ISTITUTO NAZIONALE DI FISICA NUCLEARE

Laboratori Nazionali di Frascati

INFN-16-06/LNF
6th April 2016

!CHAOS: FINAL PROJECT REPORT

S. Angius, G. Baldini, C. Bisegni, S. Caschera, P. Ciuffetti, P. Conti, G. Di Pirro, F. Galletti, R. Gargana, O. Giacinti, E. Gioscio, D. Maselli, G. Mazzitelli, A. Michelotti, R. Orrù, M. Pistoni, F. Spagnoli, D. Spigone, A. Stecchi, T. Tonto, M. A. Tota[1]
L. Catani, C. Di Giulio, G. Salina[2]
P. Buzzi, B. Checcucci, P. Lubrano[3]
E. Fattibene, M. Panella[4]
M. Michelotto[5]
S. Aurnia, S. R. Cavallaro, B. F. Diana, E. Furia, S. Pulvirenti[6]

[1)]INFN – Laboratori Nazionali di Frascati
[2)]INFN – Sezione Roma Tor Vergata
[3)]INFN – Sezione di Perugia
[4)]INFN – CNAF
[5)]INFN – Sezione di Padova
[6)]INFN – Laboratori Nazionali del Sud

Pubblicato da SIDS–Pubblicazioni
Laboratori Nazionali di Frascati

!CHAOS Final Project Report

S. Angius, G. Baldini, C. Bisegni, S. Caschera, P. Ciuffetti, P. Conti, G. Di Pirro, F. Galletti, R. Gargana, E. Gioscio, D. Maselli, O. Giacinti, G. Mazzitelli, A. Michelotti, R. Orrù, M. Pistoni, F. Spagnoli, D. Spigone, A. Stecchi, T. Tonto, M. A Tota
INFN-LNF (Laboratori Nazionali di Frascati)
L. Catani, C. Di Giulio, G. Salina - *INFN-TV (Sezione di Tor Vergata)*
P. Buzzi, B. Checcucci, P. Lubrano - *INFN-PG (Sezione di Perugia)*
E. Fattibene, M. Panella - *INFN-CNAF (Centro Nazionale Tecnologie Informatiche)*
M. Michelotto - *INFN-PD (Padova)*
S. Aurnia, S. R. Cavallaro, B. F. Diana, E. Furia, S. Pulvirenti
INFN-LNS (Laboratori Nazionali di Catania)

Index

Executive Summary

The !CHAOS project has been devoted to the realization of a prototype of Control as a Service open platform suited for a large number of applications in science, industry and society. The Control Server concept has been introduced to give emphasis to the innovative !CHAOS architecture that is represented by a scalable and distributed cloud-like infrastructure providing the services needed for implementing distributed control and data acquisition systems.
The project is based on the results of an R&D initiative promoted by INFN-LNF and INFN-Roma "Tor Vergata", aimed to the development of a new architecture for controls of large experimental infrastructures named !CHAOS (Control system based on Highly Abstracted and Open Structure). To fully profit from this new technologies the control system model has been reconsidered, thus leading to the definition of the new !CHAOS "control service" paradigm.
The key features and development strategies of !CHAOS are:

- scalability of performances and size
- integration of all functionalities
- abstraction of services, devices and data
- easy and modular customization
- extensive data catching for performance boost
- use of high-performance internet software technologies.

In 2015 the !CHAOS project, partially supported by the CNS5, concluded the activities foreseen by the *"Premiale"* proposal[1]. Two main deliverables were scheduled for 2015: firstly the release of an Alpha version in June, as conclusion of the design study of all the tasks planned in the project and the development and integration of its core functionality; secondly the release, by the end of the year, of a Beta version where all the functionalities expected have been developed, integrated, tested and qualified. All deliverables and milestones expected by "Premiale" proposal have been achieved without significant deviations.
The project has been demonstrated the feasibility of building a scalable multi-purpose controls services provider based on the !CHAOS framework and on the INFN e-infrastructure allowing, with unprecedented flexibility, the monitoring, control and data acquisition, storage and analysis of any sensors, devices and SoS.

[1] https://web2.infn.it/chaos/images/pdfs/PropostaPremialeChaos.pdf

WP1: Management, dissemination and impact

Coordinator: Giovanni Mazzitelli
Staff: Alessandro Stecchi, Francesca Spagnoli, Eliana Gioscio, Luciano Catani, Pasquale Lubrano
Expected Deliverable:

- D.1 Final Management Report - ACHIEVED

Task 1.1: Management

The project has been expected to be complete in one year since the date of assignment of the resources. Actually, the project, approved by MIUR in October 2013, received INFN budget approval only in June 2014 and, because of various delays, was able to hire personnel only between December 2014 and March 2015. Moreover, on the 9th of April 2014, the project started and it has been organized the collaboration Kick Off meeting[2] in order to plan the activities and resources.

Budget distribution

In the following, the executive detailed budget distribution. The entire budget has been administrated by LNF and mainly consumed during the project. A small amount of residual resources (about 10.000 euro of travels) will be still used in 2016 for disseminating the project.

Reference	**Description**	**Value**
U102_121405	Travels	23.000
U103_130120	Consumables	12.000
U212_520110	Infrastructures	23.000
Sub Total:		*58.000*
LNF Staff	A. Michelotti, R. Gargana, F. Spagnoli, S. Caschera, Summer Students, E. Gioscio, M. A. Tota	384.970
CNAF Staff	M. Panella	78.027
RM2 Staff	C. Di Giulio	26.113
LNS Staff	Salvo Aurnia	39.060
PV Staff	Paolo Buzzi	5.000
Sub Total		*533.171*
Gran total		**591.171**

[2] https://agenda.infn.it/conferenceDisplay.py?confId=7867

The project has been carried on by 14 Full Time Equivalent (FTE) with the following task to be done:

Table 1 - (*) Temporary positions supported by the project budget. During the summer 2014 and 2015 a summer student position (1Ke/year) has been also supported; (+) Temporary positions supported by INFN

Staff	Role	Position	FTE (%)	WP	Tasks	Aff.
Giovanni Mazzitelli	employee	Senior researcher	40	WP1	Principal investigator	LNF
Giampiero di Pirro	employee	Senior technologist	20	WP3	Drivers developer	LNF
Alessandro Stecchi	employee	Senior technologist	40	WP1	Technical manager	LNF
Andrea Michelotti	employee *	Technologist	100	WP3	WP3 leader/Drivers development	LNF
Riccardo Gargana	employee *	Technologist	100	WP5	IT manager	LNF
Eliana Gioscio	training *+	fellowship	100	WP2	Framework/GUI/Developer	LNF
Francesca Spagnoli	training *	Postdoc fellowship	100	WP1	Comm. & Diss. manager	LNF
Michele Tota	training *	fellowship	100	WP5	IT/cloud manager	LNF
Ramon Orrú	training +	fellowship	10	WP%	IT/cloud manager	LNF
Salvatore Caschera	training *	Young fellowship	100	WP2	Framework/drivers developer	LNF
Sandro Angius	employee	Technician	10	WP5	IT/storage manager	LNF
Claudio Bisegni	employee +	Technician	100	WP2	WP2 leader/framework developer	LNF
Dario Spigone	employee +	Technician	10	WP5	Network manager	LNF
Paolo Ciufetti	employee	Technician	20	WP3/WP5	GUI developer/IT manager	LNF
Tomaso Tonto	employee	Technician	10	WP1/WP5	IT manager/web manager	LNF
Galletti Francesco	employee	Technician	20	WP3	Drivers/GUI developer	LNF
Olimpio Giacinti	employee	Technician	10	WP4	HW integration	LNF
Gianfranco Baldini	employee	Technician	10	WP4	HW integration	LNF
Dael Maselli	employee	Technician	10	WP5	IT/cloud manager	LNF
Massimo Pistoni	employee	Technician	20	WP5	IT manager	LNF
Salvo Aurnia	training *	Technician	100	WP3/WP4	Drivers developer	LNS
Salvatore Pulvirenti	employee	Technician	30	WP3/WP4	LNS task manager	LNS
Furia Enrico	employee	Technician	10	WP3/WP4	Drivers developer	LNS
Diana B. Filippo	employee	Technician	10	WP3/WP4	Drivers developer	LNS
Salvo Rita Cavallaro	employee	Technician	10	WP3/WP4	Drivers developer	LNS
Luciano Catani	employee	Senior researcher	40	WP1	Management and dissemination	TV
Gaetano Salina	employee	Senior researcher	10	WP4	WP4 leader/HW integration	TV
Claudio Di Giulio	training *	Postdoc fellowship	100	WP4	HW integration	TV
Bruno Checcucci	employee	Technician	35	WP4	PG task leader/ HW integration	PG
Paolo Buzzi	training *	fellowship	100	WP4	HW integration	PG
Enrico Fattibene	employee	Technologist	20	WP5	WP5 leader/cloud manager	CNAF
Matteo Panella	employee *	Technician	100	WP5	cloud manager	CNAF
Michele Michelotto	employee	Senior technologist	5	WP5	cloud manager	PD

