

2nde JOURNÉES ONLINE

Le futur de nos technologies

Du 18 au 20 novembre 2025 à  **IJCLab**, Orsay
Irène Joliot-Curie



Retour d'expérience, forces et faiblesses

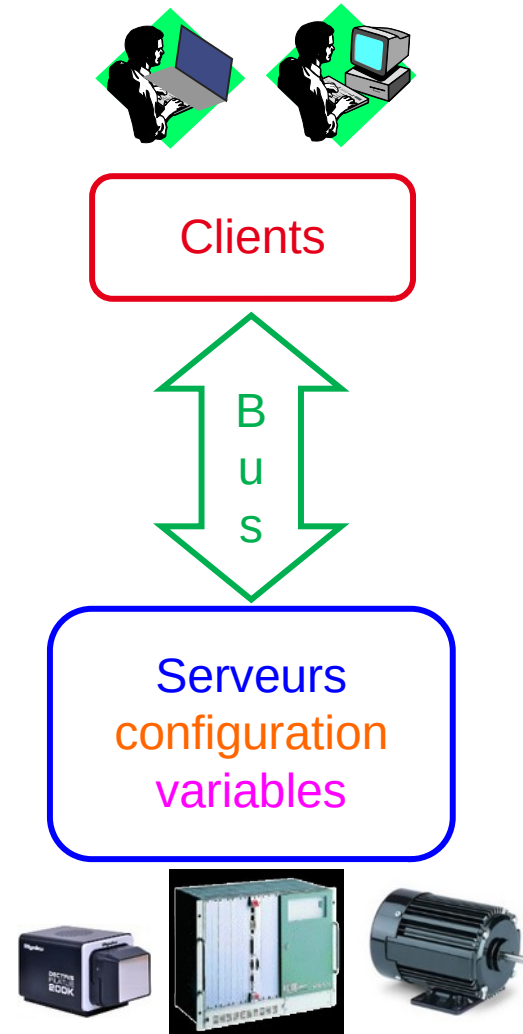
Philippe GAURON
IJCLab/IN2P3

Katy SAINTIN
CEA/IRFU & SOLEIL

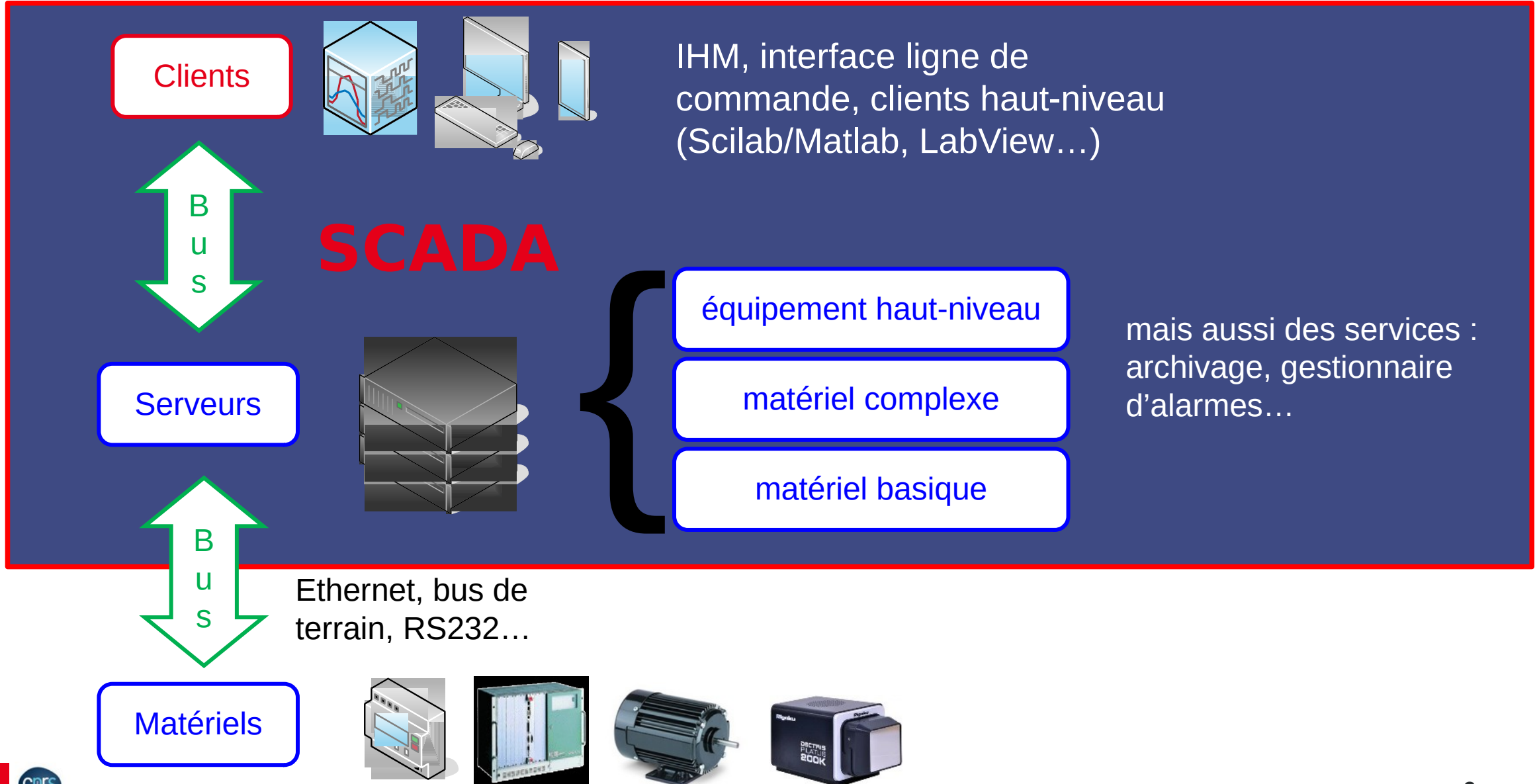


Acteurs d'un SCADA ?

- SCADA : **S**upervisory **C**ontrol and **D**ata **A**cquisition (Système de contrôle-commande)
- Framework : Cadriciel (suite d'outils et de bibliothèques)
- But : Pilotage d'infrastructures (contrôle = lecture, commande = écriture)
- **Variables** : Elles correspondent à des grandeurs physiques ou abstraites
- **Configuration** : Système de configuration (fichiers de config ou base de données...)
- Architecture : 3 tiers (matériel, communication, logique métier)
- **Protocole** : Bus de communication connecté avec tout « objet » du système
- **Serveurs** : Pilotes de matériels et traitement de haut niveau
- **Clients** : IHM, interfaces en ligne de commande, logiciel de haut niveau
- Services : Archivage, gestionnaires d'alarmes, journal, administration du système ...
- Communauté : Support, outils d'échanges, collaborations et présentations
- Initiation : Formation , doc, paquets d'installation, liste de pilotes/outils, VM



Architecture d'un SCADA



EPICS / TANGO depuis leurs débuts

1991 : 1ère publication ICALEPCS

~1990 : version initiale

Licence :

EPICS Base License Agreement (propriétaire)

2004 : chgt de licence

Licences :

- logiciel → EPICS Open licence (non validée OSI mais proche)

2010 : 1ère v4 (PVAccess)

2014 : version R3.15

2017 : dernière version v7 (v3+v4=v7)



33 structures¹ : France : ITER



, GSI, LNL, SLAC, LIGO,  diamond ...

