Construction Project

Interface Control Document
for
ACADA – Array Element Monitoring

Prepared by	Igor Oya Vallejo	Digitally signed by Igor Oya Vallejo Date: 2021.04.28 09:16:42 +02'00'	2021.04.28
	I. Oya, ACADA WP Coordinator		Date
Approved by		Digitally signed by Nick Whyborn Date: 2021.04.29 09:52:19 +02'00'	
	N. Whyborn, Lead Systems Engineer		Date
	CHIARA Montanari	Firmato digitalmente da CHIARA Montanari Data: 2021.04.28 09:38:37 +02'00'	
	C. Montanari, Interface Manager		Date
Released by	Wolfgang Wild	Digitally signed by Wolfgang Wild Date: 2021.06.10 16:38:36 +02'00'	
	W. Wild, Project Manager		Date

Change Log

Issue	Revision	Date	Section/Page affected	Reason/ Remarks / Initiation Documents
1	a	2020.11.16	All	New document. Discussed with Array Element teams, all feedback incorporated.
1	b	2021.04.22	• Note (1) of table 1 – typo fixed (40 instead of 8) • Add FRAM and LIDAR acronym definition • Fix wording in item ii in Sec. 3.3.1 • Rephrase the sentence on ACS and OPC UA component deactivation in Sec. 3.3.1 • Rephrase sentence on reading data with just one of the strategies (ACS or OPC UA) for a device • Rephrase component naming Scheme in Sec. 3.4.3. • Rephrase the 1st paragraph of Sec. 3.6 to indicate where the concrete list of prop-	• Typo found by U. Schwanke • Review of W. Wild prior to document release

List of Contributors

Name	Organization	Contribution
Igor Oya	CTA PO	The document and current version of the interface
M Füßling	CTA PO	General document revision
E Antolini	CTA PO	General document revision
Alessandro Costa, Alessandro Grillo, Pietro Bruno	INAF	Revision and OPC UA configuration

Chiara Montanari	CTA PO	Interface overall process management

List of ICD actors

Actor	Role	Agreed date and signature
I. Oya	ACADA Coordinator	Igor Oya Vallejo Digitally signed by Igor Oya Vallejo Date: 2021.04.28 09:16:24 +02'00'
D. Mazin	LST	
U. Schwanke	MST Struct	
P. Sizun	NectarCam	
M. Barcelo	FlashCam	
G. Pareschi	SST	
M. Gaug	Calibration Coordinator	

Table of Contents

List of Acronyms

ACADA	Array Control and Data Acquisition
ACS	Alma Common Software
BACI	Basic Access and Control Interface
CDB	Configuration Database
CTA	Cherenkov Telescope Array
DPPS	Data Processing and Preservation System
FRAM	F/Photometric Robotic Atmospheric Monitor
ICD	Interface Control Document
LIDAR	Light Detection and Ranging
TCS	Telescope Control System
TOSS	Technical Operations Support System

1 Scope

This document specifies the requirements of interface describing the monitoring operations of the Array Control and Data Acquisition (ACADA) System on any CTA Array Element.

Interface management is a process to assist in controlling product development when efforts are divided amongst different parties (e.g. agencies, contractors, geographically dispersed technical teams).

This ICD describes the interface between the Array Control and Data Acquisition (ACADA) System (the target system) and a CTA Array Element (the source system).

The purpose of the ICD is to define the design of the interface(s) ensuring compatibility among involved interface ends by specifying form, fit, and function.

The ICD is managed by the CTAO Interface Manager (or their delegates) and represents an agreement between the relevant actors. The actors in this ICD are shown at the beginning of this document.

The ICD is used:
1. to document the interface definition,
2. to control the evolution of the interface,
3. to document the design solutions to be adhered to, for a particular interface,
4. as one of the means to ensure that the supplier design (and subsequent implementation) are consistent with the interface requirements,
5. as one of the means to ensure that the designs (and subsequent implementation) of the participating interface ends are compatible.

This Interface Control Document (ICD) documents and tracks the necessary information required to effectively define the interface between an Array Element and ACADA for the

Monitoring information flow from the former to the latter, as well as any rules for communicating with them in order to give the development team guidance on the architecture of the system to be developed.

The purpose of this ICD is to clearly communicate all possible inputs and outputs from the system for all potential actions whether they are internal to the system or transparent to system users.

Its intended audience is the Systems Engineering personnel, ACADA and Array Element development teams, and stakeholders interested in the interfacing of these systems.

2 Applicable and Reference Documents

2.1 Applicable Documents

- [AD1] CTA Architecture Document v1.0 14.04.2018.
- [AD2] Generic Telescope Use Cases, Doc. No. CTA-TRE-SEI-000000-0015 Issue 2, Rev i, 4.12.2019.
- [AD3] CTA Telescope State Machine, Doc. No. CTA-SPE-ACD-000000-0001, Issue 2, Rev f, 21.4.2020.
- [AD4] Common Telescope Requirements, Jama.
- [AD5] Common On-Site Requirements, Jama.
- [AD6] Array Control Requirement Specification, Doc. No. CTA-SPE-COM-303000-0001, Issue 2, Rev. i, 29.4.2020.
- [AD7] R1/Event Data Model Specification, Doc. No. CTA-SPE-COM-000000-0002, Issue 1, Rev. b, 18.3.2020.
- [AD8] Top-level Data Model, Doc. No. CTA-SPE-OSO-000000-0001, Issue 1, Rev. a, 23.1.2020.

