

ALICE O² CWG10 - Control, Configuration and Monitoring Software Requirements Specifications

alice-o2-cwg10@cern.ch

v0.2 - 22/01/2014

1 Introduction

1.1 Purpose

The ALICE O² system will be designed for the upgrade of the ALICE experiment during the LHC Long Shutdown 2. This document describes the requirements for the Control, Configuration and Monitoring functions of the ALICE O² system.

This document is targeted at the members of the ALICE Collaboration, in particular those participating in the O² Computing Working Groups.

1.2 Definitions, Acronyms, and Abbreviations

1.2.1 Acronyms

- CCM** Control, Configuration and Monitoring
- DIM** Distributed Information Management System
- DIP** Data Interchange Protocol
- SMI** State Management Interface

1.2.2 Definitions

Main Process: O² process controlled, configured and monitored by the O² CCM systems. May launch many *Worker Processes* or threads and act as supervisor. Typically long-lived.

Worker Process: O² process, launched by a *Main Process*, not directly controlled, configured and monitored by the O² CCM systems. Might be short-lived.

Service Process: non O² process providing service(s) used by O² components. Controlled, configured and monitored by the O² CCM systems. Examples include database daemons, web servers and DNS servers.

Thread: launched by a *Main Process* or *Worker Process*, not controlled, configured and monitored by the O² CCM systems.

1.3 Scope

The CCM systems will implement the control, configuration and monitoring features of the future ALICE O² system described in the Lol [1], the *Panel Report* [2] and the CWG1 SRS [3].

The CCM systems will primarily focus on *Main Processes* and *Service Processes*.

Given the potentially very large number of *Worker Processes* (hundreds of thousands), it is assumed that the CCM systems cannot handle them directly. Nevertheless, specialized components might provide basic CCM functionalities for this type of processes.

Threads are considered to be out of scope and will need to be handled by the parent processes.

1.4 Overview

This document follows the usual Software Requirements Specifications structure as recommended in [4]. Section 1 provides introductory information and references used in this document. Section 2 provides an overview of the different systems covered by this document. Section 3 details the specific requirements of the different systems.

1.5 History

This section will keep track of the released document versions.

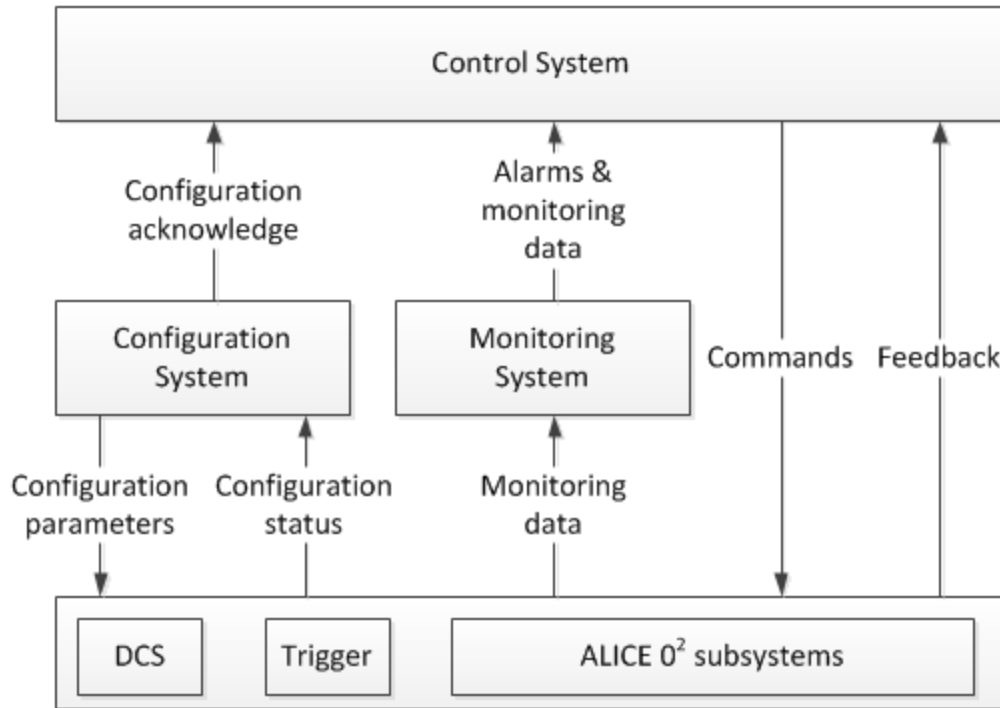
Version	Date	Comment
0.1	11/12/2013	Initial document release
0.2	22/01/2014	Changes after CWG10 review meeting on 22/01/2014

2 Overall description

2.1 System perspective

The CCM components of ALICE O² system will act as a tightly-coupled entity with a role in supporting users in day-to-day operations. The Control system will be responsible for orchestrating all the components of the ALICE O² system, based on state information as well as monitoring data. The Configuration system will ensure that both the application and the environment parameters are properly set up. Finally, the Monitoring system will receive the

status of different subsystems with the aim of determining unusual patterns and raising alarms. The CCM systems will also need to interface with other ALICE subsystems such as the DCS and the Trigger.



