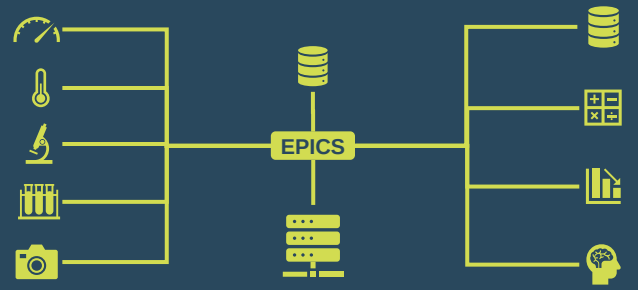
PP&B – Computer Support Group

FRITZ-HABER-INSTITUT
MAX-PLANCK-GESELLSCHAFT

# EPICS COLLECTOR

## A PYTHON-BASED MODULAR EPICS COLLECTOR FOR ON-DEMAND PV RECORDING




Simeon D. Beinlich[1,*] and Heinz Junkes[1]

[1] PP&B – Computer Support Group
Fritz-Haber-Institut der Max-Planck-Gesellschaft, Berlin, Germany

*beinlich@fhi.mpg.de

PP&B – Computer Support Group

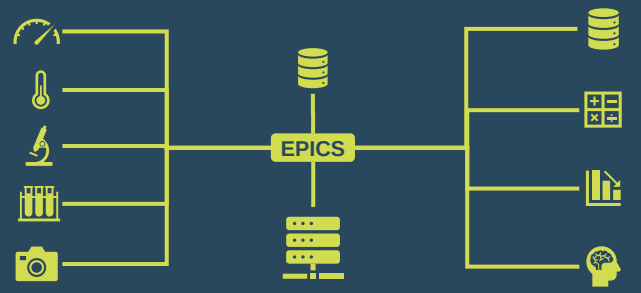
FRITZ-HABER-INSTITUT
MAX-PLANCK-GESELLSCHAFT

# EPICS COLLECTOR

A PYTHON-BASED MODULAR EPICS COLLECTOR FOR ON-DEMAND PV RECORDING

## EPICS-IN-EPICS-OUT DATA RECORDING & PROCESSING?

SIMPLE PYTHON-BASED MODULAR EPICS “DEVICES”




Simeon D. Beinlich[1,*] and Heinz Junkes[1]

[1] PP&B – Computer Support Group
Fritz-Haber-Institut der Max-Planck-Gesellschaft, Berlin, Germany

*beinlich@fhi.mpg.de

# ORIGINAL IDEA: EPICS COLLECTOR

**Record data without leaving EPICS**

- Record one or multiple PVs on-demand
- Serve history again as PV (array)
- Perform low-level tasks such as:
  - upload to Data-Management-System,
  - export to file,
  - minor processing
  - add non-EPICS data.
- Allow users to run collectors:
  - on their local machine,
  - on a remote host,
  - or both.

*… like a ‘large COMPRESS record’ …*

**Why?**

- Users can record:
  - *what they need*
  - *when they need it*
- Direct (live) access to recorded data via PVs
- No additional interfaces / APIs / protocols / data structures for accessing data.
- Treat PV Data and PV History “in the same manner”.
- Modular, distributable, scalable, extendible, …

**How?**

- Create a ‘Collector Device’ via Python + P4P
- Run it on local machine or on remote sever

# ~~ORIGINAL~~ IDEA: EPICS COLLECTOR & PROCESSING

***Why not do the same for:***

- *Data Processing*
- *Data Analysis*
- *Data Visualization*
- *AI/ML Feedback?*
- *…*

*… like a ‘large CALC record’ …*

**Why?**

- Users can process:
  - *what they need*
  - *when they need it*
- Direct (live) access to processed data via PVs
- No additional interfaces / APIs / protocols / data structures for accessing data.
- Treat all Data “in the same manner”.
- Modular, distributable, scalable, extendible, …

