

Naming REST API – brief introduction

by lars.johansson@ess.eu, ICS Software

Table of Contents

Naming REST API – brief introduction.....	1
Introduction.....	1
Background.....	2
Concepts & Terminology.....	3
The purpose of Naming.....	3
What is a Name?.....	3
Visualization – names and structures.....	5
Lifecycle for names and structures.....	6
Vocabulary.....	9
REST API.....	12
Data – Access.....	12
Endpoints.....	12
Authorization.....	12
About searching.....	13
About data.....	13
Frequently Asked Questions (FAQ).....	16
Reference.....	16

Introduction

This document will give brief introduction to Naming REST API. First set of sections, such as Background and Concepts & Terminology, explains purpose of Naming and gives visualization of names & structures. Next set of sections attempts to give more practical information such as how to use Naming REST API, what data that is sent and received and what it means together with some questions and answers.

This document will mention persistence and storage but is not about it. Its focus is on REST API.

Background

It's been noticed that there is more and more focus on ability to service other applications, systems, services with information and act as a microservice. This means more focus on REST API. This includes having proper foundation for future with ability to support implementation of Naming convention.

Brief background on storage of data

- 1 entity for ESS name
- 1 entity per System structure level – System group, System, Subsystem (3 tables)
- 1 entity per Device structure level – Discipline, Device group, Device type (3 tables)
- *storage corresponds to data in real world. This helps understanding and maintainability of storage and application.*

Brief vocabulary

- names <---> ESS names
- structures <---> System structure, Device structure

Noteworthy

- REST API with Swagger UI
- No ordinary UI available
- Json format used for data that is sent and received
- Not about authentication/authorization. Intention is to have division of possibilities, e.g. visitor / user / administrator.

In various places are framed boxes with information that is noteworthy and may help understanding

Note!

Text that is noteworthy, summary, help & more

Concepts & Terminology

Concepts & Terminology are summarized below with *text in italic*.

The application is commonly referred to as *Naming*, *Naming tool*, *Naming convention tool*. It is a web application that includes a REST API for read purposes and it is connected to a storage of names and structures. Naming is the most common way to refer to both application and system as a whole.

The application is split into a backend part and a frontend part, known as *Naming backend* and *Naming frontend*.

The purpose of Naming

Handle Naming of ESS wide physical and logical devices according to ESS Naming Convention

What is a Name?

<i>ESS Name</i>	<i>System structure</i>	<i>Device structure</i>
	<i>Which part of the facility does the device provide service to?</i>	<i>What kind of service does the device provide?</i>
<i>Must refer to System structure</i>	<i>1 System Group</i>	<i>1 Discipline</i>
<i>May refer to Device structure</i>	<i>2 System</i>	<i>2 Device Group</i>
<i>May have index for instance</i>	<i>3 Subsystem</i>	<i>3 Device Type</i>

System structure and Device structure are hierarchies with 3 levels each. A 3rd level entry refers to 2nd level entry that refers to 1st level entry. A name must refer to System structure (arbitrary level) and may refer to Device structure (level 3). If Device structure is referred to, then index is to be set.

An entry, for name and structure, has a common set of attributes such as uuid, name, mnemonic, description, comment, when requested and by whom, when processed and by whom, etc.

Each entry is identified by its universal identifier, also known as uuid. It is the common denominator to keep track of an entry through its lifecycle, e.g. when the entry is created, updated, deleted. An entry usually has another attribute called *mnemonic* which is a short name that is to be unique in its namespace. This attribute is called *mnemonic* for an entry in System structure or Device structure. For a names entry, this attribute is called *index*.

In addition, there is an attribute called name equivalence or mnemonic equivalence. This is derived from name or mnemonic by taking similar-looking characters into account and helps to ensure that name and mnemonic is unique within its namespace.

Mnemonic equivalence

o, O, 0 considered same from equivalence point-of-view

i, I, l, L, 1 considered same from equivalence point-of-view

leading 0 numerical characters removed

Uniqueness of an entry

Each entry has a unique uuid that acts as key in its line of history. This applies for both name and structure entries.

An entry in names, System structure, Device structure must have attributes such that it is unique in its namespace at any given time. This is achieved with proper values for mnemonic and index.

Namespace for a name entry is all valid names.

Namespace for structure entry is its hierarchy.

