TangoJS – a web-based interface for TANGO Control System

Michał Liszcz[1], Włodzimierz Funika[1,2], Łukasz Żytniak[3]

[1] AGH, Faculty of Computer Science, Electronics and Telecommunication, Dept. of Computer Science, al. Mickiewicza 30, 30-059, Kraków, Poland
[2] AGH, ACC Cyfronet AGH, ul. Nawojki 11, 30-950, Kraków, Poland
[3] National Synchrotron Radiation Centre Solaris, ul. Czerwone Maki 98, 30-392, Kraków, Poland

liszcz@student.agh.edu.pl, funika@agh.edu.pl, lukasz.zytniak@uj.edu.pl

Keywords: synchrotron, control system, TANGO, CORBA, Javascript, GUI, web components

1. Introduction

Control of the expensive and sensitive hardware components in large installations like scientific facilities may be a challenging task. In order to conduct an experiment, multiple elements like motors, ion pumps, valves and power-supplies have to be orchestrated. To address this problem, the TANGO Control System [1] has been developed at ESRF synchrotron-radiation facility. This paper presents *TangoJS* – a modular, standard-based library for building TANGO clients for web browsers.

TANGO is a distributed, CORBA-based system, where each piece of hardware is controlled by a *device server*. Device servers are registered in a *database*. Client applications allow operators to monitor and modify hardware parameters during an experiment. These applications are often implemented in a form of graphical *synoptic panels* [2] like Taurus.

Recently, web-based approach has become crucial in building accessible and adaptive GUI applications. Unfortunately, TANGO-based applications cannot run in web browsers. Here *TangoJS* comes into play, allowing for rapid development of TANGO clients, integrating neatly with modern frameworks and using standard frontend development tools.

Most attempts to move TANGO clients to the web have failed at a proof-of-concept stage (e.g., Taurus Web or GoTan). The one which is actively developed is mTango [3]. It consists of a frontend client API, a widget toolkit, both implemented using JMVC framework, and a RESTful web-service which communicates directly with device servers. Unfortunately, mTango depends on Java and Rhino, uses JSONP and requires some effort to setup a project. TangoJS is aimed to address these issues as a modular, extensible and lightweight solution that comes with almost no dependencies and uses standardized web and frontend features.

2. Description of a problem solution

TangoJS consists of three components: the *core API*, a *connector* and a *widget collection*.

The core API is a set of interfaces, structures and constants, partly generated from the TANGO IDL which defines all TANGO entities. The goal was to provide the interfaces that all TANGO developers are familiar with. The TangoJS API supports the most common operations, like accessing device's attributes and properties, invoking commands and browsing devices in the database. At the current stage events are not supported. The core API does not perform any communication – all calls are passed to the underlying connector.

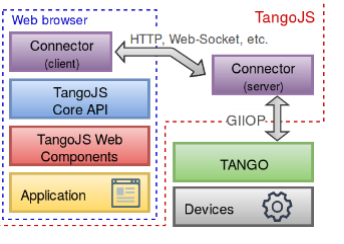

Fig. 1. TangoJS high-level architecture.

The connector is an entity responsible for communication with the backend. As mentioned above, it is not possible to use CORBA (and TANGO) directly in a web browser. The connector solves this problem in a flexible way. At the moment of writing this paper, the TANGO REST API is being standardized by the community [4]. One possible connector implementation might be a client consuming this API. In that case, mTango may be used on the server-side. The server-side is however not in the scope of this paper. TangoJS ships with an in-memory connector which mocks the database with a few devices.

The topmost part of the TangoJS stack is a widget collection, inspired by the existing frameworks (mainly Taurus). Some examples of widgets are shown in Fig. 2. The widget toolkit is framework-agnostic and utilizes modern web technologies like web-components API. No third-party dependencies are required (except plotting).

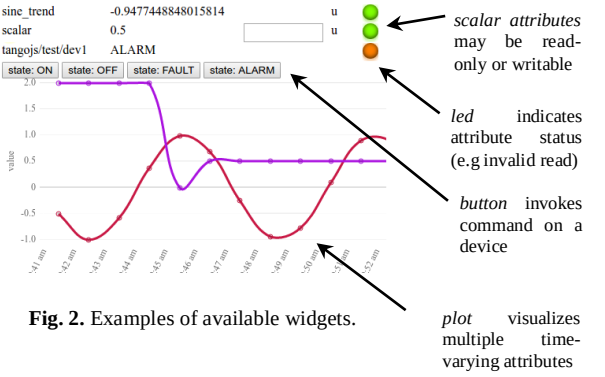

Fig. 2. Examples of available widgets.

3. Results

The proposed solution has several advantages over competitors. The core API is based on standard TANGO IDL. Due to the connector concept, the whole stack is not tied to any particular backend implementation. The widget toolkit allows integration with any frontend framework (e.g., AngularJS) or even with plain Javascript applications.

All the components can be included into existing apps using tools like npm or Bower.

4. Conclusions and future work

The first prototype of TangoJS has been implemented. The development of a proof-of-concept synoptic panel application and a RESTful connector is currently in progress. The next step will be a test deployment on a real hardware platform at National Synchrotron Radiation Centre "Solaris" in Krakow.

Future research aims to improve the security and simplify the integration with the TANGO infrastructure. One solution might be to drop the backend completely in favour of direct communication between a browser and device servers (by using HTIOP).

Acknowledgements. This research is partly supported by AGH grant no. 11.11.230.124.

References

1. J-M. Chaize, A. Götz, W-D. Klotz, J. Meyer, M. Perez and E. Taurel: "TANGO - an object oriented control system based on CORBA", in Proc. International Conference on Accelerators & Large Experimental Physics Control Systems, ICALEPCS, 1999,
2. L.S. Nadolski, J. Chinkumo, K. Ho, N. Leclercq, M. Ounsy and S. Petit: "High level control applications for SOLEIL Commissioning and Operation" in Proc. Particle Accelerator Conference, 2005, PAC 2005 pp.481-483,
3. mTango project website: `https://bitbucket.org/hzgwpn/mtango/wiki/Home`
4. Tango Feature Request: Defining a standard Tango REST API: `http://www.tango-controls.org/community/forums/post/251/`