2.2 Reference Documents

- [RD1] Alma Common Software. G. Chiozzi, et al, "CORBA-based Common Software for the ALMA project", in Proc SPIE 4848, 43, 2002. doi: 10.1117/12.461036
- [RD2] ACADA-Telescope Control ICD Doc. No. CTA-ICD-SEI-000000-0002 Issue 2, Rev.: f, 19.3.2020.
- [RD3] ACS Basic Control Interface Specification, Gasper Tkacik et al, Revision: 2.5.3, 2005-09-08.
- [RD4] ACADA Architecture Design Doc. No. CTA-TRE-COM-303000 0001, Issue 2, Rev.: e, 29.4.2020.
- [RD5] Management and Access Control Interface Specification. KGBSPE01/01, Rev. 1.4. 2005-09-08.
- [RD6] [RD06] OPC 10000-1 - OPC Unified Architecture Part 1: Overview and Concepts, Rel. 1.04, 2017-11-22
- [RD7] https://gitlab.cta-observatory.org/cta-computing/common/acada-array-elements/testacsproperties
- [RD8] ACS Supported BACI Types. M. Plesko et al, Rev.1.4, 2005-03-17.
- [RD9] http://www.eso.org/projects/alma/develop/acs/OnlineDocs/acsexmpl/ws/doc

/html/classLongDevIO.html

- [RD10] OPC 10000-12 - OPC Unified Architecture Part 12: Discovery and Global Services, Rel. 1.04, 2018-02-07
- [RD11] Requirement Specification of the ACADA Array Configuration System Doc. Num. CTA-SPE-COM-303000-0011 Issue 1, Rev. b 28.04.2020.
- [RD12] Software Programming Standards, Doc. Num. CTA-STD-OSO-000000-0001 Issue 1, Rev a, 23.01.2020.

3 Interface Requirement specification

3.1 Overview

A CTA Telescope is a system composed of both a Camera and a Structure functional unit, each composed of hardware and software, and both coordinated by a Telescope Manager component (See Figure 1). Other Array Elements such as the FRAM or LIDARs have a similar decomposition, with their corresponding Manager components.

The Array Element Manager is responsible for managing the Array Element system as a whole. The Array Element Manger synchronizes and coordinates the different controllable elements on-board the array elements in order to execute the commanded operations. The interactions regarding the control aspects are specified in [RD2].

Physical devices and software components inside the Array Elements expose *properties* that indicate either the state on which the array element is, or the measurement of a sensor at a given time. Those properties expose *monitoring* data, as classified as R1/Monitoring in the CTA data model [AD8]. Examples of such a property are the readout value of a power supply, the measurement of a temperature sensor inside a cabinet, the position of a filter wheel, or the current state in the state machine of a software component. The Array Elements expose these properties to ACADA which reads them according to an agreed software interface definition file (IDL file) and defined settings (readout rate, units, …) stored in a centralized configuration database, which is part of ACADA [RD11][1]. ACADA uses the received data to provide a hierarchical visualization of the status of array elements in the panel for the on-site Operators and uses the property measurements to raise alarms when defined safe limits are crossed[2]. Then ACADA stores the data locally and transmits all the data to TOSS and DPPS, where it is used offline for inspection by experts and analysis pipelines and kept for long-term storage. The process of ACADA reading and handling the Array Element properties is defined as *monitoring Array Elements* in this document.

[1] Note that whereas changes on the configuration settings are expected to be frequent as required by CTA personnel, changes on the interface will only be possible via a change request procedure.
[2] Note that alarms can also be triggered programmatically using the ACS alarm API.

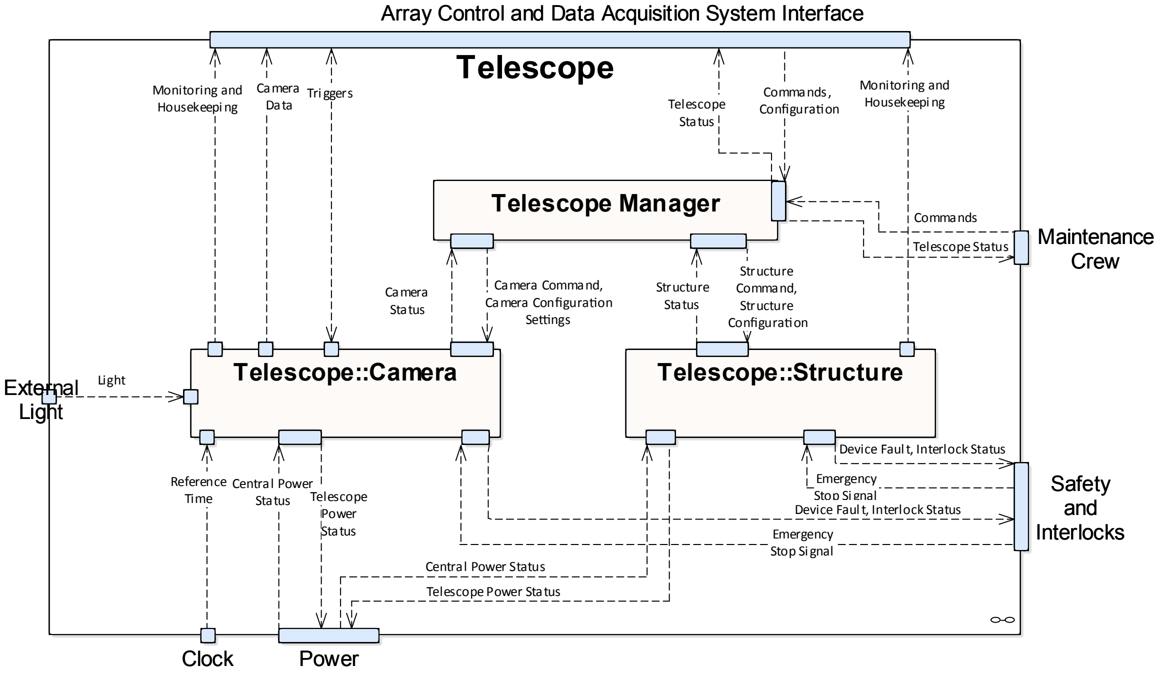

Figure 1: Decomposition of a CTA Telescope in functional units. Other array elements such as LIDARs or the FRAM have analogous decompositions, but with different type of functional units. See [AD4] for notation.

This ICD specifies the interface requirements the ACADA and Telescope Control System must meet.

This interface is a software interface, and thus it describes:

- the concept of operations for the interface,
- defines the message structure and protocols that govern the interchange of data,
- and identifies the communication paths along which the project team expects data to flow.

For each interface, the ICD provides the following information:

- A description of the data exchange format and protocol for exchange
- A description of the interface
- Assumptions where appropriate
- Estimated size and frequency of data exchange

3.2 Assumptions

The ACADA / Array Element monitoring interface supports both the ALMA Common software control framework [RD1] and OPC UA mechanisms [RD6].