Rules for names and structures

Structures

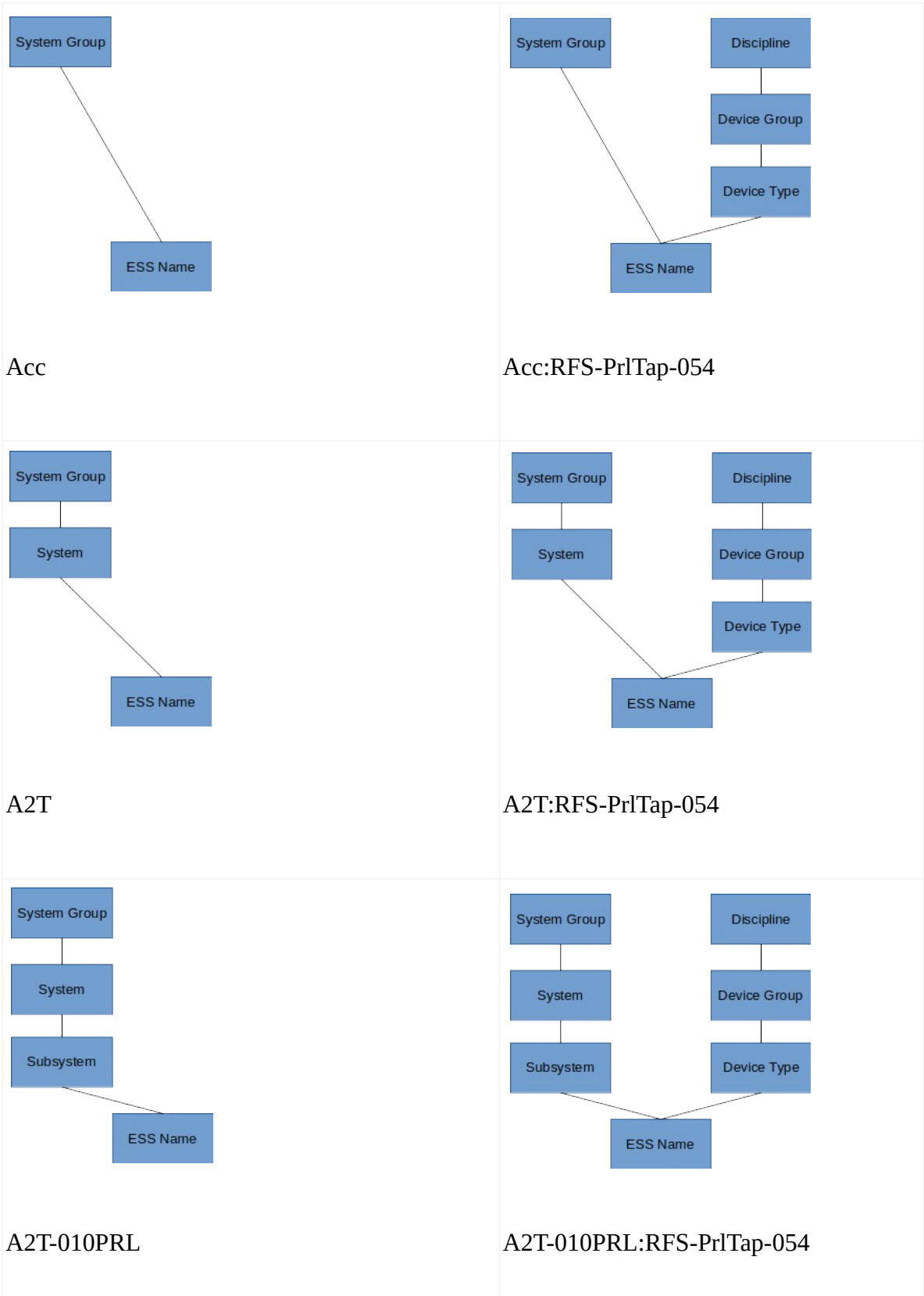
System, Subsystem, Discipline, Device Type must have mnemonic.

System Group may have mnemonic.

Device Group must not have mnemonic.

<i>ESS Name</i>	<i>=</i>	<i>System Group</i>				
		<i>System Group</i>	<i>+</i>	<i>Device Type</i>	<i>+</i>	<i>Index</i>
		<i>System</i>				
		<i>System</i>	<i>+</i>	<i>Device Type</i>	<i>+</i>	<i>Index</i>
		<i>Subsystem</i>				
		<i>Subsystem</i>	<i>+</i>	<i>Device Type</i>	<i>+</i>	<i>Index</i>

Visualization – names and structures



Lifecycle for names and structures

An entry for names is valid when it is created and saved. The entry may be modified multiple times. When entry is deleted, it has reached its end-of-life and may no longer be modified.

An entry for System structure and Device structure is valid when it is created and saved. The entry may be modified multiple times. When entry is deleted, it has reached its end-of-life and may no longer be modified.

