Slow Control

T. Le Flour

J.L Panazol

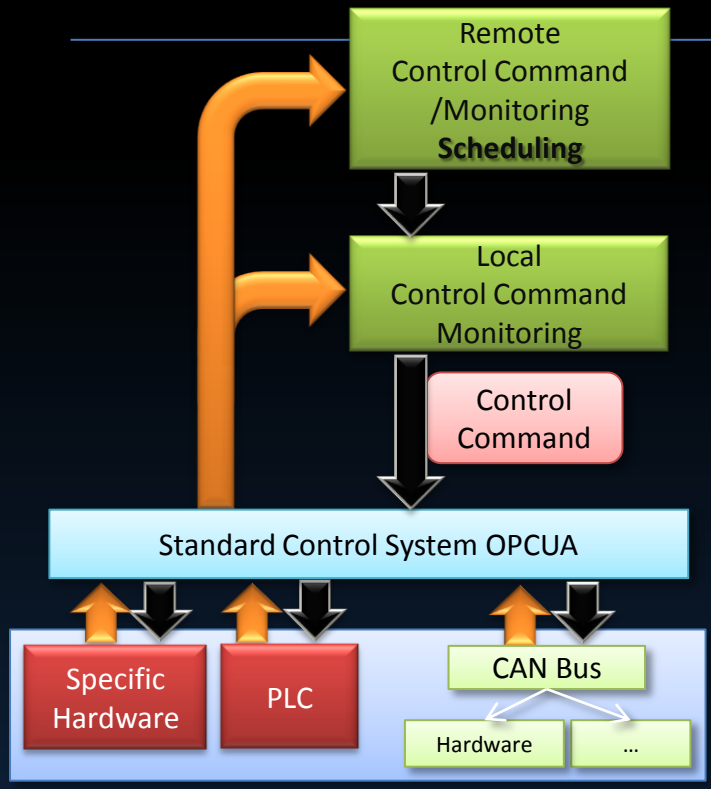
L.A.P.P/IN2P3/CNRS

Le « Slow Control »

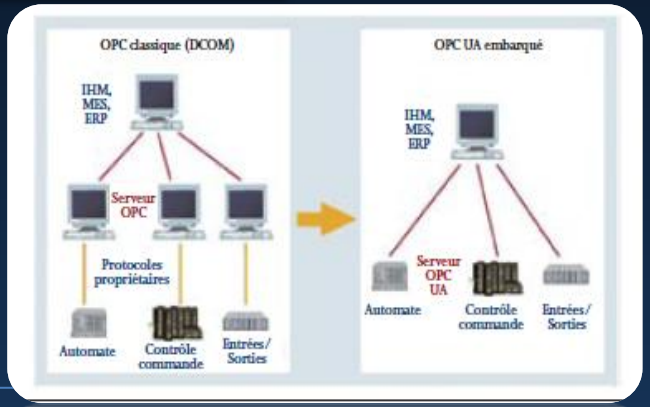
- HESS2 :
  - ✓« Drive System » pour le chargement/débarquement de la camera HESS2
    - Implique le monitoring et le contrôle/commande des mouvements au niveau « Software »
      - ❑ OPC, …
  - ✓Système de sécurité de la camera
    - Température, Lumière, portes, …
  - ✓Intégration dans l'environnement HESSDaq

Notre retour d' expérience pour CTA

- Homogénéité de l'accès au matériel
- Le matériel est vu comme un ensemble de points de contrôles/commandes.
  - ✓ Accès doit être homogène et indépendant du matériel
- Accès distant aux informations « matérielles » doit être possible depuis n'importe ou.
  - ✓ Hiérarchie de l'information
  - ✓ Simplicité d' accès a distance



