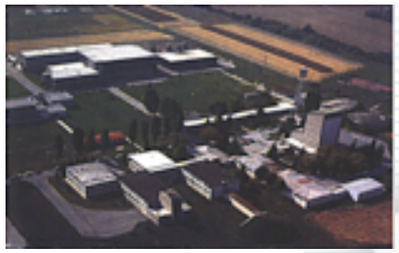
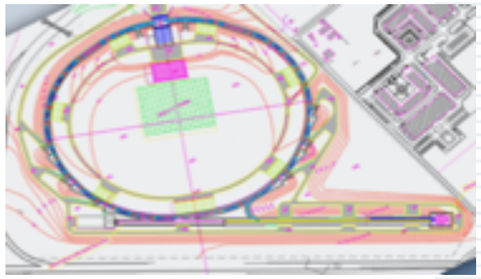
C ontrol system based on a
H ighly
A bstracted and
O pen
S tructure

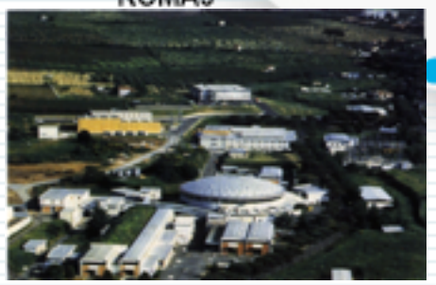
26 June 2013, ESRF Grenoble

http://chaos.infn.it

INFN infrastructure & divisions







20 divisions at university sites

4 National Laboratories
CNAF
EGO
Cabibbo-LAB

The LNF accelerators history

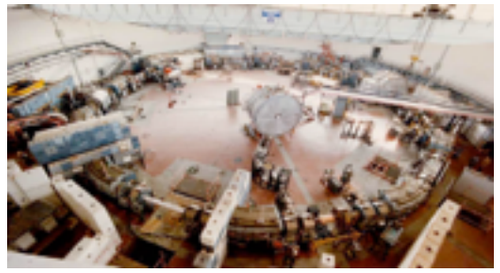
Electron Synchrotron (1959-1975) E=1 GeV

AdA 1960-1965 250 MeV

ADONE (1968- 1993) 3 GeV 100 m

DAFNE (1999) 510 MeV 100 m

SPARC_LAB (2004) 150 MeV LINAC



LNF-54/48 (1954)

Il progetto italiano di un elettrosincrotone.

G. Salvini

Istituto di Fisica dell'Università - Pisa
Istituto Nazionale di Fisica Nucleare - Sezione Acceleratore

The Frascati Storage Ring.

C. Bernardini, G. F. Corazza, G. Ghigo
Laboratori Nazionali del CNEN - Frascati

B. Touschek
Istituto di Fisica dell'Università - Roma
Istituto Nazionale di Fisica Nucleare - Sezione di Roma

(ricevuto il 7 Novembre 1960)

VOLUME 124, NUMBER 5

Electron-Positron Colliding Beam Experiments

N. Cabibbo and R. Gatto
Istituti di Fisica delle Università di Roma e di Cagliari, Italy and Laboratori Nazionali di Frascati del C.N.E.N., Frascati, Roma, Italy

(Received June 8, 1961)

N. Cabibbo

the "bible"

AdA was the first matter antimatter storage ring with a single magnet (weak focusing) in which e+/e- were stored at *250 MeV*

1961	AdA	Frascati	Italy
1964	VEPP2	Novosibirsk	URSS
1965	ACO	Orsay	France
1969	ADONE	Frascati	Italy
1971	CEA	Cambridge	USA
1972	SPEAR	Stanford	USA
1974	DORIS	Hamburg	Germany
1975	VEPP-2M	Novosibirsk	URSS
1977	VEPP-3	Novosibirsk	URSS
1978	VEPP-4	Novosibirsk	URSS
1978	PETRA	Hamburg	Germany
1979	CESR	Cornell	USA
1980	PEP	Stanford	USA
1981	Sp-pbarS	CERN	Switzerland
1982	p-pbar	Fermilab	USA
1987	TEVATRON	Fermilab	USA
1989	SLC	Stanford	USA
1989	BEPC	Beijing	China
1989	LEP ,	CERN,	Switzerland
1992	HERA	Hamburg	Germany
1994	VEPP-4M	Novosibirsk	Russia
1999	DAΦNE	Frascati	Italy
1999	KEKB	Tsukuba	Japan
2000	RHIC	Brookhaven	USA
2003	VEPP-2000	Novosibirsk	Russia
2008	BEPCII	Beijing	China
2009	LHC	CERN	Switzerland

collider in the world

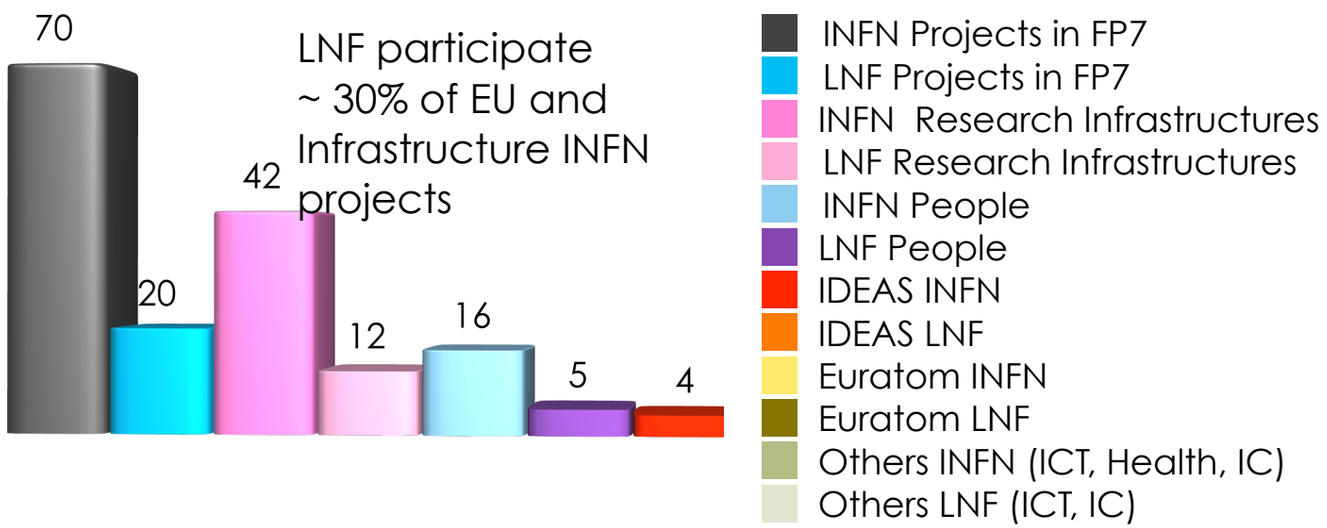
The Frascati INFN National Laboratory

Total Staff	Researchers	Technologist Engineers	Technicians	Administration Services
293	80	38	142	33

External Users	Italian	Foreign
501	257	244

Visitors	Stages	Conference Workshops	Participants to Seminars	Course for teachers of high school
3426	184	21	765	172

INFN Projects in FP7 (up to 1° Feb 2012)



