SKA CSP Local Monitor and Control Prototype Plan

(PROTO-LMC)

Document Number SKA-TEL-CSP-0000122
Context .... LMC-PL
CSP Document Number SKA-TEL.CSP.LMC-NRC-PTP-001
Revision: 1
CIDL ID .... PROTO-LMC
Author .... Elisabetta Giani
Date 2015-12-01
Status .... Released
Classification .... Unrestricted

Prepared By	Name	Elisabetta Giani	Signature	
	Organisation	INAF - AAO	Date	2015-12-01
Reviewed By	Name	Carlo Baffa		
	Organisation	INAF - AAO	2015-10-26	
	Name			
	Organisation			
Approved By	Name	David Loop	Signature	
	Organisation	NRC	Date	2015-12-10
Issued By	Name			
	Organisation			

Abstract: This document describes the Prototype Plan for the SKA CSP Local Monitor and Control Sub-element, during pre-construction.

DOCUMENT HISTORY

Issue	Date	ECN	Change Description
A	2014-09-22		80% version
B	2014-10-27		100% version for PDR submission.
C	2015-10-19		Carlo Baffa and Elisabetta Giani updated Chapter 5 Engineering Plan and Chapter 6 Prototype Structure with the plan for INAF prototype. S.Vrcic updated the rest of the document to reflect progress made since PDR.
D	2015-10-28		Elisabetta Giani took the authorship responsibility of the document
1	2015-12-01		Released for Delta PDR submission

DOCUMENT SOFTWARE

Application		Version	Filename
Word processor	MS Word for Mac 2011	14.3.2	SD_SKA-TEL-CSP-0000122_1-LMC-Prototype_Plan-Giani-2015-12-01.docx

ORGANISATION DETAILS

Designation	Name	Organisation
Primary Author and Responsible Organisation for version A, B, and C	Sonja Vrcic	National Research Council of Canada
Contributor & reviewer	Carlo Baffa	INAF – AAO
Primary Author and Responsible Organisation for version D and beyond	Elisabetta Giani	INAF – AAO

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

AAO	Arcetri Astrophysical Observatory
ADD	Architecture Design Document
API	Application Programming Interface
ARC	Architecture Work Package
ASCII	American Standard Code for Information Interchange
CBF	Central Beam Former
CDR	Critical Design Review
CIDL	Configuration Item Data List
CM	Configuration Management
COMP	Commissioning, Operation and Maintenance Plan
ConOps	Concept of Operations
COTS	Commercial Off-The-Shelf
CP	Construction Plan
CS	Control System
CSP	Central Signal Processor
DDD	Detailed Design Document
DOC	Development and Operational Cost
DRD	Document Requirements Descriptions
DT&E	Development and Test Evaluation
EICD	External Interface Control Document
FMECA	Failure Modes, Effects and Criticality Analysis
FTA	Fault Tree Analysis
GB	Giga Byte
GNU	GNU's not Unix
GPL GNU	General Public License
HDB	Historical Data Base
ICD	Interface Control Document
IICD	Internal Interface Control Document
INAF	National Institute for Astrophysics
I/O	Input/Output
IP	Intellectual Property
LGPL	GNU Lesser General Public License
LMC	Local Monitor and Control
M&C	Monitor and Control
NIC	Network Interface Card

NRC	National Research Council (Canada)
OPS	Operations Work Package
OS	Operative System
OT&E	Operation Test and Evaluation
PC	Personal Computer
PDF	Portable Document Format
PDR	Preliminary Design Review
PIP	Physical Implementation Proposal
PSS	Pulsar Search
PST	Pulsar Timing
PTP	Prototyping Plan
QA	Quality Assurance
QP	Quality Plan
RAM	Random Access Memory
R&D	Research and Development
RFI	Radio Frequency Interference
RMP	Risk Management Plan
RR	Risk Register
SKA	Square Kilometre Array
SKAO	SKA Organisation (or office)
SOW	Statement of Work
SPA	Software Product Assurance
SRR	System Requirements Review
SW	Software
SYSML	System Engineering Simulation Language and application
TBC	To be confirmed
TBD	To be decided
TDT	Time Domain Team
TM	Telescope Manager element or consortium
TRL	Test Readiness Level
UML	Unified Modelling Language
VPL	Verification Planning Work Package
WBS	Work Breakdown Structure
WP	Work Package
WPEP	Work Package Execution Plan

1 INTRODUCTION

Each prototyping activity at the sub-element level during pre-construction will produce the Prototype Plan and Prototype Report, and will maintain the equipment, software, firmware etc. used to conduct the prototyping activity under configuration management. The Prototype Plan will follow DRD-26 [AD2]. The Prototype Report will follow DRD-24 [AD2].