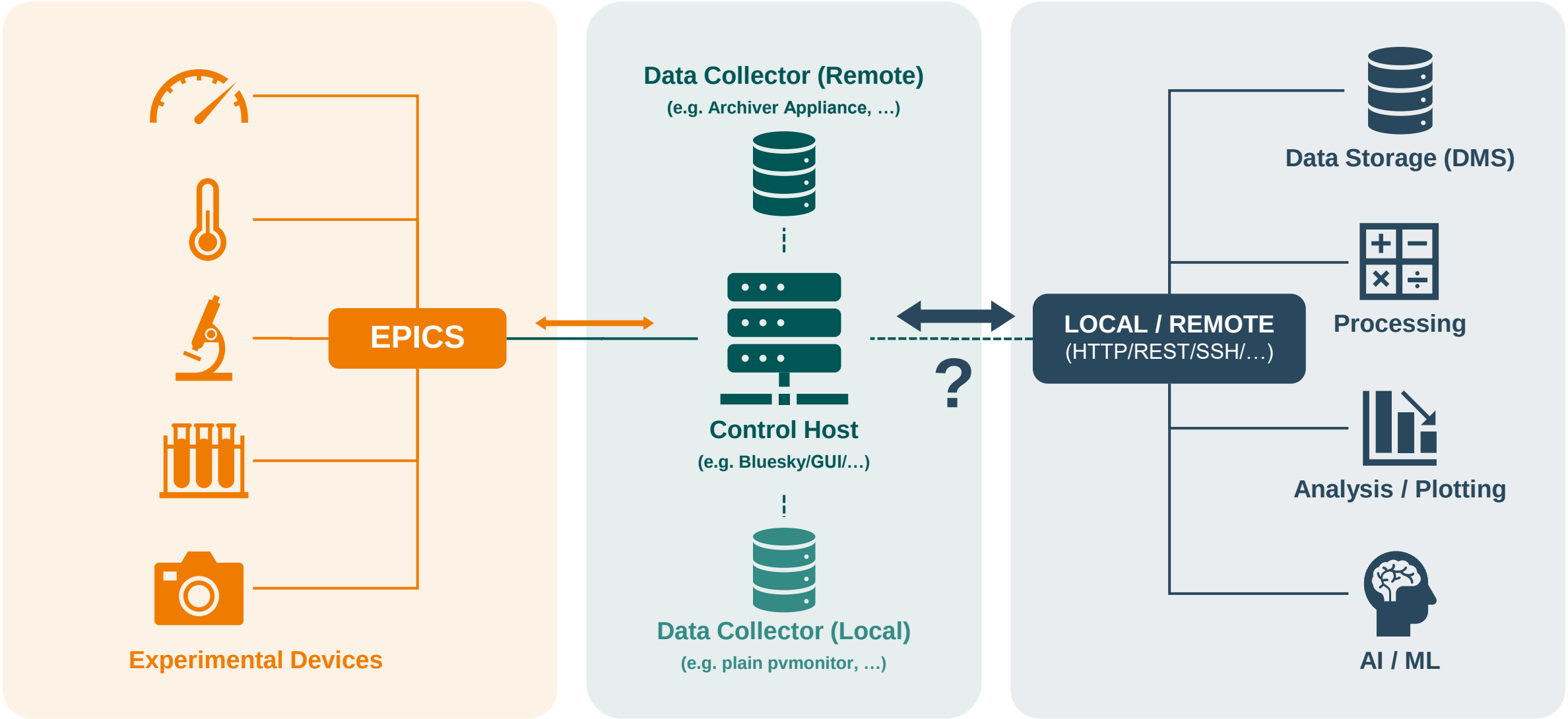
**GOAL**

Users can ‘devicify’ processing steps
and integrate them directly into their setup.