¹ officiellement recensés

² membres du comité de pilotage

1999 : version initiale, 1ère publication ICALEPCS

Licences :

- bibli système → LGPL
- IHM et pilotes → GPL
- IHM et pilotes → GPL

2024 : dernière version v10





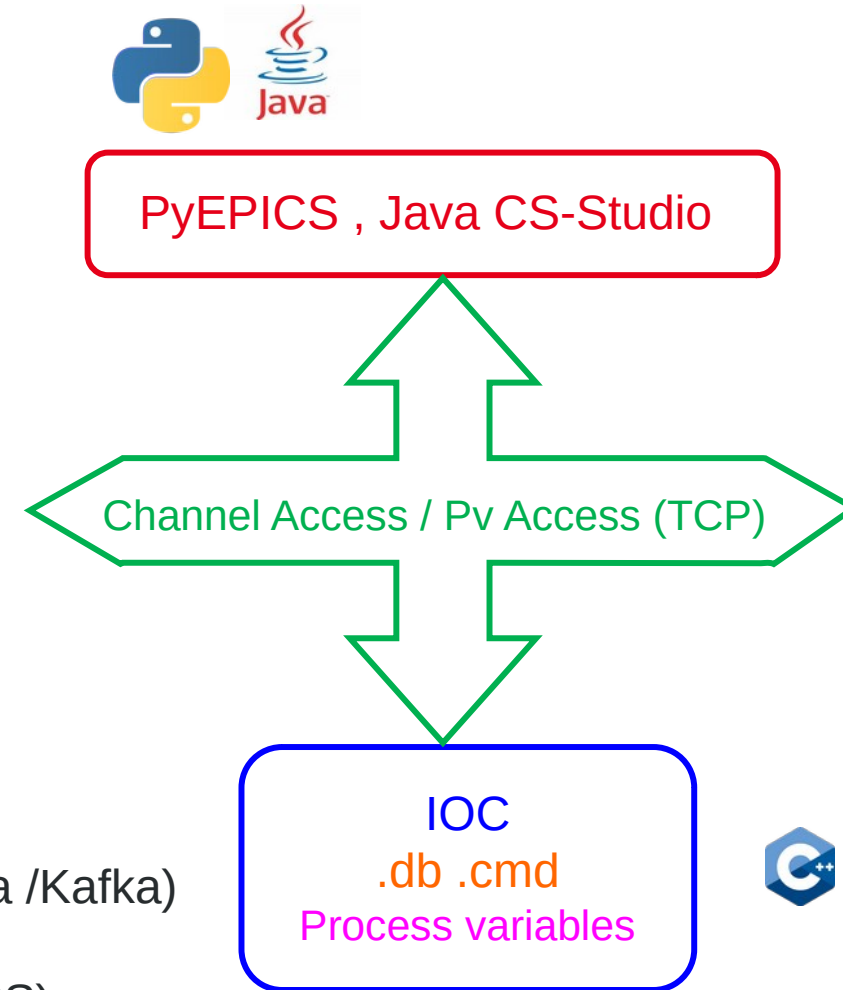
49 structures (dont 19 industriels)¹ :

France ², ², CEA, X, LULI, IJCLab, Onera, Atos worldgrid, Thales, Codra, Nexeya


DESY², ALBA², SKAO, Elettra², INAF², Solaris², Max-IV², JINR, Scandinova, Instrumentation technologies...
4

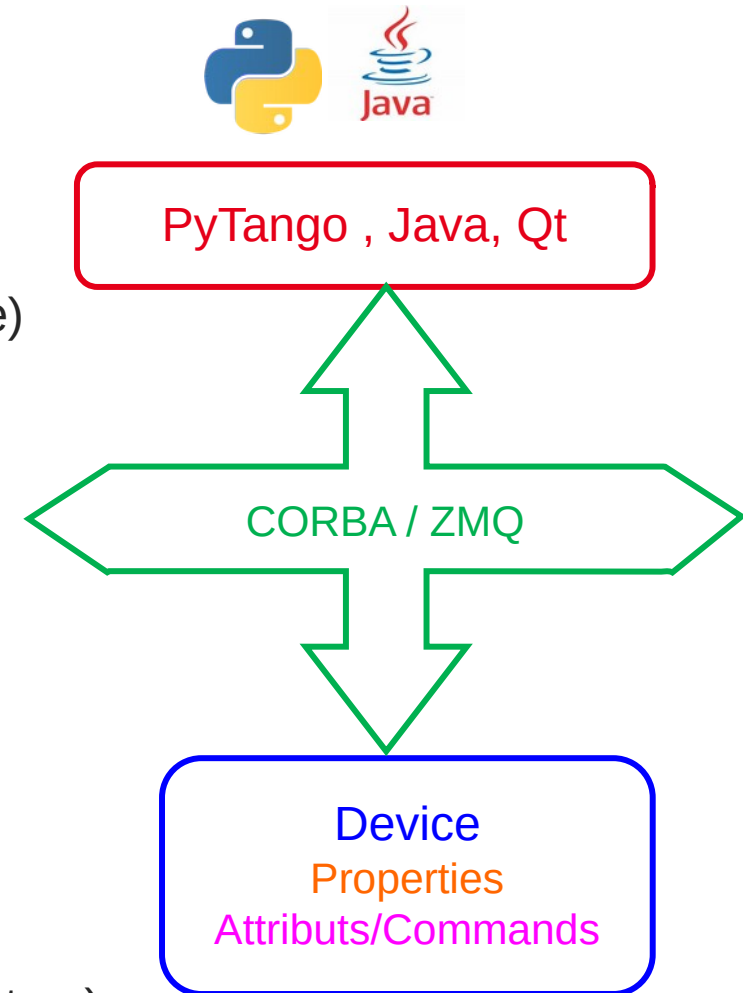
Terminologies

- OS : Majoritairement sur  mais existe sur Windows 
- **Protocoles** : TCP Channel Access (CA) et PV Access (PVA)
- **Variables** : Process variables (PV) et records
- **Configuration** : « *Database* » EPICS fichier.db (PV) et .cmd (protocole)
- **Serveurs** : **Input Output Controller = IOC** (C++, binding Python, Java)
- **Clients** : PyEPICS (Python), Java CS-Studio-Phoebus, QT, Reac
- Services : Archive Appliance (Java/Google protobuf), PhoebusAlarms (Java /Kafka)
- Communauté : <https://epics-controls.org/> (Tech Talk, Matrix, meetings EPICS)



