



Atacama Large Millimeter Array

ACS Software Development Plan


COMP-70.25.00.00-001-C-PLA

Version: C

Status: Final

2005-07-29

Prepared By:		
Name(s) and Signature(s)	Organization	Date
G.Chiozzi	ESO	2005-07-29
Approved By:		
Name and Signature	Organization	Date
Released By:		
Name and Signature	Organization	Date

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 2 of 37
--	---	--

Change Record

Version	Date	Affected Section(s)	Change Request #	Reason/Initiation/Remarks
0.9	2002-04-12			First revision
0.9-Rev.1	2002-04-18			Updated with comments from G.Raffi and extended planning sections
0.9-Rev.2	2002-05-10			Updated with comments from G.Raffi and B.Glendenning Updated dates of releases and corresponding tables.
0.9-R3	2002-05-24			Revised with final comments from B.Glendenning, G.Raffi, R.Lucas, J.Schwarz
1.0	2002-06-06	1.8, 2.2, 2.3, 3		Correct reference (offline), Glendenning is contact, HIA instead of Penticton, remove a bit of boilerplate.
1.1	2002-11-22			Updated for ALMA Internal Design Review
1.2	2003-02-14			Updated after ALMA Internal Design Review
2.0	2003-08-08			Updated after ALMA CDR-1
A	2004-05-31			Applied new ALMA template. Updated for CDR-2
B	2004-05-13			Updated for CDR-3
C	2004-07-29			Updated after CDR-3


	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 3 of 37
--	---	--

Table of Contents


1	PROJECT OVERVIEW	5
1.1	Purpose and Scope	5
1.2	Assumptions and Constraints	6
1.3	Deliverables	6
1.4	Schedule and Budget Summary	7
1.5	Document Scope	8
1.6	Document Evolution	8
1.7	Glossary	9
1.8	Applicable documents	9
1.9	References.....	10
2	PROJECT ORGANIZATION	10
2.1	Organization Interfaces	10
2.2	Group Organization.....	10
2.3	Roles and Responsibilities	11
3	MANAGERIAL PROCESS PLANS.....	13
3.1	Start-up Plan	13
3.1.1	Estimates	13
3.1.2	Staffing	14
3.1.3	Staff Meetings and Training	15
3.2	Work Plan	15
3.2.1	Work Breakdown Structure	15
3.2.2	Schedule Allocation.....	16
3.3	Project Tracking Plan.....	16
3.3.1	Requirements Management	16
3.3.2	Progress Control	17
3.3.3	Quality Control.....	17
3.3.4	Communication and Reporting.....	17
3.4	Risk Management Plan	17
3.5	Project Termination Plan.....	18
3.6	Intellectual Property	19
4	TECHNICAL PROCESS PLANS	19
4.1	Development Model.....	19
4.2	Methods and Tools.....	21
4.3	Infrastructure.....	21
4.4	Acceptance Plan.....	24
5	SUPPORTING PROCESS PLANS	24
5.1	Configuration Management.....	24
5.2	Validation	25
5.3	Documentation	25
5.4	Quality Assurance	25



ALMA Project
ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA
Date: 2005-07-29
Status: Final
Page: 4 of 37

5.5	Reviews.....	25
5.6	Problem Resolution.....	25
5.7	Process Improvement	25
6	SOFTWARE DESCRIPTION	26
6.1	Software External interfaces	26
6.2	Software Design.....	26
6.3	Packages	26
6.4	Interfaces between Packages	26
6.5	Integration of Packages	26
7	ATTACHMENTS	27
7.1	Detailed Planning.....	27
7.2	Requirements Compliance Table	29

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 5 of 37
--	---	--

1 Project Overview

This document contains the ALMA Common Software Development Plan for phase 2 (construction phase), limited to the first 5 years of development ending mid 2007.

The IEEE Std. 1058-1998, IEEE Standard for Software Project Management Plans has been taken as reference in writing this Plan.

1.1 Purpose and Scope

The ALMA Common Software (ACS) is part of the System layer of the ALMA Architecture [Architecture].

ACS is located in between the ALMA application software and other basic commercial or shared software on top of the operating systems and provides a generalized common interface between applications and the hardware in order to facilitate the implementation and the integration in the system of new hardware and software components.


ACS provides basic software services common to the various applications (such as antenna control, correlator software, data pipelining) and consists of software developed specifically for ACS and as well of Operating System (OS) builds and commercial device drivers.

ACS is designed to offer a clear path for the implementation of applications following the programming model defined in the ALMA Architecture [Architecture], with the goal of obtaining implicit conformity to design standards and maintainable software. The use of ACS software is mandatory in all applications, except when the requested functionality is not provided by ACS. Motivated exceptions (for example based on reuse considerations) have to be discussed and approved on a case-by-case basis.

The main users of ACS will be the developers of ALMA applications. In the performance of routine maintenance operations, operators and maintenance staff will also use the generic tools and GUIs provided by ACS to access logs, Configuration Database, active objects and other components of the system.

ACS is based on the Common Object Request Broker Architecture (CORBA) middleware but provides many more features that facilitate subsystem communication, error handling, data collection etc.

For a comprehensive summary see the ALMA Common Software Architecture document [ACS].

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 6 of 37
--	---	--

1.2 Assumptions and Constraints

All the standards and products defined in the Computing Plan for Phase 2 [Plan] and [Practices] are fully applicable.

All the work done by ALMA Computing in Phase 1, and in particular the ACS development, is a prerequisite and applicable to Phase 2. This is included in particular in documents defining Requirements [Requirements], Analysis and Design [Architecture] and ACS [ACS].

The Software Engineering practices as defined in [Practices] are also fully applicable.

It is expected that all subsystem teams will provide requirements on ACS, drive the ACS development with suggestions and provide feedback concerning the capability of ACS to cover their development needs. In particular Pipeline and Offline Data Reduction are expected to work in close collaboration with the ACS team to identify the requirements, implement and test the features needed for their specific applications.

It is also expected that subsystem teams will develop new components of ACS or enhance already existing components. Such enhancements will be integrated into the core of ACS, if generally useful. They will become in this way the full responsibility of the ACS team, but the original author should be available for maintenance.


It is expected that subsystems requiring new feature for ACS will test them on ACS pre-releases to warranty that the code officially released meets requirements and to allow a first bug verification.

1.3 Deliverables

All the items described in [Plan] for the various groups of categories (namely Software, Documentation, Computer equipment, Support time) are applicable to ACS.

The main deliverable of ACS at each Release is an ACS Installation Package, available online from the ACS Website (see next paragraph) in various formats, for example:

- Tar file with pre-built binary installation for Linux
- Tar file with binary basic tools and sources for the ACS packages
- RPM packages for Linux distribution
- VmWare complete Linux installation
- CVS download with build procedures, including basic tools with (eventually) patches.

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 7 of 37
--	---	--

The set of downloadables includes a complete documentation set for each of the supported hardware (HW) and software (SW) platforms.

The different downloadable formats are made available to cover the needs of the different development teams (easy and quick installation, customizable installation starting from sources...). Other formats will be made available upon request.

Given the pervasive nature of ACS, whose usage is required for the development of every ALMA subsystem, particular attention has to be given to support.

We have therefore the following additional deliverables:

- **ACS Website**
The ACS Website is the main point of access to ACS documentation and information and provides:
 - Online documentation
 - ACS software download area
 - Support areas (Frequently Asked Questions, ACS Newsgroup, Support and Discussion Group, Search Engine)
 - Design and planning discussion area (ACS Wiki)
- **Training and support time**
Each Alma software developer needs training and support on ACS. The ACS team will provide training and support in the form of:
 - Follow up of problem reports and ACS newsgroups
 - Tutorials and presentations
 - Training on the job opportunities.
 - Code inspections on other subsystems based on ACS

1.4 Schedule and Budget Summary

The **total effort** allocated for development of the ACS software is 23.5 Full Time Equivalents (FTEs) for the 5-year project described here. This corresponds to a yearly effort of 4.7 FTEs, split as 2.7 Europe and 2 North America.