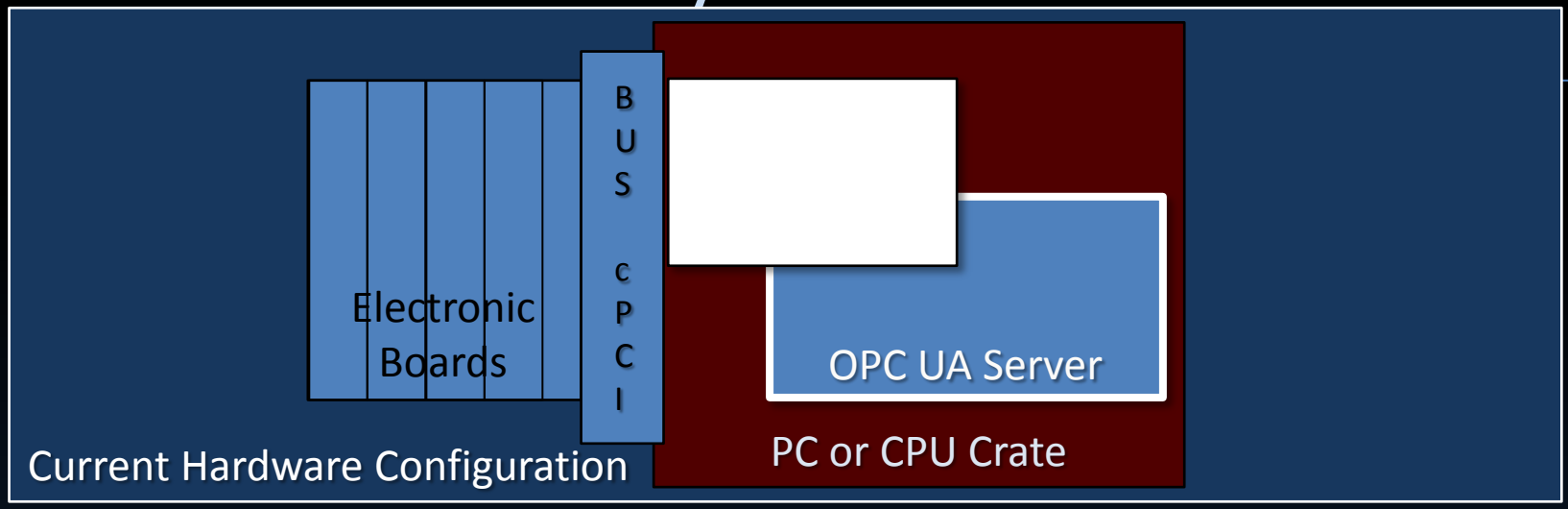
OPCUA



Nos propositions dans CTA/ACTL

- Utiliser OPCUA comme le standard d' accès au matériel
  - ✓ PLC, Electronique, …
- Etudier le Framework « ACS » et l' intégration de l'environnement OPCUA
  - ✓ Châssis sécurité HESS2 intégré dans ACS a l'aide de développement de DEVIO/OPCUA
- Considérer également l' homogénéité matérielle (CPU, …)

HESS2 Security Crate with OPCUA



SERVER PART:

- PCI Driver has been ported on the Fedora14 Platform
- OPC UA Server implemented in C++ and currently runs on a Fedora14 platform
- All the possible actions on the OPCUA Nodes have been implemented
  - Standard calls on objects(Node) methods
  - Events
  - Callbacks

