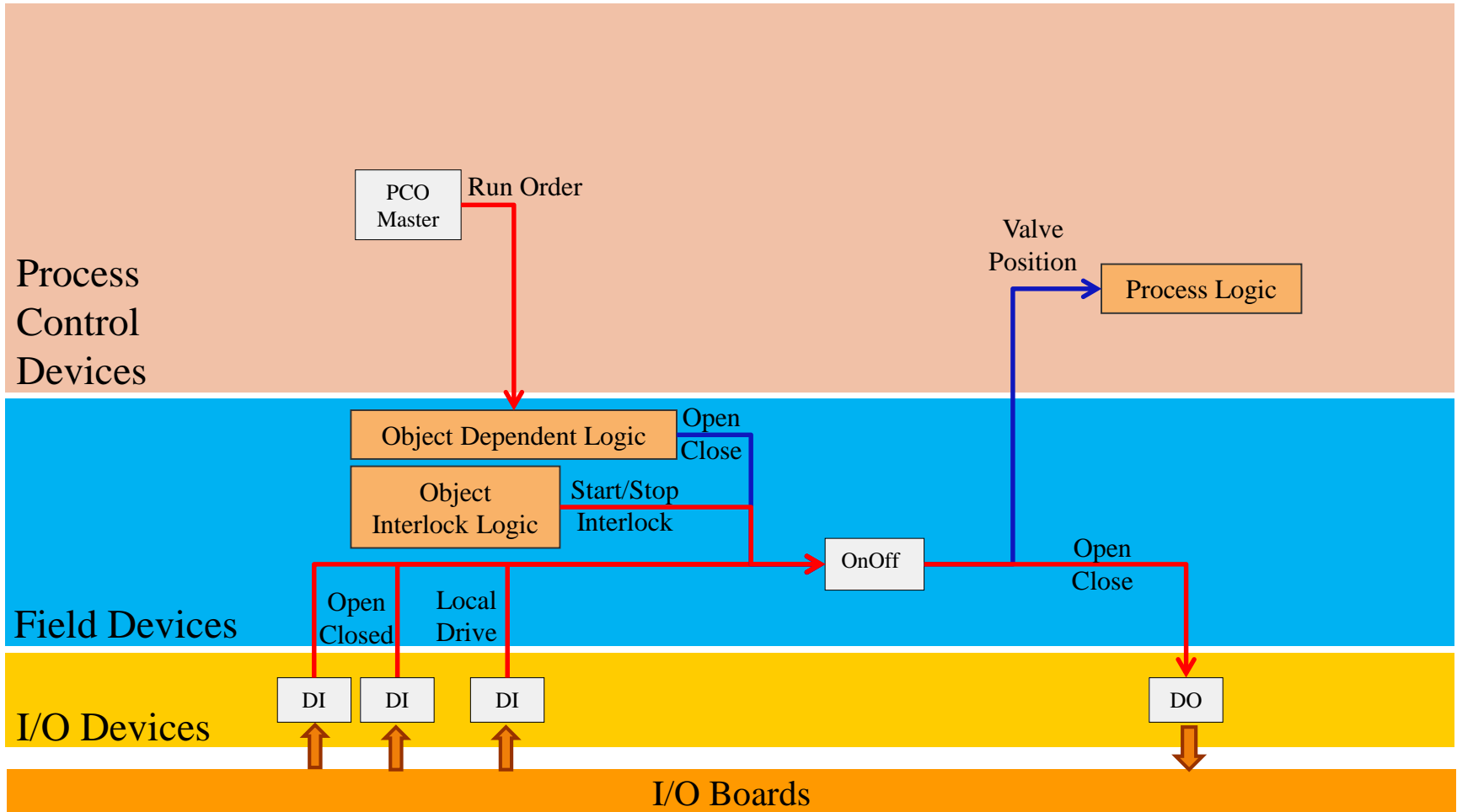
- For DO & AO Objects, when the operator forces the object, the Output Order sent to the process may be different from the Auto Position Request of the process.

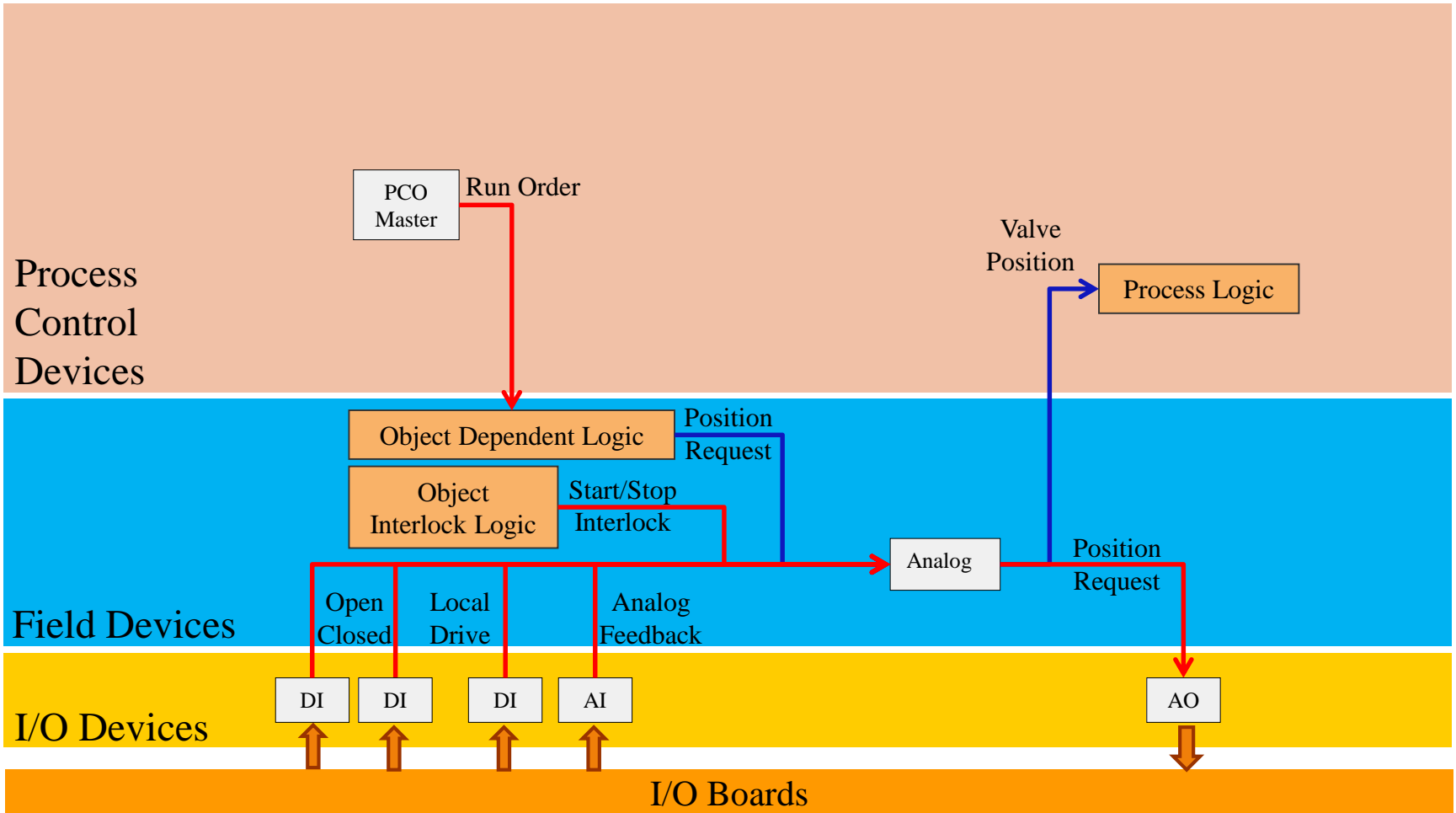
- **Forced Mode may be useful for:**
 - Electrical tests
 - Program simulation
 - Degraded operation!



Status of Local Objects may be used by all objects anywhere in the program.