**How?**

- Create a ‘Processing Device’ via Python + P4P
- Run it on local machine or on remote sever

# HOW TO ORCHESTRATE DEVICES & COMPLEX PROCESSING?




**Orchestrate Devices via EPICS**

**Orchestrate Local/Remote Processing?**

# HOW TO ORCHESTRATE DEVICES & COMPLEX PROCESSING?

***How to …***

*… distribute processing?*

*… communicate with/between processing hosts?*

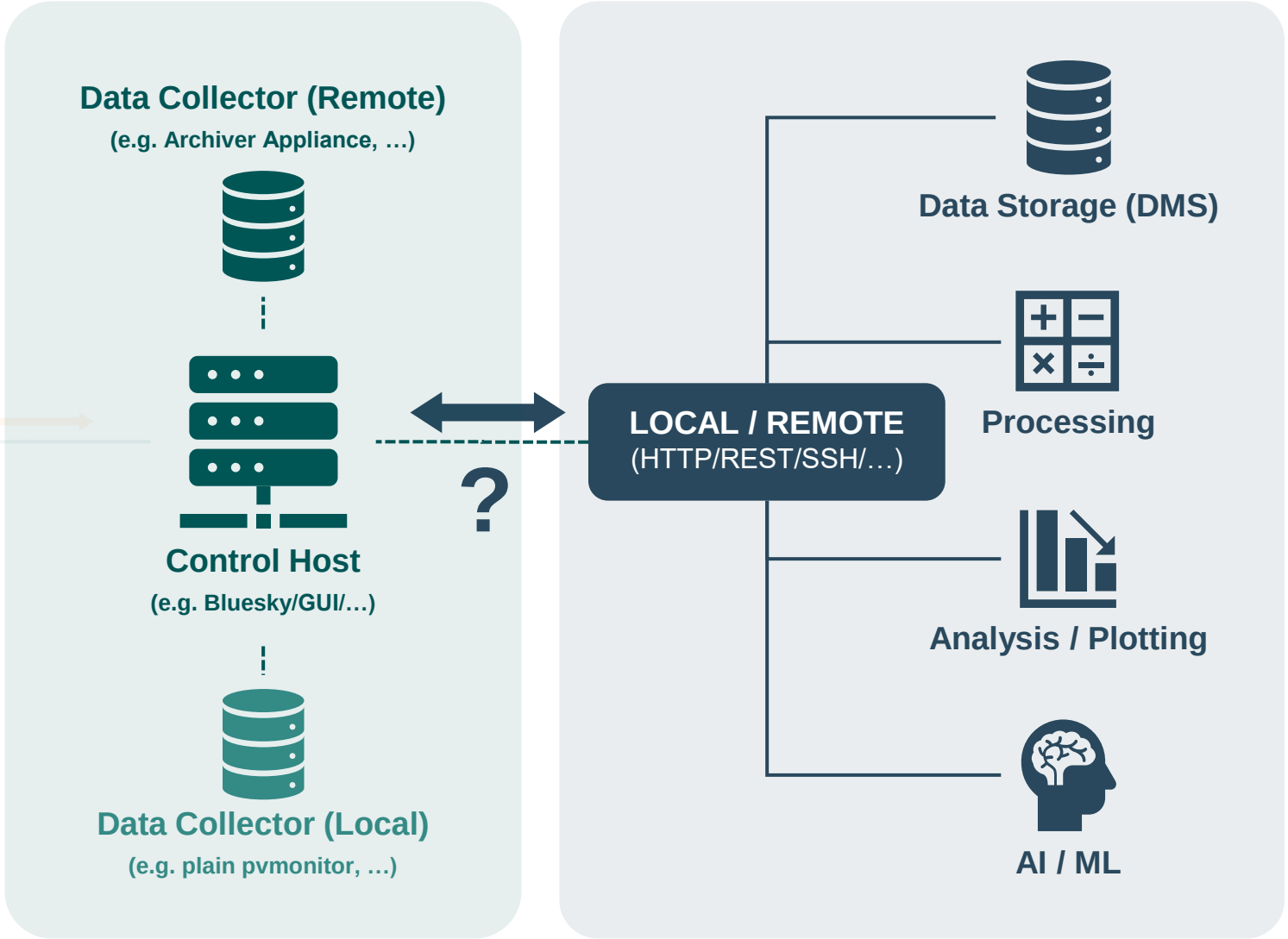
*… transfer data?*