The ACADA subsystem affected is the *Monitoring System* [RD4]. On the Array Element side, the monitoring data sources within its *Control System* are concerned [AD1].

3.2.1 Constraints

- Any monitoring data source shall be accessible from ACADA either as part of the same ACS instance deployed in the same on-site data centre, or via the standard OPC UA protocol, as a node in an OPC UA server deployed or accessible from the on-site data centre.
- The only data structures supported are those compliant with the data model presented in this document, or those specified in additional dedicated Array Element / ACADA ICDs.
- Monitoring data configuration shall be provided according to the monitoring data model presented in this document.
- Monitoring data shall be read by ACADA according to the mechanisms specified in this document.
- The total integrated amount of monitoring points shall be compliant with the ACADA requirements [AD6].
- The maximum monitoring rate shall be compliant with the ACADA requirements [AD6].

3.2.2 Functional Allocation

This interface implements the following functions:

- Gather and store monitoring data item (properties) from the Array Elements for further handling within ACADA and other CTAO computing systems.

3.2.3 Extension of the interface

Any Array Element must implement a monitoring interface with ACADA as described in this document. Each Array Element will have its concrete set of monitoring points. Those monitoring points will be included in the ACADA monitoring system according to the mechanism specified in this document.

Any monitoring data type not compliant with the generic data types presented in this ICD must be specified in an additional, dedicated ICD of ACADA and the corresponding Array Element. This refers to data types from the auxiliary instrumentation onboard of telescopes (CCD Cameras) or Array Calibration and Environmental Monitoring Devices whose complex data structures do not fit in the generic data types specified in Sec. 3.4.1, or because their readout rates go beyond 5 Hz. The strategy of ACADA for supporting this kind of data types is to handle this data via an analogous way as it is done with the Camera Data via the ACADA Array Data Handler [AD6], [RD4].

3.2.4 Data Transfer

The monitoring information of standard monitoring data is transferred from the data source components to ACADA components named *collectors* via standard ACS or OPC UA mechanisms in real-time. Depending on the overall network performance optimization and volume of data ACADA will deploy one or more collectors , for each Array Element in the same computing node as the monitoring sources, or in another computing node in the same cluster. Non-standard data transfer (see Section 3.4.4) is specified in dedicated ICDs. The transfer from the collectors to the ACADA monitoring database and the storage of monitoring data as DL0 files for DPPS storage is handled by ACADA [RD4] and outside the scope of this ICD.

3.2.5 Transactions

The following transaction types are supported:

- **ACS Property Subscriptions:** ACADA will set up individual property subscriptions (*monitors* in the ACS jargon, see [RD3]) to individual data source properties according to the defined property monitoring configuration settings in the Array Configuration System within ACADA. The usual monitoring rate of those properties is 1 Hz or below, however up to 5 Hz and readout on value change can be supported, as long as the overall ACADA requirements are met.
- **OPC UA Nodes Subscriptions:** ACADA will set up individual property OPC UA subscriptions (see [RD6]) to individual data source properties according to the property monitoring configuration settings in the Array Configuration System within ACADA. The usual monitoring rate of those properties is 1 Hz or below, however up to 5 Hz and value change triggers can be supported, as long as the overall ACADA requirements are met. See [RD3] for further details.

3.2.6 Security and Integrity

These aspects are not addressed in this document yet.

3.3 Interface specification

This section specifies all elements of the ACADA to Array Element Monitoring interface.

3.3.1 General Monitoring Scheme

ACADA interfaces with monitoring data sources within the array element control systems directly. This ensures the required modularity, flexibility, and performance since monitoring is a 24/7 activity that shall not depend on the current status of the Array Element Manager components. Moreover, the amount of expected monitoring points is too large, in general, to channel them through the Array Element Manager. Two data transfer mechanisms are supported:

i) Alma Common Software (ACS) *bridges*, which are ACS software components whose role is to adapt any low-level protocol into the ACS environment. The default low-level protocol is OPC UA [RD6], however, any protocol can be in principle bridged via the ACS DevIO mechanism [RD9], see also [RD7] for an example. It is the responsibility of the Array Element teams to implement the ACS Bridges in such a way that relevant properties are exposed to ACADA.

ii) Direct monitoring of OPC UA servers nodes. It is the responsibility of the Array Element teams to provide the OPC UA servers in such a way that relevant properties are exposed to ACADA as described in this ICD.

In both cases, ACADA pulls values from the property on the array element. Depending on the setting the readout ACADA can be triggered in value changes, or by a fixed rate (see below).

In order to avoid sending ACADA irrelevant monitoring information when the corresponding subsystem or device is offline, the corresponding ACS Component or OPC UA server shall be deactivated. In those cases when already existing hardware or software characteristics do not make the deactivation possible, the exposed monitoring values shall encode an exception in their completion (ACS), or node value (OPC UA).

Figure 2 illustrates how ACADA and an Array Element would implement the monitoring process. The Array Element in this example includes three devices with properties that need to be monitored by ACADA. For illustration purposes, the two variants of ACS bridge variant monitoring, as well as the direct OPC UA monitoring mechanism are illustrated. In a real CTA Array Element, only one of these strategies shall be used within a sub-system or device (i.e., data are not read out twice by ACADA).

