A brief introduction to the Tango Controls Concepts
by
Matteo Canzari
INAF - Osservatorio Astronomico d'Abruzzo
Summary

- Tango Controls Framework concepts
- Tango Community and Tango Collaboration
- The role of INAF in Tango
- INAF people involved
- INAF projects interested in TANGO
**Microservice** is an approach to developing a single application as a suite of **small services**, each running in its **own process** and communicating with lightweight mechanisms, often an HTTP resource API.

---

**A monolithic application puts all its functionality into a single process...**

...and scales by replicating the monolith on multiple servers.

---

**A microservices architecture puts each element of functionality into a separate service...**

...and scales by distributing these services across servers, replicating as needed.
The actor model in computer science is a mathematical model of concurrent computation that treats "actors" as the universal primitives of concurrent computation. In response to a message that it receives, an actor can: make local decisions, create more actors, send more messages, and determine how to respond to the next message received. Actors may modify private state, but can only affect each other through messages (avoiding the need for any locks).

Proposed in 1973 by Carl Hewitt and others
1. You don’t need to know CORBA to work with TANGO
2. CORBA is the acronym for Common Object Request Broker Architecture and it is a standard defined by the Object Management Group (OMG)
3. CORBA enables communication between software written in different languages and running on different computers
4. CORBA applications are composed of many objects; objects are running software that provides functionalities and that can represent something in the real world
5. Every object has a type which is defined with a language called IDL (Interface Definition Language)
6. An object has an interface and an implementation: this is the essence of CORBA because it allows interoperability.
7. CORBA allows an application to request an operation to be performed by a distributed object and for the results of the operation to be returned back to the application making the request.
8. CORBA is based on a Remote Procedure Call model
9. The TANGO Device is a CORBA Object
10. The TANGO Device Server is a CORBA Application

0mq (or ZeroMQ, ØMQ, ZMQ)

- high-performance asynchronous messaging library
- used in distributed or concurrent applications
- provides a message queue
TANGO and synchronous & asynchronous communication

Tango uses:

- **CORBA** for *synchronous* communications (messages between actors)
- **ØMQ** for *asynchronous* communication (events)

**TANGO – CAN ZMQ REPLACE CORBA?**

*Proceedings of ICALEPCS2013, San Francisco, CA, USA*


<table>
<thead>
<tr>
<th>server to client</th>
<th>data</th>
<th>throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java to java</td>
<td>8 bytes</td>
<td>14.2 k Ev/s</td>
</tr>
<tr>
<td>C++ to C++</td>
<td>8 bytes</td>
<td>230 k Ev/s</td>
</tr>
<tr>
<td>C++ to C++</td>
<td>1 kbyte</td>
<td>122 k Ev/s</td>
</tr>
<tr>
<td>C++ to C++</td>
<td>1 Mbyte</td>
<td>1.2 k Ev/s</td>
</tr>
</tbody>
</table>
What is TANGO

Tango is an open-source device-oriented controls toolkit for controlling any kind of hardware or software and building SCADA and DCS.

Born in 1999

Mostly used in Synchrotron, but also in industry and now in radio astronomy.

Connecting things together

What is Tango Controls?
A free open source device-oriented controls toolkit for controlling any kind of hardware or software and building SCADA systems.

Why choose Tango Controls?
Because it is easy to use, flexible, and highly scalable. It provides a complete set of features for controlling equipment and a lot of services for managing systems.

How to use Tango Controls?
Just download it and install it. Then reuse or write a device server, deploy and marvel at how it works!
Tango = \textit{actors} + \textit{microservices}

- Tango is based on the concept of \textit{Distributed Devices}
- This is an implementation of the \textit{Actor Model}
- Device servers implement \textit{Microservices}
- Tango = \textit{Actors} + \textit{Microservices}
- Actors + Microservices are in \textit{fashion} today
- TANGO is based on \textit{MODERN} concepts!
Tango basic concepts

- Device
- Attribute
- Command
- Pipes
- Properties
- State Machine
- Events
- Client API
- C++, Python, Java Bindings
- Tools
- Server API
- Device Server Model
- Device Server
- Database

**Tango Controls Concepts**
Tango **Devices** are the **objects** which implement the microservices of a Tango System. Devices can be **any piece of hardware or software**.

Examples: *Modbus controller, motor, powersupply, camera, data analysis service, ...*

- **Devices** belong to a **Device Class** and are in a **Device Server**. They are stateful i.e. have State. Accessed via a common API. Have a unique 3 field name (D/F/M)

- **Device Classes** can be implemented in **Python, C++ or Java**

- Devices can be built on top of other Devices
Attribute concept #2

**Tango Controls Concepts**

- Device
- Attribute
- Command
- Pipes
- Properties
- State Machine
- Events
- Client API
- C++, Python, Java Bindings
- Tools
- Server API
- Device Server Model
- Device Server
- Database
- Device Class
**Attribute concept #2**

- Tango **Attributes** represents the **data fields** a Device wants clients to **Read** or **Write** or receive **Events**.

