TANGO CONTROL SYSTEM FRAMEWORK

J. M. Chaize* and A. Götz# on behalf of the TANGO teams of ESRF[1],SOLEIL[2], ELETTRA[3] and ALBA[4]

Abstract

The ESRF and 3 other institutes are building a new control system framework around CORBA named TANGO[1].

TANGO is based on distributed objects and distributed services. It can be scaled to control a large accelerator complex, a beam line or a small embedded system.

TANGO was started at the ESRF in 2000. In 2002 ESRF and SOLEIL in Paris signed an agreement of a collaborative development of TANGO. In 2004 ELETTRA in Trieste then ALBA in Spain decided to use TANGO for the control of their accelerators and beamlines and joined the collaboration.

Thanks to this fruitful collaboration, TANGO has been constantly improved and completed. The source package allows users over the world to download it and use it.

TANGO includes an automatic code generator, a set of services to administrate, archive, display or control equipment. TANGO bindings exists for commercial software packages like Matlab, Labview, Igor , SCADA... Bridges have been written allowing interfacing other systems such as EPICS or OPC.

The 4 institutes involved are setting up a set of abstract patterns to standardize the interfaces of common equipment used in accelerators and beamlines. This abstraction is a step forward in the interoperability of software between institutes.

NEEDS

At a high level the needs of today's users for control systems are as diverse as the applications they are used for. At the lower level however all control systems have to satisfy the same basic needs. These needs are interfacing hardware in a reliable and performing manner, hiding the complexity of hardware, treating errors and alarms, distributing controls over the network, easy high-level access, archiving long term data and user friendliness.

These needs are true for synchrotron beamlines and accelerators too. Over the last ten years the main evolution in these needs has been that the hardware to interface has become capable of dealing with more complicated control tasks e.g. micro diffractometer, entire data taking systems, highly intelligent automatons. This trend will continue in the future and will include automated data taking and analysis too. TANGO has been primarily developed to satisfy these needs. It has a toolkit approach and can be used for solving the low-level interfacing problem as well as the high level part e.g. automating the data analysis.

One of the needs of users which is not often mentioned but which is assumed is that the control system should not be out of date. This is one of the problems facing current control systems e.g. TACO and EPICS.

MOTIVATION

TACO[2], the original ESRF control system, was developed 15 years ago. Although it has been continuously improved over the years, e.g. security, C++, self-describing data types, asynchronism, and events were added, it is still based on the same technology. This technology is C, Motif, ndbm and ONC/RPC as network protocol. These technologies are starting to show their age e.g. ONC/RPC and Motif are frozen, more powerful database than ndbm are now available, and C does not provide support for object oriented programming. New operating systems, languages and protocols have replaced the old ones e.g. Linux, Windows XP, standard C++, Java, Python, CORBA, http and xml to name a few. In order to profit from the developments going on around these new technologies it is necessary to integrate them into TACO. However a limited number of changes can be made without making TACO unstable.

It was decided therefore to develop a new version of TACO, called TANGO, which would be based from the beginning on new technologies. TANGO incorporates all the good points of TACO (e.g. the distributed device concept and the database), improves on its weak points (e.g. device management and development) and adds missing features like automatic startup, easy deployment, attributes, and a true object oriented interface. TANGO runs on the new operating systems and supports new languages like Java and Python. To allow for seamless deployment and migration from TACO to TANGO,

*chaize@esrf.fr
andy.gotz@esrf.fr
[1] http://www.esrf.fr
[2] http://www.synchrotron-soleil.fr
[3] http://www.elettra.trieste.fr
[4] http://www.cells.es

TANGO is able to communicate with TACO and vice-versa.

BASICS

The basic aim of TANGO is to provide a toolkit for building and deploying device servers. It can best be understood by studying this recipe:

- a piece of hardware or software needs to be controlled from another program. The program can be on the same computer but is often running on another computer.
- read the manual for the hardware/software to find out what it can do and make a list of the commands which the hardware/software needs to implement.
- if a device server does not already exist for this hardware/software then write a device class (in C++, Java or Python) which implements these commands as methods. Link everything together into a process called a device server.
- define the device in the database and start the device server.
- write the client which will send requests to the device server and start it.

Device server

The basic concept of TANGO is the device server. The main job of a control system is to implement device control. Device servers implement distributed device control. The distributed device control concept implements all devices as objects in processes called device servers distributed over one or more computers or networks. The network is totally transparent and clients and servers communicate with devices as if they were local. CORBA implements transparent network access at a low level. TANGO adds a high level interface which provides reliable, fast, modeless, and easy access to devices.