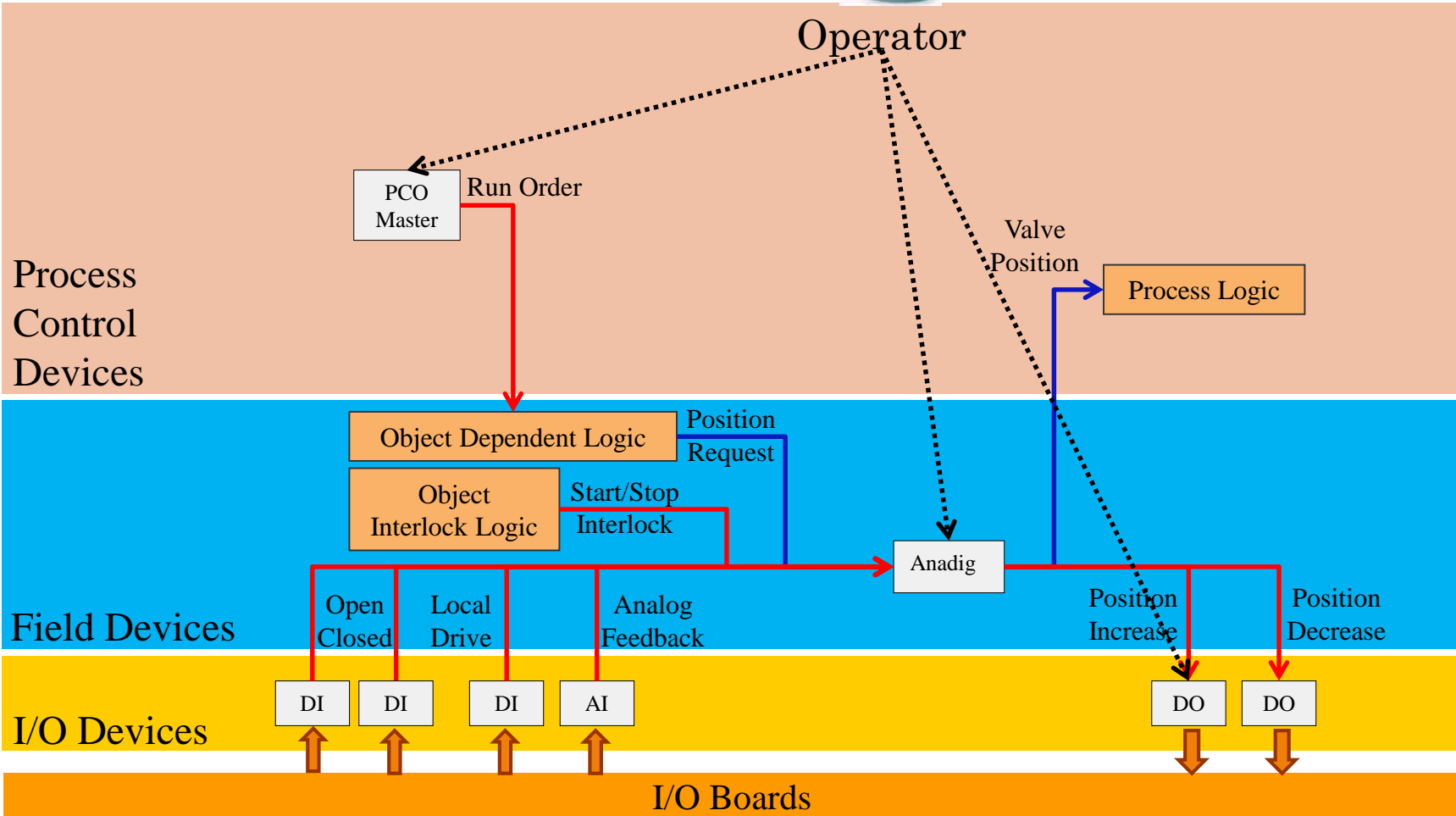
ONOFF OBJECT CONNECTIVITY





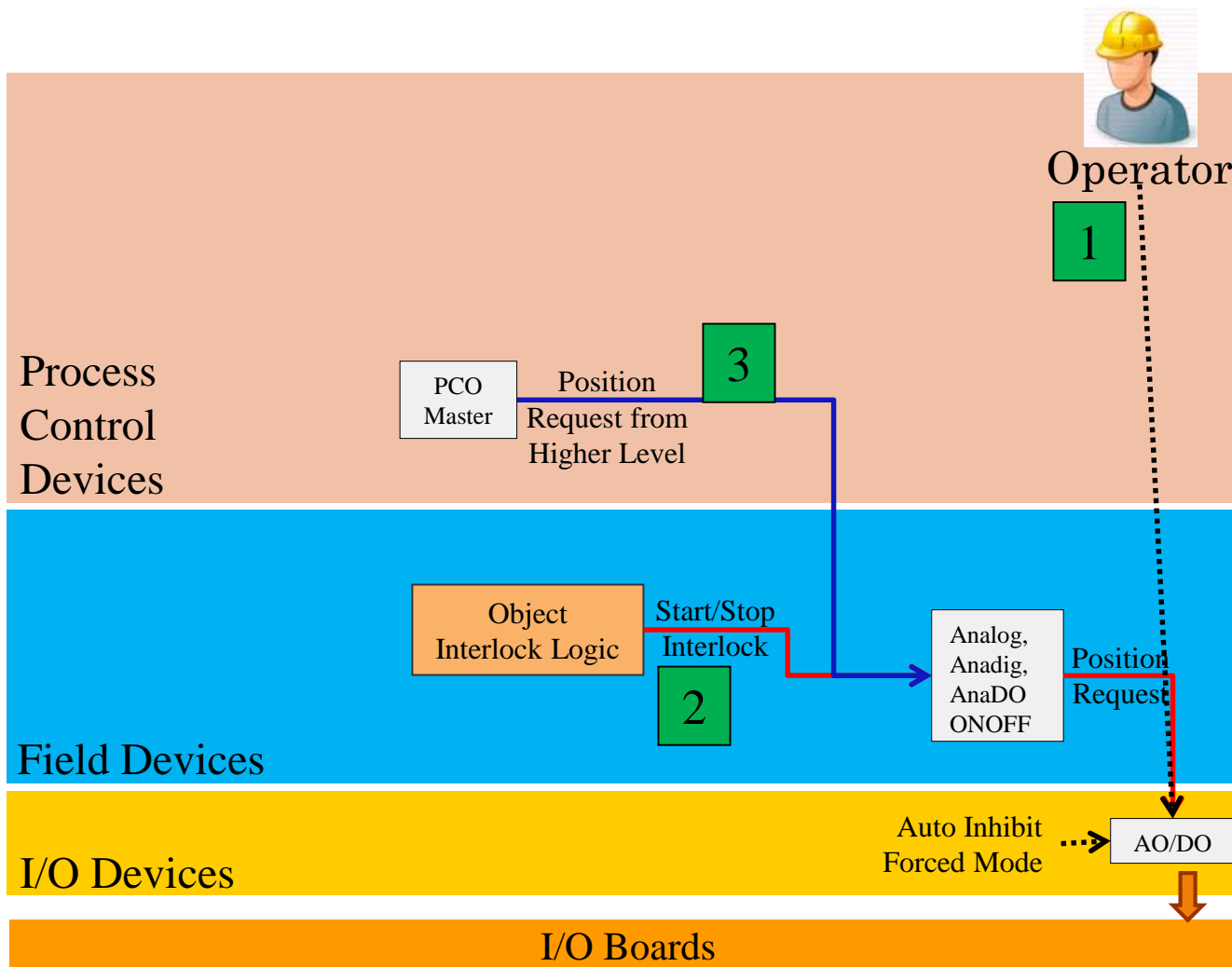


Operator



WHAT ARE THE PRIORITIES?

EXAMPLE WITH AO/DO



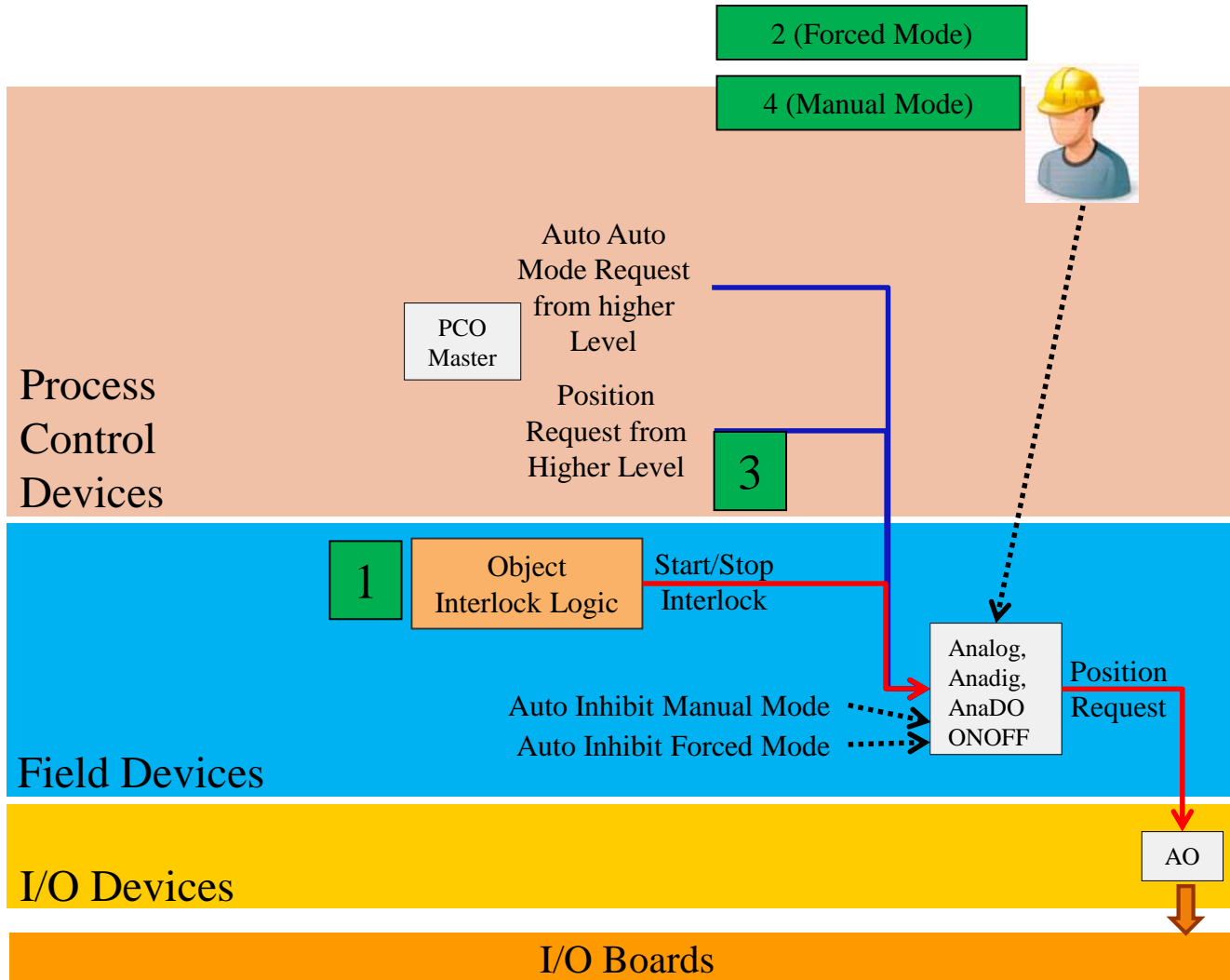
X → priority order

Operator takes Forced Mode and forces the AO/DO Objects to a position.



- This action overrides all the logic (safety) !!
- Possibility to inhibit the forced mode by program parameter

WHAT ARE THE PRIORITIES?

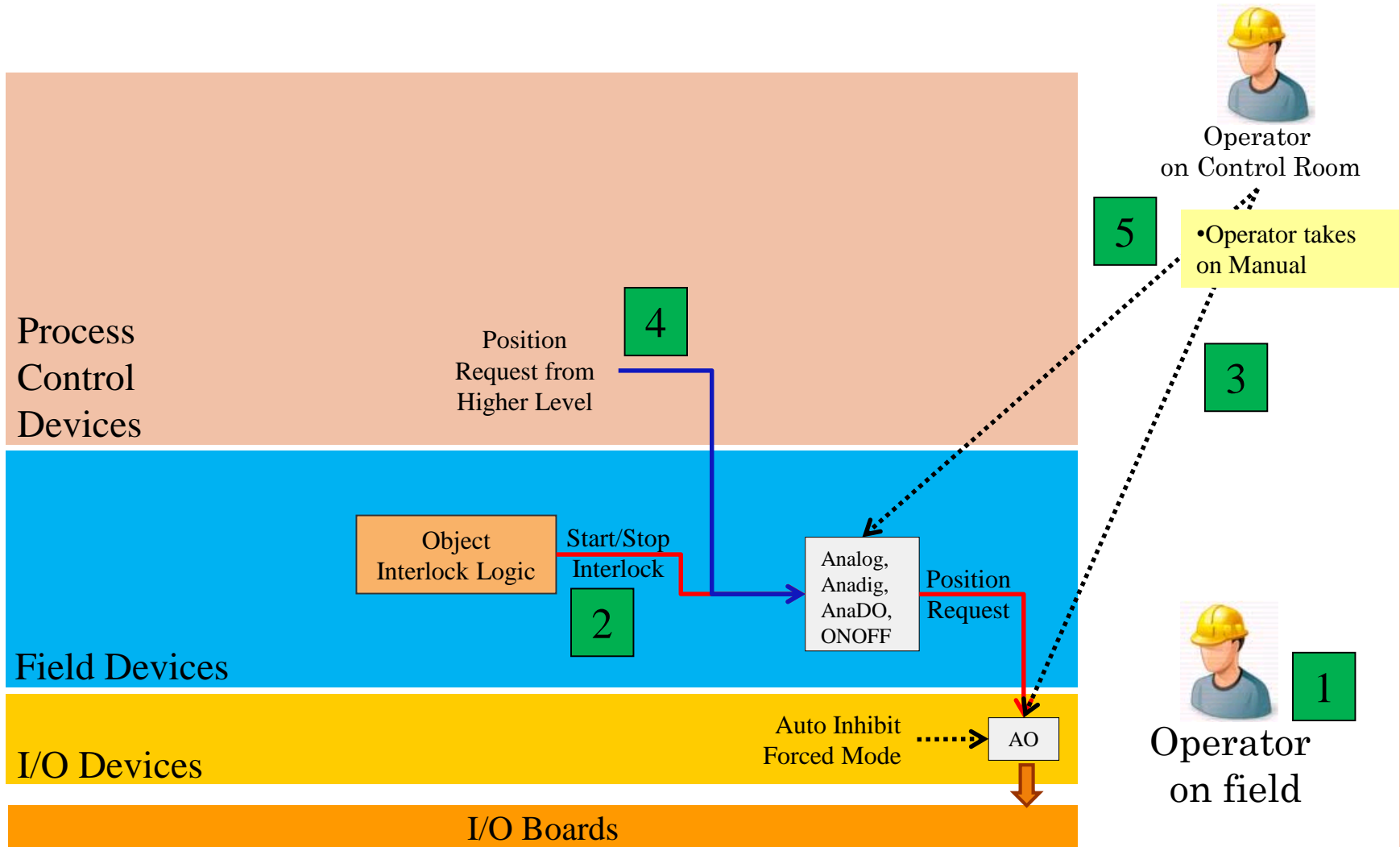


Operator takes Manual or Forced Mode and sets the Field Objects to a position.

