TANGO V8 - ANOTHER TURBO CHARGED MAJOR RELEASE

Andrew Götz, Jean-Michel Chaize, Tiago Coutinho, Jens Meyer, Faranguiss Poncet, Emmanuel Taurel, Pascal Verdier,ESRF, Grenoble, France
Stephane Perez, CEA, Arpajon, France
David Fernandez-Carreiras, Sergi Rubio-Manrique, CELLS-ALBA, Barcelona, Spain
Stefano Cleva, Marco Lonza, Lorenzo Pivetta, Claudio Scafuri, Elettra, Trieste, Italy
E Igor Alexandrovich Khokhriakov, HZG, Geesthacht, Germany
Darren Paul Spruce, MAX-lab, Lund, Sweden
Gwenaelle Abeille, Alain Buteau, Nicolas Leclercq, Frédéric Picca, SOLEIL, Paris, France

Abstract

The TANGO collaboration continues to evolve and improve the TANGO kernel. A latest release has made major improvements to the protocol and, the language support in Java. The replacement of the CORBA Notification service with ZMQ for sending events has allowed a much higher performance, a simplification of the architecture and support for multicasting to be achieved. A rewrite of the Java device server binding using the latest features of the Java language has made the code much more compact and modern. Guidelines for writing device servers have been produced so they can be more easily shared. The test suite for testing the TANGO kernel has been re-written and the code coverage drastically improved. TANGO has been ported to new embedded platforms running Linux and mobile platforms running Android and iOS. Packaging for Debian and bindings to commercial tools have been updated and a new one (Panorama) added. The graphical layers have been extended. The latest figures on TANGO performance will be presented. Finally the paper will present the roadmap for the next major release.

WHAT IS TANGO?

Tango [1] is a control system tool kit developed by a community of institutes. It is object oriented with the notion of devices (objects) for each piece of hardware or software to be controlled. Tango classes are merged within operating system processes called Device Servers. Three types of communication between clients and servers are supported (synchronous, asynchronous and event driven).

But Tango is not only the software bus which handles the communication between device servers and clients. The Tango tool chain offers software from the hardware interface to the graphical user interface for several programming languages.

Tango utilities are available, with the basic installation, for code generation, device configuration and testing and for administration and survey of a whole Tango control system.

An archiving and a configuration snapshot system usable with Oracle or MySQL are also available.

Table 1 : Available Tango Modules

Module	Description
Core Libraries	Client/Server communication libraries for C++, Python and Java
Device Classes	More than 300 hardware interface classes are available to download [2]
GUI Frameworks	Available for C++ and Python using QT, for Java using Swing and a web interface written in PHP
Client Bindings	LabView, Matlab, IgorPro and Panorama
Tools	Pogo – Code generator for device classes in C++, Python and Java Jive – Configuration and testing tool Astor – Administration and survey of the Control system
Archiving	Archiving and snapshot system with GUIs and web interface. Usable with Oracle and MySQL
Alarm System	Event driven alarm service
Sardana	Framework for experiment control : Interface standardization, configuration, sequencing, command line interface

Tango development started in 1999 at the ESRF and has made again, with today's release version 8, some major improvements compared to the last versions.

EVENT SYSTEM

The new high performance event system based on ZMQ (version 3.2.3) [3], which was already presented as a development study at the ICALECS 2011 [4], is now available with the release of Tango version 8. It replaces the former event system which was based on an implementation of the CORBA notification service.

The measured performance for event distribution was increased by a factor of 40 for events transferring small amounts of data and by a factor of 10 for events containing big data junks (>100Kbyte).

ISBN 978-3-95450-139-7

	Event rate	Latency
1 double = 64bits	95KHz	300us
1 KByte	82KHz	250us
1 MByte	1Khz	2ms

Figure 1: Event system performance.

The implementation uses the publish-subscribe pattern of the ZMQ library. The default event propagation is still the standard unicast mode. Event multicasting is possible, but needs to be configured for a set of events due to the more complex network configuration. For multicasting the Pragmatic General Multicast (PGM) protocol of the ZMQ library is used.

The new event system is available for servers and clients written in C++, Python and Java. The Java implementation is based on Jzmq (2.1.2) which is a JNI layer above the C++ ZMQ library. It also works with the pure Java implementation jeromq.

A major effort was done to allow easy integration of the new event system into a running control system. Compatibility was pushed that today event propagation is possible using both event systems in parallel. Servers are able to propagate events on both channels. ZMQ is used with Tango 8 clients and the notification service for old clients. Clients linked with Tango 8 are also able to receive on both channels. The switching is handled in the Tango API library and is completely transparent for the device server and the application programmer. Like this, just by re-compiling parts of the control system the event propagation will change from the old to the new system.

JAVA SERVERS

The Tango Java API is developed using the Jacorb framework [5] that is a Java CORBA implementation. Jacorb is stable and continuously maintained (last released in December 2012). The Java Client API is extensively used in the Tango community and up-to-date with the latest Tango features.

On the other hand, the Java server API development was on hold for several years, and lacked some major C++ API features. Meanwhile, the Java language had also released two new versions.