The whereabouts in the lifecycle for names and structures is handled with attribute deleted. When an entry is deleted (names, structures), it has reached its end-of-line.

A change for an entry is handled such that a new entry is created. Any prior entry is considered obsolete.

Default behavior

Default behavior for Naming is to handle valid entries. Therefore old values are excluded unless history requested. This means that old values can not be browsed but must be requested. Valid entries are available for browsing.

Examples

Purpose of examples is to show lifecycle of names and structures. Therefore some columns are not shown, e.g. references to parents. A name has references to parents in System structure and Device structure. A structure entry has reference to a parent. Entries are grouped per uuid and shown in ascending order.

In addition, examples contain information from both user side and storage side.

Names

lifecycle	uuid	name	description	deleted
obsolete	a	A2T-010PRL:RFS-PRLTap-052	comment	false
obsolete	a	A2T-010PRL:RFS-PRLTap-053	comment	false
active	a	A2T-010PRL:RFS-PRLTap-054	comment	false
obsolete	b	A2T-010PRL	comment 1	false
active	b	A2T-010PRL	comment 2	true
active	c	A2T	comment	false

Structures - e.g. System

lifecycle	uuid	mnemonic	description	deleted
pending	m	A0T	comment	false

obsolete	n	A1T	comment	false
active	n	A1T	comment	false
obsolete	o	A2T	comment	false
active	o	A2T	comment	false
obsolete	o	A3T	comment	false
obsolete	o	A3T	comment	false
obsolete	p	A4T	comment	false
obsolete	p	A4T	comment	false
obsolete	p	A5T	comment	false
obsolete	p	A5T	comment	false
obsolete	p	A5T	comment a	false
active	p	A5T	comment a	false
obsolete	q	A6T	comment	false
active	q	A6T	comment	false
obsolete	q	A6T	comment	true
obsolete	q	A6T	comment	true
obsolete	r	A7T	comment	false
obsolete	r	A7T	comment	false
obsolete	r	A7T	comment	true
deleted	r	A7T	comment	true

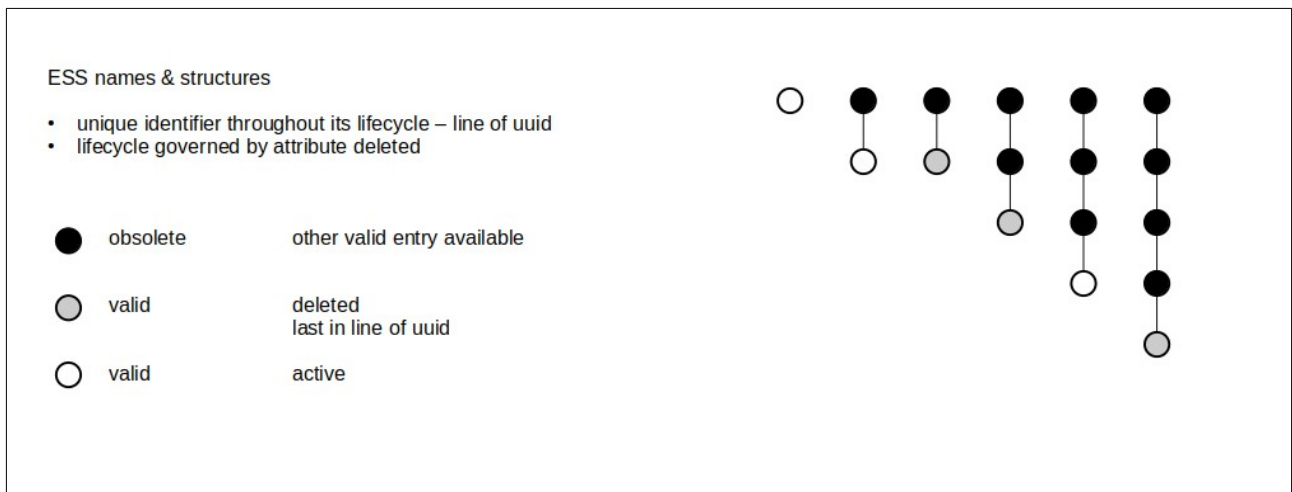
Each time a modification is done, it will result in a new entry.

There is no approval process, neither for Name nor for Structure.

When a modification is done, any prior entry is considered obsolete.

An entry with attribute deleted set to true has reached its end-of-line.

Rows with gray background color are considered history and are excluded unless history is requested. Rows become history when there is a more recent entry. History for an entry may be requested through its uuid.