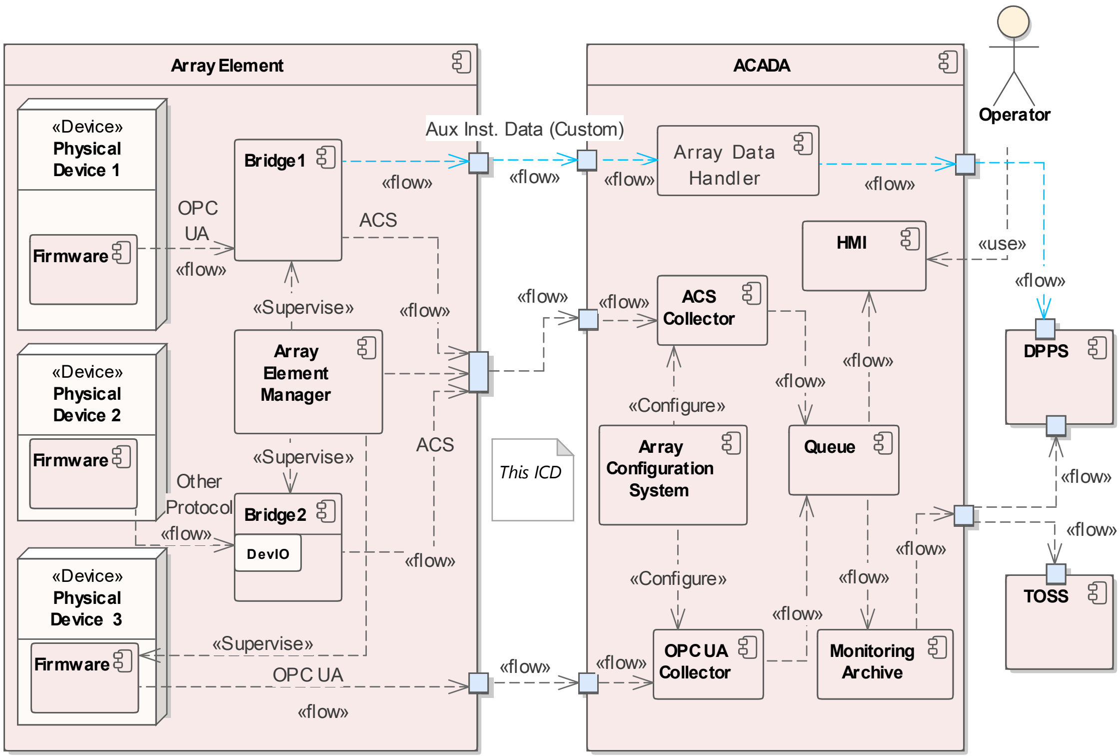

Figure 2: Diagram illustrating the monitoring data flow from an Array Element to ACADA, illustrating the different logical elements and protocols described in this ICD. This Array Element is an academic example that illustrates the three variations that can be used to deliver monitoring information to ACADA according to this ICD: i) Physical Device 1 includes an OPC UA-based firmware that is bridged into the ACS environment by Bridge 1; ii) Physical Device 2 uses a custom protocol that is bridged by Bridge 2 into ACS via the DevIO mechanism; iii) Direct OPC UA monitoring. Note: This diagram is for illustration purposes, and there will be array elements that implement architectures use dedicated protocol (e.g. Ethercat, Profinet) to collect all the monitoring points of different physical devices in one PLC which already has an OPC-UA server integrated. The arrows with blue color indicate the flow of complex data types generated at on-board auxiliary instrument that does not use the standard monitoring mechanisms, see Section 3.4.4.

3.4 R1/Monitoring Data Model and Format

This section specifies the Monitoring data model relevant for this ICD. This data category is classified as R1/Monitoring in the CTA data model [AD8].

3.4.1 Property data model and format.

The main kind of R1/Monitoring data addressed in this document is the one corresponding to device properties. Figure 3 and Table 1 specify the data model of these data types, which deals with generic primitive data types, vectors of them, as well as strings, string enumerations, and bit patterns. More complex data structures are supported but not in a generic out-of-the-box way, see Sec. 3.4.4.

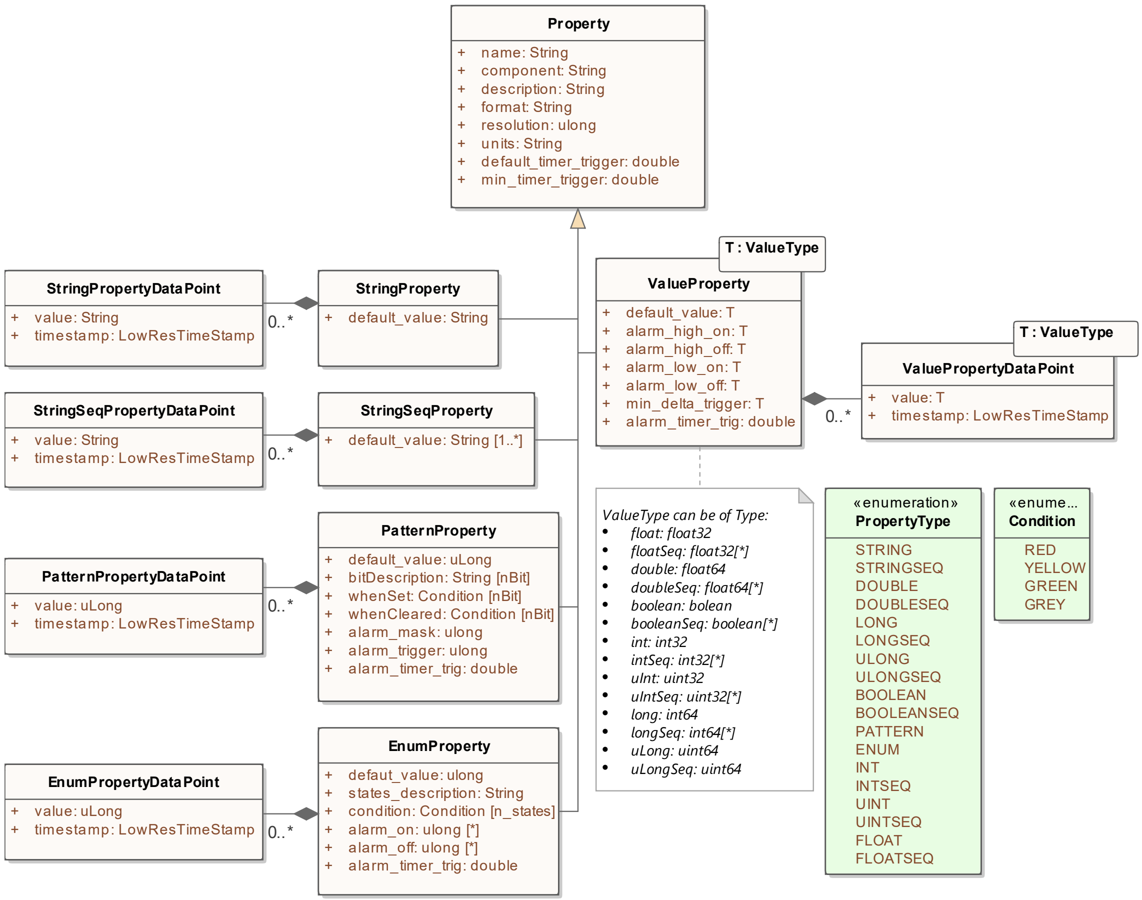

Figure 3: R1/Monitoring Property Data Model.