A Prototype Plan states what is needed to establish the Technology Readiness Level (TRL) of CSP sub-element technologies before construction. It is required to ensure high confidence that each technology intended for the CSP element can be implemented in the SKA within the defined estimated cost and schedule.

Without exception all hardware or prototypes developed as part of the study will remain property of the consortium. The consortium will undertake to keep such prototypes in safe storage for a period not less than 3 calendar years after the completion of the study.

1.1 Purpose of Document

The purpose of this document is to describe the Prototype Plan for the Work Package 2.6 CSP Local Monitor and Control. Prototyping activities described in this document will be executed during the SKA1 Pre-construction Stage 2.

1.2 Scope of Document

This document explains in detail the prototype development programme to go from a pre-existing state or TRL as described in Section 3.1 to the target state.

1.3 Intended Audience

This document is expected to be used by the CSP Element Consortium Engineering and Management Team, the SKAO System Engineering Team, and the SKAO CSP Project Manager.

1.4 Document Overview

Chapter 1 introduces the purpose, scope and intended audience for this document.

Chapter 2 lists applicable and reference documents.

Chapter 3 provides detail on the scope of the prototype.

Chapter 4 provides the management plan for the prototype.

Chapter 5 provides the engineering plan for the prototype.

Chapter 6 provides the detailed structure of the prototype

Chapter 7 provides the risk management for the prototype.

Appendix A contains intellectual property declaration.

Appendix B contains the Technology Readiness Level table.

2 APPLICABLE AND REFERENCE DOCUMENTS

2.1 Applicable Documents

The following documents at their indicated revision form part of this document to the extent specified herein.

Table 2-1 Applicable Documents

Ref No	Document/Drawing Number	Document Title	Issue Number
AD1	SKA-TEL.CSP.MGT.NRC-SOW-001	Statement of Work for Central Signal Processor	-
AD2	SKA-TEL.SE-SKO-DRD-001	Document Requirements Descriptions	1
AD3	SKA-TEL.CSP.PROT-NZA-PLA-001	SKA CSP Element Prototyping Plan and Guidelines (SE-14)	-

2.2 Reference Documents

The following documents provide useful reference information associated with this document. These documents are to be used for information only. Changes to the date and/or revision number do not make this document out of date.

Table 2-2 Reference Documents

Ref No	Document/Drawing Number	Document Title	Issue Number
RD2	SKA-TEL-CSP-0000102	SKA1 CSP Local Monitor and Control Sub-Element Detailed Design Description (EG-2a)	-
RD3	SKA-TEL.TM.TELMGT-TMC-LIG-001	SKA1 Local Monitor and Control Interface Guidelines	-
RD4	ICALEPCS 2005	"Development of the TANGO Alarm System", L.Pivetta	-
RD5	SKA-TEL-CSP-0000155 SKA-TEL-CSP-0000156	CSP to TM ICD	A1.1
RD6	SKA-TEL-CSP-0000019	LMC to CSP ICD	A1.2
RD7	NA	The TANGO Control System Manual	8.1

3 PROTOTYPE SCOPE

3.1 Sub-element Technology

The technologies employed in the CSP.LMC sub-element are Commercial of the Shelf (COTS) hardware products and software packages that may be defined as software infrastructure or software environment, such as operating system, programming languages, communication protocols, etc. Technologies used in the prototype will be as close as possible to the hardware and software selected and document in the Detailed Design Document [RD2].

3.1.1 Target Technology

The TANGO Control System has been chosen as a communication infrastructure for interface between Telescope Manager and other SKA Elements. Agreement to use TANGO CS as a communication framework was achieved in April 2015 by all interested SKA Consortia based on the existing TANGO documentation and user experience. Further investigation and prototyping is required to verify that indeed TANGO CS is suitable for the SKA.

The TANGO Control System is based on recent technology:

- CORBA and ZeroMQ for communication between servers and clients;
- C++, Python and Java as programming languages;
- Linux and Windows as operating systems.

3.1.2 Libraries

TANGO Core libraries (RD7).

CORBA and ZermoMQ middleware.