The lifecycle for names and structures is handled with attribute deleted as shown in examples above.

deleted – true, false

A value that no longer is valid corresponds to OBSOLETE.

Note that application needs to set values for attribute deleted when an entry is created, modified or deleted.

Vocabulary

Vocabulary below with headlines with *text in italic*.

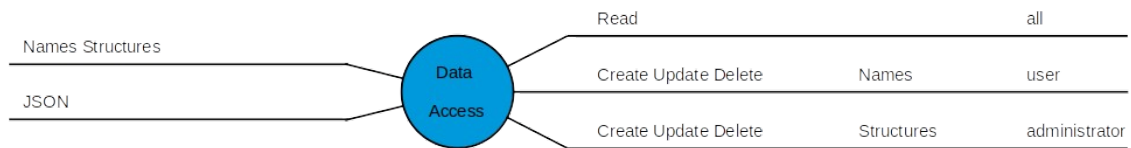
What	Description	Example
<i>Names & Structures</i>		
Name	Name for ESS physical and logical device	A2T-010PRL:RFS-PRLTap-054
System structure	Which part of the facility does the device provide service to?	A2T-010PRL
Device structure	What kind of service does the device provide?	Cryo-TT
System group	System structure level 1	Accelerator (Acc)
System	System structure level 2	Accelerator to Target (A2T)
Subsystem	System structure level 3	01 Phase Reference Line (010PRL)
Discipline	Device structure level 1	RF Systems (RFS)
Device group	Device structure level 2	Phase Reference Line
Device type	Device structure level 3	Phase Reference Line Tap (PRLTap)
<i>Concepts & Terminology</i>		
ESS Naming Convention	Rules for naming ESS Systems and Devices and Components in EPICS-based control system.	https://chess.esss.lu.se/enovia/link/ESS-0000757/21308.51166.45568.45993/valid
Equivalence	Derived from name or mnemonic by taking similar-looking characters into account and helps to ensure that name and mnemonic is unique within its namespace.	o, O, 0 - considered same from equivalence point-of-view i, I, l, L, 1 - considered same from equivalence point-of-view leading 0 numerical characters removed
Lifecycle of names and structures	The lifecycle of ESS name and structure entries. Each entry has a unique identifier throughout its lifecycle. The lifecycle is governed by attribute deleted.	An entry that is deleted may no longer be updated (or revived).
Line of uuid	A collection of ESS name or structure entries that share the same identifier and together make up an entry's history.	

Namespace	Line of uuid from top level to bottom level, for system structure and device structure, respectively. An index or a mnemonic or mnemonic equivalence may exist only once in a namespace for entries that are not deleted. Namespace for a name entry is all valid names. Namespace for a structure entry is its hierarchy.	
Rules for names and structures	System, Subsystem, Discipline, Device Type must have mnemonic. System Group may have mnemonic. Device Group must not have mnemonic.	
<i>REST API methods</i>		
POST	Http request method for create	create
GET	Http request method for read	read, find, search, equivalence, exists
PUT	Http request method for update	update
DELETE	Http request method for delete	delete
<i>REST API schemas</i>		
Name element	A collection of fields that represent an ESS name entry (comprehensive). From server to client.	
Name command element	A collection of fields that represent an ESS name entry (minimum). From client to server. Purpose to simplify communication client to server.	
Structure element	A collection of fields that represent an ESS system structure or device structure entry (comprehensive). From server to client.	
Structure command element	A collection of fields that represent an ESS system	

	structure or device structure entry (minimum). From client to server. Purpose to simplify communication client to server.	
<i>REST API fields (sub-selection)</i>		
Type	Kind of structure.	SYSTEMGROUP, SYSTEM, SUBSYSTEM, DISCIPLINE, DEVICEGROUP, DEVICETYPE
Index (Instance index)	A string. May be considered mnemonic for a name. To distinguish devices of the same type in the same system. Two different set of rules for index are identified for the <i>Scientific</i> and <i>P&ID</i> disciplines.	
Mnemonic	A set of characters and numbers to identify an entry in system structure and device structure.	
Mnemonic equivalence	A mnemonic with rules for equivalence applied.	
Deleted	To show if entry is deleted in its line of (uuid) entries.	true, false
<i>REST API media type</i>		
application/json	Supported	
application/xml	Not supported	
<i>Authentication & authorization</i>		
Authentication	ESS username and password	
Authorization	None (read), User, Administrator	None – Read-only access User – All operations for names. Administrator – All operations for names and structures.