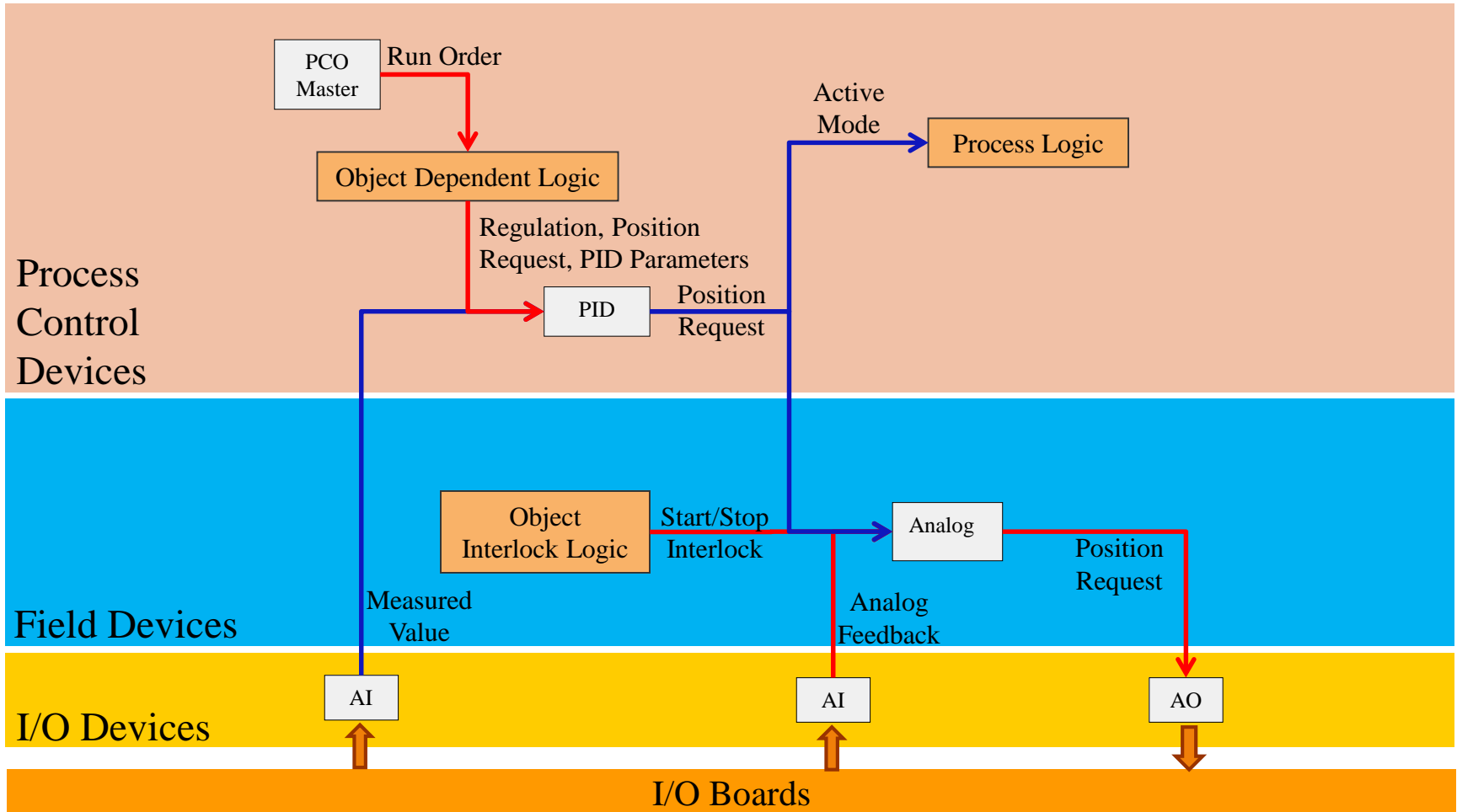


- This action overrides the position request coming from higher level, **but the interlock logic has higher priority.**
- Auto Mode may be requested by Higher level if Manual Mode
- Possibility to inhibit the forced/manual mode by program parameter