The first Work Package (WP1) has been in charge of the coordination, communication and dissemination of the project. In order to achieve this task, staff regular meetings [3] for monitoring and coordinating the technical

[3] https://agenda.infn.it/categoryDisplay.py?categId=673

development have been organised. In December 2014 a two days training school[4] has been organized with 12 attendants. Moreover, in 2015 the WP1 staff has worked to a new proposal submitted to MISE and called *"Industria 4.0"* in collaboration with the IMA enterprise[5], Scuola di Ingegneria ed Architettura e Scuola di Disegno Industriale dell' Università di Bologna, ELEIDIA Università di Trento, JKU Johannes Kepler University, Linz (AU) and CNR NANO/NEST Scuola Normale Superiore di Pisa. This new project, called MaXima, has been approved and founded by MISE. Within this project, the !CHAOS group will be in charge of transferring the !CHAOS technology and cloud based applications in order to realize a prototype of a computing system for prognostics, fault prediction, data analysis, system monitoring and optimization.

Task 1.2: Dissemination

The WP1 staff has also worked on the communication and dissemination of the project through outreach events [5,9], seminars [1,3,4,6,710,11], conferences [2,8,12] and publications in national and international contexts and publications (see below). Furthermore, the staff take care of the maintenance of the national web site http://chaos.infn.it and the documentation the whole project https://opensource-confluence.infn.it/.

Presentations

[1] G. Mazzitelli - CSN5 meeting - Ferrara, 29 September 2014, Italy
[2] L. Catani - 10th International Workshop on Personal Computers and Particle Accelerator Controls, 14th - 17th October 2014, Karlsruhe, DE
[3] A. Michelotti - NI Week - Big Physics Symposium (Austin-TX) - August 2014, USA
[4] G. Mazzitelli - CCR Workshop, 27-30 May, 2014 Catania, Italy
[5] G. Mazzitelli - ANSA Scienza LAB gNE2014 – 30 May Campus X Tor Vergata, Italy
[6] A. Stecchi - 1st Synergy LNF-OAR Workshop, 16-17 April, 2014 Frascati, Italy
[7] E. Fattibene – Workshop di CCR sull'Infrastruttura Cloud, 15-17 December 2014, Napoli, Italy
[8] G. Mazzitelli - Controls, Monitoring and DAQ, HEP Software Foundation Workshop - 20-22 January 2015, SLAC - USA
[9] F. Spagnoli - A Cloud of Controls (IFAE), IFAE Conference - 8-10 April 2015, Rome – Italy
[10] E. Gioscio, !CHAOS, WIRE15 Workshop Impresa, Ricerca Economia, Scuderie Aldobrandini, 21 May, Frascati, Italy
[11] M. Tota, !CHAOS: a Cloud of Controls (CCR), CCR Workshop - 25-30 May 2015, Frascati - Italy
[12] R. Orrù, !CHAOS: un prototipo nazionale di infrastruttura open source per il controllo di sistemi distribuiti, 101° Congresso Nazionale della Società Italiana di Fisica - 21-25 September 2015, Rome – Italy
[13] F. Spagnoli, Cloud of Controls (itAIS),12° Conference of the Italian Chapter of AIS - 9-10 October 2015, Rome - Italy

[4] https://agenda.infn.it/conferenceDisplay.py?confId=8999
[5] http://www.ima.it/

Publications

[1] !CHAOS: a cloud of controls - MIUR project proposal, INFN-14-15/LNF, 19 November 2014

[2] First operational experience with the !CHAOS framework, PCaPAC 2014, Karlsruhe (DE), 14 - 17 October

[3] !CHAOS STATUS AND EVOLUTION, C. Bisegni, S. Caschera, C. Di Giulio, G. Di Pirro, L. G. Foggetta, R. Gargana, E. Gioscio, D. Maselli, G. Mazzitelli, A. Michelotti, R. Orrù, S. Pioli, F. Spagnoli, A. Stecchi, M. Tota, May 2015, Proceedings of IPAC2015, Richmond, VA, USA

[4] !CHAOS: a cloud of controls, S. Angius, C. Bisegni, P. Buzzi, l: Catani, S.R. Cavallaro, B Checcucci, P. Ciuffetti, B.F. Diana, C. Di Giulio, G. Di Pirro, F. Enrico, E. Fattibene, L. G. Foggetta, F. Galletti, R. Gargana, E. Gioscio, P. Lubrano, D. Maselli, G. Mazzitelli, A. Michelotti, M. Michelotto, R. Orrù, M. Panella, M. Piccini, M. Pistoni, S. Pulvirenti, G. Salina, F. Spagnoli, D. Spigone, A. Stecchi, T. Tonto, M. A. Tota, April 2015 Proceedings of IFAE2015, Rome, Italy

[5] **!**CHAOS: a prototype of a Cloud Computing infrastructure to control High Energy Physics systems in Italy, *C. Bisegni, S. Caschera, L. Catani, C. Di Giulio, G. Di Pirro, E. Fattibene, L. G. Foggetta, R. Gargana, E. Gioscio, D. Maselli, G. Mazzitelli, A. Michelotti, M. Michelotto, R. Orrù, M. Panella, M. Pistoni, F. Spagnoli, M. Piccini, A. Stecchi, T. Tonto, M. Tota, October* 2015, Proceedings of itAIS2015, Rome, Italy

WP2: Software Development

Coordinator: Claudio Bisegni
Staff: Salvatore Caschera, Andrea Michelotti
Expected Deliverable:

- D.2.1 !CHAOS as a Service – ACHIEVED

Expected Milestones:

- M2.1 Common Framework – ACHIEVED
- M2.2 Metadata Service – ACHIEVED
- M2.3 Live and History Data Service – ACHIEVED
- M2.4 Control Unit, Execution Unit and User Interface – ACHIEVED
- M2.5 Test and Qualification of all the Service – ACHIEVED

The objectives of the WP2 have been addressed to the development of the remaining part of the !CHAOS architecture. The !CHAOS R&D actually started 5 years ago with the idea to design a new controls architecture able to overcome today standards and guarantee the technical requirements and performance of a novel accelerator complex and large experiments.

Task 2.1: Common Framework

For the above reasons, the work on the !CHAOS framework has been focused on the stabilization of the basic structures and on the development of the DirectIO, which enables high throughput communications among the !CHAOS nodes[6]. The

[6] The !CHAOS general architecture consists of five node five nodes of the system: data acquisition (CU–Control Unit), presentation (UI–User Interface), proxies/indexing/storage (CDS-!CHAOS Data Service), data handling (EU–Execution Unit) and system state information (MDS-Metadata Service); three communication channels: RPC, direct IO, Events.

DirectIO is a dual channels system, one for the transfer of data and the other one for sending service messages. It provides dynamic connections (client) and endpoints (server) that can allocate channels (client/server) exposing APIs. Furthermore, the RPC (Remote Procedure Call) communication channel has been optimised to reuse sockets and reduce memory consumption, especially for embedded systems (ARM Board). The relevant changes to the RPC communication channel have been two:

- the development of a front-end HTTP REST (Representational State Transfer),
- the development of a fail-over communication channel with more client endpoints.

The HTTP front-end enables to recall, in a synchronous way, the RPC functions of a !CHAOS node. This process, for some specific applications, simplifies the connection with environments using HTTP massages to cooperate, for instance within some nodes in the cloud systems, old devices and embedded systems not able to support the load of !CHAOS framework or not compatible with !CHAOS minimal system requirements. Moreover, the new communication channel with multiple endpoints allows managing the sending of messages and round-robin requests with nodes exposing the same set of RPC functions (such as more metadata services). It manages the forwarding of the requested files on a node and its retransmission on a still available one. Control and management of unavailable servers should be still improved.
Two management systems have also been created: one for the publication of information related to resources and heart beating to data services; and the other one for the publication of metrics on consoles or files. The two communication channels - RPC (client/server) and DirectIO (client/server) - have been updated to provide metrics.