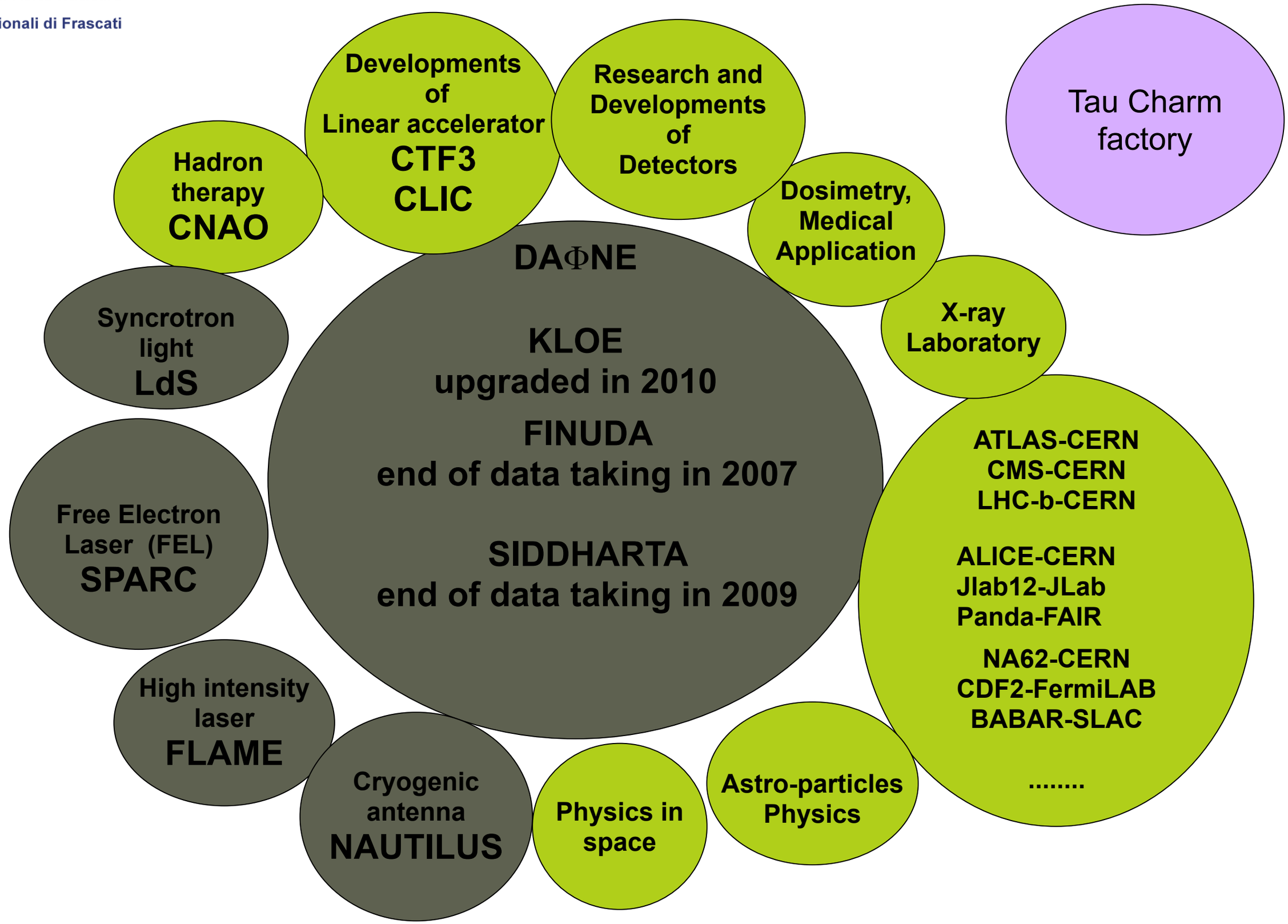
Type of Projects

Hadron Physics 2 In FP7 (end December 2011)
Transnational Access Activities 1880 assigned days in 2009 1673 assigned days in 2010 2853 assigned days in 2011

Istituto Nazionale di Fisica Nucleare
Laboratori Nazionali di Frascati

Accelerators infrastructure at LNF today

The LNF research areas



Motivation

- The SuperB/Tau Charm opportunity
- The DANTE experience
- New concepts - mainly web based - in software architecture
- Software developer and accelerator physicist collaboration
- DAFNE and SPARC test facilities upgrade, benchmark and opportunity

Objective

design a new controls architecture in order to achieve:

- intrinsic **scalability** ensuring high **throughput** data acquisition and high data **computation** performance;
- intrinsic **redundancy** ensuring controls system fault tolerance;
- minimize **configuration and management** efforts;
- support “on the fly” **connectivity** functionality;
- ensure hardware and software (drivers and standards) implementation **versatility**;
- **uniformity** and **standardization** of data and data handling

Objective (cont)

- Complete **abstraction of data** in order to ensure **portability** and reduce constraint respect to hardware and software choice;
- Complete **abstraction of the instruments** and **class** of instruments in order to provide **flexible and portable** implementation of **control algorithm**;
- Embed data historization and storing for large data **throughput**;
- Ensure easy and **open GUI/HLS** implementation with all standards

Objective

open source project

for an

open software platform

able to provide

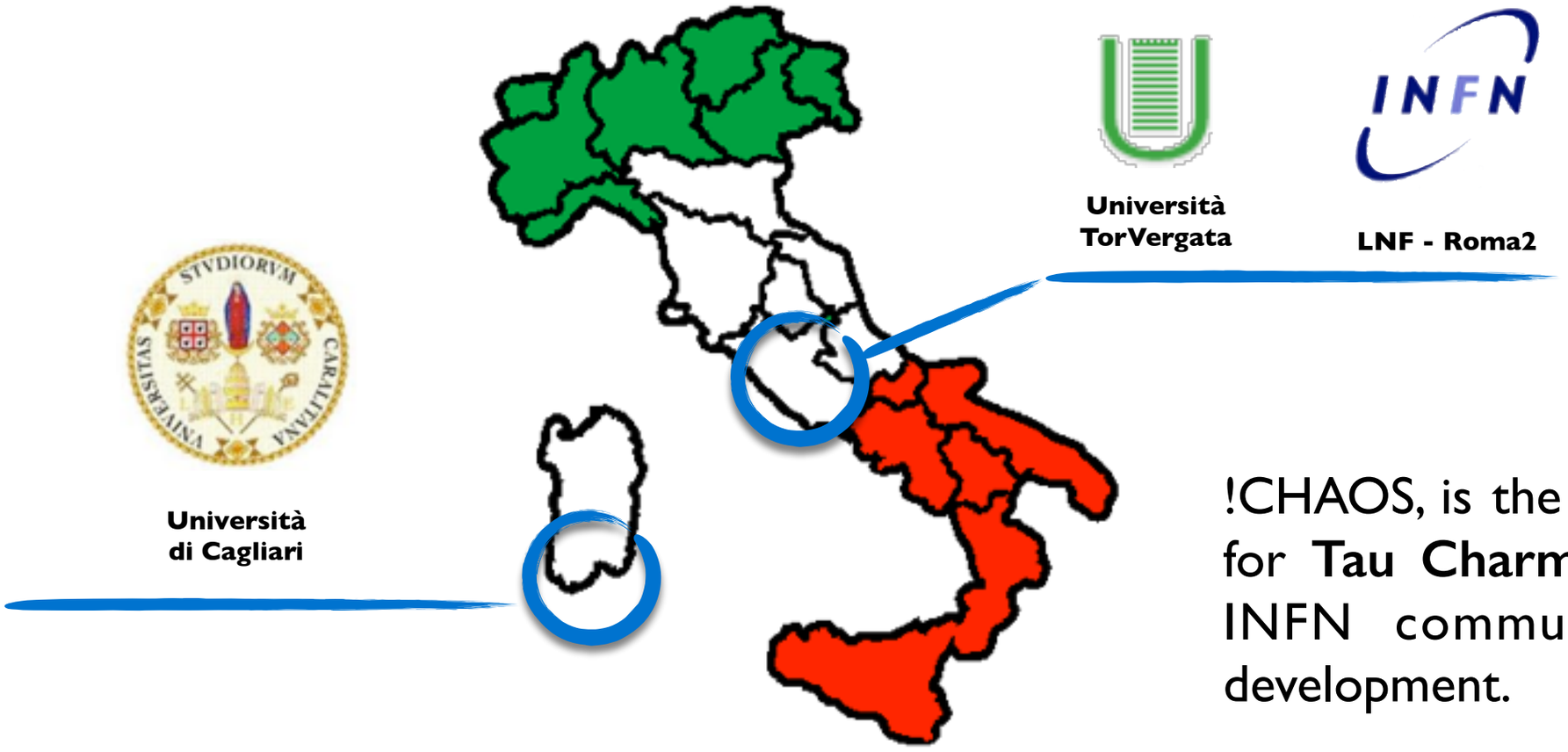
controls as a services

Target

What !CHAOS is today

!CHAOS (**C**ontrol system based on **H**ighly **A**bstracted and **O**pen **S**tructure), is an experiment of CSN 5(technological research experiments) at the **LNF** and **Roma-TV** section.

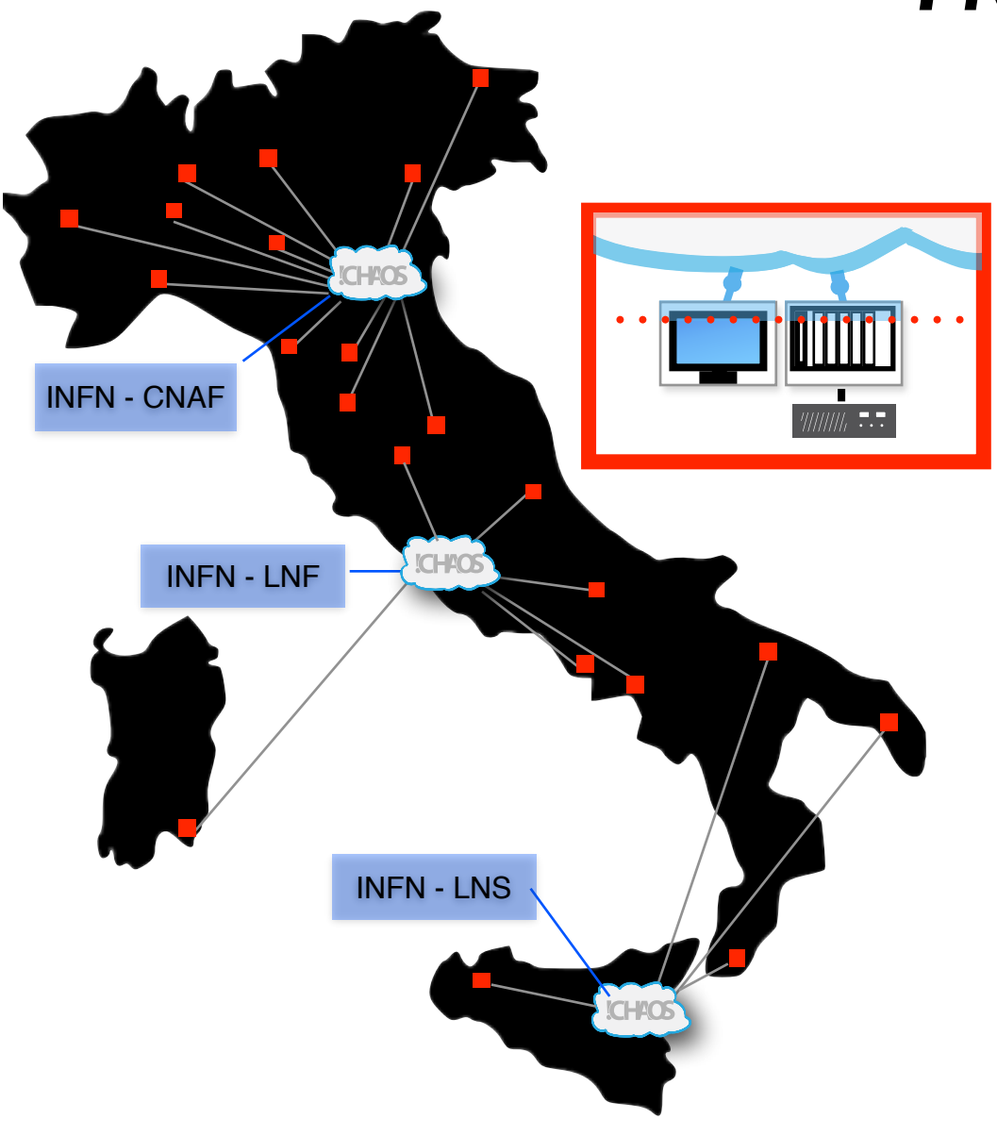
The project has been appreciated by educational world and the University of Tor Vergata and Cagliari have joined the development team.



