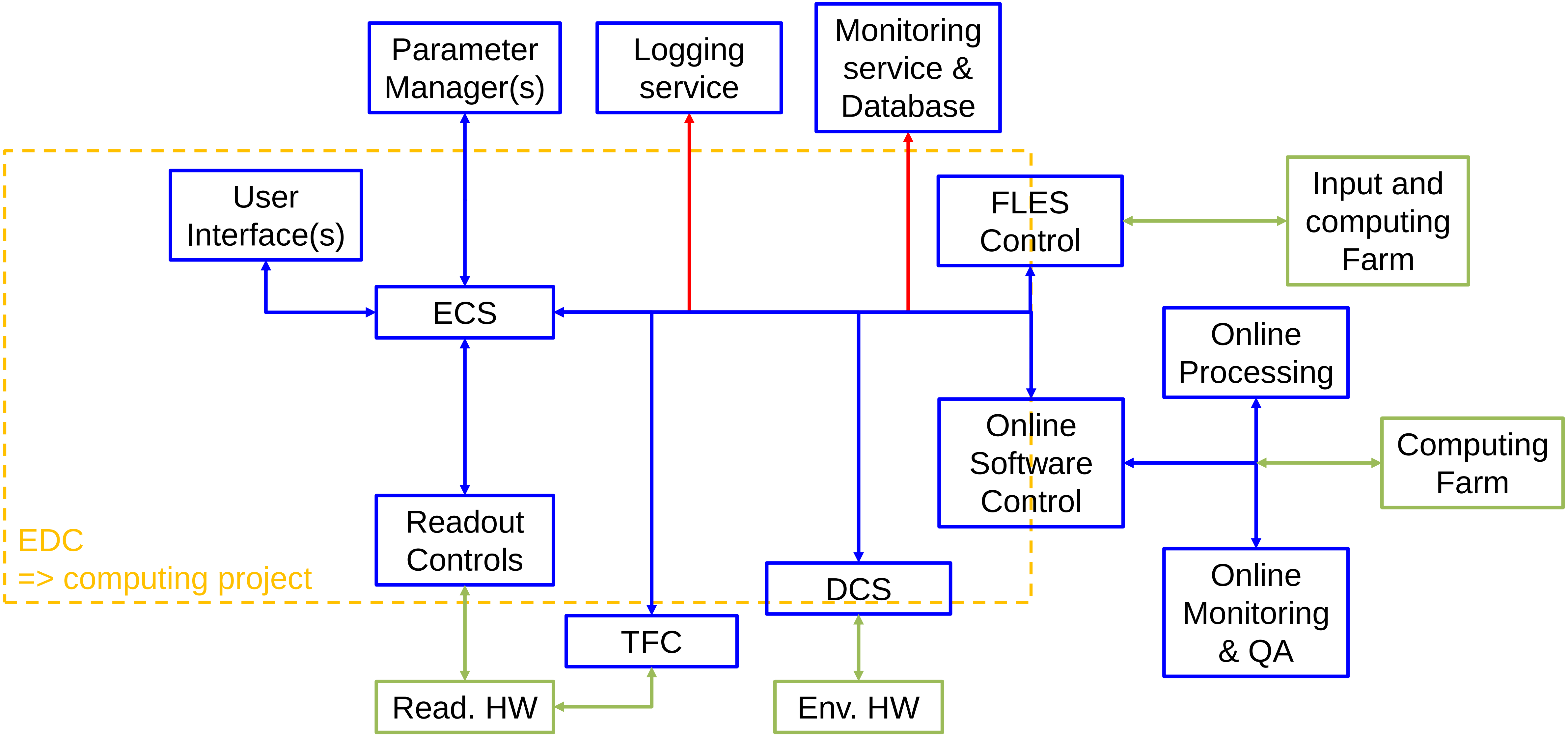
46th CBM Collaboration Meeting
IMP-CAS
Lanzhou, China

EDC: Focus on SCA and DCS interface, Status

- EDC Concepts and acronyms (Many from DCS Workshop 04/2025: ★)
- + Focus on SCA: Mission, concept, requirements, examples
- + Focus on DCS from ECS side: Mission, requirements, interfaces
- Latest developments in EDC project (ECS, SCA, auxiliary systems, …)

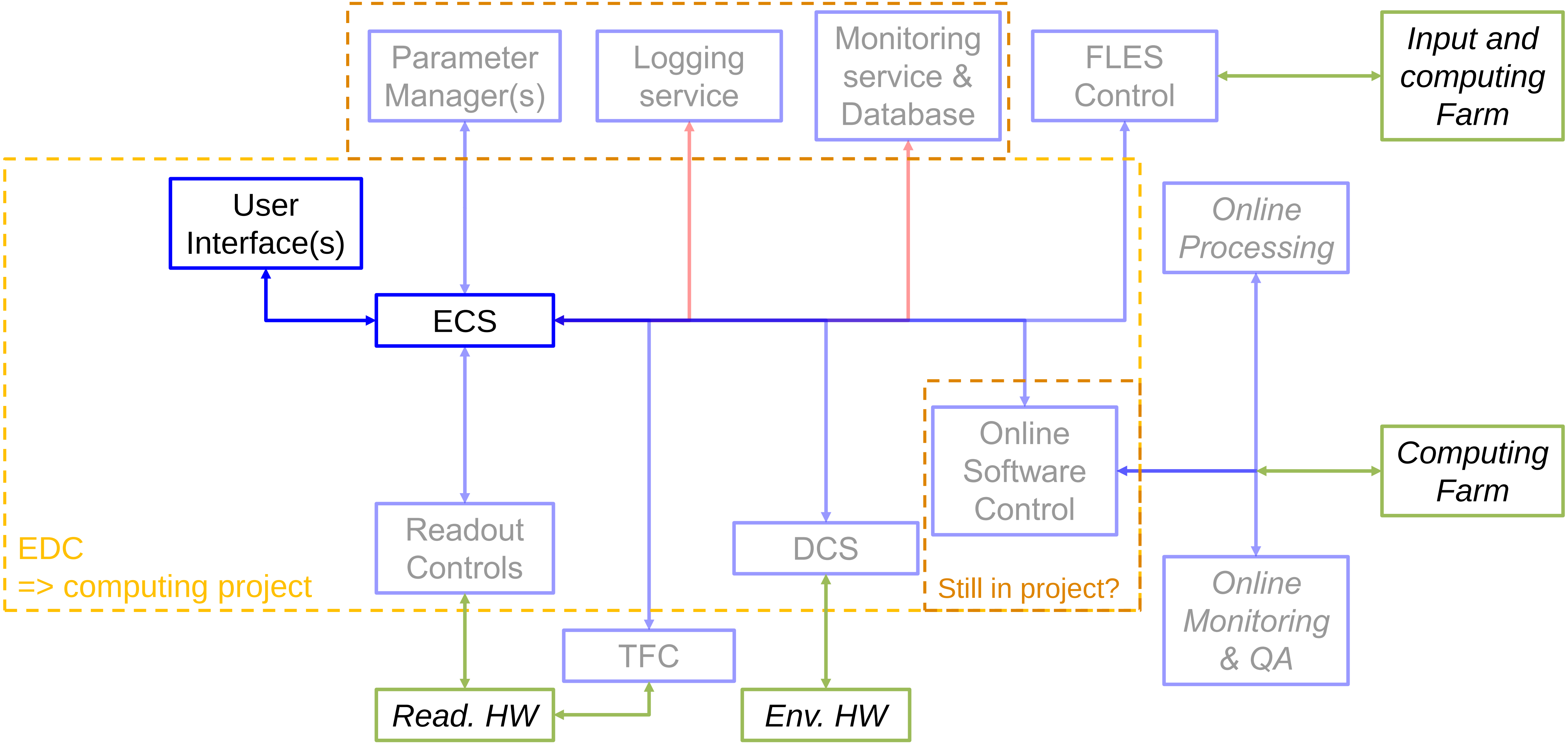
Some slides are meant as support for discussion and may be passed quickly/skipped, especially ★

Controls > EDC > ECS



Concepts and acronyms

ECS?