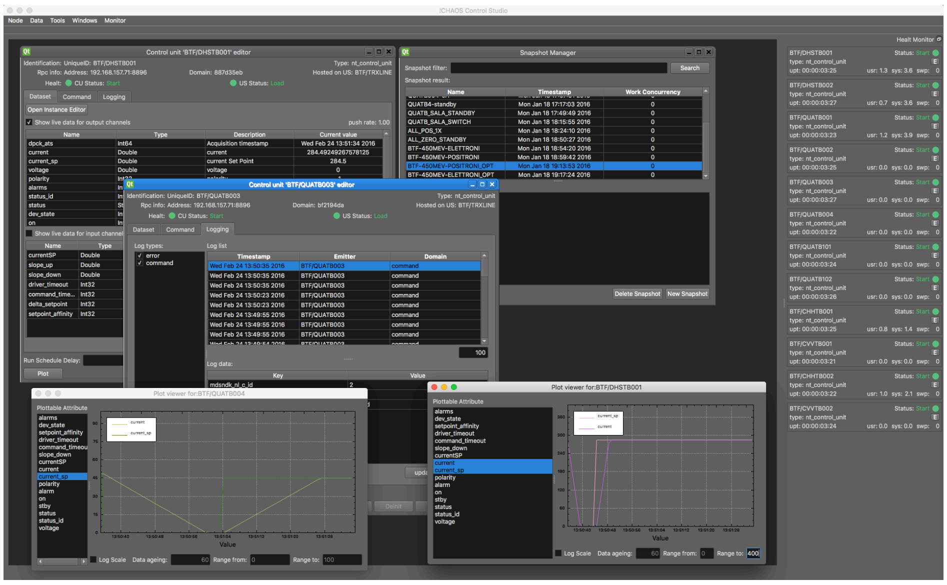
Functionalities	**Milestones released with the alpha version at 31 July 2015**	**Deliverable foreseen with the beta version the 31 December 2015**
Metric system	done	done
Heath system	done	done
HTTP Rest RCP call	preliminary	done
RCP Channel multi-endpoint (fail-over)	preliminary	done
DirectIO	done	done

Table 2 - Task 2.1 Functionalities and/or products released and expected (here and in the following tables “preliminary” means case studied and partially developed, “done” means that all the foreseen functionalities have been implemented)

Task 2.2: Metadata Service

The Metadata Service (MDS), storing the information about the state, the type and the structure of all nodes, answers to search queries and manage the control algorithm and tasks, has been completely rewritten in C++. It has been structured in two levels, a server and a client library for querying and monitoring the various nodes registered.
The server employs an abstraction of the back-end database to allow writing drivers that will enable the use of different software solutions, like a database. The current implementation is based on MongoDB.

The client is divided in two sections: querying of metadata and monitoring of nodes. To each server, an API has been associated to a client proxy simplifying the creation of data required to call specific APIs. The monitoring phase allows developing C++ classes, as '*handles*' used to receive, by specifying a certain range and the value of an attribute in a dataset associated with a key published in the !CHAOS Control Studio (the on line motor and control system of !CHAOS datasets and value, see below) or its non-presence.
The !CHAOS Control Studio (CCS) is a C++ application developed for managing and controlling metadata, unit servers and control units. It allows representing input and output over time through a display window of plots. APIs published through the RPC metadata server have been deeply described and documented[7].



Control unit 'BTF/DHSTB001' editor
Identification: UniqueID: BTF/DHSTB001 Type: nt_control_unit
Rpc info: Address: 192.168.157.71:8896 Domain: 887d35eb Hosted on US: BTF/TRXLINE
Healt: CU Status: Start US Status: Load

✓ Show live data for output channels push rate: 1.00

Name	Type	Description	Current value
dpck_ats	Int64	Acquisition timestamp	Wed Feb 24 13:51:34 2016
current	Double	current	284.49249267578125
current_sp	Double	current Set Point	284.5
voltage	Double	voltage	0

Show live data for input channel

Name	Type
currentSP	Double
slope_up	Double
slope_down	Double
driver_timeout	Int32
command_time...	Int32
delta_setpoint	Int32
setpoint_affinity	Int32

Run Schedule Delay:

Snapshot Manager
Snapshot filter:
Snapshot result:

Name	Timestamp	Work Concurrency
QUATB4-standby	Mon Jan 18 17:17:03 2016	0
QUATB_SALA_STANDBY	Mon Jan 18 17:49:49 2016	0
QUATB_SALA_SWITCH	Mon Jan 18 18:15:55 2016	0
ALL_POS_1X	Mon Jan 18 18:24:10 2016	0
ALL_ZERO_STANDBY	Mon Jan 18 18:50:27 2016	0
BTF-450MEV-ELETTRONI	Mon Jan 18 18:54:20 2016	0
BTF-450MEV-POSITRONI	Mon Jan 18 18:59:42 2016	0
BTF-450MEV-POSITRONI_OPT	Mon Jan 18 19:13:53 2016	0
BTF-450MEV-ELETTRONI_OPT	Mon Jan 18 19:17:24 2016	0

Control unit 'BTF/QUATB003' editor
Identification: UniqueID: BTF/QUATB003 Type: nt_control_unit
Rpc info: Address: 192.168.157.71:8896 Domain: bf2194da Hosted on US: BTF/TRXLINE
Healt: CU Status: Start US Status: Load

Log types: ✓ error ✓ command
Log list

Timestamp	Emitter	Domain
Wed Feb 24 13:50:35 2016	BTF/QUATB003	command
Wed Feb 24 13:50:35 2016	BTF/QUATB003	command
Wed Feb 24 13:50:35 2016	BTF/QUATB003	command
Wed Feb 24 13:50:23 2016	BTF/QUATB003	command
Wed Feb 24 13:50:23 2016	BTF/QUATB003	command
Wed Feb 24 13:49:55 2016	BTF/QUATB003	command
Wed Feb 24 13:49:55 2016	BTF/QUATB003	command
Wed Feb 24 13:49:55 2016	BTF/QUATB003	command

100

Log data:

Key	Value
mdsndk_nl_c_id	2

Healt Monitor

Node	Status	type	upt	usr	sys	swp
BTF/DHSTB001	Start	nt_control_unit	00:00:03:25	1.3	3.6	0
BTF/DHSTB002	Start	nt_control_unit	00:00:03:27	0.7	3.6	0
BTF/QUATB001	Start	nt_control_unit	00:00:03:23	1.2	3.9	0
BTF/QUATB002	Start	nt_control_unit	00:00:03:25	0.0	0.0	0
BTF/QUATB003	Start	nt_control_unit	00:00:03:27	0.0	0.0	0
BTF/QUATB004	Start	nt_control_unit	00:00:03:22	0.0	0.0	0
BTF/QUATB101	Start	nt_control_unit	00:00:03:24	0.0	0.0	0
BTF/QUATB102	Start	nt_control_unit	00:00:03:26	0.0	0.0	0
BTF/CHHTB001	Start	nt_control_unit	00:00:03:25	0.8	1.4	0
BTF/CVVTB001	Start	nt_control_unit	00:00:03:21	0.0	0.0	0
BTF/CHHTB002	Start	nt_control_unit	00:00:03:22	1.0	2.1	0
BTF/CVVTB002	Start	nt_control_unit	00:00:03:24	0.0	0.0	0

Fig. 1 - !CHAOS Control Studio (CCS): the software hallow to monitor and controls all datasets and value in !CHAOS framework

Functionalities	**Milestones released with the alpha version at 31 July 2015**	**Deliverable foreseen with the beta version the 31 December 2015**
Unit Server management	done	done
CU management	done	done
CU parameter monitor	done	done
CU Change input dataset attribute	done	done
Command template	preliminary	done
Forwarding command template to node	preliminary	done
Control unit and unit server monitoring	done	done
Node group management	no	preliminary
Command group forwarding	no	preliminary
Command inspection	no	preliminary
Command history	no	preliminary
Search on historic data	No	preliminary
Save & Restore	preliminary	done

Table 3 – Task 2.2 Functionalities and/or products released and expected

[7] https://opensource-confluence.infn.it/confluence/display/chd/High+Level+API

Task 2.3: Data Service (Live and history data)

The !CHAOS Data Service (CDS) providing data proxy, indexing, history and stage management, has been written in C++ and replaces the previous direct communication between control units and user interfaces through a centralised caching system. It enables to storage the data in a live cache and historicizes data through a single data stream. A specialized DirectIO communication channel (client/server) has been studied and developed to manage, query and search data and datasets. Moreover, the CDS exports all its functionalities through the DirectIO and implements different abstractions for managing the backend caching, file system and indexing database. Up to now, different implementations for each of them have been provided:

- Caching
    - Memcached (no fail over)
    - Couchbase (failover embedded in the driver)
- Filesystem
    - Posix
- Index Database
    - MongoDB

The Data Service functionalities are mainly two:

- Response to queries (insert, last date, simple query history) on the control unit (CU), interface unit and metadata service (client and server)
- Indexing of stage files.