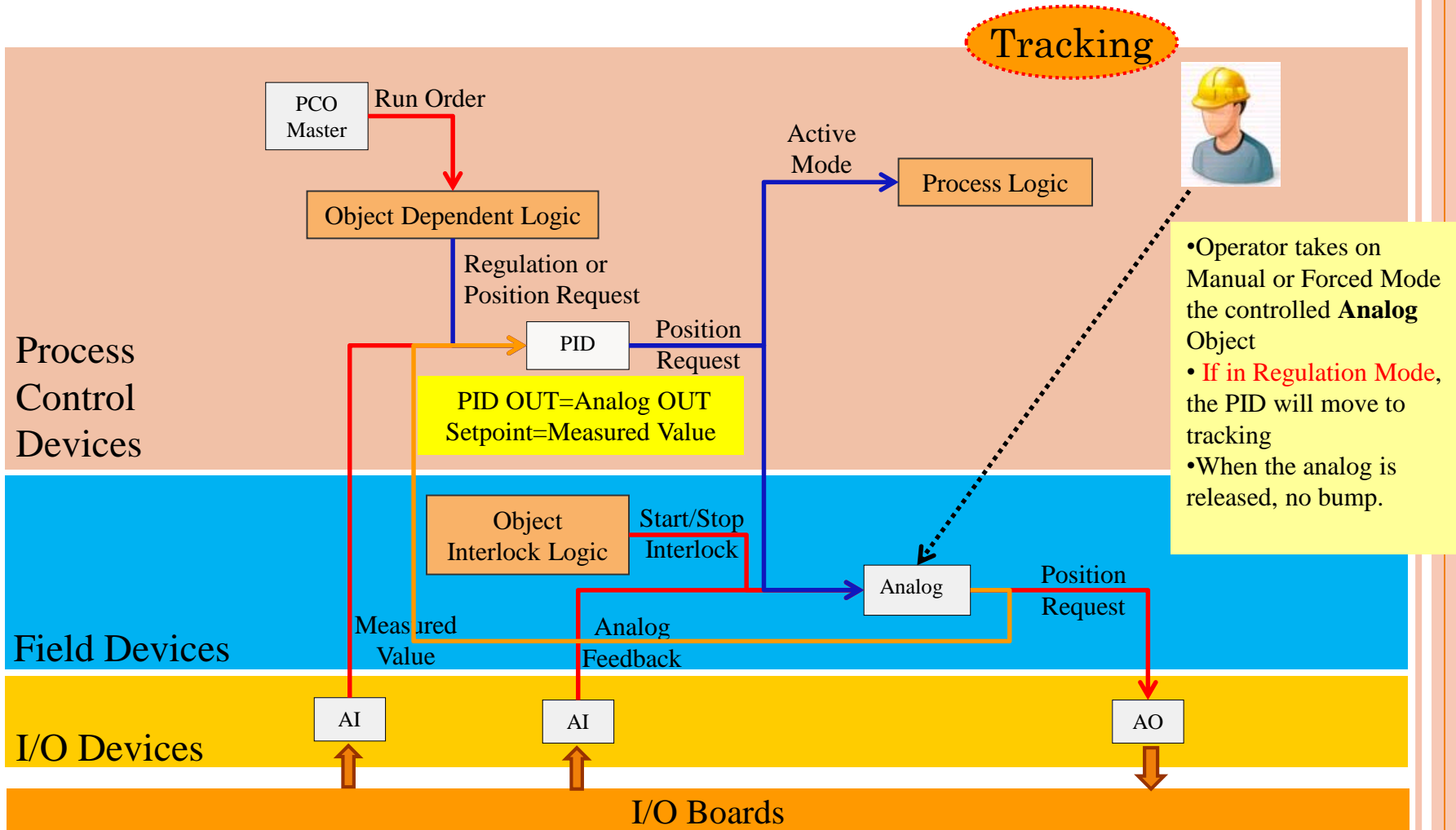
OPERATOR ON FIELD IS STRONGER



CONTROLLER OBJECT CONNECTIVITY



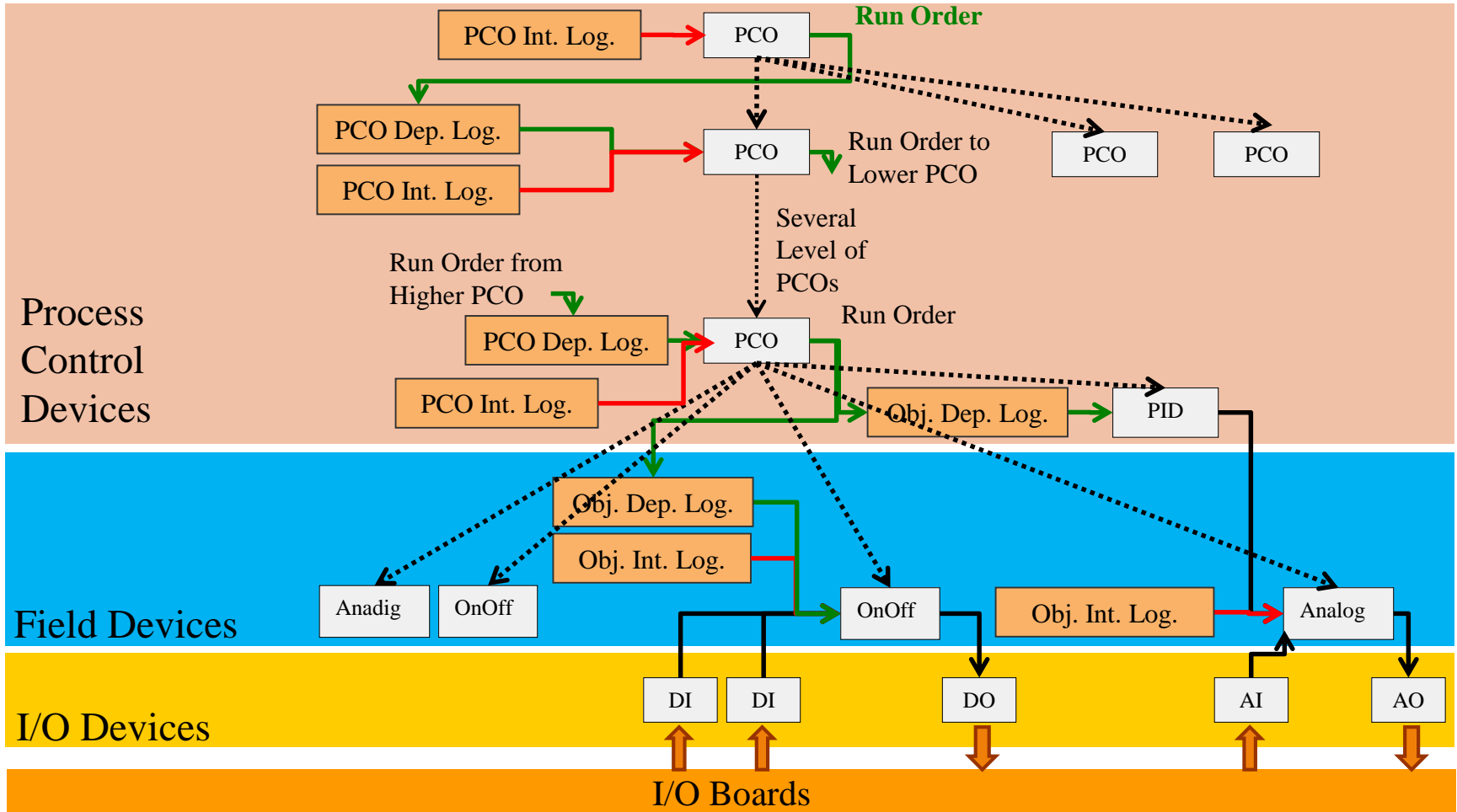
CONTROLLER OBJECT TRACKING



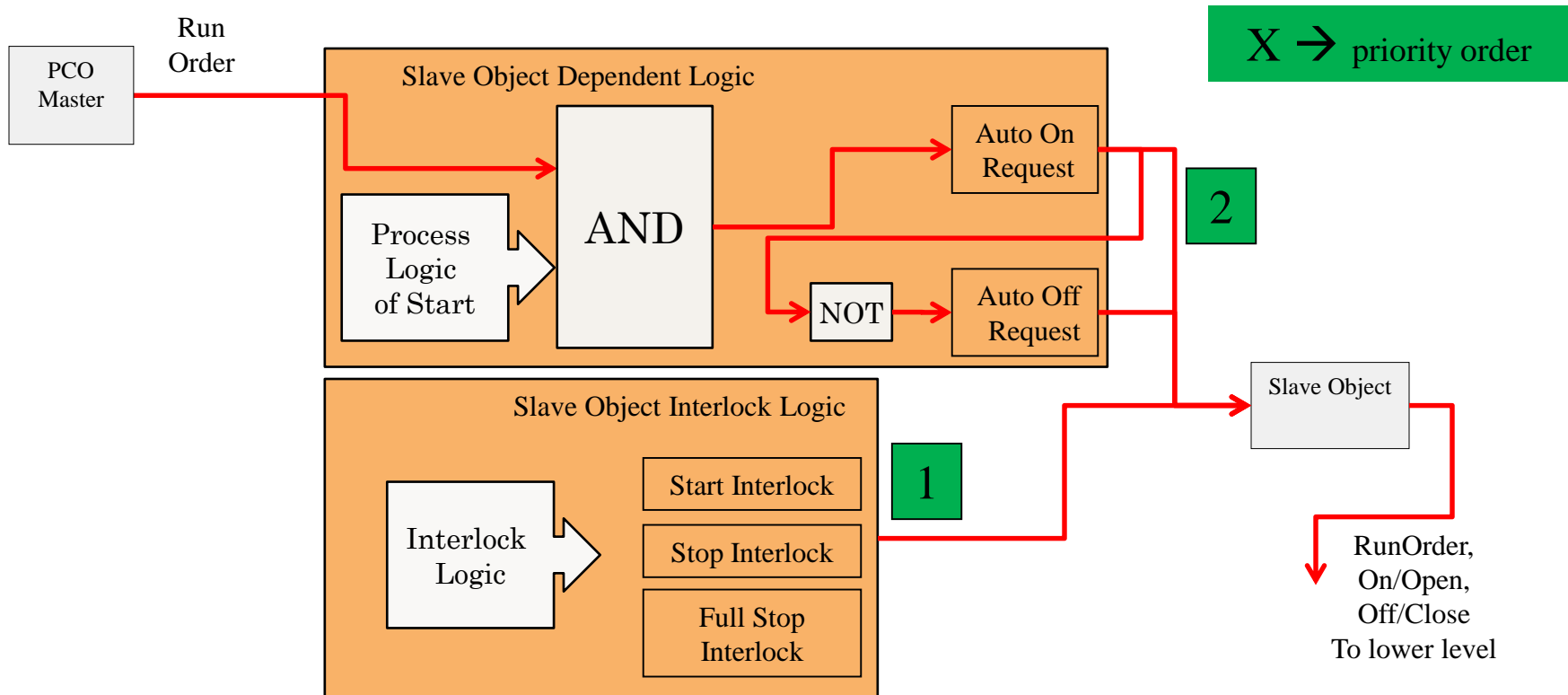
WHAT'S A PCO?

- PCO may be considered as a large OnOff object with some additional functionalities :
 - Full Stop Interlock (as field objects)
 - Block Alarm (Deactivate Interlocks Logics!)
 - Control Stop
 - Options (operation mode)
- PCO is used to represent sets of fields objects that performs a common function:
 - Compressor, Turbines
 - Vacuum System
- PCO is also used to represent sets of other PCOs, this allows to manipulate large process parts:
 - Cold Box
 - Set of compressors