!CHAOS, is the proposed control system for **Tau Charm**, it has collected a large INFN community interested in his development.

but !CHAOS is more...

Premiale INFN "!CHAOS: A Cloud of Controls"



INFN-LNF (Laboratori Nazionali di Frascati)
INFN-TV (Sezione di Tor Vergata)
INFN-PG (Sezione di Perugia)
INFN-CNAF (Centro Nazionale Tecnolgie Informatiche)
INFN-PD (Padova)
INFN-LNS (Laboratori Nazionali di Catania)

National Instruments (NI)

ADF Solaris

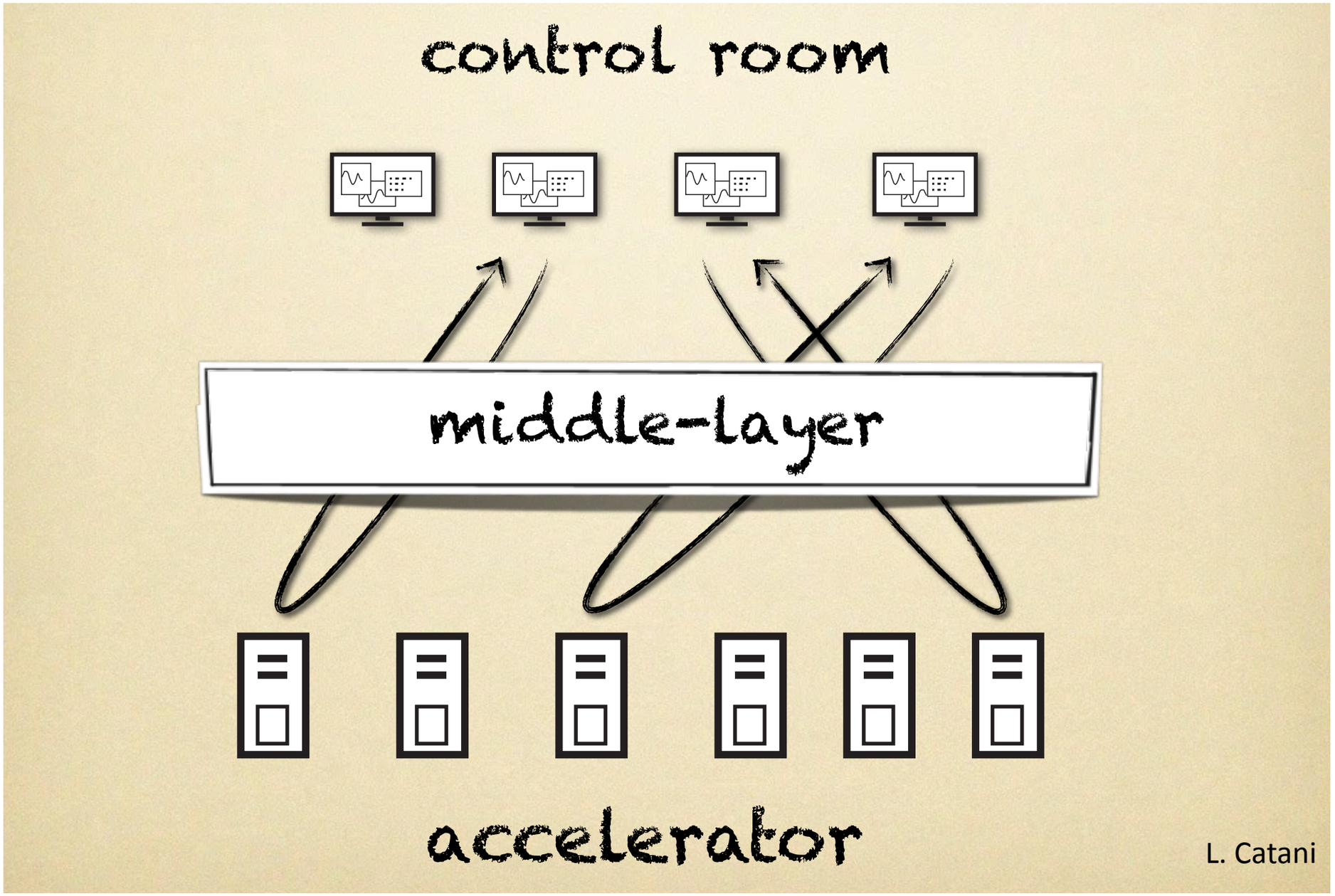
a **prototype of Control as a Service**, an infrastructure at national level which realizes a cloud of services and procedures distributed and shared over the LAN/WAN, which allows the **monitoring and control of any hardware device, system, or intelligent component** and carries a network of resources to provide processing services, data logging and archiving.

Control System Standard Model

The “standard model” consists of a local area network providing communication between front end microcomputers, connected to the accelerator, and workstations, providing the operator interface and computational support

(ICALEPCS) B. Kuiper (1991)

The standard Model

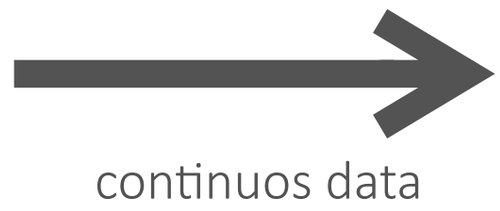


The devices are controlled conceptually by an extension of the device data bus. Services provided by middle-Layer help for the configuration and setting of system

Accelerator data

- data can be divided essentially in three different type:

- **slow data** (a few bytes @ Hz)
  - eg: magnets, vacuum, temperature, etc
- **fast data** (Kbytes of bytes @ kHz)
  - eg: BPM, beam lost monitor, luminosity monitor, synchronized bump, etc