The Data Service manages the data flow to the storage, caching and/or placing data in the stage area. Queries on the indexed data are limited to a time range for each device at the time (no join) and they are asynchronous.

The data indexing is in a preliminary development phase, the recovery of a stage file that has not been completely indexed it is not possible up to now. Ageing management and the customisation of the indices has not yet been enabled. The CDS uses the infrastructural part of the common metrics to publish performance information, such as:

- Storage Driver, to manage the storage and reading of data-pack in the filesystem.
    - **w_packet_count,** the packet written per second
    - **w_kb_sec,** the bandwidth written per second
    - **r_packet_count,** the packet read per seconds
    - **r_kb_sec,** the bandwidth read per second
    - **flush_count,** the total flush done by the driver per second
    - **packet_count**: the total count of packet sent for second
    - **kb_sec**: the total bandwidth in kilobytes sent per second
- Cache Driver, to manage the storage and reading of data-pack in the cache.
    - **set_packet_count**, the packet set into the cache per second
    - **set_kb_sec**, the bandwidth set into the cache per second
    - **get_packet_count**, the packet read from the cache per second
    - **get_kb_sec**, the bandwidth read from the cache per second
    - **packet_count**: the total count of packet sent for second
    - **kb_sec**: the total bandwidth in kilobytes sent per second
- DataWroker, is the subsystem processing inputs in the live and historical stage area.

- **nput_packet_count**, is the number of packet elaborated (set in cache or queued for fs write) per second
- **input_badnwith_kb**, is the bandwidth elaborated (set in cache or queued for fs write) per second
- **output_packet_count**, is the number of packet that has been written on the filesystem per second
- **output_bandwith_kb**, is the bandwidth that has been written on the file system per second
- **queued_packet_count**, is the number of packet that are currently stored in the queue
- **queued_packet_size_kb**, is the size that is currently stored into the queue.

Functionalities	**Milestones released with the alpha version at 31 July 2015**	**Deliverable foreseen with the beta version the 31 December 2015**
Insert data-pack	done	done
Data-pack indexing	preliminary	done
Ageing data management	no	preliminary
Asynchronous query	preliminary	done
Synchronous and paged query	no	preliminary
Custom index creation and query	no	preliminary
No index crash recovery	preliminary	done

Table 4 – Task 2.2 Functionalities and/or products released and expected

Task 2.4: Control Unit, Execution Unit and User Interface

The Control Unit (CU) Toolkit has been modified to enable the publication of four datasets:

- Output dataset attribute, the values of dataset output channels are updated at each run.
- Input dataset attribute, the input dataset values are published at each update of the control unit.
- Custom variable dataset, which enables to publish custom variables. The channel is updated every time the variables are modified.
- System dataset attribute, which is a channel system grouping the information used by the metadata server and the !CHAOS Control Studio for each control unit.

The control unit provides 4 caches area (one for each dataset) where the developer can write (output channels and custom attributes) and from which he can read the value (input and system attributes). Each dataset is created automatically by reading the dataset configuration made by the developer.
The framework publishes the datasets in which at least one single attribute has been modified by the developer or by the framework (e.g. when is required to valorised the attributes of inputs).
The CU Toolkit enables to create a Unit-Server. Each CU-toolkit process is a Unit-Server instance, which allows publishing the list of the control units available into it and configured on the metadata server. For each of them, through the RPC, it dynamically creates instances by specifying different parameters (load param, driver list, input dataset attribute default values).

A new abstract model in the low class of the CU has been implemented to restore the save data value. An MDS specific command enables the CU to load its status directly saved into the CDS and restore a new cache.
The developer in this way can see both the current and the future required status and can decide to apply or not the stored dataset.

Functionalities	**Milestones released with the alpha version at 31 July 2015**	**Deliverable foreseen with the beta version the 31 December 2015**
Control unit instantiation	done	Improved
Control unit initial configuration	done	Improved
Control unit command logging on MDS	no	preliminary
Control Unit command publishing	done	Improved
Control Unit driver publishing on MDS	no	preliminary
Partial dataset attribute publication	no	preliminary
Save & Restore	preliminary	done

Table 5 – Task 2.4 Functionalities and/or products released and expected

WP3: Front end drivers

Coordinator: Andrea Michelotti
Staff: Riccardo Gargana, Claudio Bisegni, Eliana Gioscio, Salvatore Caschera, Giampiero Di Pirro, Salvo Pulvirenti, Salvatore Cavallaro, Benedetto Diana, Enrico Furia, Paolo Buzzi, Claudio Di Giulio, Pierandrea Conti, Paolo Ciuffetti, Francesco Galletti.
Expected Deliverables:

- D3.1 Realization of a control system dedicated to ESCO use case - ACHIEVED
- D3.2 Prototype of critical elements of FP control system - ACHIEVED

Expected Milestones:

- M3.1 Definition in collaboration with wp4 the software and hardware specification and requirements - ACHIEVED
- M3.2 LabVIEW® and Tango compatibility layers at driver and/or GUI level – ACHIEVED (Tango layer compatibility only at design stage)
- M3.3 Integration of !CHAOS in LabVIEW® framework – ACHIEVED

The main objective of WP3 has been to provide an initial library of drivers, interfaces and compatibility layers built on real use cases to be used as reference to minimize development time and the expert knowledge of the hardware (HW). The job has been carried out in strong collaboration with WP4.

Task 3.1: !CHAOS Implementation for ESCO (Energy Service Company)

The ESCO use case has explored different fields of application of the !CHAOS infrastructure: to investigate a use case where !CHAOS could has been useful and in collaboration with industries; to investigate the use of !CHAOS on low cost and low consumption device like System on Chip (SoC).
The objective of this use case has been to monitor the environmental parameters of a given building (ED36 of INFN-LNF) and apply a feedback to the UTA (Air Treatment Unit) conditioning system. The final aim has been to minimize power consumption while maintaining a constant level of comfort perceived. The WP3, in agreement with WP4, has decided to implement the monitoring through lightweight nodes connected to humidity, temperature and CO2 sensors communicating via zigbee with a collector node. The Control Unit collects the data from the different sensors and pushes them into !CHAOS. The data can be visualized on a WEB page and will be used to apply a feedback to the UTA. For this use case the WP3 has realized the low and high-level drivers, Control Units, the HTTP REST APIs to implement the GUI inside the WEB (Fig. 2).

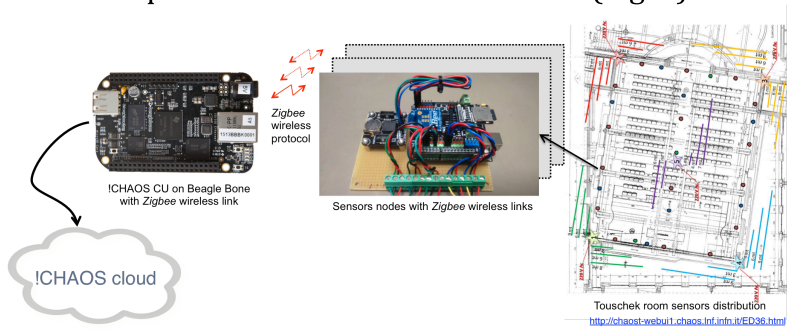

Zigbee wireless protocol
!CHAOS CU on Beagle Bone with Zigbee wireless link
Sensors nodes with Zigbee wireless links
!CHAOS cloud
Touschek room sensors distribution
http://chaost-webui1.chaos.lnf.infn.it/ED36.html

Fig. 2: Test bed ESCO – low cost SoC controller wireless (zigbee) connected to microcontroller SoC monitoring the sensor (pressure, temperature, CO_2, ecc) – The WEB interface of the ED32 at LNF (see fig 7 for details)

The data acquisition chain has been realized with low cost SoC, where a BiggleBone has been connected from one side to the !CHAOS cloud pushing data and from the other side to a zigbee wireless local area to six Arduino's SoC collecting the sensors (pressure, temperature, humidity, CO_2, ecc).
The control of the UTA has also been developed through an in house electric panel developed by the WP4, where a beagle CPU runs a Control Unit able to drive shutters and monitor UTA's input and output temperatures. The WP3 realized also a WEB control page to manually controlling the UTA.
This task has been partially operated in collaboration with the ESCO ADF Solaris Company, partner of the project, which has designed the sensors distribution and requirements and is interested to exploit the results of the job done.

