A UNIFIED MATLAB API FOR TINE AND DOOCS CONTROL SYSTEMS AT DESY

J. Wilgen, S. Meykopff, DESY, Hamburg, Germany

Abstract

At the European XFEL, MATLAB will play an important role as a programming language for high level controls. We present xcomm, a standard MATLAB API which provides a unified interface for TINE and DOOCS control systems. It supports a wide variety of data types as well as synchronous and asynchronous communication modes.

DOOCS AND TINE

The two main control systems at DESY, DOOCS [1] and TINE [2], have been in operation for decades at different accelerators at DESY. The TINE control system, originally created for the HERA collider, is in operation at PETRA, DESY2/LINAC2, and REGAE [3]. The DOOCS control system operates the FLASH accelerator [4]. For the European XFEL, ongoing efforts are being made to integrate both control systems [5].

MATLAB IN CONTROLS

During the past years, MATLAB has become increasingly popular for application development in DESY Control Systems. MATLAB programs are in operation at FLASH, PETRA, and REGAE. At the European XFEL, MATLAB will be used as a major programming language for high level control applications.

Several MATLAB APIs for DOOCS and TINE already exist at DESY, with different scopes and interfaces. Each of them supports only subsets of the data types and the features of the control systems. Some are platform dependent. For the high level controls at European XFEL and FLASH, it was required to have a common, complete and well-tested API with an intuitive and robust interface. Therefore we decided to build a new, unified MATLAB API.

XCOMM FEATURES

Communication Protocol

Xcomm uses the TINE protocol to communicate with both TINE and DOOCS servers. DOOCS servers have already integrated TINE by means of a built-in adaptor [6]. When the adaptor is enabled, a DOOCS server automatically becomes visible in the TINE name space and can be accessed by any TINE client as shown in Fig. 1. Relying on TINE, xcomm can therefore serve as a common MATLAB API for accelerator control systems at DESY. An exception is the Karabo control system [7] of the European XFEL undulator beamlines, which is currently not supported, but is planned to be accessed through a TINE gateway, as far as needed.

Simple Interface

Xcomm is implemented as a MEX (MATLAB Executable) function in the C programming language. It is called as a function which can be controlled by a variable list of parameters, as common in MATLAB. Using a single function interface, programs can send data to or receive data from any control system address. The complexity of the underlying client API is hidden from the user.

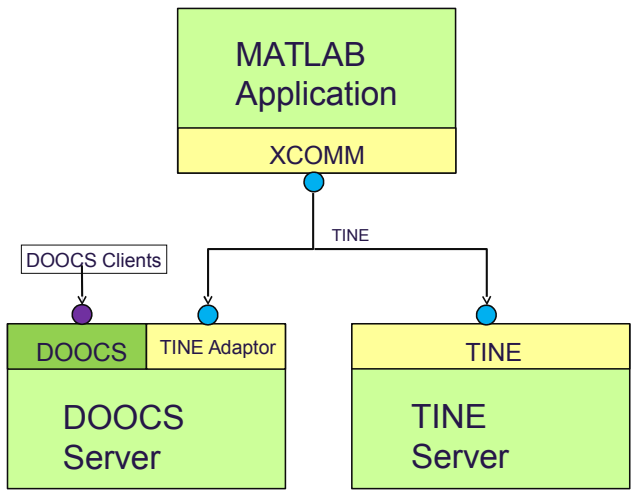

Figure 1: Communication with different servers.

```
result = xcomm('<target address>'
         [,<outputData>]
         [,'optionName',optionValue,...])
```

Both the DOOCS and TINE address styles are supported. The programmer does not have to know if the connected server is a DOOCS or TINE server.

The result of an xcomm call is a structure array which includes an error string, a time stamp in both MATLAB and text formats, and the received data. Since the actual content of the data depends on the request, the data may contain a scalar value, an array, a structure array or a cell array of any of these, dependent on the data type and the number of data units returned. Data sent to the server must be formatted in the same way.

Multiple Data Types

TINE and DOOCS have many pre-defined structure types which xcomm supports. A structure type is mapped to a corresponding structure array in MATLAB.

TINE servers can also have properties with user-defined data types. Xcomm can handle these data types

ISBN 978-3-95450-146-5

and maps them in the same way as pre-defined structure types.

Special types like images or multi-dimensional arrays have specific raw formats which need to be converted for further processing. An image, for instance, should be represented as a matrix in MATLAB. Xcomm converts such data automatically so that the results can be used directly. Still the raw data remain available.

Asynchronous Communication

In addition to synchronous communication, TINE offers different communication schemes [2], and makes extensive use of asynchronous communication in order to reduce the network load and the CPU load of servers. On the client side, this means that asynchronous events need to be processed when data arrive. Since MATLAB programs are single-threaded and typically have a synchronous data flow, xcomm uses a feature of the TINE client library to handle incoming data in a background thread. An xcomm call which uses an asynchronous connection reads data from a local buffer which is updated automatically. As shown in Fig. 2, this makes synchronous and asynchronous communication simple and transparent to the programmer, since it does not require different programming models. Still, callbacks may be added in a future version as an extension for special cases where either latency is important or to reduce unnecessary local polling.

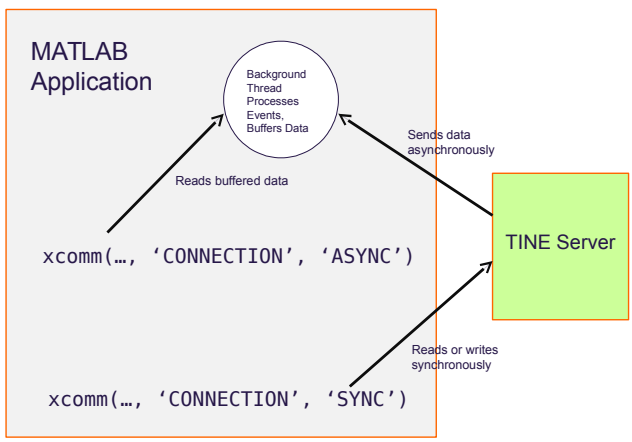

Figure 2: Synchronous and Asynchronous Communication.

Bulk Operations

Xcomm also allows to bundle multiple requests into one call, returning a set of results.

```
result = xcomm({'<address1>', 'address2'},
         [,'optionName',optionValue,...]);
```

where result is a struct array of partial results. Bulk operations can be useful in cases when multiple properties are needed at once.

CONCLUSION

Xcomm is a new MATLAB API to access TINE and DOOCS control systems. First applications have already been developed for FLASH and the European XFEL. We expect that a common standard API will improve the quality and maintainability of programs and make more control system features available to MATLAB programmers. High level control applications developed for the European XFEL will use xcomm as the standard interface.

REFERENCES

[1] DOOCS. http://doocs.desy.de
[2] TINE. http://tine.desy.de
[3] R. Bacher, P. Bartkiewicz et al, "TINE: The Control System for PETRA3 and the DESY Preaccelerators", ICFA Beam Dynamics Newsletter, No.47, 2008.
[4] A. Aghababyan et al., "The Accelerator Control Systems at DESY", ICFA Beam Dynamics Newsletter No. 47, 2008.
[5] P. Duval, A. Aghababyan, O.Hensler, K.Rehlich, "Control System Interoperability, an Extreme Case: Merging DOOCS and TINE, PCaPAC, 2012
[6] K. Rehlich, Control System Interfaces, XFEL Collaboration Meeting, 2013
[7] B. C. Heisen et al., "Karabo: an integrated software framework combining control, data management, and scientific computing tasks, ICALEPCS, 2013

ISBN 978-3-95450-146-5