2.2 System functions

In this section, we describe in more detail the main functions of the Control, Configuration and Monitoring components of ALICE O² system. Briefly, the Control system is responsible with process management and management/automation of tasks, the Configuration system propagates configuration parameters to all concerned ALICE O² subsystems, while the Monitoring system collects status information from the underlying systems and raises alarms whenever abnormal conditions are detected.

2.2.1 Control

2.2.1.1 - Process Management

- Scope
 - *Main Processes and Service Processes.*
- Examples
 - Readout processes (part of O²).
 - Database server, ssh daemon (needed by O²).
- Functions

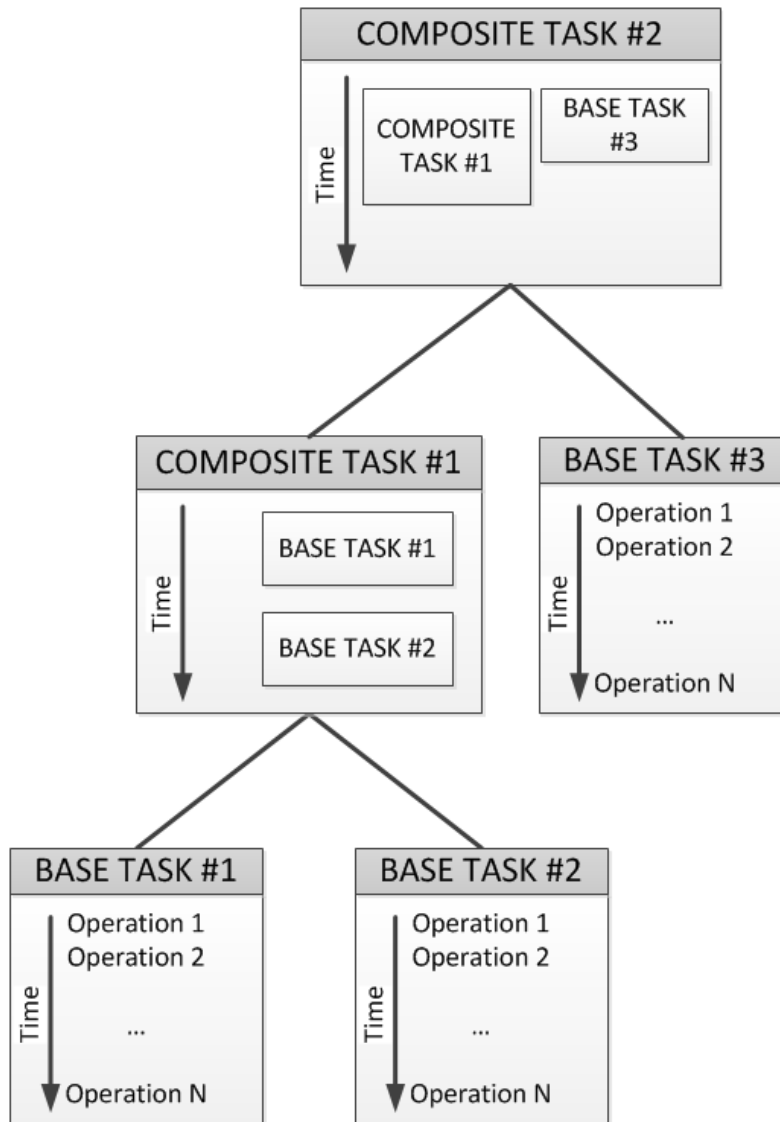
- Start, stop and send commands to *Main Processes* and *Service Processes*.

2.2.1.2 - Worker Process Management

- Scope
 - *Worker Processes*.
- Examples
 - Data Reduction or Reconstruction instances.
- Functions
 - Provide mechanism that allows *Main Processes* to control (start, stop and send commands to) *Worker Processes*.

