FUTURE OF TANGO

Andy Götz, Jens Meyer, Emmanuel Taurel, Jean-Michel Chaize, Pascal Verdier, Faranguiss Poncet, ***ESRF*** *(Grenoble, FRANCE), Majid Ounsy, Nicolas Leclercq, Alain Buteau,* ***SOLEIL*** *(Paris, FRANCE), Claudio Scafuri, Marco Lonza,* ***ELETTRA*** *(Trieste, ITALY), David Fernandez-Carreira, Jörg Klora,* ***ALBA*** *(Barcelona, SPAIN), Thorsten Kracht,* ***DESY*** *(Hamburg, GERMANY) on behalf of the Tango Collaboration*

Abstract

Tango is a control system based on the device server concept. It is currently being actively developed by 5 institutes, 3 of which are new institutes. In October 2006 the *Tango* development community met at the *Hotel des Skieurs* in the French Alps to discuss the future of Tango. This paper summarises the conclusions of this meeting. It presents the different areas *Tango* will concentrate on for the next 5 years. Some of the main topics concern services, beamline control, embedded systems on FPGA, 64-bit support, scalability for large systems, faster boot performance, enhanced Python and Java support for servers, more model-driven development, and integrated workbench-like applications. The aim is to keep powering *Tango* so that it remains a modern, powerful control system that satisfies not only the needs of light-source facilities but other communities too.

PHILOSOPHY

The philosophy of Tango has right from the beginning been to build a modern control system which is constantly being improved based on user needs and technology trends. This kind of approach needs resources and constant reflection on how to improve Tango. Agreeing on needs is helped by the fact that the Tango development community consists largely of synchrotron light sources. However, technological trends are more difficult to agree on. Often there are as many opinions as participating institutes. This paper presents the current reflections of the Tango community on how Tango will evolve in the future based on the philosophy of ***constant improvement***.

CURRENT STATUS

Tango is just over 5 years old and is been adopted by 5 institutes in the following domains:

- ESRF – accelerator and beamline control
- SOLEIL - accelerator and beamline control
- ELETTRA – accelerator control
- ALBA - accelerator and beamline control
- PETRA III (DESY) – beamline control

The institutes share the following in Tango:

1. the CORBA protocol
2. the device server model
3. the database
4. management tools
5. navigation + test tools
6. common device servers
7. a tool to generate device servers
8. an archiving database

Institutes do not share:

- device servers for institute specific hardware
- graphical user toolkits for building user interfaces
- domain specific applications for accelerator physics, beamline control, online data analysis

Tango is very flexible and does not impose constraints on the choice of hardware or the operating system (Tango runs on Linux and Windows). Due to this flexibility and the fact that there is a huge variety of hardware to choose from, each institute has made different hardware choices. This means that in practice an institute which adopts Tango invests a significant amount of its resources in writing device servers. This could be avoided by sharing hardware choices between institutes but the advantage of choosing hardware based on institute specific criteria is mostly seen as an advantage.

This is similar for the choice of graphical user interfaces. Tango supports language bindings to the following languages:

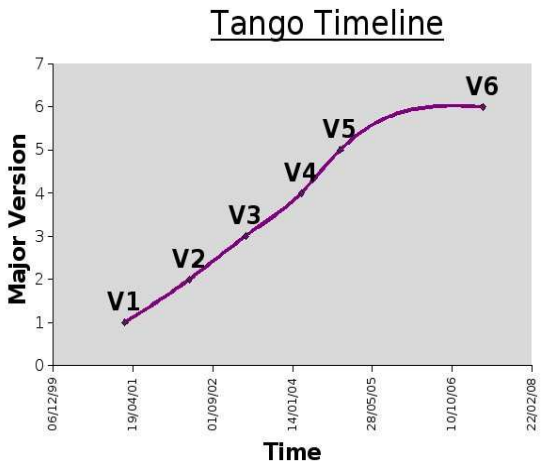
- C++
- Java
- Python
- Matlab
- Labview
- Igor

Each institute has expertise and preference for a subset of these languages and their graphical toolkits. This means that it is difficult to agree on only one single graphical toolkit. In practice this results in each institute creating and maintaining their own graphical user interface.

Domain specific solutions are what the end-user is most interested in. The aim of the Tango community is to share as much as possible in this area.

CONSTANT EVOLUTION

Tango has been constantly improved since its inception. This can be readily seen from the plot of the Tango major version release dates:



TANGO - QUO VADIS ?

The October 2006 meeting on the future of Tango decided that Tango should concentrate on the following areas:

- the development collaboration
- stability, quality and packaging
- scalability and reliability
- new needs-driven features
- more and improved tools
- sharing of domain specific solutions

TANGO FEATURE REQUESTS

In order to identify and track the evolution of Tango the Tango Feature Request system has been proposed. New features to be added to Tango are registered as ***Tango Feature Requests*** (TFR's).