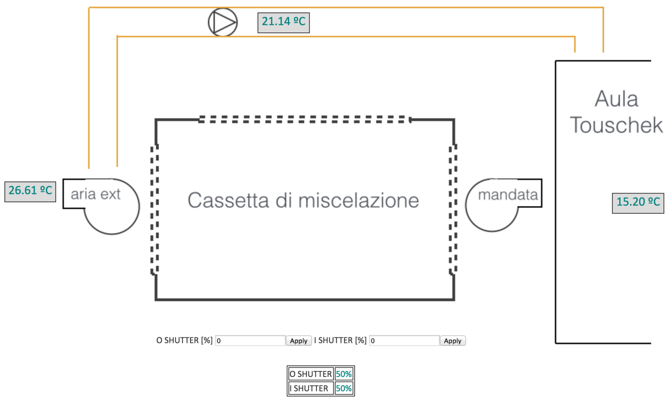


Fig. 4: !CHAOS UTA WEB control interface

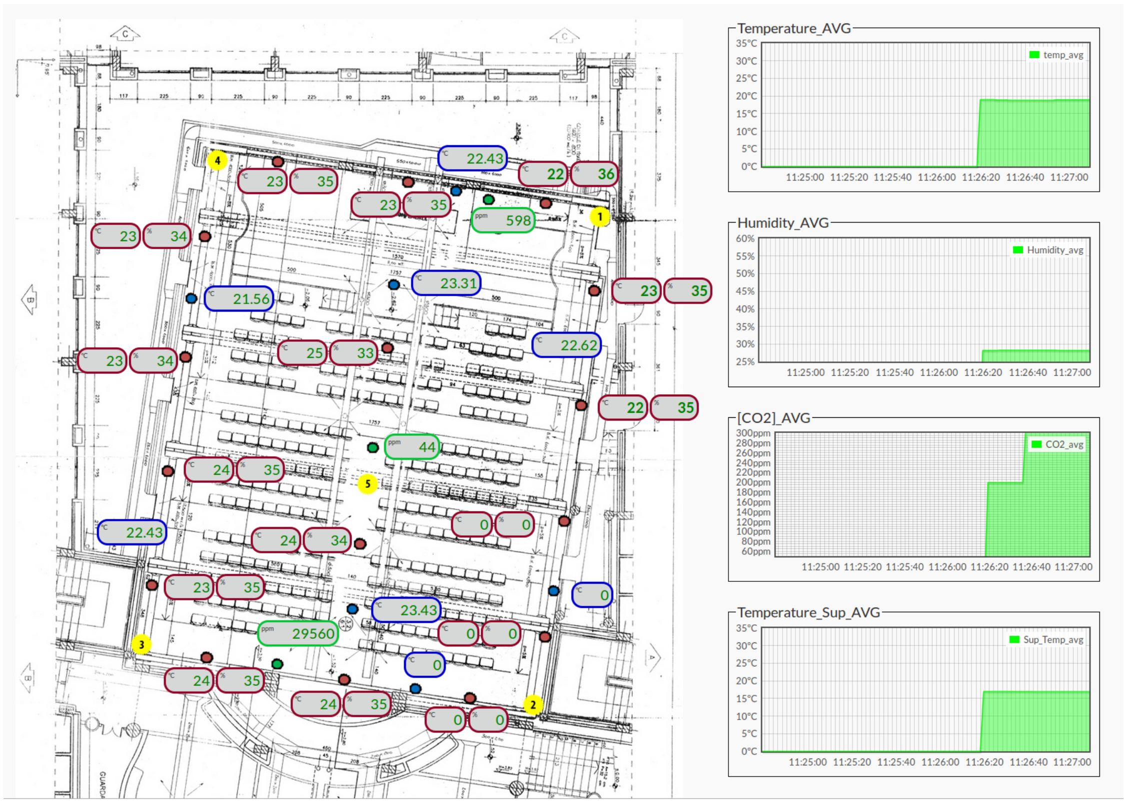


Figure 4 ESCO system operating at INFN-LNF: values of temperature, humidity and concentration of CO_2

Task 3.2: !CHOAS implementation for FP (Flagship Project)

In late 2013, the SuperB INFN Flagship Project has been shutdown and the !CHAOS project fully evolved from a candidate of Distributed Control Systems (DCS) & Data Acquisition (DAQ) for accelerators to the today R&D to exploit innovative solution for DCS and its applications for industries and the society. Anyhow, we strongly have developed the application in the accelerators field, demanding test bench for our framework, as well as by forecasting future applications on INFN accelerators. In this contest, the LNF group has tested !CHAOS for controlling and monitoring the Beam Test Facility (BTF), while the LNS group applied !CHAOS in the Catania Laboratory to ion beam sources on the specific hardware provided by National Instruments, commercial partner of the project.

The BTF is a beam transfer line, part of the DAFNE complex, optimized for the production of a single electron/positron in the range of 25-700 MeV, mainly used for HEP detectors testing and calibration. The facility hosts many different devices to monitor and control the beam quality and characteristics, typically handled by non-expert users. The test has been focused on exploiting the !CHAOS framework in a real operating condition, testing the IT infrastructure and the Graphical User Interface (GUI), verifying the stability and capability of the information service – Meta Data Service (MDS) and storing – the !CHAOS Data Service (CDS).

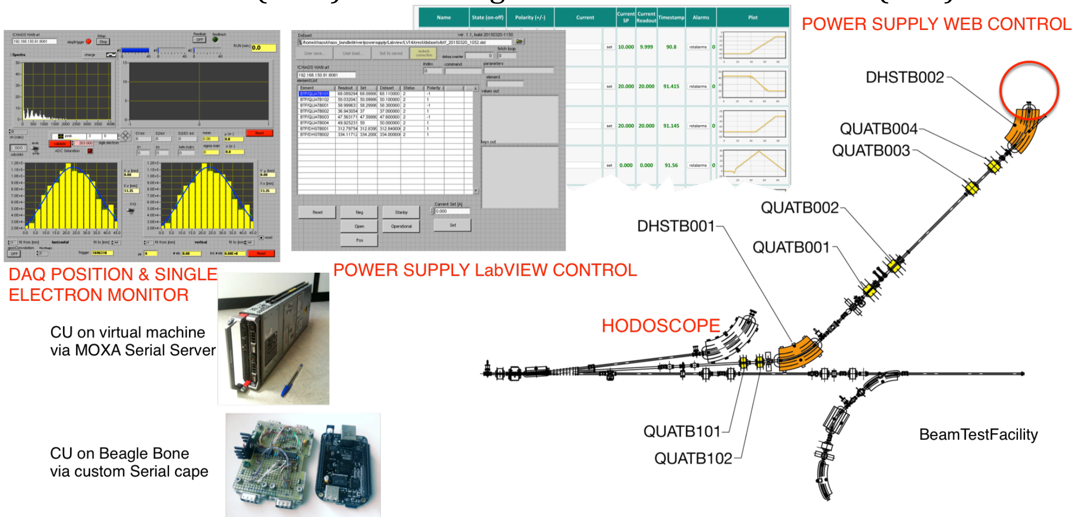

POWER SUPPLY WEB CONTROL
DHSTB002
QUATB004
QUATB003
QUATB002
DHSTB001
QUATB001
DAQ POSITION & SINGLE ELECTRON MONITOR
POWER SUPPLY LabVIEW CONTROL
HODOSCOPE
CU on virtual machine via MOXA Serial Server
CU on Beagle Bone via custom Serial cape
QUATB101
QUATB102
BeamTestFacility

Fig. 5: Test bed@BTF: on the left the DAQ monitor displaying the single electron energy spectra from the BTF electromagnetic calorimeter; below the XY position monitor from scintillating fibbers detector; the LabVIEW and WEB power supply manual controllers, the BTF layout where the magnet under controls are named.