2.2.1.3 - Task Management

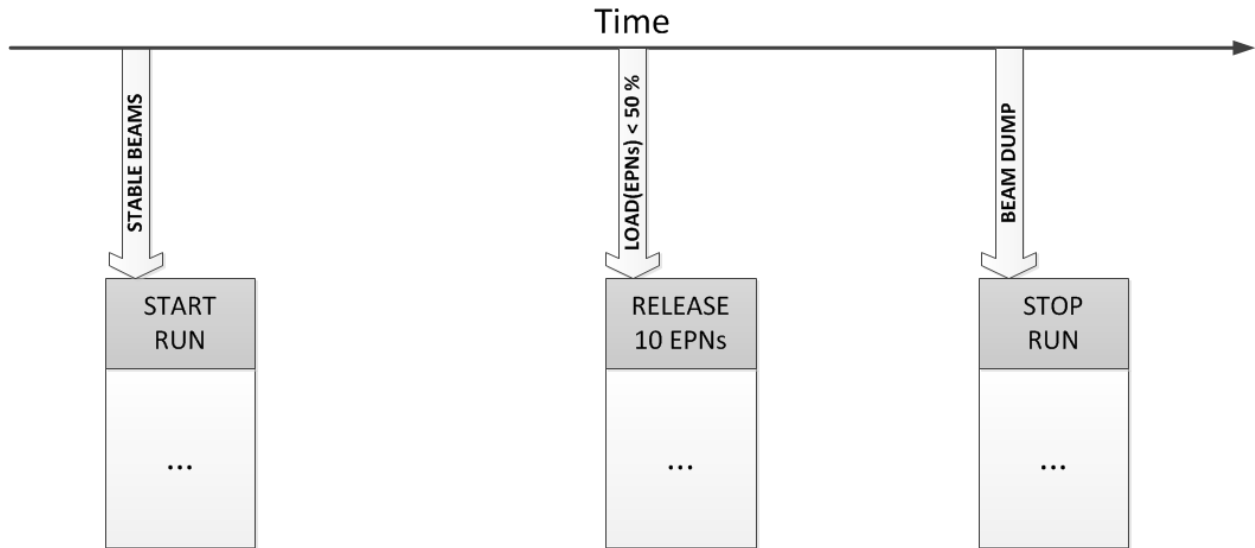
- Definitions
 - Base Task: a sequence of operations required to execute a specific functionality and executed by a single *Main Process*.
 - Composite Task: a group of Base Tasks and/or Composite Tasks that can be executed by a single or by multiple *Main Processes* in sequence or concurrently.
- Scope
 - All Tasks needed for the operations of the O² system.
- Examples
 - Base Task: start readout *Main Process* on a single FLP.
 - Composite Task: start a physics data taking session.
 - Composite Task: configure all FLPs.
- Functions
 - Ensure that all tasks are executed.
 - Ensure that tasks are executed in the correct order.
 - Ensure that task execution failure is properly handled.



2.2.1.4 - Automation

- Definitions
 - Internal Event: asynchronous action or occurrence generated by a component of the O² system.
 - External Event: asynchronous action or occurrence generated by an external system (e.g. LHC, DCS, detector).
- Scope
 - All Events relevant to the operations of the O² system.
- Examples
 - Internal Event: shifter action, dead *Main Process*, DQM bad quality data flag.
 - External Event: stable beams declaration, beam dump, detector not ready.

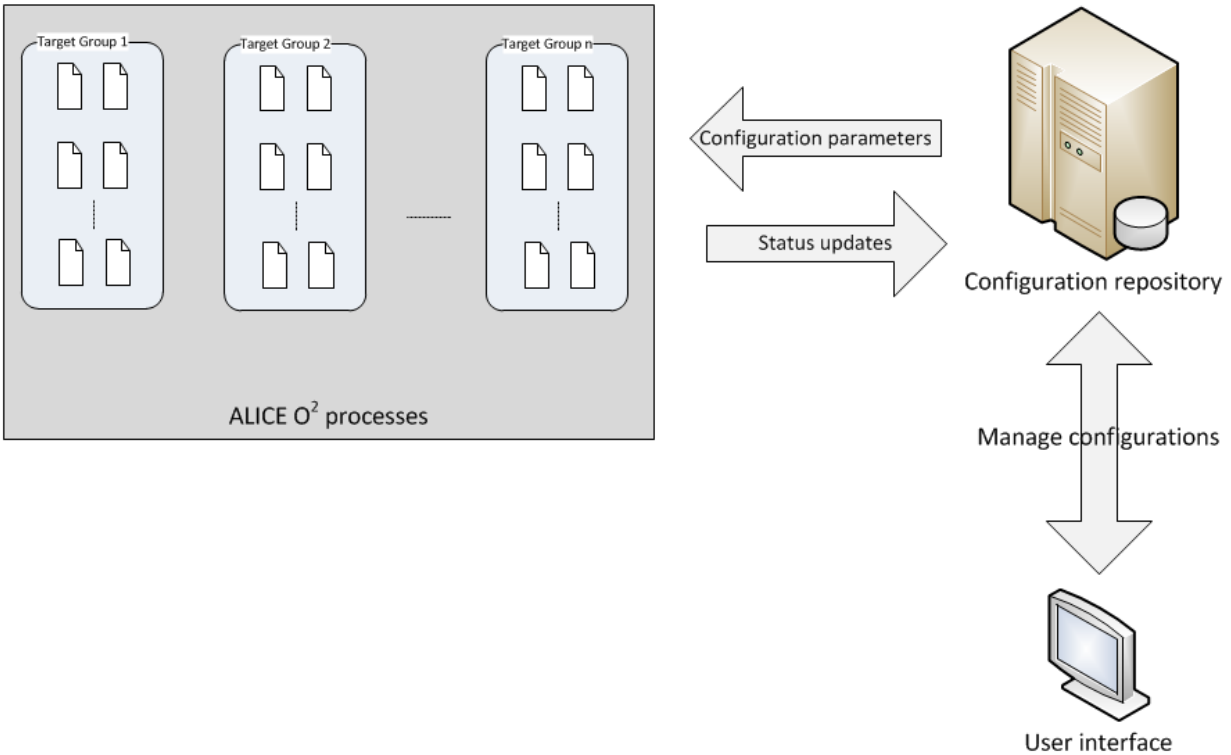
- Functions
 - Automatically execute tasks based on internal or external events.
 - Automatically recover from errors.



