



Atacama Large Millimeter Array

ALMA Common Software Technical Requirements

COMP-70.25.00.00-0004-C-SPE

Version: C
Status: Draft

2005-09-26

Prepared By:	Organization	Date
Gianni Raffi Brian Glendenning Joseph Schwarz	European Southern Observatory National Radio Astronomy Observatory European Southern Observatory	2005-09-26
IPT Leader Approvals:	Organization	Date
B. Glendenning	National Radio Astronomy Observatory	
System Engineering Approvals:	Organization	Date
R. Sramek	National Radio Astronomy Observatory	
C. Haupt	European Southern Observatory	
Configuration Control Board Approval:	Organization	Date
C. Haupt	ALMA Configuration Control Board Secretary, signing for the Control Board	
JAO Director Release Authorization:	Organization	Date
M. Tarenghi	Joint ALMA Office	



**Atacama
Large
Millimeter
Array**

	Project Director	
--	------------------	--

Document Name: ACSTR_050926.doc



ALMA Project
ALMA Common Software
Technical Requirements

Doc # : COMP-70.25.00.00-0004-C-SPE
 Status: Draft
 Date: 2005-09-26
 Page: 3 of 28

Change Record

Version	Date	Affected Section(s)	Change Request #	Reason/Initiation/Remarks
A	2000-03-10	All	G.Raffi	First version
A	2000-06-05	All	G.Raffi	Reviewed version
B	2003-05-21	End part	J.Schwarz	Container/component requirements, reformatted with ALMA doc. template
C	2005-07-14	5.2, 5.4, 8.1, 8.3, 8.4, 9, added 15	G.Raffi	Added section 15. (from eVLA) and other requirements with change bars (in 5.2, 5.4, 8.1, 8.3,8.4). Reformatted according to new template. Deleted some standards (in 9) referring to Computing standards.
<u>C</u>	<u>2005-09-26</u>		<u>G.Raffi</u>	<u>COMP review comments taken into account</u>



Table of Contents


1 INTRODUCTION.....	6
1.1 Scope.....	6
1.2 Overview.....	6
1.3 Reference Architecture.....	6
1.4 Applicable and Reference Documents.....	7
1.5 Glossary.....	7
2 GENERAL REQUIREMENTS	7
1.1 ACS scope.....	7
2.2 User requirements	8
2.3 System requirements.....	8
3 DISTRIBUTED OBJECT VALUES.....	9
3.1 Value retrieval.....	9
3.2 Database.....	10
3.3 Sampling	10
4 TOOLS AND LIBRARIES.....	10
4.1 Tools.....	10
4.2 Scripts.....	11
4.3 Libraries	11
5 CONTROL INFORMATION FLOW	11
5.1 Messages.....	11
5.2 Logging	12
5.3 Errors.....	13
5.4 Alarms	13
6 ASTRONOMICAL DATA FLOW.....	14
6.1 Bulk data transfer.....	15
6.2 Data format.....	15
7 TIMING SIGNALS FLOW.....	15
7.1 Time System.....	15
8 ATTRIBUTES.....	15
8.1 Security.....	16
8.2 Safety.....	16
8.3 Reliability.....	16
8.4 Performance.....	17



ALMA Project
ALMA Common Software
Technical Requirements

Doc # : COMP-70.25.00.00-0004-C-SPE
Status: Draft
Date: 2005-09-26
Page: 5 of 28

9 STANDARDS.....	17
9.1 Standards and products.....	17
9.2 Software standards.....	18
9.3 Communication standards	18
9.4 Reference products.....	19
10 LIFE CYCLE ASPECTS	19
11 INTERFACE REQUIREMENTS.....	20
11.1 User interface.....	20
11.2 Hardware interfaces.....	21
11.3 Software interfaces.....	21
12 ACS DESIGN REQUIREMENTS	21
13 REQUIREMENTS FOR APPLICATIONS.....	22
13.1 Style guide requirements.....	22
14 CONTAINER/COMPONENT REQUIREMENTS.....	23
14.1 Components.....	23
14.2 Container Services.....	24
14.3 Entity Objects.....	24
15 APPLICABLE EVLA REQUIREMENTS.....	25
15.1 Device Browser Tool.....	25
15.2 Online Support.....	27
15.3 Plotting.....	27

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 6 of 28</p>
--	--	--

1 INTRODUCTION

1.1 Scope

The purpose of this document is to define the technical requirements for the ALMA Common Software (ACS).