One of the objectives of the tests in BTF has been to add DAQ functionalities to the DCS. The first challenge has been to overcome the problem related to the BTF HW configuration, in particular the machines interfacing the DAQ system, were VME controllers are quite old and running on Linux 2.4 kernel with maximum 512MB RAM. This poor configuration did not allow running a full-featured Control Unit (CU). For this purpose, in this test release of !CHAOS it has been added the support for old Bare Metal Devices that could not run a full CU/UI !CHAOS software. We have built a very light layer completely decupled from the core layers of !CHAOS, this provides REST APIs to push/pull live data in/out from the !CHAOS cloud based infrastructure. This is a key functionality allowing to access

to any HW device/WEB browser/application of the !CHAOS cloud based resources in a lightly and simple way. For this use case, the WP3 realized the low and high-level drivers, Control Units, the REST interfaces to implement GUI inside LabVIEW and the WEB.
The same REST APIs has been used to connect to the !CHAOS cloud device running on particular environments, where it has not been possible to embed !CHAOS native software (Fig. 6).

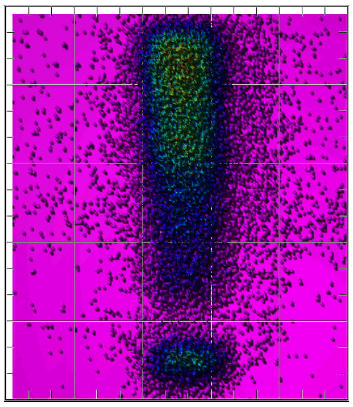
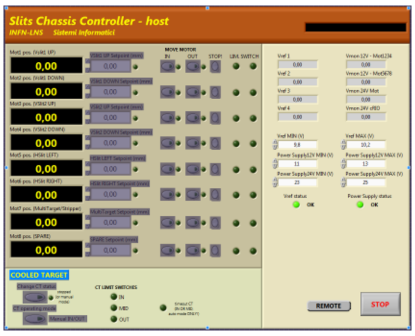


Fig. 6: Image detected by a Medipix (particle tracking pixel detectors) interfaced with !CHAOS cloud: in this case a python scripts has been realized to interface the HW with the HTTP REST APIs.

The final implementation of !CHAOS in BTF has been able to control it for two weeks without interruptions due to the control system. A concurrent environment based on virtual devices that mime the production installation has been used to decuple the development of control algorithms and interfaces from the development of HW drivers, matching in this way its high availability with improvement and bug fixing needs.
Southern National Laboratories (Laboratori Nazionali del Sud, LNS) have worked on a use-case in which !CHAOS has run on National Instruments Linux RT cRIO devices to monitor and control LNS ion beam source. The work has consisted in adapting the existing hardware subsystem to push data into the !CHAOS cloud infrastructure, access and show the data via a LabVIEW GUI by modifying the query to the !CHAOS CDS. It should be noted that some extra coding work has been needed since the framework at that time did not have full support for the ARM family processor installed on the cRIO-9068 chassis.



Mot1 pos. (VSlit1 UP) 0,00 — VSlit1 UP Setpoint (mm) 0,00
Mot2 pos. (VSlit1 DOWN) 0,00 — VSlit1 DOWN Setpoint (mm) 0,00
Mot3 pos. (VSlit2 UP) 0,00 — VSlit2 UP Setpoint (mm) 0,00
Mot4 pos. (VSlit2 DOWN) 0,00 — VSlit2 DOWN Setpoint (mm) 0,00
Mot5 pos. (HSlit LEFT) 0,00 — HSlit LEFT Setpoint (mm) 0,00
Mot6 pos. (HSlit RIGHT) 0,00 — HSlit RIGHT Setpoint (mm) 0,00
Mot7 pos. (MultiTarget/Stripper) 0,00 — MultiTarget Setpoint (mm) 0,00
Mot8 pos. (SPARE) 0,00 — SPARE Setpoint (mm) 0,00

Vref 1 0,00 — Vmon 12V - Mot1234 0,00
Vref 2 0,00 — Vmon 12V - Mot5678 0,00
Vref 3 0,00 — Vmon 24V Mot 0,00
Vref 4 0,00 — Vmon 24V cRIO 0,00
Vref MIN (V) 9,8 — Vref MAX (V) 10,2
Power Supply12V MIN (V) 11 — Power Supply12V MAX (V) 13
Power Supply24V MIN (V) 23 — Power Supply24V MAX (V) 25
Vref status OK — Power Supply status OK

Firstly, a test development suite has been set-up, with a virtualized environment (VMware player). For greater flexibility, two virtual machines have been installed, one running Windows7 with LabVIEW 2014 plus the C/C++ Development Tools for NI Linux RT Eclipse Edition[3] and one running Ubuntu 14. The former has been needed to synthesize the FPGA and perform some quick tests while the latter, thanks to the GCC ARM tool-chain for Linux, has mostly acted as a cross compilation machine for the whole !CHAOS framework, since it has not been possible to natively compile the whole source code on the cRIO due to its relatively limited hardware resources.

Secondly, we has started to practice with C code natively running on the cRIO and interacting with the FPGA: a couple of tests have been carried out, basically synthesizing extremely simple code to the FPGA to play with, and making sure everything was working as intended. This was necessary because we had always relied on LabVIEW to generate code for the cRIO, and we had never had the opportunity to try the FPGA Interface C APIs provided by National Instruments until then.
Finally, we have taken the original LabVIEW source code interacting with the FPGA and ported it to C/C++, appropriately integrating everything with !CHAOS.
As of the time of writing, !CHAOS framework is nearing a beta release so there will be room for improvements in many areas. We have now to start accurate performance profiling for the LNS use-case, but early, rough tests seem to show promising results.

Task 3.3: LabVIEW® & Tango compatibility layer

The aim of this use-case has been the integration of !CHAOS into LabVIEW, a product from National Instruments, partner of the project, used in accademia and industrial automation and controls applications worldwide. For this use case, VIs (LabVIEW modules) have been written to access the !CHAOS resources, this has allowed to easily build control GUIs also for the other use case of the project (see task 3.2 for applied use-case). The task to integrate LabVIEW inside !CHAOS has been much more complex, even because we had no access to the internals of LabVIEW and we had to use the SDKs that National Instrument provided. For this purpose, a new National Instrument promising architecture has been selected, the NI9068, composed by a controller and a FPGA. This is the ideal architecture to implement a !CHAOS CU on the controller side and interfacing LabVIEW to drive and program the FPGA. To include the ARM7V CPUs of cRIO architecture into !CHAOS, the SDK from National Instrument has been used, it has written a LabVIEW application communicating with a *main* C running on the controller. Next steps include a Control Unit able to communicate with the LabVIEW FPGA application in order to acquire data (see task 3.2 for applied use-case).
In the meantime, a study on Tango layer compatibility has been performed and a preliminary integration of the Tango !CHAOS software has been developed, although far from completion, this mainly because we have preferred to work on the MATLAB integration wildly used in accelerator use-cases. We have also developed and tested an interface between the !CHAOS framework and the MATLAB environment. Moreover, we have adapted the open source MATLAB toolbox middle layer for particle accelerators, developed and used in many accelerators plants, to the DAFNE collider, interfacing with the !CHAOS framework.

Functionalities	Milestones released with the alpha version at 31 July 2015	Deliverable foreseen with the beta version the 31 December 2015
Power Suppy abstraction and Driver OCEM-PS	done	done
Serial abstraction and driver	done	done
Modbus abstraction and driver	done	done
VME abstraction and driver	done	done
Chaos independent debug layer	done	done
Sensor abstraction	done	done
Zigbee collector driver	done	done
Sis3800 driver(TDC)	done	done
Hiadc (ADC)	done	done
Caen 513 (PIO)	done	done
Caen 965 (QDC)	done	done
Caen 792 (QDC)	done	done
Libera Brilliance (ADC) from BPM	done	done
Wall current monitor driver	done	done
UTA drivers	done	done
Other BTF diagnostic drivers (table, oscilloscope, …	no	preliminary
Automatic build system	done	done
Miscellaneous tools to maintain code	done	done
Test engine	done	done
CREST server to access CHAOS resources via CREST	done	done
CREST client, small portable C library to interface bare bone targets with CHAOS via REST	done	done
JavaScript client engine for CHAOS WEB interfaces	done	done
LabVIEW compatibility layer (native, CREST)	done	done
Support for ARMV7V architectures	done	done
Support for ARMV5 architectures	done	done
Support for CRIO architectures	preliminary	done
TANGO compatibility layer	preliminary	preliminary
MATLAB compatibility layer	preliminary	done

Table 6 – Task 3.1, 3.2 and 3.3 Functionalities and/or product released and expected

WP4: Hardware Development

Coordinator: Gaetano Salina
Staff: Andrea Michelotti, Riccardo Gargana, Eliana Gioscio, Paolo Buzzi, Bruno Checcucci, Pasquale Lubrano, Claudio Di Giulio, Olimpio Giacinti, Gianfranco Baldini.
Expected Deliverables:

- D4.1 ESCO HRP - ACHIEVED
- D4.2 FP HRP - ACHIEVED
- D4.3 Prototypes hub and gateway G-HRP - ACHIEVED

Expected Milestone:

- M4.1 Definition in collaboration with WP3 the software and hardware specification and requirements - ACHIEVED

Task 4.1: ESCO HRP implementation

In 2015 the Hardware Reference Platform (HRP) implementation for the ESCO use case has been completed, this includes: collecting the ESCO requirements, defining and testing the HRP, implementing the sensors interconnection and topological structure for both wired and wireless solutions, developing the first revision of the reference platform for its assessment, test release and qualification. This has been developed to realize a generic "!CHAOS box" formed by an ADC/DAC controller, relays and zigbee for a wireless connection to sensors to be used in different applications (as planned in Task 4.3 Wired and wireless network connections and Task 4.4, General purpose HRP).