2.2.2 Configuration

2.2.2.1 - Configuration Distribution

- Definitions
 - Configuration: set of parameters required by one or more processes at runtime.
- Scope
 - *All Main Processes and Service Processes.*
- Examples
 - Maximum event size, source of input data (e.g. online data or file), number of *Worker Processes*/threads to launch.
- Functions
 - Automatically distribute configuration parameters from one central repository to all concerned nodes and processes in the O² system.



2.2.2.2 - Dynamic Process Reconfiguration

- Definitions
 - Dynamic reconfiguration: the ability to load new values for the configuration parameters without restarting the processes.
- Scope
 - *Main Processes*.
- Examples
 - Log level, number of *Worker Processes*/threads to launch.
- Functions
 - Automatically distribute new values for the configuration parameters to all concerned processes in the O² system without the need to restart them.

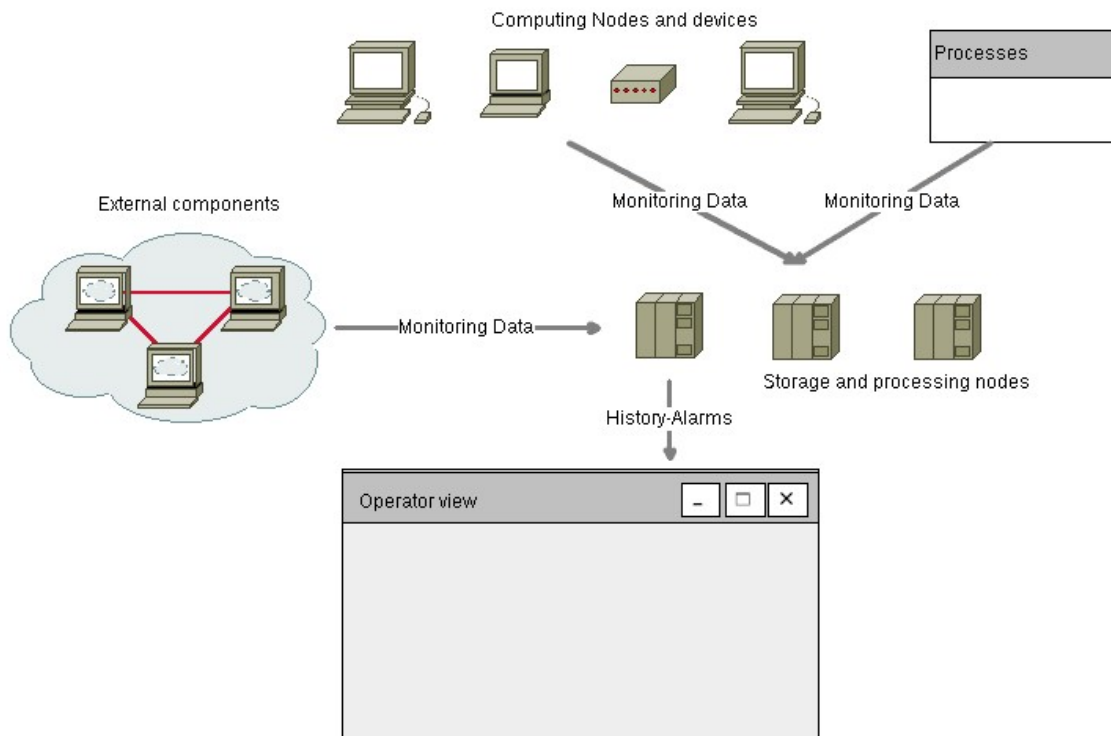
2.2.3 Monitoring

2.2.2.1 - Data Collection and Archival

- Definitions
 - Monitoring data: set of measurements that reflect the status of O² system components.
- Scope
 - All relevant measurements of infrastructure, hosts, *Main Processes* and *Service*

Processes.