These requirements are complementary to the high level scientific and technical requirements (given in [SSR]). The user of the ACS software is typically a developer, producing specific application software to satisfy those requirements. The requirements expressed here reflect the needs of developers and the constraints of the ALMA project.

1.2 Overview


The ACS software is located in between the application software (on top) and other commercial or shared software over the operating systems. ACS is meant to be the kernel of a larger system of software that relies on ACS. Any software that utilizes ACS, either as an extension to ACS or through use of the ACS primitives, must be tested and distributed as an integrated unit.

Along with Requirements for ACS there are also Requirements for Application software written using ACS. It is in fact one of the purposes and advantages of ACS, to enforce standards simply by asking to use the ACS software in all applications. The requirements for application software are grouped together in a separate chapter.

Requirements that are not fully clear are marked as TBD or TBC.

1.3 Reference Architecture

A distributed architecture based on computers at the ALMA antennas and a number of central computers is the reference architecture. The antennas will

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 7 of 28</p>
--	--	--

be linked to the central computers via a Local Area Network (LAN) while a Fieldbus (CAN) will connect devices within a single antenna.

1.4 Applicable and Reference Documents

1.4.1 Applicable Documents

[STD] COMP-70.20.00.00-001-B-STD, ALMA Software and Computer Hardware Standards

1.4.2 Reference Documents

[SSR] ALMA-70.10.00.00-002-A-SPE, ALMA Software Science Requirements and Use Cases

[EVLAOps] EVLA-SW-003, EVLA Array Operations Software Requirements

[EVLAEng] EVLA-SW-004, EVLA Engineering Software Requirements

1.5 Glossary

The ALMA Software Glossary can be found at the URL <http://almaedm.tuc.nrao.edu/forums/alma/dispatch.cgi/Glossary/docProfile/100004/d20030408125457/No/Glossary.pdf>

2 General requirements

1.1 ACS scope

2.1.1 Scope. ACS shall contain software that is common and supporting the various applications (like antenna control, correlator software, observation preparation etc). It shall fulfill the technical and scientific requirements of ALMA.



ALMA Project
ALMA Common Software
Technical Requirements

Doc # : COMP-70.25.00.00-0004-C-SPE
Status: Draft
Date: 2005-09-26
Page: 8 of 28

2.1.2 Design. The design of ACS shall offer a clear path for the implementation of applications, with the goal of obtaining implicit conformity to design standards and the production of maintainable software. In particular, the separation of technical and functional concerns in the application software shall be supported by ACS.

2.1.3 Use. The use of ACS software is mandatory in all applications, except when the requested functionality does not exist. Exceptions to this shall be explicitly authorized at design review and shall be implemented with a design compatible with ACS.

2.2 User requirements

2.2.1 Users. These are Developers, Operators performing routine maintenance operations via standard tools of ACS, and Maintenance staff, who typically access low level functionality of the system.

2.2.2 Value retrieval. It shall be possible to retrieve locally and remotely information in read mode without interfering with control actions.

2.2.3 Value setting. This shall normally be done programmatically or via GUIs enforcing access to different kind of values (engineering GUIs for engineering values, user GUIs for normal users).

2.2.4 Local and central operation. Every software operation available shall also be available at the Control centre in San Pedro, without significant degradation in performances.

2.2.5 Remote access. Remote access from the US, Europe and Japan shall be possible and reliable within given rules and security ~~limits~~ ~~(TBD)~~ boundaries. Performances will be on a best effort basis.

2.3 System requirements

2.3.1 Size. ACS shall be capable to cope with 100.000 Distributed Objects (extrapolating from the assessment of about 1000 I/O points/antenna, which will




the basis for most distributed objects in the system and adding the need for purely logical objects on top).

- 2.3.2** **Serialization.** ACS shall provide a simple serialization mechanism to create on demand a persistent image of certain types of objects. Default persistent values will be applied at start-up, and default or other persistent values can be used for re-initialization.
- 2.3.3 **Migration.** It must be possible for special objects (e.g. observing blocks) to make use of Serialization to migrate across links (rather than being accessed where they are).
- 2.3.4 **Simulation.** ACS shall support distributed objects in simulation. This allows testing when some or all the hardware is unavailable.

