EPICS TO TANGO TRANSLATOR

R. Stefanic, Cosylab, Ljubljana, Slovenia
L. Geoffroy, Maatel, France

Abstract

We were faced with a problem of integrating an MD2 diffractometer device into EPICS [1] control system, where the integration into TANGO [2] control system (TANGO device server) already existed. We have developed a generic TANGO to EPICS translator, which provides EPICS control system with an interface to an existing TANGO device server. EPICS Asyn driver [3] is used for handling the communication through a library which is based on the CORBA protocol. The interface provides a generic way for executing commands with different data types as arguments. Attribute manipulation for all major TANGO data types is supported.

Article also includes the description and details of the translators which provide communication between the TINE [4] and TANGO control system in both directions

INTRODUCTION TO EPICS TO TANGO TRANSLATOR

This one-way translator provides an EPICS application with an interface to an existing TANGO device server. In the implementation, Asyn driver is used for managing the communication through the library, which is based on the CORBA protocol, using TANGO API calls. This relieves the Asyn driver of any direct CORBA usage.

The implementation provides a generic way for executing commands with different data types as arguments.

Also provided is the attribute manipulation for all major TANGO data types.

Translator was designed for the MD2 micro diffractometer produced by a company Maatel in collaboration with EMBL at ESRF and had to be robust enough to allow small changes in the TANGO device server implementation.

In principle, the translator may be used for many more applications and the only thing required is the EPICS database and start-up files configuration.

OVERVIEW AND DESIGN OF EPICS TO TANGO TRANSLATOR

The MD2 device server and its control graphic user interface are intended to be run under the Windows OS. The translator was implemented under cygwin [5] environment, so both applications could be executed on the Windows OS based platform.

The communication between EPICS application and MD2 TANGO device server can also be performed over the Ethernet, but that was not the case in our project.

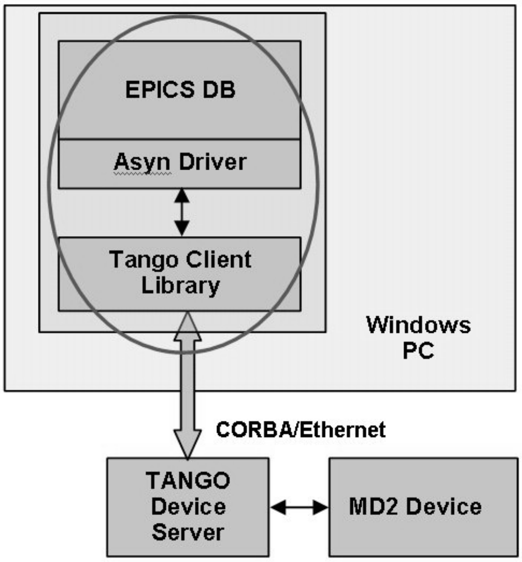

Figure 1: Implementation diagram.

As illustrated in Figure 1, the interface from EPICS application to TANGO device server is provided through a client library, which handles all CORBA communications using TANGO API.

Since the TANGO device server for the MD2 micro diffractometer is running under Windows OS, the library used had been implemented as a dll. The dll can be used as normal library under cygwin environment.

All EPICS records [6] that are intended to communicate with TANGO device server's commands and attributes require specially defined naming convention to be used in the INP or OUT record fields besides regular Asyn port declarations.

- First part defines whether EPICS record will be operating on TANGO command or attribute and is noted by short term "cmd" or "attr".
- Second part defines the name of the TANGO command or attribute on which EPICS record will be operating. The name must be the same as names in TANGO device server.
- Third part can be used as a definition of the data type used for command calls with no or one value as an argument, or writing or reading a single value from and to an attribute. If the attribute or command on which EPICS record consists out of more than one element, this part has extended meaning in order to define the array of elements.

Example of an AI type of records INP field to access the “phiposition” attribute at the TANGO device server:

```
"@asyn($(PORT),$(ADDR))attr_phiposition_dbl"
```

Example of MBBI type of records INP field to access the “State” command at the TANGO device server:

```
"@asyn($(PORT),$(ADDR))cmd_State_state"
```

Translator Start-up and Initialization

Translator must be provided with a valid IP address and a port number of the workstation where the TANGO device server is running in the st.cmd file where Asyn port is also defined.

On application start-up the desired Tango device server is pinged and if the reply is legal the connection is established. Each record in the EPICS record database is specially processed at registration time, where it’s INP or OUT fields are parsed for arguments – with previously described naming convention, from where the desired attribute or command name and data type information are retrieved. A list of all records is created, holding additional information about the desired TANGO command or attribute that is going to be accessed by the individual EPICS record.

In case that the given name in the records INP or OUT field is invalid and refers to an attribute or command that does not exist, an error will be reported and read/write operations will be discarded.

Translator Operation

Every request from EPICS record to TANGO attribute or command is handled through appropriate functions in Asyn driver. Each record is associated with an index, which is used to retrieve additional information about the record from the list created at initialization, which is used for the actual call using the TANGO client library.

Since TANGO is object oriented in comparison to EPICS, the translator in general provides communication between two very different designs through conversion of data, from objects, to EPICS PV’s.

All functionality is implemented in the Asyn port driver and uses standard Asyn driver interfaces.