- Examples
 - CPU load average, process memory usage, event rate.
- Functions
 - Collect monitoring data from both internal and external components.
 - Provide access to current and historical record of monitoring data.
 - Distribute monitoring data to consumers. For certain critical data, ensure delivery.



2.2.2.2 - Alarms and action triggering

- Definitions
 - Alarm: event triggered by conditions outside the normal operating range of the monitored component.
 - Monitoring action: response of the monitoring system to alarms.
- Scope
 - All parameters gathered in the monitoring framework that indicate a potential problem and require an intervention (automatic or human).
- Examples
 - Alarm: high CPU temperature.
 - Monitoring action: shutdown machine on very high CPU temperature.
- Functions
 - Raise alarms and trigger actions based on collected monitoring data.

2.3 User characteristics

Below we define the different user roles that will interact with the system. Each role has a list of allowed actions. The same person can have different roles. Additionally, a specific role can have different scopes which translate into different actions (e.g. a DAQ system expert is different than an Offline system expert).

2.3.1 - General Public

The General Public might be interested in the status of the experiment's operations. Additionally, the general public might find uses for the published data not foreseen inside the collaboration. As such the system should allow:

- Restricted read access to data and/or status displays.

2.3.2 - ALICE Collaboration member

The ALICE Collaboration members require no specific knowledge, since their access is read-only. They should nevertheless have unrestricted read access to the CCM systems, thus allowing ALICE collaborators spread around the world to use the published information freely. The access permissions include the ones for the General Public adding:

- Read access to bookkeeping messages.
- Unrestricted read access to data and status displays.

2.3.3 - Shifter

The shifters are ALICE Collaboration members with a general understanding of the system, following a brief training. Their knowledge is specialized in a certain area, depending on the subsystem they are operating (e.g. DCS, DAQ, etc). As a prerequisite, they should at least have a minimal set of competences in using the control room workstations.

- Operates the experiment.
- Applies configurations.
- Acknowledges alarms, triggers corrective actions.
- Writes bookkeeping messages.

2.3.4 - System Expert

The System Experts are ALICE Collaboration members with a deep knowledge of their subsystem(s). They should be able to debug, recover and operate their subsystem both onsite and remotely.

- Defines and applies configurations.
- Acknowledges alarms, triggers corrective actions.
- Writes bookkeeping messages.

2.3.4 - Administrator

The Administrators are ALICE Collaboration members in charge of supporting and maintaining the Control, Configuration and Monitoring systems described in this document. Therefore, they should have a deep knowledge of these systems.

- Defines monitoring metrics.
- Defines alarms thresholds.
- Defines recovery actions for whole environment.
- Manages access control.
- Writes bookkeeping messages.

2.4 Constraints

2.4.1 Data Links

The number of data links is defined in the Lol [1] as **2500**.

2.4.2 Computing Nodes

The number of FLPs and EPNs is defined in the Lol [1]. Other dedicated nodes will also be necessary (e.g. QA/DQM farm), but their number remains unknown for the time being. Therefore, certain values have to be estimated:

Function	# of Nodes	Source
FLPs	250	Lol
EPNs	1250	Lol
DB servers	5	Estimated by CWG10
Control servers	6	Estimated by CWG10
Configuration servers	6	Estimated by CWG10
Monitoring servers	6	Estimated by CWG10
QA/DQM servers	30	Estimated by CWG10
Calibration servers	30	Estimated by CWG10
Storage servers	10	Estimated by CWG10
Network servers	5	Estimated by CWG10
Operator terminals	25	Estimated by CWG10
Total	1623	

2.4.3 Processes

The number of processes that will run in the O² facility is one of the biggest constraints for the CCM systems, since it determines the number of actions the control will execute, the number of configurations to be distributed and the monitoring data volume to be handled. At this time, it is difficult to estimate an exact number since it will greatly depend on the computing model adopted for the O² framework by CWG1, especially for the functional blocks that will run on the EPNs (part of the Data Reduction and the Reconstruction). Currently, there are 2 models being evaluated: multi-threading and multi-processing.