Parameter Manager(s)
Logging service
Monitoring service & Database
FLES Control
Input and computing Farm
User Interface(s)
ECS
Online Processing
Online Software Control
Computing Farm
EDC
=> computing project
Readout Controls
DCS
Still in project?
Online Monitoring & QA
TFC
Read. HW
Env. HW

ECS

Experiment Control System …. ~ ”the Boss”/”the Architect”

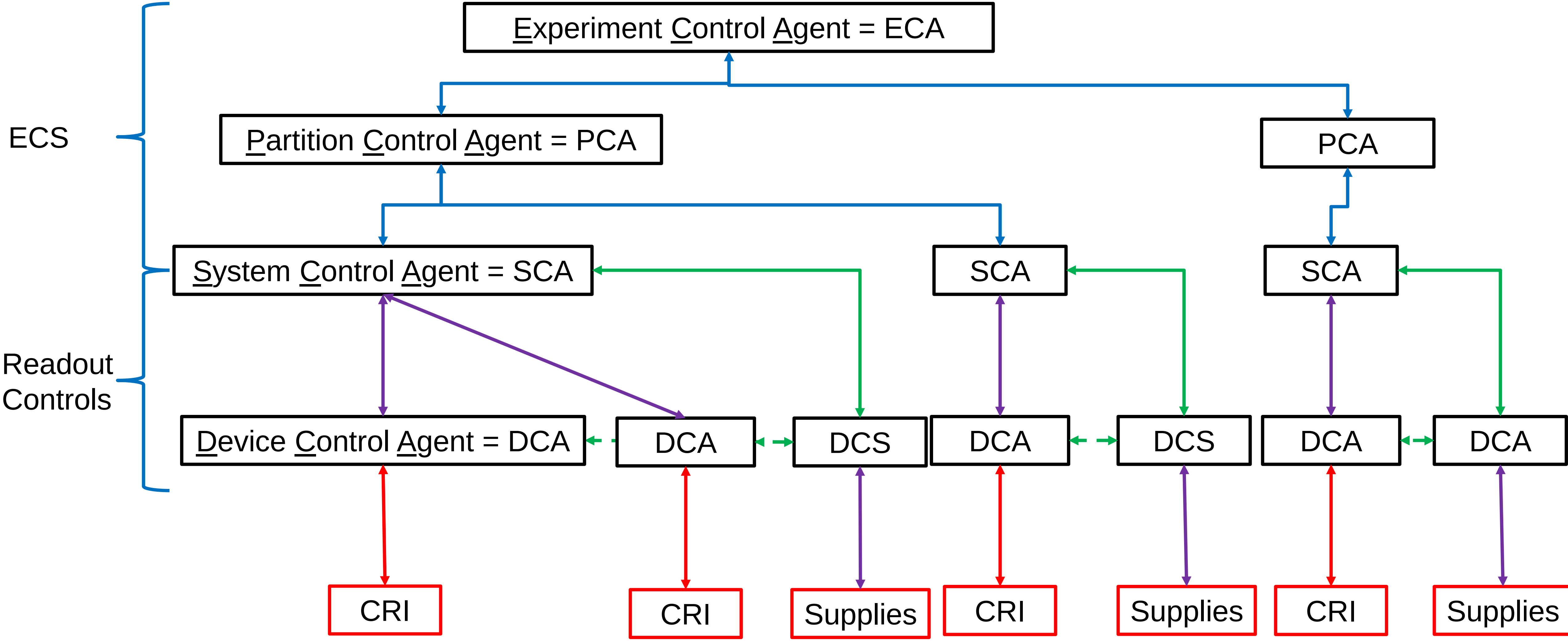
Missions:

- Derive combined state from states of participating systems
- Allow definition and control of state transitions (both automatic and manual ones)
- Apply or detect the configuration of all systems to ensure combined configuration known at all time
- Allow emission and propagation of commands toward the participating systems
- Provide the main interface for the shift crew through its UI(s)

Specifications:

- Global states, common to all systems
- Local states as sub-states of the Global states, specific to given systems
- Configurations internally propagated as sets of tags and sub-tags
- Allow operation of subsets of detectors systems in parallel (Partitions)
- Initial version in Python

Planned ECS Agents (~Layers) and interfaces


Experiment Control Agent = ECA
ECS
Partition Control Agent = PCA
PCA
System Control Agent = SCA
SCA
SCA
Readout
Controls
Device Control Agent = DCA
DCA
DCS
DCA
DCS
DCA
DCA
CRI
CRI
Supplies
CRI
Supplies
CRI
Supplies