An additional Japanese contribution of 3Fts (1 per year from 2005) is needed to cover the increased support burden and specific development. Another 0.3Fts per year will be provided in Japan for on site support.

The **material construction costs** are based on the development infrastructure described in section 4.3. If we take into account the material already bought in phase1 and in 2003-2004 we have:



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 8 of 37

- 10 Linux development workstations (already available, some shared with Integration and Testing for control model and old ACS version). Some are old PCs and will need to be upgraded in the coming years.
- 1 “VmWare factory” is available in Garching to provide multiple virtual machines to cover testing, build/distribution and training needs. This machine can host multiple virtual machines running in parallel and with different hardware and software configurations.
- 2 Linux workstations for real time development with RTAI and VxWorks (1 already available in Garching, complete with CAN board, **one still missing in Socorro**)
- 2 Linux VME Crates/LCUs for real time Linux development (1 already available in Garching, complete with CAN board and Parallel IP carrier, **one still missing in Socorro**)
- 3 Sun workstations (already available in Garching and Socorro). These are needed to run VxWorks tools and utilities not available under Linux.
- 2 LCUs for VxWorks development (already available in Garching. **We can share with CONTROL in Socorro**).

When porting of TICS and Control to RTAI will be completed, we will not need any more the Sun workstations and the VxWorks LCUs.

The **travel cost** is according to [Plan].


1.5 Document Scope

This Plan represents the terms of a technical agreement between the ALMA Computing Group and ALMA Management. It is subject to configuration control as soon as reviewed. Changes are subject to approval by the ALMA Change Control Board.

1.6 Document Evolution

The structure of this document is in compliance with the recommendations of IEEE Std 1058-1998.

This Plan has been produced as Version 1 for time T0 (see 4.1). It has been revised for Internal Design Review (IDR) (Version 1.1). It has been upgraded for Preliminary Design Review (PDR) (Version 1.2) but no new version has been released with changes after PDR. This same version has been used as the base discussion document for Critical Design Review 1 (CDR-1). Version 2.0 has been prepared with the feedback from CDR-1. Version A and has been prepared for CDR-2, according to the numbering scheme adopted officially for ALMA

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 9 of 37
--	---	--

documents and applied to this document for CDR-2. The document will be upgraded yearly for each CDR. This is version B, prepared for CDR-3

1.7 Glossary

A complete list of ALMA software terms can be found in [Glossary]

A number of key terms, although contained in [Plan] are also retained here for reading clarity.


<i>SDG</i>	<i>Software Development Group, responsible for a Subsystem</i>
<i>CDR 1-3</i>	<i>(Incremental) Critical Design Review 1 to 3.</i>
<i>FTE</i>	<i>Full Time Equivalent (staff-year)</i>
<i>ICD</i>	<i>Interface Control Document</i>
<i>IDR</i>	<i>Internal Design Review</i>
<i>Package</i>	<i>Major component of Subsystem</i>
<i>PAR</i>	<i>Preliminary Acceptance Review</i>
<i>PDR</i>	<i>Preliminary Design Review</i>
<i>RR</i>	<i>Readiness Review</i>
<i>R0-5</i>	<i>(major) Release of software (and its number)</i>
<i>Subsystem</i>	<i>Subsystem of the ALMA Software System</i>

1.8 Applicable documents

The documents referenced are fully applicable to the ACS development effort.

- [1] [Plan] ALMA-SW-Draft, ALMA Computing Plan, G.Raffi, B.Glendenning, ALMA-70.05.00.00-001-J-PLA
- [2] [SSR] ALMA Science Software Requirements and Use Cases, Robert Lucas et al., ALMA-70.10.00.00.00-002-I-SPE
- [3] [Architecture] ALMA Software Architecture, J.Schwarz et al., ALMA-70.15.00.00-001-H-GEN
- [4] [Practices] ALMA Software Engineering Practices, M.Zamparelli, ALMA-70.20.00.00-003-A-PRO
- [5] [ACS] ALMA Common Software Architecture, G.Chiozzi et al., COMP-70.25.00.00-002-A-DSN
- [6] [ACSReq] ALMA Common Software Technical Requirements, G.Raffi, B.Glendenning, J.Schwarz, COMP-70.25.00.00-003-A-SPE

Comment: To be updated just before official release

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 10 of 37
--	---	---

[7] [Glossary] ALMA Software Glossary, COMP-70.15.00.00-003-A-GEN

1.9 References

ACS design documents, interface control documents, user and maintenance manuals, test procedures, plans are listed and downloadable from the ACS Web Page:

<http://www.eso.org/projects/alma/develop/acs/>

In particular, the ACS Online Documentation page always contains specification, design documents and manuals for the actual release:

<http://www.eso.org/projects/alma/develop/acs/OnlineDocs/index.html>

2 Project Organization

2.1 Organization Interfaces


The ACS team works in close collaboration with the High Level Analysis and Design team and with the Software Engineering team.

The High Level Analysis and Design team defines in collaboration with the ACS team the Technical Architecture and the Programming Model for the whole ALMA Project, based on the ALMA SW Requirements and on the ALMA Functional Architecture. The ACS team is then responsible for the implementation of the components allocated to ACS itself.

The Software engineering team defines in collaboration with the ACS team the standards and the SW engineering practices to be used in ALMA. This has a direct impact on the implementation of ACS both in terms of external packages integrated in ACS and of true ACS development.

2.2 Group Organization

Figure 1 shows the organization of the software development group for the ACS Subsystem. The Contact Person for the ACS Subsystem is [B.E.Glendenning].

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 11 of 37
--	---	---

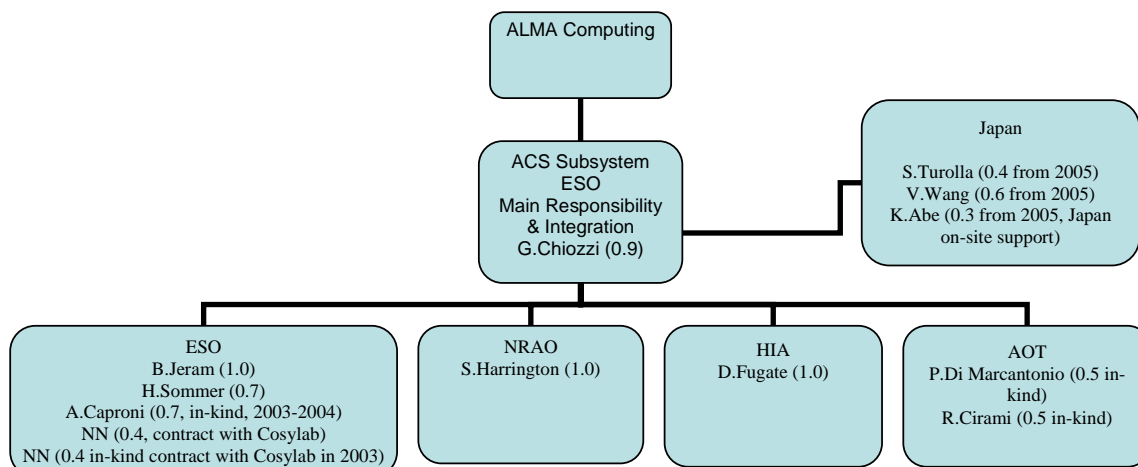


Fig.1 : Organization overview of the ACS Software Subsystem

2.3 Roles and Responsibilities

The ACS Development Group is described in the following table.