3.1.3 Development, Testing, Archiving and Administrative Tools

The TANGO Controls System is equipped with a series of standard tools for software development, administration, GUI interface and long term archiving purposes.

They are:

- Pogo: code generator for TANGO device classes
- Jive : the TANGO database browser and device testing;
- Astor/Starter: manager to control the device servers.

A possibility to use some kind of round-robin database, such as RRDtool, will be investigated in order to retain detailed information for a long time without the necessity of keeping all the logged data on disk.

3.2 Scope of Prototype

The scope of the LMC prototype is to validate the technology (namely TANGO Control System) chosen as a communication platform for interface between TM and CSP. The goal is to determine whether TANGO CS suits CSP needs and whether the same technology can be used for monitor and control interfaces within CSP (i.e. for communication between CSP LMC and CSP Sub-elements and within individual CSP Sub-elements). Optimally, the same technology should be used for all monitor and control interfaces. If the conclusion is that the same technology cannot be used for external and internal LMC interfaces, further prototyping may be required to select alternative technology (to be used for internal CSP monitor and control interfaces).

4 MANAGEMENT PLAN

4.1 Rationale and Strategy for Prototype

The main goal for the prototyping activity will be to gain hands-on experience with TANGO CS, in order to determine whether the same set of protocols and the same approach can be used for all data flows (monitor and control messages, real-time updates, etc.) and all interfaces (external and internal). Prototyping activity will also help the CSP LMC Team make more accurate estimates of the effort required to develop CSP LMC during construction phase; and to advance TRL.

4.2 Constraints on Development Personnel Constraints

All contributors to Work Package 2.6 will participate to some extent in prototyping activities. The initial plan for Work Package 2.6 assigned about 20% of resources to prototyping. During the Pre-construction Stage 1, CSP Consortium came to the conclusion that, the main goal for the WP 2.6 (CSP.LMC) should be to generate detailed Interface Control Documents and Detailed Design. Rationale: CSP.LMC will use well established technologies and the main risk for the CSP LMC Sub-element is not related to technology used for exchange (transport) of monitor and control information, but the content to be exchanged between TM and CSP.LMC and between CSP.LMC and CSP Sub-elements. Prototyping activities will be reduced to15% use of resources, if not less. INAF-AAO is expected to give a larger contribution capitalizing on the experience gained in the development of the PSS LMC.

4.2.2 Budgetary Constraints

Testing of the CSP LMC prototypes will not require acquisition of hardware or software; no funds have been allocated for such a purpose.

4.2.3 Hardware and Facility Constraints

Participating organizations will use existing hardware (servers and/or laptop computers) at existing facilities.

4.2.4 Software Constraints

Testing of the CSP LMC prototype will use the Tango CS 8.1 release.

We will leave open the possibility to shift the prototype to Tango CS 9 as soon as we have a serviceable prototype.

4.3 Development Schedule

In both SKA1 telescopes (LOW and MID), CSP LMC will interface with 4 different entities:

1. Telescope Manager Element – An instance of the TM exists in each telescope. At this time we assume that the same approach and technology will be used in both telescopes. A single prototype will be sufficient for this interface.
2. Correlator and Beamformer (CSP_Mid.CBF, CSP_Mid.CBF)
3. Pulsar Search Non-imaging Processor (CSP_Low.PSS, CSP_Mid.PSS)
4. Pulsar Timing Non-imaging Processor (CSP_Low.PST, CSP_Mid.PST)

There is no need for extensive prototyping for each interface. Instead, prototyping will target specific types of data or a specific aspect of each interface, as appropriate. These will be selected during Stage 2.

INAF team will concentrate on prototyping for interfaces related to Pulsar Search.

INAF started prototyping activities in May 2015.

Review of the final results will be performed in March 2016.

4.4 Development Team Management

Due to the small number of staff involved in the prototyping activities, and the low risk associated with the prototype, minimal management for the prototype development team is anticipated. Table 4-1 lists the schedule for the key milestones for the management of the development team.

Progress will be reported once per month as a part of work package report.

The overall prototyping report will be released at the end of the prototyping phase.

Table 4-1 Development Team Management Milestones

Task	Resources	Date
Establish prototyping plan	Giani, Vrcic	By September 2015
Periodically review progress	Vrcic, Baffa	Progress related to prototyping activates will be reported in writing once per month.
Review final results	Vrcic, Baffa	March 2016

4.5 Configuration Management

All software used for the prototyping activities will be versioned and tagged in configuration management software repositories.