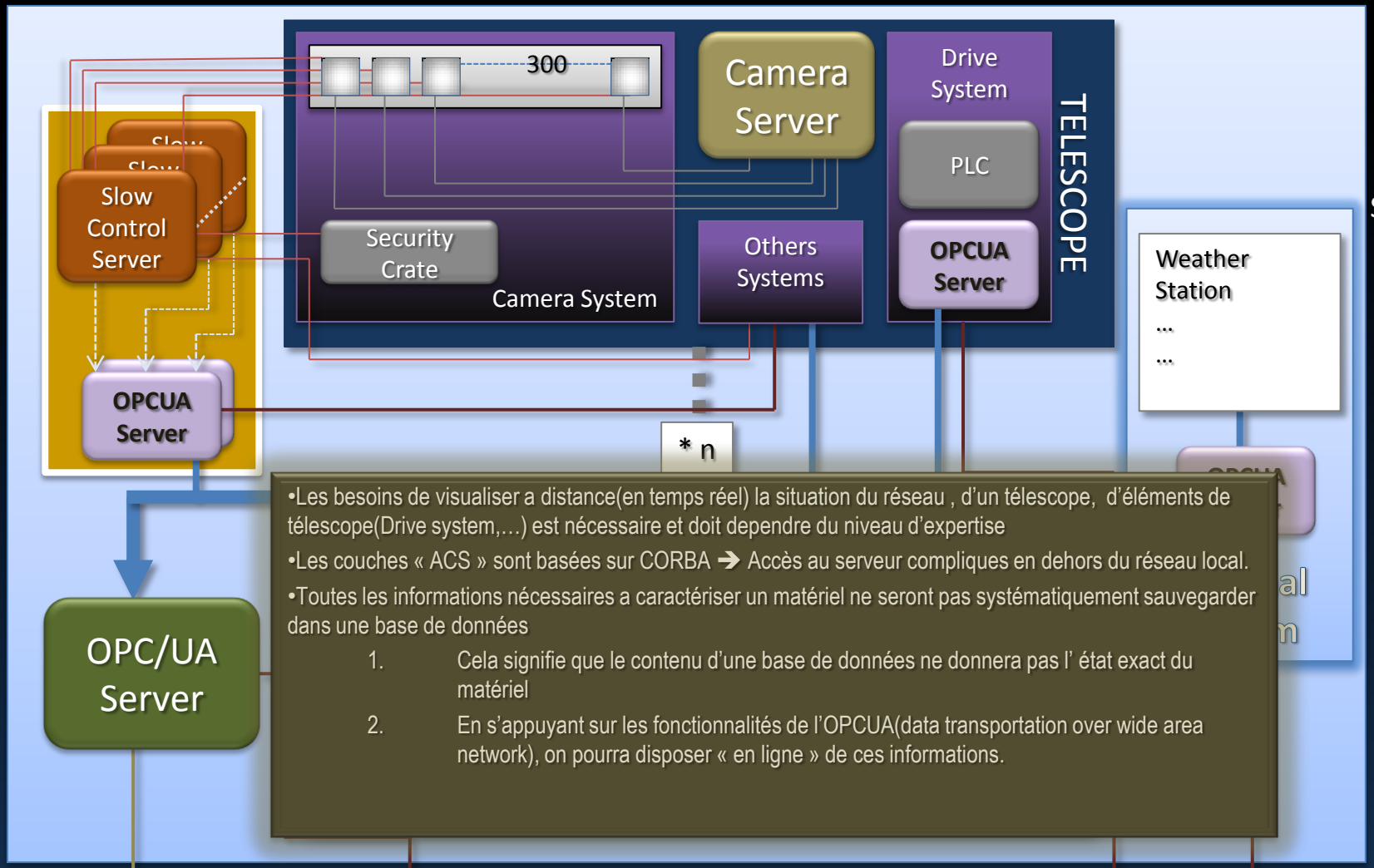
CLIENT PART :

- Server has been tested by using the generic client included in the *Unified Automation* distribution
- Server has accessed via a specific client written in C++
- An *ACS DevIO* encapsulating the OPC UA Server connection has been written, configured via Container description and tested successfully via the *ACS Command Center*
- A basic JAVA client also tested

Camera Slow Control

- Based on discussion and collaboration with CPPM and LPNHE people (MST NectarCam)
- Different behaviors for the slow control/monitoring have been identified
  1. One part has to be embedded in the camera for security reason (quick action ➔ FPGA coding, …)
  2. A second part being externalized on the ground(control room) for the all actions with low security level
     - ✓ Example : For cooling and humidity slow control, decisions of action are less critical
       - ✓ Can be driven either from the local place or from a remote system part of the ACTL framework.
     - ✓ heavy computing environment(computer, dedicated network connection, …) in the camera is not needed.
     - ✓ Commands can be managed and addressed to the telescope/camera from outside.