- Examples: *modbus register, interlock bit, read/set value, spectrum, image, ...*

- Attributes can be **scalar**, **spectrum** (1D) or **images** (2D) and are **self describing** (units, min, max, alarms, display,...)

- **All Device data** should be provided as **attributes** (well almost all!). Attributes can be read one by one or many. Device developers have hooks for optimising attributes. Attributes read/write check the State Machine.
Command concept #3

**Tango Controls Concepts**

- Device
- Attribute
- Command
- Pipes
- Properties
- State Machine
- Events
- Client API
- C++, Python, Java Bindings
- Tools
- Server API
- Device Server Model
- Device Server
- Database
Tango **Commands** are the **actions** of a Device the clients needs to execute. Commands can change the State of a Device (Attributes don’t)

Examples: *On, Off, Calibrate, Move, ...*

Commands take **one input** and **one output** parameter. Parameters can be of any of the 20+ Tango data types.

Commands always **check** the **State Machine** before and after execution (Attributes only before).
Pipe concept #4

**Tango Controls Concepts**

- Device
- Attribute
- Command
- Pipes
- Properties
- State Machine
- Events
- Client API
- C++, Python, Java Bindings
- Tools
- Server API
- Device Server Model
- Device Server
- Database
- Device Class
Pipe concept #4

- Tango **Pipes** are **data streams** or channels for exchanging a stream of any Tango data type. Data types can be sent individually or grouped together in a Blob.

- Examples: *scanning data stream of mixed data types*

- Also used to circumvent the fixed data type set of Tango by sending mixed data types or a JSON blob.

- **DO NOT** only use Pipes (except in special cases)! Pipes were added to Tango in V9.x

- **DO** use Attributes!
Properties concept #5

**Tango Controls Concepts**

Device

Attribute

Command

Pipes

Properties

State Machine

Events

Client API

C++, Python, Java Bindings

Tools

Server API

Device Server Model

Device Server

Database

Device Class
Tango **Properties** are **data stored** in the database and used to **configure** Devices at **startup**. Properties can be any Tango Data types. Properties enable Device Classes to be generic. Properties are edited with Jive usually.

- **Examples**: *channel address, initial or current settings, sub-device names, ...*

- Changes to Properties can be persisted in the Database.

- **DO NOT** exit if Properties are wrong!

- **DO use** sensible default Properties!
State machine concept #6

**Tango Controls Concepts**

- Device
- Attribute
- Command
- Pipes
- Properties
- State Machine
- Events
- Client API
- C++, Python, Java Bindings
- Tools
- Server API
- Device Server Model
- Device Server
State machine concept #6

- All Tango Devices have **State**. Tango States are limited to 14 discrete values. Each Tango Device Class **State Machine** implements the state transitions.

- **ON, OFF, CLOSE, OPEN, INSERT, EXTRACT, MOVING, STANDBY, FAULT, INIT, RUNNING, ALARM, DISABLE, and UNKNOWN**

- State is a very powerful mechanism for **protecting** Devices and for communicating changes to clients or servers.

- **DO NOT** ignore State!

- **DO set a default State!**
All Tango Devices are implemented by a **Device Class**. The Device Class implements a **generic** Device **behaviour**. Properties are used to configure the specific Device.

- Examples: *PowerSupply, SerialLine, Polly*

- Device Server developers are in fact developing Device Classes.

- C++ developers have an extra class to develop – the so-called DeviceClassClass e.g. MyPowerSupplyClass. This is uses one of the Gang of Four patterns. Python and Java have only the DeviceClass.
**Tango Controls Concepts**

- Device
- Attribute
- Command
- Pipes
- Properties
- State Machine
- Device Class
- Database
- Server API
- Device Server Model
- Device Server
- C++, Python, Java Bindings
- Tools
- Client API
- Events

Events concept #8
Tango **Events** are a **Pub-Sub** communication between clients and servers. Events are only supported for **Attributes** and **Pipes**. Multiple Event types are supported – **Change, Periodic, Archive, User, ...**

- Examples: *Send Event if Attribute changes by x%*

- Events use **ZeroMQ** + are the most efficient way to communicate. Events rely on Polling to be triggered.

- Events are configured in the database or code

- Tango implements a Polling algorithm to trigger events.