The hardware components to be used for testing of prototypes are COTS products and so will be identified with hardware, firmware and, where relevant, software/driver version numbers and recorded in the Prototype Test report. This will allow for complete reconfiguration and repeat of the prototyping test to validate the results.

5 ENGINEERING PLAN

This chapter describes the engineering plan for the prototype developed by INAF.

The definitions for TRLs used in this document are taken from the U.S. Department of Defence and are defined in [AD3]. For the ease of reference a copy of TRL definition is provided in Appendix B.

5.1 Starting TRL for Prototype

All hardware components (e.g., servers, NICs) are COTS: they can each be considered at a TRL of 9. As an assembled hardware system, the prototype hardware is assessed at a TRL of 5.

TANGO Control System has been chosen as a communication framework for the SKA monitor and control. TANGO CS is an open source technology used by a number of synchrotrons and other large physics facilities. TANGO CS TRL is 9. However, TRL for the new version of TANGO which uses ZeroMQ as the communication infrastructure will be lower (and is still to be defined).

CSP LMC group will develop new software to verify not only the functionality and performance of the underlying protocol stack (or communication framework), but also time required to adopt the technology and implement functionality required by LMC. Starting TRL for the software developed by CSP LMC group will be at the lowest level (i.e. TRL 1).

5.2 Target TRL for Prototype

Depending on the technology chosen, the target for the LMC prototype(s) will be to achieve TRL 4 or 5. Higher TRL (5) will be achieved, in the case that the same technology can be used for external and internal interfaces, and if underlying communication software can be used as-is (i.e. no additional work required by CSP.LMC).

5.3 Critical System Requirements and Technologies

At this time it is believed that CSP LMC can be verified in the general purpose network. The prototype hardware may consist of one server and several laptop computers connected in a network. To emulate CSP_Mid.LMC environment 6 computers connected in a network will be required.

5.4 Verification of Progress

Each of the major categories of prototyping activity will have its progress and results analysed as follows:

1. Create test plan, i.e. determine the goals (what exactly should be tested and verified).
2. Create development plan.
3. Install prototype environment, i.e. protocol stack or communication framework chosen for TM to CSP communication.
4. Develop CSP LMC prototype.
5. Develop software to emulate TM and CSP Sub-elements, i.e. software that generates messages, receives messages from CSP LMC and generates replies.

6. Execute test plan, record results.
7. Analyse results and generate prototype report.

5.5 Prototype Verification

To assess the success of the prototype we need to implement a subset of the final Test Plan. The aim of our prototype is to move the LMC software stack TRL from 3 to 5 and maybe 6, if all sub-element interfaces share the same internal approach.

The tests to be performed are detailed in Table 5-1.

Table 5-1 Tests for prototype verification

Test name	Description
Connectivity test	This test checks if the TM and sub element emulators can connect to LMC prototype.
Configuration SET and GET	This test verifies that the CSP.LMC: 1. accepts valid configuration parameters SET by Telescope Manager. 2. forwards valid configurations to appropriate CSP sub-elements. 3. queries the CSP sub-elements in response to GET command request from TM and sends the required configuration information back to TM.
Monitor points status report	This test confirms that the monitoring points status of CSP sub-elements is reported to TM
Logging Test	This test checks if the logging requirements of CSP.LMC are met.
Faults and Errors - Reporting and Latency	This test make sure that: 1. The faults and errors generated by CSP.LMC are reported to TM. 2. The alarm messages are sent within five seconds of detecting an alarm.
Operational Mode Persistence Test	This test ensures that the CSP.LMC maintains the operational mode in case of a system restart, reboot or power-shutdown.
CSP.LMC to TM Interface Tests	This test verifies if the form and format of the information exchange between CSP.LMC and TM takes place as in the TM-CSP ICD.
Start-up and Low Power Standby Mode Tests	This test verifies if the CSP.LMC becomes fully operational upon start-up and it remains operational also in low-power mode.

6 PROTOTYPE STRUCTURE

The prototype will consist of hardware and software.

The hardware will include few related COTS PC servers, connected by a fast network.

The software will be composed by:

1. a TANGO database.
2. some SKA custom TANGO classes, as device drivers.
3. the TANGO core engine
4. some TANGO diagnostic tools.

Item 1 will run in only one server, while *item 3* and *4* are included in each machine. The SKA TANGO Classes (*Item 2*), will be also present, in different version, on each server.