Terminologies TANGO

- OS : Majoritairement sur  mais existe sur Windows 
- **Configuration** : Propriétés stockées en base de données (Mysql device Database)
- **Variables** : Attributs / Commandes (State & Status communs)
- **Protocoles** : CORBA → polling, ZMQ → évènementiel
- **Serveurs** : **Device servers = DS** (C++, Python, Java)
- **Clients** : PyTango (Python), ATK (Java Swing), Taurus (Qt/Python) + bindings
- **Services** : Archivage HDB++ (C++), Panic Alarm System (Python), journalisation
- **Communauté** : <https://www.tango-controls.org/> (listes. forum, Mattermost, rencontres)

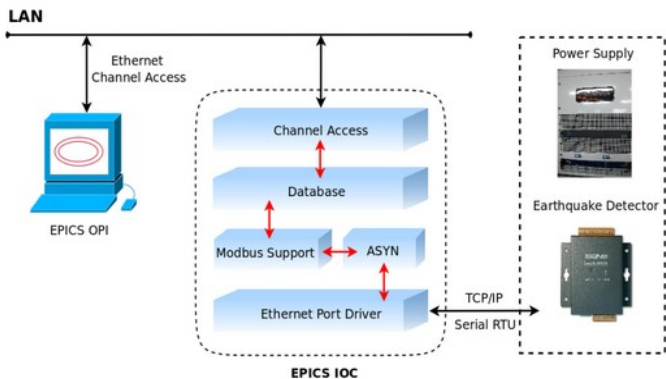


Exemple de développement d'un IOC

Un IOC en Channel Access est basé sur des modules supportant des protocoles spécifiques :

- EPICS en référence 68 <https://docs.epics-controls.org/en/latest/software/epics-related-software.html>
- les modules les plus utilisés sont : StreamDevice, Asyn et Calc
- certains modules s'appuient eux-mêmes sur Asyn tel que Modbus
- une configuration de « base de données » EPICS (fichier d'extension db, qui décrit des records)
- chaque record est décrit par des champs EPICS qui représentent eux-mêmes une PV
- une configuration des paramètres du protocole dans un fichier .cmd

Fichier cmd / protocole Modbus



Fichier db

```
record(ai, "LIN:TPT-002:LIMR"){
  field(DTYP, "asynInt32")
  field(INP, "@asyn(POS_CRIO_AS_READ_0,2,1000)FLOAT32_LE")
  field(SCAN, "I/O Intr")
  field(DESC, "exit linac t? limit")
}

record(ao, "POS:TPT-001:LIM"){
  field(PINI, "YES")
  field(DTYP, "asynInt32")
  field(DESC, "bunket t? limit")
  field(OUT, "@asyn(POS_CRIO_AS_WRITE_0,0,1)FLOAT32_LE")
}

record(bi, "LIN:FLIN:FT-007"){
  field(DTYP, "asynUInt32Digital")
  field(INP, "@asynMask(POS_CRIO_TE_READ_0,0,0x0001,1000)MODBUS_DATA")
  field(SCAN, "I/O Intr")
  field(DESC, "RESERVE")
}

record(bo, "LIN:LIN:LIN-001:IV"){
  field(PINI, "YES")
  field(DTYP, "asynUInt32Digital")
  field(DESC, "LINAC interlock from CRIO")
  field(OUT, "@asynMask(POS_CRIO_TS_WRITE_0,0,0x0001,1)MODBUS_DATA")
}
```

```
# Configure serial or TCP connections
#
# Modbus/TCP example
#
#drvAsynIPPortConfigure(portName, hostInfo, priority, noAutoConnect, noProcessEos)
#noAutoConnect flag is set to 0 so that asynManager will do normal automatic connection management
#noProcessEos flag is set to 1 because Modbus over TCP does not require end-of-string processing

drvAsynIPPortConfigure("POS_CRIO0502", "127.0.0.1:502",0,0,1)

# modbusInterposeConfig(portName, linkType, timeoutMsec, writeDelayMsec)
# linkType: 0 = TCP/IP, 1 = RTU, 2 = ASCII
# timeoutMsec: The timeout in milliseconds for write and read
# writeDelayMsec: delay in milliseconds before each write from EPICS to the device (only needed for Serial RTU devices, default is 0)

modbusInterposeConfig("POS_CRIO0502",0,1000,0)

# Modbus port configuration
# drvModbusAsynConfigure(portName, tcpPortName, slaveAddress, modbusFunction, modbusStartAddress,modbusLength,dataType,pollMsec, plcType
# modbusLength : length in bits for TE (max 2000) and TS (max 1968) and in bytes for AE (max 125) and AS (max 123)
# modbusFunction = 1 : Read Coils (read TS)
# modbusFunction = 2 : Read Discrete Inputs (read TE)
# modbusFunction = 3 : Read Multiple Registers (2 bytes = one word) (read AS)
# modbusFunction = 4 : Read Input Registers (2 bytes = one word) (read AE)
# modbusDataType = 0: UINT16 Unsigned 16-bit binary integers; 7 = FLOAT32_LE ; 8 = FLOAT32_BE : 32-bit floating point
# TYPE_BE Big Endian (most significant word at Modbus address N, least significant word at Modbus address N+1)
# TYPE_LE Little Endian
# pollMsec: period delay in millisecond of the request
# plcType : This parameter is currently used to print information in asynReport.

#
# portName, tcpPortName, slaveAddr, modbusFct, addr, length, dataType, pollMsec, plcType
drvModbusAsynConfigure("POS_CRIO0502_AE_READ_0", "POS_CRIO0502", 1, 4, 0, 125, 7, 1000, "SIEMENS")
drvModbusAsynConfigure("POS_CRIO0502_AE_READ_125", "POS_CRIO0502", 1, 4, 125, 7, 1000, "SIEMENS")
drvModbusAsynConfigure("POS_CRIO0502_TE_READ_0", "POS_CRIO0502", 1, 2, 0, 14, 0, 1000, "SIEMENS")

# modbusFunction = 5 : Write Single Coil (write TS)
# modbusFunction = 16 : Write Multiple Holding Registers (2 bytes = one word) (write AS)
# pollMsec: For write functions, a non-zero value means that the Modbus data should be read once when the port driver is first created.
#
# portName, tcpPortName, slaveAddr, modbusFct, addr, length, dataType, pollMsec, plcType
drvModbusAsynConfigure("POS_CRIO0502_AS_WRITE_0", "POS_CRIO0502", 1, 16, 0, 36, 7, 1, "SIEMENS")
drvModbusAsynConfigure("POS_CRIO0502_TS_WRITE_0", "POS_CRIO0502", 1, 5, 0, 68, 0, 1, "SIEMENS")
```

L'écosystème EPICS

Voir catalogue EPICS : <https://docs.epics-controls.org/en/latest/software/epics-related-software.html>