Project main responsible: G.Chiozzi (total 4.5 FTEs=at 0.9 FTEs/year), ESO, TEC/SW Project coordination, planning, quality control and development in C++ and Java areas
Development staff at ESO for ACS (total 9 FTEs=2.1 FTE/year): The ESO team is principally responsible for the development of the core ACS components and for ACS integration and release. This includes the ACS common services (see [ACS]) and the Java Component Based framework (see [Architecture]). B.Jeram at 100%, Mainly responsible for C++ development and testing. Whole responsibility of LCU/VxWorks ACS parts. Main packages: C++ Component/Container, Error System. Testing H.Sommer at 70%: Development of ACS Java Component Container NN1 at 40% (for 2003 and 2004 NN1 correspond to a 0.4 FTE consultancy and development contract with Cosylab. Main responsibilities are ACS Manager, C++ Component/Container, Configuration Database, Java GUI Components).
Development staff at NRAO for ACS (total 5 FTEs=1.FTE/year): D.Fugate in 2003, S.Roberts 1 st Q 2004 (50%), S.Harrington starting 2 nd Q 2004 at 100%.



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 12 of 37

The NRAO team is principally responsible for the ACS packages specifically designed for the Control System, Pipeline and Scheduling subsystems and for supporting the respective development teams.

Development staff at HIA for ACS (total 4 FTEs=1.FTE/year starting 2004):

D.Fugate at 100%.

The HIA team is primarily responsible for higher level ACS framework components and for Python ACS packages to support to data reduction and data flow applications. D.Fugate is responsible for the Notification Channel implementation, the Python Component Container development and support on all platforms and languages (C++, Java and Python)

Japanese contribution. Staff at ESO. (total 3FTEs=1FTE/year starting 2005)

S.Turolla starting 05 2005 at 40% : ACS development related to system management, configuration, platforms, deployment and packaging. Testing

V.Wang starting 07 2005 at 60% : ACS support to Japanese staff, development related to Japan extensions.

K.Abe starting 07 2005 at 30% : ACS on-site support in Japan. Main ACS reference person in Japan. No development responsibilities.

We assume that during the first two months the new staff will be busy mainly with training activities and that the whole team will have additional training costs, partially compensated by training on the job tasks.

In general each ACS team member is expected to bear part of the additional load to provide support to the Japanese ALMA developers. This will allow the two ACS team members financed by Japan to work also on activities not directly related to Japan specific features but of general ACS scope.

For 2003, 2004 and 2005 the ACS team can count on In-Kind contributions to cover the peak development foreseen:

In-Kind development staff at ESO for ACS in 2003, 2004 and 2005:

A.Caproni at 70% in 2003, 2004 and 50% in 2005: Support to Java development, documentation and testing

In 2003 we had 0.4 FTE consultancy and development contract with Cosylab, on top the 0.4 FTE in the previous table.

In-Kind development staff at AOT (Astronomical Observatory of Trieste) for ACS in 2003, 2004 and 2005: 1 FTE

The AOT is providing In-Kind development resources for C++ ACS development



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 13 of 37

in areas related to control system: optimization in the implementation of properties, sampling system, bulk data.

In the past years, in kind contributions have been used to:

- Compensate for resources officially foreseen but not actually allocated, for example because of delays in hiring procedures.
- Compensate for missing contributions originally foreseen as training on the job activities from other subsystems
- Deal with scope increase.

For a more detailed discussion about resources allocation see section 7.1.

3 Managerial Process Plans

This part specifies the project management processes for the Subsystem, for what is additional or different for this Subsystem with respect to [Plan].

3.1 Start-up Plan

3.1.1 Estimates

All Global activities and Subsystems have been estimated by the Computing group to assess the cost of Phase 2. Therefore they are fixed in terms of both effort and material cost. The Main Milestones are derived from the ALMA Phase 2 planning.

The scope of ACS has been widened in the last part of 2002 as a consequence of the work done by the Analysis and Architecture team. Another increase in the scope of ACS took place in 2004 based on the requests from the Pipeline and Data Reduction groups.

The new ACS features requested by the Analysis and Architecture team have had as a consequence in 2003 an increase of about 25% in ACS manpower estimates. As described in previous versions of this document, this has been counterbalanced by the ACS work done in phase 1, which can be estimated in about 25% of the total work (about 5 FTEs of work done in Phase 1 over 13.5 planned for Phase 2).

The work needed to properly support Pipeline development from 2003 has been put into the planning by de-scoping or delaying other activities originally planned for these years in previous versions of this document and by using some in-kind contributions. We expect also a direct participation of the Pipeline team in the design development of specific features as collaboration with the ACS team/

We consider essential for ACS to keep the development plan flexible and able to react to the requests of the other subsystems. The way to reach this objective is to discuss with all subsystems the plan of each ACS release immediately after the



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 14 of 37

previous release, in order to collect feedback and set priorities. This has been done for all ACS releases until now with very good results.

Also, we have to take into account that the time that needs to be allocated for support was initially largely underestimated. Actually the estimates done by the Computing group were based on the development effort, without taking into account the specific needs for ACS of providing extensive support and training to the other subsystems. According to our statistics we have been spending about 20% of ACS manpower in support and training in 2003. In 2004 the actual support time has been around 15%, but has increased to about 30% in the last part of the year for the R2 integration. In the first part of 2005 went again around 20% that is therefore a good steady state figure. To that we have to add training in the form of ACS courses for another 5%.

In 2004 we had also increased the time allocated to ACS documentation to about 25%. Unfortunately in practice we have been very far from this figure because development of features and support always get higher priority. In the development of ACS 4.1, Documentation work has taken less than 5% of the time.

Statistics are available in the ACS Planning Wiki and in the ACS weekly meeting reports.

We had originally planned to partially absorb this training and support time with tasks assigned in on-the-job-training activities and with contributions to ACS from other teams, and in particular from the Architecture and Analysis team for the development of the Java Component Based architectural framework.

The work done by the Architecture and Analysis team has satisfied our expectations.


On the other hand, we have seen that the effectiveness of the training assigned tasks with respect to the time spent by the ACS team members to train members of other teams is very low. We have not been able to compensate for training costs in this way.

ACS training and support remains an area of risk and only training on the job periods of not less than 3 weeks would help in absorbing this effort.

Another failure has been the idea of having ACS contact points in the subsystems. This has never worked has a first contact point. Support goes always directly to the geographically nearest ACS team member.

On the other hand the support by means of the ACS mailing lists seem pretty efficient.

3.1.2 Staffing

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 15 of 37
--	---	---

The ACS core development team has to be formed by members almost 100% allocated to the ACS project.

Part of the development work is also done by external contractors with a specific expertise in the design and development of CORBA Distributed applications.

Members of other ALMA teams assigned to ACS for training periods will be considered for that amount of time full ACS members and will work at least 50% on ACS assignments for that period.

3.1.3 Staff Meetings and Training

Each ACS staff member will spend an extensive initial training period in Garching (minimum 6 weeks).

The ACS team gives high value to good communications. Remote team members will be constantly in contact with the other members using Internet technology: instant messaging, email, newsgroups and web collaborative development tools.

A weekly phone/video conference status meeting is used to assess the status of the project and to synchronize activities.

Important analysis and design periods will include face-to-face workshops. At least one ACS all hands workshop per year will be organized in addition to the yearly Computing staff meeting.

Joint workshops with other teams (and in particular the Architecture and Analysis team) will be held to analyze new requirements for ACS coming from their development work.

3.2 Work Plan

3.2.1 Work Breakdown Structure

The ACS package structure is defined in [ACS] and identifies the Work Breakdown Structure. Version A of the document includes also packages associated to the features for ACS coming from [Architecture].

Some details on resource allocation are provided in section 7.1.