Focus on the SCA

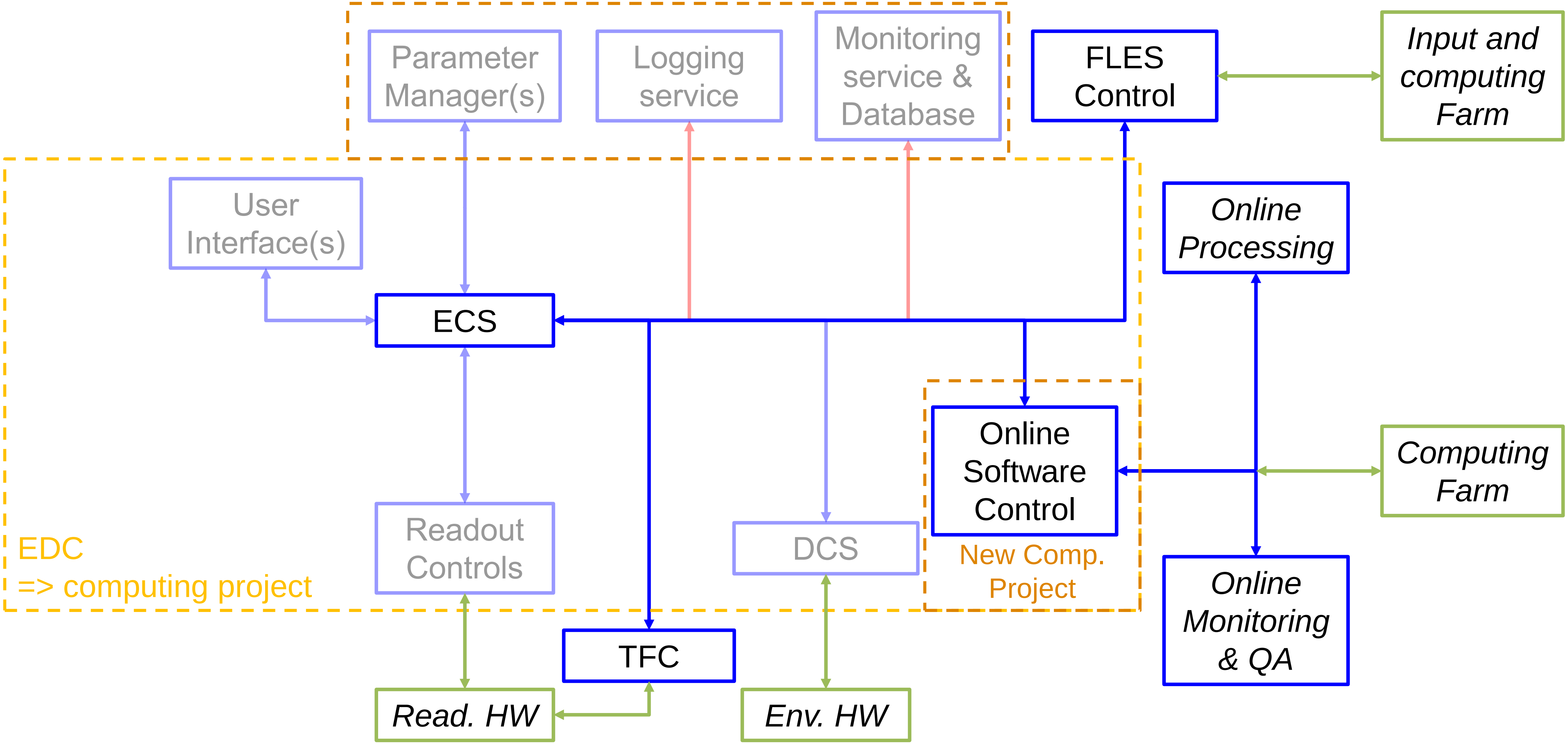
Aim: try to clear some confusion and questions

What is the SCA?

- Layer of the ECS, should be able to operate stand-alone for detector systems
- Represents a full system or the share of a full system assigned to a partition
  - Detectors
  - Central systems specific to a partition: DAQ, Online, TFC, …
  - Central systems common to full setup and “not-controlled”: Magnet, Cave, Accelerator, …
- Provides an aggregated State based on the states of the sub-elements of this system

  ⇒ Critical for <Run Control>, can be based on usage of “sub-agents” but not mandatory
- Accepts commands from higher layers of ECS (UI or automatized) to perform
  - common state transitions
  - system specific operations
- Manages the resolution of the configuration of the system
  - from descriptive tag(s) input
  - to actual values applied
- For detectors: manages the interface to both the DCA (readout chain) and the DCS (environment)

Central systems: non-detectors SCAs



Central systems: SCAs

Central systems are:

- Systems not producing physics data in the stream (but maybe adding auxiliary data in it)
- Systems consuming the data stream

Typically representing:

- Resources shared between partitions: TFC, Accelerator infos, Magnet
- Resources spread between partitions: FLES, Online processing
- Some of these systems are read-only and maybe handled in “DCS-SCA”: Accelerator, Magnet, …

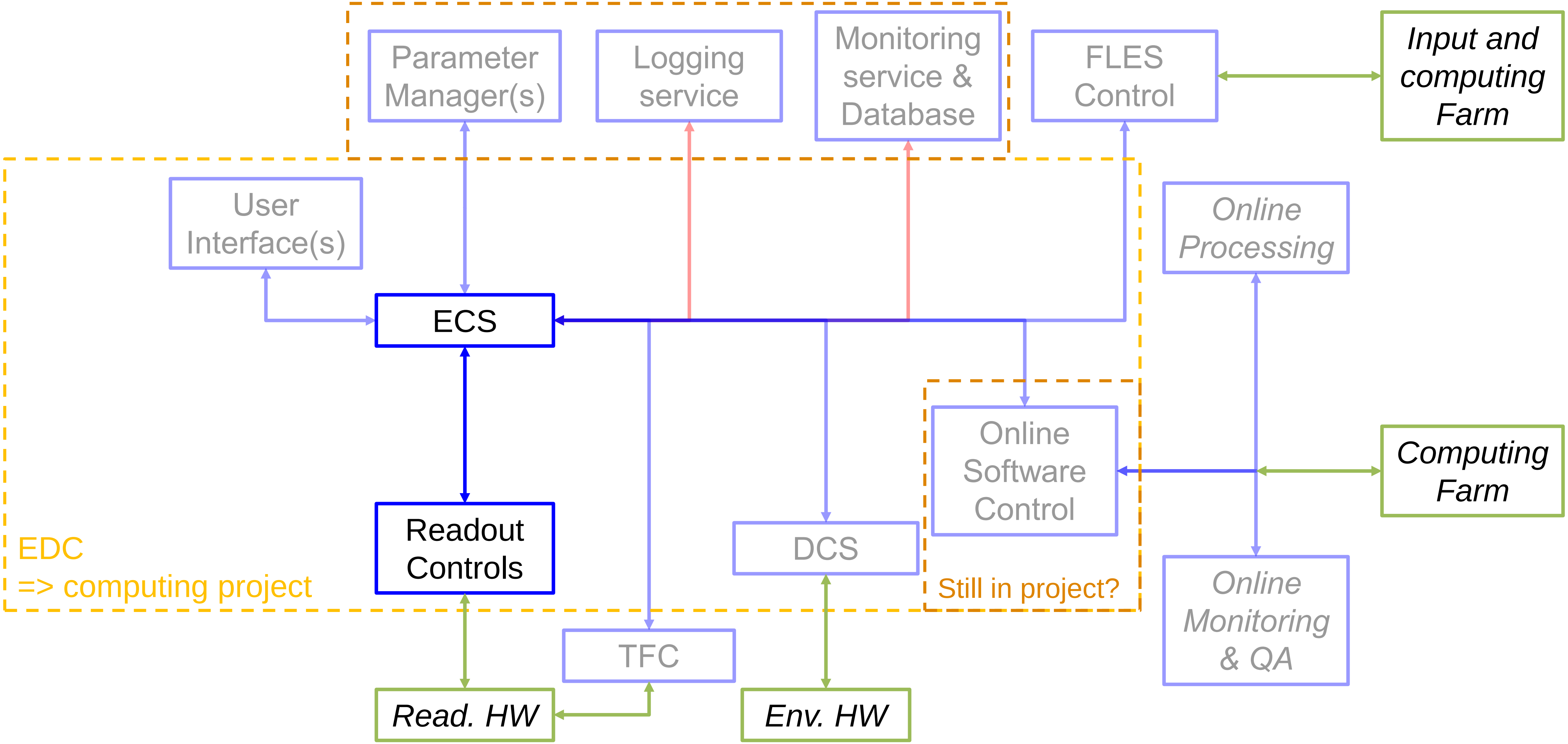
⇒ Need special treatment in order to allow partitions!

