CERN
CH-1211 Geneva 23
Switzerland

the
Large
Hadron
Collider
project

LHC Project Document No.
LHC-FS-TIM-0001.00 rev 0.02

CERN Div./Group or Supplier/Contractor Document No.
AB-OP

EDMS Document No.
123456

Date: 2006-08-01

Functional specification

LHC SEQUENCER – OPERATIONAL FUNCTIONALITY, INTERFACES AND REQUIREMENTS

Abstract

This document describes the high level LHC operations' functionality, requirements and interfaces for the LHC Sequencer.

Prepared by :	***Checked by :***	***Approved by:***
Mike Reyes		

History of Changes

Rev. No.	***Date***	***Pages***	***Description of Changes***

0.1	21 April 2006	First draft
0.2	31 July 2006	Following discussions and revisions by authors.
0.3	25 Sep 06	Revised by Reyes. Includes comments and ideas from Jorg, Claudio and Karel.
0.4	7 Oct 06	2nd pass by Mike

Table of Contents

1. INTRODUCTION

LHC operations will need to drive the LHC accelerator through a variety of cycles. This will involve executing a large number of tightly coupled tasks (activities) in strict order. Many of these tasks must complete successfully for the sequence to continue. It is envisaged that the operational cycle will be sub-divided into modes, with a well-defined set of tasks associated with a given mode.

The tasks associated with a given mode have to be performed successfully to allow the LHC machine to transition from one mode to another. Based on this paradigm the sequencer should provide a tool to allow reproducible, and reliable beam based LHC operation.

Among the tasks to be managed are included:

- equipment state control
- loading of equipment settings
- timing event request (start ramp etc.)
- instrumentation configuration
- beam measurements
- coordinated data acquisition and saving.

The sequencer should also be able to monitor external signals and react accordingly. For example, if the Beam Presence Flag is false it would not attempt to execute the intermediate beam injection sequence.

2. SCOPE

It is envisaged that the sequencer being capable of driving:

- Nominal cycle with beam including the injection process
- Into and out of access
- Switch On
- Machine Development

3. OPERATIONAL CONTEXT

3.1 SUBSYSTEMS

The beam related hardware is divided into a number of well defined sub-systems (Power converters, RF, collimators…). These sub-systems, although essentially independent, are coupled during the LHC cycle in that they sometimes need to do things at precisely the same time e.g. during the ramp. This coordination is achieved using the timing system. Thus the actions performed by the sequencer are largely sub-system specific.

The list of "possible" sub-systems the Sequencer will have to interface is the following:

Sub-system	Sub-system details
Power Converters	1720 units
Magnets	- Dipoles - Quadrupoles

	- Sextupoles - Octupoles - Correctors All associated settings – driven by power converters.
Injection septa and kickers	
RF (Ring1 RF and Ring2 RF)	- Main 400 MHz (ACS) - Staged 200 MHz Capture (ACN) - Transverse Dumping & Feedback System (ADT) - Low-Level RF
Collimators	- Primary collimators - Secondary collimators - Tertiary collimators (experimental insertions) - Scrapers and special collimators for injection protection - Collision debris collimators ~ 150 collimators
Beam Instrumentation	- Beam Position Monitors (BPM): 2x1032 - Beam Loss Monitors (BLM): > 3000 monitors - Beam Current Transformers (FBCT & BCTDC) - Transverse Profile Monitors (single pass, few pass-matching, synchrotron light and rest gas monitors and wire scanners) - Luminosity monitors - Tune, chromaticity and betatron coupling monitors - Aperture and non-linear monitors - Dedicated BPM - High Frequency pick-up - Schottky System
Beam Dump (one per Ring)	- MKDs - MSDs - Dilution kickers - TDEs - TCDS (static) & TCDQ
Beam Flags	Safe Beam Flag – monitor and condition on Beam Presence Flag – monitor and condition on Safe Beam Parameters
Beam Interlock System	Status of BIC
Machine Protection System	PIC
Quench Protection System	QPS states
Spectrometers Magnets	Alice and LHCb
Central Beam and Cycle Manager (CBCM)	- General Machine Timing (GMT) - Beam Synchronous Timing (BST)
FIDEL	(except if FIDEL interfaces directly the Magnet Control Software)
Post-Mortem Analysis	

Access System	
DIP (Data Interchange Protocol)	

Many of these sub-systems need cycle dependent settings to be able to drive through the cycle is being played. In addition the settings may also depend on the beam intensity, on the beam energy, etc.