*… handle errors/alarms?*

*… extend processing?*

*… feed-back data for closed loops?*

*… get live-updates?*




**Orchestrate Local/Remote Processing?**

# USE EPICS / PVACCESS?

***How to …***

*… distribute processing?* ✓

*… communicate with/between processing hosts?* ✓

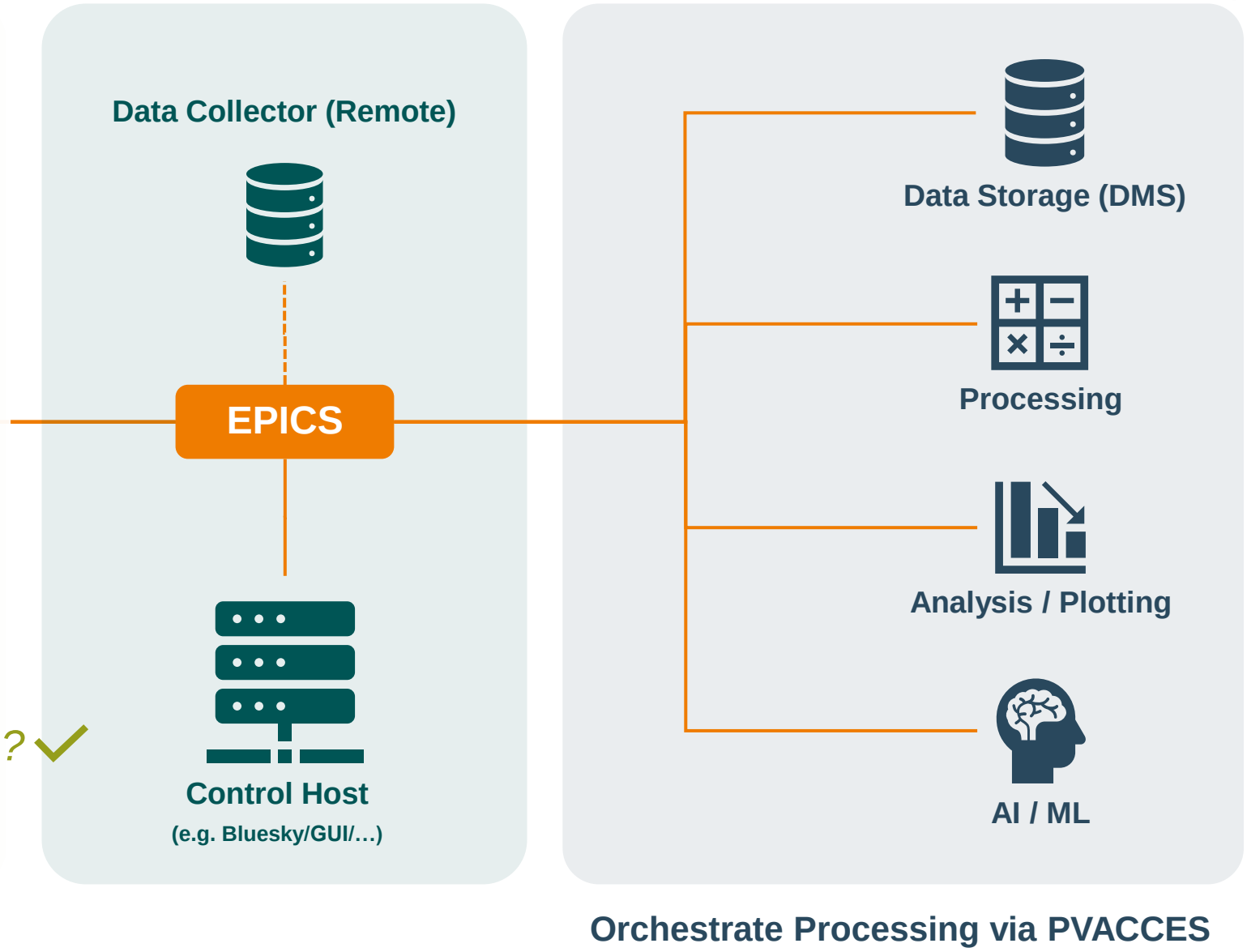
*… transfer data?* ✓

*… handle errors/alarms?* ✓

*… extend processing?* ✓

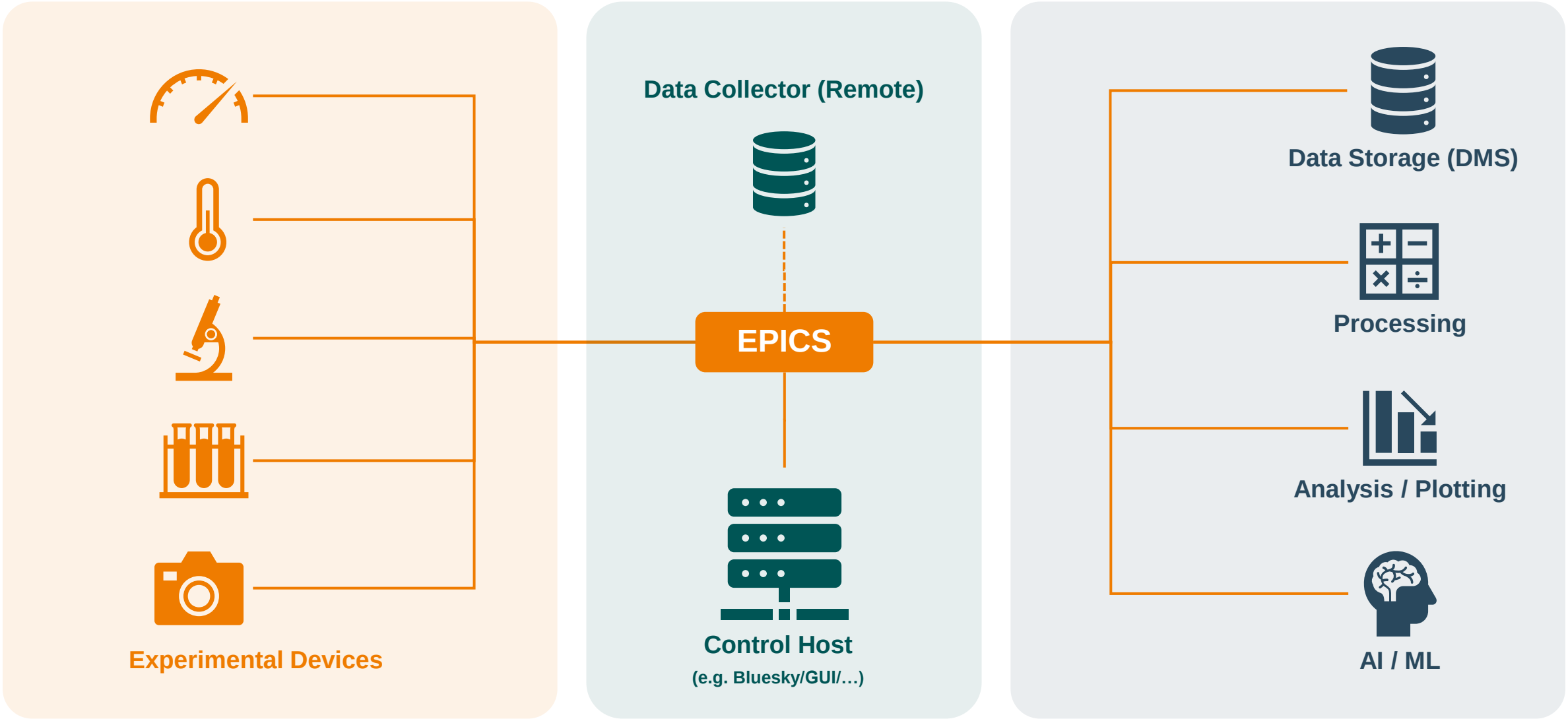