⇒ Main SCA entity representing the full system

⇒ “Partition SCA” sub-agent representing

- either the share of resources assigned to a given partition
- or a clone of the common device state

Detector systems: SCA and DCA



Detector systems: SCA and DCA

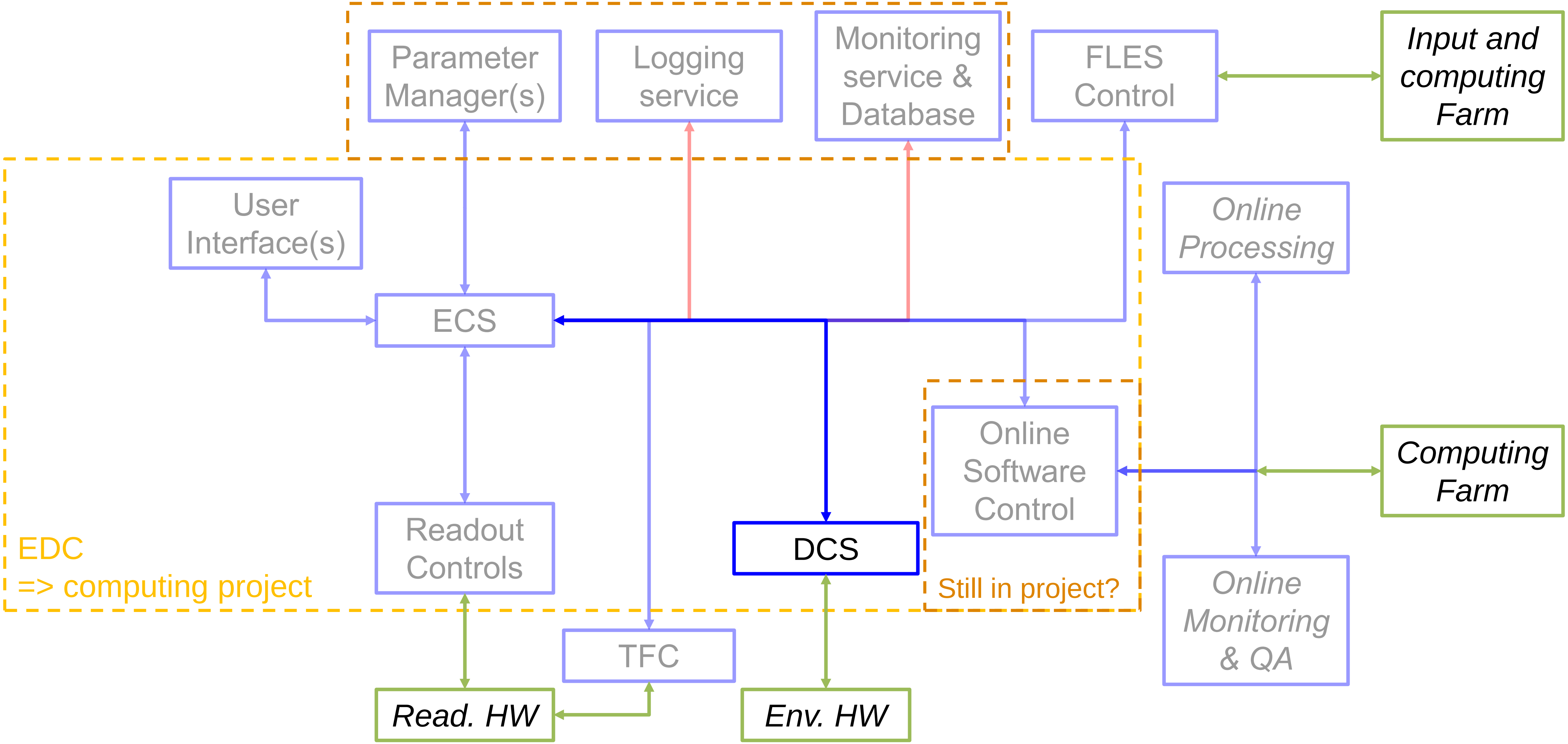
System Control Agent …. ~ ”the Team Leader”/”the Bauleiter”

- Derive system state from sub-elements states
- Convert state transitions (=“order”) into sequence of commands (=“actions”)
- Retrieve the parameter values corresponding to configuration (sub)tag(s)
- Entry point for “expert commands” access bypassing the ECS
- May have sub-agents corresponding to “islands” in both readout controls and slow controls
- Lower layer(s) of the ECS, Higher layer(s) of System specific controls

Device Control Agent …. ~ “the (Specialized) Worker”

- Interface to CRI driver for the readout/controls logics (everything outside of microslice readout)
- Access to all registers those controlled by FLES
- Protocols for the elements behind the CRI (GBTx, ASICs, …)
- Execute commands toward the CRI as “single sequence”
- Perform some monitoring accesses in background in “non-perturbative” way
- Carefully using threads for performance and stability reasons
- Written in C++ for performance reasons

DCS (brief)


Parameter Manager(s)
Logging service
Monitoring service & Database
FLES Control
Input and computing Farm
User Interface(s)
Online Processing
ECS
Online Software Control
Computing Farm
Readout Controls
DCS
EDC
=> computing project
Still in project?
Online Monitoring & QA
TFC
Read. HW
Env. HW

Focus on the DCS and interface to ECS

Detector Control Systems

Goals/Properties of the DCS relative to ECS?

- Implemented in EPICS
- Both system independent of ECS and “element of the ECS” (in the case of common/central/setup PVs)
  - ⇒ Interfaces at different levels of the CBM systems
  - ⇒ Should be able to be operated even if ECS is down!
- Controls and monitors the environment of CBM systems
  - – Supplies (LV, HV, gas, local cooling, …)
  - – All monitoring not linked to the data chain (temperature, pressure, light, …)
  - ⇒ Independent instances per Detector System
- Accepts commands from corresponding layers of ECS (UI or automatized) to
  - – perform changes to the operating conditions (single value change or state transitions)
  - – provide information on system/environment status
- Monitors and archives
  - – Alerting/Alarming
  - – Automatic actions = must be shared to the corresponding SCA if effect on readout HW or data expected

Interfaces to DCS #1: DCS-SCA

- Represents DCS as a **system** + Monitors/Controls CBM environment
- State of the DCS HW (main nodes, network, …)
- Interface for all non-system specific requests
  - Cave environment
  - Common supplies
    - Cooling
    - (Main) Power
    - Network
  - Eventually shared read-only resources? (Magnet, …)

  ⇒ … anything which would be hooked to EPICS but not assigned to a detector or central system

⇒ **Expected operation mode: goes green at startup, stays green until end of operation period**