Slow Control : Logical View

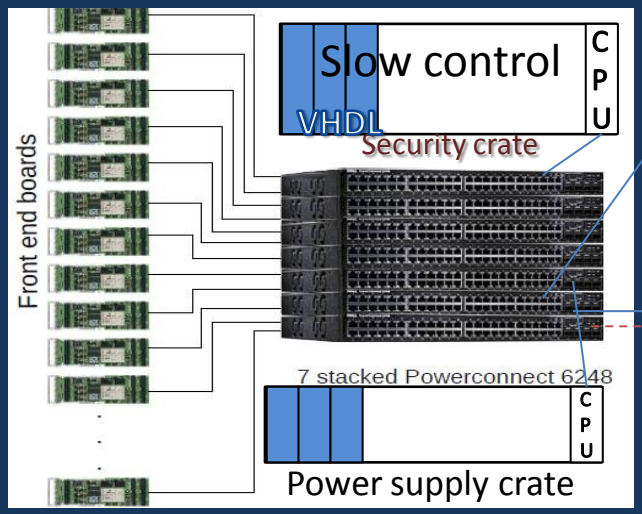


Read out Channel

Monitoring/ Slow Control Data Channel

OPCUA Data Channel

Slow Control : Integration in the DAQ Network



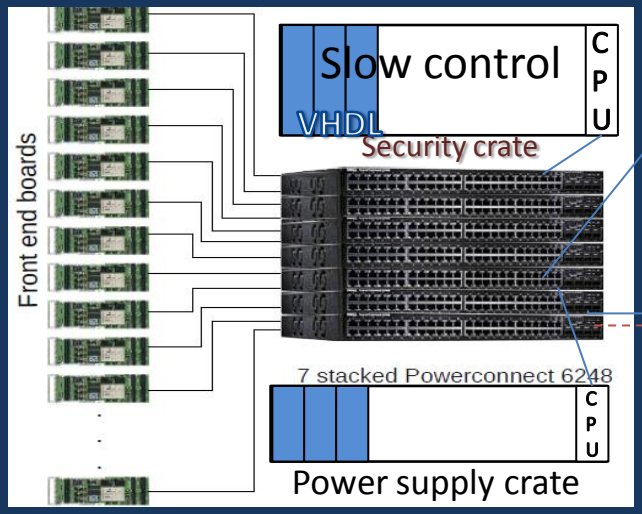
Camera server

DAQ Switch 40 ports

Slow Control Switch 40 ports

Slow control server OPCUA

40*Télescope for MST



Camera server

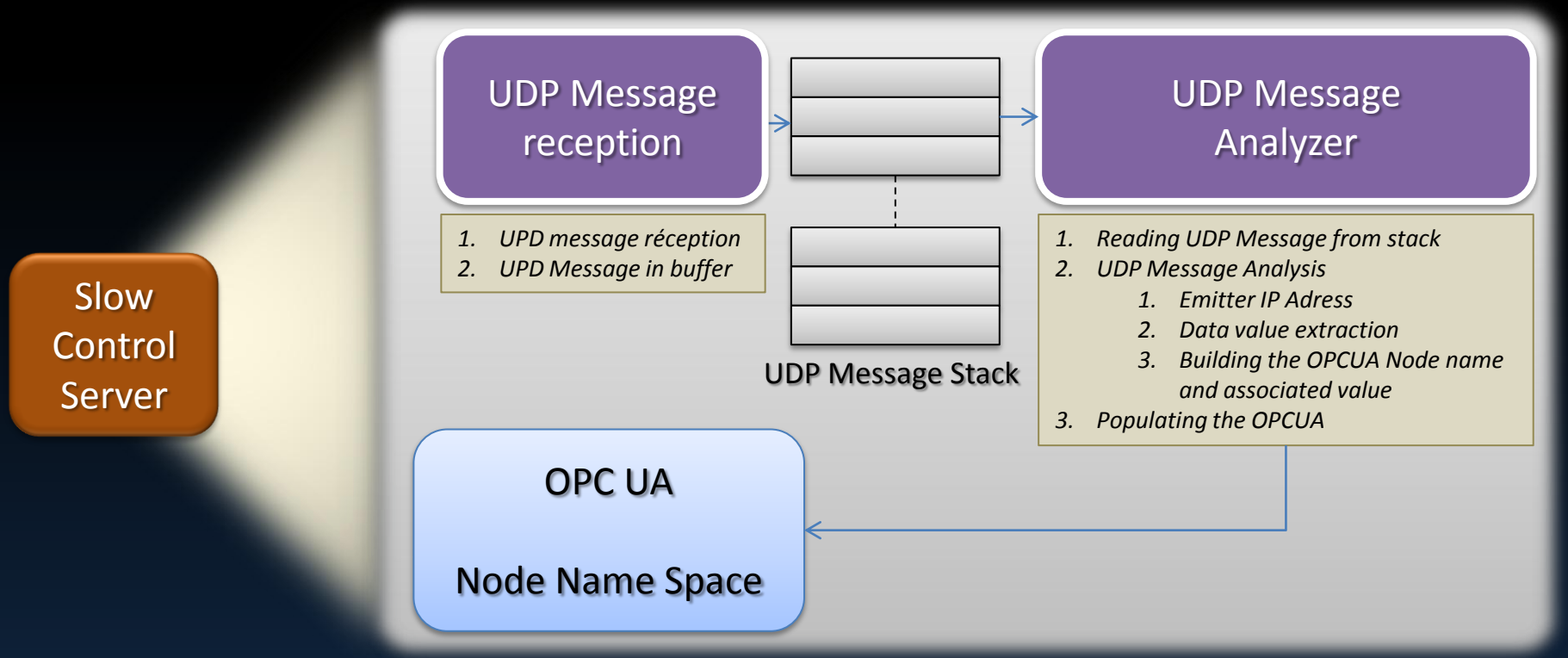
Readout

Slow Control Data & Monitoring

Monitoring Data

- Based on NecTar data transfer :
  - Each system emitting monitoring data from hardware will send this data via structured UDP frames.
- A UDP frame is uniquely identified corresponding to the data description packet.
- UDP Frame content example:
  - <IP Address>, <Message Type><Data list>
    - IP Address is used to locate easily the message emitter.
    - The data list is analyzed by using to the message description (XML file, …)
- The decoded values (node name and value) are used to identify the **OPCUA Node** and populate it.
- Some estimated numbers :