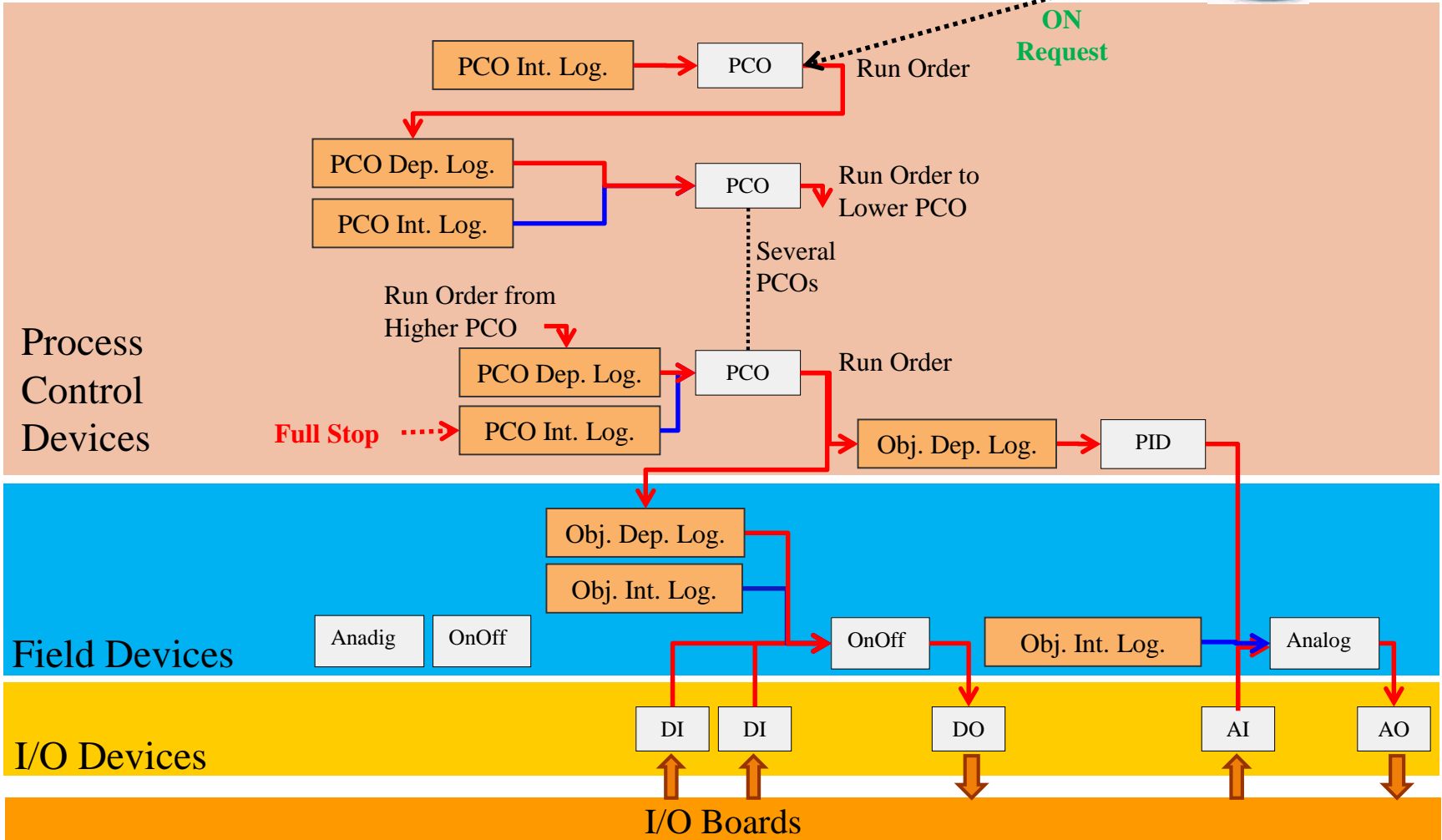
PCO OBJECT CONNECTIVITY



- ◆ Run Order is the Start Request of the PCO to all the dependent objects

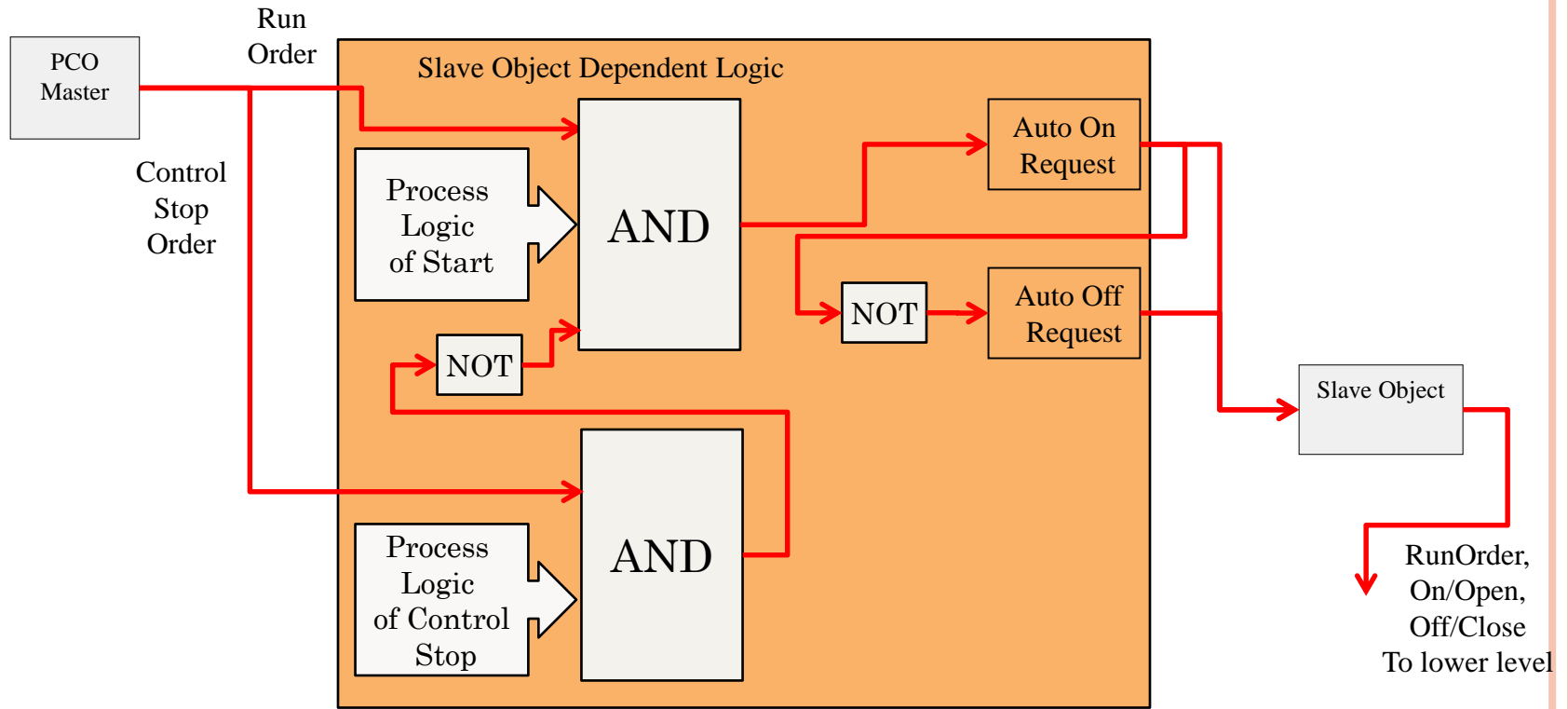


PCO RUN ORDER PROPAGATION

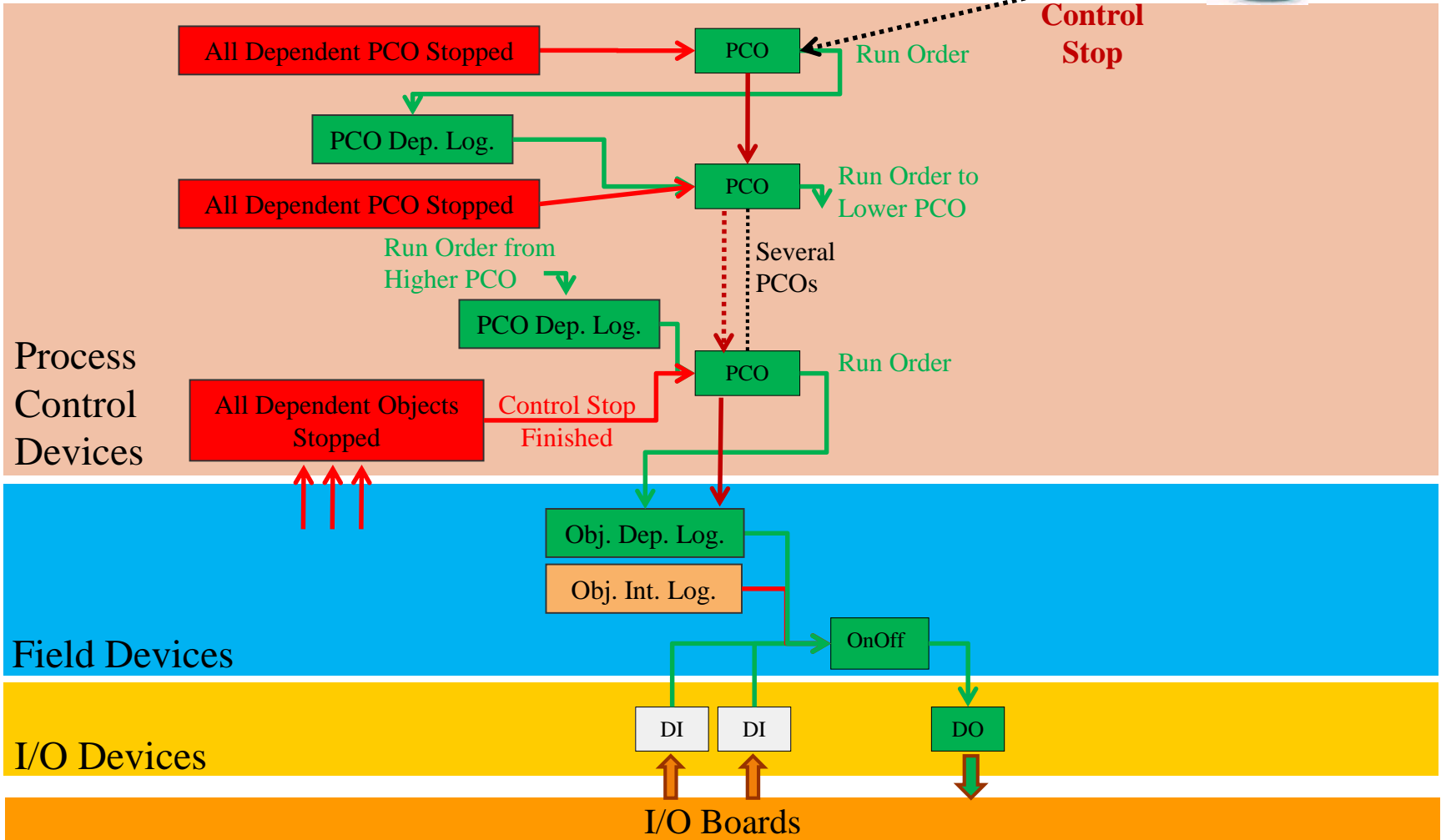


PCO CONTROL STOP ORDER

- The control stop order is used to initiate a sequenced stop
- When a PCO receives a *control stop*, the RunOrder remains **ON** until the Control Stop ends