REST API

Data – Access



Endpoints

Kind of	(Base) Path	Description
Names	/api/v1/names	Handle names data for Naming application
Structures	/api/v1/structures	Handle structures data for Naming application
Process variable names	/api/v1/pvNames	Handle validation of process variable names data for Naming application
Convert	/api/v1/convert	Handle conversion of names and structures data from one format to another for Naming application
Report	/report	Provide reports for Naming application
Healthcheck	/healthcheck	Perform healthcheck for Naming application

Authorization

HTTP method	Authorization	Description
GET	Not required	Read
POST, PUT, DELETE	User	Create, Update, Delete (names)
POST, PUT, DELETE	Administrator	Create, Update, Delete (names, structures)

About searching

Exact match

- Is exact match = no search

Search

- Default behavior is exact match
- No regex
- Two additional characters may be used to help search and may be written anywhere in search string to give regex-like behavior
 - `_` underscore, 0 or 1 occurrences of any character
 - `%` percent, any number of any character
 - e.g.
 - A2T-010PRL:RFS-PRLTap-054
 - A2T-010PRL:RFS-PRLTap-0_ will not give match
 - A2T-010PRL:RFS-PRLTap-0__ will give match
 - A2T-010PRL:RFS-PRL% will give match

About data

What is sent to and received from Naming REST API

- a string
- json

Examples

Name and NameElement

A2T-010PRL:RFS-PRLTap-054

System structure

- Accelerator
- Accelerator to Target
- 01 Phase Reference Line

Device structure

- RF Systems
- Phase Reference Line
- Phase Reference Line Tap

Index

- 054

Json

```
{
  "uuid": "07bce0ae-0947-47c8-941e-cc76678fd29a",
  "parentSystemStructure": "c2fce615-ed5d-40f9-8fb5-0b91502536e5",
  "parentDeviceStructure": "bb1e68a6-e233-4595-ae88-f9186b6760c6",
  "index": "054",
  "description": null,
  "systemStructure": "A2T-010PRL",
  "deviceStructure": "RFS-PRLTap",
  "name": "A2T-010PRL:RFS-PRLTap-054",
  "status": "APPROVED",
  "deleted": false,
  "when": "2017-10-20T12:53:27.229+00:00",
  "who": "johannorin",
  "comment": null,
}
```

Rules

- A name must have exactly one system structure parent, either systemgroup or system or subsystem
- A name may have device structure parent, devicetype

In name above

- subsystem uuid refers to subsystem *01 Phase Reference Line* that in turn refers to system *Accelerator to Target* that in turn refers to system group *Accelerator*
- device type uuid refers to device type *Phase Reference Line Tap* that in turn refers to device group *Phase Reference Line* that in turn refers to discipline *RF Systems*

Since name above was created, implementation of Naming was changed so that description is mandatory.

Structure and StructureElement

Accelerator to Target A2T

Json

```
{
  "uuid": "e67a497c-9c55-4942-97fc-700c8ec56031",
  "type": "SYSTEM",
  "parent": "4262e1e7-2444-412e-83d7-aeabf58262c6",
  "mnemonic": "A2T",
  "ordering": null,
  "description": "Accelerator to Target. The Accelerator to Target  
Station interface including the dogleg",
  "mnemonicPath": "A2T",
  "level": 2,
  "status": "APPROVED",
  "deleted": false,
  "when": "2016-07-04T10:06:38.873+00:00",
  "who": "danielpisofernandez",
  "comment": "Approved by Daniel Piso",
}
```

Frequently Asked Questions (FAQ)

Topics / questions / answers, no particular order

- Capabilities of REST API
 - All operations in Naming are available in REST API
- Json format used for data that is sent and received
 - Data is provided or received as json, as single entry or arrays of entries,
- Retrieval available with exact match and search. Default for search is exact match but may be adjusted for true search. Regex is not available. There are instead two special characters to use for search and regex-like behavior.
 - `_` underscore, 0 or 1 occurrences of any character
 - `%` percent, any number of any character
- Retrieval available with abilities
 - search on individual fields
 - sorting
 - pagination
- Index for a name
 - Field is alphanumerical. Existing names usually have index. Values include alphabetical, numerical, alphanumerical. When index is to be set, it is to be set explicitly and not auto-generated.
- History
 - History consists of lines of uuid. The line of uuid is not broken in retrieving history. If combination of parameters is found in name or structure entries, the entire lines of uuid are returned.
- Validation
 - Ability to validate modifying operation before invoking modifying operation. Modifying operation internally use same validation.

Reference

Naming convention

- <https://chess.esss.lu.se/enovia/link/ESS-0000757/21308.51166.45568.45993/valid>