
# Présentation générale d’EPICS

Stéphane TZVETKOV

stephane.tzvetkov@cea.fr

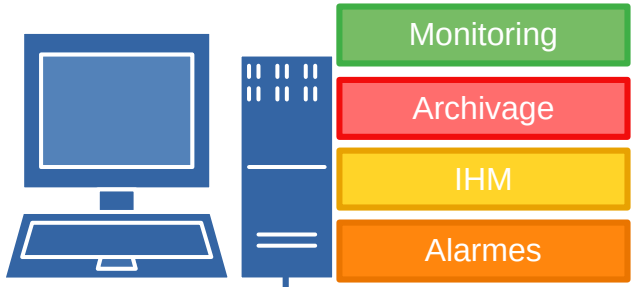
# Sommaire

1. Pourquoi ? Comment ça fonctionne et à quoi ça ressemble ?
2. Avantages et inconvénients
3. Nouveautés de la communauté
4. Perspectives

# 1. Pourquoi ? Comment ça fonctionne et à quoi ça ressemble ?

EPICS expliqué rapidemment/simplement,

# Point de départ

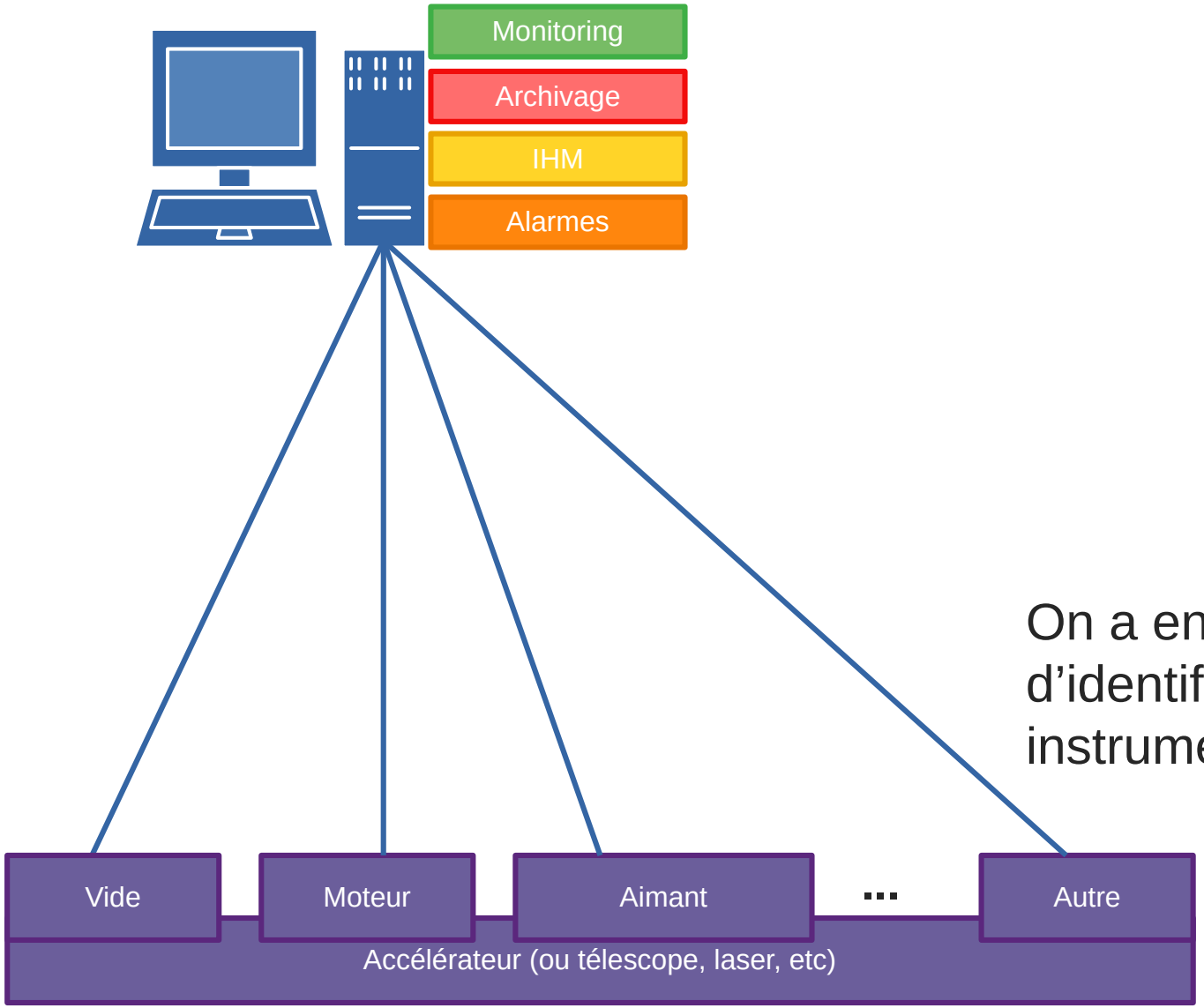



On souhaite piloter une installation.

Pour cela on a déjà besoin de plusieurs services clients...

Accélérateur (ou télescope, laser, etc)