3 Distributed object values

3.1 Value retrieval

- 3.1.1 **Direct value retrieval.** It shall be possible to retrieve values directly, typically for programmatic use.
- 3.1.2 **Indirect value retrieval.** This consists in retrieval of values that have been cached, to provide a mirrored view of the running system. This allows to keep system and network load under control.
- 3.1.3 **Rate.** Indirect value retrieval shall be available both in periodic form and on change.
- 3.1.4 **Transparency.** Applications should be the same for getting direct or indirect values. This could simply depend on start-up parameters, but the code should not need changing.
- 3.1.5 **Values at given time.** Time stamped data requests shall allow to retrieve values at a specified array time.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 10 of 28</p>
--	--	---

3.1.6 Data channel. Event/data channels to provide connections between producing and consuming objects shall be supported by ACS.

3.2 Database

3.2.1 Configuration Database. Distributed object (deployment) information shall be stored in a Database, to be downloaded at start time and to be accessed whenever needed to retrieve definitions.

3.2.2 Database design. The database design will be compatible with that of the Archive subsystem.

3.3 Sampling

3.3.1 Sampling. It shall be possible to store samples for later analysis or plotting, sending buffered values to keep network load under control.

4 Tools and Libraries

4.1 Tools

4.1.1 Framework. The common software shall provide an application framework that allows an easy implementation of a standard application with all essential features and standard messages.

4.1.2 Procedures. Start-up and shut-down procedures shall exist in ACS, providing also a framework for applications.

4.1.3 Browser. There shall be a Tool to browse through, display and modify object values. This should know the configuration of the objects installed and their availability, including the fact that they are accessible or not to a given user.

4.1.4 Analysis tool. An existing engineering data analysis tool to display several object values at the same time shall be integrated in ACS.



4.2 Scripts

- 4.2.1 Scripts. A technical scripting language shall be part of ACS. It will be used for high level procedures such as coordination functions, prototyping and test scripts.
- 4.2.2 Functionality. The scripting language shall provide access to the basic features of ACS, having interfaces to CORBA as a pre-requisite.

4.3 Libraries

- 4.3.1 Time conversion, arithmetic and formatting (could be a Time class).
- 4.3.2 Mathematics (e.g. FFT library for correlator)
- 4.3.3 Astrometry

5 Control information flow

5.1 Messages

- 5.1.1 Messages include commands and normal or error replies. Command replies can return information along with command acknowledgement.
- 5.1.2 Commands. They are the way to perform actions or to request information to distributed objects. They normally shall have a set of associated input and output parameters.
- 5.1.3 Syntax Check. The syntax of commands shall not need to be checked for correctness when the message is sent programmatically. This does not exclude that there might be tools (e.g. GUIs, interactive commands, scripts) where it is desirable to do a syntax check also at the origin.
- 5.1.4 Validation. Range checking and full context validation of message syntax shall in any case be done at the receiving end.
- 5.1.5 Programmatic use. It shall be possible to invoke commands both programmatically and interactively, including a generic tool performing also



ALMA Project
ALMA Common Software
Technical Requirements

Doc # : COMP-70.25.00.00-0004-C-SPE
Status: Draft
Date: 2005-09-26
Page: 12 of 28

syntax checking. There shall only be one file being used for the syntax and range checks mentioned under Syntax Check, Validation and Programmatic use.

- 5.1.6 Command delivery. Command delivery shall be reliable, i.e. commands shall always be acknowledged.
- 5.1.7 Mode. It shall be possible to send commands either synchronously or asynchronously.
- 5.1.8 Timeout. An adjustable timeout shall be supported for commands (so that users get a reply in any case).
- 5.1.9 Intermediate replies. These are allowed for asynchronous commands and there shall be a way to link replies to messages (e.g. headers) and to identify the final reply.