Table 1: R1/Monitoring Data Model Description

Name	Type	Description
name	String	The display name of the property, e.g. “Current”.
component	String	Name of the Software Component where this property is exposed, should correspond to the monitoring data source it is serving e.g. “PowerSupply1”.
description	String	The description of the property, e.g. “Weather Station Power supply current readback”.
format	String	The format in C-syntax how to display values of the Property, e.g “%9.4f”. (Optional)
resolution	uint64	Bitpattern representing the significant bits of the word carrying the value, see note (1). E.g. "16777215" (corr. to 24 bit resolution). (Optional)
units	String	A string representing the units (normally base SI units or combinations of SI units) of the quantity represented by the property. E.g. “A”
default_timer_trig-ger	float64, seconds	Default time interval between two consecutive monitor-ing requests. (2)
min_timer_trigger	float64, seconds	Minimum allowed time interval (in seconds and frac-tions of second) for values published by timers. (3)
Specific for value properties		

Default_value	<T>	Default value of the property at startup. (Optional)
alarm_high_on	<T>	Above this value alarm is set. (Optional)
alarm_high_off	<T>	Below this value alarm is cleared. (4) (Optional)
alarm_low_on	<T>	Below this value alarm is set. (Optional)
alarm_low_off	<T>	Above this value alarm is cleared. (5) (Optional)
min_delta_trigger	<T>	The minimum allowed threshold for value change (+/-) in value triggers. The property won't report changes below this time interval, irrespective of client requests.
alarm_timer_trig	float64, seconds	The minimum time interval (in seconds and fractions of seconds) for the notification of changes to be used to trigger alarms based on the property value. Clients should not look for value-change trigger alarms faster than this value. (6) (Optional).
Applicable to bit pattern properties (7)		
bitDescription	String[n_state_bits]	One description for each bit, e.g. "On, Remote, Sum Failure, External Interlock, DC Overcurrent, Phase Failure, Not Ready, State Inconsistent, Ramping"
whenSet	Condition[n_state_bits]	For each bit, an associated condition as described in table 3 (Optional). Example *whenSet ="3, 2, 0, 0, 0, 0, 1"*
whenCleared	Condition[n_state_bits]	For each bit, an associated condition as described in table 3 when the bit is cleared. (Optional). Example *whenCleared ="1, 0, 2, 0, 1, 0, 1"*
alarm_mask	uint64	Bit mask for alarm triggering. (Optional)
alarm_trigger	uint64	Defines if an alarm is to be triggered, by the bit value being 1 or 0. (Optional, default is 0)
Applicable to Enum properties		
states_description	String[n_states]	One description for each Enumeration, e.g. "DISABLED, ENABLED, DIAGNOSE, SHUTDOWN, INITIALIZE, ON, OFF "
condition	Condition[n_states]	Condition at each state. (Optional) Example: *condition="3, 2, 0, 0, 0, 0, 1"* Means together with the state list above that the "DISABLED" states triggers a "gray" condition and "ENABLED" a "green" condition.
alarm_on	uint64 [n_states]	all alarm states (if device goes into that states, an alarm will be triggered to the ACADA alarm system). (Optional)
alarm_off	uint64 [n_states]	States when raised alarms should be cleared. (Optional)
Data Point		
value	<T> or ulong, or String	Value of the property measured at the associated timestamp.
timestamp	ACS (OMG Time).	OMG Time, 100-ns since Oct 15, 1582. See note (8) below.

Notes:

(1) For example, a long is represented by 64 bits, but a physical device might deliver a value consisting only of 24 bits. In that case the resolution attribute has only the first 24 bits. The remaining 40 bits shall be ignored by the application. This attribute is useful for example for returning the resolution of analog-digital conversion. The specific usage must be documented case by case.

(2) When a timer is created without requiring a specific time interval, this is the value used. Normally, the value of a property has an intrinsic change rate that should be specified here (in seconds and fractions of seconds).

(3) Independently from the requested time interval or from the frequency of change in case of monitors on changes, no values will be published less than min_timer_trig seconds from a previously published value. This characteristic is

meant to limit the bandwidth and to avoid flooding the system with new values. This characteristic is used for user defined monitors and is independent from the min_timer_trig characteristic, that is equivalently used for archiving property monitoring data.
(4) Note: this value is normally different from alarm_high_on to avoid alarms triggering on and off for small oscillations around the alarm threshold.
(5) Note: this value is normally different from alarm_low_on to avoid alarms triggering on and off for small oscillations around the alarm threshold.
(6) It corresponds to the time the alarm system of ACADA uses to check for the value of the property. It might be the same as default_timer_trig, but depending on the application and on the criticality of the property, they might be different.
(7) Bit patterns and Enums require the provision of tailored XML Schema (XSD files) in addition to component XML files, see [RD7] for an example.
(8) ACS is using OMG timestamps out-of-the-box, but CTA decided to use TAI seconds since 1970-01-01T00:00:00.0 for timestamps. For the time being we use ACS OMG time out of the box, and later expect a way to using TAI in the monitoring stream.

Value properties support the usual primitive types, see Table 2. Some property attributes relate to the concept of 'condition', the meaning of it is described in Table 3.

Table 2: Supported primitive types.