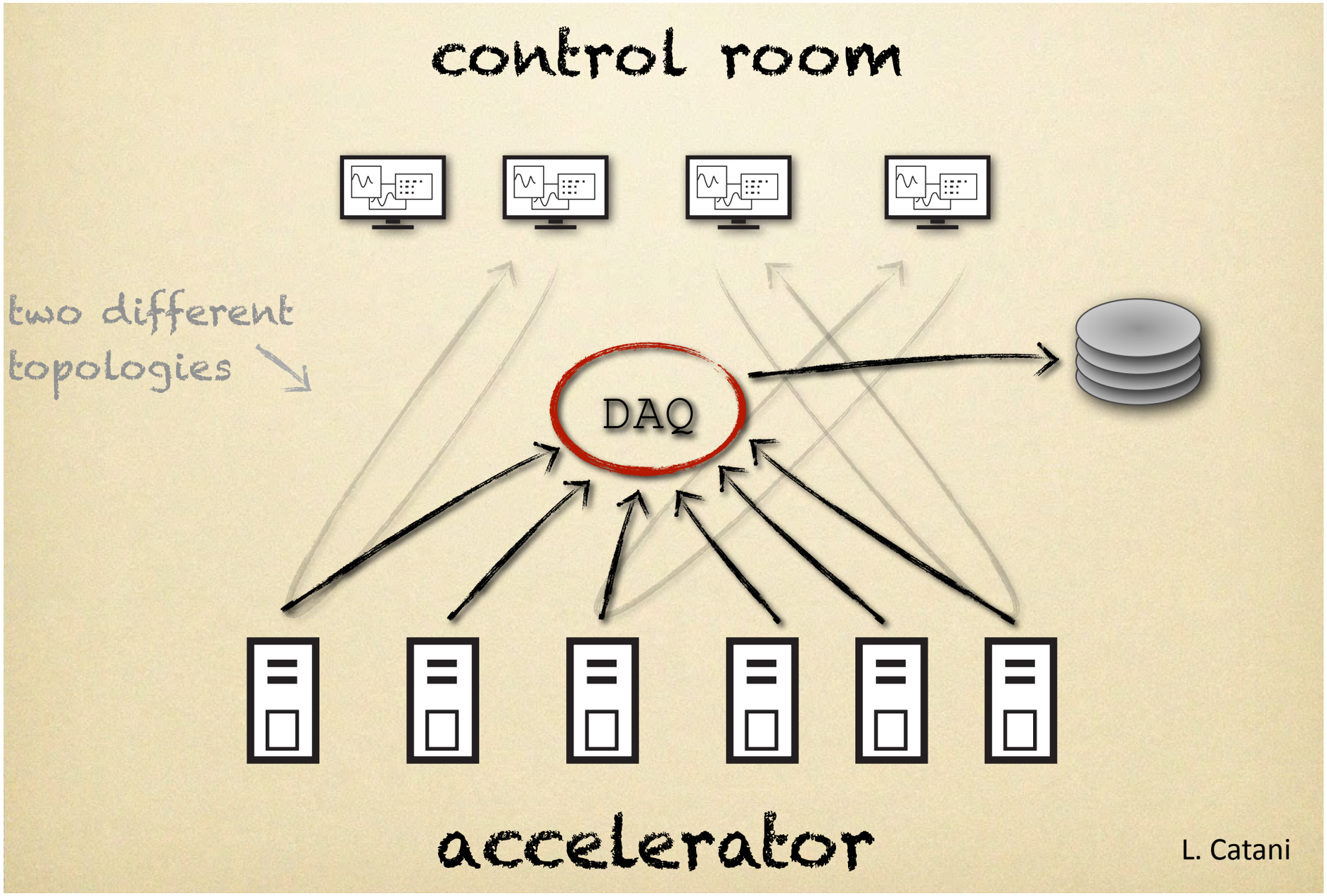


continuos data

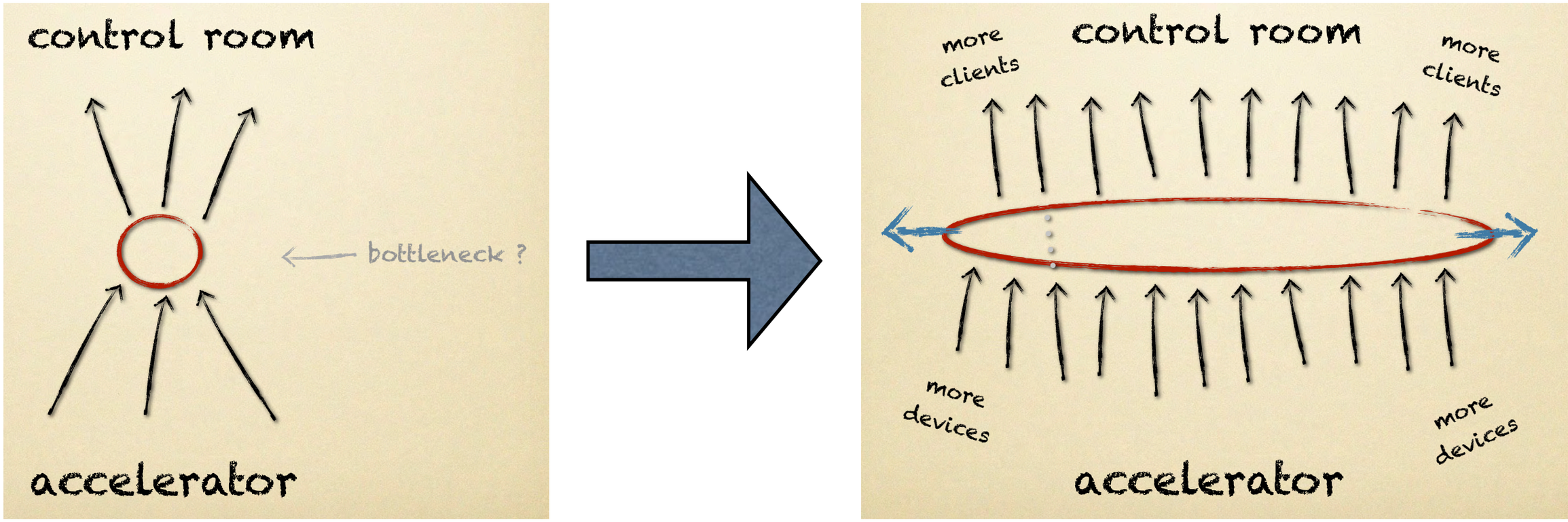
- **very fast and/or large data** (Mbytes @ GHz)
  - eg: BPM single pass, scope, RF, etc

data bursted, limited by hardware and software dead time

DAQ topology



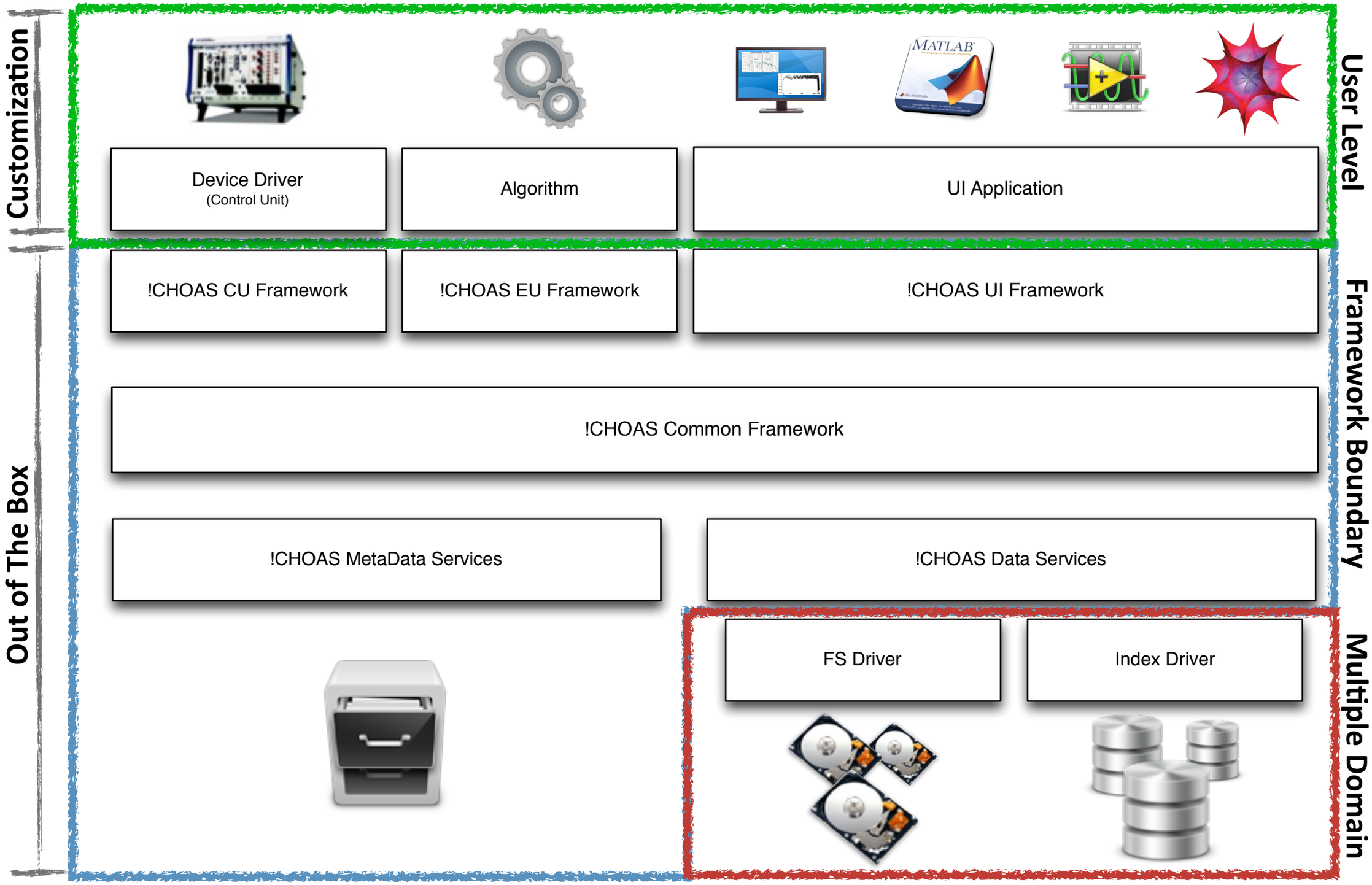
the new topology



the new technologies developed mainly for **web application** on large and fast database (Key Value DB - Distributed Object Cache) ensure today to be able to improve a **scaleable a flexible** set of services to mix the stands of controls.

The !CHAOS Implementation

Conceptual layout



User Customization Boundary

User Interface for monitoring and controls

Abstraction Boundary

User Interface Toolkit

Common Toolkit

RPC - Event - Direct I/O

Execution Unit Toolkit

Common Toolkit

RPC - Event - Direct I/O

Controls Algorithm

Data Boundary

CQL Proxy Nodes

Indexer Nodes

Management Nodes

Live / History domain

Live / History domain

Information Boundary

Node Information

Security

Node Activity

Chaos Directory Nodes

HTTP - API

RPC - Event - Direct I/O

Common Toolkit

Control Unit Toolkit

Driver for sensor and instruments

The hard job with computer scientists!

- **Device** - A specific physical device such as:
  - ➡ apparatus of a given brand and model (e.g.: Stanford - Digital Delay - DDG-535);
  - ➡ custom apparatus (e.g.: RF system with many different outputs of different types (analog, digital, field-busses)
- **Object** - (aliased to Device Class) Virtual representation of a [device | group of devices], with associated
  - ➡ read attributes
  - ➡ write attributes
  - ➡ services
- **Dataset** - data structure (format of the structure), that is: the ensemble of the object **attributes**
- **I/O System Driver** - System library needed to utilize a specific hardware resource (Ethernet, serial port, USB, Bluetooth, i2c, etc...)
- Protocol Driver - Library needed to communicate by mean of a specific protocol, through a specific communication channel
- **Device Driver** - A function which is able to communicate with a device through a specific communication channel by mean of a specific Protocol Driver. A Device Driver is able to issue commands to a device and/or to query the device for getting information back. A driver must be able to perform its actions as a standalone application and must NOT rely on functionalities/data/statuses of the !CHAOS system.