Certains services sont fournis au travers de modules EPICS :

- Autosave <https://github.com/epics-modules/autosave>
- Sscan <https://epics-modules.github.io/sscan/sscanDoc.html>

D'autres services partagent une configuration définie à la fois côté IOC et à la fois côté client :

- Service d'archivage (Archive Appliance) = fréquence d'archivage (IOC & Appliance)
- Gestionnaire d'alarmes (CS-StudioAlarm) = seuils d'alarme et sévérité (IOC), historique (CS-Studio)

IOC : Fichier db, champs SCAN

```
record(ai, "LIQUE4008:H2_CH0_INSTANT")
{
    field(DESC, "CR_VALREMP Lique 4008 Muscade PV")
    field(SCAN, "1 second")
    field(CALC, "A")
    field(INPA, "CR_VALREMP CP NMS")
}
```

Paramétrage archivage dans Appliance

PV Name	Status	Appliance	Connected?	Monitored?	Sampling period
LIQUE4008:H2_CH0_INSTANT	Being archived	appliance0	true	true	3.0
LIQUE4008:H2_CH0_TOTAL	Being archived	appliance0	true	true	1.0
LIQUE4008:H2_CH1_INSTANT	Being archived	appliance0	true	true	1.0

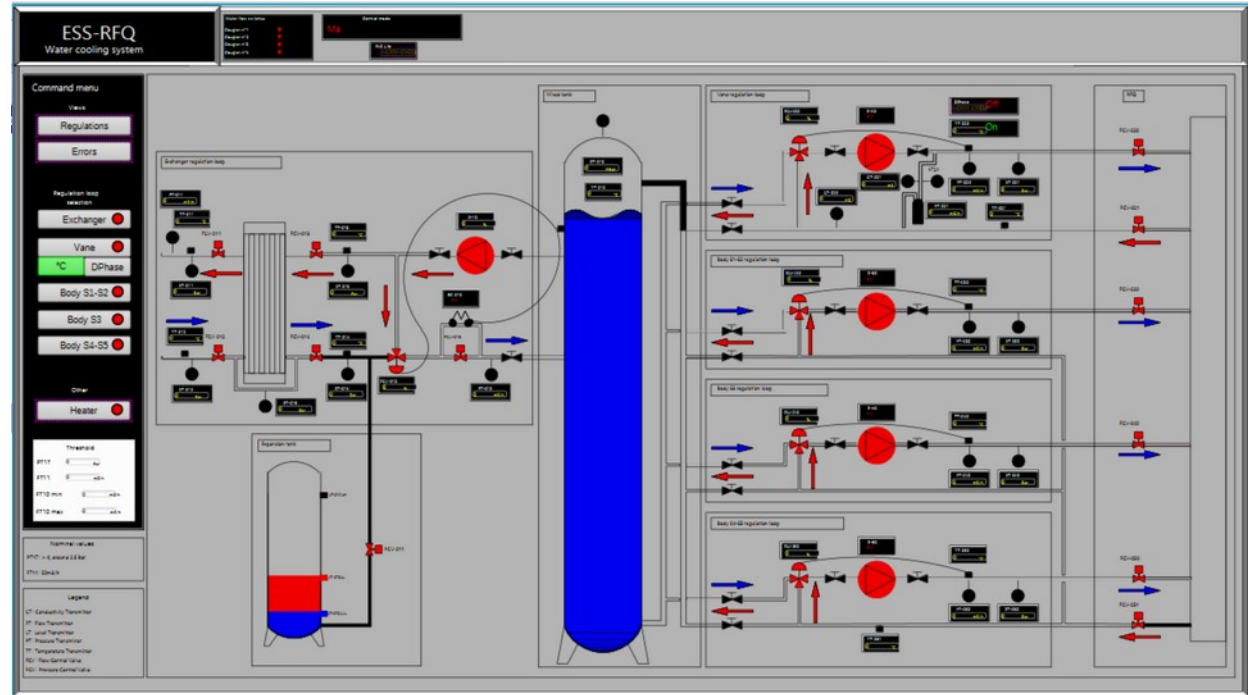
IOC Fichier db, champs HIHI...

```
field(HIHI, "100")
field(HIGH, "80")
field(LOW, "50")
field(LOLO, "20")
field(HHSV, "MAJOR")
field(HSV, "MINOR")
field(LSV, "MINOR")
field(LLSV, "MAJOR")
```

Alarm Table x			
	PV	Description	Alarm Severity
?	CF_CU:PWSS_XXLA8610Red:Sts	*Attention. CNMS grinder pahmp station level alarm	UNDEFINED
!	HEBT_Diag:MPS_BLM_Sum1	* HEBBIT and LINAC DAHMP BLM trip	MAJOR
?	HEBT_HPRF:Mod1:PSN_Flt	* HEBT modulator personnel protection fault	UNDEFINED
!	ICS_MPS:FPAF_LDmp:FPAF_MEB...	* mps fault	MAJOR
!	ICS_Tim:MPS:AR	* mps fault	MAJOR
!	SCL_Diag:MPS_BLM11t25_Sum1	* SCL BLM trip	MAJOR

- pour réaliser des synoptiques par drag&drop

CS-Studio Phoebus (Java FX)



Architecture, concepts : serveurs



Device TANGO : objet représentant une fonctionnalité (matériel, traitement...), interface commune pour tous les devices

API : C++, Java , Python

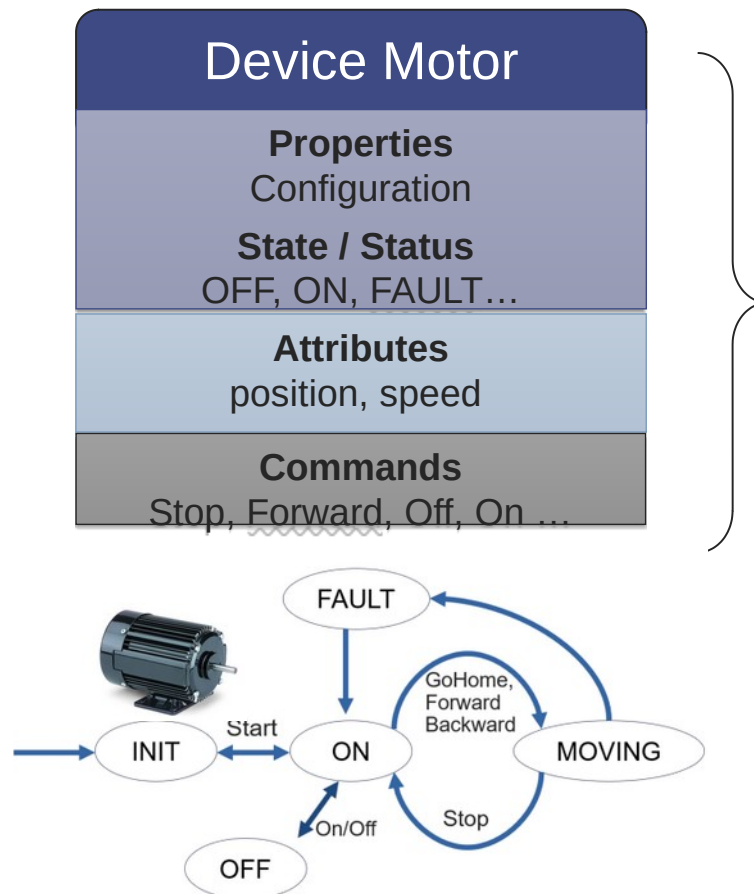
Espace de nom unique : nom unique indexé dans un annuaire des devices (device database) → introspection