We could also foresee interfacing to DIP to get the background, among other information, from the experiments and if a given threshold (that should be configurable) is reached, then a pop-up window from the Sequencer GUI could appear for warning the operator. Then the operator will decide if to change the "Stable Beam State" to "Unstable Beam", for example. This transition could also happen asynchronously.

The Sequencer will have to inform the experiments via DIP about the current state of the machine and other information relevant for the experiments.

For example, typical tasks to bring the LHC to the pre-injection level are shown in Table 1.

Command	Sub-system	Setting
Load	All power converters	Pre-injection level
Send event	Timing	Start ramp
Set	All collimators	Parking
Set	TDI, TCDQ	Parking
Set	Kickers	Standby
Set	Beam Dumps	Active
Read/Compare	Kickers	Setting

Table 1: Tasks to bring the LHC to the pre-injection level.

3.2 STATES

For some equipment sub-systems, it is certainly useful to model their behaviour using a finite state model, and this is certainly done for, for example, in the case of the power converters. Knowledge of these state models is vital for high level functionality.

Nevertheless, beyond this, an operation state model might also be useful. For example, in the case of the collimators the state model for an individual collimator might span:

- FULLY OUT (on end stops), FULLY IN (on end stops), MOVING, FAULT, SET

However from an operational standpoint, the states might include:

- FULLY OUT
- PARKED
- COARSE
- PROTECT
- MOVING
- FAULT
- HALO REMOVAL

with a fully consistent state model describing the transition between these operational states. Each state necessarily has data associated with it.

Given the implementation of such a state model, the sequencer could simply act by initiating a state transition, and avoid having to be involved in any data handling etc.

The LHC Sequencer has to interface also the cryogenics system, the vacuum system, the Quench Protection System (QPS), Machine Protection System, interlock system and the access system. Those systems should be in the appropriate state in order the Sequencer can play a given cycle or going from one mode to another inside the cycle. The interface here is simpler because only the "current state" of the inquired system has to be known by the Sequencer.

3.3 THE LHC AS A STATE MACHINE

This is not obvious. In the LHC we have:

- A finite number of loosely coupled systems, some of which maybe accurately described by a state model.
- Some systems have a simple internal state model - collimators, kickers for example. Upon these one can presumably build operational state models.
- Other systems have a internal state model (power converters - on, armed, executing ramp, etc.) which if we thought about it could build up a operational state model (specify both internal & operational states in any diagram). The problem is that the data content of the states is fluid and critical. Is a corrector with a setting of 1.0 A in a different state from 1.1 A).. In physics self-transitions with a change of data content (i.e. a trim) are common, of course.
- The state of the whole accelerator is rather nebulous (except in special circumstances). This might be due to our inability to properly describe internal transitions. Manual Operator actions are certainly foreseeable, at least in the commissioning stage.
- We cannot pre-load all our settings: 1. the power converter FGCs do not implement this 2. we expect re-loading to be necessary because of the machine dynamics.

A strict finite state machine during the commissioning phase would certainly not seem to be possible. Whether we can work towards this in the final analysis will depend on experience.

3.4 LSA FUNCTIONALITY

In developing the sequence it should be bourn in mind that LSA provides comprehensive settings management and drive facilities. A task based interface to LSA middle-tier functionality is clearly required. LSA functionality shall be used as appropriate.

4. INJECTION

There is clear need for high level functionality to drive the injection process and to coordinate requests for beam to the injectors via the timing system.