# + les instruments

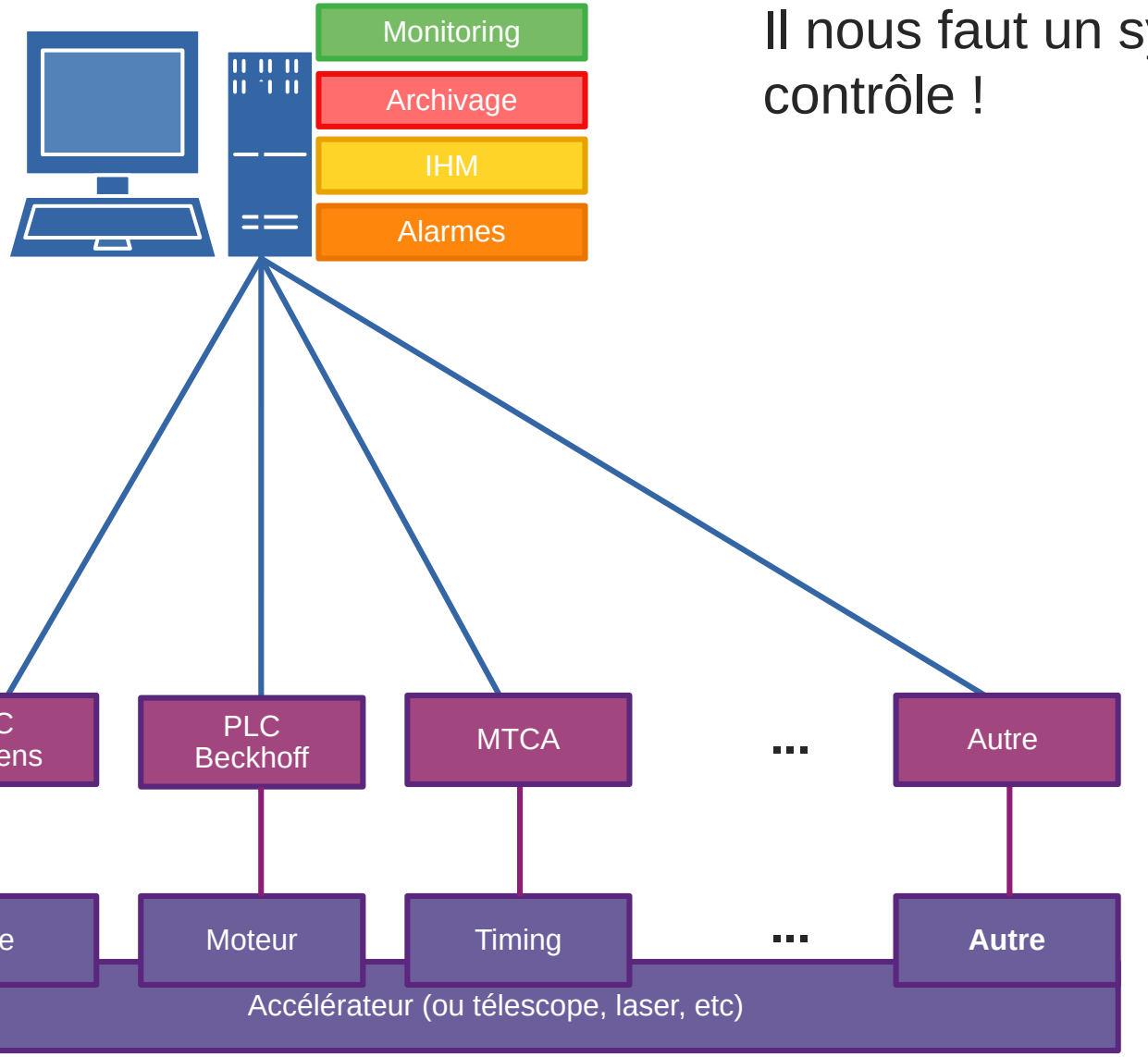



On a ensuite besoin d'identifier clairement les instruments à piloter.

# + les équipements

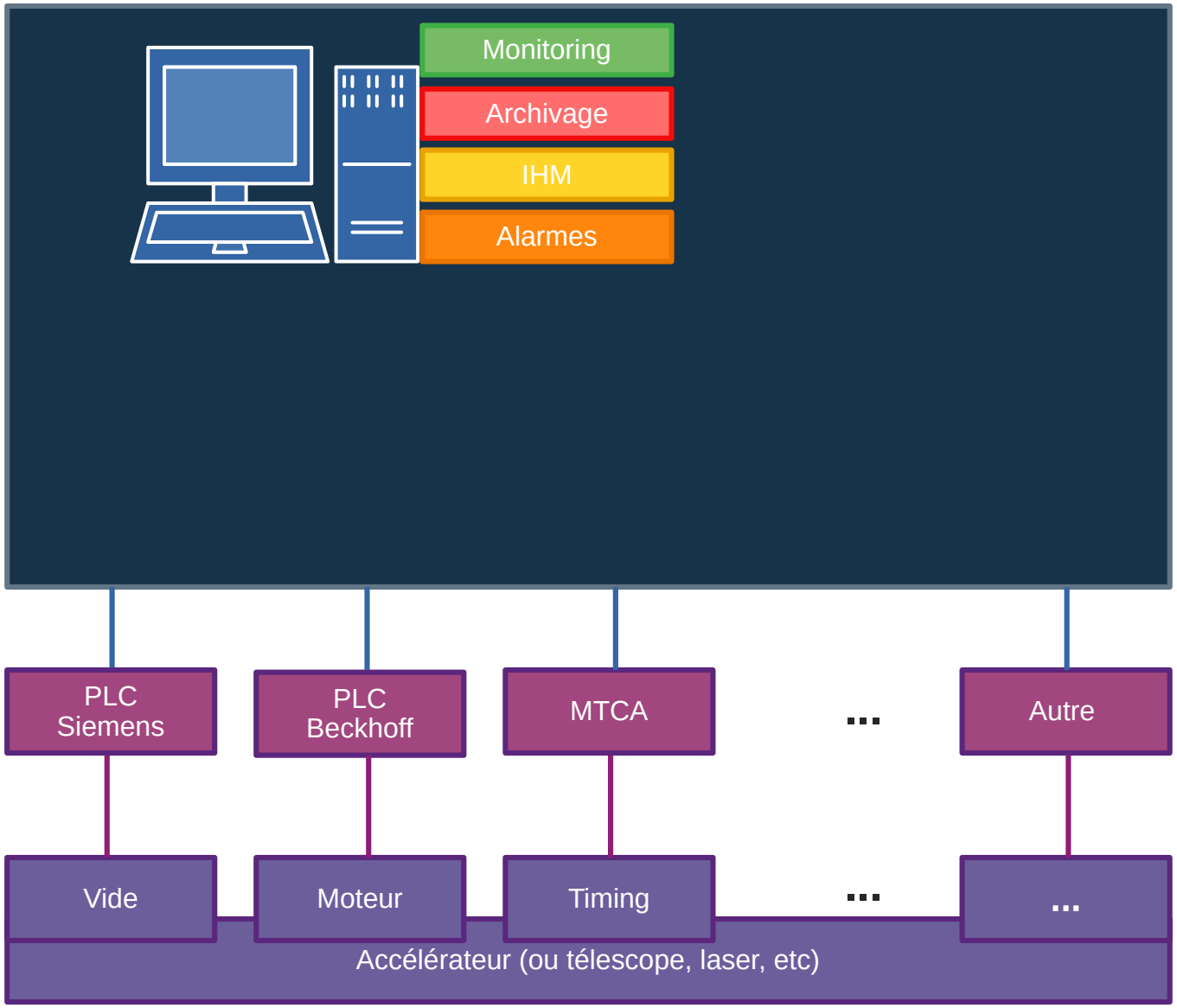
On a ensuite besoin de sélectionner le matériel intermédiaire qui permettra de s'interfacer aux équipements…

Et ici, on commence à se rendre compte que ça commence à faire beaucoup pour un seul PC client.