*… feed-back data for closed loops?* ✓

*… get live-updates?* ✓




**Orchestrate Processing via PVACCES**

# USE EPICS / PVACCESS?

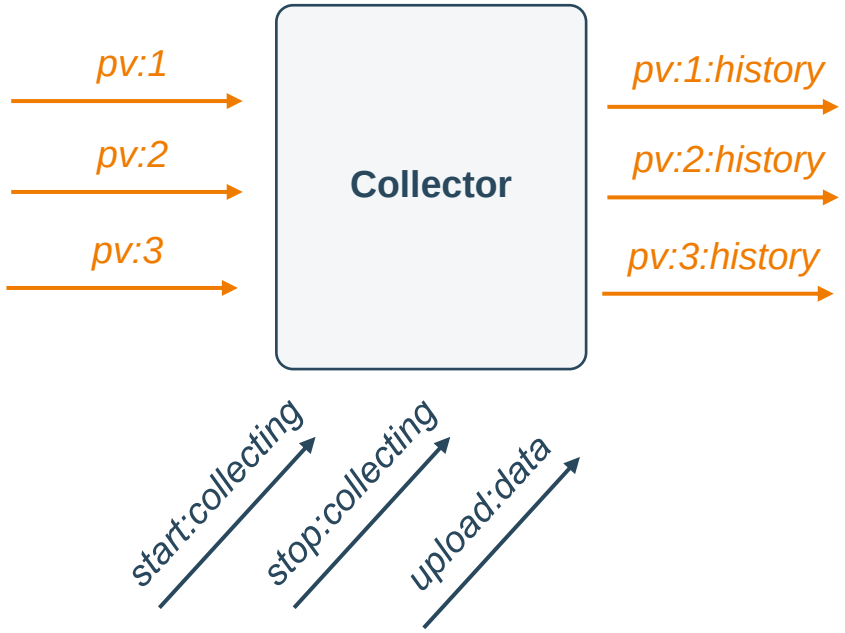



**Orchestrate Devices via EPICS**

**Orchestrate Processing via PVACCES**

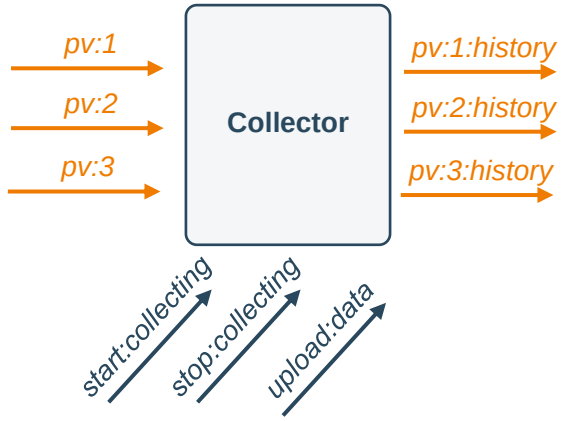
# EPICS COLLECTOR “DEVICE”

- Records one or multiple PVs and serves them as array PVs
- Python + PVAccess (P4P client + server)