PropertyTypeName	Type
float	float32
floatSeq	float32[*]
double	float64
doubleSeq	float64[*]
boolean:	Boolean
booleanSeq	Boolean[*]
int	int32
intSeq	int32[*]
uInt	uint32
uIntSeq	uint32[*]
long	int64
longSeq	int64[*]
uLong	uint64
uLongSeq	uint64[*]

Table 3: Description of the "Condition" data type

Condition name	Meaning
0: red	implies the property of the component is in an error state.
1: yellow	implies the property of the component is in a warning state.
2: green	implies the property of the component is in an OK state. (works properly)
3: gray	implies the property is not reachable (e.g. the physical device is turned off).

The mapping of this data model to ACS and OPC UA formats is specified in Table 4.

Table 4: mapping of the data model and concrete ACS and OPC UA types.

Property-TypeName	ACS BACI Type (1)	OPC UA Type
float	ACS::ROfloat	Float
floatSeq	ACS::ROfloatSeq	Variant({dataType: "Float", arrayType: opcua.VariantArrayType.Array})
double	ACS::ROdouble	Double
doubleSeq	ACS::ROdoubleSeq	Variant({dataType: "Double ", arrayType: opcua.VariantArrayType.Array})
boolean	ACS::ROboolean	Boolean
booleanSeq	ACS::RObooleanSeq	Variant({dataType: "Boolean ", arrayType: opcua.VariantArrayType.Array})
int	ACS::ROlong	Int32
intSeq	ACS::ROlongSeq	Variant({dataType: "Int32", arrayType: opcua.VariantArrayType.Array})
uInt	ACS::ROlong	UInt32
uIntSeq	ACS::ROlongSeq	Variant({dataType: "UInt32", arrayType: opcua.VariantArrayType.Array})
long	ACS::ROlongLong	Int64
longSeq		Variant({dataType: "Int64", arrayType: opcua.VariantArrayType.Array})
uLong	ACS::ROuLongLong	UInt64
uLongSeq	*Not Supported*	Variant({dataType: "UInt64", arrayType: opcua.VariantArrayType.Array})
String	ACS::ROstring	String
StringSeq	ACS::ROstringSeq	Variant({dataType: "String", arrayType: opcua.VariantArrayType.Array})
Pattern	ACS::ROpattern	UInt64
Enum	*Custom: complexType + enumpropMACRO (2)*	*Enumeration*
(1) For each read-only (RO) type, corresponding read-write (RW) type is also available, see [RD8] (2) The full definition is provided as ACS MIDL file, XML schema and defining file		

3.4.2 Special types

The following needs special types are supported by the monitoring system:

- **Sets of properties of the same type handled atomically.** These are sets of coupled (two or more) properties handled with ACS as a single entity. For example, a pair RA/DEC pointing directions handled simultaneously.
- **Sets of property value and source timestamp handled atomically.** These are properties for which the array element team wishes to transfer the source timestamp as generated in the origin instead of the timestamp generated by ACS upon arrival. *Note:* This is in general not necessary for slow control purposed since the precision is usually within 1s or less. However, may be needed for high-accuracy measurements such as pointing directions. This mechanism shall only be used when high accuracy is needed.

The implementation details of these special types are presented in Table 5.

Table 5: Special property types

PropertyTypeName	Base Type ACS BACI Type (1)	Conventions
propertySet	ACS::ROfloatSeq ACS::ROdoubleSeq ACS::RObooleanSeq ACS::ROlongSeq ACS::ROstringSeq	• The name of the property has to end with the word **Set** • The *description* attribute shall include the information of the field names with the following format: (0=<field 0 entry>, 1=<field 1 entry>, …). Example: description="sky direction (0=RA, 1=DEC)" • The *units* attribute shall include the information of the units of the set. If common, with a single entry (units="deg"), if different as units="(0=deg, 1=rad)"
propertyTime	ACS::ROdoubleSeq	• The name of the property has to end with the word **Time** • The *description* attribute shall include the information of the field names with the following format: (0=<field 0 entry>, 1=<field 1 entry>, …, N=source_timestamp). Example: description=(0=voltage, 1=source_timestamp) • The *units* attribute shall include the information of the units as in the case of the propertySet. Example: units="(0=volt, 1=seconds_in_TAI)"
(1) For each read-only (RO) type, corresponding read-write (RW) type is also available, see [RD8]		

3.4.3 Component and Property Naming Conventions

“Bridge” components and OPC UA Servers shall be named using *CapWords* style, in English (Example: “PowerSupply” and not “POWER_SUPPLY”) and following the nomenclature of the Array Element PBS as applicable.

Properties shall be named using the *camelCase* naming convention, in English, and following the Programming Guidelines conventions [RD12]. Component names do not need to be encoded in the property names because the ACADA monitoring system will record from which component it is recorded. (Example “voltageReadout” and not “VOLTAGE_READOUT” nor “powerSupplyVoltageReadout” etc.).

The only exception to this rule is when there is the intention to represent the hierarchy within the device being monitored in the property. In that case, the hierarchy can be specified via the underscore (“_”) separator. For example: *safety_power_main_400v_currentReadout*.

3.4.4 Other R1/Monitoring data types model and format.

Any other monitoring data types beyond those specified in the previous section must be specified and agreed upon one-by-one in corresponding dedicated ICDs. These can be, for example:

- CCD image data structures
- LIDAR measurements
- FRAM measurements

- Properties to be monitored at high rates (>5 Hz)
- Properties that need to be monitored in a *push* mode.
- …

It is the responsibility of the Array Element team to make a data model proposal for each of these items, which will follow with the specification of a dedicated ICD. All non-standard data structures are transferred to the ACADA Array Data Handler (see [AD6], [RD4] for further details). Commonalities among these data structures will be pursued when possible, after analysing the proposal of the array element teams.

3.5 Array Elements Monitoring Configuration Settings

ACADA can discover and start the monitoring activity of the Array Elements data sources by using the information stored in the ACADA Array Configuration System [AD6], [RD4] and by automatically detecting which data sources are currently online. For that, the Array Element teams must deliver the initial configuration settings (the ACS configuration database (CDB) details or the OPC UA ICD scheme) of the array element properties configuration, which is then afterward be kept under configuration control.

3.5.1 ACS Data Sources

At runtime, ACADA auto-discovers available ACS components via the ACS manager, and identifies the settings for property configuration using the standard ACS CDB entries. In order to enable this setting, the Array Element teams must deliver an initial CDB XML files that have to be validated by the ACADA team.