Convention de nommage : domain/family/member ex SOLEIL : ANS-C01/VI/VacuumValve.1

Machine à état : pour autoriser/interdire l'exécution de chaque commande TANGO

Les 14 états Tango

Valeur	Etats	Couleur	Visu/RVB
0	ON	Vert	0,255,0
1	OFF	Blanc	255,255,255
2	CLOSE	Blanc	255,255,255
3	OPEN	Vert	0,255,0
4	INSERT	Blanc	255,255,255
5	EXTRACT	Vert	0,255,0
6	MOVING	Bleu	128,160,255
7	STANDBY	Jaune	255,255,0
8	FAULT	Rouge	255,0,0
9	INIT	Gris	204,204,122
10	RUNNING	Bleu	128,160,255
11	ALARM	Orange	255,140,0
12	DISABLE	Rose	255,0,255
13	UNKNOWN	Gris	155,155,155



Type d'attributs :

Read, Write, Read_Write

Propriétés d'attributs :

Unit, Format

Min , Max, Seuil min, Seuil max

Qualité d'attributs :

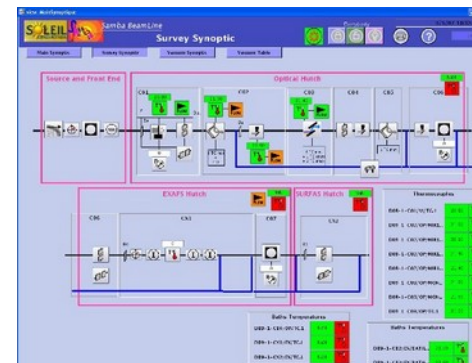
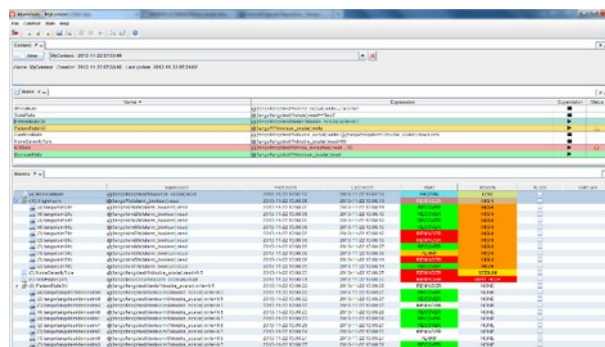
Qualité	Couleur	Visu/RVB
INVALID	Gris	128,128,128
ALARM	Orange	255,200,0
WARNING	Orange	255,200,0
CHANGING	Bleu	128,160,255
VALID	Vert	0,255,0
UNKNOWN	Gris	128,128,128

Architecture, concepts



Application de haut niveau **IHM**

Interface Humain Machine
graphique/ligne de commande



Bus logiciel de communication



Device

Process

Matériel

$$\frac{\Delta NC - \Delta}{\Delta NC - \Delta FC}$$



Monochromateurs, fentes ...



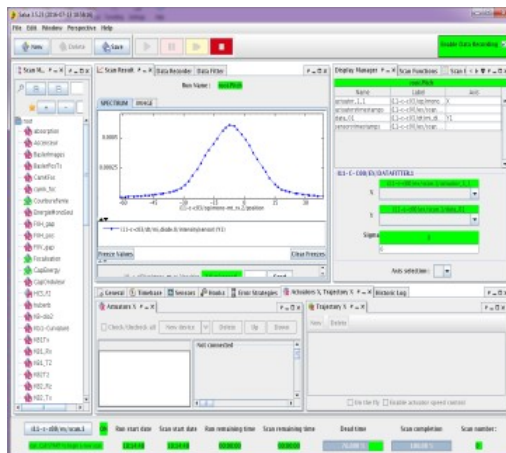
Moteurs, PLC , compact PCI ...

L'éco système TANGO

Software - TANGO Controls

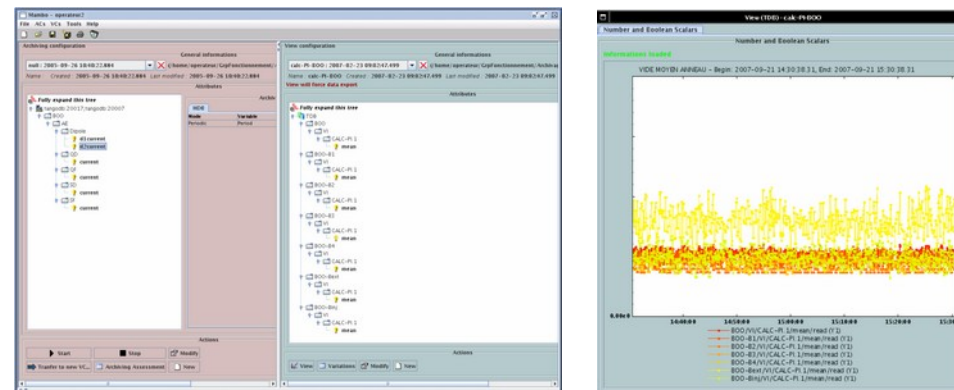


Salsa / device ScanServer Application de Scan



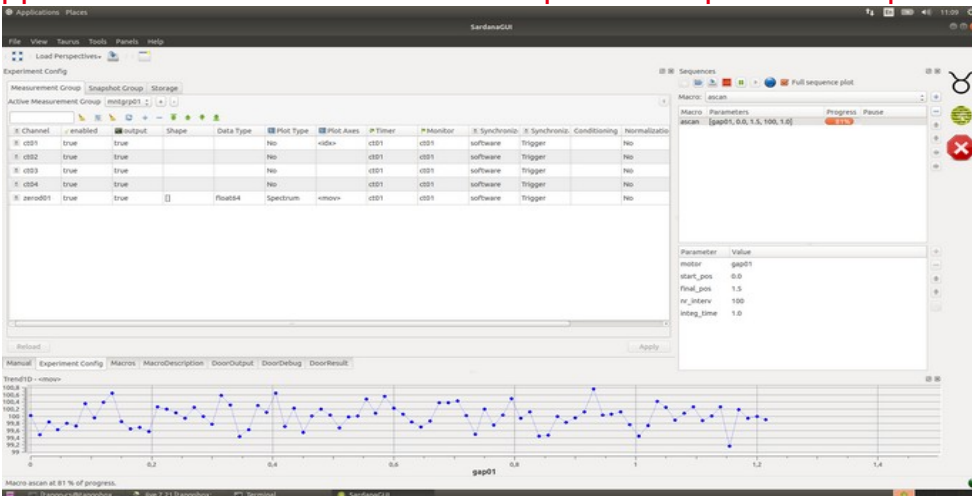
Mambo / device Hdb/TdbArchiver (Java)