In the development phase, the different server instances could be run on the same system. However, the final tests must run in a network of connected servers so to get sound results.

6.1 The Hardware Structure

The hardware component of this prototype will consist of a fast network of connected PC servers running a suitable Operating System. Our present intention is to employ a recent brand of Linux/GNU.

Each server should have a minimum amount of Ram, (>4 GB), a fast hard disk, and one or more 1GB Ethernet interfaces. We can safely assume they will share the same OS version.

The first server will act as the CSP-LMC node, while others will play the role of CSP sub-element LMC nodes, and the last will contain a TM simulator. To have a better understanding of performance we can add a small network of very low-end machines to act as PSS nodes.

6.2 The Software Logical Components

The software component can be logically split, in its TANGO implementation into:

1. attribute collection (from TM)
2. attribute forwarding (to sub-elements)
3. semi-permanent data tables forwarding to sub-elements
4. info collection (from sub-elements)
5. execution of commands
6. alarm and event handling
7. logging

The CSP LMC will see sub-elements as TANGO devices. To access a TANGO device, a TANGO client (CSP LMC) uses the TANGO API library.

The API supplies a set of high level classes and methods to access a device and communicate with it.

High level API in C++, Python and Java have been implemented to completely hide the low level interface.

To these components, there are also some software tools which can be profitably employed during the prototype work to speed up the testing/optimization phase:

a) Statistics of execution of TANGO components and device drivers (Astor).
b) Medium term storage of programmed parameters and attributes of sub-elements (HDB and HDB++).
c) Graphical display of status and performances (E-Giga, Jive, Astor).

6.3 The LMC TANGO Devices

The SKA TANGO device drivers will be organized in a layered structure, following the recommendation of the TANGO Project (RD7). The software structure of LMC Prototype will reflect such organization.

At the top level of the software structure, in direct communication with TM, there will be a single device driver. This will be a *logical* driver, as it aims to control a logical structure and not a physical object. This single device driver will connect to the second and third level devices only, to be isolated from the external sub-element LMC (or sub-element Master) nodes.

At the second level will reside the Alarms Handler Device Drivers, one for each sub-element. These drivers will receive data and alarm condition warnings mainly from the appropriate sub-element alarm driver. As some error conditions that are easily spotted only at CSP-LMC level may occur, these Alarm Drivers will also be able to receive error conditions from the *internal logical* device drivers (level 3). There will be one such alarm server for each sub-element and one related to the CSP as a whole.

The third level will contain four *logical* drivers, which will be the LMC node interfaces towards the four CSP sub-elements. This level also contains a *physical* driver which is relative to the server system (hardware and OS monitor and control). This structure does not impose significant overhead to the machine (all communications are inside the same machine) but it makes a clean and logical structure where each module has separate and non-interacting roles, with obvious advantages in terms of development speed and maintainability.

We propose to also include in this level, one or more additional *logical* drivers in order to monitor, and perhaps control, the Capabilities defined inside CSP. The information handled by these drivers is redundant, as it can be obtained by summing up information contained in one or more sub-element drivers, but may come in very handy for the engendering monitor and for procedure optimization.

The required characteristics of the interface toward TM are described in RD5, and that of the internal interfaces are described in document RD6. The prototype will adhere to such specifications for the TANGO device attribute definition, while adding some TANGO specific items described in the following.

