ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA173

A WEB-BASED REPORT TOOL FOR TANGO CONTROL SYSTEMS VIA WEBSOCKETS

M. Broseta†, D. Roldan, A. Burgos, G. Cuní, D. Fernandez-Carreiras, S. Rubio-Manrique
ALBA-CELLS Synchrotron, Barcelona, Spain

Abstract

Beamlines at Synchrotron Light sources operate 24 hours/day requiring Beamline scientists to have tools to monitor the current state of the Beamline without interfering with the measurements being carried out.

The previous web report system developed at ALBA was based on cron tasks querying the Tango Control system and generating html files. The new system integrates all those automatic tasks in a Tornado Tango Device letting the users create their own reports without requiring the intervention of the software support groups.

This device runs a Tornado [1] web server providing an Html5 [2] web interface to create, customize and visualize its reports in real time (via WebSockets [3]). Originally designed for the vacuum engineers to monitor the vacuum, is actually used by the scientists and engineers involved in the experiment and the different on-call services to remotely check the beamline overall status.

REPORTS TO BE UPDATED

A report is an informational task made with the specific intention of relaying information or recounting certain events in a widely presentable and scrutinized form. Scientists and the different sections involved in a Beamline operation need the control system to provide the necessary tools to generate accurate reports with the status of the Beamlines. Moreover, these reports need to be generated constantly without interfering with the experiments that are being carried out. From the different items available to control and monitor a Beamline, the scientist or the report receiver is who decides which ones provide relevant information to be included in the report. There are already tools to create reports in file format or emails, but for those signals that need to be continuously monitored they are not efficient.

Within ALBA computing section we have improved the way to create automatic reports using a new web-based report tool. It is integrated in the Tango system through a new device server, the Tornado DS, which acts as a gateway between all the devices in the database and the report tool. The control system engineers have just to install and run the device server and then the scientist or other report client can easily create their own html reports. Reports are refreshed automatically and periodically. They can contain strings, numbers, curves or trends, which are automatically displayed according to the data type selected to be displayed.

HOW IT WORKS

The Tango Device Server inherits from Dynamic DS, so it contains all its properties, attributes and functions from its parent class, like the dynamic creation of attributes. This feature allows the creation of new attributes dynamically with formulas that may or may not depend on the values of other attributes available in other device servers within the same database. These dynamic attributes can be included manually using Tango tools or easily added from the HTML form provided with the Tornado DS server.

When the Tornado DS starts, an independent thread is responsible to periodically collect not only the values for the dynamic attributes, but also their label, quality or unit. The collected data is sent to the available clients in JSON [4] format to refresh its contents on the HTML page. The result is the default page provided with the Tornado DS shown in Figure 1.

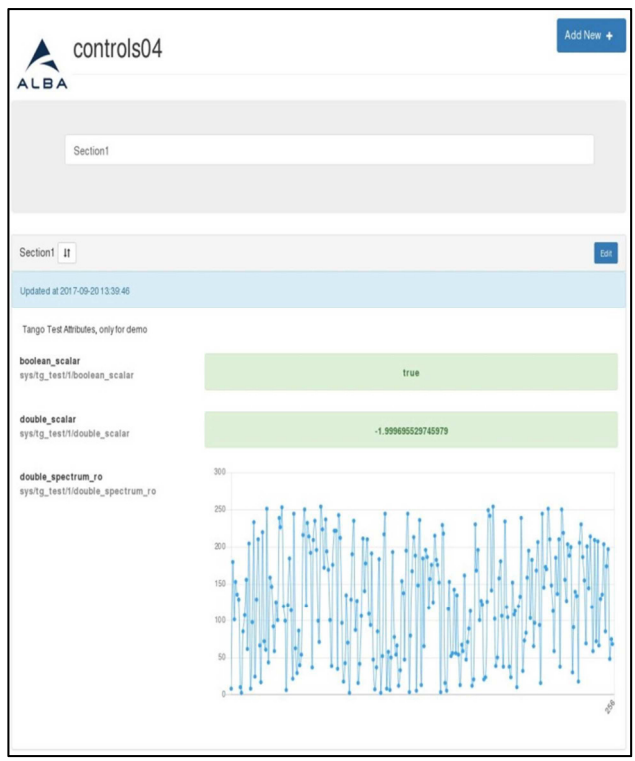

Figure 1: Default Tornado DS index page.

The default web page provided is divided by sections. All defined sections are available in the same page, and if needed, the can be hidden individually. Each section has a name, description and a set of attributes that are periodically refreshed. These attributes are shown with its label, unit and value, but also with a different background colour depending on the attribute quality. New sections can be added, removed or edited easily from the main page as shown in Figure 2: Edit, add or

† mbroseta@cells.es

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA173

remove section menu. Scientist or web users can configure and create their own reports without the need of a control system engineer.

Figure 2: Edit, add or remove section menu.

More complex reports can be created within the Tornado DS just editing the HTML page or creating new ones, with more functionality or with a different style sheet. But, in those cases, help form control system engineers is required to integrate the new designs into the server.