The Java Server API refactoring was performed in 2011 by Soleil.

The first goal was to implement all the "already existing in C++ features" of Tango. Here are the major improvements:

- Attributes State and Status: were only defined as Commands in the old API.
- Write Spectrum and Image attributes: only scalar attributes were writable.
- Dynamic Commands and Attributes: this feature was already possible in the old API but was very complicated and done with an additional API.
- Black box: detailed histories of any client request: this feature was incomplete.

To validate the Java API equivalence with the C++, the C++ tests suite was executed on a Java test server.

The second goal on this upgrade was to take advantages of the latest Java add-ons. A main novelty of Java 5 was the 'annotation' [6] that allows adding metadata to a Java source code. This feature is right now a key driver when it comes to create a Tango device. Here is a sample code that creates a server 'TestDevice' with a read/write attribute called 'myAttribute':

- The '@Device' annotation defines the class 'TestDevice' as a Tango device
- And the '@Attribute' annotation defines the class field 'myAttribute' as a Tango Attribute.

```
@Device
public class TestDevice {

 @Attribute
 public double myAttribute;

 public double getMyAttribute() {
  return myAttribute;
 }

 public void setMyAttribute(double myAttribute) {
  this.myAttribute = myAttribute;
 }
}
```

Previously, a device developer needed to have a strong knowledge of inherited methods and attributes and an overall understanding of the entire API. With these features of the current release, the code is clearly focused and simplified.

The API is one year old now, and is intensively used in production at Soleil. Most of the Java servers used at Soleil respond to a high-performance requirement since there are collecting data on thousands of other Tango devices and are also polled by many client applications.

The Java server API is fully documented [7] and referenced in the tango web site.

MISCELLANEOUS

Pogo

Major work was done to re-structure and improve the templates for the Tango code generator Pogo. The templates for Python servers have been re-structured and the templates for the Java servers completely re-written according to the new Java server API.

The code generation was ported from the Eclipse Xpand to the Xtend [8] package. Much easier modelling in a Java like environment is now possible.

The handling of dynamic attributes was added to the code generation templates.

ISBN 978-3-95450-139-7

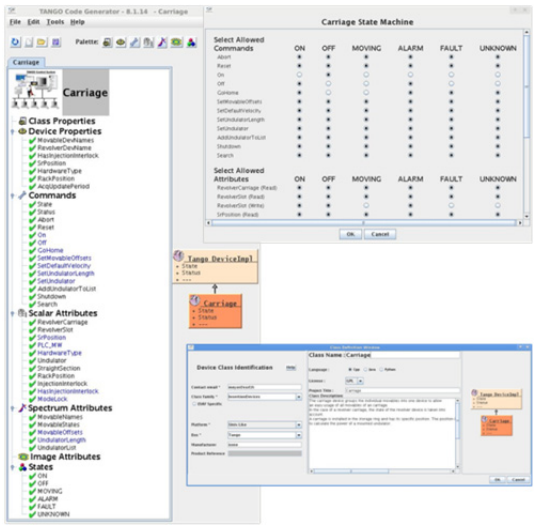

Figure 2: The Tango code generator Pogo.

Database API

The interface to the Tango configuration database was extended to allow applications to work in an environment with multiple Tango control systems. It is now possible to clone or move device server configuration dynamically, from a GUI interface, from one control system to another one. This is extremely useful at a light source where equipment is often moved from one beamline to another one.

Panorama

On top of the LabView, Matlab and Igor Pro bindings, a new Tango binding for the industrial SCADA system Panorama [9] is now available for download.

Tango and Industries

Not only the Tango collaboration is growing but also the interest of industrial companies in Tango is increasing. After the delegation of Tango training courses to one of our industrial partners we are in the process of creating a legal representative for Tango as central contact for industrial partners and new industrial users. See the paper MOPPC078 [10] for a more detailed description of the on-going efforts.

QUALITY AND DOCUMENTATION

To insure the good quality of the growing Tango code base, the test suite was re-furbished using a customized version of the Cxxtest [11] unit testing library. It was also completed to achieve code coverage of 60%. Using Jenkins [12] as continuous integration tool for compilation and testing allows quick detection of introduced problems.

Figure 3: Continuous integration and testing.

On top of good code quality new users must be introduced in a comprehensive way to the Tango control system philosophy. Soleil has written the device server design and implementation guidelines. The document explains clearly how a device server should be designed and programmed. More documents like this are planned to complete the technical documentation.

MOBILE AND EMBEDDED PLATFORMS

Tango on Mobile Devices

Tango offers several ways to interact with mobile devices. The first solution uses the Cordova (PhoneGap) [13] framework and jQuery mobile to allow the writing of java script applications which connect to Tango device servers via proxy servlets on a TomCat server.

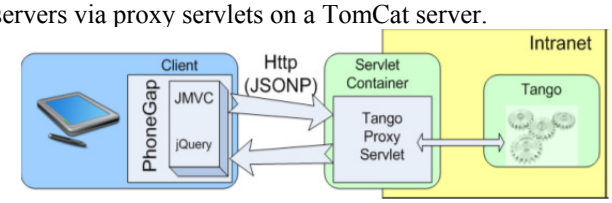

Figure 4: Cordova framework.