Mode d'archivage : périodique, absolu, relatif, sur changement
Voir présentation Jean-Claude Marrucho (ThomX)



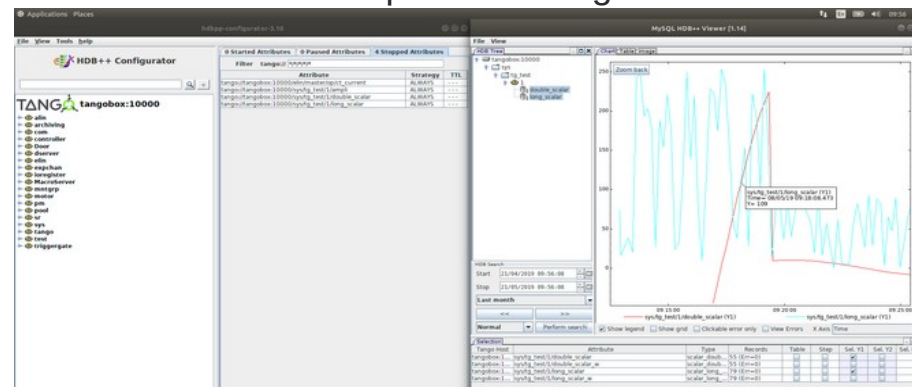
Sardana

Application de scan : lecture valeur après des opérations répétées



HDB++

Archivage : enregistrement de valeurs à une fréquence donnée, ou lorsqu'elles changent...



IHM

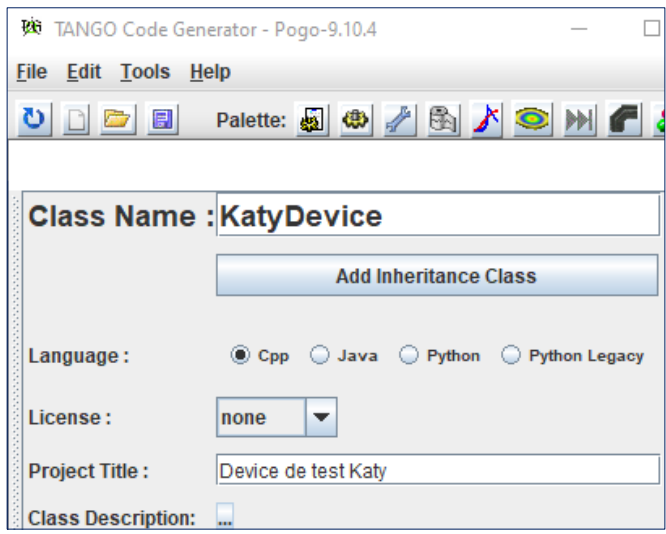
Client légers via API REST

Outils d'administration et clients génériques



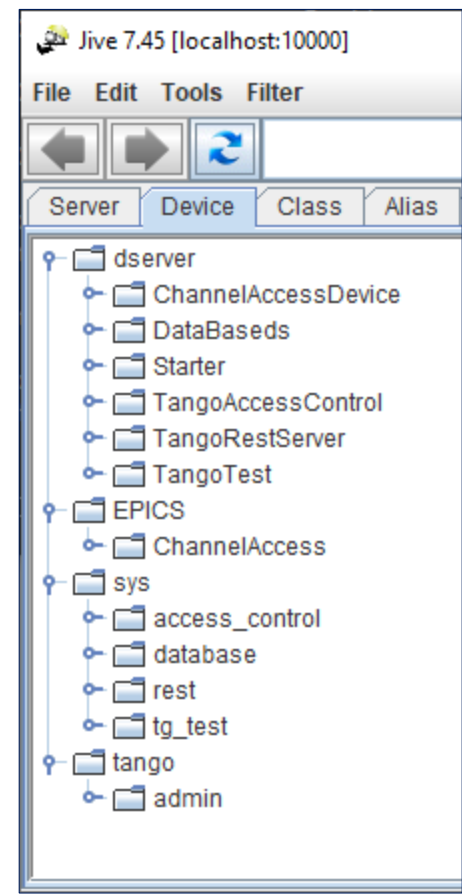
Pogo

Générateur de squelette de Code de device TANGO en C++ , Java ou Python avec machine à état



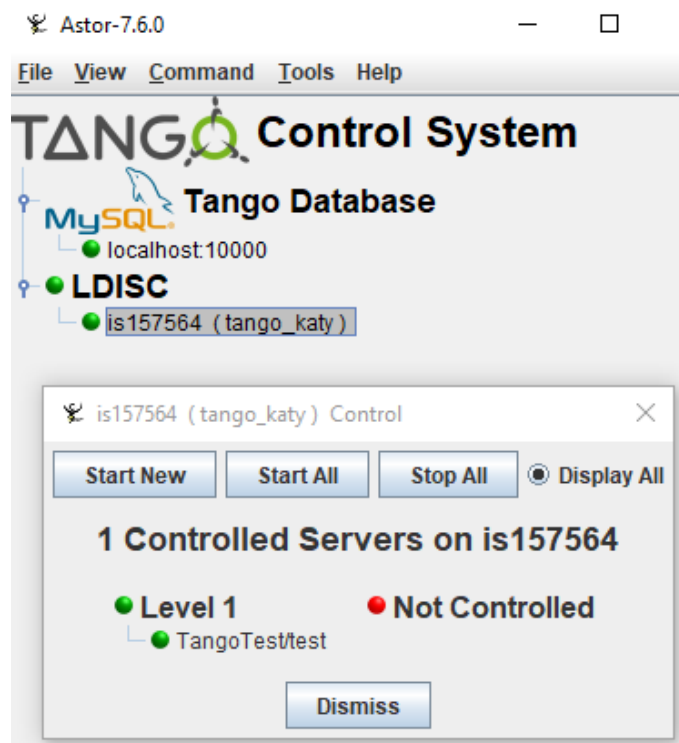
Jive

IHM de configuration basée sur le device Database



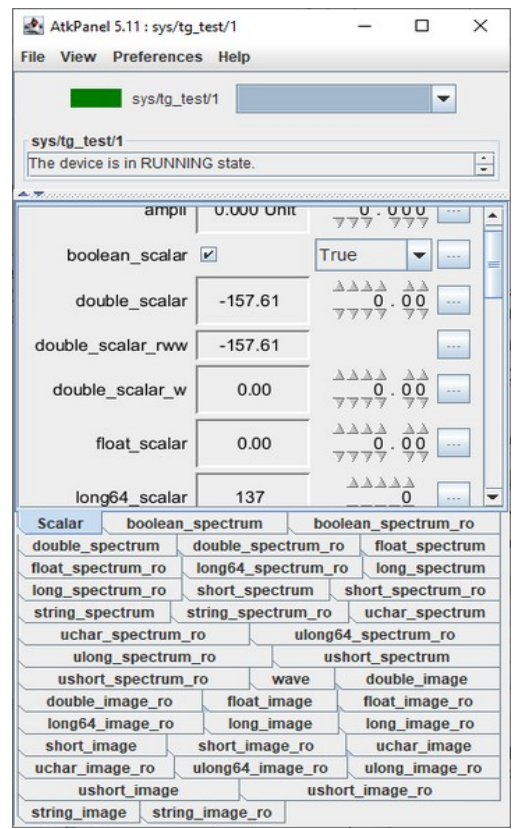
Astor

IHM d'admin de devices basée sur le device Starter (démarrage, arrêt, stat...)