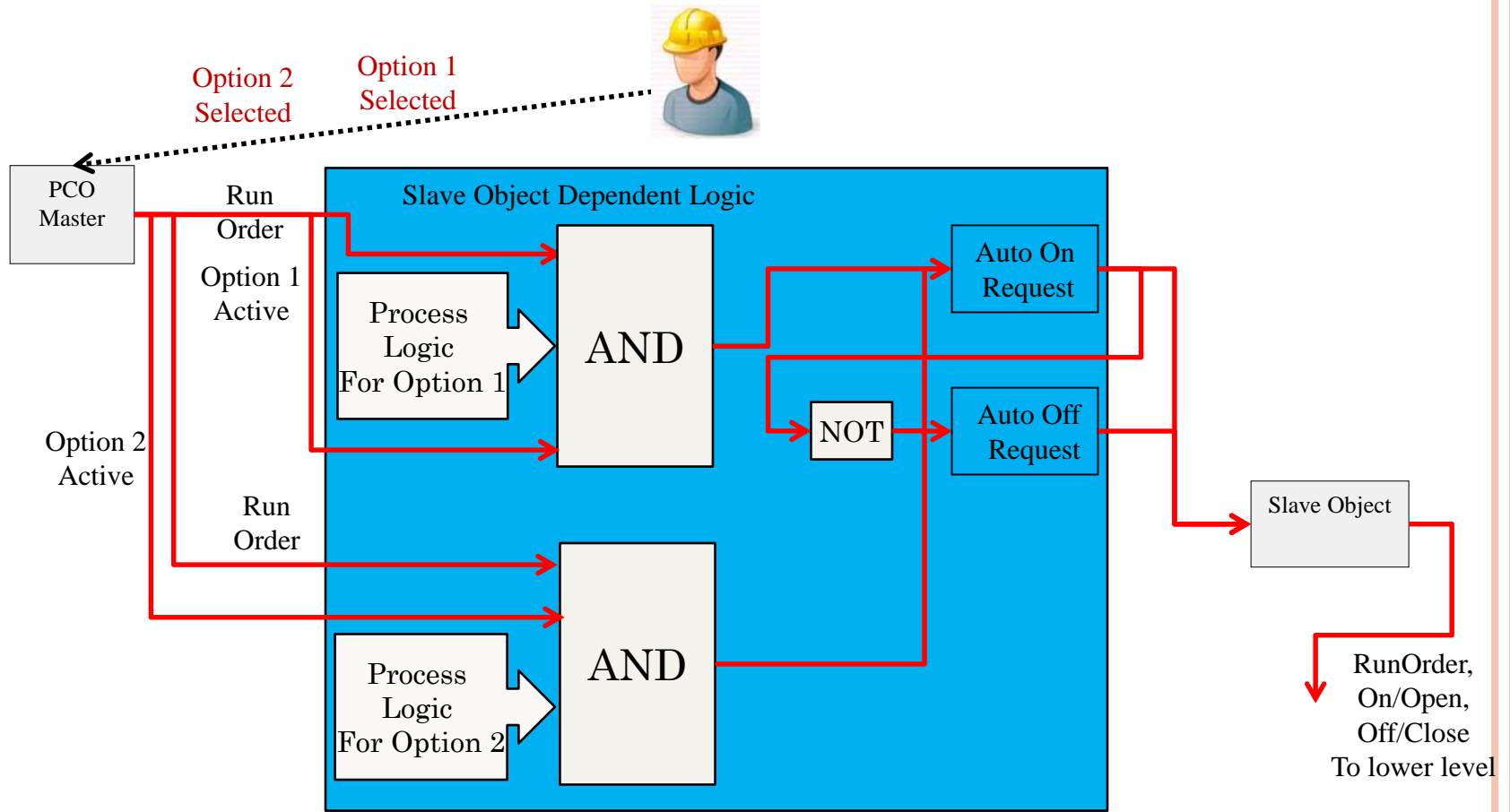


CONTROL STOP ORDER PROPAGATION



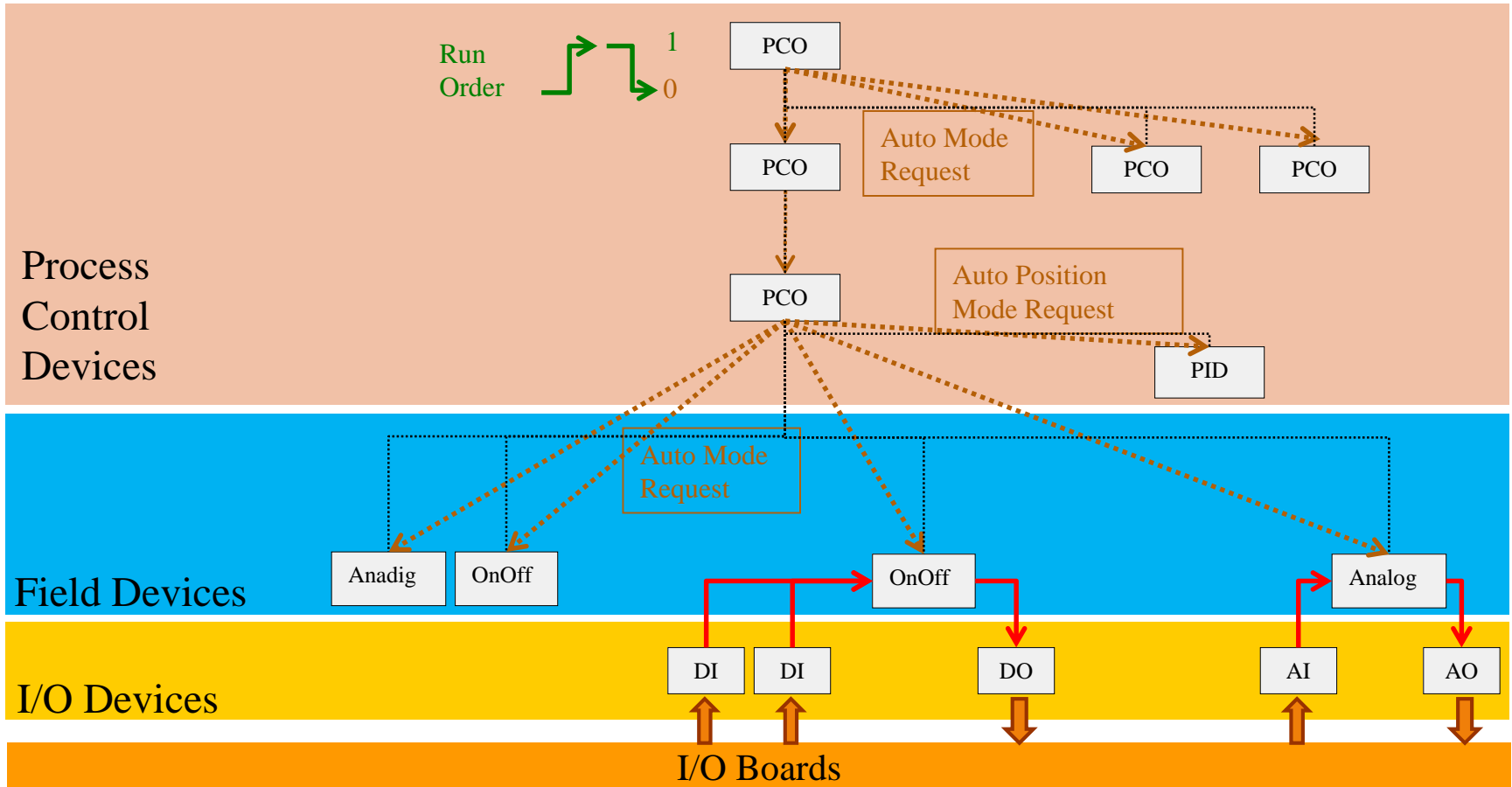
PCO OPTION

- The PCO Option allows the implementation of separate sets of logic. The option is selected by the operator.



AUTO MODE PROPAGATION

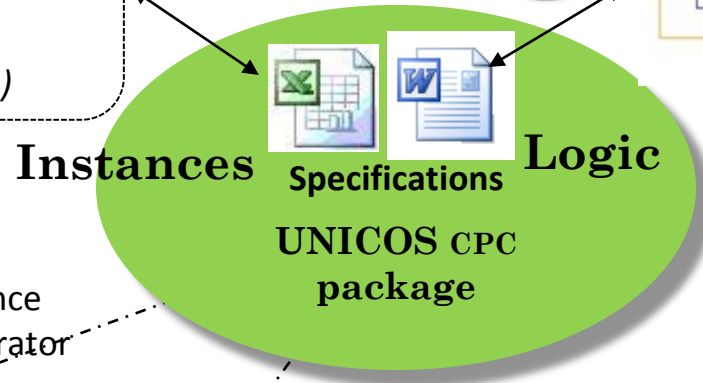
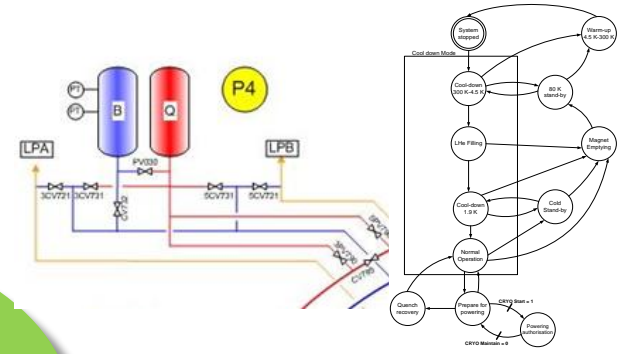
- ◆ When a PCO Object is started or stopped, all dependent objects are requested to Auto Mode



- UNICOS vs. UNICOS-CPC
- Objects main functionality and connectivity
- **From specs to implementation: Overview**
 - Life cycle
 - Generation Tools

UNICOS CPC OVERVIEW

- I/O Channels
- Field Objects (*Valves, Heaters, ...*)
- Process Control Objects (*Compressors, feedbox, ...*)



PLCs, Touch Panels and SCADA Baselines

PLC, Touch Panels and SCADA Instances

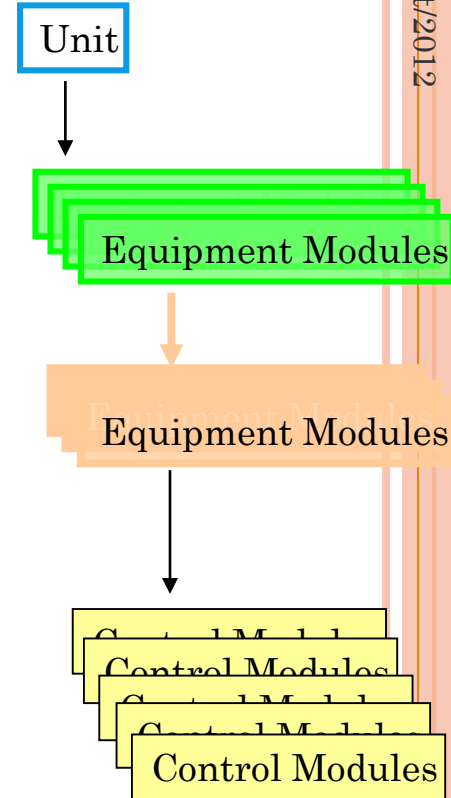
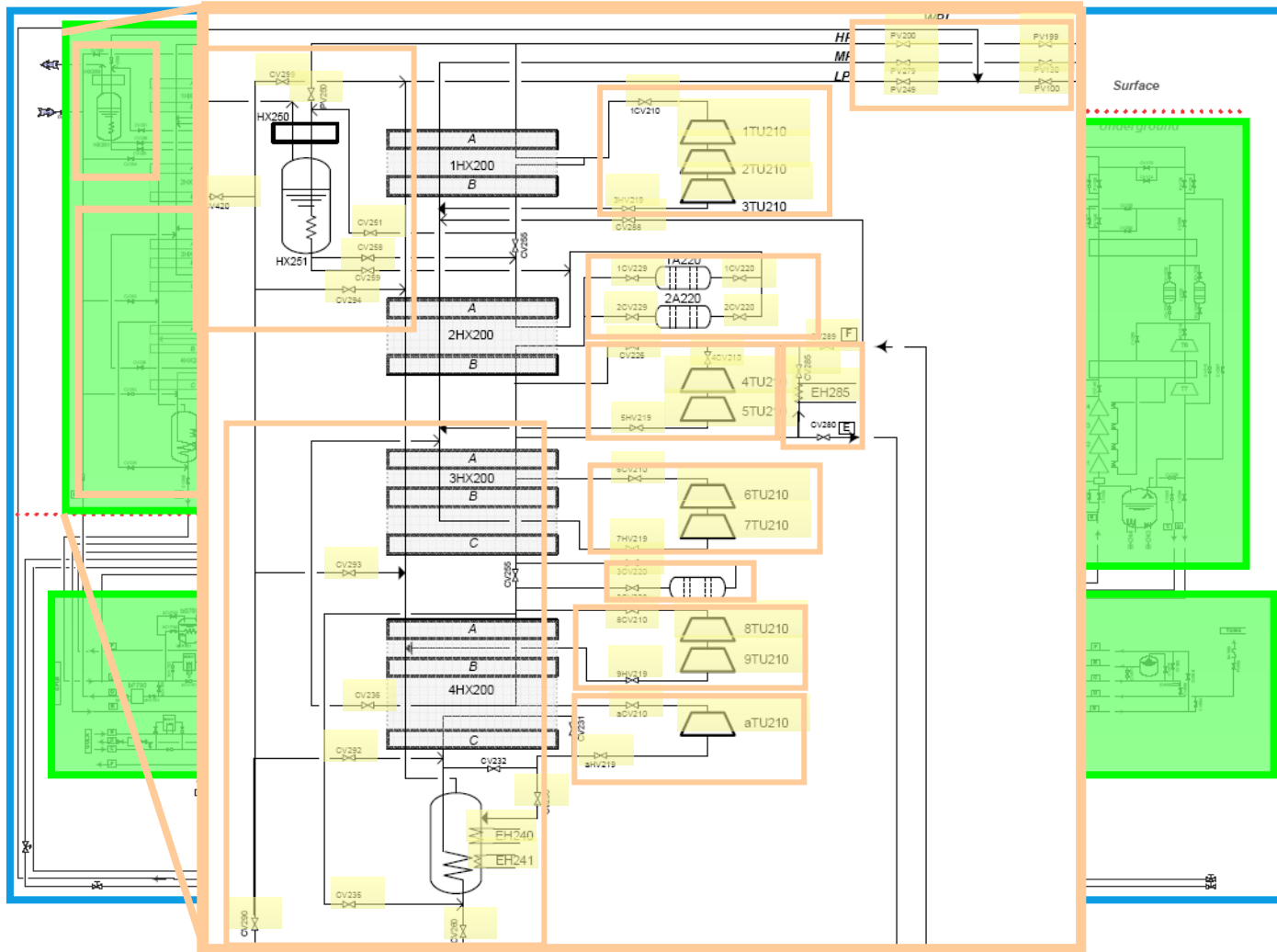
Precise **placeholders** where the control engineer must write the process logic

Simplified **HMI** tool to create process **synoptics (drag & drop)**

Additional services:
 CMW interface
 Long-Term archiving
 LHC alarm system

Diagnostics tools
 System Integrity

- Operators
- Process Engineer
- Control Engineer



IEC 61512-1
Physical model

| DeviceIdentification | DeviceDocumentation | | | FEDeviceIOConfig | | FEDeviceParameters | | | | | |
|----------------------|-----------------------------------------|-------------|--------------------|------------------|------------------|--------------------|-----------|-----------|---------|---------|--------------|
| | Name | Description | Electrical Diagram | Remarks | FE encoding type | InterfaceParam1 | Range Min | Range Max | Raw Min | Raw Max | DeadBand (%) |
| QSDN_4_1TT4001 | Vessel 1- Heater section1-Temp. control | AI1.0 | | | | %IW1.1.0 | 80 | 350 | 0 | 10000 | 0.025 |
| QSDN_4_AI1 | SPARE | AI1.1 | | | | %IW1.1.1 | 0 | 100 | 0 | 10000 | 0.025 |
| QSDN_4_1TT4002 | Vessel 1- Heater section2-Temp. control | AI1.2 | | | | %IW1.1.2 | 80 | 350 | 0 | 10000 | 0.025 |
| QSDN_4_1TT4003 | Vessel 1- Heater section3-Temp. control | AI1.3 | | | | %IW1.1.3 | 80 | 350 | 0 | 10000 | 0.025 |
| QSDN_4_1LE400 | Vessel 1- LN2 Level | AI1.4 | | | | %IW1.1.4 | 0 | 1350 | 0 | 10000 | 0.025 |
| QSDN_4_1PT400 | Vessel 1- LN2 Vessel Pressure | AI1.5 | | | | %IW1.1.5 | 0 | 4.0 | 0 | 10000 | 0.025 |

UNICOS CPC Specs (xls/xml file)

Functional Analysis + Logic specification
(Word templates)

CERN
CH1211 Geneva 23
Switzerland

EDMS NO. **0000000** REV. **1.0** VALIDITY **DRAFT**

REFERENCE **XXXX**

Engineering Department

Date : 2010-11-24

FUNCTIONAL ANALYSIS
UNICOS-CPC (Continuous Process Control)

TEMPLATE FOR
FUNCTIONAL ANALYSIS

[sub title]

Engineering Department REFERENCE XXXX EDMS NO. 0000000 REV. 1.0 VALIDITY DRAFT Page 5 of 16

2. PROCESS DESCRIPTION

2.4 Process decomposition

3.2 Operational States

3.2.4 Actuator operation

Dependent on the following...