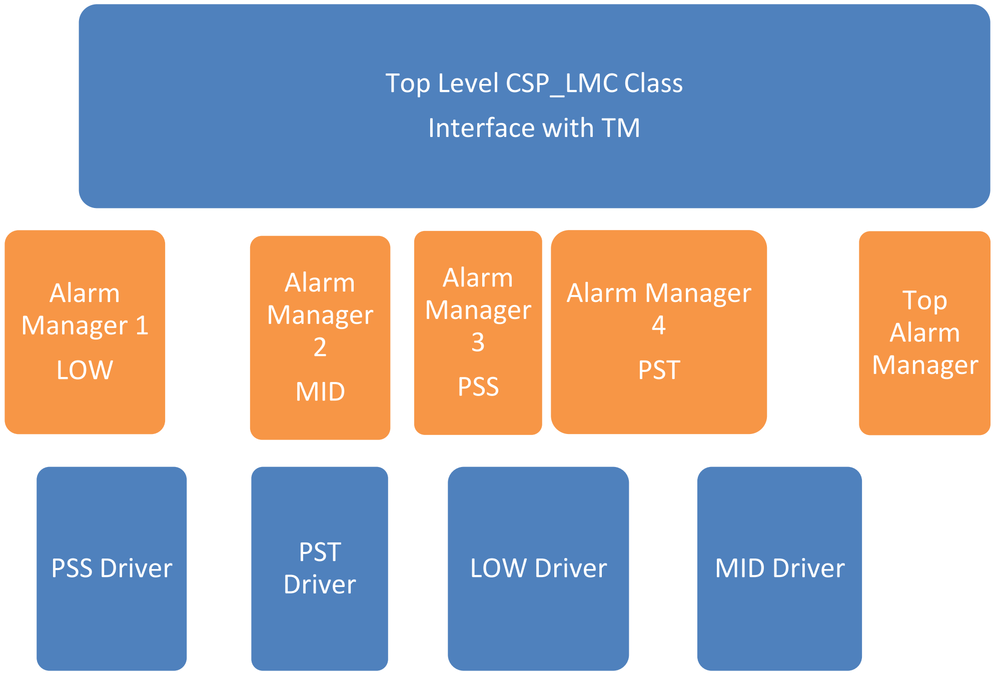

Figure 1: Layered structure of TANGO Device Classes

6.4 TANGO Device Driver Implementation

6.4.1 The TANGO Base Classes

The basic classes TANGO uses in each device driver are:

- the DeviceProxy class which is a *proxy* to the real device
- the DeviceData class to encapsulate data sent/received to/from devices via commands
- the Device Attribute class to encapsulate data sent/received to/from devices via attributes
- the Group class which is a proxy to a group of devices
- the Database, DbDevice, DbClass, DbServer or DbSata classes to interface with the database that is implemented as any other TANGO device.

6.4.2 Commands Execution

The API allows command execution on a device or to read/write a device attribute(s). The client can execute a command both in synchronous and asynchronous mode. The latter mode makes both the polling and callback method available.

TANGO offers an "event system" as well. In this case the client subscribes its interest once in one or more events (attribute changes, parameter alarm conditions, status changes etc.) and the device server informs the client every time the event has occurred.

6.4.3 TANGO Alarm System

The TANGO Control System provides a service inside the Device Server Class to raise an alarm condition if a monitored variable is outside a programmed interval or the server code explicitly detects such a condition. TANGO does not possess an Alarm engine in its core implementation, so more complex *alarm conditions* cannot be handled in the core system.

However a fairly general Alarm handler class has been made available by one of the members of the TANGO community (Elettra, see [RD4]).

Each instance of this class receives alarm signals from up to 30 groups of device servers and, according to one or more logical expressions programmed at run time and stored in a database, can raise one or more *alarm conditions.*

6.4.4 TANGO Logging System

A logging service incorporated in TANGO (TANGO Logging Service) provides support for logging to:

- the console
- a file
- a specific TANGO device (log consumer)

There is also a specific TANGO class (HDB++) capable of permanently storing (MySql or Cassandra) a substantial amount of attribute data in a database. This feature is not required at the date of this document, but can be added easily and can prove valuable for debugging, profiling and optimization purposes.

6.5 The TM Interface

The interface towards TM will consist of status variables, commands and operating parameters, and they will be all handled by the TANGO driver as attributes.

6.5.1 Modes and States

The monitoring and control of the overall status of CSP and its sub-elements are handled by a set of modes and states attributes (RD5). The first one will be implemented as read-write attributes, while the second one as read-only because they will reflect the internal status of each element and sub-element.

At the present time the standard set of modes and states for the SKA telescope is not yet defined. The prototype will adhere to the definition of modes and states provided in the ICD CSP to TM (RD5).

6.5.2 Parameters

All parameters specified in ICD CSP to TM (RD5) and ICD LMC to CSP sub-elements (RD6) can be mapped to scalar or vector TANGO types listed in Table 6-2 **Error! Reference source not found.**types.

Also the monitoring quantities (also known as *telemetry*), are returned to TM as scalar or vector of Table 6-2 types.

6.5.3 Data Tables

For some sub-elements (most notably PSS), the ICD LMC to CSP sub-elements (RD6) lists a set of TM generated data tables, slowly changing with time. Examples of such tables are known RFI and known Transient and Artificial Satellite parameters.

Such tables can be easily handled as attributes (the maximum expected dimension is less than 100K elements). However we suggest use of a 'data server' model, already in place for the parameter distribution for the data analysis pipeline of PSS. Such data server is configured as a simple Json data server and receiver.