5.2 Logging

- 5.2.1 Logging. Applications shall be able to log information at run time according to specific formats in order to log a record of: execution of actions, status, anomalous conditions. All log records will have an associated array time.
- 5.2.2 Persistency. Short term persistency of logs shall be provided by having the total size and/or time span of such logs adjustable. Long term persistency shall be provided by a backup policy into persistent store.**
- 5.2.3 Filtering. It shall be possible for log messages to be searched and filtered (both in input, e.g. to avoid flooding of identical messages and in output).
- 5.2.4 Standard API: Where the implementation language provides a standard logging API, this API will be used to wrap the underlying functionality provided by ACS, so that application developers can use this standard.
- 5.2.5 The user shall have the ability to query the persistent store (by time range, message, type, program, etc.).
- 5.2.6 The logging system must be distributed and platform independent (so that it can be easily embedded in other applications).




5.3 Errors

- 5.3.1 Definition. An Error is a failure being reported back either synchronously after a command or asynchronously.
- 5.3.2 Tracing. Errors shall be traceable at interesting levels.
- 5.3.3 Severity. Errors shall have a Severity attribute. For serious unrecoverable errors the subsystem concerned shall fall back to a safe state.
- 5.3.4 Presentation. A standard way to report errors to users shall be defined for all user interfaces.
- 5.3.5 Configuration. Error, help text and descriptions of manual recovery procedures, shall be predefined in error configuration files.
- 5.3.6 Scope. Error messages shall be reported only where they occur (normally to the action requester), so that a faulty subsystem cannot flood with errors other parts of the system.


5.4 Alarms

- 5.4.1 Definition. Alarms are events associated with subsystem anomalies, which rely on continuous retrieval of relevant distributed object values.
- 5.4.2 Behaviour. It shall be foreseen to categorise alarms, so that some alarms just disappear when the alarm condition is cleared, while others will require an explicit acknowledgement.
- 5.4.3 Severity. Alarms shall have a Severity attribute. For serious unrecoverable alarms the subsystem concerned shall fall back to a safe state.
- 5.4.4 State. Alarms should have a state, e.g., **FIRST OCCURRENCE**, **MULTIPLE OCCURRENCE**, **TERMINATED**, **IDLE**, etc.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 14 of 28</p>
--	--	---

- 5.4.5 Hierarchy. It shall be possible to define hierarchical alarms: by default only high level alarms are shown to the operator.
- 5.4.6 Alarm logging. All alarm transitions shall be logged.
- 5.4.7 Configuration. Alarms shall be predefined in alarm configuration files. These should also provide in addition help text and descriptions of manual recovery procedures.
- 5.4.8 Binding. It shall be possible to define actions to be started on the occurrence of alarms. In general event driven software shall be supported by ACS.
- 5.4.9 The system shall allow messages to be color-coded based on the type and severity level of the message.
- 5.4.10** The system shall be able to provide audible alerts for fault conditions based on the type and severity level of the message.
- 5.4.11** The user shall have the ability to view the fault conditions for the following groups: a single antenna, all antennas, a sub-array, a subsystem
- 5.4.12** At a minimum, the system shall provide access to the following fault condition properties: the date/time of the message, origin of the message (computer, subsystem, etc), the message code, a terse description of the problem, a severity level, a detailed description of the problem
- 5.4.13** The user shall have the ability to redirect warning or error messages to: the operator (or system) log, a printer.
- 5.4.14 The user shall have the ability to retrieve and sort messages by time, message code, type, source (subsystem) and severity level.

6 Astronomical data flow

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 15 of 28</p>
--	--	---

6.1 Bulk data transfer

6.1.1 Image pipeline data transfer rates of 4 MB/sec sustained and 40 MB/sec burst shall be supported.

6.1.2 Transport mechanism. A bulk data transport mechanism shall exist within ACS.
(~~TBD~~)

6.2 Data format

6.2.1 FITS format shall be supported.

6.2.2 VOTable format shall be supported.

7 Timing signals flow

7.1 Time System

7.1.1 Standard. Array time shall be based on a standard ~~TBD~~ (UTC or TAI).


7.1.2 API. There shall be an API to access time information in ISO format (one time zone, e.g. UTC~~TBC~~). This shall make use of the Time library defined earlier.

7.1.3 Distributed timing. The distributed timing system consisting of a periodic pulse with a period of 48 millisecond. The leading edge of each pulse marks a Timing Event.

7.1.4 Services. ACS shall support attachment to the Timing Event via event synchronization.

7.1.5 Resolution. The API shall be the same for all platforms, but provides higher time resolution and performances on platforms that have specific hardware support.