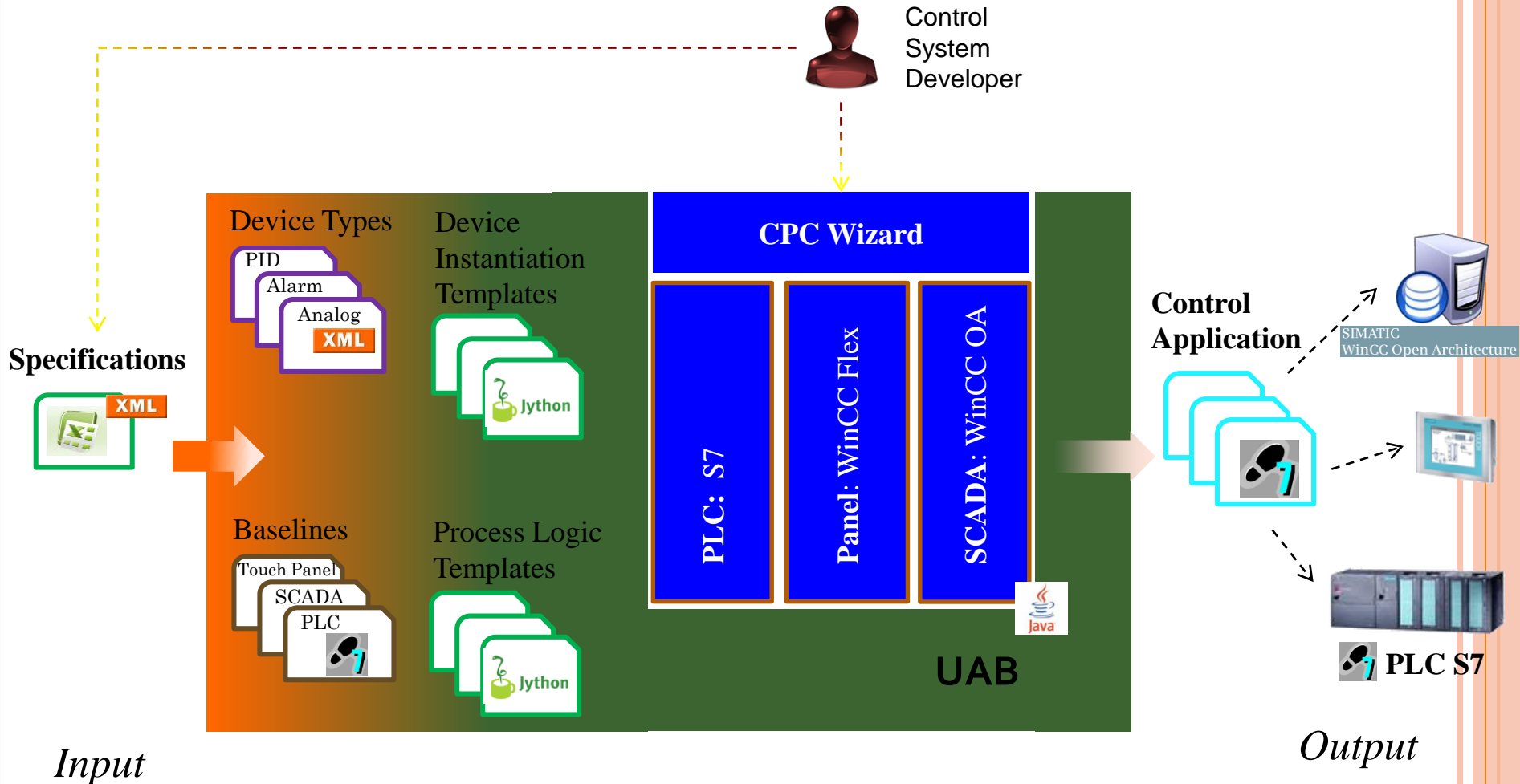
3.5 Unit Alarms

3.5.1 Unit hardware alarms

| Name | Condition | Action* | Message |
|----------|--------------|---------|---------------------------------|
| DNCT_FS1 | ESSCOK Off | FS | equipment emergency s |
| DNCT_FS2 | 24VPwOn. Off | FS | Presence 24VDC Po |
| DNCT_FS3 | 24VIOOn | FS | Presence 24VDC I |
| DNCT_FS4 | 20Q6. Off | FS | Circuit breaker 24VDC for emerg |
| DNCT_FS5 | 26Q2. Off | FS | Circuit breaker 24VDC D |
| DNCT_FS6 | 26Q1. Off | FS | Circuit breaker 24VDC |

- Flexibility & Scalability
 - Improve Templates edition
 - Creation of new Objects
 - Introduce new platforms (e.g. Labview) and/or UNICOS components (e.g. CIET)
- Performance
 - Large size user application
- User friendly
 - Graphical interface based on a Wizard
- Versioning management

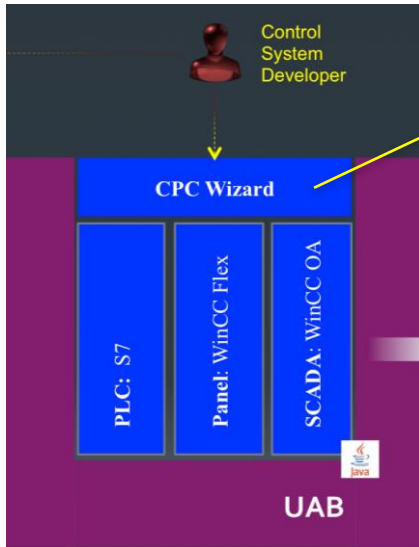
UAB: UNICOS APPLICATION BUILDER



Input

Output

WORKFLOW BASED ON WIZARDS



UAB CPC-Wizard v1.3.2-beta-02

CPC-Wizard: test - test v1.0

Unity Logic Generator

Resources: 1.3.2-beta-02

General Data

Templates Folder: C:\temp\wizard\Release\Schneider\Resources\UnityLogicGenerator\ Open

User Templates Folder: C:\temp\wizard\Release\Schneider\Resources\UnityLogicGenerator\ Open

Output Folder: C:\temp\wizard\Release\Schneider\Output\UnityLogicGenerator Open

Output File: C:\temp\wizard\Release\Schneider\Output\UnityLogicGenerator\plc_ Open

Process Semantic Rules: Generation Language: ST

Import and Generate

| Master | Section | Type | Master | Logic File |
|------------------|------------------|----------------------|------------------|-----------------------|
| DEMON_1_DemonPCO | DEMON_1_Demon... | Interlock Logic | DEMON_1_Demon... | SchLogic_IL_Stand... |
| DEMON_1_PCO3 | DEMON_1_Demon... | Configuration Logic | DEMON_1_Demon... | SchLogic_CL_Stan... |
| DEMON_1_PCO1 | DEMON_1_Demon... | Basic Logic | DEMON_1_Demon... | SchLogic_BL_Stan... |
| DEMON_1_PCO2 | DEMON_1_Demon... | Instantiation | DEMON_1_Demon... | SchLogic_INST_St... |
| | DEMON_1_Demon... | Global Logic | DEMON_1_Demon... | SchLogic_GL_Stan... |
| | DEMON_1_Demon... | Transition Logic | DEMON_1_Demon... | SchLogic_TL_Stan... |
| | DEMON_1_Demon... | Sequencer Logic | DEMON_1_Demon... | SchLogic_SL_Stan... |
| | DEMON_1_Demon... | Common Depend... | DEMON_1_Demon... | SchLogic_CDOL_St... |
| | DEMON_1_A1_DL | Analog | DEMON_1_Demon... | SchLogic_Analog_... |
| | DEMON_1_A5_DL | Analog | DEMON_1_Demon... | SchLogic_Analog_... |
| | DEMON_1_AD1_DL | AnalogDigital | DEMON_1_Demon... | SchLogic_AnalogDi... |
| | DEMON_1_Ctrl1_DL | Controller | DEMON_1_Demon... | SchLogic_Controlle... |
| | DEMON_1_PCO3_DL | ProcessControlObject | DEMON_1_Demon... | SchLogic_ProcessC... |
| | DEMON_1_OO4_DL | OnOff | DEMON_1_Demon... | SchLogic_OnOff_S... |

Filter: Interlock Logic, Configuration Logic, Basic Logic, Instantiation, Global Logic, Transition Logic

Generation Status

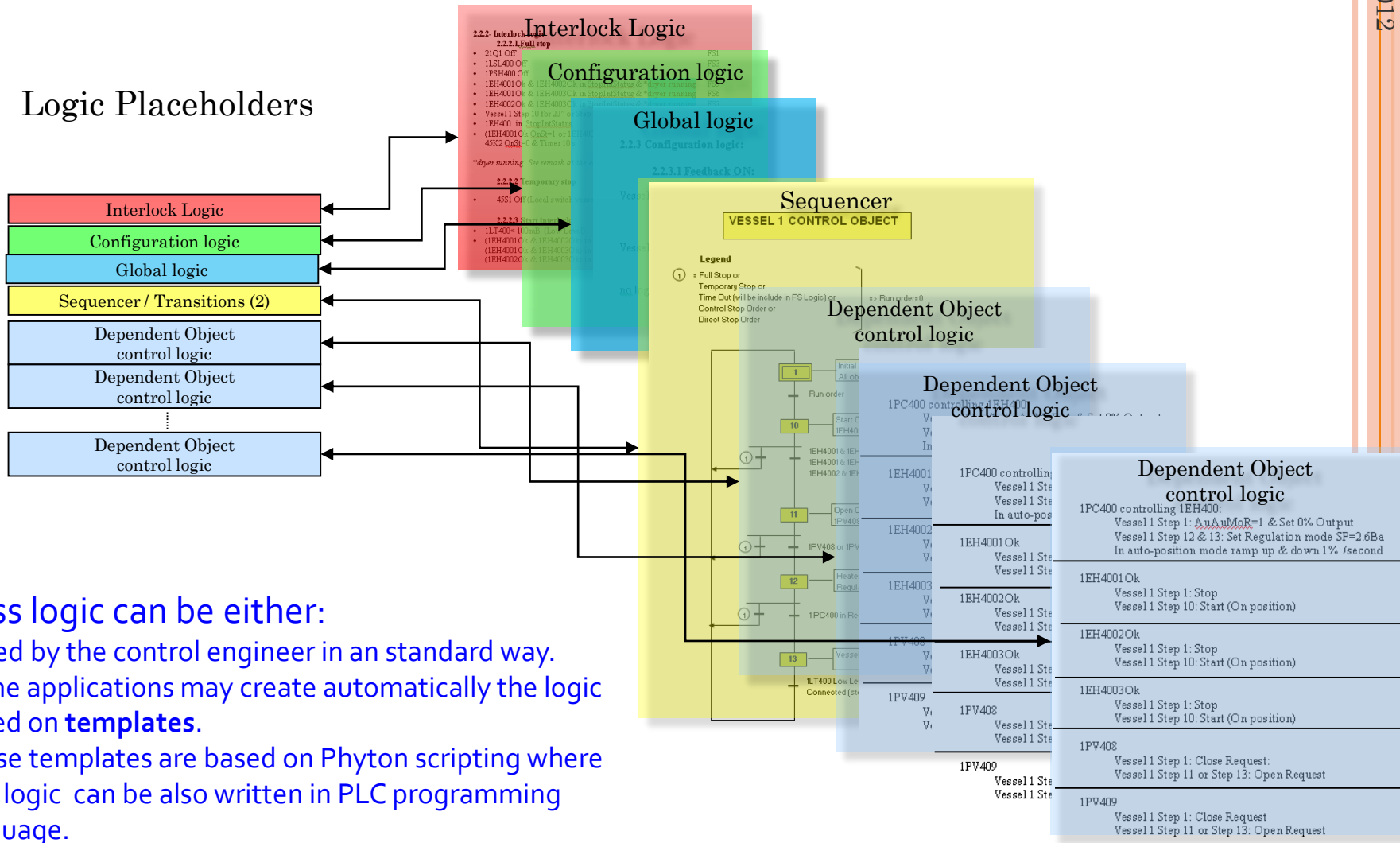
Instance Generator Logic Generator

WinCC OA Gener... WinCC Flex Gene...