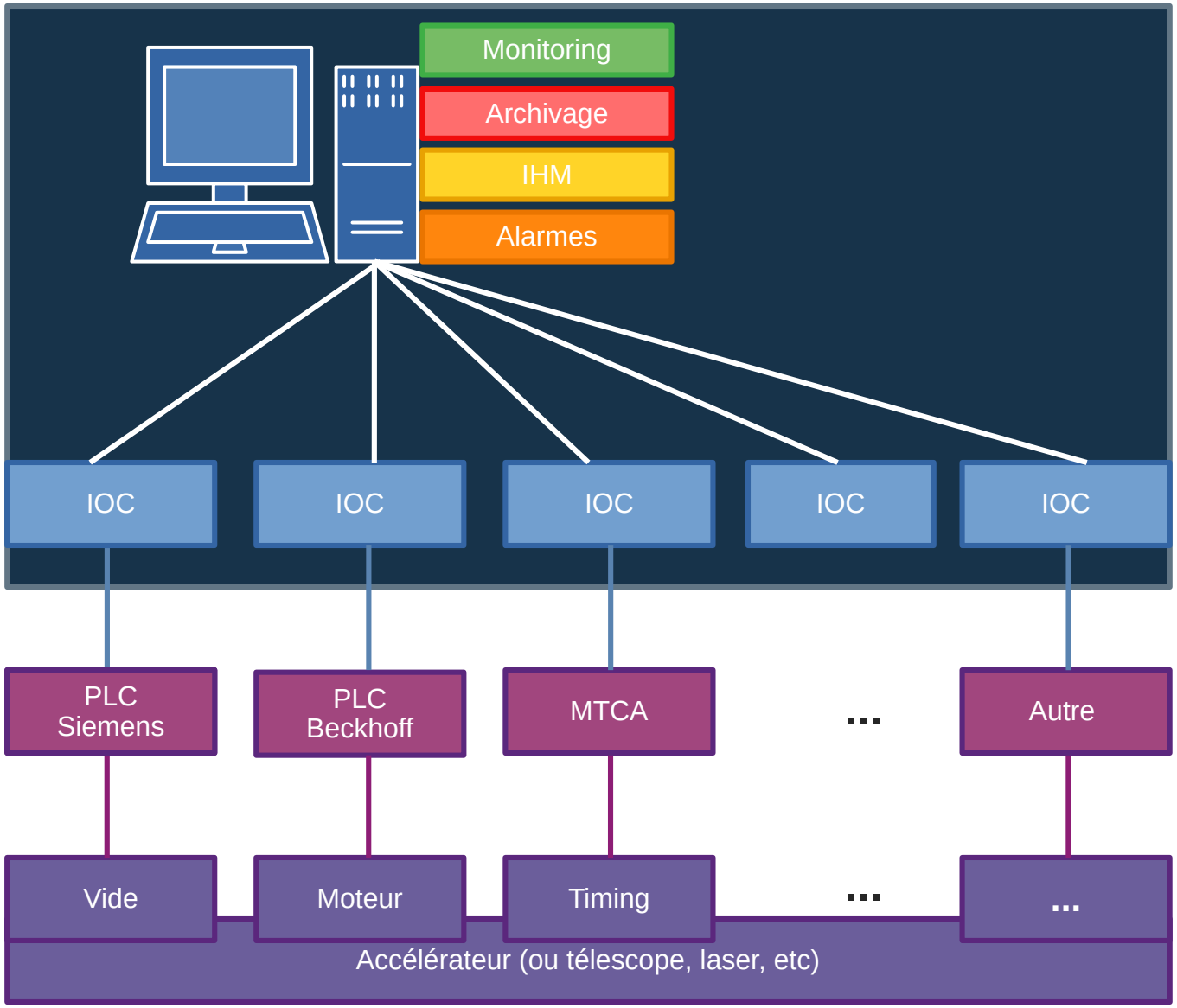

Il nous faut un système de contrôle !