THE TANGO DEVICE SERVER IN DETAIL

The device server starts a Tornado webserver, hence its name. This is a Python web framework and asynchronous networking library that using non-blocking network I/O, can scale to tens of thousands of open connections, making it ideal for long polling, It is based in WebSockets protocol, and other applications that require a long-lived connection to each user. WebSocket enables bidirectional, message-oriented streaming of text and binary data between client and server. WebSocket is a computer communications protocol, which provides full-duplex communication channels over a single TCP connection. The simple and minimal API enables us to layer and deliver arbitrary application protocols between client and server in a streaming fashion, where either side can send data at any time. WebSocket protocol is currently supported in most major browsers including Google Chrome, Microsoft Edge, Internet Explorer, Firefox, Safari and Opera. WebSocket also requires web applications on the server to support it

The Tornado DS requires to have installed in the server machine at least Python 2.7 or newer and the Tornado library. This library is listed in PyPi [5] and although its best performance is for Linux, it also runs on windows, although this configuration is only recommended for development use

The Tornado DS, configures the Tornado webserver before it is started as shown in Table 1. The configuration requires defining the *templates* path, folder that contains the html files, and also the *static* path, that contains the JavaScript libraries or images. Apart from these two paths, it is important to define the handlers for each html page that will control the possible actions to perform over that html page. So basically, the source to start the tornado webserver is resumed as follows:

Table 1: Tornado class

```
from tornado.web import Application
 import tornado.ioloop

class Tornado(object):
    def __init__(self, parent=None, port=8888, extra_cb=None):
        ....
        # Configure tornado Application
        template_path = "WebTornadoDS/templates"
        static_path = "WebTornadoDS/static"
        Json_static = "JSONfiles/"
        handlers = [(r"/", MainHandler),
                    (r"/index.html", MainHandler),
                    (r"/tangoDS/*",TangoDSSocketHandler,  {"parent": self}),
                    (r'/JSONfiles/(.*)', tornado.web.StaticFileHandler,
                                                    {'path': Json_static})
                ].
        self.application = Application(handlers, static_path=static_path,
                                       template_path=template_path, debug=False)
        self.application.listen(self._webport)

def start(self):
       # Start Tornado Server
       try:
          tornado.ioloop.IOLoop.current().start()
          ioloop = tornado.ioloop.IOLoop.instance()
          ioloop.add_callback(ioloop.stop)
        except KeyboardInterrupt:
           print "^C received, shutting down the web server"

    def stop(self):
        # Stop tornado server
        ioloop = tornado.ioloop.IOLoop.instance()
        ioloop.add_callback(ioloop.stop)
```

A default index.html page generated using Bootstrap [6] is provided with the Tornado DS. Bootstrap is just an open source toolkit to develop web pages, including HTML, CSS and JS files. The client access to the host where the Tornado DS is running, through the specified ***Port*** number defined as a Tango property. As mentioned before, Tornado stablishes an independent communication channel for each connected client, so once the page is ready, the protocol is changed automatically in background to web socket instead of http. The JavaScript libraries included in the web page provide the template forms to create automatic reports; adding, editing or deleting sections and attributes to those sections, with a predefined format. Different styles and properties are displayed for the attributes depending on their quality or type. Spectrum or array attributes are plotted with graphs, using *chartjs* [7] open source JavaScript library, instead of showing the list of all its values.. In that sense, as soon as the page is loaded the protocol changes.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA173