We will need an INJECTION SEQUENCE to be configured before the injection process is started. Set-up offline, this will drive the pre-loading of equipment settings and timing. And some of the checks to be performed before injection begin. It will then be used to sequence the requested injection sequence.

No.	**Requirement**	**Priority**
IN.R0	Injection sequence should be easily configurable and reusable.	Critical
IN.R1	Should allow fully automatic invocation of given injection sequence	Expected
IN.R2	Manual Step through of injection sequence shall be possible	Critical

IN.R3	Manual abort of running injection sequence shall be possible.	Critical
IN.R4	Repeat task Repeat task after abort	Critical
IN.R5	Continue sequence after aborted injection	Critical
IN.R6	Injection sequence should be playable from the main sequencer or from a dedicated GUI	Expected [commissioning]
IN.R7	Need to monitor of injection efficiency into the LHC. The injection sequencer should accept input from these processes and halt the injection if thresholds are exceeded e.g. beam losses, beam lifetime etc.	Critical
IN.R8	Accept input from Software Interlocks and halt sequence if machine conditions go unsafe	Expected
IN.R9	Load beam intensity dependent settings to BDI as required. Check that everything that needs to be loaded is loaded before starting injection process.	Critical
IN.R10	Load batch dependent settings to RF as required. Check that everything that needs to be loaded is loaded before starting injection process.	Critical

5. TASK MANGEMENT

The sequencer shall initiate the execution of atomic tasks. These tasks might be executed by the sequencer itself or another agent. The sequencer tracks the status of an executing task.

No.	**Requirement**	**Priority**
TM.R0	The sequencer shall never initiate a task that is incompatible with the status of the beam flags, BIC, beam dump and MPS.	Critical
TM.R1	The sequencer should be able to initiate tasks in parallel. Handle multithreading/distributed processing logic	Critical
TM.R2	It should catch return code of executed tasks and react appropriately: - stop, continue, issue warning, abort, start another sub-sequence.	Critical
TM.R3	Display progress and status of executing tasks.	High (important for debugging)
TM.R4	Allow operator to abort executing task(s) safely.	High (important for debugging)
TM.R5	Allow operator to manually abort sequence safely.	High (important for debugging)
TM.R6	Allow operator to manual execute tasks. To manual repeat task.	High (important for debugging)

TM.R7	Allow operator to skip tasks safely.	High (important for debugging)
TM.R8	Support synchronous and asynchronous commands. If the command execution in a sub-system takes lot of time, allow notification from the sub-system to the sequencer of command conclusion	Critical

6. SEQUENCES

No.	Requirement	Priority
SQ.R0	Support multiple sequence definitions.	Critical
SQ.R1	Allow re-use of sub-sequences in different sequences.	Critical
SQ.R2	It should be easily configurable - add/delete/change task specification	High (important for debugging)
SQ.R3	Allow operator to manually drive sequence	High (important for debugging)
SQ.R4	Allow operator to manually drive sequence for given sub-system	High (important for debugging)
SQ.R5	Is able to accept external input from monitoring or machine protection process and modify behaviour appropriately. These external inputs shall include the Beam Presence Flag, the Safe Beam Flag, the Safe Beam Parameters, the Beam Interlocks, the beam dump status.	Critical
SQ.R6	Is able to react rapidly to external events to be more efficient. For example, if one second before injection starts a beam dump takes place, the sequencer should avoid to start the injection sequence to gain time	Critical
SQ.R7	In principle the Interlock System takes care of driving the some of the machine sub-systems into safe mode after a beam dump or quench. The Sequencer should take care of the sub-system that are not under the Interlock System supervision.	Critical
SQ.R8	Most probably, Machine Development periods will require a more flexible Sequencer giving room to the implementation of scripting language or some other technology that offers operation flexibility	High (important for debugging)

7. GUI

No.	Requirement	Priority
GU.R0	The LHC GUI should be a dynamic display in the sense that the buttons the operator can click and the information displayed will depend on which cycle is being played	High