The following table lists the foreseen resource allocation per package in the period 2003-2007. The resources allocated to the software package listed in [Architecture] include time for design, development, testing and documentation

Package	
Management activities	0.5
Coordination	2
Support training	6
ACS Integration	1.4
Base tools	1



ALMA Project
ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA
Date: 2005-07-29
Status: Final
Page: 16 of 37

ACS component	1.2
Configuration Database	0.5
Event and Notification system	0.7
Error System	0.6
Logging System	0.4
Time System	0.2
ACS Container	1.7
Serialisation Plug	0.5
Archiving System	0.2
Command system	0.7
Alarm System	1
Sampling	0.5
Bulk Data	1.1
UIF libraries	0.6
Scripting	0.4
ACS Framework	1.4
ACS Installer	0.5
Integrated APIs and tools	0.4
TOTAL	23.5

3.2.2 Schedule Allocation

The software development model is incremental and based on a 6 month Releases cycle, as explained later. These follow the design work started with the Preliminary Design Review and are followed by the detailed design (for the various Releases) done in a number of incremental Critical Design Reviews.

Given the strong dependence of all other subsystems on ACS, the deadline for Release will be followed strictly. Eventually, slips in planning will be absorbed by de-scoping the content of a release, so that the missing features will have minimal impact on the development of dependent subsystems.

3.3 Project Tracking Plan


ACS project control shall be organized as explained below.

3.3.1 Requirements Management

ACS requirements are defined in [Architecture] and [ACSReq].

The features described in this document (in the various versions) and in the ACS architecture [ACS] and detailed design documents should be tracked back to requirements via a Requirements Compliance table, given in section 7.2.

The formal requirements will be periodically revised by the HLA team and by CIPT management, at least once every two Release cycles to allow steering ACS

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 17 of 37
--	---	---

development in phase with the evolution of ALMA development. The ACS planning meetings for each release can also trigger changes in the ACS formal requirements.

The ACS and HLA teams collaborate tightly together to determine how ACS can support in the best way the overall ALMA Architecture and what is the impact on ACS (both in terms of features and planning) of choices done by the HLA team. The primary formal document used by ACS for this purpose is [Architecture].

Proposed new requirements and changes to existing requirements will be indicated together with their impact in the ACS development plan.

Change requests not affecting science requirements will be filed via the Software Problem Reporting system and handled within the Computing Group (see [Practices] for more information).

Changes affecting science requirements will be discussed with the Science Software Requirements Committee (SSR). The ALMA Change Control Board will be involved if changes affect ALMA system functionality or performance.

New requirements, change requests and priorities will be mapped into the ACS development plan in two ways:

- When this document is updated for each CDR.
- At each ACS release planning meeting, before the development of each ACS release is started. These meetings involve the leaders of each subsystem and are used to define the detailed content and scope of each ACS release.

3.3.2 Progress Control

See [Plan]

3.3.3 Quality Control

See [Plan]

3.3.4 Communication and Reporting

See [Plan]

3.4 Risk Management Plan

See [Plan].

In addition to what is specified in [Plan], the estimate of time allocated for ACS support was considered a risk. This time had been considerably underestimated for Phase 1. From the statistics collected since 2003, we see that ACS Support and Training have stabilized around the 20% of the workload of the ACS team. This is the amount that we consider now when preparing the ACS plan for each release.



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 18 of 37

Still, Documentation is an area of risk and concern. At CDR 2 we have pointed out that the time we should have allocated for ACS Documentation had been underestimated. We have therefore decided to increase this time from 20% to 25%.

In the second part of 2004 we have made important progress with the documentation.

But during the development of ACS 4.1 unfortunately we have been in practice very far from the 25% figure because development of features and support always got higher priority. In the development of ACS 4.1, Documentation work has taken less than 5% of the time. This is a worrying figure and we will have to be strict on the 25% for the next development cycles.

At CDR 2 we have stressed the importance of having ACS contact persons in the subsystems. They have been formally nominated, but they are in practice not having any practical role: requests for support go directly to the ACS team. The most popular form of support is through the geographically nearest ACS team member. We think this is efficient for the subsystems on the short time scale, but on the long time scale it limits the growth of knowledge of ACS inside the subsystems. Despite this consideration and seen the practical situation, we think that it does not make any more sense to implement the ACS contact person.

We strongly recommend that new ALMA SW developers be trained on ACS with an intensive course and a period of work in the ACS team. During this period they would receive small but significant ACS development assignments. This work would be performed under the supervision and with the support of an ACS team member. Until now we had at least one major ACS training each year, but only few developers have taken advantage of the opportunity of training on the job periods to do some work together with the ACS team. We still think this would be a very good way of learning how to use ACS and of providing valuable feedback.


ACS performance in critical areas (and in particular bulk data transfer and notification channel) needs to be verified and monitored. Performance measures have been performed with ACS 3.1 and the development of a standard performance measurement has taken place in ACS 4.0. Still we had no time to allocate for ACS 4.1 to make an extensive performance measurement campaign with the tools implemented with ACS 4.0. This is an potentially risky area and therefore we need to do it as soon as possible.

For more details see the ACS performance pages on the ACS Wiki:

- <http://almasw.hq.eso.org/almasw/bin/view/ACS/AcsPerformances>

3.5 Project Termination Plan

See [Plan]

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 19 of 37
--	---	---

3.6 Intellectual Property

In addition to what has been specified in [Plan] we have to take into account that some core components of ACS (BACI, MACI and ABeans) [ACS] are derived from a previous development by the JSI institute in Ljubljana and are now developed in collaboration with this institute and the Cosylab company under a consultancy contract.

All ACS software is therefore covered by LGPL license with a parallel special license agreement with JSI and Cosylab to allow more flexible usage of the code for these specific components.

4 Technical Process Plans

4.1 Development Model

See [Plan]

The software development model for ACS is incremental and based on a 6 month Release cycle. Given that this model is described in [Plan] only the Milestone names and their due dates are given below.

In addition to what is described in [Plan] we have to take into account that all subsystems depend on ACS. Therefore the global ALMA plan has to be aligned in such a way that a new Release of ACS is available to the subsystem teams at the beginning of their own development cycle, i.e. just after (Incremental) Subsystem Integration Release. This includes the integration of the Archive in the ACS release.

In this way, the first step of each new subsystem development cycle will consist of installing the new ACS, porting the existing code to it and analyzing the new features provided by the ACS Release.

Afterwards the ACS Release will be stable for the whole development cycle of the subsystem.

This is possible because ACS development was started well in advance during Phase 1 and ACS has already completed many development cycles.

<i>Subsystem Start (T0): 2002-06-01</i>
--

2002-06-01 is the official Project Start date, but we can consider the work for ACS already started in Phase 1. Therefore the dates for PDR and R0 do not follow the same rules that apply to the other subsystems.

<i>Internal Design Review (IDR): 2002-12-01</i>
--

Main activity for IDR has been the update of the existing ACS Architecture document [ACS] based on the experience accumulated in Phase 1 and to extend it to take into account [Architecture].
--



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 20 of 37

Preliminary Design Review (PDR) (T0+8 month): 2003-03-15

The ACS Architecture [ACS] has been updated based on the results of IDR and on the feedback from the other subsystems that have started their development activity, in particular using the new prototype for the Java Component/Container Mode specified in [Architecture].

(Incremental) Release (R0,5)

ACS releases are shifted with respect to subsystem release so that a new ACS release is available at the beginning of a subsystem development cycle.

ACS 2.0 (November 2002) should have been normally used for R0 development, but this release did not contain the Java Component/Container prototype that was needed for high level subsystems development.

This prototype has been included in an “*update*” release called ACS 2.0.1 and distributed at the end of January 2003.

We will have therefore:

- R0 - ACS 2.0.1 – January 2003
- R1 - ACS 3.0 – October 2003
- R2 - ACS 4.0 – October 2004
- R3 - ACS 5.0 – October 2005
- R4 - ACS 6.0 – October 2006

ACS 2.0.1 has been distributed at the end of January 2003, before normal subsystem's R0 (2003-05-01), to make it possible to use it.

All other releases are aligned to be released at the end of the same month as the other Subsystems, with the actual ACS in use by Subsystems being the one distributed 6 months before, considering that between two major releases there will be also a *.1 intermediate, bug fixing release.