To monitor the properties of the ACS components, the ACADA team must be provided with an ACS CDB entry for each array element component type and unit to be monitored, containing the following files with the following directory structure:

```
ArrayElement\
     CDB\
          Alma\
                ComponentTypeA\
                                  ComponentTypeAInstance1.xml
                                  ComponentTypeAInstance2.xml
                                          …
                ComponentTypeB\
                                  ComponentTypeBInstance1.xml
                                  ComponentTypeBInstance2.xml
                                          …
                  …
          MACI\
                Components\
                                 Components.xml
          Schemas\
                ComponentTypeA.xsd
                ComponentTypeB.xsd
                 …
```

See RD7 for details and for an ACS example implementing all supported ACS BACI property types in all the supported languages, and [RD5] for further details.

3.5.2 OPC UA Data Sources

At runtime, ACADA can connect directly to OPC UA servers or auto-discover available OPC UA servers by means of Local Discovery Server. OPC UA Local Discovery Server is a special OPC UA service that provides information about all OPC UA servers available. OPC UA Servers may register at Discovery Servers so that they can be discovered by clients. [RD10]. Once connected to an OPC UA server, ACADA identifies the nodes subscription configuration by using a specific ICD configuration file (see Figure 4). In order to have the monitoring process set up, the Array Element teams must deliver such file for its validation by the ACADA team.

3.5.3 Provision of Monitoring Configuration Settings to ACADA

The Array Element teams shall deliver the property configuration files as described in this section.

After the initial delivery and validation of the configuration setting, the monitoring configuration files are kept under configuration control in the corresponding location in the code repository. The CTA personnel will update the configuration settings in the ACADA Array Configuration System according to changes defined by the CTA Change Control Board.

3.6 Properties to be Included in the Monitoring Stream

A concrete and complete list of properties that must be included in the monitoring stream for each CTA Array Element is outside the scope of this document and largely depends on the kind of Array Element and its characteristics. For each Array Element interfacing with ACADA, a specific monitoring ICD (e.g. LST – ACADA Monitoring ICD, MST-ACAD Monitoring ICD) shall specify the complete list of properties monitored by ACADA. In any case, and according to the requirements, at least the following must be included for each Array Element:

- Any pointing information for Array Elements pointing to the sky (B-TEL-0440, B-ACADA-0300). The pointing information has to include both raw pointing (without pointing corrections) and corrected pointing (after bending model corrections) values.
- Current Machine and operational state, according to the generic Telescope State Machine or equivalent for other Array Element types ([AD3], B-ONSITE-0760).
- Camera Pre-Calibration monitoring information ([AD7]).
- Any property that define the condition of the Array Element, (B-ACADA-2280, B-ACADA-2205).

4 Appendix I: Level B requirements applicable to this interface

This interface addresses the following requirements, repeated here from Jama [AD4, AD5] and the ACADA Level-B Requirement Specification document [AD6] for convenience:

- **B-TEL-0440 Structure Pointing Information:** The Telescope Structure must provide its nominal current pointing direction (in Horizon coordinates) on request from the ACADA, independently of the Telescope and Camera States.
- **B-ONSITE-0760 State Change Notification:** The system must notify the ACADA whenever it changes State.
- **B-ACADA-2280 Monitoring interface:** ACADA shall provide a Human Machine Interface which provides hierarchical access to graphical display of monitoring information from the system down to the level of the sub-systems of Controllable Systems.
- **B-ACADA-2205 Monitoring Points:** ACADA shall be able to monitor all hardware related monitoring points provided by Controllable Systems up to an overall maximum of 200k points.
- **B-ACADA-2210 Monitoring Stream Content**: ACADA shall be able to process monitoring information, including metadata, from all Controllable Systems using a common data format.
- **B-ACADA-2215 Monitoring data collection:** ACADA shall collect monitoring information from ACADA internal processes and from all Controllable Systems and Devices on a CTA site in an asynchronous fashion, a configurable rate of up to 5 Hz per monitoring point, regardless of the state of the Array and the Event data acquisition rate.
- **B-ACADA-2220 Monitoring Framework:** ACADA shall monitor the functioning of all Controllable Systems and Devices on a CTA site, with the capability to sample incoming monitoring streams at a configurable rate to give appropriate time sampling for each monitoring point.
- **B-ACADA-2225 Value Change Monitoring:** It shall be possible for ACADA to store monitored quantities according to changes in value based on configurable threshold criteria.
- **B-ACADA-2235 Monitoring system availability:** ACADA by design shall collect and store monitor data for all available systems subject to 2205 at all times with the following availability requirements:
    - Availability >99.75% within any year (A-GEN-3050 specifies 99.5%);
    - Duration of any period of unplanned downtime: <12 hours;
    - Duration of any period of planned downtime: <2 hours in any 24-hour period.
- **B-ACADA-2236 Engineering Data during Technical Operations:** ACADA shall be able to collect engineering data when the Array is in the Technical Operations Mode, and during every Maintenance, Technical or Calibration activities performed to a single Array Element during Scientific Observations.
- **B-ACADA-0300 Telescope Position Monitoring:** ACADA shall monitor telescope pointing directions online, as well as collect a pointing information stream that can

be used to refine the position off-line.

- **B-ACADA-4190 Onboard Auxiliary Instrumentation Data Handling:** ACADA shall handle data produced by acquisition systems of auxiliary instrumentation onboard telescopes during an Observing Night up to five instruments per telescope and a total limit of 100 Mbps per telescope. (Note: This requirement refers to CCD camera data acquisition, high-rate metrology measurements, and the like.)
- **B-ACADA-4193 Environmental Monitoring System Devices Data Handling**: ACADA shall handle data produced by acquisition systems of each environmental monitoring device during an Observing Night up to 10 devices and a total limit of 200 Mbps for the Environmental Monitoring System. (Note: This requirement refers to LIDAR, all sky cameras, etc.)
- **B-ACADA-4195 Array Calibration System Devices Data Handling:** ACADA shall handle data produced by acquisition systems of each array calibration device during an Observing Night up to 10 devices and a total limit of 100 Mbps Array Calibration System.