# + EPICS




EPICS à la rescousse

# + les serveurs distribués

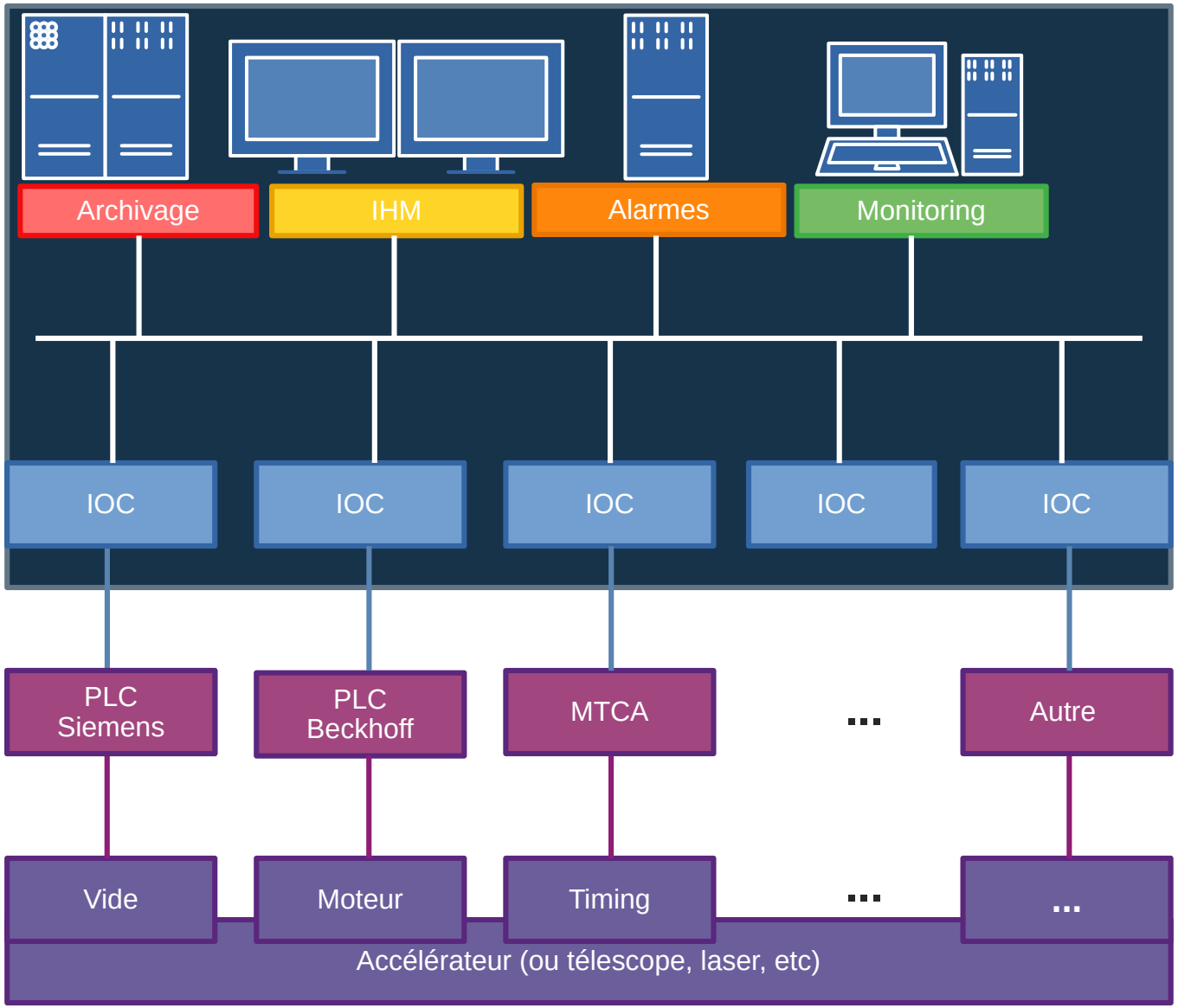



EPICS

Avec des serveurs distribués

IOC = Input/Output Controller

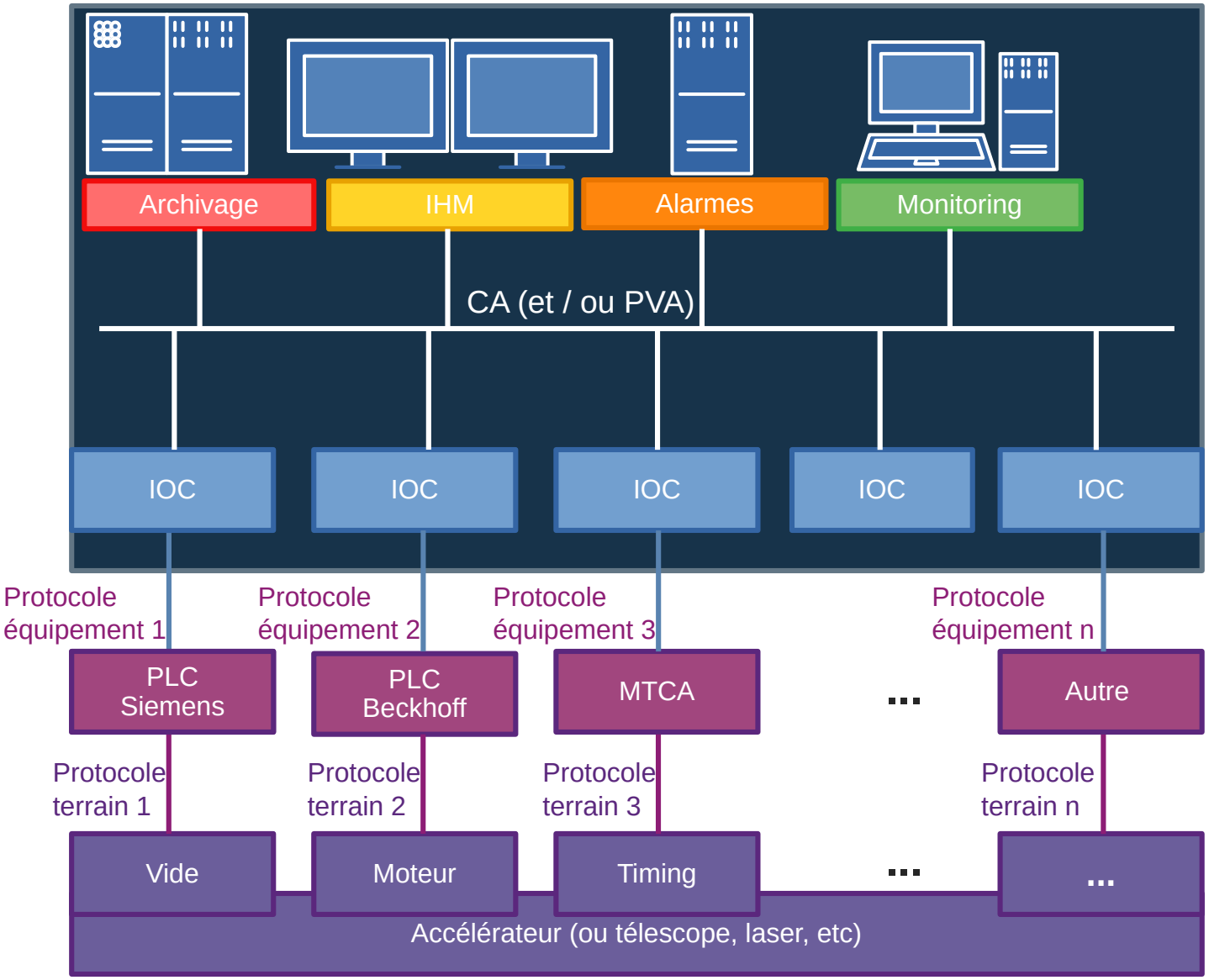
# + les clients distribués




EPICS

Et avec des clients distribués

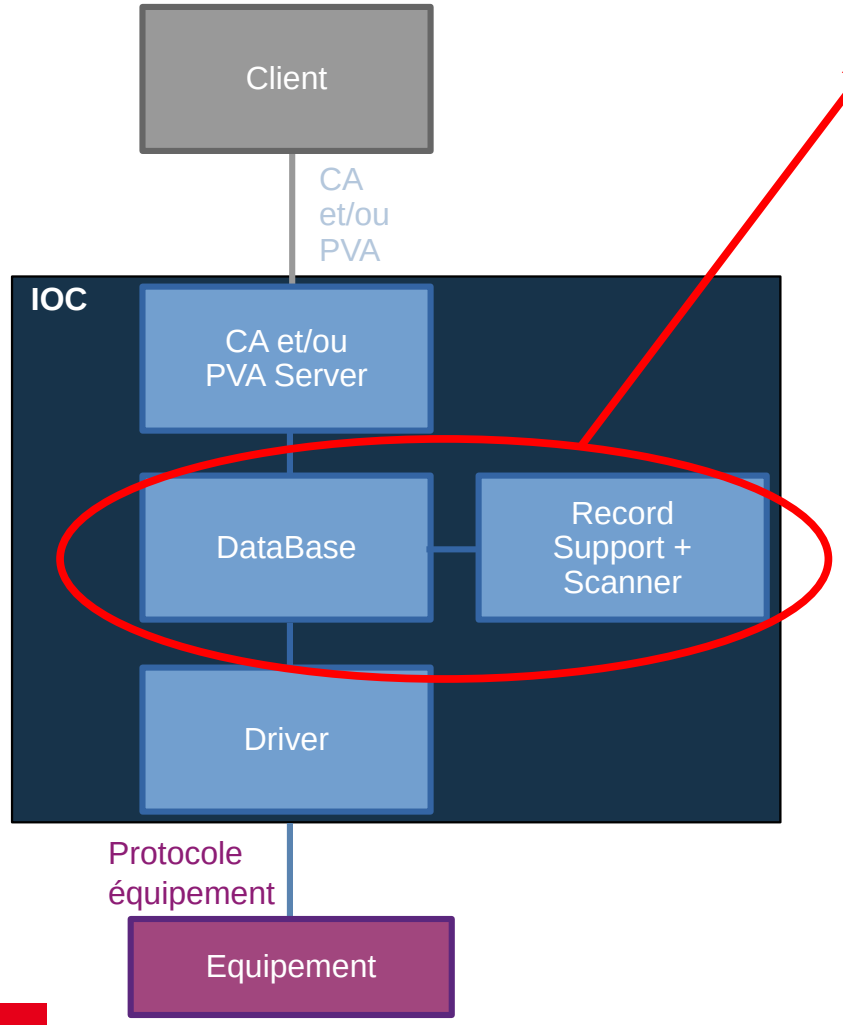
# + les protocoles de communication




EPICS

Grace à un protocole de communication commun (Channel Access
et/ou PV Access)

# Petit zoom sur l'architecture d'un IOC

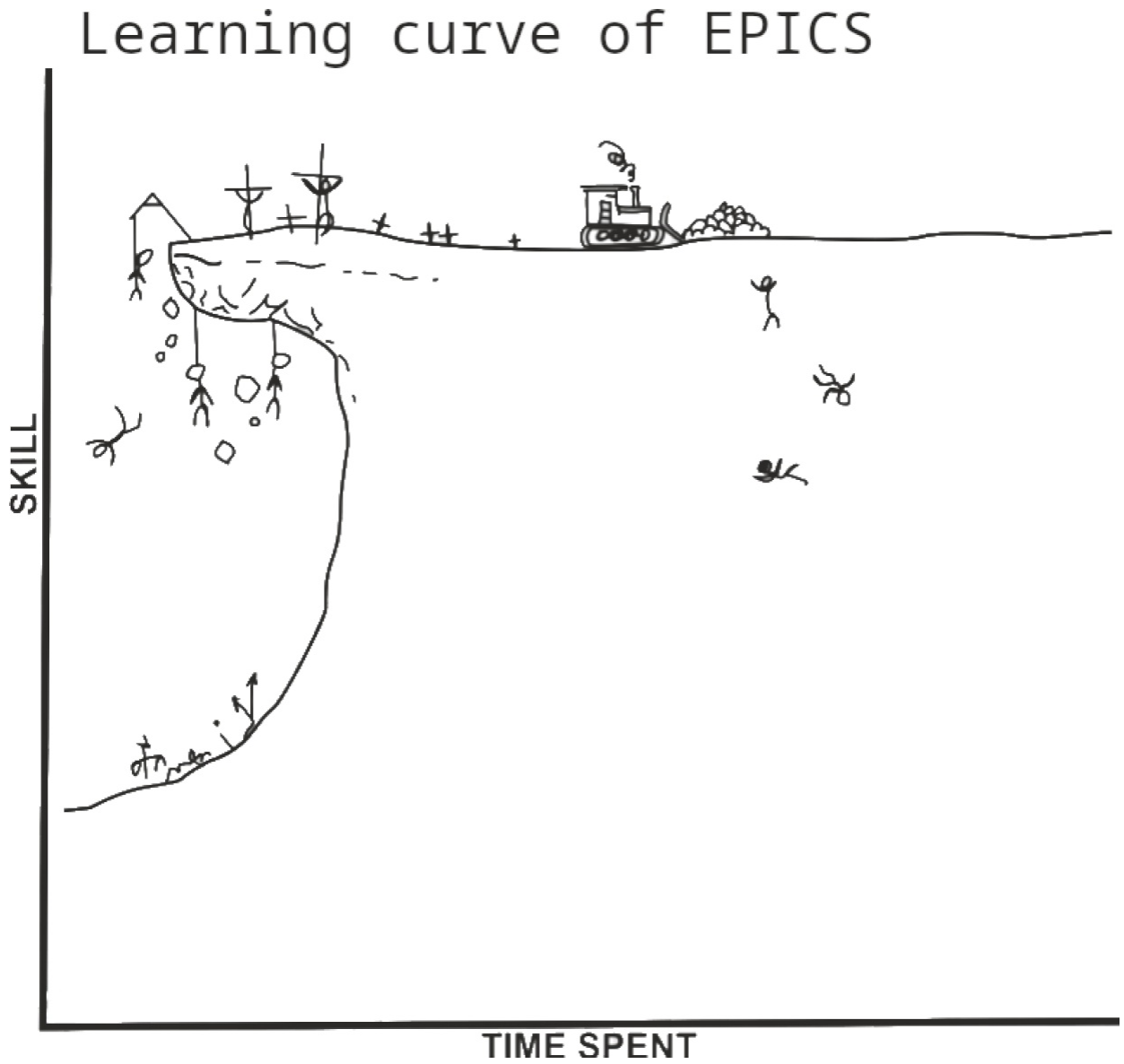



```
### Write functions

## Address 0
# Forward power set point
record(longout, "${P=CEA:}ForwdPwrSet") {
    field(DESC, "Set forward power")
    field(EGU,  "${POWER_EGU=W}")
    field(DRVL, "${POWER_MIN=0}")
    field(DRVH, "${POWER_MAX=2000}")
    field(DTYP, "asynUInt32Digital")
    field(OUT,  "@asynMask(${write=sgmp20ked-modbus-wr}, 0, 0xFFFF, 1000)MODBUS_DATA")
    field(UDF, 0)
    field(STAT,"NO_ALARM")
    field(SEVR,"NO_ALARM")
}

## Address 1
# Reflected power set point
record(longout, "${P=CEA:}ReflPwrSet") {
    field(DESC, "Set reflected power")
    field(EGU,  "${POWER_EGU=W}")
    field(DRVL, "${POWER_MIN=0}")
    field(DRVH, "${POWER_MAX=2000}")
    field(DTYP, "asynUInt32Digital")
    field(OUT,  "@asynMask(${write=sgmp20ked-modbus-wr}, 1, 0xFFFF, 1000)MODBUS_DATA")
    field(UDF, 0)
    field(STAT,"NO_ALARM")
    field(SEVR,"NO_ALARM")
}

## Address 2
# Configuration and control
record(mbboDirect, "${P=CEA:}SetupCmd") {
    field(DESC, "Configuration and control")
    field(DTYP, "asynUInt32Digital")
    field(OUT,  "@asynMask(${write=sgmp20ked-modbus-wr}, 2, 0xFFFF, 1000)MODBUS_DATA")
    field(UDF, 0)
    field(STAT,"NO_ALARM")
    field(SEVR,"NO_ALARM")
}
```