!CHAOS Framework Toolkit

Common Toolkit

is the !CHAOS core software layer to handle RPC, I/O, Event and others utility codes need to implement the system

CU Toolkit

abstracts the !CHAOS resources to the device drivers developers.

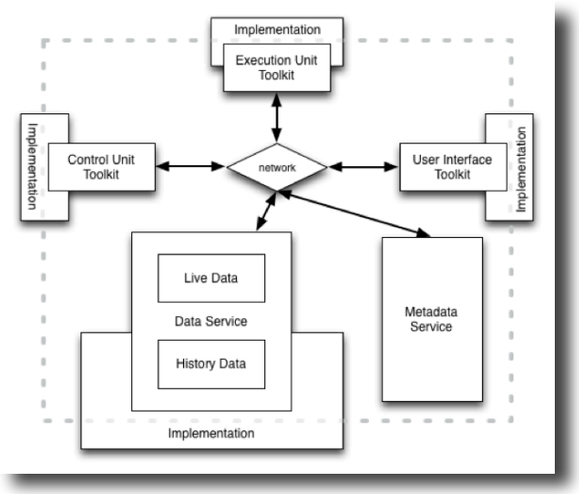
UI Toolkit

abstracts the CHAOS resources to the developers that need to develop control and monitoring user interface

EU Toolkit

abstracts the CHAOS resources to the developers that need to develop control and computing algorithms

!CHAOS Services



Metadata Server

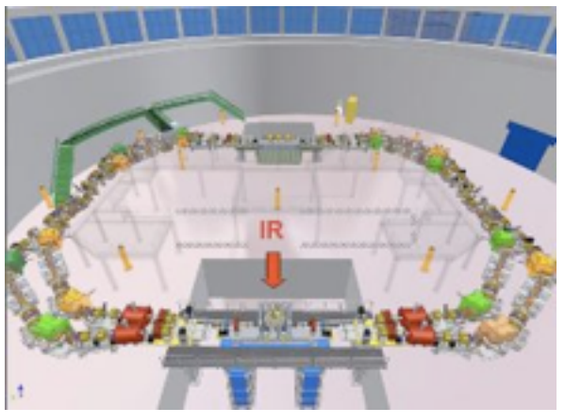
MDS is the information system taking care of devices and nodes setup and configuration. It contains information about all devices controlled, data topology and data domains. Balance the load of the sub-systems, logs information, etc

Data Cloud Domain

DCD provides services for live and historical data management. All instanced services (Proxy, Storage, Indexer) are scalable. Every domain can be implemented on different technologies (FS, IndexDB)

!CHAOS Services

Metadata Servers



Slow Control Data

SC Data Cloud
[live/HST]

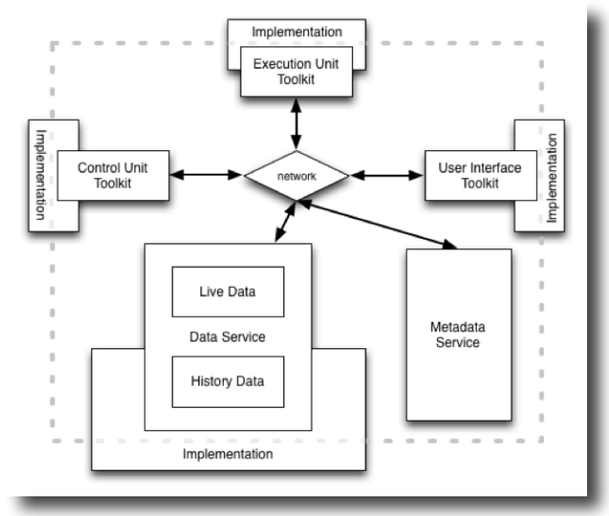
Slow Control Data

Detector Data

Experiment Cloud
[HST]

!CHAOS Nodes Communication System

Communication Systems

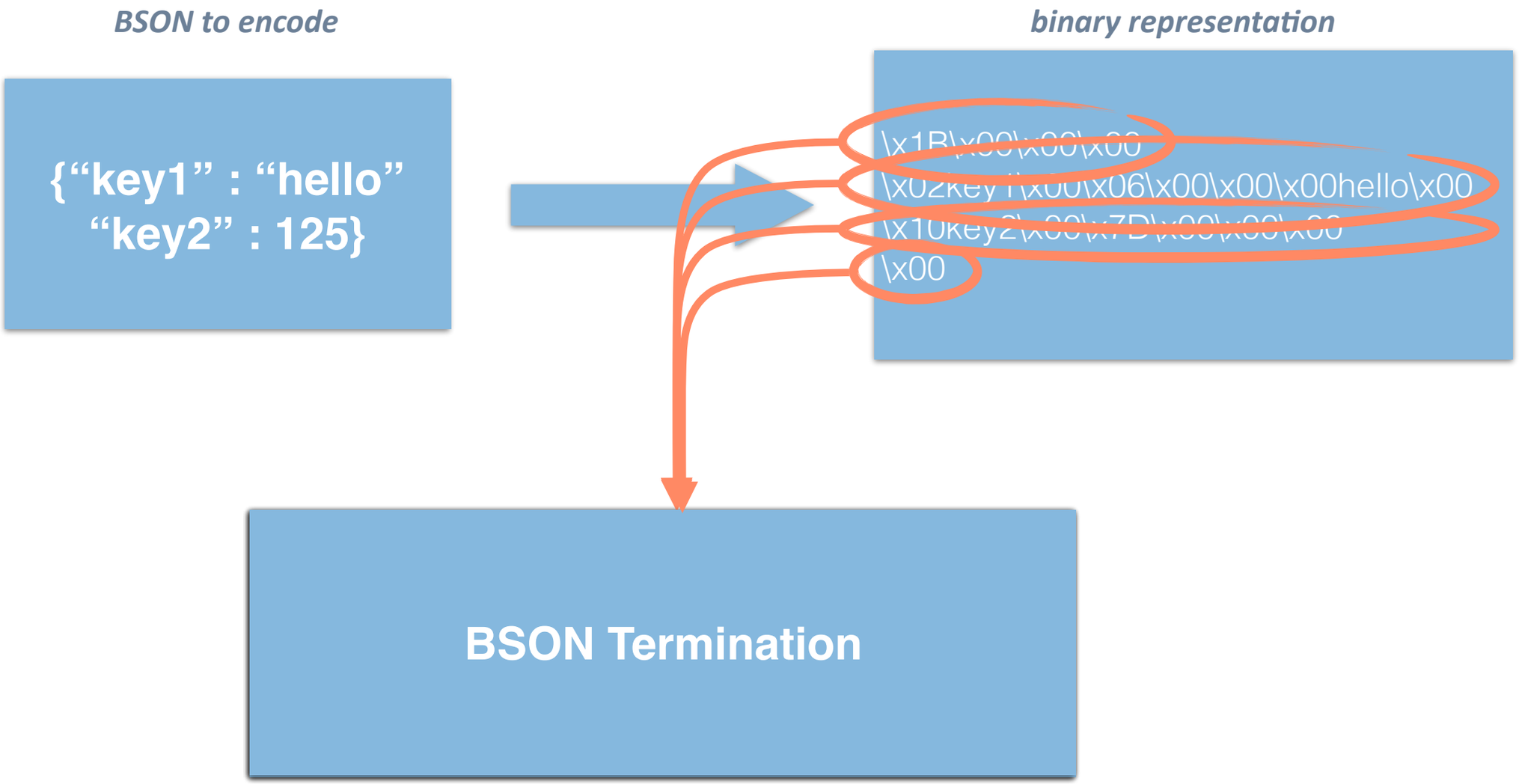


!CHAOS has three different communication system, implemented via plug-ins:

- **Event:** is a lightweight data protocol in **multicast UDP** that is used to bring information about **internal node events** (heartbeat, fault detection, etc.) or to handle other **general purpose data** (locking, discover, load balance information, etc. etc.);
- **RPC:** is used to call **node API**. This method permit to be sure that a called API can be executed by the node and permit to **asynchronously** receive an answer; this methods is **used for commands**.
- **Direct Stream I/O:** permits the **fast transfer of data** (packet o raw data) between two nodes; this methods is used for high throughput data transfer.