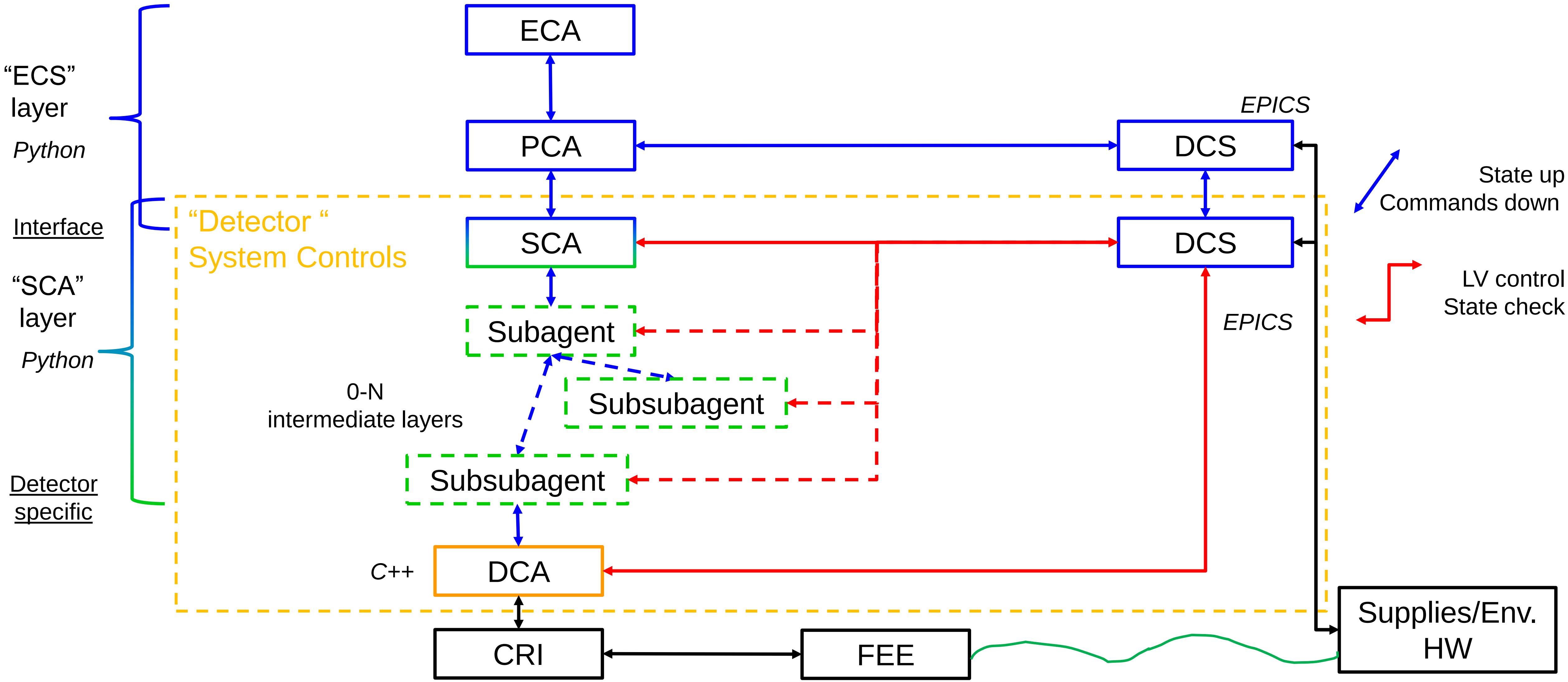
⇒ **State changes here indicate major/experiment level problems**

Interfaces to DCS #2: detector DCS ⬄ detector SCA

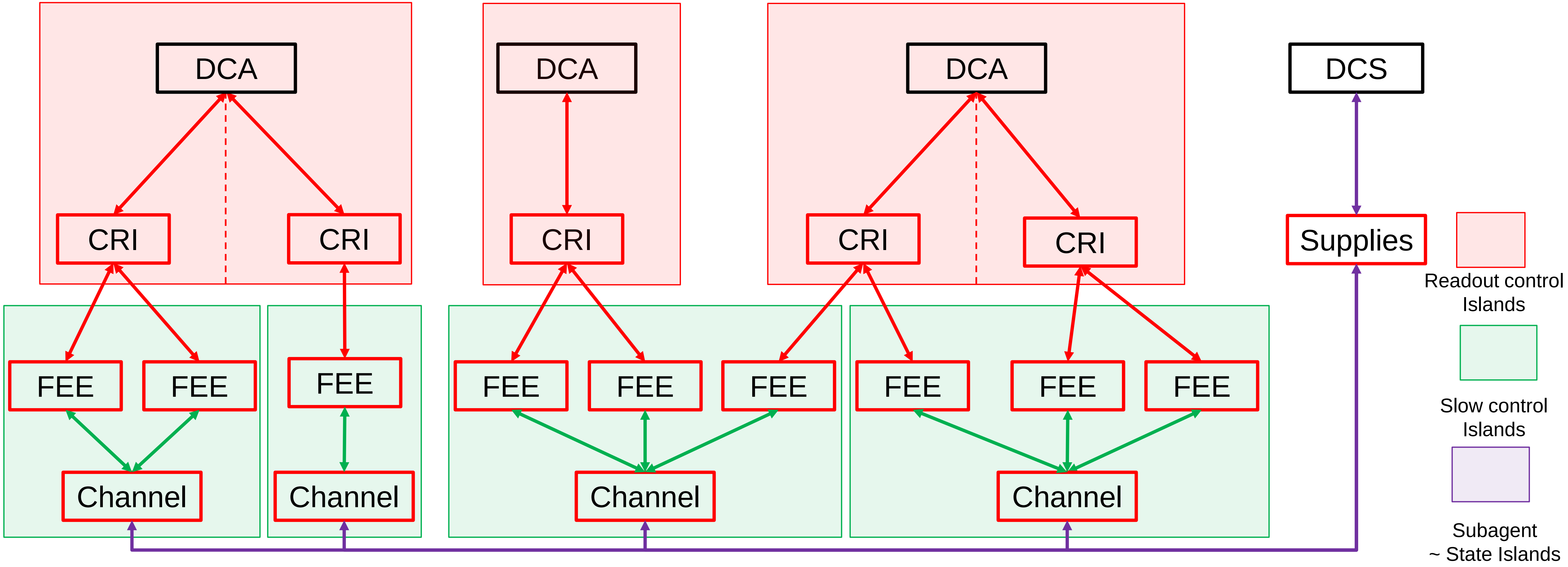
- SCA <A> needs to be able to request current state from DCS <A> (“on demand” updates)
- SCA <A> needs to be able to sent config point “change requests” to DCS <A>
- DCS <A> needs to be able to inform SCA <A> of changes in the relevant PV state (“auto” updates)