The arguments in favor of such an approach are:

- simplicity of server and less load on time critical TANGO drivers
- ASCII protocol, easily tested by a browser

The only drawback is that a ‘data server’ solution would take more time to develop and test.

6.5.4 Commands

TANGO provides a basic set of commands (see Table 6-1) which covers nearly all CSP LMC needs. TANGO commands include set parameters, query parameters, start/stop polling, subscribe/unsubscribe, change notification and logging management.

Table 6-1 Tango commands

Command name	Input data type	Output data type
State	void	TANGO::DevState
Status	void	TANGO::DevString
Init	void	Void
DevRestart	TANGO::DevString	Void
RestartServer	void	Void
QueryClass	void	TANGO::DevVarStringArray
QueryDevice	void	TANGO::DevVarStringArray
Kill	void	Void
QueryWizardClassProperty	TANGO::DevString	TANGO::DevVarStringArray
QueryWizardDevProperty	TANGO::DevString	TANGO::DevVarStringArray
QuerySubDevice	void	TANGO::DevVarStringArray
StartPolling	void	Void
StopPolling	void	Void
AddObjPolling	TANGO::DevVarLongStringArray	Void
RemObjPolling	TANGO::DevVarStringArray	Void
UpdObjPollingPeriod	TANGO::DevVarLongStringArray	Void
PolledDevice	void	TANGO::DevVarStringArray

Command name	Input data type	Output data type
DevPollStatus	TANGO::DevString	TANGO::DevVarStringArray
LockDevice	TANGO::DevVarLongStringArray	Void
UnLockDevice	TANGO::DevVarLongStringArray	TANGO::DevLong
ReLockDevices	TANGO::DevVarStringArray	Void
DevLockStatus	TANGO::DevString	TANGO::DevVarLongStringArray
EventSubscribeChange	TANGO::DevVarStringArray	TANGO::DevLong
ZmqEventSubscriptionChange	TANGO::DevVarStringArray	TANGO::DevVarLongStringArray
AddLoggingTarget	TANGO::DevVarStringArray	Void
RemoveLoggingTarget	TANGO::DevVarStringArray	Void
GetLoggingTarget	TANGO::DevString	TANGO::DevVarStringArray
GetLoggingLevel	TANGO::DevVarStringArray	TANGO::DevVarLongStringArray
SetLoggingLevel	TANGO::DevVarLongStringArray	Void
StopLogging	void	Void
StartLogging	void	Void

6.5.5 Timed Set Command

The ICD CSP to TM (RD5) states that to expedite configuration changes, especially related to observing modes, the SET parameter message can specify the Activation Time. In this context it’s important to add a *timed set command* to the list of default commands. Because it is available a timeserver with a precision of at least tens of milliseconds, each group of set commands will be registered at once, but programmed only at the wall time contained in the *timed set command argument.* This command is useful mainly in the top levels of the control hierarchy, but it might come in handy at deeper levels as well. In our view, this command will ease the coordination of the various portions of SKA.

Table 6-2 Tango available parameter types

TANGO available types	
TANGO::DevBoolean	Boolean
TANGO::DevShort	Mapped to short signed integer
TANGO::DevLong	Mapped to long signed integer (4 bytes)
TANGO::DevLong64	Mapped to int64 signed integer
TANGO::DevFloat	Mapped to single float
TANGO::DevDouble	Mapped to double precision float
TANGO::DevUChar	Mapped to unsigned char
TANGO::DevUShort	Mapped to short unsigned integer

TANGO available types	
TANGO::DevULong	Mapped to long signed integer (4 bytes)
TANGO::DevULong64	Mapped to int64 signed integer
TANGO::DevString	Mapped to character string
TANGO::DevState	TANGO specific structure
TANGO::DevEncoded	Structure: format plus character string

6.6 Interface LMC to CSP Sub-elements

The interfaces toward the external sub-elements are composed by a standard set of status variables (as in paragraph 6.5.1) with some values censored or not applicable.

Programming parameters, statistical and telemetry data can be returned as attributes in the form of scalar or vector of Table 6-2 types.

Inside the *logical* device driver, the same form of data reduction on telemetry can be performed, such as condensing many point temperatures in a min-average-max triplet. Also, for maintenance purposes for most relevant telemetry quantities, a round robin RRD tool repository will be in place to get a graphical instant view of present and short term history.