The second fundamental concept of TANGO is the database. The database provides a persistent store for device properties e.g. configuration, limits, device server startup information, and device network addresses. The database is implemented as a device server which acts as a high level database interface.

Attributes and Properties

TANGO supports normalized data types called attributes. Attributes can be 0, 1 or 2-dimensional data. Each attribute has a description describing what the data represents, what the limits, units and alarms are. Normalized data types are necessary in order to write generic data display and archiving applications. The framework for attributes is implemented as part of the device root class.

Every device can support a list of properties. The properties are the permanent information related to a device e.g. static or dynamic information which needs to be restored after a startup e.g. position counter for a motor. These properties are stored in the database.

Communication

Communication in TANGO can be synchronous, asynchronous or event-driven. Synchronous communication is the easiest to understand and program. A client sends a request to a server and waits for the server to execute the request and sends the reply. In asynchronous communication the client sends a request to the server and continues immediately without waiting for the reply. The server executes the request and sends the reply. The client unpacks the reply when it has time or can wait for the reply if it needs it before going on. Asynchronous communication is an efficient way to start multiple requests on more than one server simultaneously. Event-driven communication is like asynchronous but the origin of the request is the server in this case and not the client. A client subscribes to the event service and then receives events every time the server detects an event e.g. a state change, trigger or alarm detected.

Graphical User Interfaces

Graphical users interface (GUI) can be written in Java, C++ and QT or Python. An important effort has been made for building Java GUIs

TangoATK is a collection of java-classes to help building applications based on Java/Swing which interact with Tango-devices.

It is developed using the design-pattern Model-View-Controller (MVC)[7] used in Swing. TangoATK provide a set of graphical objects called “viewers” adapted to each type of Tango Attributes. The communication between the non-graphic and graphic objects are done by having the graphic object registering itself as a listener to the non-graphic object, and the non-graphic object emitting events telling the listeners that its state has changed. TangoATK helps minimizing development time, avoid code duplication and provide a common look and feel for all GUIs.

In addition to ATK, TANGO offers client bindings allowing a certain number of commercial products to communicate with TANGO devices. Bindings have been developed for Matlab, Labview, IgorPro, and a Java SCADA named GlobalScreen. For instance, a physicist developing mathematical algorithms can access TANGO devices from a Matlab macro. In Labview, a TANGO device is represented as a “Virtual Instrument”.

Instances and security

TANGO supports the notion of multiple instances. This means there can be multiple copies of TANGO control

systems running simultaneously each one with its own database and devices. Communication between various instances of TANGO is transparent. The multiple instance concept is useful for managing large installations e.g. at the ESRF for the machine and numerous beamlines. It can also be used to avoid a single point of failure in TANGO by running two copies of the same TANGO control system.
Security is implemented at the device level. Client access can be limited to read-only or read-write per request (command) sent to each device. Permissions are stored in the database.

Tools

TANGO provides a full set of Java graphical tools to develop device servers, add them in the database, edit their properties, start and test a device and manage a TANGO control system.
POGO, the code generator, generates the C++ or Java source from the interface specification. It also automatically generates a minimum HTML documentation. This tool allows developing device servers very rapidly. Furthermore, it guarantees a coherence in the style. All device server's source code have the same structure, the same style and have at least a minimum documentation independently of the programmer.

Abstract device pattern

In the world of particle accelerator or beam line control, we often control the same kind of device e.g. beam position monitors, magnet power supplies, signal generators, CCD cameras, counters etc… and we often do the same kind of use of it e.g. scanning, sequencing, data acquisition, feedback, correlation to name a few.

Since TANGO is an object-oriented system, it is well adapted to exploit the Abstract Device pattern[8] and to use inheritance to define a so-called Abstract Control System.

In TANGO, abstract pattern are currently being written for most common control system devices. These patterns specify the common interface that all devices belonging to a certain family have to implement. It defines the list of methods and attributes and the minimum behavior for devices belonging to that family.

Abstract classes are supported by POGO. The code generator proposes a list of abstract interfaces to the device server programmer from which her future device could inherit. In this manner, the programmer benefits from the experience of the programmers who have written the abstract interface. The abstract interface becomes a kind of guideline for the programmer and guarantee that the new device will be usable by the entire TANGO community. TANGO device servers, which implement these interfaces, become easily shared in the wider community.

These abstract patterns allow system integrators to build a large collection of device servers that can be used in different institutes. These institutes can then exchange hardware control and high level software. For instance, if in several institutes, the beam orbit server and the steerers server are compliant with their corresponding abstract interfaces, these institutes can exchange their beam orbit correction process even if they do not use the same kind of hardware.