8 Attributes

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 16 of 28</p>
--	--	---

8.1 Security

Security. Access to the system shall be password protected in software and firewall protected in hardware (~~normally no~~, but it is not expected that this will result in a development for ACS).

8.1.1 Tracing. Personal login with password should allow access to the system but the access of each individual should be logged for security (work for ACS TBD).

Booking. There is no requirement to try to book or protect access at component level.

8.2 Safety

8.2.1 All human and machine safety will have mechanical or other systems that will operate even in the face of malfunctioning software. Feedback to the affected software shall be provided, except in case of catastrophic failures. (implications for ACS TBD).

8.2.2 The use of software limits shall be supported by ACS. These are used to allow the software to stop potentially dangerous actions before hardware limits are reached, under normal software operation conditions.

8.3 Reliability

8.3.1 Robustness. Should a subsystem fail, ACS shall allow the rest of the system to continue operation, supporting the possibility to reconfigure without shutting down the system.

8.3.2 Backup. Backup copies of configuration Database information shall be supported.

8.3.3 Roll-back. A 'breakpoint' system with a standard roll-back should also be considered.



~~The Operational Interface shall have a Mean Time Between Failures (MTBF) of no less than 7 days (from [EVLAOps] 2.1, assumed realistic as a goal also for ACS).~~

~~The system shall be available 99.5%2 of the time (unavailability of 48 hrs over a year) (from [EVLAOps] 2.2).~~

8.4 Performance

8.4.1 Performance. A minimum goal rate of ~~1k~~1000 messages/sec shall be supported in communicating between any two CPUs of any of the considered OS/RTOS.

8.4.2 Command reception shall be acknowledged immediately, i.e. within a TBD time.

8.4.3 User replies. When actions get started from a user interfaces there shall be either a positive or an error reply returned to the user within 2 sec. For actions requiring longer time, there shall be a way to see that they are on-going within the time specified above.


8.4.4 Panel updates. Data refreshed on screens shall arrive to the user within 0.5 s and refresh rate should not be less than 10 per second. This extends to users of the Control centre, but not to remote users.

8.4.5 System down time. Restarts and reboots shall be contained within a goal maximum time of 10 minutes (for the whole ACS environment). This defines also the maximum time for remote access interruptions.

9 Standards

9.1 Standards and products

9.1.1 Official or de-facto standards shall be used whenever possible.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 18 of 28</p>
--	--	---

9.1.2 The choice of products to be used in ACS shall be based on their compliance with standards. Reference products are given below following the corresponding standards.

9.1.3 Open source products shall be preferred over commercial products, when functionally satisfactory and if they represent a real proven alternative to commercial ones. The first ones have the advantage of long term safety and portability across platforms (e.g. observing tools).

9.2 Software standards

9.2.1 RTOS. A standard real-time operating system shall be supported by ACS.

9.2.2 High level operating system. ACS shall support Posix compatible Unix. A version of ACS, perhaps with reduced functionality, shall be executable on a TBD large variety of common operating systems. A provisional criterion should be that this “ACS Lite” should run on any operating system that supports the Java Virtual Machine.


9.2.3 Compiled Languages. ACS shall primarily be implemented in JAVA and C++. Limited low level parts of ACS may be implemented in C.

9.2.4 IDL. All Distributed Object interfaces shall be written in CORBA/IDL

9.3 Communication standards

9.3.1 CORBA. Communication on LANs and remote links shall be based on CORBA. Additional transport protocols (for example, http and SMTP) will be provided for cases where the use of CORBA is not practical or necessary (e.g., for access by non-ALMA sites).

9.3.2 ORB independence. ACS shall be as insensitive as possible to the choice of ORB vendor and therefore it shall be based on POA rather than BOA.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 19 of 28</p>
--	--	---


9.4 Reference products

This section has been updated to give only reference products that are not already part of the Computing standards given in [STD].

- 9.4.1 Configuration DB. The configuration database will be based on the MicroArchive provided by the ALMA Archive Subsystem.
- 9.4.2 Analysis and plotting. Labview will be evaluated (based on WinNT).
- 9.4.3 CORBA ORB. ORBs will be chosen and/or changed as necessary to best meet the needs of the software system; in any case, no more than one ORB per implementation language will be used.
- 9.4.4 Data transfer. There shall be a feasibility check of CORBA in relation to astronomical data transfer.