GU.R1	For security, only the commands that can be currently issued by the operator will appear activated on the window and can be clicked with the mouse. The active commands will depend on the current Sequencer state.	High
GU.R2	The GUI should display the states of the Sequencer and should highlight the current one to be easily distinguishable from the other states	High
GU.R3	A convention should be established on the states colour code. For example: • Error or Fault states in RED • Successful states with no beam in the machine in BLUE • Successful states with beam in the machine in GREEN • Intermediate states indicating an action is being executed in YELLOW • Warning states in ORANGE	High
GU.R4	Allow operator to switch from automatic sequence driving to manually sequence driving	Critical
GU.R5	Allow operator to break in a given task	Critical
GU.R6	Allow operator to mask errors safely	Critical
GU.R7	Allow operator to skip tasks safely	Critical
GU.R8	Graphical tool to add or take out states and corresponding actions in the proposed state machine	Low
GU.R9	Pop-up messages of warning to remind the Operator if a given task or action has been done before continuing the sequence execution	
GU.R10	Different users levels, e.g. a superuser have access to more flexibility in the GUI than a normal user	

The following pictures show a proposal for a LHC Sequencer GUI.

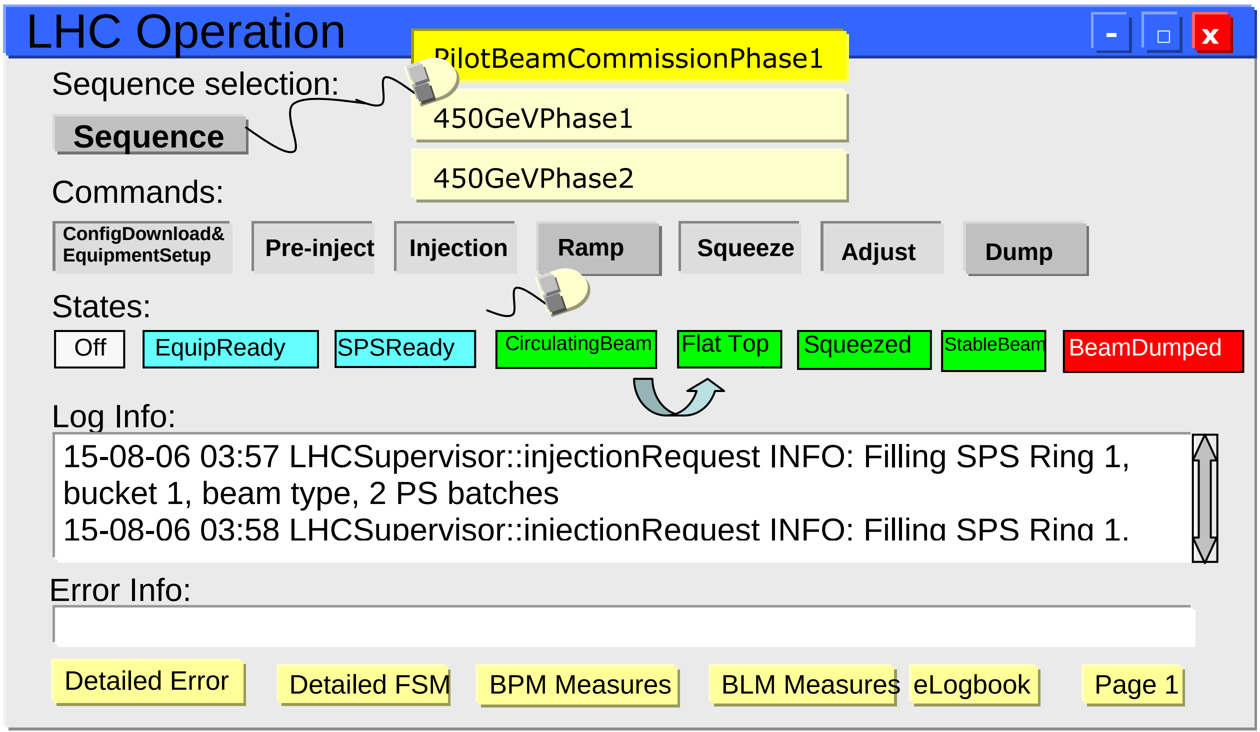

Figure 1: Example of the LHC GUI when the Sequence "PilotBeamCommissionPhase1" has been selected. The Commands and States will appear on the window when the Operator chooses a Sequence. As can be seen, only the commands that can be issued, taking into account the current state of the Sequencer, appear activated and can be clicked. The current state "CirculatingBeam" appears contrasted in colour with respect to the other states.