# 2. Avantages et inconvénients

Seulement deux gros avantages
et deux gros inconvénients

# Inconvénient : la courbe d'apprentissage (« un peu raide » )




- Documentation éparpillée et vieillissante
- Architecture d'un projet EPICS est "surprenante" comparé à d'autres frameworks modernes
- Méthode de développement perpendiculaire aux standards de l'industrie informatique
- Nécessite un apprentissage très détaillé, voir un peu encyclopédique

# Avantage : la communauté

- EPICS est une palette d’outils open-source développés collaborativement
- Une mailing-list ”EPICS tech-talk” très populaire (étonnant pour une mailing-list, mais je ne juge pas)
- Un chat très actif
- Des meetings réguliers (en général deux par an)
- Des ”Codeathons” et ”Documentathons” ponctuels
- EPICS est utilisé sur beaucoup de projets / au sein de nombreuses installations, dans le monde entier
- Une communauté de gens bienveillants (et compatissants)

# Inconvénient : des « standards  de programmation » datés

- EPICS se configure plus qu'il ne se développe
- EPICS multiplie les fichiers de configurations complexes et spécifiques (.db, .template, .substitutions, .cmd, .dbd, CONFIG, RELEASE, etc.)
- Par défaut, la logique ne se développe pas, elle se configure (via les records de sa base de données), ce qui devient vite inmaintenable (et frustrant)
- EPICS permet nativement de réellement développer de la logique, mais :
    - En C/C++ (avec les risques mémoires que cela implique), et pas de façon très agréable ni élégante, un peu comme si on injectait du C dans des records
    - En SNL, un langage de machine à états spécifique à EPICS, que je ne recommande pas vraiment

# Avantage : matériel et logiciel

- EPICS est très stable (testé et utilisé depuis les années 80) côté IOC et CA / PVA
- Les IOC EPICS sont très légers (même pour l'embarqué)
- L'architecture distribuée d'EPICS le rend adapté pour tout type de projet (des plus petits jusqu'aux plus grands)
- Tout un écosystème d'outils communautaires gravite autour d'EPICS et vient considérablement l'enrichir (archivage, alarmes, IHM, monitoring, etc), avec un peu de fragmentation cela dit (tout le monde n'utilisera pas les mêmes outils)
- Un grand nombre de protocoles de communications sont déjà supportés (Modbus, SNMP, S7, etc)
- EPICS peu fonctionner sur beaucoup de matériel : PC industriels, serveurs, MTCA, VME, RTEMS, VxWorks, autres cartes embarquées, etc

# Pour une comparaison plus détaillée avec d’autres systèmes de contrôle (par exemple Tango)…

Voir la présentation de Katy SAINTIN et Philippe GAURON

« TANGO & EPICS, retour d’expérience, forces et faiblesses »

Jeudi à 11h20

# 3. Nouveautés de la communauté EPICS

# CA → PVA

```
A   : 1             ACKS: INVALID       ACKT: YES           ADEL: 0
AFTC: 0             AFVL: 0             ALST: 1             AMSG: field INPL
ASG :               ASP : PTR (nil)     B   : 1
BKLNK: ELL 0 [(nil) .. (nil)]           BKPT: 00            C   : 1
CALC: A+B+C+D+E+F+G+H+I+J+K+L > 0 ? 1:0 D   : 1             DESC:
DISA: 0             DISP: 0             DISS: NO_ALARM      DISV: 1
DPVT: PTR (nil)     DSET: PTR (nil)     DTYP:               E   : 1
EGU :               EVNT:               F   : 1             FLNK: CONSTANT
G   : 1             H   : 1             HHSV: NO_ALARM      HIGH: 0
HIHI: 0             HOPR: 0             HSV : NO_ALARM      HYST: 0
I   : 1             INPA: CA_LINK ACC-RFQ:REFRIG-hydro_calc1 CP NMS
INPB: CA_LINK ACC-RFQ:REFRIG-hydro_calc2 CP NMS
INPC: CA_LINK ACC-RFQ:REFRIG-hydro_calc3 CP NMS
INPD: CA_LINK ACC-RFQ:REFRIG-hydro_calc4 CP NMS
INPE: CA_LINK ACC-RFQ:REFRIG-hydro_calc5 CP NMS
INPF: CA_LINK ACC-RFQ:REFRIG-hydro_calc6 CP NMS
INPG: CA_LINK ACC-RFQ:REFRIG-hydro_calc7 CP NMS
INPH: CA_LINK ACC-RFQ:REFRIG-hydro_calc8 CP NMS
INPI: CA_LINK ACC-RFQ:REFRIG-hydro_calc9 CP NMS
INPJ: CA_LINK ACC-RFQ:REFRIG-hydro_calc10 CP NMS
INPK: CA_LINK ACC-RFQ:REFRIG-hydro_calc11 CP NMS
INPL: CA_LINK ACC-RFQ:REFRIG-hydro_calc12 CP NMS            J   : 1
K   : 1             L   : 1             LA  : 1             LALM: 1
LB  : 1             LC  : 1             LCNT: 0             LD  : 1
LE  : 1             LF  : 1             LG  : 1             LH  : 1
LI  : 1             LJ  : 1             LK  : 1             LL  : 1
LLSV: NO_ALARM      LOLO: 0             LOPR: 0             LOW : 0
LSET: PTR 0x7cae7e0 LSV : NO_ALARM      MDEL: 0
MLIS: ELL 3 [0x7efcd8050e90 .. 0x7efcd804c9d0]
MLOK: 10 02 cc 07 00 00 00 00           MLST: 1
NAME: ACC-RFQ:REFRIG-hydro_calc0        NAMSG: field INPL   NSEV: NO_ALARM
NSTA: NO_ALARM      PACT: 0             PHAS: 0             PINI: NO
PPN : PTR (nil)     PPNR: PTR (nil)     PREC: 0             PRIO: LOW
PROC: 0             PUTF: 0             RDES: PTR 0x31ae810
RPCL: 04 05 20 06 20 07 20 08 20 09 20 0a 20 0b 20 0c 20 0d 20 0e
RPRO: 0             RSET: PTR 0x7efd1e48a4a0             SCAN: Passive
SDIS: CONSTANT      SEVR: NO_ALARM      SPVT: PTR (nil)     STAT: NO_ALARM
TIME: 2025-06-13 14:19:24.079718058     TPRO: 0             TSE : 0
TSEL: CONSTANT      UDF : 0             UDFS: INVALID       UTAG: 0
VAL : 1
```

```
structure
    structure  value
        int      index
        string[] choices
    string     descriptor
    structure  timeStamp
        long     secondsPastEpoch
        int      nanoseconds
        int      userTag
    structure   alarm
        int      severity
        int      status
        string   message
```