The ACS release includes integration with Archive and is synchronized to allow the other subsystems switching to the new release after integration of their previous release.

Notice therefore that each major Subsystems release will be based on the previous stable ACS *.1 bug fixing release. This should warranty the best stability of the final system.

(Incremental) Critical Design Reviews (CDR1,3) (T0+11 months and yearly afterwards):

- CDR1 - 2003-06-15 (1.5 month later to accommodate for PDR)



ALMA Project
ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA
Date: 2005-07-29
Status: Final
Page: 21 of 37

- CDR2 – 2004-07-31
- CDR3 – 2005-06-09

Readiness Review (RR) (T1-12 months = 2006-06-01)

Preliminary Acceptance Review (PAR) (T1-6 months = 2006-12-01)

Support Completion (T1=2007-06-01, normally T0+5 years)

Between any two releases new “builds” will be made available whenever necessary. A new “build” will contain patches and high priority new features requested by the subsystem teams and will be numbered with a third digit (for example, ACS 2.1.2 is the second “build” of ACS 2.1).

Constraints for a new “build” are:

- It is fully backward compatible with the official release and no changes to the application code are necessary.
- It is based on all the same core components as the official release (operating systems, CORBA tools, compilers).
- Requires minimal effort for upgrade, i.e. it does not require a complete ACS re-installation.

This model has been successfully used. For the past releases of ACS we had considerably less than one “build” per month and this has allowed responding in a quite agile way to requests coming from the subsystem teams.

4.2 Methods and Tools

See [Plan]

Details are provided in [ACS] and [ACSReq]

4.3 Infrastructure

Each site where ACS development takes place must have independent access to a complete ACS development environment. Since real-time ACS development and support are concentrated in Garching and Socorro, only here it is necessary to have direct access to RTOS machines.

Each developer must be able to work independently from other developers, sharing only the most expensive resources that are not needed all the time.

Based on these considerations and taking into account what is specified in [Architecture], [ACS] and [ACSReq]:

- Each developer must have available:



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 22 of 37

- 1 Windows personal computer for Windows/Java development and office automation
- 1 Linux workstation for ACS development (or a VmWare virtual machine with Linux, running inside the Windows personal computer).
- Each site must have available
 - 1 Linux Server for Database applications and for centralized ACS services. In Garching we make use also of a Web Server machine for ACS web hosting and Web Start deployment. This is shared with other teams.
- Garching and Socorro (**Socorro still does not comply with this infrastructure**) sites must have available for real time Linux development and support to the teams developing real time code:
 - 1 PC with real time Linux and a CAN board defined by Control and Correlator teams. This would be used for real time but not critical tests and as a CAN simulator.
We assume here that the CAN simulator code will be ported and made available on RH 9.
 - 1 VME crate with:
 - VMIVME 7766 main board
 - CAN board defined by Control
 - Parallel IP board to connect to the timing pulse. This is needed also if not timing pulse is connected because software from Control needs the board to be present
 - IP carrier
 - Some CAN devices for testing
- Garching and Socorro (**Socorro still does not comply with this infrastructure**) sites must have available VxWorks development environment until VxWorks will be phased out from ALMA development. Garching will maintain anyway also afterwards some limited support of VxWorks as a backup and for other projects:
 - 1 Linux workstation for VxWorks cross development (supported since ACS 4.1)
 - 1 Sun Solaris workstation for VxWorks development tools not available under Linux.



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 23 of 37

- 1 VxWorks LCU.
- ACS Integration and testing will be done in Garching on a dedicated and isolated ACS Control Model.
- At any time there will be two versions of ACS in use by the team:
 - The last released version (used by all other ALMA teams), that has to be supported.
 - The version under development.

This requires the availability, at least in Garching, of one or two systems with the last released version of ACS. These systems can be based on VmWare and share common HW with development systems. For Garching and NRAO this includes a Sun Workstation in order to be able to run the VxWorks tools not available under Linux (but the NRAO installation can be shared with the NRAO Control Software teams, which is the one that needs actual support on the latest release).

- A pool of ACS Linux and Windows workstations will have to be available in Garching for ACS training. This pool can also be based on a single PC with dual operating systems hosted by VmWare. Another interesting alternative would be that trainees come to Garching with their own laptop with VmWare where ACS gets installed. They can in this way take the installed system back to their own sites at the end of the training.


It is assumed that ACS integration testing is done in collaboration with the ALMA SW integration team and that System Management resources are available for the maintenance of the systems described in the previous paragraphs.

It is also assumed that all HW will be upgraded or replaced at least once in the 5 year period. On the other hand, Linux workstations are not under heavy usage, since are essentially single user machines. This allows to effectively use also older machines, in particular for training and support of previous ACS versions.

Most of the development HW has been already purchased.

The infrastructure for the whole team is therefore, with the given assumptions and considering 4 + 2 (Japan) team members in Garching, 1 at NRAO and 1 HIA:

- 8 Windows PCs for Windows development and office automation (not to be put at budget)
- 16 Linux development workstations (1 each developer, with VmWare + 1 each site for database + 2 in Garching for Control Model + 2 in

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 24 of 37
--	---	---

Garching for training + 1 in Garching and 1 in Socorro for real time Linux development)

- 3 Sun workstations (Garching and Socorro for development + one in Garching for previous release and control model)
- 2 Linux VME crates/LCUs for real time Linux development (1 in Garching and one in Socorro).
- 2 VxWorks VME crates/LCUs (1 in Garching and one in Socorro for development. The one in Socorro will be phased after having fully replaced VxWorks with real time Linux).

See section 1.4 for details on the HW that needs to be actually bought, taking into account what is already available from Phase 1.

4.4 Acceptance Plan

See [Practices]

All ACS code will be validated with tools for code checking, and test coverage as specified by the Software Engineering team.

The Software Engineering team will select the tools to be used and the ranges of values for the used metrics that define the acceptability of a release.

Each ACS module has its own modular test designed to test the functionality implemented by the module.

We also adopt an “incremental coverage” approach. The modular test of each module includes also tests designed to exercise features of the lower level modules in the dependency tree. This allows writing comprehensive tests in an easier way, profiting of the features made available by higher level modules.

Some top level ACS modules contain ACS integration tests whose purpose is to test the complete integrate ACS software.


In this way the test coverage of ACS is the result of the coverage of the complete test suite.

ACS pre-releases are distributed to subsystems that are particularly interested in new features, so that they can test them and provide feedback before the final release. This allows early discovery of bugs and tuning of the implementation.

5 Supporting Process Plans

5.1 Configuration Management

The process described in [Practices] is applicable.

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 25 of 37
--	---	---

5.2 Validation

See [Plan]

ACS development includes automatic regression tests both in the form of modular and integration tests, according to what described in section 4.4.

These tests are used to validate each release.

There is no validation of ACS with external users, since ACS is only visible directly to the subsystem developers.

5.3 Documentation

See [Plan] and [Practices].

ACS delivers extensive documentation as described in section 1.3.

5.4 Quality Assurance

See [Plan] and [Practices].

5.5 Reviews

See [Plan] and [Practices].

5.6 Problem Resolution

See [Plan] and [Practices].

ACS provides extensive support to subsystems, in particular during integration with new releases. A sizable amount of ACS resources are allocated to support.


5.7 Process Improvement

Feedback and contributions from the users are not sufficient.

We would like to have a more tight interaction with the users in order to:

- be able to suggest solutions (how ACS should be used to solve a specific problem)
- identifies what parts of ACS need changes or are not used
- improve documentation

During R2 and R2.1 (although less) the ACS team has been heavily involved in supporting the I&T team. In this activity we have in many occasions inspected the code produced by the other subsystems. We have noticed that very often ACS was not used in the way it was supposed to be used or features provided by ACS were not known and/or not used. A better documentation can improve the situation in this respect, but we believe that code reviewing can be much more effective in this respect. This was discussed also at the R2.1 meeting, but in order to be able to really apply it, we would have to increase the time allocated by ACS for support. At CDR3 we have agreed that we will allocate time for code reviews. We have

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 26 of 37
--	---	---

also discussed the possibility of pair programming sessions (considered very effective by the ACS team), but we have agreed that this is impractical for our development situation.