Applications

TANGO is a toolkit for doing distributed control. It is ideally suited to controlling embedded hardware via the network. It can also be used to do direct control without the network. Examples of the first hardware being controlled by TANGO device servers are serial lines, GPIB, OPC servers, stepper motors, power supplies, digital and analog I/O, CCD cameras, sample environment controllers. The next generation of devices will be controlled via USB or the network directly e.g. using http or xml.

Services

TANGO device server is not limited to controlling only hardware. It can also be used to implement general interest services such as number crunching applications for streamlining data analysis, scanning services, automatic beam line alignment or alarm management.
These general services are directly reusable by all the community independently of the hardware choices. Any contributor can easily extend the list. When associated with abstract devices these services can become the key of the interoperability between institutes.

Communicating with other systems

In beamline or accelerator control systems, when starting a modernization process, we have to integrate the legacy software currently operating. It is also often necessary to integrate macro devices supplied with another control system.
TANGO offers the possibility to integrate easily other control system protocols. This is achieved by offering wrapper servers to maps those protocols.
These classes allow to build TANGO devices from a set of channels coming from a non object oriented protocol. The following protocols have been integrated: EPICS, OPC, TACO, Labview DataSocket, Modbus. In such a manner, TANGO clients can directly access any hardware controlled with those systems.

TECHNOLOGY

TANGO is based on new technologies and is therefore compatible with emerging software standards. The new technologies are :

CORBA

is the main TANGO protocol on the wire. CORBA is a recent standard for distributing objects. It is managed by the OMG[3] consortium. It defines mappings for C++, Java, C, Python and other languages. In addition to the network protocol it defines a number of services of which events and asynchronous messaging are interesting for TANGO. A variety of free and commercial implementations exist. We have chosen OmniORB[4] for C++ and JacORB[5] for Java. CORBA runs on a wide variety of platforms. It is supported by emerging protocols like SOAP[6].

C++

is the main programming language for implementing device servers. It is as efficient as C but is an object oriented programming language. The standard C++ library is used.

Java

is the main graphical programming language. It runs on all platforms and has a rich class library for graphical programming. It can run standalone or be integrated in web applications. TANGO supports writing device clients and servers in Java.

Python

is a popular scripting language. TANGO supports writing clients and servers in Python.

MySQL

is a fast relational database. It will be used to store all device related information. Because it is relational it can evolve to fit the needs of the beamlines and machine as needed. For example it could be used to store archiving data or application data. Its fast access is a major asset. It is possible to run without the database as well. This option is useful for embedded and mobile devices.

THE COLLABORATION

The development of TANGO started in 2000 at ESRF. In 2002, the new French light source project SOLEIL, studied several solution to implement the control of it's accelerator complex and selected TANGO. A collaboration agreement has been signed allowing both institutes to share the development workload and take together the best strategic decisions. In January 2004, the Italian Light source ELETTRA, joined the collaboration and participate actively to the development. A coordinator has been nominated in each institute and regular meeting are organized to take decision and follow-up the action plan[9]. All the code sources are released in the CVS repository of Sourceforge[10]. TANGO has been rapidly packaged in such a way that it can be easily downloaded[11] and installed on one of the supported platforms. Since then, several other users have downloaded and are using TANGO successfully. End of 2004, ALBA the future Spanish source decided to base the control of it's accelerators and beamlines on TANGO and jointed the collaboration. There is lots of advantages of working several institute on the same project. First of all, it improves the quality of the documentation, it imposes careful release and synchronization methods, each piece of software used on several platform by different users is better debugged, and last but not least, the different experiences and point of view enrich the features by taking benefit of the best ideas.

REFERENCES

[1] http://www.esrf.fr/tango
[2] http://www.esrf.fr/taco
[3] http://www.omg.org/
[4] http://omniorb.sourceforge.net/
[5] http://www.jacorb.org
[6] http://www.w3.org/TR/SOAP
[7] Pattern-Oriented Software Architecture: A System of Patterns by Frank Buschmann, and all
[8] Design Patterns: Elements of Reusable Object-Oriented Software, by E.Gamma, R.Helm, R.Johnson, Addison-Wesley, 1995
[9] Tango Collaboration documents http://www-controle.synchrotron-soleil.fr:8001/collaboration/index.htm
[10] Sourceforge project for Tango , http://sourceforge.net/projects/tango-cs
[11] Source packaging for downloading Tango http://www.esrf.fr/tango/tango_src.