- Optional easy to use Python client
  (Todo: Ophyd-interface)
- Additional functionalities
  (start, stop, upload, to-hdf5, …)



# EPICS COLLECTOR “DEVICE”

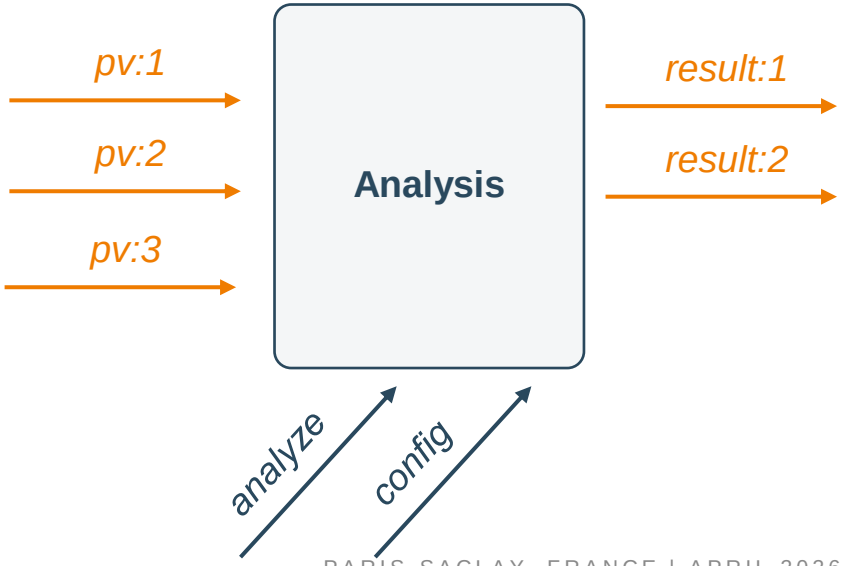
- Records one or multiple PVs and serves them as array PVs
- Python + PVAccess (P4P client + server)
- Optional easy to use Python client
  (Todo: Ophyd-interface)
- Additional functionalities
  (start, stop, upload, to-hdf5, …)