ATKPanel

IHM générique de pilotage de l'ensemble des attributs/commpandes pour chaque device TANGO




Exemple de développement d'un device



Le développement d'un device TANGO suit l'approche orienté objet , exemple avec un Keithley Electrometer. Multimètre qui utilise le protocole GPIB pour la communication.

- 1. identification du protocole, 42 devices TANGO pour la communication (voir le *catalogue*)
- 2. vérification de l'existence du device TANGO (863 devices) voir le *catalogue des devices servers*
- 3. vérification de l'existence d'une classe abstraite (9 classes) dans le *catalogue*
- 4. génération du squelette de code avec POGO et développement
- 5. déploiement et configuration du device avec Jive

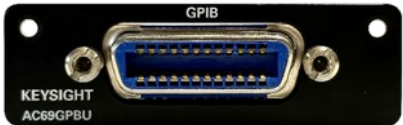
<https://www.tango-controls.org/developers/dsc/families/?family=Communication>




About usCommunitySoftware

Below you will find device servers implementing classes of Commun

Device Server	Family	Manufacturer	Products
Acs16	Communication	Avocent	acs16
EigerReque stsCtrl	Communication		
Gpib	Communication		



<https://www.tango-controls.org/developers/dsc/families/?family=MeasureInstruments#ds-table>



About usCommunitySoftwarePartners

Below you will find device servers implementing classes of MeasureInstruments family.

Device Server	Family	Manufacturer	Products
Keithley2001	MeasureInstruments	Keithley	2001
Keithley2182A	MeasureInstruments	Keithley	2182A
Keithley2602B	MeasureInstruments	Keithley	2602B

Jive 7.45 [localhost:10000]

File Edit Tools Filter

ANS-16/DIAG/Multimeter.1

Device properties

Property name	Value
CtrlDevice	ANS-16/DIAG/GPIB.1
GpibBoardName	BOARD1
GpibDeviceAddress	198.0.0.1

RefreshApplyNew propertyCopyDelete

KeithleyElectrometers

Properties :
CtrlDevice

Attributes :
Voltage
Intensity

GpibDevice
ANS-16/DIAG/Multimeter.1

Properties :
GpibDeviceAddress
GpibBoardName

Commands :
Read
Write

Des ponts entre les deux mondes

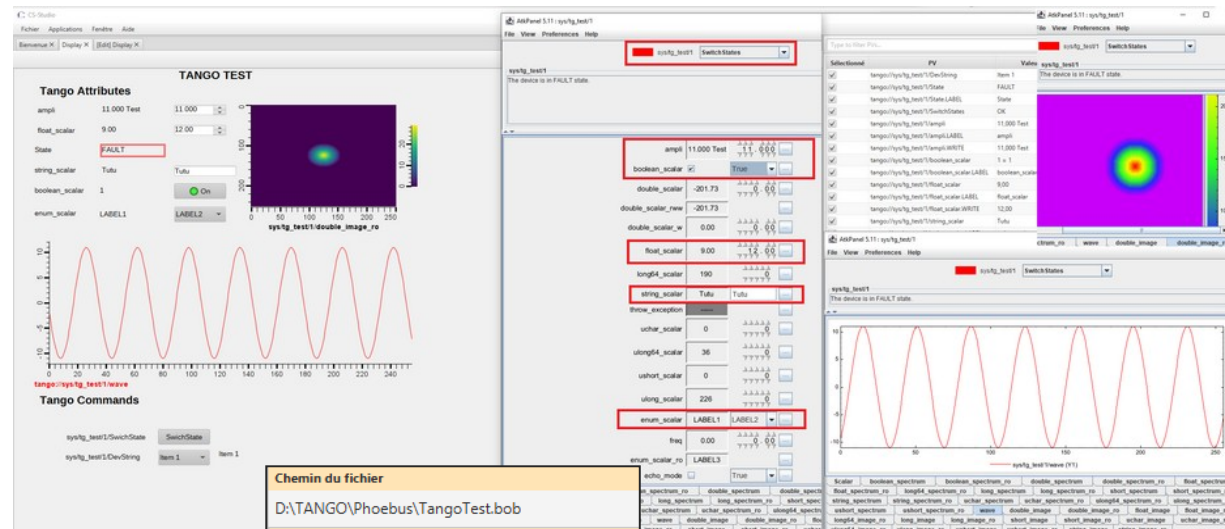
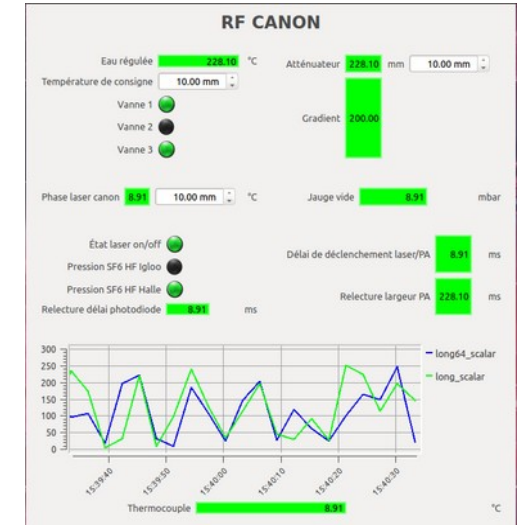
Passerelles pour intégrer un matériel piloté EPICS dans Tango

- Tango2Epics
- TangoEpics (qui est différent !)