- Mieux spécifié
- Plus performant (déjà couramment utilisé pour les flux vidéo)
- Plus flexible (on peut créer les sous-structures que l'on veut)
- Aide à la transition prévue : depuis EPICS v7, tous les IOCs peuvent discuter à la fois en CA et en PVA

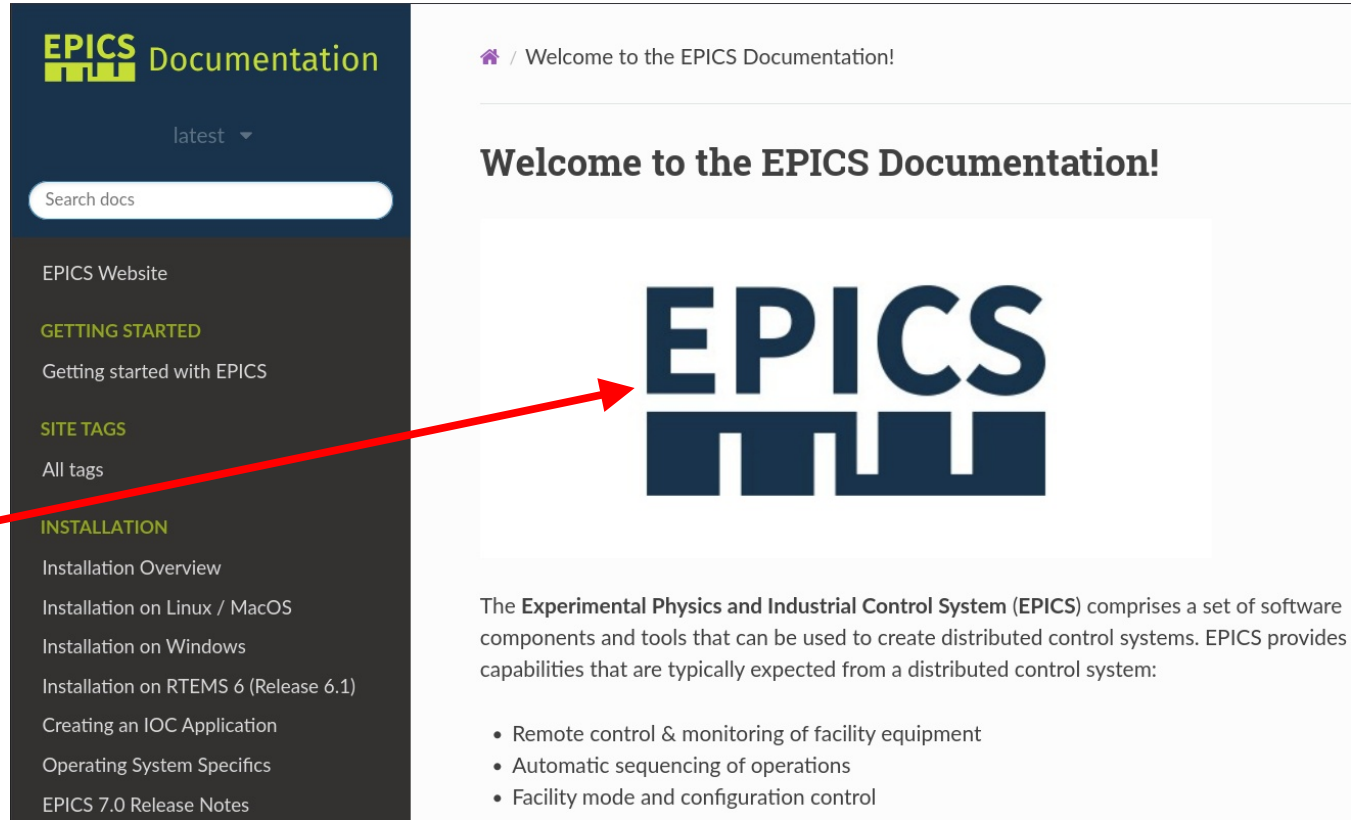
# Documentation unifiée

Next Up Previous Index

**Next:** 1. Introduction **Up:** AppDevGuide **Previous:** AppDevGuide **Index**

**Contents**






- https://docs.epics-controls.org/en/latest/
- Beaucoup de sources de documentations différentes fusionnent actuellement vers cette nouvelle documentation

# Nouveau chat

- La mailing-list ”tech-talk” est toujours utilisée
- Le chat matrix d’EPICS est cependant moins intimidant, surtout pour les débutants