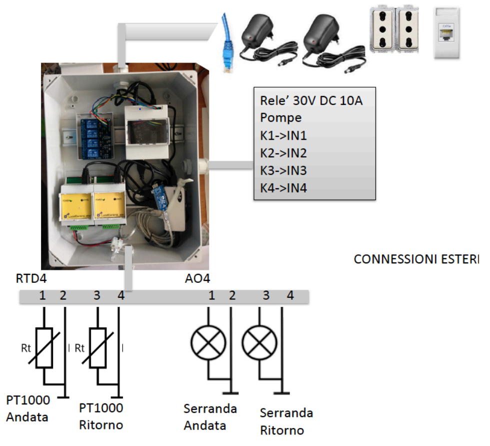


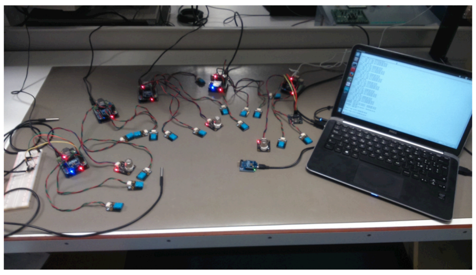
Fig 7 !CHAOS UTA control electric panel

In order to put into practice this use case in a real environment, we have decided to implement it in the Touschek auditorium of the building 32 at the Laboratori Nazionali di Frascati of INFN. The ADF Solaris ESCO Company, partner of the project, have defined the requirements to monitor the Touschek auditorium: sensors needed, specifications, positioning, accordance to the UNI CEI standard 11352, ect. The design defined by ADF Solaris has been implemented by INFN PE, in charge to test the hardware needed and arrange a first prototype. At the same time, the LNF staff has developed the drivers and the CUs. In June 2015, the PE prototype has been assembled in a 6 clients box equipped with sensors and a microcontroller connected wireless to a master controller box and installed in the auditorium to test communication, reliability, stability and perform the first measurements. Another box has been assembled to monitor and control the UTA (fig. 7, see also previous chapter showing the GUI interfaces). In the last months of 2015, the system has been wildly tested, demonstrating the feasibility of a "Sistema di Supervisione e Gestione di Sistemi Energetici per ESCO (SSGSEE)" proposed, based on a low cost "system on chip" and a mixed wireless/wired network to control a wide area through an open, accessible, scalable software, such as

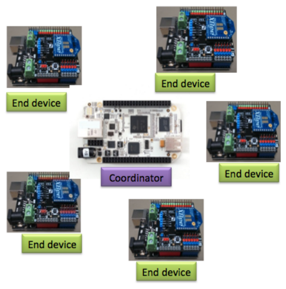
!CHAOS. At the same time, this application has also helped us to test the IT infrastructure where data where collected.

Task 4.2: FP HRP implementation

The work of this task has been to support the implementation of the two use cases dedicated to accelerators, as described in the previous chapter (WP3). The job has mainly consisted in assembling the diagnostics devices to control and develop the interfaces for the beagle bone with the accelerators devices.



The LNF Staff has organised various tests at the BTF - Frascati Beam Test Facility already described in pervious chapter – to proof the !CHAOS functionality in a real accelerator environment. The first test has been done in March 2014 to debug the framework and attempt to control the BTF line magnet trough MOXA serial interface and CU completely running on a virtual machine. During the second run, it has been implemented a mixed infrastructure based on a beagle bone, equipped with a 485 serial drivers developed by the DAFNE control Team and a virtual machine interface with several MOXA serial devices, successfully controlling the beam and it's parameters. Moreover, for the same test, a LabVIEW GUI and JavaScript GUI for magnets have been developed.
The !CHAOS framework has been applied on the LNS accelerators' ion sources. With the support of National Instruments (NI), who has provided for free the hardware, a test case based on the new cRIO-9068 has been designed. The objectives and the results has been already described in the previous chapter, nevertheless part of the job has been dedicated to the signal conditioning needed to interface the new hardware.



Task 4.3 - Identification of the operating standards for wired and wireless network connections

In order to develop the local wireless network for the ESCO use case and the LAN infrastructure to implement the !CHAOS based controls for the BTF use case, part of the job has been dedicated to study the best solution and implement two use cases. In particular for the ESCO use case, the zigBee solution has been chosen and tested in a standalone system similar to the one implemented in the real use case. The scope of the job has been to understand performance, nose resistance, stability and scalability of the chosen hardware, coming out from a auditorium full of electromagnetic noises, such as during conferences hosting hundreds of people with laptops connected via wifi, cellular phones, etc. No disturbance has been detected.

Task 4.4 - General purpose HRP

Actually, all the jobs described below have been explored and a chaos box based on a board CoS devices able to host the !CHAOS infrastructure has been developed. The developments done have been addressed to implement an open and modular system that can be re-used in any other application.

WP5: Computing, Storing and Access Policy

Coordinator: Enrico Fattibene
Staff: Sandro Angius, Paolo Ciuffetti, Riccardo Gargana, Eliana Gioscio, Dael Maselli, Ramon Orrù, Massimo Pistoni, Dario Spigone, Tomaso Tonto, Michele Tota, Matteo Panella, Michele Michelotto
Expected Deliverable:

- D5.1 Implementation of a cloud based infrastructure – ACHIEVED

Expected Milestones:

- M5.1 Implementation of the hardware infrastructure – ACHIEVED
- M5.2 Implementation of cloud IaaS and PaaS services and infrastructure – ACHIEVED
- M5.3 Access Policies implementation – ACHIEVED
- M5.4 Qualification, testing and documentation – ACHIEVED

The objective of this Work Package has been to find and implement the IT solutions to deploy the !CHAOS framework on a Cloud infrastructure (!CHAOS as a Service). At the same time, this WP was aimed to develop static infrastructures to develop and test the latest version of !CHAOS, provide IT services for developers and its related documentation.

Task 5.1: Identify applications and user requirements, implement the IaaS infrastructure

The WP5 started in 2014 to analyse the project requirements and the best choice needed to offer !CHAOS as a Service: documentation is available on https://opensource-confluence.infn.it/. In the meanwhile, two static infrastructures accessible via VPN based on !CHAOS services on virtual machine infrastructure (one for tests, one for development) have been deployed and are currently available for accounted users. The layout of these infrastructures is shown in figure 7.
An LNF installation of an OpenStack IaaS infrastructure (open source software for creating private and public clouds) has been achieved, implementing the CDS (based on remote NoSQL Couchbase and MongoDB databases and local POSIX file systems) and an MDS java prototype. Moreover, the OpenStack infrastructure running at CNAF has been made available to test the solutions to deploy all the !CHAOS components on open source Cloud.