2.4.3.1 Main Processes

The number of *Main Processes* is estimated to be between **7 K** and **70 K**. These values are calculated based on input given by other CWGs and, whenever input was unavailable, internal CWG10 reasoning. The wide range is a consequence of the number of *Main Processes* estimated to run on the EPNs for the Data Reduction and Reconstruction functional blocks. The lower limit assumes 1 *Main Process* per EPN while the upper limit assumes 20 *Main Processes* per EPN for both functional blocks. More details on the *Main Processes* estimates can be found in CWG10 “Number of Processes Estimate” [5].

2.4.3.2 Worker Processes

The number of *Worker Processes* is estimated to be either **0** or up to **280 K**. These values are highly speculative since the usage of *Worker Processes* will depend on the chosen computing model and its actual implementation. The lower limit comes from a multi-threading model while the upper limit comes from a multi-processing model with 1 *Worker Process* per CPU core in the EPNs. More details on the *Worker Processes* estimates can be found in CWG10 “Number of Processes Estimate” [5].

2.4.5 Interfaces to external systems

The O² system will need to interface with existing external systems that might change until Run 3. Based on the existing systems, the following constraints have been identified:

- **Interface to DCS:** needed to send commands, read current state and exchange information with the DCS system and the ALICE detectors. Currently done via SMI and DIM.
- **Interface to Trigger:** needed to send commands, read current state and exchange information with the Trigger system. Currently done via SMI and DIM.
- **Interface to LHC:** needed to have access to the LHC operational status (e.g. STABLE BEAMS) and different LHC operational and configuration values. Currently it is done via DIP.

2.4.6 Control commands and state changes

Considering that the *Main Processes* will receive commands from the Control System and that the command's execution will be represented by states, we can estimate the number of peak commands and state transitions that the Control system will need to handle. Based on past experience, the most *active* periods are Start of Run and End of Run, when many processes from different systems need to be either launched, initialized and configured or closed and stopped.

Therefore, if we consider the number of *Main Processes* range provided in 2.4.3.1 and assuming 2 commands (e.g. CONFIGURE and START) and 6 state transitions (e.g. INITIALIZING, INITIALIZED, CONFIGURING, CONFIGURED, STARTING, RUNNING) per *Main Process* during Start of Run, the expected peak values are:

- **Commands**
 - $SOR_{peak} = N_{MainProcesses} * 2$
 - between **14 K** and **140 K**
- **State Transitions**
 - $SOR_{peak} = N_{MainProcesses} * 6$
 - between **42 K** and **420 K**

3 Specific requirements

3.1 Control

3.1.1 Functional Requirements

- Process Management
 - Start and stop all *Main Processes* and required *Service Processes*.
 - Send commands to *Main Processes* and receive acknowledgments.
 - Handle *Main Processes* commands execution failure.
- Worker Process Management
 - Enable a *Main Process* to start at least 200 *Worker Processes* (1 *Worker Process* per EPN CPU core).
 - Enable a *Main Process* to stop any/all *Worker Process* it controls.
 - Enable a *Main Process* to send commands to and receive acknowledgments from all *Worker Processes* it controls.
 - Enable handling of failures in starting and stopping *Worker Processes*.
 - Enable handling of command execution failures in *Worker Processes*.
- Task Management
 - Start and stop a data taking run with an arbitrary number of detectors.

- Start and stop calibration tasks.
- Start and stop QA/DQM tasks.
- Start and stop data migration tasks.
- Execute recovery actions during or outside data taking runs, including the execution of detector control commands via the readout link (equivalent to Run 2 PAR).
- Allow concurrent usage of exclusive data taking partitions (i.e. current ECS partitions).
- In parallel with data taking operations, start and stop simulation jobs on unused resources (e.g. EPNs).
- In parallel with data taking operations, start and stop re-reconstruction jobs on unused resources (e.g. EPNs).
- During data taking operations, include or exclude resources (e.g. EPNs) without stopping the run.
- Automation
 - Automatically stop ongoing tasks and prepare the O² system for data acquisition during one of the pre STABLE BEAMS phases of the LHC.
 - Automatically start data taking after stable beams declaration.
 - Automatically stop data taking on beam dump or beam dump warning.
 - Automatically perform recovery actions on detector or subsystem request (e.g. DQM).
 - Automatically reassign resources (e.g. EPNs) between data taking and other tasks based on system load.
 - Automatically execute calibration tasks based on QA results.

3.1.2 Performance Requirements

- Time to “warmup” (cold boot) the O² system for data taking activities should be less than 5 minutes.
- Time to start a data taking run (including reconfiguration) after “warmup” should be less than 20s^{1 2}.
 - System should be able to handle at least 140 K commands in a fraction of the 20s.
 - System should be able to handle at least 400 K state transitions in a fraction of

¹ The 20s and 10s values for SOR and EOR were extracted from the analysis performed after the 2013 pA run and presented in the “[Overview of inefficiencies and strategies for improvement](#)” session (slide 38) of the “[Workshop on Run 1 conclusions and Run 2 outlook](#)” workshop of 11/03/2013. The aim is to, given a similar number of runs as experienced during the 2013 pA run, ensure that the total downtime due to SOR and EOR sequences is not greater than 1% of the total stable beams time.