Figure 2 illustrates how the Commands and States that are displayed change when another Sequence is selected.

When the operator clicks on the button "ConfigDownload&EquipmentSetup" a window will pop-up with the detailed state machine for that mode as shown in Figure 3. From there the operator can choose to play automatically or manually the tasks. The operator can also put a Break point in the state where (s)he want to stop the sequence. As the sequence is being executed the current state appears highlighted w.r.t. others. When playing the sequence manually, the buttons of the tasks (getConfigParameters, siwtchOnEquipment, etc) appear activated depending on the current state. In the example of Figure 3 given the current state, "EquipmentReady", only the task "cycleDownMagnets" appears activated.

If the state is "EquipmentError", like in the example of then the Operator has the possibility to look into the details of the error by clicking on the "EquipmentError" bar, and also repeat the task by clicking again on again on "switchOnEquipment" button since it appears activated again.

The Operator may have also the possibility of masking "safely certain" errors (the ones that can not put the machine in danger).

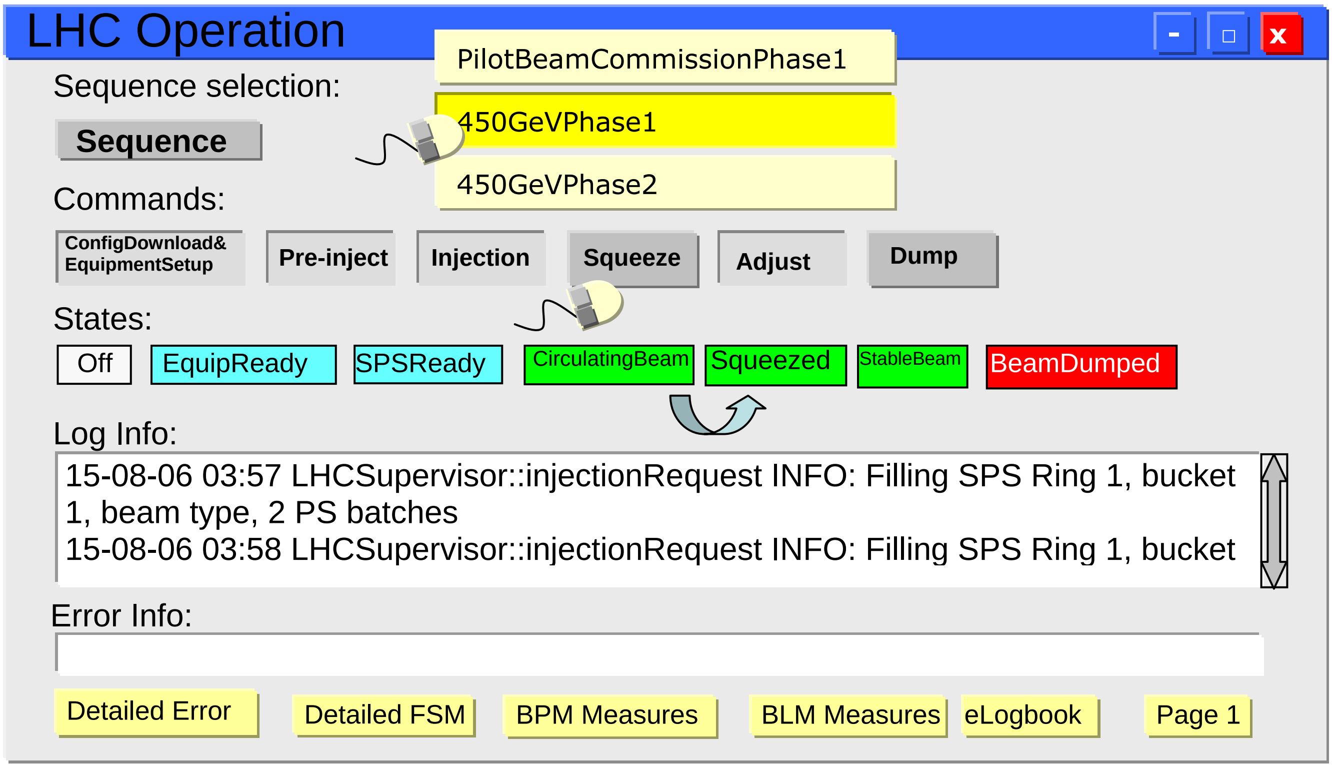

Figure 2: Example of the LHC GUI when the Sequence "PilotBeamCommissionPhase1" has been selected. Now the commands and states are not exactly the same as in Figure 1.

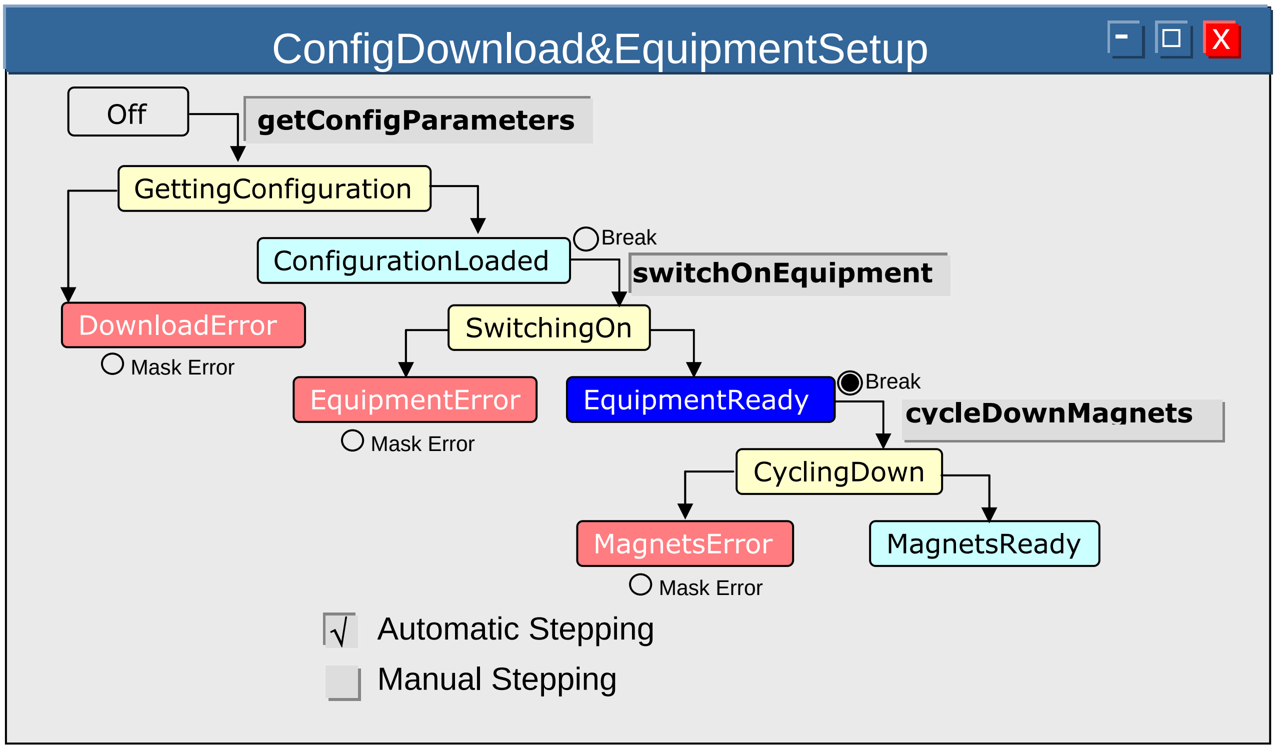

Figure 3: An example of the state Machine and tasks that will be pop-up when clicking on "ConfigDownload&EquipmentSetup".