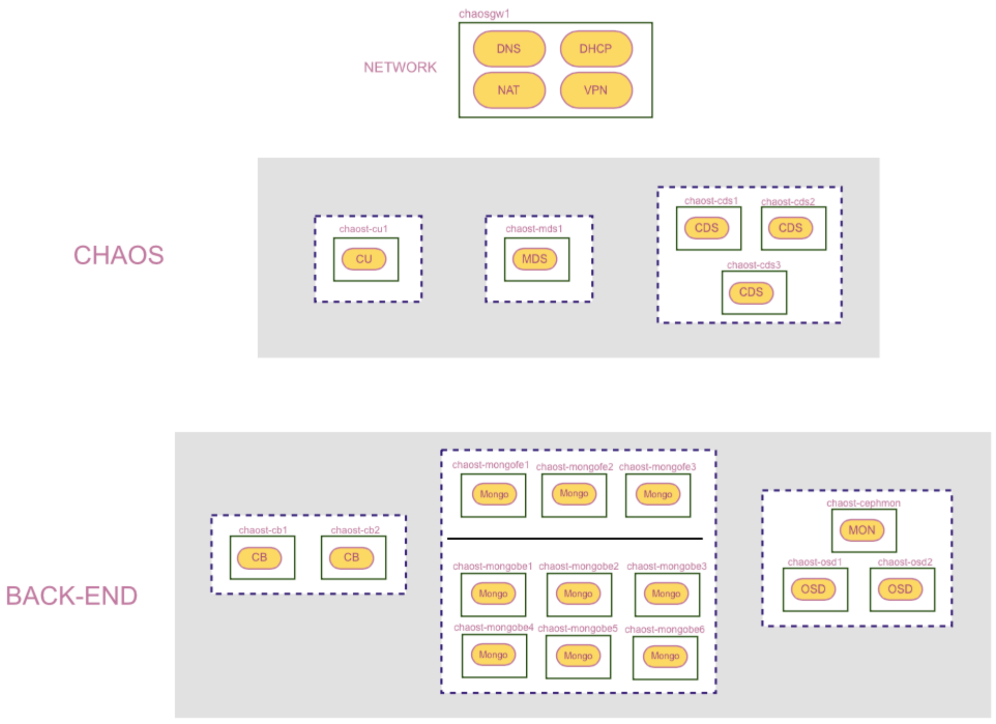


Figure 7 - !CHAOS static infrastructure layout

The focus of the second part of the task has been to investigate the technical solutions to deploy the !CHAOS components on a Cloud environment focusing on private IaaS infrastructures based on OpenStack.
The following macro-areas have been investigated:

- database services
- network access services
- shared POSIX filesystem
- automated deployment of the major !CHAOS application components (MetaData Service, Data Service)

Task 5.2: Extension of the infrastructure providing PaaS capabilities and Grid core services

The final goal of this activity has been to produce a Platform as a Service (PaaS) implementation of !CHAOS and to provide users the possibility to automatically deploy a complete !CHAOS instance on a private Cloud environment. The Figure 8 represents a simplified schema of the !CHAOS deployment on OpenStack. The common backend services are on top of the OpenStack components and can be automatically provided within the Cloud environment. The !CHAOS frontend services (CDS and MDS) exploit the virtual instances of backend components and can be run as unique or multiple instances. A VPN service is needed to enable the frontend services to communicate in a bidirectional way with the remote !CHAOS clients, such as the Control Unit (CU) and the User Interface (UI).

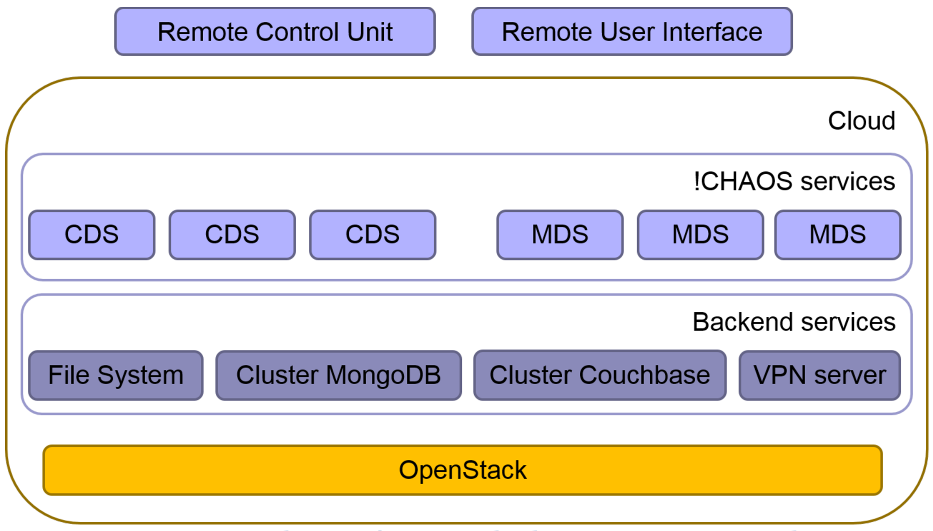


Figure 8 - Schema of !CHAOS deployment on OpenStack

To achieve the Cloud deployment of all these services, an OpenStack Heat (orchestration component) template has been developed. This is composed by different sub-templates describing the !CHAOS services installation and configuration phases.
The database services required by !CHAOS are of two kinds: a MongoDB cluster for runtime configuration management and a Couchbase cluster for live data caching. The MongoDB cluster is composed by 3 shards – each implemented as a 3-way replica set. The current Heat sub-templates for MongoDB have been derived from an open source implementation by Rackspace and are capable of performing cluster bring-up in a completely unattended way.
The Couchbase cluster sub-template has been partially derived from the MongoDB template, but the software configuration has entirely been developed in-house. The setup is again fully unattended and can handle horizontal scaling of the Couchbase cluster via HTTP POST requests to an endpoint provided by the OpenStack orchestration service, without the need for human intervention on the Couchbase cluster itself.
The shared POSIX filesystem is a virtualized Ceph cluster. It provides a 3-way replicated POSIX-like filesystem to !CHAOS services for archival. The same cluster may also act as an object storage if so configured. The setup is fully unattended and the available quota on the target cloud only limits the size of the file system.
Network access services have been implemented as a virtualized OpenVPN server and a Heat template, which configures the OpenStack component to manage virtual network services (Neutron) to route traffic to/from the VPN. Additional security can be guaranteed by Neutron services (like Firewall-as-a-Service) if supported by the targeted cloud.The !CHAOS application components (CDS and MDS) have been deployed by using a specialized template, which automatically populates the configuration files with the major backend service addresses. Also, the shared POSIX file system has automatically been mounted on all Data Service nodes. The deployment has required a heat-software config-enabled Ubuntu 14.04 image with the !CHAOS application archive injected via guestfish.
The developed Heat template (with sub-templates) has been successfully tested on the CNAF OpenStack infrastructure.

Task 5.3: Web portal and Access Policies

In order to simplify the usage of the specialized Heat template, a prototype of web interface has been developed in PHP language, exploiting the Heat APIs. Through this interface the user can instantiate a complete !CHAOS infrastructure, choosing the filesystem size and the number of MongoDB shards and Couchbase instances. As soon as the infrastructure is running, the web interface offers the possibility to scale the infrastructure, adding an instance to the Couchbase cluster.
Access to the orchestration service (both through the web interface or the OpenStack command line utilities) is reserved to people allowed to use the specific OpenStack Cloud used as IaaS to deploy the !CHAOS framework. For authentication is used the standard OpenStack method (username/password).

Functionalities	Milestones released with the alpha version at 31 July 2015	Milestones released with the beta version the 31 December 2015
Shared POSIX Filesystem	Done	Done
Couchbase	Done	Done
MongoDB	Done	Done
VPN Services	Done	Done
Resource scaling	No	Done
Unified orchestration of all backend/frontend services	No	Done

Table 8 – Task 5.2 and 5.3 Functionalities and/or product released and expected

During the project, some tools have been implemented for the software development, publication and documentation. The Atlassian suite has been tested by providing a tool for team developing and documentation. Then, the four suite packages (Confluence, JIRA, Agile and Stash) have been installed at CNAF and integrated with other national services, like continuous integration. A repository of !CHAOS latest release has also been implemented and is currently providing the software for distribution of different HW/OS targets. A work has been carried on continuous integration (entrusted to JIRA and Stash) and automatic building of the !CHAOS work, delegated to Jenkins.
Finally, the LNF team has collaborated to the development of a multisite Cloud infrastructure for distributing !CHAOS at regional level. The main objective is to achieve a distributed data centre which in it's possible deploy complex services including openstack in high availability zone. This infrastructure will ensure reliability and availability of !CHAOS services, or other, deploy inside. The project is called RMLAB, is a collaboration from INFN - LNF, TIER 2 – LNF, INFN -ROMA2 and INFN - ROMA3.

Acknowledgments

We would like to strongly thank Speranza Falciano who followed and supervised the !CHAOS project since the beginning. We also warmly acknowledge Umberto Dosselli, Gaetano Maron and Giacomo Cuttone for the support provided.