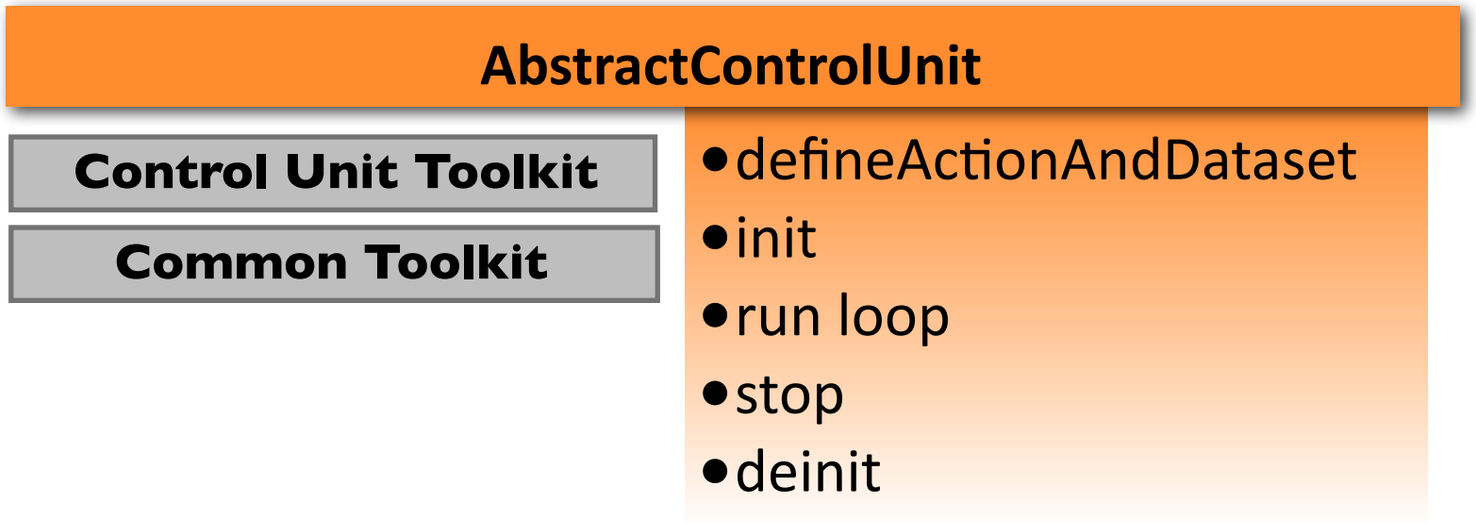
Data Serialization - BSON

data abstraction and **portability** in the system is ensured by it's serialization in BISON binary version of JSON (JavaScript Object Notation)



The !CHAOS Control Unit

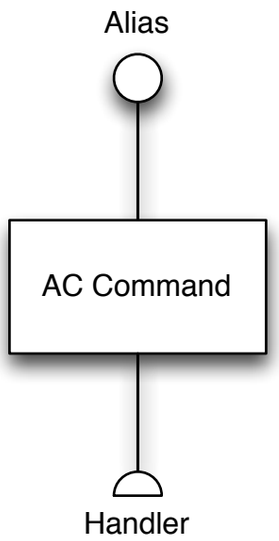
Control Unit

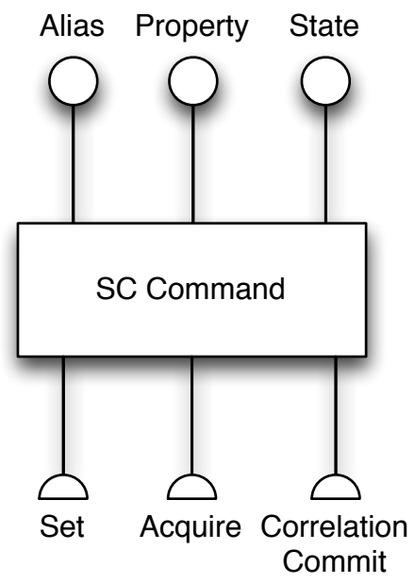


The **Control Unit** (CU) **toolkit abstracts the instrument's drivers**. It consists in a set of API and C++ class that help the developer to realize a **device driver** and the **hardware integration** in the ! CHAOS system.

- handle Command and their implementation as AC (Action) and SC (SlowControl)
- handle the schedule for data upload in Live and historical DB and the timing of Commands
- handle main control run-loop (Acquisition/Control)
- handle device configuration and initialization

CU Toolkit Command





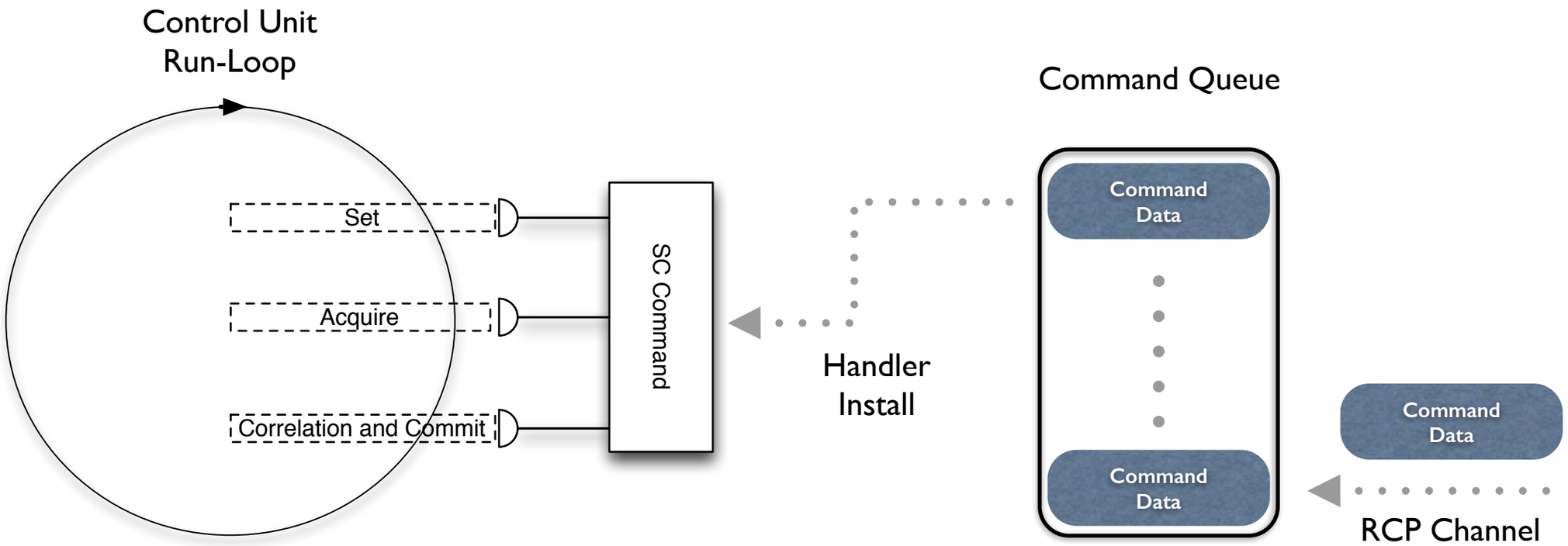
Action Commands handler

- AC Command is a simple RPC call for an action attached to an alias
- it is executed in a **different thread** than the main run-loop, the appropriate synchronizations need to be done by developer

Slow Control commands handlers

- **Set Handler** (mandatory): set the device register for achieve the command purpose
- **Acquire Handler** (optional) need to **acquire** the data needed by next handler
- **Correlation and Commit** (optional), need to **check** the current value and execute hardware control for achieve the command purpose

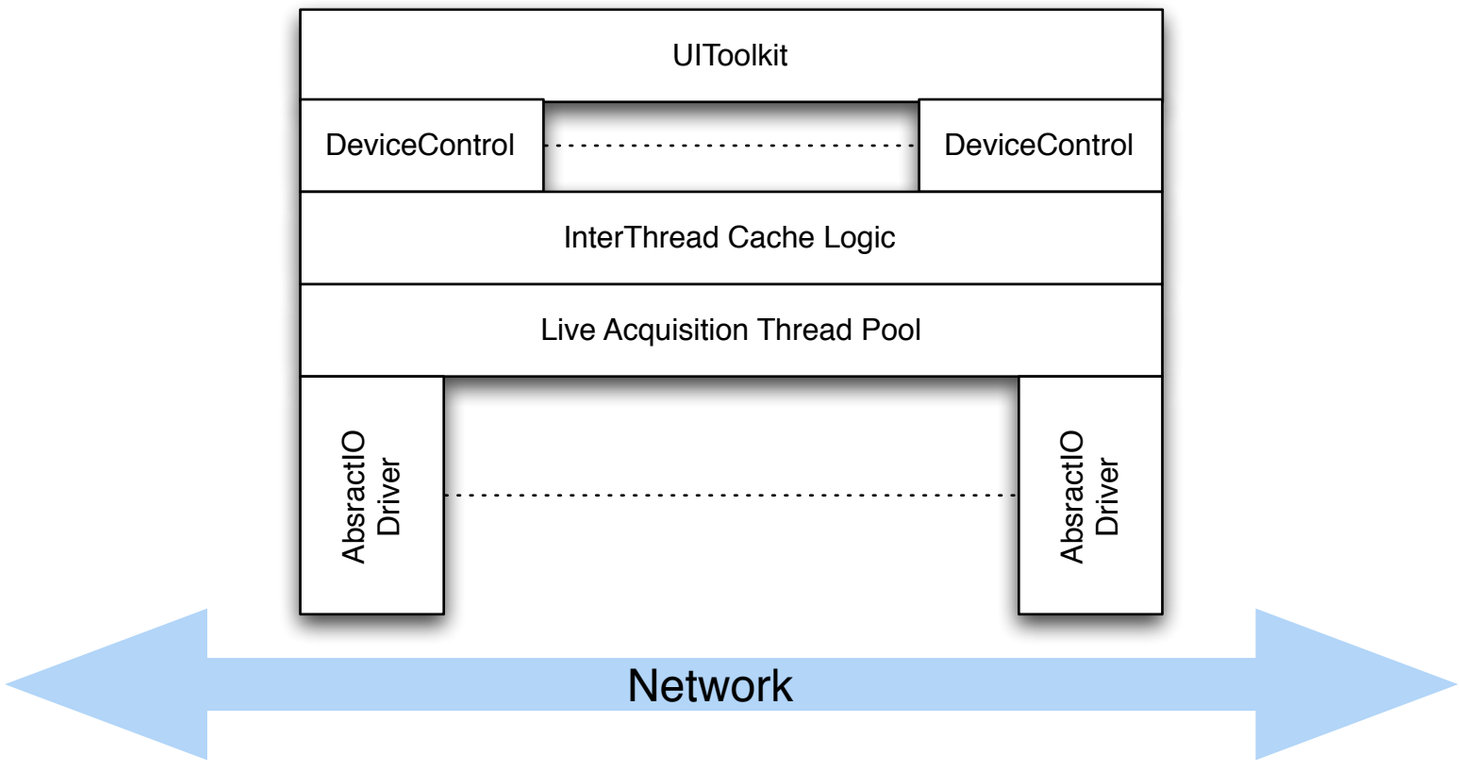
CU Toolkit Commands Queue



!CHAOS User Interface Toolkit

User Interface (UI) Toolkit

The User Interface (UI) toolkit abstracts the user interface and connection with specialized **Graphical User Interface** (GUI). It consist in a set of **API and C++ class** that help the developer to realize the user interface for monitoring and for control the **devices/systems/ subsystems and/or general-purpose algorithms** (EU)



UI Toolkit C language API

An API in C language has been developed to provide access at UIToolkit for language that don’t permit to use C++.