COLLABORATION

Collaboration is the key to the continuous evolution of the Tango control system. In the future we need to find ways to strengthen the collaboration and actively work on common projects. Improvements to the Tango core would be implemented more efficiently if the collaboration financed one or two dedicated system developers. This leads to the first Tango Feature Request for the future:

TFR 1*: each member institute must actively work on a part of the Tango core*

STABILITY, QUALITY, PACKAGING

Stability is an essential part of any software system and even more when it is used to control complicated expensive apparatus like accelerators and experiments 24 hours a day, 7 days a week. The first requirement for the future of Tango was to consolidate the existing version of Tango. Although Tango is stable (a major release is made on average once every 12 months) it was decided to concentrate on bug fixes rather than new features for the next major release. After that emphasis will be put on new features again. This was done between V5 and V6. The next major release will concentrate on new features again.

Software Quality measures how well software is designed, and how well the software conforms to that design. The design of the Tango libraries is maintained by a small group of system developers. They take input from the community and ensure that the design satisfies the needs. The developer team is open to all member institutes of Tango. Tango currently has a test system for the system libraries which tests that the features are correctly implemented. There is currently no test system for device servers, hence the following feature request for the future in Tango:

TFR 2: *automatically generate unit tests for device servers*

Packaging and documentation are considered to be the main problems faced by new users to Tango. Good packaging makes the difference between pain and pleasure in the life of a system integrator. Tango offers source code packaging of the system libraries for Linux platforms and binary code packaging for Windows platforms. There is currently no packaging system for device servers. Therefore in the future the following request will be satisfied:

***TFR 3**: adopt a packaging system for device servers which supports source and binaries*

Documentation is a strong and weak point in Tango. A lot of documentation exists for Tango - the Tango book is almost 500 pages long, but it is difficult to keep up to date. Tango will adopt a new system of documentation which is easier to keep up to date and which is generated automatically as much as possible:

***TFR 4**: reorganise and update the Tango book*

SCALABILITY AND REDUNDANCY

Tango is based on a binary protocol (CORBA) which is fast and has little overhead. The exchange of information between clients and device server is totally distributed - control system clients have a dedicated link to servers. An event system allows efficient asynchronous communication between hundreds of clients and hundreds of servers. However Tango has scaling problems when thousands of servers are started simultaneously. The central database forms a bottleneck due to increased traffic from servers and clients. This bottleneck needs to be solved for Tango installations with thousands of devices e.g. the size of the ESRF, SOLEIL, or even bigger installations like the ILC:

***TFR 5**: distribute the load of the tango naming service to be able to support tens of thousands of servers and clients starting simultaneously*

Reliability is an essential feature in high availability systems. One way of increasing reliability is by means of redundancy. Tango supports redundancy for the central database but not for device servers. Therefore in the future:

***TFR 6**: add redundancy for device servers which will enable multiple copies of the same device to be running with automatic switch over in the case of failure*

NEW FEATURES

As part of the constant evolution philosophy adopted by Tango there is a list of minor and major new features to add in the future. Note that some of the features listed here require the Tango libraries to be refactored to make sure the code remains logical. Here is a list of features which have been requested to be added to Tango in the future:

Tango device servers can be written in C++, Python or Java. The Java servers have fallen behind the other two languages:

***TFR 7**: update the Java server framework*

Multi-channel hardware is very common in control systems. It would be useful to have a multichannel device class to optimise access to multi-channel hardware:

***TFR 8**: implement a multi-channel device class in the Tango device library*

Security is available as a service for Java clients. This needs to be extended to C++ clients:

***TFR 9**: extend security service to C++ clients*

It is often useful to limit the choice of user input. Enumerated data types are the obvious way to go:

***TFR 10**: add enumerated data types to the Tango data types transferred over the network*

Tango data types are raw types in the sense that they represent sequences of simple data types with a minimum of information on what the data mean e.g. voltage, current. It would be useful to be able to transmit so-called “cooked” data types with pre-defined data representations e.g. JPEG, TIFF, Nexus data file, etc.:

***TFR 11**: add cooked data types to Tango*

Performance at device startup time can be improved by implementing a cache of device properties:

***TFR 12**: cache device properties in the database server*

Tango uses a polling thread to generate events. There is currently only one polling thread per server. Performance can be improved in the future by extended the polling thread to one per device:

***TFR 13**: extend polling thread to one per device*

SERVICES

A major trend in distributed computing is the widespread use of Service Oriented Architecture (SOA). SOA encourages decoupling of clients and servers by using generic user interfaces to provide access to services. Tango already has a security service. Tango will be extended to support user services e.g. computation services, via a new services-oriented interface:

***TFR 14**: extend the Tango API to add support for system and user services*

MODELLING

Model driven architecture (MDA) is a useful programming technique for generating source code from models. The Tango tool for generating and writing device servers, Pogo, generates source code from the programmer's model of the device server. Pogo uses its own libraries to parse the source and generate source. In the future we will investigate how open source tools for model driven development (MDD) could improve Pogo. MDD is an essential part of any software system, especially one as diverse and large as Tango.