Passerelle pour intégrer un matériel piloté Tango dans EPICS : epics-tango-bridge

IHM :

- Taurus : API pour IHM Epics et Tango
- CS-Studio-Phoebus IRFU : CA, PVA, MQTT et TANGO (fiabilisé récemment)
- Voir démonstration vidéo CS-Studio pour TANGO



```
# ----- Predefined colors -----
# May be overridden in here
# TANGO DevState Color
ON/OPEN/EXTRACT = 0,255,0,255
OFF/CLOSE/INSERT = 255,255,255,255
MOVING/RUNNING = 128,160,255,255
STANDBY = 255,255,0,255
FAULT = 255,0,0
INIT = 204,204,122,255
ALARM = 255,140,0,255
DISABLE = 255,0,255,255
UNKNOWN = 155,155,155,255

# TANGO Attribute Quality Color
INVALID_ATTR/UNKNOWN_ATTR = 128,128,128,255
ALARM_ATTR/WARNING_ATTR = 255,200,0,255
CHANGING_ATTR = 128,160,255,255
VALID_ATTR = 0,255,0,255
```

```
settings.ini
# Default PV Type in TANGO
org.phoebus.pv.default=tango
# Default tango_host get from en variable TANGO_HOST
#org.phoebus.pv.tango/tango_host=$(TANGO_HOST)
#org.phoebus.pv.tango/tango_host=tangodb:20000

#org.csstudio.display.builder.model/color_files=color.def
org.csstudio.display.builder.model/color_files=tangocolor.def
```

Expression booléenne	Expression de valeur
pv0 == 0	ON/OPEN/EXTRACT
pv0 == 1	OFF/CLOSE/INSERT
pv0 == 2	OFF/CLOSE/INSERT
pv0 == 3	ON/OPEN/EXTRACT
pv0 == 4	OFF/CLOSE/INSERT
pv0 == 5	ON/OPEN/EXTRACT
pv0 == 6	MOVING/RUNNING
pv0 == 7	STANDBY
pv0 == 8	FAULT
pv0 == 9	INIT
pv0 == 10	MOVING/RUNNING
pv0 == 11	ALARM
pv0 == 12	DISABLE
pv0 == 13	UNKNOWN



Forces :

- consensus sur les modules et outils clients : StreamDevice, Asyn, PyEPICS, CS-Studio, Appliance
- un peu plus ancien et davantage adopté par les accélérateurs : ESS, Diamond, DESY, SARAF...
- partage d'expertise, et réutilisation des développements faits par chaque institut.

Faiblesses (détails en annexe)

- besoin d'une grande expertise EPICS pour développer, installer et déployer un IOC .
- pas d'index de la liste des IOC et PV (pas d'introspection, pas de nom unique)
- pas de notion de métadonnées, par ex : unité, description, format, chaque métadonnée est une PV
- une PV , ne peut être que read ou write, mais pas les 2 (complexifie énormément l'IHM)
- rarement déployé sur Windows (du moins à ma connaissance)
- les services EPICS demandent une expertise solide dans d'autres technologies



Forces :

- interface commune des devices TANGO → simplifie le dév. des IHM (introspection, state, status ...)
- IHM d'admin. et clients → accélère les phases de test et de commissionning
- centralisation annuaire devices → simplifie l'archi réseau car seulement un TANGO_HOST à définir
- déploiement des paquets TANGO facile, sans expertise Tango
- installation TANGO sur tout système GNU/Linux et Windows
- déploiement/configuration d'un device TANGO sans expertise (Jive)
- développement de devices accessibles à des scientifiques (Pogo).
- consensus sur un nb limité d'états TANGO+ code couleur ATK → homogénéité des IHM
- composante read et write d'attribut → différencier valeurs de consignes et de lecture

Faiblesses :

- développement très accessible → plusieurs outils d'IHM et de services
- plus de développement d'outils en parallèle pour les mêmes fonctionnalités (outils)
- déployé principalement en Europe (mais SKA)
- un peu moins connu/ancien que EPICS, et ses forces ne sont pas connues de la communauté EPICS

Conclusion



Force des 2 SCADA

Des communautés bienveillantes, actives et collaboratives

Faiblesse des 2 communautés

Peu d'échange entre les 2 communautés



Retex Tango personnel IJCLab en tant que développeur haut niveau (et qui débute sur EPICS)

- Entrée dans le projet
 - déploiement d'une salle maquette/test facilité par les paquets Debian
 - la doc évolue, les rencontres résolvent bcp de soucis, les listes de discussion sont peu utilisées (appréhension) mais Mattermost est bien plus réactif
- Au quotidien
 - outils qui permettent de savoir quand a été modifié un attribut (« non je n'ai touché à rien »)
 - outils de config et admin qui permettent de déléguer une partie des tâches pour se concentrer sur le dév
 - communauté réactive, possibilité d'échange en français
 - archivage : préférer la solution majoritaire HDB++ (support++), qui permet d'installer le SGBD souhaité
 - métaDS : Lima (caméra/détecteur) au déploiement simplifié avec paquets conda, IcePap (moteurs) permet le pilotage de nombreux moteurs
- Développements
 - dév d'IHM pas initialement prévu mais réalisé grâce à l'uniformisation de la techno (Taurus=Python+Qt) + bon support
 - dév. aidé par Pogo et contrôle fin avec machine à état (permet d'interdire cmd dans certains états)
- Comme pour tout SCADA : remonter les pb aide à
 - 1) savoir si notre usage est améliorable
 - 2) savoir si une solution temporaire existe
 - 3) obtenir une solution + pérenne que le patch manuel archivage

Retex IRFU/SOLEIL

Retour d'expérience personnel IRFU en tant que développeur haut niveau

Besoin d'une grande expertise EPICS pour développer, installer et déployer un IOC EPICS :

- demande des compétences en admin système,
- demande une maîtrise des drivers principaux : Stream Device, Asyn pour développer un module
- développements de nouveaux modules rares : ex S7CEA IRFU
- le nom des champs EPICS n'est pas intuitif : ex EGU=unité , ZNAM=Valeur textuelle de la PV lorsque =0 ...
- les contraintes sur les noms de PV, ne se découvrent qu'au runtime d'une PV (interdit de mettre des « . » , « / »)
- documentation difficile à trouver, et peu compréhensible
- temps de « maîtrise » du framework plutôt long , (1 semaine pour installer une base EPICS sans module)

Pas de base de données pour indexer la liste des IOC et des PV :

- l'introspection des IOC pourrait permettre de faire des IHM génériques comme Jive et ATKPanel
- 2 PV qui portent le même nom peuvent co-exister sur le réseau, ce qui est potentiellement dangereux

Il n'y a pas de notion de méta donnée, ex : unité, description, format, chaque méta data est une PV :

- pour construire une IHM, il faut lire autant de PV que de méta donnée = Cela génère des problèmes mémoires dans CS-STUDIO
- le nom des PV et la description des PV ont une limitation en nombre de caractères (40)

Une PV EPICS, ne peut être que read ou write, mais pas les 2 (ça complique énormément l'IHM) :

- cela complique les IHM, car un champs doit être connecté à 2 PV pour lire et écrire une consigne.
- la valeur de consigne n'est pas mémorisée côté serveur (Affichage des PV EPICS en DISCONNECT tant qu'elles ne sont pas processées)
- il n'y a pas de convention de nommage pour le nom d'une PV de relecture , à l'IRFU PV consigne = name et PV lecture = nameR