10 Life cycle aspects

- 10.1.1 Prototyping. Any technology presenting a certain risk shall be prototyped before a final decision on its use is adopted. This is valid also for the initial choice of CORBA, given that suitable ORBs with appropriate performance have to be chosen.
- 10.1.2 Performance tuning. ACS shall be performance tuned on a computer model and then by using performance metrics on an existing telescope.
- 10.1.3 Methodology. The ACS software shall be developed within the same guidelines adopted by the project for the development of all other ALMA application software.
- 10.1.4 Releases. A formal Release cycle of 6 months shall be started with ACS and encompass the whole common software.
- 10.1.5 Test. Automatic regression tests shall exist for all releases.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 20 of 28</p>
--	--	---

10.1.6 Applications. Application groups will submit code of general [utility-interest/use](#) to ACS for consideration of inclusion. Developers are responsible for the maintenance of their code to be provided together with test procedures.

10.1.7 Procedures. Standard software management tools (e.g. CVS) shall be applied. The same applies to coding and documentation standards, as soon as they will be defined by the Software Engineering Team.

11 Interface Requirements

11.1 User interface

11.1.1 Portability. User interfaces shall be platform independent, so that they can be run both centrally, locally and remotely.


11.1.2 Extended portability. Packages requiring porting to platforms external to ALMA (e.g. observing tools to be run on any Laptops) shall be identified, if existing, within ACS.

11.1.3 Tools. A standard set of tools for the development of user interfaces shall be provided, including: an interactive GUI builder, a standard set of widgets, standard support libraries for the integration with applications and ACS.

11.1.4 Modularity. User interfaces shall not implement any control logic, to mark the difference with control applications, but might need to contain some interaction logic (e.g. a button being blocked while some action occurs).

11.1.5 ACS GUIs. Specific panels shall allow to display sampled data both in quasi-real-time or off-line, alarms and logs inclusive of filtering and sorting capabilities.

11.1.6 Location. All data, logs and alarms shall be available both centrally and locally, to allow use [and display](#) in the antenna area, at the control centre and at remote locations.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 21 of 28</p>
--	--	---

11.2 Hardware interfaces

11.2.1 Hardware interfaces. These will be provided as part of the M & C software development and will be integrated into the ACS releases.

11.3 Software interfaces

11.3.1 The ACS shall be integrated and tested with any common open-source and commercial software, off-the-shelf software, legacy and embedded software used for ALMA. [This includes Operating Systems, CORBA ORBs, Tools, Libraries etc.](#)

12 ACS Design requirements

12.1.1 Distributed Objects and Commands. Commands shall be implemented via remote method invocations of Distributed Objects, based on the CORBA technology .

12.1.2 Standard methods. There shall be a list of standard methods for Distributed Objects that is supported by ACS.


12.1.3 Events. ACS should allow attaching callbacks to events.

12.1.4 Configuration. It shall be possible to use ACS in a development set-up and in the real control situation (e.g. control model and real telescope), even in parallel.

12.1.5 Modularity. It shall be possible to build only the modified components of the system, without having to rebuild the entire system.

12.1.6 Portability. ACS shall be designed to be portable across the RTOS, Unix and Linux platforms for what concerns its upper level components. There must be parts though which are only applicable to the RTOS.

12.1.7 ~~Not all features need to be supported for all programming languages; ...~~ [There might be restrictions on ACS features available for certain programming languages, according to the needs of the corresponding application domain.](#)


	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 22 of 28</p>
--	--	---

13 Requirements for applications

This is a temporary section, meant to be replaced later by a style guide document.

13.1 Style guide requirements

- 13.1.1 Logging of commands. All user commands shall be logged (to correlate in time relevant events to user activities).
- 13.1.2 Normal commands. They shall not assume deep knowledge of system and cannot lead to situations requiring recovery actions.
- 13.1.3 Maintenance commands. Not required during normal operation. Require deep knowledge of the subsystem to which they apply and might need a certain sequence of actions. Normally to be used by maintenance staff only.
- 13.1.4** All subsystems shall be monitored in a standardized way.
- 13.1.5 Reliability. System reliability shall be improved by implementing error recovery whenever reasonable. A trace of the recovery attempts shall exist in the logs, even when recovery succeeded.
- 13.1.6 Subsystems shall be monitored when in operation to notify malfunctions (with alarms) when they occur.
- 13.1.7 Warnings. The number of low severity alarms (warnings) shall be small during normal operation.
- 13.1.8 Disabled subsystems. It shall be possible to globally inhibit actions on part of the ALMA software.
- 13.1.9 Dynamic configuration. It shall be possible to change all parameters dynamically without resetting the ALMA software.
- 13.1.10 Simulation. Control software must be able to run in simulation mode, without requiring any normally associated hardware