² The SOR and EOR sequences mentioned here include the actions performed by other ALICE subsystem which do not belong to the O² project (DCS, Trigger, detectors). Therefore, any review of the time budgets must take into account that certain actions might have a time compression limit, specially those involving specialized hardware.

the 20s.

- Time to stop a data taking run should be less than $10s^{12}$.
- Automatic inclusion or exclusion of EPNs during a run should not require an interruption of the data acquisition.

3.1.3 Interface Requirements

- Input Requirements
 - Status of all *Main Processes* and *Service Processes*.
 - DCS detector status.
 - Central DCS status.
 - Trigger system status.
 - LHC status.
 - Status of all tasks being executed.
 - Monitoring values.
 - User actions via GUI.
 - Action execution requests from detectors or subsystems.
 - Commands acknowledgments from *Main Processes*, DCS, detectors and Trigger system.
- Output Requirements
 - Status of executed tasks.
 - Commands to be executed by *Main Processes*.
 - Commands to be executed by DCS, detectors and Trigger system.

3.1.4 Operational Requirements

- Execute main tasks with a single human action (e.g.start/stop run, send detector control command).
- Execute any task automatically based on external conditions (e.g. start run on stable beams declaration, stop run on beam dump).
- Feedback on tasks being executed.
- Visualize hierarchical structure of tasks/processes.
- Create and edit tasks (including execution order) via configuration tools.
- Multiple operators can control exclusive resources (e.g. data taking partitions, simulation jobs).

3.1.5 Resource and Reliability Requirements

- The control “agents” should have a negligible impact (CPU, memory, etc) on the controlled nodes (FLPs, EPNs, etc).
- The crash of a single Control server should not affect the Control system functions.

- The crash of a Control process should not affect the Control system functions.

3.2 Configuration

3.2.1 Functional Requirements

- Process configuration
 - Configure (dynamically) *Main Processes* and *Service Processes*.
- Machine installation
 - Network boot.
 - Automatic installation of base OS and starting the configuration service.
- Package management
 - Add/remove/update packages.

3.2.2 Performance Requirements

- A Start of Run O² reconfiguration of *Main Processes* should take less than 10s (warm reconfiguration).
- A full O² reconfiguration of *Main Processes* (does not include software distribution and installation) should take less than 60s (cold reconfiguration).
- The system should be able to handle a volume of configuration data of approximately 700 MB: $\sim 100\text{B/param} * \sim 100 \text{ params/process} * 70 \text{ K processes}$ (as shown in CWG10 “Number of Processes Estimate” [5]).
- The system should be able to manage simultaneously $\sim 70 \text{ K}$ configuration responses (received from the configured processes).

3.2.3 Interface Requirements

- Input Requirements
 - Define/save/load/propagate configurations via GUI.
 - Validate input parameters.
- Output Requirements
 - Status of deployed configurations (failed, pending or finished).
 - History of deployed configurations.
 - The actual configuration parameters.

3.2.4 Operational Requirements

- Possibility to define target groups of processes.
- Apply specific configurations depending on the target group.
- Possibility to create and edit configuration tasks (including execution order).

- Possibility to assign priority levels to defined configurations.
- Block concurrent attempts to configure the same process.
- Compare configurations.
- Automatically propagate changes made on the central configuration repository.
- Scheduled/triggered distribution of configuration parameters.
- Allow parallel distribution of configuration changes.
- Selective configuration of processes.
- Keep a history of subsequent configurations of a process or set of processes.
- Add configuration parameters for new components.

3.2.5 Resource and Reliability Requirements

- The local client managing the configuration should have negligible CPU, memory and network footprint.
- The crash of a single Configuration server should not affect the Configuration system functions.
- The crash of a Configuration process should not affect the Configuration system functions.

3.3 Monitoring

3.3.1 Functional Requirements

The monitoring system will focus on the infrastructure and the *Main Processes* while allowing explicit application parameters to be sent from any entity in the system, or other external components.