During the planning and development of ACS 4.1 we have applied a new technique to involve the user community: all new ACS features and refactoring activities has been described in details in Wiki design pages and officially reviewed by selected users.

This experience has been technically very positive, but also very expensive and has to be accounted for the fact that we could not allocate proper documentation time in this development cycle (although these design and discussion pages can be consider as well a good documentation of the discussed features). For the coming development cycles this will have to be taken into account in the planning.

6 Software Description

The description of the ACS Software is provided by the [ACS] document and by the Technical Architecture section of [Architecture].

The [ACS] document has been extended to cover the aspects currently described in [Architecture].

6.1 Software External interfaces

See [ACS] and [Architecture]

6.2 Software Design

See [ACS] and [Architecture]

6.3 Packages

See [ACS] and [Architecture]


The current packaging structure is described in [ACS]. A few aspects of the High Level Analysis and Design work in [Architecture] still need to be fully expanded in [ACS] and requirement from other teams, in particular for high level applications, pipeline and data reduction are being collected and evaluated.

6.4 Interfaces between Packages

See [ACS] and [Architecture]

6.5 Integration of Packages

See [Plan]

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 27 of 37
--	---	---

7 Attachments

7.1 Detailed Planning

The planning for ACS development is affected by the following primary factors:

- The planned PDR resource allocation over the 2005-2007 period is of 6.0 FTEs/Year including Japan contribution. (plus about 2 in-kind contributions for 2005)
- Delays in the allocation of resources from ALMA and increase in scope in the past years have been handled by in-kind contributions not accounted in the ALMA budget.
- Therefore the increase in scope is effectively compensated by the ALMA resources not allocated but planned until 2004. The total number of FTEs for ACS development allocated by ALMA remains of 23.5 FTEs + 3 FTEs from Japan for the overall period, although more total resources have been allocated to ACS using in-kind contributions.
- Availability of ACS features is a prerequisite for the development of many subsystems.
- Training and support for 2005 has to take into account the need of providing training to the new Japanese collaborators, both in ACS and in the Other Subsystems. As described above, we would also consider important to allocate time for code reviewing.
- We need to allocate more time to documentation in the coming cycles.

Given these factors and the experience of the previous years, we have updated the estimate of resources in FTEs as follows, taking into account in-kind resources and not only ALMA officially allocated resources:

Cycle	Design/ Development	Testing	Documentation	Support/ Training	Manag.	Coordination	Total
0a-ACS 0.x	1.1	0.4	0.4	0.3	0.1	0.2	2.4
0b-ACS 1.x	1.2	0.4	0.4	0.6	0.1	0.2	2.9
1-ACS 2.x	1.5	0.5	0.5	1.7	0.1	0.4	4.7
2-ACS 3.x	2.0	1.2	1.2	1.3	0.1	0.4	6.2
3-ACS 4.x	2.0	1.3	1.7	1.3	0.1	0.4	6.8
4-ACS 5.x	2.4	1.2	2.0	1.8	0.1	0.5	8.0
5-ACS 6.x	2.4	1.2	1.1	0.8	0.1	0.5	6.0

Where we have considered that:



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

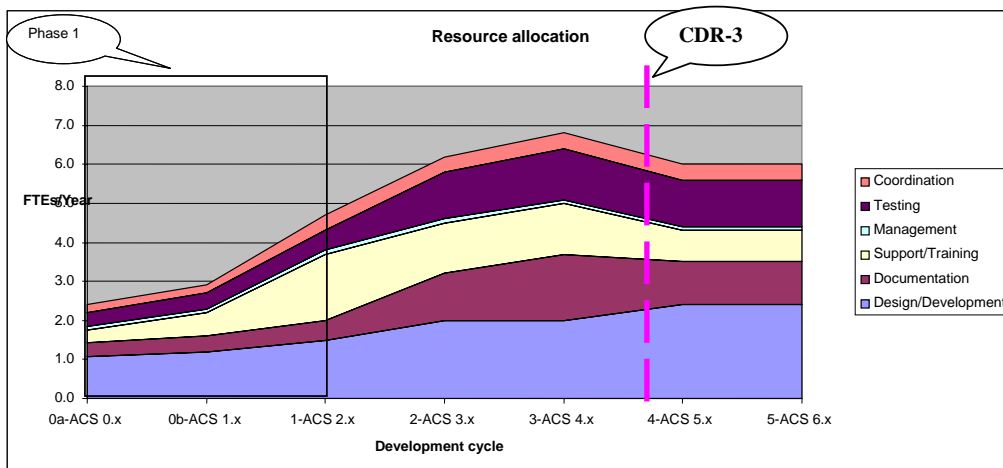
Status: Final

Page: 28 of 37


- A considerable amount of design/development has been already done in phase 1. This covers important core ACS packages in particular in the area of Control System applications.
- 0.3 FTEs of training in cycles 1 and 2 would have been recovered by training-on-the-job tasks
- Support requests will grow up again in the last cycle, with final release approaching
- Testing is about 20% of design/development time.
- Documentation is about 20% of design/development time, but requires a stronger effort in cycle 4.
- Managing account for team's manager (G.Chiozzi) activities for the management of the ACS team, ACS planning and reporting to ALMA SW management.
- Coordination time is equally distributed among all ACS team members and includes the participation to global ALMA meetings and to ACS weekly status and coordination meetings, including the time needed for the preparation of reports.

This can be seen with a different representation in the following diagram:

Comment: To be revised.



It is therefore very important to accurately select per each major and minor release the new ACS features that will have to be implemented, by discussing at each CDR and at ACS planning meetings a priority list with the subsystem teams. This has been done already successfully for all ACS releases.

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 29 of 37
--	---	---

Changes in priorities and special requests can be handled with In-Kind contributions to the ACS development team from other teams or from external groups.

The general idea is that all ACS packages will be developed in parallel, implementing at any cycle only the features with higher priority and revising the priorities at each CDR.

In the first two cycles, priority has been given to the design/development of interfaces with a bare bones implementation. Priority in the first cycle has been given to the Java Component Model. Priority in the second cycle has been given to features requested by the pipeline, OT and executive high level teams and in particular at extensions of the Manager capabilities for high-level applications (like dynamic deployment of components and administration) and at the implementation of the Python Container.

The third cycle has been dedicated mainly to the development of the BulkData system and of features required for the Offline development (Task and Parameter). Re-factoring and cleanup of interfaces and implementations also had an important role.

The following cycle will be devoted mainly to the optimization and scalability of the implementation of the interfaces that will remain backwards compatible.

In the third cycle we have dedicated a sizable amount of time to performance measurements, but this activity is not completed yet.

The last cycle will be essentially dedicated to the stability and reliability of the system, with extended tests under stress conditions.

Details on the ACS planning for each release are available in the ACS wiki:

- <http://almasw.hq.eso.org/almasw/bin/view/ACS/AcsPlanning>

and in the ACS CVS documentation branch where the older versions of this document are archived:


- CVS:ACS/Documents/ACSSoftwareDevelopmentPlan

Since ACS 3.0, the detailed allocation of resources is maintained in the wiki document.

These documents provide details of the time allocated for the development of ACS features and describe the changes in planning (re-scoping, additions) at each release and after the release.

7.2 Requirements Compliance Table

Last update: 2005-07-29.

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 30 of 37
--	---	---

The following table provides a trace of all requirements expressed in the ACS Requirements Document [ACSReq].

The entries in this table have been extended to take into account the requirements expressed in the in the ALMA Software Architecture & High-Level Design document [Architecture] and not already listed in the ACS Requirements Document [RD01].