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 23 of 28</p>
--	--	---


- 13.1.11 Stand-alone. Subsystem software must be able to run in stand-alone mode, meaning without the rest of the ALMA system (e.g. sub-reflector being tested without antenna). This might require that other control software (interfaced with the considered subsystem) runs in Simulation.
- 13.1.12 Additions. Existing software, whose interfaces don't change, shall not need to be modified when new software components are added to the existing system.
- 13.1.13 States. Distributed Objects may have states (like off, standby, online). When States are used they should have standard names and meaning, with standard methods supported by all objects.

14 Container/Component Requirements

The purpose of the Container/Component model is to facilitate the separation of technical from functional concerns in the software, allowing application developers to concentrate on the latter.

14.1 Components

- 14.1.1 Components, in the meaning given by the ALMA Software Architecture Document, will be supported. They will hide the details of the underlying CORBA mechanism from the application developer, except where access to these details is necessary to meet an application's requirements.
- 14.1.2 Lifecycle Interface: components will be controllable via a standardized lifecycle interface that includes 1) initialization; 2) start of execution; 3) update; 4) orderly shutdown; and 5) immediate shutdown.
- 14.1.3 Service Interface: components will publish one or more service interfaces.
- 14.1.4 Deployment: components may be deployed at run-time and redeployed while the system is running.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 24 of 28</p>
--	--	---


14.2 Container Services

- 14.2.1 Lifecycle: the Container will be the unique means by which components are managed.
- 14.2.2 Languages: Container versions will be available to manage components written in C++, Java or Python. It is acceptable to have a different container for each different language.
- 14.2.3 Logging: the Container will provide components with access to the logging facilities of ACS.
- 14.2.4 Archive: the Container will mediate access to the Archive subsystem and/or to the MicroArchive and will do so in the same way for each.
- 14.2.5 Tight Containers: in general, Containers will mediate all accesses to their components' service interfaces.
- 14.2.6 Porous Containers: where performance considerations demand, Containers will provide only the initial reference to a component's service interface, allowing direct access thereafter.

14.3 Entity Objects

Entity objects are generally complex data structures that need to be accessed by more than one subsystem. Examples include Scheduling Blocks, Observing Projects, and Correlator Configurations.

- 14.3.1 Definition: entity objects will be defined via XML schemas.
- 14.3.2 Serialization: facilities will be provided for simple serialization of entity objects for network transmission and/or persistent storage.
- 14.3.3 Automatic generation of binding classes from these schemas will be provided for Java; tools and procedures for parsing serialized entity objects will be provided for C++ and Python.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 25 of 28</p>
--	--	---


15 Applicable eVLA Requirements

The requirements in the next sections are being introduced in Version C of this document and are taken from the eVLA (see [EVLAOps] and [EVALEng] references).. While only requirements that seem applicable are given here and they normally give more details than available before, there is surely some new scope. So their implementation should be taken as a goal only, compatibly with existing priorities and planning.


15.1 Device Browser Tool

The following requirements are extracted from [EVLAEng] 1.1R1-R8 (with priority 1). They extend the previous requirement 11.1.5.

- 15.1.1 There shall be a Device Browser Tool, with the capability to interact with ALMA devices. Given the similarity with the ACS Object Browser this requirement and the ones linked to this should be seen as possible upgrades of the existing browser.
- 15.1.2 The Device Browser Tool, upon connection to a module, shall display labeled numeric data from each monitor point in the device. Multiple pages of such displays may be needed for devices with many monitor points.
- 15.1.3 The Device Browser Tool shall have a display consisting of a plot of a selected monitor point against time, with automatic scaling of axes and appropriate labels. The plot shall be empty when started, and will grow as data arrives.
- 15.1.4 The Device Browser Tool shall have a display consisting of a list of all commands that can be sent to the device.
- 15.1.5 The Device Browser Tool shall support sending any of these commands to the device, with data entry as numeric, hexadecimal, logical, or text as appropriate.