- Data Collection and Archival
 - Decentralized operation: Monitoring agents should be independent of external components especially data consumers.
 - Scalability: the monitoring system should allow to add new *Main Processes* and *Service Processes*, new nodes, devices, parameters for existing entities and define new alarms.
 - New devices/nodes/processes should be automatically discovered.
 - A heartbeat from all *Main processes* is important to realise when a component is not working any more and trigger alarms.
 - Possibility to push monitoring data into the system.
 - The system should also provide a mechanism to monitor *Service Processes* or components (e.g. hardware) that cannot be directly instrumented.
 - Interested parties should be able to filter the monitoring data stream based on expressions, for both historical data and persistent subscriptions for newly generated values.
 - The monitoring framework should allow new parameters to be monitored and at

- arbitrary collecting frequencies.
- The monitoring system should allow the administrating users to interact with the system without having to care about where the monitoring services are running and where the data is stored.
- The monitoring system should allow the interconnected systems (such as the Control system) to easily readout monitoring values.
- Action Triggering
 - Certain events might be critical for data taking (e.g. readout process dies). For these events, the communication between the monitoring system and the consumer of such events (e.g. Control system) must include acknowledgement.
 - The system should be able to deliver specific alarms to required users.
 - The system should inform other subsystems about events related to them.
 - The system should use flexible means to distribute alarms (GUI, email, SMS).

3.3.2 Performance Requirements

- The system should be able to collect, transport and eventually store high frequency monitoring data.
- Below is a list of metrics expected for the different categories of applications:
 - Host monitoring: CPU (10 metrics); networking (4 metrics / interface); memory (10 metrics); processes status (5 metrics); sockets status (10 metrics); disk status (10 metrics / device).
 - Process monitoring (from the system point of view: cpu, memory profile, handles): 10 metrics / process.
 - Explicit process data: heartbeat (1 / process); state (1 / process); explicit, application-specific monitoring data (5 / process).
 - Infrastructure monitoring: 4 metrics / port.
- Based on current use cases, the expected resolution for the monitoring data is ~30s for host and infrastructure monitoring, ~15s for general process monitoring, ~1s for the explicit process data
- As such the base monitoring frequencies are:
 - $10+4*4+10+5+10+10*8 + 4 / 30s = 4.5 \text{ Hz} / \text{host}$
 - $10 / 15s + 7 / 1s = 7.67 \text{ Hz} / \text{Main Process}$
- Taking into account the estimated number of hosts (section 2.4.2) with 4 network interfaces and 8 local disks and of *Main Processes* (section 2.4.3.1), the system should be able to handle between:
 - $1623 * 4.5 \text{ Hz} + 7 \text{ K} * 7.67 \text{ Hz} \approx 60 \text{ kHz}$
 - $1623 * 4.5 \text{ Hz} + 70 \text{ K} * 7.67 \text{ Hz} \approx 544 \text{ kHz}$
- Considering the criticality of some of the data, the system should be able to transport this amount of data in ~real time to the consumers
- In terms of data volume, currently ~100 bytes are used to serialize one parameter for transport or storage (unique series identifier, timestamp, value, protocol overhead /

indexes). Correlated with the above, the system should be able to transport up to 52 MB/s of monitoring data, which should not be a problem for the infrastructure.

- The high frequency will impose however restrictions on how much data can be archived.

3.3.3 Interface Requirements

- **Input Requirements**

- Systems to be monitored.
- Processes to be monitored.
- Configuration of required measures.
- Configuration of parameters for measures (interval, publish rate).
- Start command.
- Stop command.

- **Output Requirements**

- Measurements.
- Alarms.
- Historic data.
- Trends.

3.3.4 Operational Requirements

- Provide a simple interface to configure and visualize monitoring data.
- Allow visualization of historic records and trends.
- Allow configuration of rules and actions.
- Provide API to interact with Monitoring system (read and push values, etc.) via scriptable language.
- Possibility to create and edit metrics.
- Allow addition of new monitored components and new data consumer.

3.3.5 Resource and Reliability Requirements

- The monitoring “agents” should have a negligible impact (CPU, memory, etc) on the controlled nodes (FLPs, EPNs, etc).
- The crash of a single Monitoring server should not affect the Monitoring system functions.
- The crash of a Monitoring process should not affect the Monitoring system functions.

Appendix

A. References

[1] Letter of Intent for the Upgrade of the ALICE Experiment

<http://cds.cern.ch/record/1475243?ln=en>

[2] ALICE Computing software framework for LS2 Upgrade

<https://edms.cern.ch/document/1272165/1>

[3] ALICE O² System Requirements Specifications

<https://twiki.cern.ch/twiki/pub/ALICE/Requirements/ALICEO2SystemRequirementsSpecifications.pdf>

[4] *IEEE Recommended Practice for Software Requirements Specifications* – 06/1998, IEEE Std 830-1998

<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=720574>

[5] CWG10 Number of Processes Estimate

<https://twiki.cern.ch/twiki/pub/ALICE/Cwg10/NumberOfProcessesEstimate.pdf>