SYSTEM TOOLS

Tango system tools consist of two applications – Jive for navigating and testing devices and Astor for managing a Tango control system. These tools are used by all institutes and need to be constantly improved to take into account feedback from users. Here is a list of improvements currently requested:

***TFR 15**: display device dependencies*

Individual device servers sometimes need tuning in function of their hardware and the client load. Today metrics are available only for the database server. This should be extended to all devices:

***TFR 16**: provide metrics for all devices, monitoring and plotting tools*

***TFR 17**: ease deployment of large groups of device servers*

STANDARD INTERFACES

The definition of Standard Interfaces for families of devices is essential to achieve hardware decoupling and encourage client application sharing. Tango defines Standard Interfaces with the help of Abstract classes. The current list will be extended in the future to cover a complete catalogue of common hardware:

***TFR 18**: extend the Tango Standard Interfaces catalogue to cover all common hardware*

INTEGRATED WORKBENCH

Today's Tango applications do not share more than a common library. The system tools and end-user applications run independently without sharing windows or information. A new paradigm has appeared recently in industry based on an integrated workbench paradigm. Examples of this are the Eclipse and Netbeans platforms. Tango foresees tying multiple applications together into a single integrated Tango workbench. This will not only make Tango easier for users but also encourage sharing applications between institutes and even between control systems. A similar effort has been started in the EPICS community called Control System Studio (CSS). All efforts will be made to collaborate with the CSS group.

***TFR 19**: develop an integrated workbench for system and end-users of Tango*

BEAMLINES

Currently the main application domain of interest for the Tango community is beamline control of synchrotron experiments. The aim is to share applications and eventually a complete solution for doing beamline control. The first candidates for sharing are

- HKL library for scanning in reciprocal space
- device servers for detectors

- fast scanning techniques
- Python, Spectra, SPEC bindings

Eventually entire frameworks can be shared like Bliss Framework, Device Pool and Passerelle.

TFR 20*: define a common set of beamline applications to share*

PROTOCOLS

Tango was built on top of CORBA. CORBA provides the network layer. The choice of CORBA has been key to the success of Tango. The CORBA IIOP is one of the few high-level binary protocols for which highly efficient free open source implementations exist. Tango uses the omniORB implementation of CORBA. Despite the success of CORBA, CORBA has not evolved significantly over the last few years. The last major extension to CORBA, the CORBA Component Model (CCM) was adopted by the OMG in 2001 but has not been widely implemented. CORBA has failed to offer a solution for integrating web protocols and firewalls. This contrasts with the protocol landscape today where a new set of HTTP based protocols dominate.

We see Tango being extended in the future to support multiple protocols. The Tango Java libraries are already being extended to support the SOLEIL http web protocol. This allows transparent access across firewalls. In the future Tango will investigate adding support for Web Services and multi-cast protocols.

TFR 21*: support the use of multiple protocols in Tango*

Tango can be seen as a wrapper technology for implementing components. Tango has successfully implemented a component model similar to what the OMG set out to do with the CCM but did not achieve.

COMMON TECHNOLOGY

Common technology refers to the technology of common interest to the participating institutes.

Parallel Computing

FPGAs could be used on a routine basis for controls and online data analysis. They could be made available as a Tango service.

Libera

The Instrumentation Technology Libera devices are widely used at most institutes for beam diagnostics. There is a common device server and know-how is shared via a dedicated interest group.

FUTURE TRENDS

Tango is open to future trends in technology. Because of its flexibility it is easy to extend to add system wide support for new technologies. The aim of the Tango community is to integrate new technologies in a coherent manner. Some of the technologies which are currently under consideration are:

GIS

Map devices to a standard Geographical Information System to display their position and relate that to other information in the control system, and to connect this to open source GIS tools like GRASS.

Browser applications

Ajax, Google and others are opening the way to rich clients implemented in browsers.

Ubiquitous Computing

Ubiquitous computing refers to computer everywhere. Refer to the paper "*Ubiquitous Tango*" at this conference to find out how Tango could be deployed everywhere.

CONCLUSION

The Tango control system has adopted the philosophy of "***constant improvement***". This is reflected in the steady stream of new features which have been added to Tango since its creation. The Tango community keeps on finding new areas of common interest for collaboration. We see Tango as a wrapper technology *par excellence* for accessing and controlling hardware and any kind of software. The challenge for the Tango community is to share applications in domains like beamlines, accelerator physics, data analysis.

REFERENCES

[1] TANGO home page: www.tango-controls.org