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 26 of 28</p>
--	--	---

- 15.1.6 The first request to send a command should require some security check (password or other).
- 15.1.7 These commands shall include setting of monitor point sampling rates, separately for each monitor point. The fastest allowed sample rate for this tool is 48 msec (the standard ALMA timing tick). When sending a command to change a monitor point sampling rate, it should also be required to set the time interval over which this change is requested. The default should be that the change is implemented until the Device Browser Tool is exited.
- 15.1.8 It shall be possible to send a command while a display is active.
- 15.1.9 It shall be possible to send a command repetitively with a specified repetition rate.
- 15.1.10 The Device Browser Tool shall operate if it is connected to the same ethernet subnet as the computer controlling the device. No other computer or network activity shall be required for the Tool to function.
- 15.1.11 The Device Browser Tool shall operate if connected to other subnets than the one to which the device computer is connected. Appropriate provisions for security and for transportation of the data shall be provided.
- 15.1.12 The Device Browser Tool shall have the capability of writing a named file on its local computer while running the display specified above. The file will consist of either blank or comma separated fields, the first of which will be the time (in MJD and decimal fraction), and a subsequent column for each of the monitor points. The first row of the file will be the monitor point labels.
- 15.1.13 The default behavior shall be that rows will be written at the rate of the most rapidly updating monitor point, with repeated values for monitor points which update more slowly.
- 15.1.14 It shall be possible to interrupt the file writing during data recording, gracefully, without exiting the Tool completely (i.e., execute a “stop writing to file now” command).

	<p>ALMA Project ALMA Common Software Technical Requirements</p>	<p>Doc # : COMP-70.25.00.00-0004-C-SPE Status: Draft Date: 2005-09-26 Page: 27 of 28</p>
--	--	---

15.2 Online Support

A help facility should be incorporated into the design of the software system from the start and it should allow the array operator to find the information he/she seeks quickly and accurately. At a minimum, it should include a table of contents, an index, full-text search and context sensitive help (taken from [EVLAOps] 1.4 and 1.6).

15.2.1 The system shall provide an online help facility with the following features: table of contents, index, full-text search

15.2.2 The user shall have the ability to capture and print any display.

15.3 Plotting

A general-purpose plotting component will be needed. Several of the standard M&C displays will likely contain plots of monitor points relevant to that display and a user must have the ability to plot any monitor points on the fly (taken from [EVLAOps] 3.5). Hopefully no specific development will need to be done by ACS on this apart from evaluation work (see also requirements 5.1.4 on Analysis tool and 9.4.2 on LabView evaluation).

15.3.1 The user shall be able to generate the following plot types: scatter plot, histogram, line plot

15.3.2 The user shall be able to view the plot dynamically (real-time) or statically (offline).

15.3.3 The user shall have the ability to give the plot a title.

15.3.4 The user shall have the ability to define the plot axis labels in the following manner:

15.3.5 user defined character string, import predefined labels from a monitor point definition database

15.3.6 The user shall have the ability to specify the number of major and minor tick marks for a plot.



- 15.3.7 The user shall have the ability to view multiple monitor points on a single axis.
- 15.3.8** The user shall have the ability to specify whether the major and minor tick marks are: Linear, logarithmic
- 15.3.9 The user shall have the ability to auto-scale plots.
- 15.3.10 The user shall have the ability to specify the scale of the plot for all axes.
- 15.3.11 The user shall be able to plot any monitor point in the system.
- 15.3.12 The user shall be able to plot the same monitor point for all antennas or selected antennas.
- 15.3.13 The user shall have the ability to infinitely zoom in and out of the plot.
- 15.3.14** The user shall be able to select various point styles: None, dots, points, shapes (triangles, squares, circles, etc.)
- 15.3.15 The user shall be able to show error bars.
- 15.3.16 The user shall have the ability to view the plot's legend.
- 15.3.17 The user shall be able to print a hardcopy of the plot.
- 15.3.18** The system shall identify the following characteristics of a monitor point over a specified time range: minimum value, maximum value, average, RMS
- 15.3.19 The user shall have the ability to set maximum and minimum limit markers.