- **DO NOT** only read Attributes, use Events
Tango **Database** implements the **Configuration** and **Naming** Service for Tango. It can also persists **settings** values.

**Examples:** *configuration properties, export/import*

Tango Database is implemented as a Device Server. Clients use the Tango Client API and Data Types to access the Database. Only **MySQL** (or **MariaDB**) is supported. A (non-official) version exists for **SQLite** + yaml. It also possible to use a **single file** (not recommended).

Database is **only fixed** address `TANGO_HOST=host:port` or `/etc/tango.rc` environment variable.

**Multiple** Databases supported.
Device server concept #10

**Tango Controls Concepts**

- Device
- Attribute
- Command
- Pipes
- Properties
- State Machine
- Events
- Client API
- C++, Python, Java Bindings
- Tools
- Server API
- Device Server Model
- Database
- Device Class
The admin device helps in creating and managing the devices, i.e. restart device, kill the device server (the process), creating polling mechanism and so on.
Device server model concept #11

**Tango Controls Concepts**

- Device
- Attribute
- Command
- Pipes
- Properties
- State Machine
- Events
- Client API
- C++, Python, Java Bindings
- Tools
- Server API
- Device Server Model
- Device Class
- Database
- Device Server
Tango Developers map

Control System Java Development

- mTangoSDK
- tango-js
- JYSE
- mTangoREST.server

mTangoREST.server

- ATKPanel
- DeviceTree
- TangoATK

Synoptic Viewer
- JDraw

Java Device API

- Communication Layer (CORBA / ZMQ)
- DServer Device(s)
- Devices (C++, Java, Python)

POGO
Many **tools** exist for Tango to configure + test Devices

- **Configuration tool**: Jive (alternatives are dsconfig, Waltz)

- **Start/stop control system**: Astor (alternatives system 5, Supervisord)

- **Test + monitor Device tool**: ATKPanel (alternatives are PyTango, Taurus, Waltz, WebJive, Jyse, ...)

- **Archiver**: HDB++ (alternatives are HDB)

- **View logs**: Logviewer (alternatives Elasticsearch)

- **Alarm Handling**: Panic
Read the Documentation!

Tango Controls documentation

Welcome to Tango Controls documentation!
- Authors
- Overview
- Installation
- Getting Started
- Developer’s Guide
- Tools and Extensions
- Administration
- Tutorials and How-Tos
- Reference

The documentation is organized in the following categories (some of them overlap):

- **Overview** will give you a quick overview of what Tango Controls is, its origins and who uses it. Start reading here.
- **First steps** will lead you through getting started with Tango Controls. This category includes an overview of Tango Controls concepts, procedures for installation and starting the system as well as **Getting started** tutorials.
- **Developer’s Guide** documents the tools and information for Developers needed for development of Device Servers and client applications.
- **Administration** section is important mainly for System Administrators. However, it may provide some information for both End Users and Developers, too. It contains useful information on Tango Control system deployment, startup and maintenance.
- **Tools and Extensions**. Tango comes with rich set of command line tools, graphical toolkits and programming tools for management, developing graphical applications and connecting with other systems and applications. All, End Users, Developers and System Administrators should take a look at the toolkits’ manuals.
- **Tutorials and HOWTOs** give step by step guidance and teach you how to work with Tango Controls.
- **Table of Contents** provides access to all documents.
- If you want to contribute to the documentation please read the document How to work with Tango Controls documentation and the Documentation workflow tutorial.

Indices and tables

Jun 19, 2016 – Oct 28, 2020

Contributions to master, excluding

<table>
<thead>
<tr>
<th>tango-doc</th>
</tr>
</thead>
<tbody>
<tr>
<td>The official documentation for Tango controls</td>
</tr>
<tr>
<td>TeX</td>
</tr>
<tr>
<td>26</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>57</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>Updated yesterday</td>
</tr>
</tbody>
</table>

https://tango-controls.readthedocs.io/
# Tango Controls Core Projects

The official place for Tango Controls core projects

http://www.tango-controls.org  info@tango-controls.org

## Pinned repositories

<table>
<thead>
<tr>
<th>Repository</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TangoTickets</td>
<td>The official place to create an issue/ticket related to Tango when it impacts several repositories or you don’t know where to create your issue</td>
</tr>
<tr>
<td>cppTango</td>
<td>TANGO kernel C++ implementation</td>
</tr>
<tr>
<td>pytango</td>
<td>Python binding to Tango C++</td>
</tr>
<tr>
<td>rest-api</td>
<td>Tango REST API specification</td>
</tr>
<tr>
<td>JTango</td>
<td>TANGO kernel Java implementation</td>
</tr>
<tr>
<td>tango-doc</td>
<td>The official documentation for Tango controls</td>
</tr>
</tbody>
</table>
The Tango Community has **more than 500 participants** from all over the World!