This allow to realize an User Interface or controller algorithm into another software as plugin, ex. LabVIEW.

The C language API provides a minimal set of services implemented to simplify the device controller allocation, deallocation and usage.

!CHAOS Execution Unit Toolkit

Execution Unit (EU) Toolkit

The Execution Unit (EU) toolkit **abstracts the control's algorithms**. It consists in a set of API and C++ class that help the developer to realize a **general-purpose algorithm** (math library, feedbacks, etc) specialized by setting algorithms parameters and input/output data. It can be used in two mode:

- Collect data from **device** and push it on the **data services**;
- Collect data from a **device** to control **another device**.

Execution Unit Example 1

Ouput attribute data is
read from device

some logic is done

CU_I

Device Driver
(Control Unit)

Algorithm
(Custom Execution Unit)

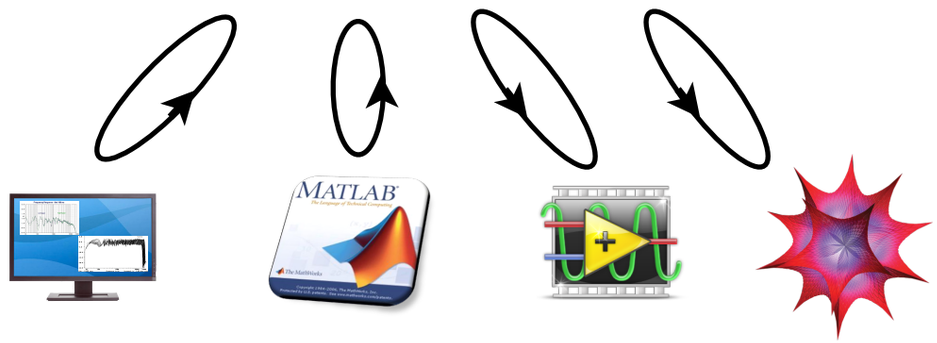
Output attribute data
is pushed
on live memory block
from CU_I

Output CU_I attribute
data is read from live

CU_I DATA

live data/HDS

ELABORATED DATA



Execution Unit Example 2

Ouput attribute data is
read from device

some logic is done

The Attribute is set on HW
by CU_2

CU_1

Device Driver
(Control Unit)

Algorithm
(Custom Execution Unit)

Device Driver
(Control Unit)

Output attribute data
is pushed
on live memory block
from CU_1

Output CU_1 attribute
data is read from live

The Input Attribute on CU_2
is set with RPC command

CU_1 DATA

live data

!CHAOS Data Cloud & Services

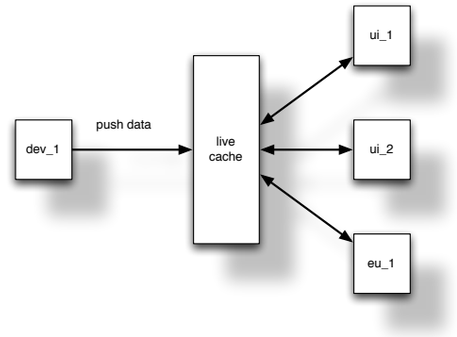
Data Management

!CHAOS has two different data management system

Live Data (LDS)

History Data (HDS)

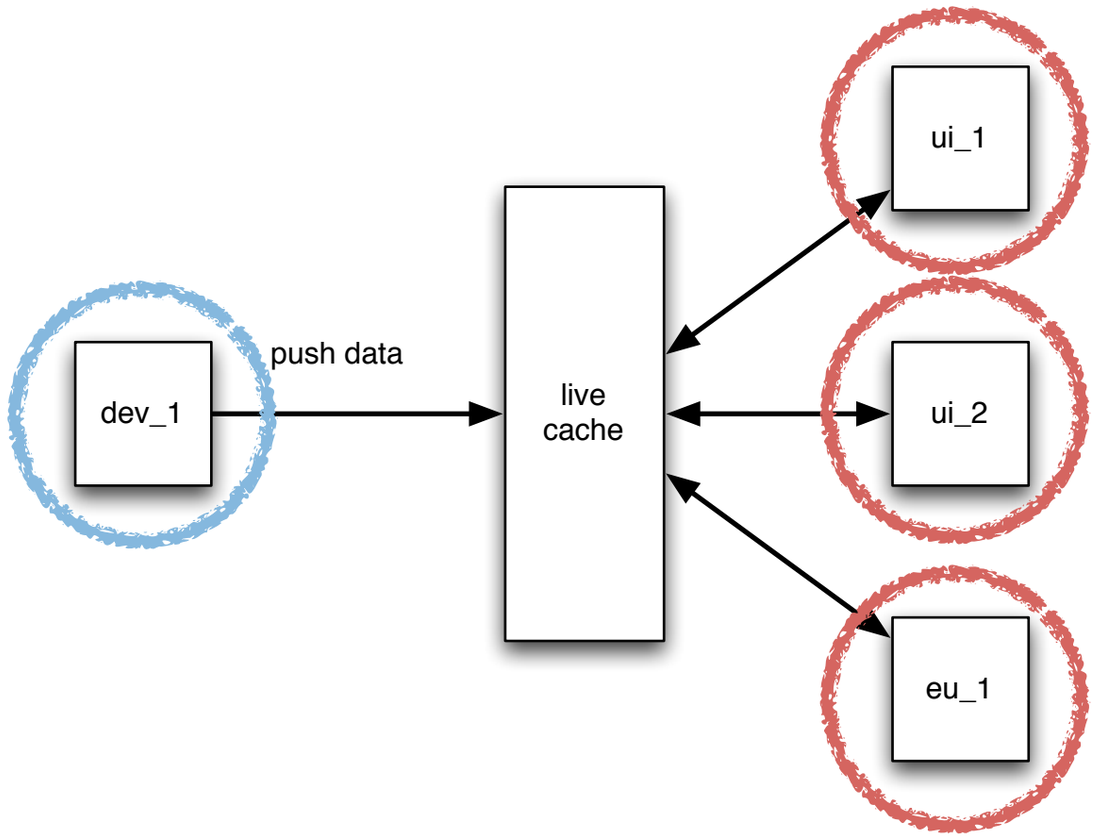
!CHAOS Live Data



Live Data - Distributed Object Cache

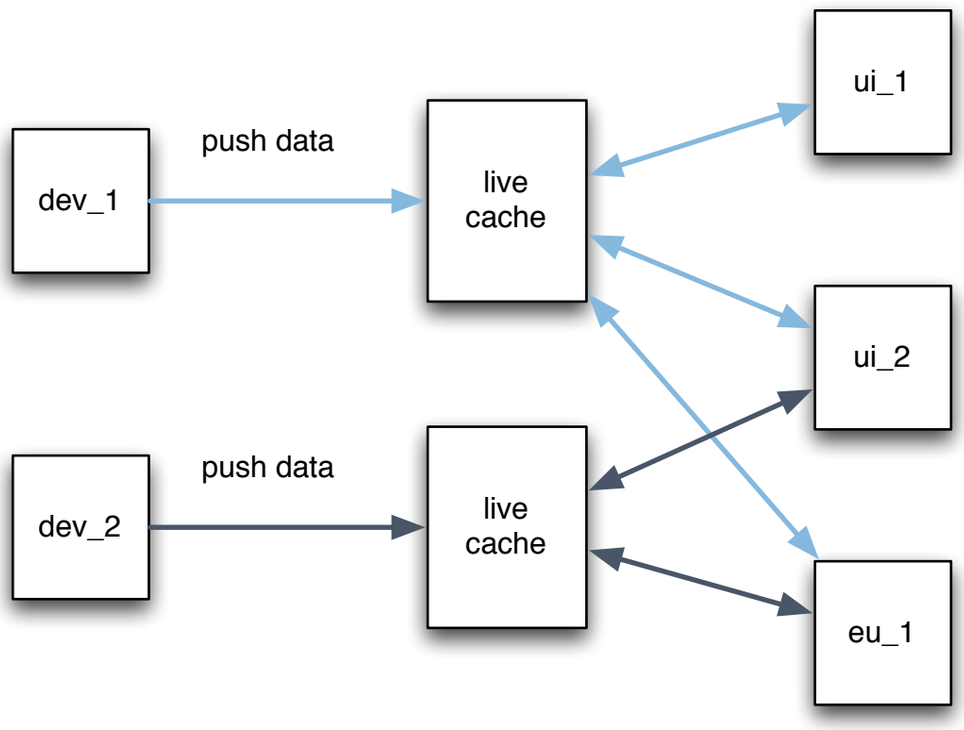
1. every node can read data from the central cache at any time[polling]
2. the nodes can register to central cache for receive update[push]