  - 40 telescopes * 300 « Front End » boards * 45 infos (HV, PM, Temperature, Power, …) = 540000 control points➔
    - ❑ The overload of an OPCUA server has to be evaluated ➔ this should imply the use of many OPCUA servers.

Monitoring data flow



- Slow Control Server ➔
  - 2 options for the implementation :
    1. The slow control server is a single process(multithreaded) acting as an OPCUA server
    2. The slow control server and the OPCUA Server are 2 separated processes running on distributed environment
       - Imply to have as many OPCUA client licenses as running slow control servers.

Hardware integration : First ideas

Distribution of the monitoring data
Organization of the hardware control command

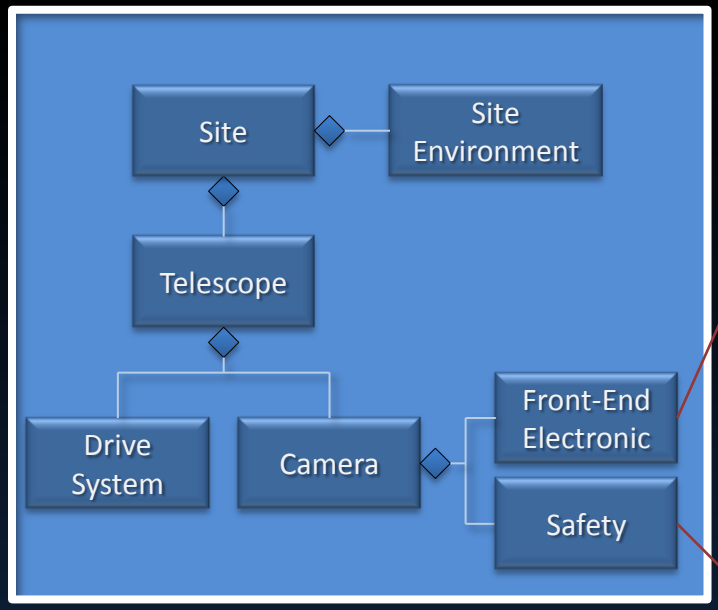
- via the Slow Control Server
  - ✓ Based on messaging description and UPD frame distribution
- Direct access to OPCUA Server embedded on the electronics boards or associated to the electronic board.
  - ✓ Link between very specialized OPCUA server and higher level OPCUA server has to be studied.