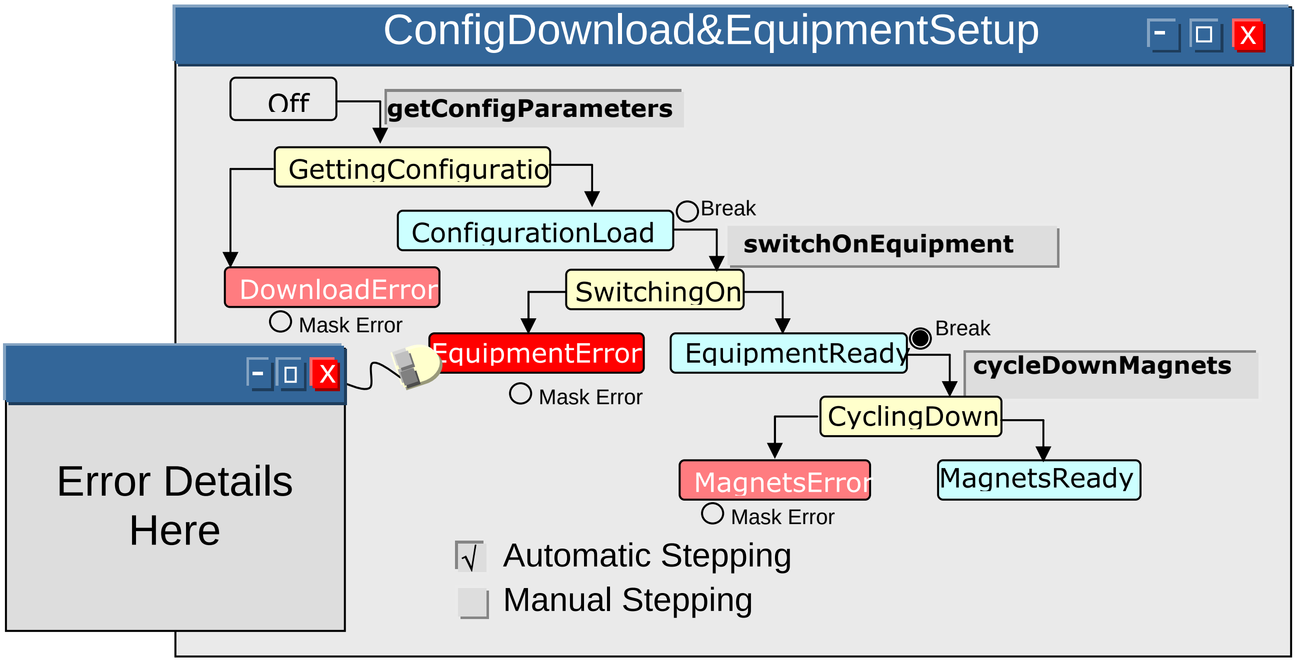

Figure 4: Error state example. The Operator has the possibility to look into the details of the error by clicking on the "EquipmentError" icon, and also repeat the task by clicking on "switchOnEquipment" button since it appears activated again. The Operator may have also the possibility of masking "certain" errors (the ones that can not put the machine in danger).

8. SECURITY

No.	**Requirement**	**Priority**
SE.R0	The sequencer shall only be executable from predefined consoles	High

9. LOGGING AND ERROR REPORTING

Logging and error reporting: records all tasks performed (& timestamp, parameters etc) and any error conditions.