Every instrument pushes the data of his I/O channels into the shared cached through the controller CU



Live Data

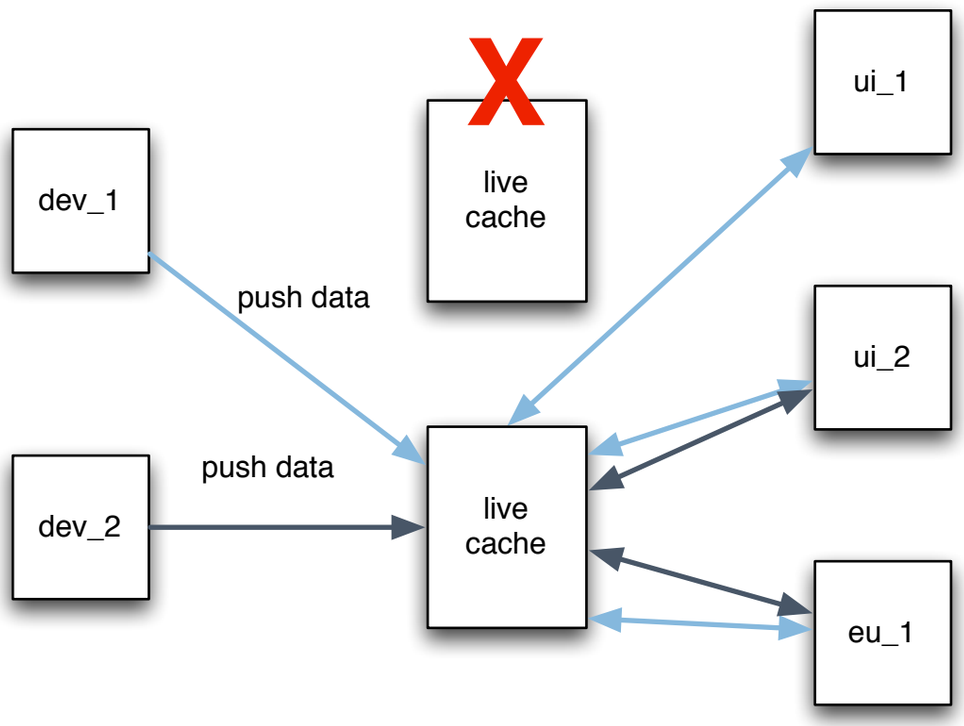
scalability & reliability



All data of all instruments are distributed across many cache server

Live Data

scalability & reliability



with the algorithm previously shown, when one server goes down, all client that push's, or read, data on that server, use another server, with highest priority

Push data service

every node, if authorized, can register itself on the Control Unit for “push” data service.

In !CHAOS this mode is permitted only for those nodes that need to get the device value in a deterministic time.

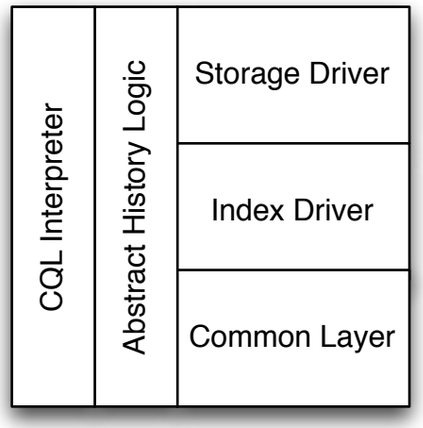
The number of “Push” client is regulated by the control management console.

!CHAOS History Data

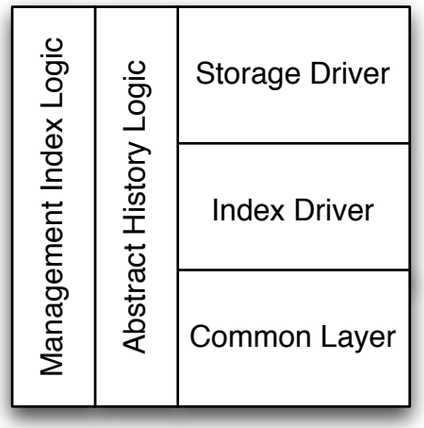
History Data

It is realized by three different **services** and two **driver**

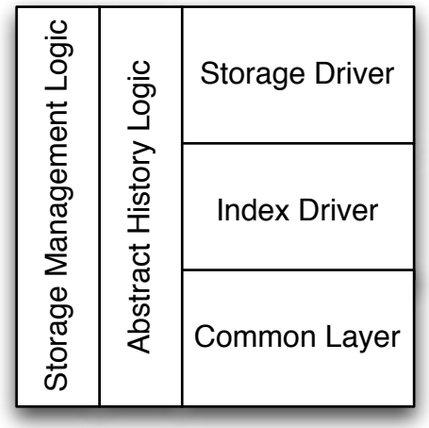
ChaosQL Proxy



Indexer Node



Management Node



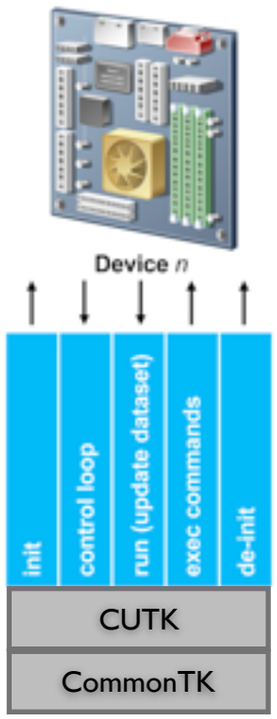
Storage Driver abstraction

Index Driver abstraction

Chaos Query Language

ChaosQL implement the following operation on data

- push history/live data per device
- create index on device::attribute::{rule}
- delete index on device::attribute
- retrieve data with logic operation on attribute and index



Push Data

CQL Proxy

Store Data

Storage

get data pack

Launch Index

Indexer Cluster

Management Cluster

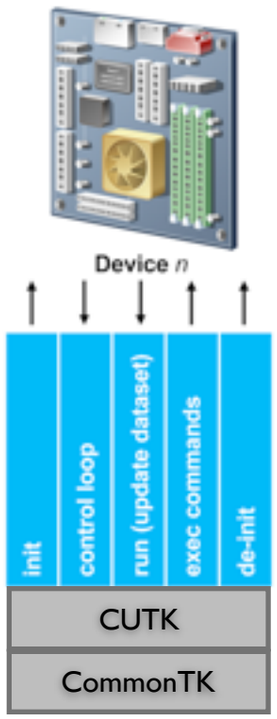
Store Index

CQL Proxy

CQL Proxy

Index Database

Multiwrite features



Push Data

CQL Proxy A

CQL Proxy B

CQL Proxy XXX

CQLProxy A cache file

D1_DP1		D2_DP1		D1_DPN

CQLProxy B cache file

D1_DPN+1		D2_DPN

data are stored into the file cache of each proxy service

Device *n*

init | control loop | run (update dataset) | exec commands | de-init

CUTK

CommonTK

CQL Proxy A

CQL Proxy B

CQL Proxy XXX

Push Data

CQLProxy A cache file

D1_DPI		D2_DPI		D1_DPN

CQLProxy B cache file

D1_DPN+1		D2_DPN

FS Worker

Storage

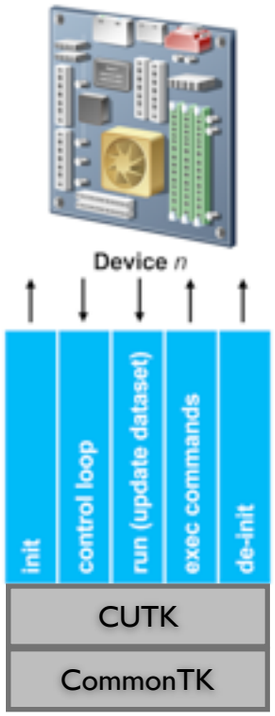
devices logical file

D1_DPI		D1_DPN

D2_DPI		D2_DPN

FS Worker

cached data are reorganized, by the FS worker, into the normal, time ordered, device logical files



Management Cluster

Storage

get pack at location

Indexer Cluster

execute search on idx DB

CQL Proxy

CQL Proxy

CQL Proxy

Query Live Data

send search on idx

Receive History Data

Query History Data

EUTK
CommonTK

UITK
CommonTK

Index Database

Status of the project

- CU toolkit prototype ready
- UI toolkit prototype ready
- MDS java simple prototype ready, but we want to rewrite in C++
- Live data prototype ready
- EU toolkit under development
- History data under development

...conclusion

- ✓ !CHAOS project is trying to realize a new approach to controls based on novel concepts in computing science and technology
- ✓ !CHAOS project is open source and open software R&D program submit to the Controls System community to study and overcome limitation of existing controls architecture (if any)
- ✓ !CHAOS is also an opportunity to study new implementation of subsystems (es high performing HDS) needed for many applications in controls and more...

Thanks

http://chaos.infn.it/