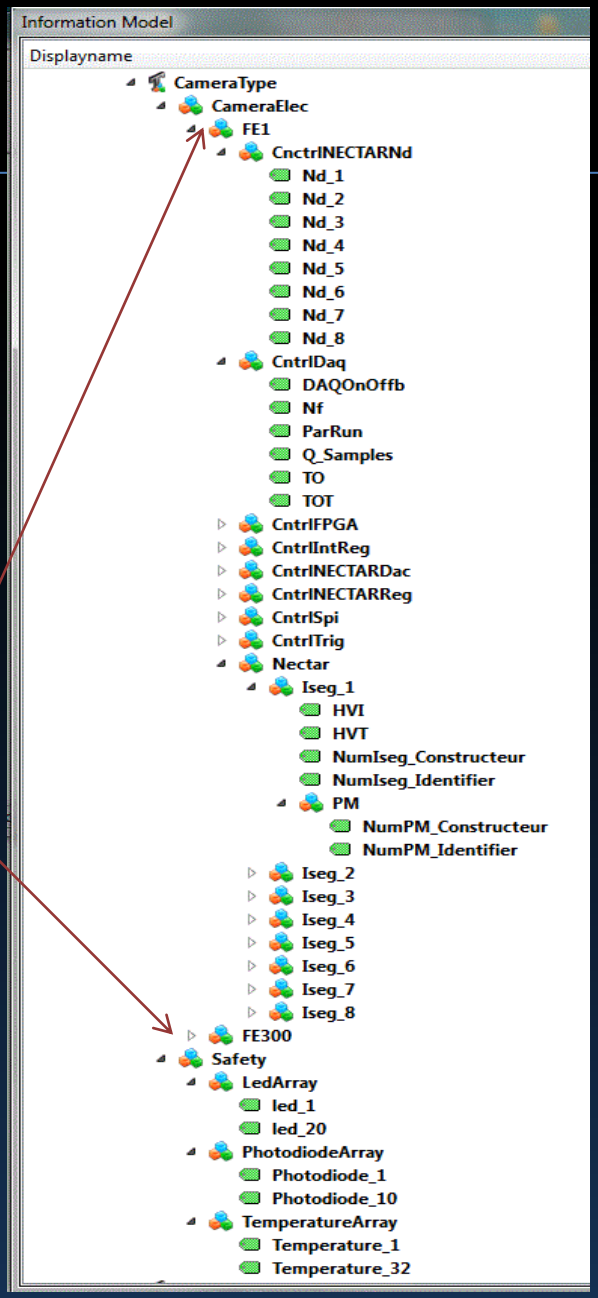
- Control command :
  - ✓ Hardware command sets or subsets can be described
  - ✓ Each set or subset of command can be seen as a set of node(write mode) in an OPCUA server
    - Writing in the OPCUA node triggers the associated action
  - ✓ From the client point of view :
    - Command configuration file for a specific hardware allows to identify the node name in the OPCUA server and the parameters associated to the command.

Current status

- Hierarchical model defined in the OPCUA Server as followed via UAModeler

  - Some limitation with UAModeler ➔
    - Code generation(complex structure data points: array,…)
    - Event behavior not yet fully implemented
  - May be in April release





- Evaluation of the load of the OPCUA depending on the amount of nodes.
  - Node updating speed ➔
    - #Nodes = 40 * 300 * 4 = 48000
    - Write mode(loop) = 3 seconds/by loop for the all node update
    - Read mode not yet fully evaluated

ToDo List

- OPCUA Node Hierarchy ➔ Node dependencies over distributed namespaces
- Definition of the granularity of the OPCUA nodes
  - ✓ At a certain level, a node value depends on the computation of low-level node subset.
    - Ie : Telescope state results on a computation of a node set states.
- Study the policy of inter-connection between distributed OPCUA namespaces
  - ✓ Callbacks management for the node's changes and node dependencies .

Conclusions

- Identification des besoins indépendamment du type de télescope.
- Assurer un développement s'adaptant a tous les environnements.
- Extraire des développements en cours(MST, …) des concepts :
  - ✓ Intégration du matériel ➔ vers une description générique du matériel(points de contrôles, commandes, …)
  - ✓ Développement de « toolkit » et/ou d'API pour permettre une intégration plus simple dans l'environnement OPCUA/ACS