A few Visual/Graphical examples of ECS-DCS interplay

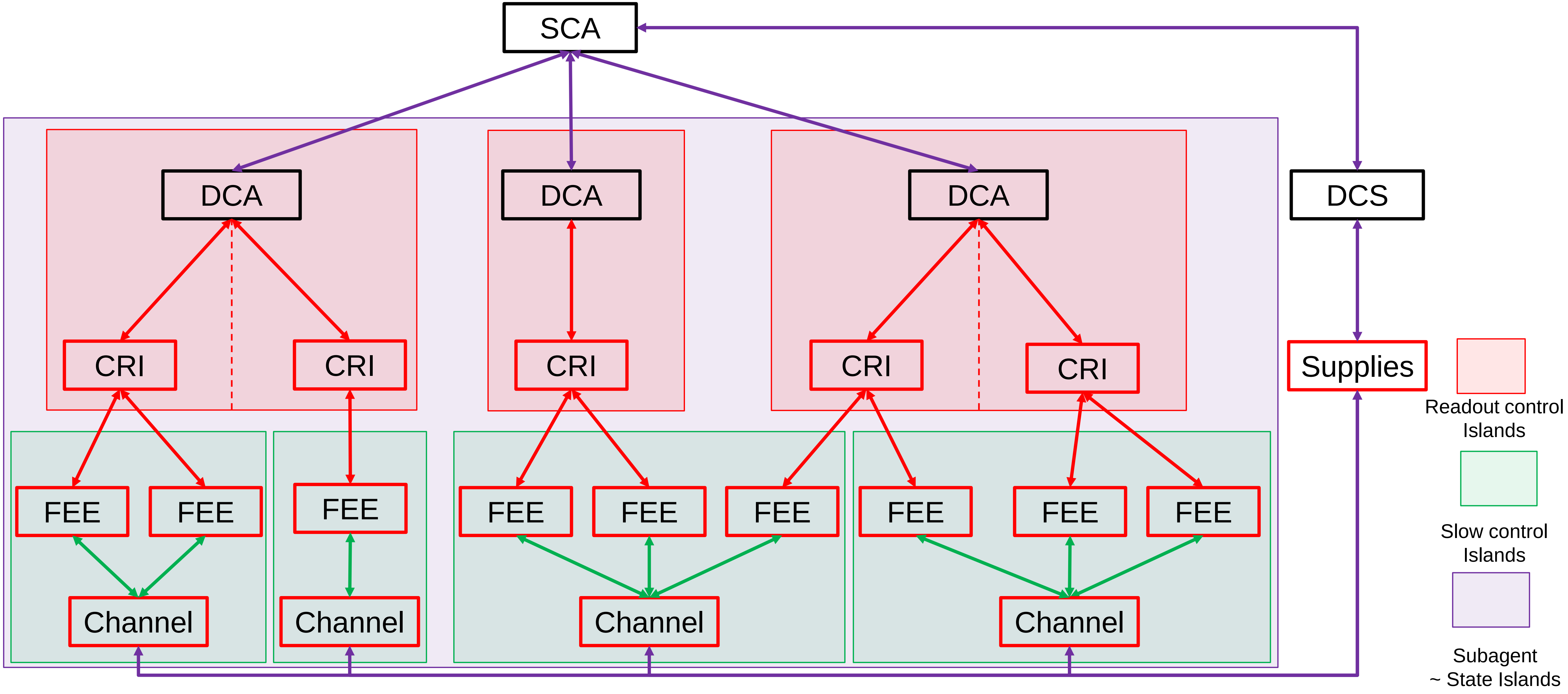
EDC “execution”/“logical” layers, 1 System partition