Details of those quantities are described in ICD LMC to CSP Sub-elements (RD6).

7 RISK MANAGEMENT

Table 7-1 Risks and Risk Mitigation Strategies

Risk	Likelihood	Status	Strategies
Choice of communication platform. (CSP Risk Register LMC-8)	Low	Active	Develop prototype to verify whether the communication platform chosen by TM Consortium is suitable for CSP and can meet all CSP minor and control needs. Document findings and communicate to SKAO, TM Consortium and CSP sub-element teams. Participate in decision making. If necessary find other candidates and develop additional prototypes.
Suitable prototyping hardware cannot be found in time to execute test plan (there is no plan to acquire additional hardware for CSP LMC prototyping).	Very Low	Retired	INAF started work on prototyping and has adequate hardware.
Communication platform (protocol stack or framework) is not open source, requires significant investment.	Low	Retired	Retired in April 2015 when TANGO CS was chosen as communication framework.
Choice of the communication platform delayed. CSP.LMC not sure which platform to use for prototyping.	Low	Retired	Retired in April 2015 when TANGO CS was chosen as communication framework.
Communication platform requires significant additions and learning time. (CSP Risk Register LMC-10)	Low	Active	Develop prototype.
CSP.LMC resources deflected due to changes or late arrival of requirements and information from the CSP sub-elements and Telescope Manager. During Pre-construction Stage 2, generation of detailed Interface Control Documents and detailed design should take precedence over prototyping.	Medium to High	Active	It is quite likely that CSP Sub-elements will provide detailed list of parameters, alarms, monitor points and such only towards the end of the Pre-construction Stage 2 (as the design progresses). Start prototyping as early as possible (but not too early so that prototype is irrelevant to the actual design), complete prototyping activities as early as possible.

A APPENDIX: INTELLECTUAL PROPERTY DECLARATION

A1 Tango intellectual property status

The TANGO Controls Core system is distributed under a mixture of LGPLv3 and GPLv3.

The TANGO libraries are released under LGPLv3

The TANGO tools are under GPLv3.

A2 INAF intellectual property declaration

All code developed at INAF is released under the GPL licence, hence it does not require any implicit or explicit fees for licensing.

B APPENDIX: TECHNOLOGY READINESS LEVEL DEFINITIONS

Table B-1 U.S. Department of Defence TRL definitions

Technology Readiness Level	Description
1. Basic principles observed and reported	Lowest level of technology readiness. Scientific research begins to be translated into applied research and development (R&D). Examples might include paper studies of a technology's basic properties.
2. Technology concept and/or application formulated	Invention begins. Once basic principles are observed, practical applications can be invented. Applications are speculative, and there may be no proof or detailed analysis to support the assumptions. Examples are limited to analytic studies.
3. Analytical and experimental critical function and/or characteristic proof of concept	Active R&D is initiated. This includes analytical studies and laboratory studies to physically validate the analytical predictions of separate elements of the technology. Examples include components that are not yet integrated or representative.
4. Component and/or breadboard validation in laboratory environment	Basic technological components are integrated to establish that they will work together. This is relatively "low fidelity" compared with the eventual system. Examples include integration of "ad hoc" hardware in the laboratory.
5. Component and/or breadboard validation in relevant environment	Fidelity of breadboard technology increases significantly. The basic technological components are integrated with reasonably realistic supporting elements so they can be tested in a simulated environment. Examples include "high-fidelity" laboratory integration of components.
6. System/subsystem model or prototype demonstration in a relevant environment	Representative model or prototype system, which is well beyond that of TRL 5, is tested in a relevant environment. Represents a major step up in a technology's demonstrated readiness. Examples include testing a prototype in a high-fidelity laboratory environment or in a simulated operational environment.
7. System prototype demonstration in an operational environment.	Prototype near or at planned operational system. Represents a major step up from TRL 6 by requiring demonstration of an actual system prototype in an operational environment (e.g., in an aircraft, in a vehicle, or in space).
8. Actual system completed and qualified through test and demonstration.	Technology has been proven to work in its final form and under expected conditions. In almost all cases, this TRL represents the end of true system development. Examples include developmental test and evaluation (DT&E) of the system in its intended weapon system to determine if it meets design specifications.
9. Actual system proven through successful mission operations.	Actual application of the technology in its final form and under mission conditions, such as those encountered in operational test and evaluation (OT&E). Examples include using the system under operational mission conditions.