Notice also that the ACS Requirements Document [RD01] is often necessarily generic and has been last updated for CDR-1. In many cases, requests for new requirements or extensions to ACS come from the discussions with the other ALMA sub-system's teams. This happens in particular for the recent requirements from pipeline and high level applications. We have therefore added such items in this table as further specification of items already foreseen in general terms in [RD01] or as new entries. For more details look at the minutes of the ACS planning meetings in the ACS wiki:

- <http://almasw.hq.eso.org/almasw/bin/view/ACS/AcsPlanning>

A **Release** column specifies when the core or major extensions of the described requirement has been or will be implemented.

What follows is the planned set of releases (1 major release every year, with an intermediate minor release):

- 0.0 is the first prototype release (2000-11-15, Done!)
- 1.0 is the first production release, that should implement all basic functionality (2001-09-27, Done!)
- 1.1 is a bug fixing release (after a 6 month cycle), providing no new functionality (2002-04-16, Done!)
- 2.0 is a major release (November 2002, Done!)
- 2.1 is a minor release (June 2003, Done!)
- 3.0 is a major release (November 2003, Done!)
- 3.1 is a minor release (April 2004, Done!)
- 4.0 is a major release (November 2004, Done!)
- 4.1 is a minor release (May 2005, Done!)
- 5.0 is a major release (October 2005)
- 5.1 is a minor release (April 2006)
- 6.0 is a major release (October 2006)
- 6.1 is a minor release (April 2007)

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 31 of 37
--	---	---

Many details are provided for the next releases, while the detailed planning for the following releases will be dynamically defined based on the priorities identified together with the Subsystem's contact persons and at each CDR. Only the major blocks are identified.

We foresee that releases 4.0 and 5.0 will for a major part contain extensions/re-factoring requested by the sub-systems on the core functionality developed in the previous cycles. A major part of releases 5.0 and 6.0 will consist in performance and scalability optimization in view of the operation with the complete array and with the deployment of the off-mountain components.

When only partial functionality is implemented in a release, the release number is put in parenthesis. Some requirements (like 2.3, 3.1.1 and so on) are at the core of the whole ACS and therefore it does not make any sense to specify a release for implementation. In this case the release is marked as Not Applicable (N/A).

Item in ACS Technical Requirements [RD01]	ACS Architecture	Release
2.3. Reference Architecture	1.3	N/A
3.1.1. ACS Scope: Scope.	1.2	N/A
3.1.2. ACS Scope: Design.	1.2, 2.1	N/A
3.1.3. ACS Scope: Use.	1.2	N/A
3.2.1. User Requirements: Users.	1.2	N/A
3.2.2. User Requirements: Value retrieval. Implementation of interfaces. Basic engine. Hierarchy of access classes (Memory, Properties) Performance enhancements	3.6.3	1.0 1.1 (4.1) 5.0
3.2.3. User Requirements: Value setting.	2.5.11.2	0.0
3.2.4. User Requirements: Local and central operation.	2.2	(0.0) 1.0
3.2.5. User Requirements: Remote access.	2.2	1.0
3.3.1. System requirements: Size.	4.4	0.0
3.3.2. System requirements: Serialization. Prototype Pluggable transport interfaces XML format definition CORBA transport HTTP and Email transport	3.5.3.1, 3.5.5	2.0 2.1 2.1 2.1 5.1
3.3.3. System requirements: Migration.	3.5.6	3.1
3.3.4. System requirements: Simulation.	3.5.9	4.0
4.1.1. Value retrieval: Direct value retrieval.	3.5.11.1	0.0
4.1.2. Value retrieval: Indirect value retrieval.	3.6.3	1.0
4.1.3. Value retrieval: Rate	3.5.12.2, 3.5.12.3	1.0
4.1.4. Value retrieval: Transparency.	3.6.3	1.0



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 32 of 37

4.1.5. Value retrieval: Values at given time.	3.5.12.3	(0.0) 1.0
4.1.6. Value retrieval: Data channel. CORBA Notification Service High Level API Event Browser (from I&T team)	3.6	1.0 (2.0) (3.0) 3.1 3.1
4.2.1. Database: Configuration Database. Implementation based on VLT CDB Implementation based on XML files Online database browser Write interfaces Implementation based on ALMA Archive Database configuration tools	3.5.3, 3.13.14	0.0 2.0 2.1 3.1 (3.0) (3.1) 5.1
4.2.2. Database: Database design.	3.5.3.2	1.0
4.3.1. Sampling: Prototype and interfaces Optimized engine	3.15.4, 3.15.7	(1.1) (2.1) 3.0 5.1
5.1.1. Tools: Framework.	3.18.1	2.0
5.1.2. Tools: Procedures.	3.11.1	2.0
5.1.3. Tools: Browser.	3.11.10	1.0
5.1.4. Tools: Analysis tool.	3.15.5	1.0
5.2.1. Scripts: Scripts.	3.17.1, 3.17.2	(0.0) 1.0
5.2.2. Scripts: Functionality.	3.17.3	0.0
5.3.1. Libraries: Time conversion	3.9.3.1	(1.0, TICS) 3.0
5.3.2. Libraries: Mathematics	3.19	2.0
5.3.3. Libraries: Astrometry	3.10	2.0
6.1.1. Messages: Messages	3.13.3	(0.0) 1.0
6.1.2. Messages: Commands.	3.13.1	0.0
6.1.3. Messages: Syntax Check.	3.13.2	4.1
6.1.4. Messages: Validation.	3.13.2	1.0
6.1.5. Messages: Programmatic use.	3.13.7, 3.13.14	(1.0) 4.1
6.1.6. Messages: Command delivery.	-	
6.1.7. Messages: Mode.	3.13.3	(0.0) 1.0
6.1.8. Messages: Timeout.	3.13.3	(0.0) 1.0
6.1.9. Messages: Intermediate replies.	-	
6.2.1. Logging: Logging. Interfaces and implementation with CORBA Telecom Logging Prototype GUI Integration with ALMA Archive Java client services Final GUI	3.8.2, 3.8.4	(0.0) 1.0 1.1 (TICS? 3.1) 2.1 3.0
6.2.2. Logging: Persistency. (see also configuration	3.8.6	

Comment: Commands map more or less on parameter tasks. To be better analysed.



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 33 of 37

database)		
Object-Relational mapping scheme		2.0
RDBMS and XML mappings		2.0
Advanced query capabilities		4.1
6.2.3. Logging: Filtering.	3.8.11	(0.0) 1.0
6.2.4. Logging: standard API	3.8.7	2.1
6.3.1. Definition.	3.7.6	N/A
6.3.2. Errors: Tracing.	3.7.3	1.0
6.3.3. Errors: Severity.	3.7.10	1.0
6.3.4. Errors: Presentation.	3.7.8	(0.0) 1.0
6.3.5. Errors: Configuration. Error definition GUI	3.7.9	(2.0) 2.1 5.1
6.3.6. Errors: Scope.	3.7.2	0.0
6.4.1. Alarms: Definition.	3.14	N/A
6.4.2. Alarms:Behavior.	3.14.4	(2.0) (4.1) 5.0
6.4.3. Alarms:Severity.	3.14.4	(2.0) (4.1) 5.0
6.4.4. Alarms:State.	3.14.4	(2.0) (4.1) 5.0
6.4.5. Alarms:Hierarchy.	3.14.5	(4.1) 5.0
6.4.6. Alarms:Alarm logging. Prototype Final implementation	3.14.3	1.1 (4.1) 5.0
6.4.7. Alarms:Configuration.	3.14.4	(4.1) 5.0
6.4.8. Alarms:Binding.	3.14.6	(4.1) 5.0
7.1 Bulk data transfer Interfaces Prototype implementation Performance verification Optimized performances	3.19.1	3.0 3.1 4.0 4.1
7.1.1. Bulk data transfer : Image pipeline	3.19.1	5.1
7.2.1. Data format: FITS format shall be supported. C++ support Java and/or Python support	3.19	2.1 3.1
8.1.1. Time System: Standard.	3.9.2	1.0
8.1.2. Time System: API.	3.9.3.1	1.0
8.1.3. Time System: Distributed timing. Integration of TICS Time System into ACS	3.9.1	(3.0) 3.1
8.1.4. Time System: Services.	3.9.3.3	1.0
8.1.5. Time System: Resolution.	3.9.5	1.0
9.1. Security. Passive security. Basic DO access User authentication and access rights	4.1	(1.0) (4.1) 5.0
9.2.1. Safety: All human and machine safety	4.2	0.0
9.2.2. Safety: The use of software limits shall be supported	4.2	1.0