# EPICS “PROCESSING DEVICE”

For Processing / Analysis / Visualization / ML …

- Get data from PVs
- Do something
- Serve results as PVs



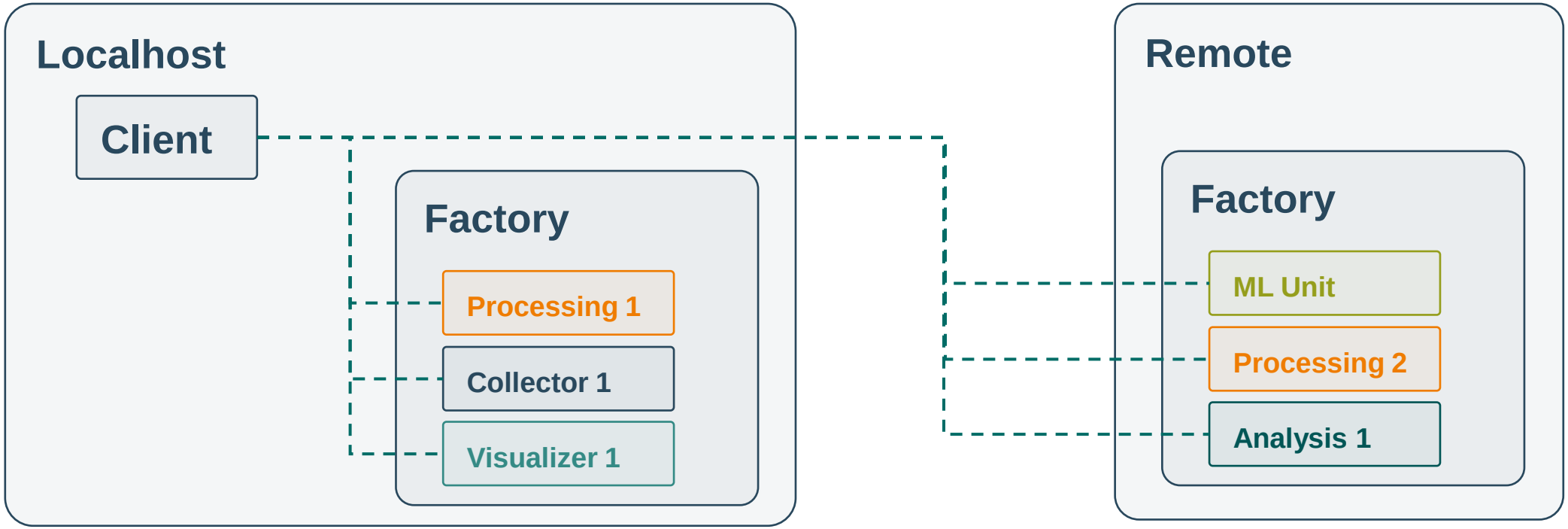
# FACTORY DESIGN

*How to run multiple devices on local & remote hosts?*

**Users can:**

- Create local & remote devices on-demand
- Switch between local & remote devices

→ Modular, scalable, & **distributable**



# WORK IN PROGRESS

*Gitlab Repo: TBA*

***Thanks!***

**GOAL:** “Base”-Toolset for Devices/Factories/Clients

- Similar approaches already in use?
- Experiences with ‘devicifying’ collector/analysis/processing steps?
- Other approaches in use to orchestrate post-acquisition steps in unified way?

→ *beinlich@fhi.mpg.de*