The second solution is a browser based web solution, which uses the web interface of the Taurus GUI framework [14] from Tango. It allows writing java script applications which communicate via web sockets with a Tornado web server.

```
<td id='current'   class='value' data-taurus-model="sys/machstat/tango/sig_current
   "> </td>
```

Figure 5: Taurus web integration.

Inside the web server the Taurus web module takes care of translating Tango data into events on the web socket. Dependent on the underlying Tango server Taurus will switch automatically from polling to an event based communication schema to refresh data on the web page.

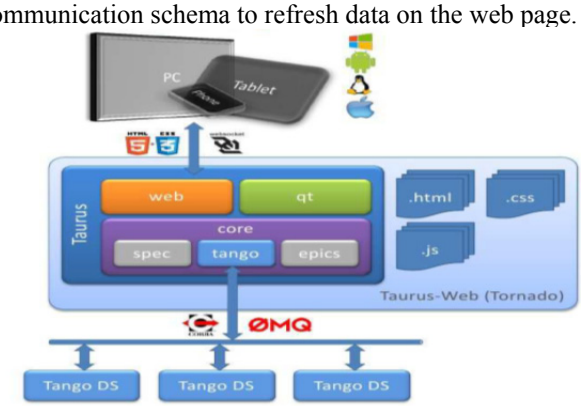

Figure 6: Browser based web solution.

For applications on the local network a third solution is available. The Tango java client API was ported to Android and can be used in applications accessing directly the Tango control system.

ISBN 978-3-95450-139-7

Embedding Tango

TANGO is well suited to be put on embedded systems to add intelligence to devices and to reduce the number of protocols to manage in a distributed system. The latest hardware platforms supported are the Rasberry Pi [15] and the Beagle Bone [16]. Both are low cost fully fledged ARM based computers running Linux.

The Rasberry Pi platform is a very low cost and ideal for learning about Tango and doing demonstrations.

Figure 7: Rasberry Pi.

The Beagle Bone is ideal for embedding Tango in production serious projects. Development projects are on-going at ELETTRA and ESRF.

Figure 8: Beagle Bone development at ELETTRA.

TANGO V9 ROADMAP

Since Tango version 8 is released, work has already started on the features for Tango version 9. The main improvements will be

Pipes

Pipes are a third communication type between clients and servers on top of commands and attributes. Via a data pipe data blobs can be transferred. A data blob is variable set of data composed out of basic data types like a C-structure. The data blob is self-describing and the composition of the data blob might change with every data transfer.

Every data pipe will have a name like a command or an attribute and ways to retrieve the composition of the sent data blobs.

The main usages for data pipes will be the transfer of synchronized sets of data, like for example the result of scan.

Forwarded Attributes

Another key feature for the version 9 will be attribute forwarding. The same attribute, or data value, might be used by several Tango classes. Instead of re-implementing the attribute in a second class we want to instantiate automatically an attribute with the same interface which forwards all its read and write requests as well as configuration changes to the source attribute.

No manual coding is required, because the code generator Pogo can produce the necessary code when designing the interface of a Tango class.

CORBA and ZMQ

Tango uses both protocols today for the data transport on the network. ZMQ is used for event driven communication and CORBA for all synchronous and asynchronous requests. This proves that Tango has encapsulated the network protocol properly and allows its replacement.

For Tango 9 investigations will be carried out how to replace the aging CORBA protocol. ZMQ might be a replacement candidate, but does not bring all the needed features. See the paper TUCOCB07 [17] for more detailed information.

REFERENCES

[1] http://www.tango-controls.org
[2] http://www.tango-controls.org/device-servers
[3] http://www.zeromq.org/
[4] E.Taurel et al., “Tango Collaboration and Kernel Status” ICALEPCS’2011, Grenoble, France, October 2011.
[5] http://www.jacorb.org/
[6] http://en.wikipedia.org/wiki/Java_annotation
[7] http://tangocs.svn.sourceforge.net/viewvc/tangocs/api/java/server/JTangoServer/trunk/doc//download
[8] http://www.eclipse.org/xtend/
[9] http://uk.codra.net/panorama/
[10] Andrew Götz, Jean-Michel Chaize, Alexandre Delorme, “TANGO Steps Toward Industry,” MOPPC078, ICALEPCS 2013, to be published.
[11] http://cxxtest.com/
[12] http://jenkins-ci.org/
[13] http://cordova.apache.org/
[14] http://www.tango-controls.org/static/taurus/v300/doc/html/users/ui/taurusgui.html
[15] http://www.raspberrypi.org/
[16] http://beagleboard.org/Products/BeagleBone
[17] Andrew Götz, Emmanuel Taurel, Pascal Verdier, “Can ØMQ Replace CORBA,”TUCOCB07, ICALEPCS 2013, to be published.

ISBN 978-3-95450-139-7