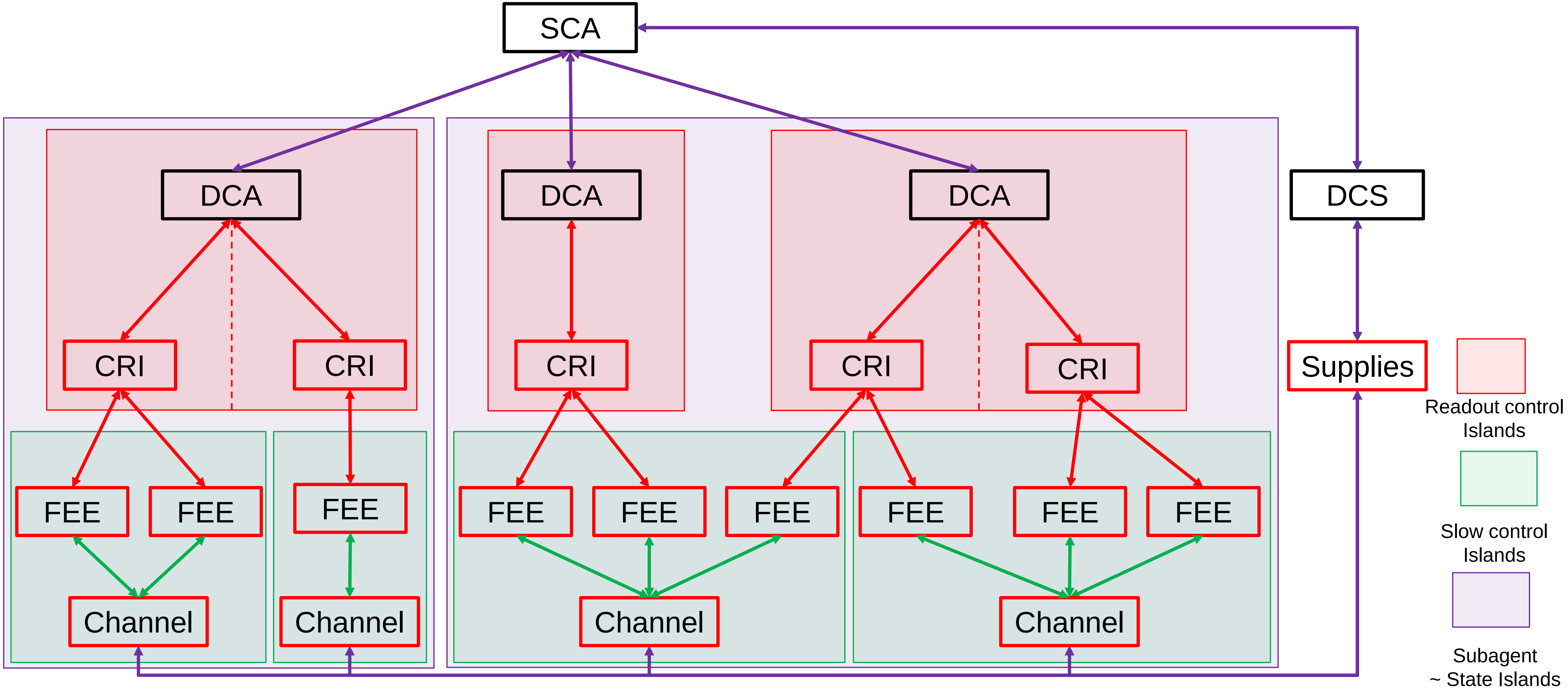
Detector systems: SCA-subagents and DCA



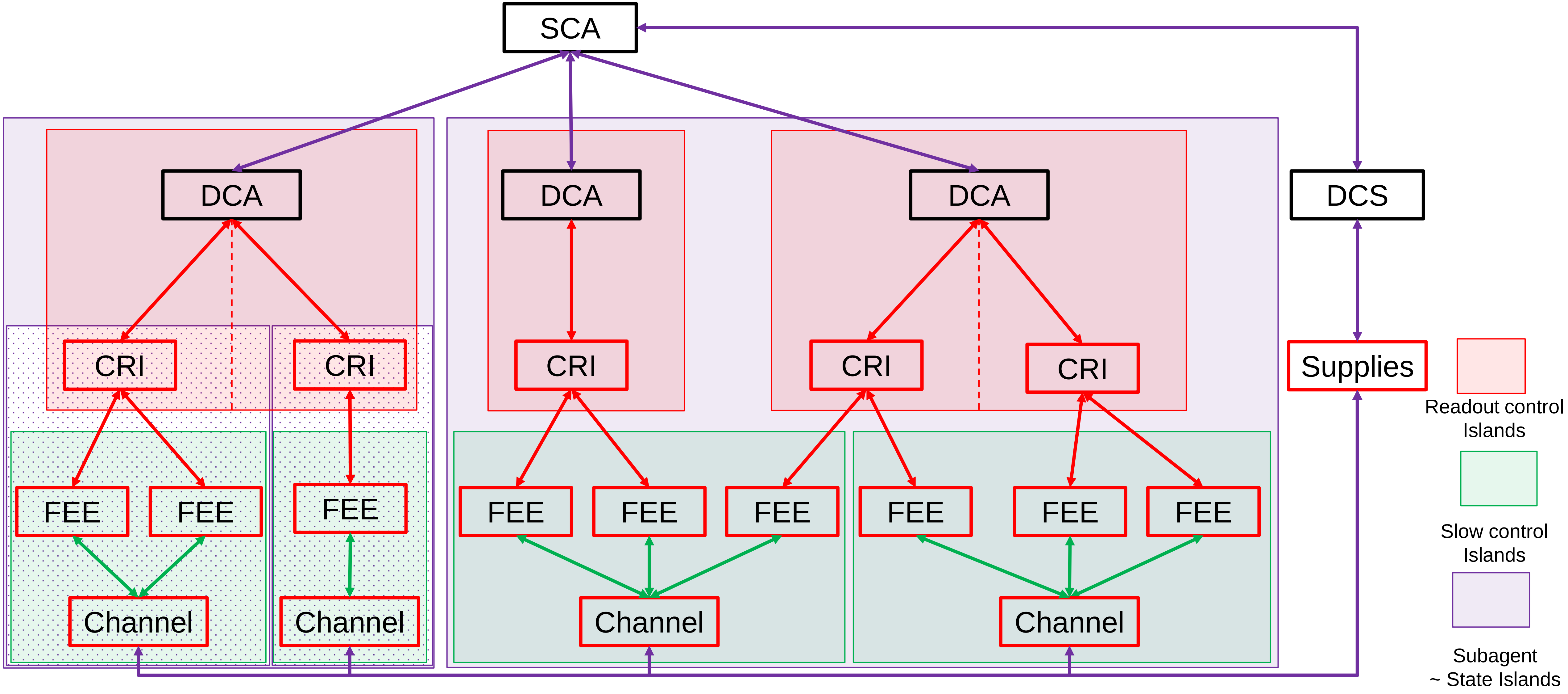
Detector systems: SCA-subagents and DCA



Detector systems: SCA-subagents and DCA



Detector systems: SCA-subagents and DCA



Latest developments outside of DCS

mCBM main campaign concluded with 2 new detectors without major blocking point

Auxiliary systems: Logging

- Master student internship Spring 2025
  - ⇒ On ElasticSearch + Kibana stack
  - ⇒ Proof of concept using 4 types of legacy sources from mCBM
  - ⇒ Now need expansion to practical example and test in mCBM infra (Docker, permissions, archives, …)
  - ⇒ Corresponding emitters merged in DCA, Python-Cri and ECS-core

Auxiliary systems: Monitoring and QA

- PHD student starting soon under T. Stockman on “IA-based alerting”
  - ⇒ Using HYDRA package from Jefferson Labs, first contacts established

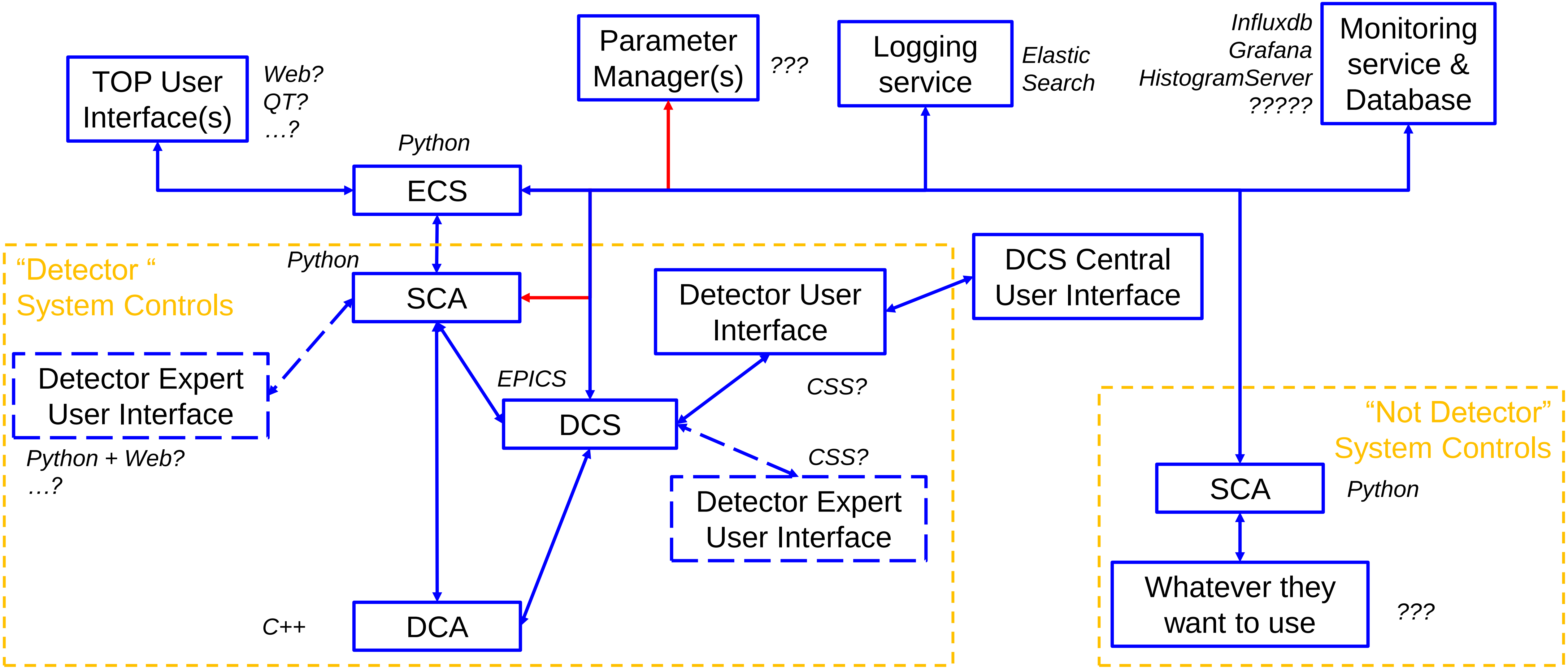
Improvements and new tasks identification in ECS-core and DCA

- DCA: Triggered by first attempt at making an SCA for the mTFC 2.0 prototype system
  - ⇒ I2C usage in CRI1/CRI2, Monitoring loops, DCA Python bindings packaging
- Core: Triggered by feedback from DCA development
  - ⇒ Usage of Linux File-Descriptors for improved polling in background loops = better performances?