We are proud to have **more than 45 institutional** and **more than 15 industrial partners**.
3 main events

Write the docs camp
next event ???

St. Petersburg, Russia

Kernel meeting
03 Nov 2020 -> ONLINE!


Autumn 2020 Status webinar
17 – 18 November 2020

https://www.tango-controls.org/community/events/autumn-2020-status-webinar/

POSTPONED

34th Tango Collaboration
meeting, 2020
Saint-Petersburg, Russia from 9
Jun 2020 to 11 Jun 2020
Tango Collaboration is an international effort to finance and sustain the development of Tango in the long-term. The Tango Controls executive body is the Steering Committee. It makes strategic decisions about core developments in the Tango collaboration.

- **INAF** is part of the TANGO Collaboration as contributing member

- INAF main contribution is about **SKA**

- SKA decided to adopt TANGO as control framework for the whole telescope thanks to INAF

[ALBA (core)]
[ESRF (core)]
[SOLEIL (core)]
[DESY (contributing)]
[INAF (contributing)]
[MAX-IV (contributing)]
[SKA-Q (contributing)]
[SKA-ZA (contributing)]
[SOLARIS (contributing)]
[ELI-Beamlines (Contributing)]

https://www.tango-controls.org/about-us/#steering%20committee
TANGO Event organized by INAF

- 5-6 May 2015 - **Trieste Tango@SKA training course**
- 25-27 March 2015 – Trieste - **LMC Standardization Workshop**
- 06 Jun - 08 Jun 2017 – Firenze - **31st TANGO Collaboration Meeting**
- 2018 – **TANGO Training @Torino**
- 2018 – **TANGO Training @Cagliari**
## Main people involved

<table>
<thead>
<tr>
<th>Name</th>
<th>Affiliation</th>
<th>Skills</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matteo Canzari</td>
<td>OAAb</td>
<td>INAF member representatives, framework, tango related web technologies</td>
</tr>
<tr>
<td>Matteo Di Carlo</td>
<td>OAAb</td>
<td>SKA TANGO CoP chair, framework, container, CI/CD</td>
</tr>
<tr>
<td><strong>2 open positions</strong></td>
<td><strong>NEW</strong></td>
<td>Will be involved in SKA and LOFAR2</td>
</tr>
<tr>
<td>Elisabetta Giani</td>
<td>INAF Arcetri</td>
<td>framework, tango c++</td>
</tr>
<tr>
<td>Carlo Baffa</td>
<td>INAF Arcetri</td>
<td>framework, tango c++</td>
</tr>
<tr>
<td>Marina Vela Nunez</td>
<td>OATs</td>
<td>framework</td>
</tr>
<tr>
<td>Simone Riggi</td>
<td>OACt</td>
<td>framework</td>
</tr>
<tr>
<td>Cristina Knapic</td>
<td>OATs</td>
<td>framework</td>
</tr>
<tr>
<td>Riccardo Smareglia</td>
<td>OATs</td>
<td>framework</td>
</tr>
<tr>
<td>Elisa Londero</td>
<td>OATs</td>
<td>framework</td>
</tr>
<tr>
<td>Andrea Bignamini</td>
<td>OATs</td>
<td>framework</td>
</tr>
<tr>
<td>Robert Butora</td>
<td>OATs</td>
<td>framework</td>
</tr>
<tr>
<td>David Tosolini</td>
<td>OATs</td>
<td>framework</td>
</tr>
</tbody>
</table>

### Industrial Partner

<table>
<thead>
<tr>
<th>Description</th>
<th>ELETTRA</th>
</tr>
</thead>
<tbody>
<tr>
<td>MoU with Elettra Synchrotron</td>
<td></td>
</tr>
</tbody>
</table>
### International collaborations

<table>
<thead>
<tr>
<th>Project</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SKA</td>
<td>Developing the SKA control software</td>
</tr>
<tr>
<td>LOFAR2</td>
<td>Monitor and Control Subsystem for LOFAR 2 Station</td>
</tr>
</tbody>
</table>

### Internal projects

<table>
<thead>
<tr>
<th>Project</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRT</td>
<td>Web user interface (proof-of-concept), ACS-TANGO bridge</td>
</tr>
<tr>
<td>NADIR</td>
<td>Data archiving and distribution</td>
</tr>
<tr>
<td>PRISMA</td>
<td>Monitoring and control full-sky camera</td>
</tr>
</tbody>
</table>
Join the community

BE SMART

JOIN THE TANGO COMMUNITY
THANKS!

Contact me: matteo.canzari@inaf.it