EPICS

EPICS

Experimental
Industrial C

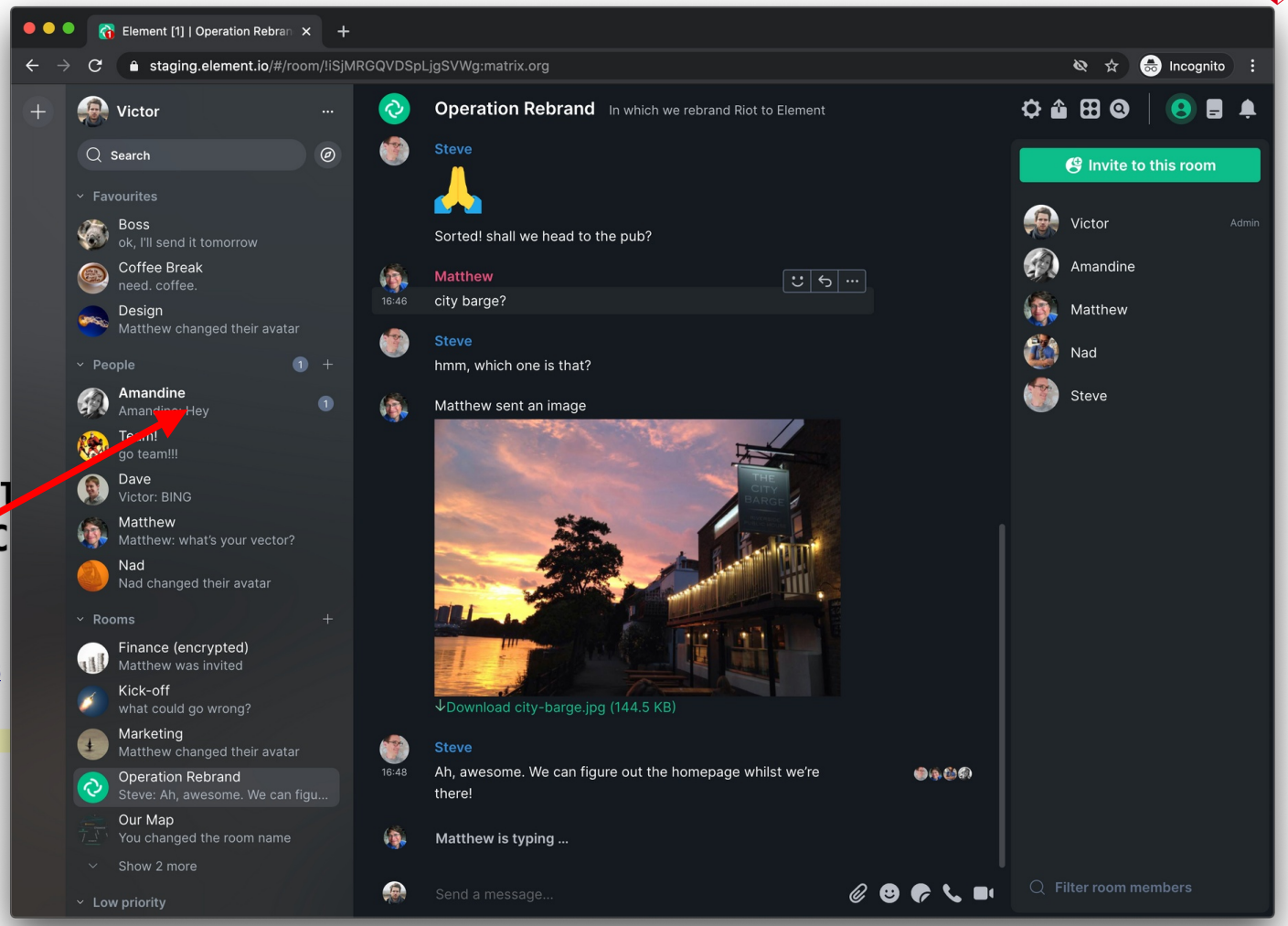
- Home
- News
- About
- Base
- Modules
- Extensions
- Distributions
- Download
- Search
- IRMIS
- Talk
  - Tech-Talk
    - Mailman
    - Search
    - 2025
    - 2024
    - 2023
    - 2022

**Tech-talk Messages by Date (by Thread)**

<1994> 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005
2020 2021 2022 2023 2024 2025

April 1994

01 Re: NI 1014 GPIB problem *winans*
04 AR data file format *greene%denali . UUCP*
Re: AR data file format *Carl Timmer*
CVS and unbundling *mrk*
05 Ascii 'db' -vs- binary 'database' file loading times *winans*
Modicon Support *Bob Dalesio*
06 Building EPICS *Matthew Needes*
Re: Release 11 installation *Jeff Hill*
arAccessLib.c *greene%denali . UUCP*
Release 11 installation *Gabor Csuka, DESY - Germany*
07 Re: arAccessLib.c *mrk*
EPICS drivers for CAN *Jeff Hill*



# OPC-UA

- Supporté depuis peu par EPICS
- Driver EPICS développé et maintenu par ITER
- Mieux sécurisé que beaucoup d’autres protocoles industriels (chiffrement + authentification par certificats)
- Performant
- Fonctionnalités timing intéressantes (timestamp sur PLC et plus sur IOC)
- Excellente intégration avec les PLC (notemment les automates Siemens) ce qui facilite le développement des automaticiens
- Olivier DELAHAYE parlera de ce sujet plus en détails demain à 9h40

# Gestionnaires de paquets
*(avez-vous déjà entendu parlé de Nix et NixOS ?)*

- Actuellement le gestionnaire de paquets EPICS le plus connu est e3 (ESS EPICS Environment : https://e3.pages.ess.eu/)
- Cela permet d'importer des modules EPICS plus facilement, de simplifier les builds, de facilité le développement, etc.
- Car EPICS n'a pas de système de packaging intégré (genre pip, maven, npm, etc)
- Depuis peu, un nouveau gestionnaire de paquets est disponible : https://github.com/epics-extensions/EPNix
- Développé par un collègue (Rémi Nicole), je fais sa publicité
- Nix est un gestionnaire de paquet utilisé par la distribution Linux NixOS
- Nix est aussi le nom d'un langage de programmation qui adopte un style déclaratif
- Si vous êtes curieux, alors je vous recommande de jeter un oeil, la documentation est excellente : https://epics-extensions.github.io/EPNix/nixos-25.05/

# 4. Perspectives

# Clients web

- … → edm → medm → CSS → Phoebus → Web ?
- Beaucoup d’outils se sont succédés pour créer et exploiter des interfaces graphiques. Aujourd’hui la tendance semble s’orienter vers les technos web
- Cela présente l’avantage de se rendre moins exposé aux changements d’outils poussés par la communauté EPICS (e.g. CSS déprécié en faveur de Phoebus)
- Olivier DELAHAYE parlera de ce sujet plus en détails demain à 16h20

# PVA (encore)

- Il est à présent possible de discuter en PVA (le nouveau protocole de communication EPICS) directement avec des langages de programmation traditionnels
- Actuellement, des library/modules disponibles en C++, Java et Pyhon
- On ne parle alors plus d'IOC mais de serveur PVA

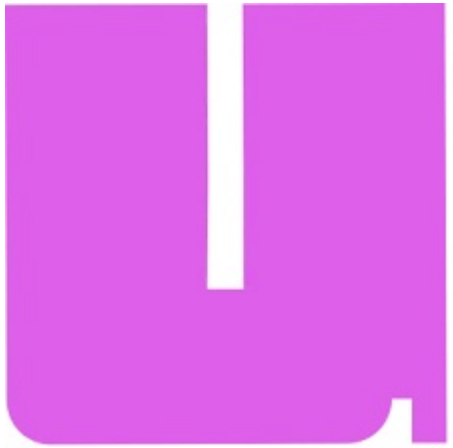
- C'est un peu révolutionnaire, car cela permet de complètement supprimer les inconvénients discutés plus tôt. EPICS permet ainsi de réaliser des développements dans les standards de l'industrie informatique. Il sera possible de développer un serveur PVA comme n'importe quel autre projet informatique « classique ».

# Plus besoin de gestionnaire de paquets spécifique à EPICS ?

Avec des développements du type « serveur PVA », donc juste en utilisant un langage de programmation unique (à l'heure actuelle : C++, Python ou Java), le gestionnaire de dépendances  du langage utilisé devrait suffire.







# Remerciements

Olivier ZIMMERMANN et tous les autres organisateurs, pour la réalisation de cet évènement

Alexis GAGET, pour m’avoir partagé des slides permettant de vulgariser EPICS

Rémi NICOLE, pour sa relecture

# Questions ?