

# SEP 5 - Instrument

Tiago Coutinho

27/08/2009

## Contents

### Introduction

This document describes a sardana enhancement proposal (SEP) for a new “Instrument“ feature in the Device Pool. The main idea is to be able to associate each pool element (motor, counter, OD, etc.) with a certain physical instrument. Instruments should be hierarchical in nature and should have an instrument type associated to them.

### Motivation

It would be very useful to be able to rapidly identify to which physical instrument a pool element belongs to. For example, motor 'th' belongs to instrument '/Hubber2C(Diffrac2C)'.

With this information, a GUI can be built providing a visual aid associating each pool element to its instrument.

This information will also be useful when storing data for some procedure (ex.: scan). Post data analysis could them benefit from having this information, specially when it is done sometime after the experiment and in another physical location.

### Use cases

- Identify the instrument for each pool element
- Display the list of all elements of one instrument
- store instrument metadata in a procedure context (ex.: scan). For example, in the NeXUS file format the instrument is an important part of the specification

### Requirements

- Each instrument should be a pool element.
- Unlike other pool elements, there should be no tango device associated with it (at least so far).
- An instrument should have an instrument type (or instrument class) associated with it.

- An instrument can be a part of another instrument

## Pool API

In order to support the Instrument feature, the Pool Tango device API needs to be extended. Here is a list of the new attributes and commands:

### InstrumentList

InstrumentList is a new attribute that contains a list of existing instruments. From this list it should be possible to extrapolate the hierarchy of instruments and also determine the instrument type of each instrument.

- Type: DevString
- Format: SPECTRUM
- Access: READ

Each string item in the list should represent a single instrument. The instrument name should contain the complete instrument hierarchy plus the instrument type:

```
<full instrument name>'(<instrument type>')
```

Example:

```
/Mono1(NXMonochromator)
/Mono1/Mirror1(NXMirror)
```

### CreateInstrument

CreateInstrument should be a new tango command to create a new instrument in the device pool:

```
DevVoid CreateInstrument(DevVarStringArray)
```

Where the string array parameter should be a sequence of two strings:

```
<instrument type>, <full instrument name>
```

Example:

```
CreateInstrument("NXMirror", "/Mono1/Mirror1")
```

It is an error to:

- Create an instrument with the same name as an existing instrument
- Create an instrument with the name not following the instrument name syntax
- Create an instrument without previously creating its parent instrument

## DeleteInstrument

DeleteInstrument should be a new tango command to delete a previously created instrument:

```
DevVoid DeleteInstrument(DevString)
```

Where the string parameter should be the full instrument name. Example:

```
DeleteInstrument("/Mono1/Mirror1")
```

It is an error to:

- delete an inexistent instrument
- delete an instrument with pool elements associated with it
- delete an instrument with sub instruments

## Element API

Each pool element will need to have an additional attribute 'Instrument':

- Type: DevString
- Format: SCALAR
- Access: READ\_WRITE

This attribute will allow to configure and determine to which instrument a pool element belongs. It is an optional feature. By default it should contain the empty string. When configured it should obey the following syntax:

```
<full instrument name>'(<instrument type>')
```

Example:

```
/Mono1/Mirror1(NXMirror)
```