Problems and Issues

During the implementation couples of issues were raised.

Main problem was presented by attempt to implement compound TANGO data types into EPICS. Besides ordinary data types, TANGO provides the device server with the option to have a command which takes or returns an argument in form of two dimensional array consisting out of pairs of string and long or double values. Solution that would represent this in EPICS was not found and implementation lacks this option and was noted.

Other but minor problem is the EPICS string length limitation. Since Tango does not limit the string length in any way, this was unpleasant problem in case of controlling the MD2 device where long strings are used to report actual device status.

Asyn interfaces used during the implementation are: asynOctet, asynFloat64, asynUInt32Digital, asynInt32.

INTRODUCTION TO TINE TO TANGO, AND TANGO TO TINE TRANSLATORS

Translators provide the ability for TINE control system clients to operate on existing TANGO device servers and the ability for TANGO control system clients to operate on existing TINE device servers.

By using the translators, TINE clients are able to operate on TANGO attributes and commands as TINE properties, and TANGO clients to operate on TINE properties as TANGO attributes.

Translators were designed and implemented as two separate applications which provide exporting of an existing TANGO device server as TINE device server, and exporting of an existing TINE device server as TANGO device server.

DESCRIPTION AND DESIGN OF TANGO TO TINE TRANSLATOR

Translator is used for exporting an existing TANGO device server as a TINE device server.

All TANGO commands and attributes are exported as TINE properties preserving their original names and if possible their data type. Since TINE control system by default supplies a client and device servers with an enormous number of possible data types of the properties, all data types of TANGO commands and attributes are supported under exported TINE device server.

For translators operation, the user must provide only the name of the existing TANGO device server, by which it is exported as a TINE device server.

Translator was implemented as a standalone application for Linux and Windows operating systems.

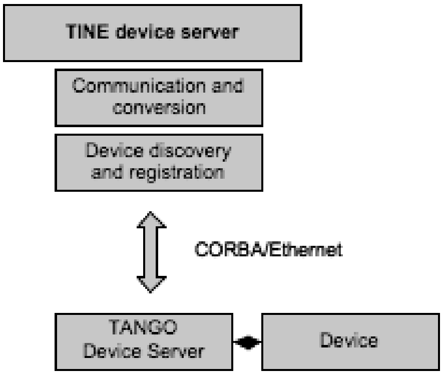
TINE device server
Communication and conversion
Device discovery and registration
CORBA/Ethernet
TANGO Device Server
Device

Figure 2: Implementation diagram.

DESCRIPTION AND DESIGN OF TINE TO TANGO TRANSLATOR

Translator is used for exporting an existing TINE device server as a TANGO device server.

All TINE properties are exported as TANGO attributes with preserved names and if possible their data type.

Since TANGO control system by default only supports a limited number of data types to be used when operating

For translators operation, the user must provide only the “Context”, “Sub” and “Device name” of the existing TINE device server in the TANGO database, which holds all the necessary information for TANGO device server to connect to.

Attributes as equivalent for TINE properties are created dynamically from a list of properties of the existing TINE device server.

Translator was implemented as a standalone application for Linux and Windows operating systems.

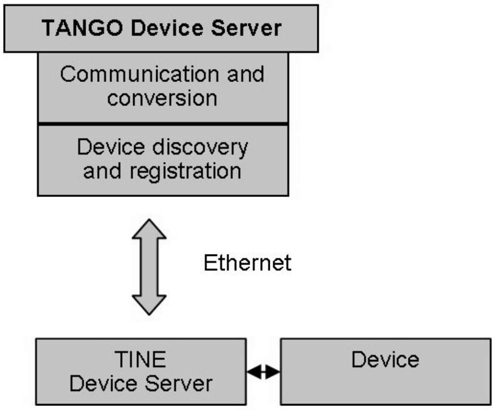
TANGO Device Server
Communication and conversion
Device discovery and registration
Ethernet
TINE Device Server
Device

Figure 3: Implementation diagram.

with commands and attributes, a selection is performed at the TANGO device creation time, registering only the attributes of the appropriate data types. Properties of more complex data types i.e. custom structures, xml files or unusual sets of combined arrays are ignored when exporting them as attributes.

The application is implemented to run as a normal TANGO device server but without previously defined attributes.

ACKNOWLEDGEMENTS

I would like to thank the TANGO development team at ESRF for all constructive ideas and guidance when developing the EPICS to TANGO translator as well Laurent Geoffroy for his help during the implementation.

I have to thank Phillip Duval from DESY institute, for his commitment, time, and suggestions during the implementations of the TINE to TANGO and TANGO to TINE translators, especially on the TINE side.

Special thanks go to Maatel and EMBL for enabling this work by funding it.

REFERENCES

[1] EPICS Control System, http://www.aps.anl.gov/epics/
[2] TANGO Control System, http://www.tango-controls.org/
[3] EPICS Asyn Driver, http://www.aps.anl.gov/epics/modules/soft/asyn/
[4] TINE Control System, http://tine.desy.de
[5] Cygwin Linux-like environment for Windows, http://www.cygwin.com/
[6] EPICS record reference, http://aps.anl.gov/epics/EpicsDocumentation/AppDevManuals/RecordRef/Recordref-1.html