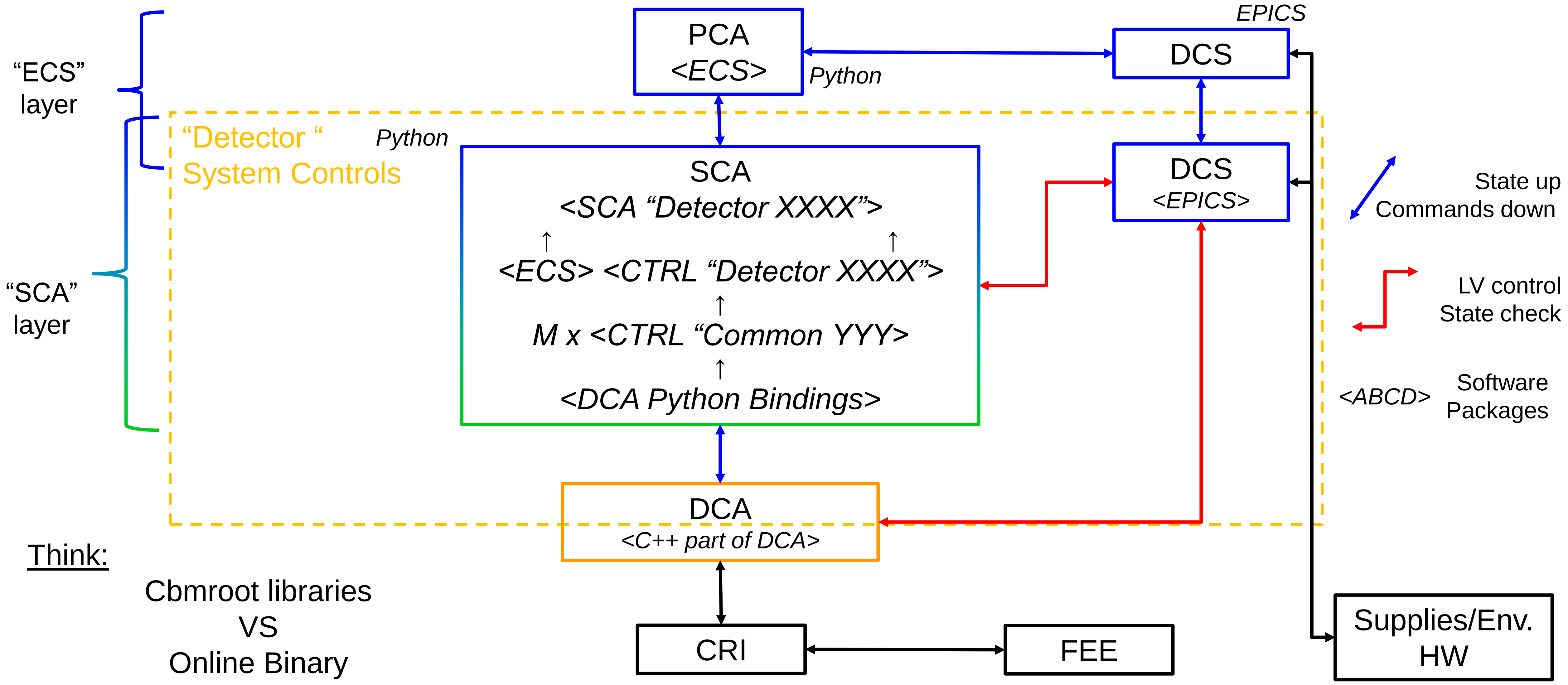
Legacy/Prototype readout controls (Python-cri): Final merges and Cleanup in preparation for SCA dev

Thank you for your attention

Components and technologies (current baseline)



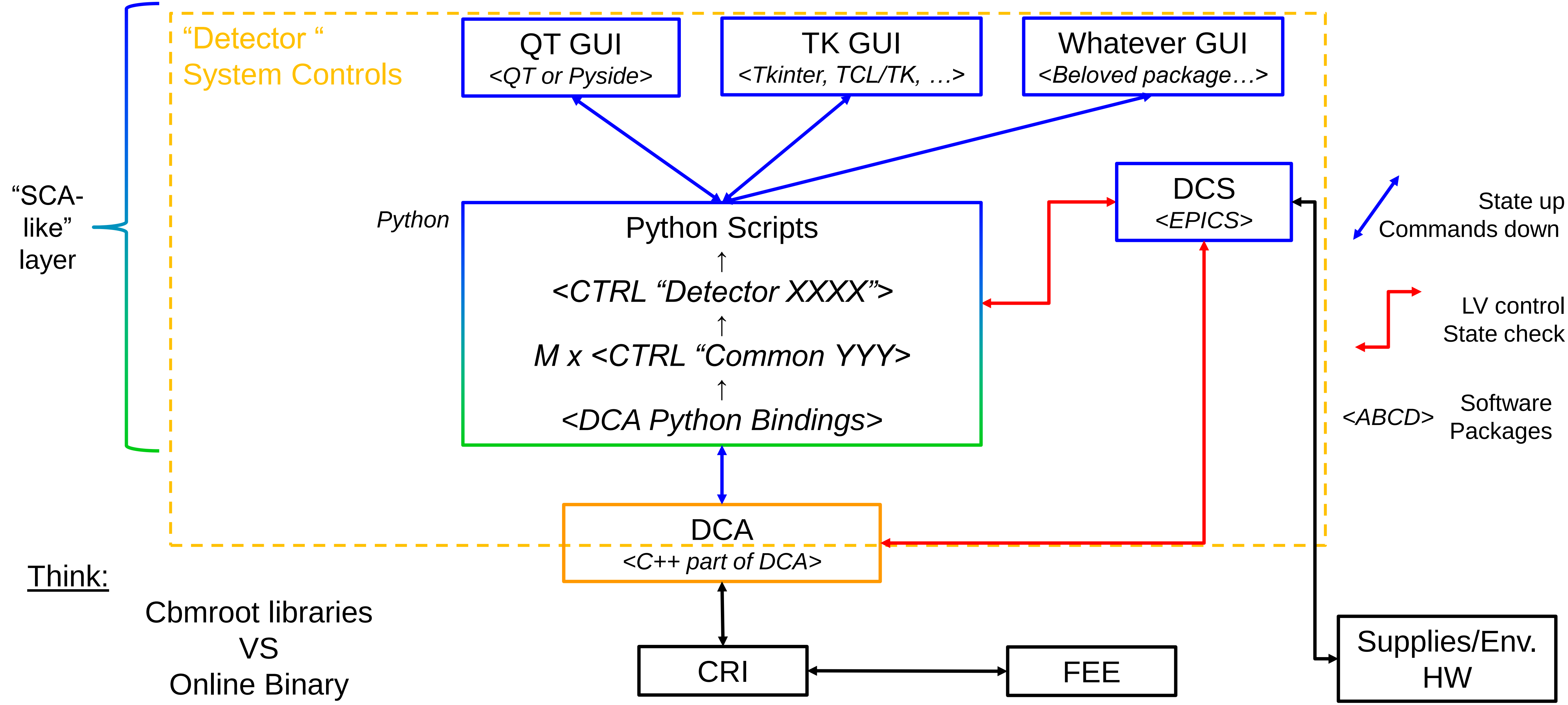
EDC "software" layers, 1 System partition



Think:

Cbmroot libraries
VS
Online Binary

EDC “software” layers, Expert/debug/hack mode



Think:

Cbmroot libraries
VS
Online Binary

DCS (brief)

Detector Control Systems

Missions:

- Controls of supplies to the detectors and electronics
  - Gas
  - Low Voltage
  - High Voltage
  - …
- Environment monitoring and controls
  - Temperature
  - Pressure
  - Cooling
  - …

Specifications:

- Implemented in EPICS
- Independent instances per Detector System

⇒ More details by following presentations