No.	**Requirement**	**Priority**
LE.R0	Support standard software error format, for example Table x	High
LE.R1	Support standard logging format with different logging levels (WARNING, INFO, DEBUG, custom defined)	High
LE.R2	Support standard system (equipment errors, state machine errors, etc) error format, in the form of a database table. This kind of errors has to be stored in a conditions database, accessible and compatible with the Post-Mortem Analysis System	Critical

LE.R3	Support on-line definition of logging and software error levels	Medium

Item	Description
Identifier	Unique identification of the error, e.g. the URI if applicable
Source	Originator of the error message
Date-Time	
Message	Verbose text describing the message
Severity	String identifying the error gravity (WARNING, FATAL, ERROR, …)
NestedError (0..∞)	Collection of nested errors (if any)

Table 1: Error report format. Something very similar for logging messages could be envisaged.

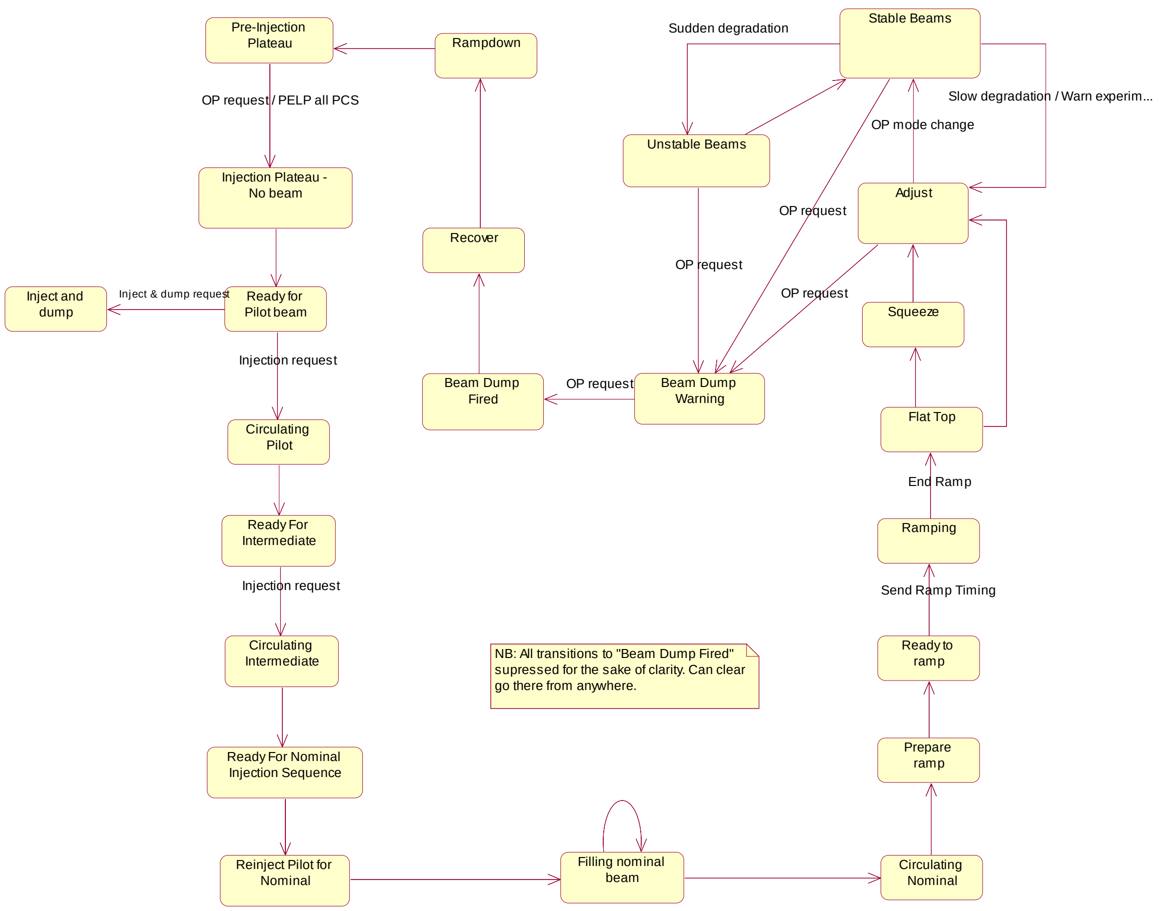

Figure 1: Nominal LHC sequence