5 Appendix II: Example ACS Implementation

Example code implementing the data model described in this document can be found in [RD7].

6 Appendix III: Example OPC UA Implementation

```
▼<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  ▶<xs:element name="OPCUA_SERVER">...</xs:element>
  ▶<xs:element name="ServerProperties">...</xs:element>
   <xs:element name="HTML_PORT" type="xs:string"/>
   <xs:element name="OPC_PORT" type="xs:string"/>
  ▶<xs:element name="FolderProperties">...</xs:element>
   <xs:element name="DESCR" type="xs:string"/>
  ▶<xs:element name="NewDevice">...</xs:element>
   <!-- here we insert the Definition node variables  -->
  ▼<xs:element name="NodeVar">
    ▼<xs:complexType>
      ▼<xs:all>
         <xs:element ref="OPC_UA_Data_Type"/>
         <xs:element ref="DefaultValue"/>
         <xs:element ref="Min"/>
         <xs:element ref="Max"/>
         <xs:element ref="Unit"/>
         <xs:element ref="isMonitored"/>
         <xs:element ref="Definition"/>
         <xs:element ref="OPC_UA_node"/>
         <xs:element ref="arrayDim"/>
         <xs:element ref="vectorSize"/>
       </xs:all>
     </xs:complexType>
   </xs:element>
   <xs:element name="OPC_UA_Data_Type" type="xs:string"/>
   <xs:element name="DefaultValue" type="xs:string"/>
   <xs:element name="Min" type="xs:string"/>
   <xs:element name="Max" type="xs:string"/>
   <xs:element name="Unit" type="xs:string"/>
   <xs:element name="isMonitored" type="xs:string"/>
   <xs:element name="Definition" type="xs:string"/>
   <xs:element name="OPC_UA_node" type="xs:string"/>
   <xs:element name="arrayDim" type="xs:string"/>
   <xs:element name="vectorSize" type="xs:string"/>
   <xs:element name="NAME" type="xs:string"/>
```

Figure 4: *A snapshot of an OPC UA ICD schema*

7 Appendix IV: Definitions

This sub-section describes the main definitions used through this document. See the CTA Glossary for further definitions [2].

- **Array Element:** A system deployed on an Array Site which is needed for the scientific operation of CTA and which is interfaced to the ACADA and IPS systems. Array Elements (and in cases of sufficient complexity, such as that of a Cherenkov Telescope, also their sub-systems) are required to implement well-defined common States. An Array Element consists of one or more Controllable Systems and may also include Sensors, plus a software system to manage these elements.
- **Array:** All of the Cherenkov Telescopes and other Array Elements at one of the Observatory Sites.
- **Array Element Control System:** Generalization of Telescope Control System concept. See Telescope Control System.
- **Camera:** All of the hardware and software associated with Cherenkov image detection, digitisation, transmission and pre-processing. A Cherenkov Camera forms part of a Cherenkov Telescope System and has as its principle elements a Camera Unit and software deployed on a Camera Server. Cherenkov Camera may be abbreviated as 'Camera'.

- **Configuration Settings:** data used to configure the parameters and initial settings for an array element, device or software component.
- **Controllable Systems:** A system on an Array Site consisting of hardware elements, a Local Control System (LCS) at the location of the hardware, and a software component (the LCS Controller) running on the on-site data centre. Controllable Systems implement standard interfaces to the ACADA and/or IPS systems and are integrated into Array Elements for the purpose of science operations, but independently controllable for engineering purposes / technical operations.
- **Cherenkov Telescope**: A system composed of a Cherenkov Camera and Telescope Structure which is used to collect and image Cherenkov light from Air Showers.
- **Device:** A mechanical or electrical part of a telescope.
- **DL0 data:** All archival data from the data acquisition hardware/software, transmitted from the ACADA to the DPPS. This is the lowest level of data that are intended for long-term storage in the bulk archive. This includes both camera event data and technical data from other sub-systems, such as non-camera devices or software.
- **Hardware Configuration:** The types and models of devices installed as part of the CTA system, and their operative status.
- **Machine state:** A state that is intrinsic to the hardware state of an Array Element. These states and their transitions are not managed via the ACADA. The following Machine States exist: Off, On, Maintenance. (See [AD4] for more details).
- **On-site Data Centre:** hardware and operating system infrastructure located at a CTA site where software systems such as ACADA are deployed. See [RD3].
- **Operational State:** A logical state of the element with respect to the operations the Array Element is performing. These are the states and their transitions that ACADA is managing to perform the telescope operations. The following Operational States exist: Initializing, Initialized, Standby, Ready, Observing, Fault, Technical. (See [AD4] for more details).
- **Sensor:** A data collecting instrument plus an associated software component which delivers information to ACADA via a standard interface. The input is a physical entity like light, temperature, etc.
- **Software Component:** A software component (referred to as “component” in this document) is a software package, service, resource, or module that encapsulates a set of related functions or data. A software component can be deployed independently and is subject to composition by third parties.
- **R1 Data:** on-site raw data which is streamed from the CTA Array Elements to the ACADA system.
- **State**. A State represents a situation where some invariant condition holds; this condition can be static (waiting for an event) or dynamic (performing a set of activities). The behaviour of a system can be described through its state at different points in time. When a system is in a given state, it can perform different actions or do a transition to another state so that other actions can be performed.
- **Structure:** All of the hardware and software associated with a single optical telescope capable of pointing to different parts of the sky and collecting light on to a Cherenkov Camera. A Telescope Structure forms part of a Cherenkov Telescope System. Telescope Structure may be abbreviated as 'Structure'.

- **Sub-state.** A state within another state, where transitions can be managed and triggered internally by the system according to external conditions (e.g. available time inside the current state).
- **Telescope Target:** A fixed celestial direction, horizon/equatorial system direction, or defined trajectory to which the optical axis of a Telescope should be aligned.
- **Telescope Control System (TCS)**: Software deployed at the central computing cluster providing the high-level control interface to the ACADA for an individual Cherenkov Telescope (see [RD1]).
- **Telescope Manager**: The highest level component in the TCS, responsible of receiving commands from ACADA and distributing them to other components in the TCS.