Back Generate Exit

For Each PCO the process engineers supply the logic associated to each PCO in a template document (WORD)

Logic Placeholders



Process logic can be either:

- coded by the control engineer in an standard way.

- Some applications may create automatically the logic based on **templates**.

- These templates are based on Phyton scripting where PLC logic can be also written in PLC programming language.

- Possibilities according to repeatability and developers habits/skills
 1. Generation of Default Logic Code
 - User logic written by hand typically inside the PLC software
 - Need to maintain specs + PLC software up to date
 - Generation of logic is performed only once
 2. Generation of User Logic Code
 - Creation of User Logic Templates and specs including logic parameters
 - Need to maintain specs + logic templates up to date
 - Need to generate logic at each logic modification

- Generated automatically:
 - **BL**: Basic Logic : Basic logical links between objects and PCO
 - **CDOL**: Common Dependent Object Logic: Auto request of all dependent objects (alarm ack. included)
- To complete by user:
 - **CL**: Configuration Logic : State calculation of the PCO (On/Off)
 - **IL** : Interlock Logic : Specific logic for PCO interlocks
 - **DL**: Dependent Logic : Object Specific logic depending of the PCO
 - OnOff, Analog, Anadig, AnaDO, Controller and MFC
 - **TL**: Transition Logic : Compute transitions of a grafcet (*optional*)
 - **SL**: Sequencer Logic : grafcet/stepper (*optional*)

IL
CL
BL
INST
CDOL
TL
SL
GL
DL



HMI SYNOPTICS

- Manual intervention (or automatic if known a priori)

SIEMENS SIMATIC MULTI PANEL

Demonstrator SIEMENS

Level controller Temperature controller

+30.10 °C +36.76 cm +39.75 mbar

+28.20 °C +22.40 cm +27.57 mbar

CV01Ana

| Status | Mode | Alarms |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Position:30.0 % | Auto <input type="checkbox"/> Manual <input type="checkbox"/> Forced <input type="checkbox"/> Local <input checked="" type="checkbox"/> Hardware Local <input type="checkbox"/> | Full Stop Int <input type="checkbox"/> Start Interlock <input type="checkbox"/> Temp Stop Int <input type="checkbox"/> Alarm not ack <input type="checkbox"/> Warning <input type="checkbox"/> IOError <input type="checkbox"/> Man <-> Auto <input type="checkbox"/> Position Warning <input type="checkbox"/> |

Auto Manual ON OFF Set value... Ack Alarm

Tank 1 Tank 2 Alarms

TOUCH

Water

5CV211

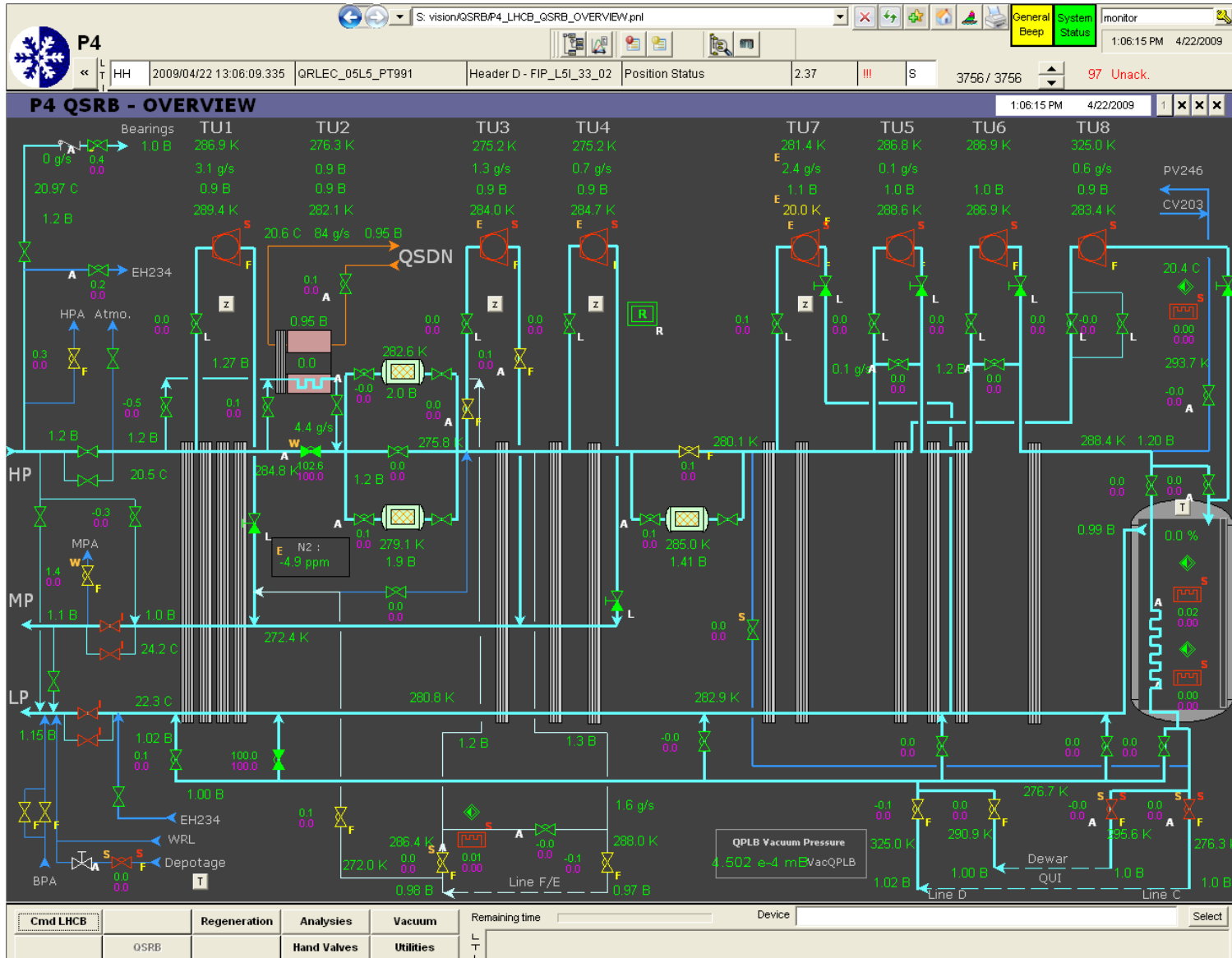
4ST210 4TT219 5ST210 5TT219

UN_STATIC

Synoptic design

- by drag & drop (manual operation)
- Automatically created (xml)

UNICOS CPC HMI



Session 0: From specs to implementation

GENERAL FEATURES: HMI COLORS

- Reduced set of colors

| | Faceplate | Widget |
|----------------------------------|----------------------|--------|
| INFORMATION/ORDERS/STATUS | | |
| Status (status/modes) | Green {RGB: 0,255,0} | 56.9 |
| Request (Request/Orders) | Green {RGB: 0,175,0} | 56.9 - |
| Parameters / Information | Black | 4.56 ✓ |
| PROCESS ALARMS | | |
| Alarms | Red | - |
| WARNINGS/FORCED | | |
| Warnings | Orange | 43.1 |
| Forced by operator | Yellow | 45.5 |
| ABNORMAL SITUATION | | |
| Invalid/Old data | Cyan | 56.9 |
| Data not accessible | Violet | ??? |

COMMISSIONING

- Early I/O commissioning : Immediately after the I/O specs generation

The screenshot displays the unicosHMI interface for LHC COLLIMATION. The main window shows a table of 150 device(s) with the following columns: Device Type, Alias, Value, Local Time, State, Inv, F, and S. The table lists various analog input devices (e.g., TCSM_4L3_Bx_TTA, TCSM_4L3_Bx_TTB) with values around 3276.7 C and states indicating 'Auto Mode Status|O Error Warning'. The interface also includes a left-hand device tree, a top status bar with 'System Status' and 'monitor', and a bottom section for 'Remaining time' and 'Device' selection.

UNICOS CPC FEATURES

- **DEVELOPMENT:** Provides the developer with tools to produce rapid control applications . **Automatic generation of code**
- **COMMISSIONING:** Allows early plant commissioning, reducing check tasks to the instruments and electrical level (cabling). Access to devices without SCADA development.
- **OPERATION:**
 - Provides the operator a standard way to interact all the objects
 - Navigation capabilities between panels and trends: WWW browser-like, contextual buttons, pop-up navigation
 - Access Control
- **DIAGNOSTICS:** Tools to diagnose problems (process alarms, Infrastructure: control components,...)
- **MAINTENANCE:** Allows optimized code evolution and maintenance by a reduced development team

ONGOING WORKS

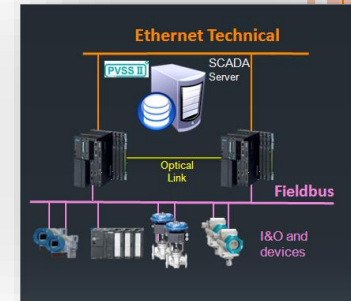
○ New platforms

- UNICOS with **Labview** as light supervisor for small size or lab projects
- **CoDeSys** (Controller Development System)
 - Software tool which turns any PAC into an IEC 61131-3 programmable controller



○ New PLC architectures

- PLC **Redundancy**
 - Increasing availability for critical applications (Siemens & Schneider)
- **Safety** Instrumented Systems (SIS)
 - Integration of the SIS into the UNICOS CPC framework



○ Integration of tuning and control algorithms in UCPC

- **Advanced and modern control**
 - Lowering advanced algorithms to the PLC level (e.g. predictive control...)
- Online controller **tuning**
 - Tools allowing the operator to tune the PIDs online

MORE INFORMATION

- Check out the web page:
 - <http://www.cern.ch/unicos>



About UNICOS

Introduction

UNICOS (UNified Industrial Control System) is a CERN-made framework to develop industrial control applications. It deals with the two upper layers of a classical control system: Supervision and Control. UNICOS proposes a method to design and develop the control application which will run in commercial off-the-shelf products (e.g. SCADA and PLCs). The framework employs terminology and models of the ISA-88 standard for batch control systems.

The goal of UNICOS is to standardize the development of control applications at CERN by:

- Emphasize good practices for both, design and operation, of the continuous process control applications
- Reduce the cost of automating continuous processes (e.g. cooling, HVAC...)
- Optimize life-cycle engineering efforts (e.g. using automatic code generation tools)