ALMA Project

ACS Software Development Plan

Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 34 of 37

9.3 Reliability	4.3	1.0
9.4 Performance	4.4	0.0, 3.1, 4.0, 5.0
ACS performance test suite		(3.1) (4.1) 5.0
10.3.1. Software Standards: RTOS. VxWorks support Real Time Linux support	5.1.1	0.0, 3.0, 4.1 (Linux) 3.1, (4.1) 5.0
10.3.2. . Software Standards: High level operating system.	5.1.1	0.0
10.3.3. . Software Standards: Compiled Languages.	2.2, 5.1.1	0.0
10.3.4. . Software Standards: IDL.	3.13.1.2	0.0
10.4.1. Communication Standards: CORBA.	3.13.1	0.0
10.4.2. Communication Standards: ORB independence.	3.5.8	0.0
10.4.3. Communication Standards: LAN.	1.3	0.0
10.4.4. Communication Standards: Backbone.	1.3	0.0
10.4.5. Communication Standards: Field-bus.	1.3	0.0
10.5.3. Reference Products: RTOS	3.4.2, 5.1.1	0.0, 3.1
10.5.4. Reference Products: OS	3.4.2	0.0, 3.1
10.5.5. Reference Products: Configuration DB Prototype of Configuration Database File-system based CDB Implementation of CDB on ALMA Archive	3.5.3.2	(0.0) 2.0 (3.0) (3.1) 5.1
10.5.6. Reference Products: Scripting language	2.2, 3.17.4	(0.0) 1.0
10.5.7. Reference Products: Analysis and plotting	3.15.5	4.1
10.5.8. Reference Products: CORBA ORB	3.5.8	0.0
10.5.9. Reference Products: Data transfer	4.4	(0.0) 1.0
10.5.10. Reference Products: The XML format	3.5.5	2.0
10.5.11. Reference Products: LAN	1.3	0.0
10.5.12. Reference Products: Field-bus	1.3, 3.4.1	0.0
11. Life cycle aspects	5	0.0
11.1.1. Life cycle aspects: Prototyping	5	0.0
12.1.1. User Interface: Portability	3.16.1.2	0.0
12.1.2. User Interface: Extended portability	3.16.1.2, 3.16.1.3	(0.0) 1.0
12.1.3. User Interface: Tools	3.16.2	0.0
12.1.4. User Interface: Modularity	3.16.4	0.0
12.1.5. User Interface: ACS GUIs	3.16.3	(0.0) 1.0
12.1.6. User Interface: Location	2.2	1.0
12.2.1. Hardware interfaces	3.4.1	(0.0) 1.0
12.3.1. Software Interfaces: The ACS shall be integrated and tested	5.1	(0.0) 1.0
13.1.1. ACS Design Reqs.: Distributed Objects and Commands.	2.1, 3.5, 3.5.1.5, 3.13.1	0.0
13.1.2. ACS Design Reqs.: Standard methods.	3.5.4	2.0



ALMA Project

ACS Software Development Plan


Doc # : COMP-70.25.00.00-001-C-PLA

Date: 2005-07-29

Status: Final

Page: 35 of 37


13.1.4. ACS Design Reqs.: Events.	3.5.12.1	(0.0) 1.0
13.1.5. ACS Design Reqs.: Configuration.	5.1	1.0
13.1.6. ACS Design Reqs.: Modularity.	5.1	1.0
13.1.7. ACS Design Reqs.: Portability.	5.1.1	0.0
14.1.1. Style guide reqs.: Logging of commands	3.8.3	1.0
14.1.9. Style guide reqs.: Dynamic configuration	3.5.3.5	1.0
15.1.1. Components: Components C++ Java Python	2.2, 3.4	1.0 2.0 3.0
15.1.2. Components: Lifecycle Interface Java Prototype Implementation for all languages	2.2, 3.4.1	2.1 (3.0) 3.1
15.1.3. Components: Service Interface	2.2	1.0
15.1.4. Components: Deployment Deployment information in CDB Manager federation and scalability [Architecture] Dynamic deployment interface (Pipeline req.) Deployment based on Load Balancing recipes (Pipeline req.) Advanced task management ([Architecture], pipeline req.)	2.3, 3.10.2, 3.10.6	1.0 (4.1) (5.1) 6.0 3.0 (3.1) 6.0 (4.1) 6.0
15.2.1. Container Services: Lifecycle Java Prototype Implementation for all languages	2.2, 3.4.1	2.1 3.0
15.2.2. Container Services: Languages C++ Java Python (Pipeline req.)	2.3, 3.10.9 (to be extended with Python container)	1.0 2.0 3.0
15.2.3. Container Services: Logging C++ Java Python	3.10.7	1.0 2.0 1.1
15.2.4. Container Services: Archive C++ Java Python	3.10.7	3.0
15.2.5. Container Services: Tight container	3.10.8	2.1
15.2.6. Container Services: Porous Container	3.10.8	2.1

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 36 of 37
--	---	---

15.3.1. Entity Objects: Definition	3.11	2.1
15.3.2. Entity Objects: Serialization	3.11	2.1
15.3.3. Entity Objects: Automatic generation of binding classes	3.10.9, 3.11	2.1
Manager Administration APIs (Executive req.)		3.1
Master Component with standard state machine (Executive req.)		3.1
Generic State Machine Component (Correlator req.)		5.1 ?
IDL generic simulator (Santa Fe design Meeting 2003)		4.0
ACS flexible packaging and installation: pure Java ACS installation (OT req.) web start installation (OT req.) run-time only/development ACS installation (various subsystems req.) modular installation with selected ACS components (data reduction req.)	2.3	3.0 3.0 5.0 6.0
Java ACS on single Virtual Machine (OT req.)	2.3	3.1

Notes:

- Requirements in sections [RD01 - [10.1 Standards and Procedures](#)], [RD01 - [10.2. Computer hardware standards](#)] and [RD01 - [10.3 Software](#)] are mostly not directly traced in the document. They are in any case taken into account and satisfied by the choices done for the reference products listed in [RD01 - [10.5 Reference products](#)].
- Requirement's section [RD01 - [14. Requirements for applications](#)] is in principle N/A to this document, since states requirements for applications and not for ACS. Nevertheless, ACS will try to support as much as possible applications in satisfying these requirements. For this reason a few sub-sections are directly traced in the document.
- The Alarm System [RD01 – 6.4] has been implemented as a first prototype with ACS 3.1. That initial prototype is being now re-written based on the Laser Project developed at CERN for the LHC Accelerator (<http://proj-laser.web.cern.ch/proj-laser>). This project has requirements very similar to the once expressed for ALMA and therefore allows ACS to profit from a very detailed analysis and from existing code.

	ALMA Project ACS Software Development Plan	Doc # : COMP-70.25.00.00-001-C-PLA Date: 2005-07-29 Status: Final Page: 37 of 37
--	---	---

- Requirements [RD01 - [6.1.6 Command Delivery](#)] and [RD01 - [6.1.9 Intermediate reply](#)] are not satisfied by ACS. As discussed during the review phase, they are left for the time being to the responsibility of applications.