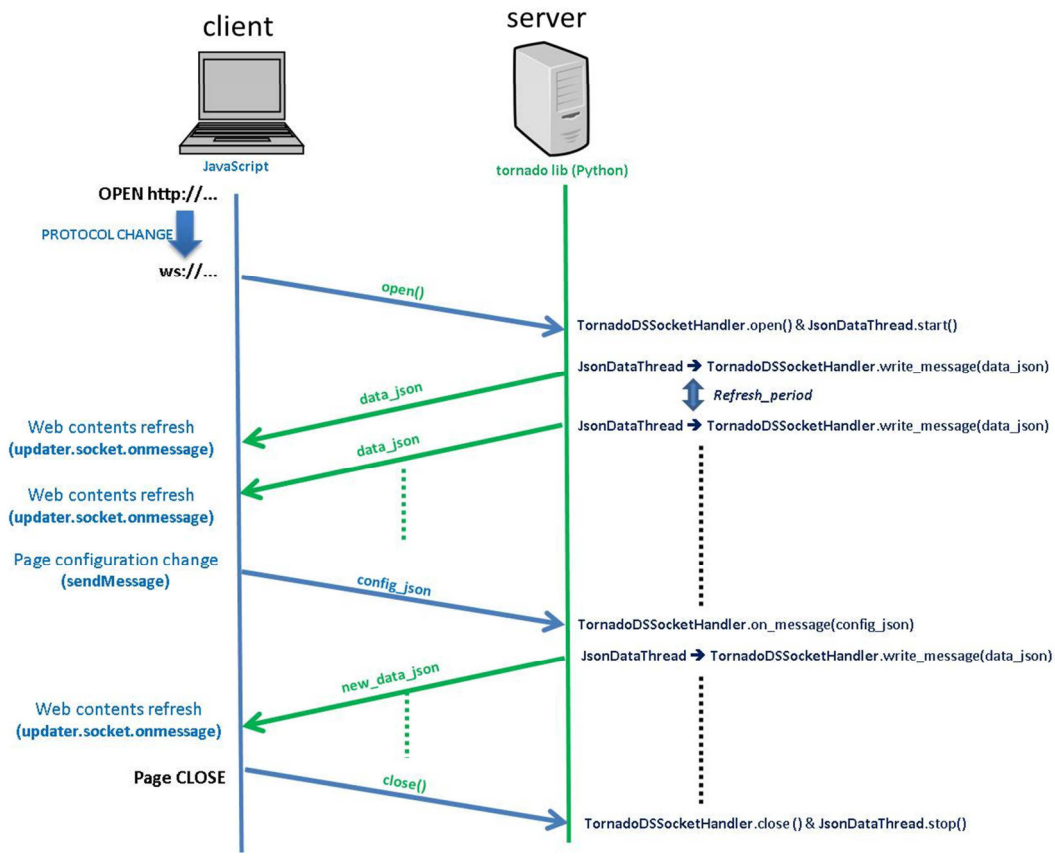

Figure 3: Communication flow between client and server for Tornado DS.

In the server, values of the attributes displayed in the webpage are collected periodically through an independent thread. The refresh period can be modified through the DS property ***Refresh Period*** that is set by default to 3 seconds. The data collected is converted to JSON format before being sent to all the available tornado clients. The server sends a message to the clients with the data, and they execute the corresponding call-back function defined in the JavaScript modules to refresh the values in the web page. For those cases where the web report is not enough and scientist wants to integrate the data into other applications or reports, it is also possible to store the JSON data to a file. In that case, the ***AUTOGENERATEJSON*** property should be set to true, and then the JSON file generated will be overwritten periodically with the new data.

The web client also sends commands to server, every time the user changes the configuration. Adding, editing or removing sections change the configuration of the page. This new configurations is sent to the server to continue generating the JSON data file but with the new configuration. Figure 3 shows the complete communication flow between a client and the Tornado DS

CURRENT STATUS & FUTURE PLANS

First version of the device server is ready to be installed in ALBA Beamlines. The first test reports have been already created for the ALBA accelerator vacuum section with successful results. The device has been already presented to scientist and soon will be installed in the Beamlines. Other experiments within the ALBA Computing section are also using the web reports to present their results [8].

The first version of the Tornado DS is limited to one HTML page including different sections. *Multipage support* will allow the creation of multiple reports for different users and with different formats, all together controlled by the same Tornado DS. Linked to this feature, it has been considered also the possibility to have independent refresh times for each page and section. Actually the thread responsible to collect the data is getting the all defined attribute values periodically. But the frequency of change could be different from one group of attributes to another. Independent refresh time per section will improve the performance of the server in case of having multiple pages or attributes to display, reducing the work load of the thread responsible to collect the attributes values and reducing the data size to send to by the communication channel between server and client.

Custom HTML pages per reports can also be prepared at user request. A good example of this is the current Machine status web report of the ALBA Synchrotron. Different cron tasks and device servers are actually being used to display the overall status of the Synchrotron. Using the Tornado DS this could be

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA173

simplified a lot, just adding the attributes to display as dynamic attributes and customizing the style of the HTML page, as shown in Figure 4.

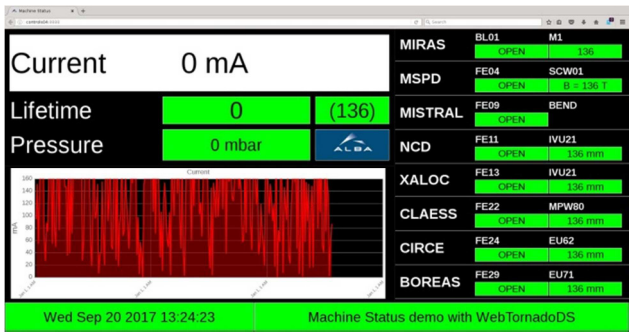

Figure 4: ALBA machine status using the Tornado DS.

CONCLUSIONS

This new web-base report tool provides a flexible way to create dynamic reports that are already integrated in the control system autonomously. Users can do their own reports selecting the information they want to monitor easily without having to implement complex mechanisms which may interfere with the measurements being carried out.

Custom reports can also be prepared without having to change the server code and with just modifications in the HTML web pages. That opens a window to the development of new web-based user interfaces integrated in the control system.

REFERENCES

[1] Tornado Documentation
http://www.tornadoweb.org
[2] HTML5 developer guide
https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5develop
[3] Web-sockets protocol
https://www.websocket.org/aboutwebsocket.html
[4] JavaScript Object Notation (JSON):
http://www.json.org/
[5] PyPI - the Python Package Index
https://pypi.python.org/pypi
[6] Bootstrap open source
http://getbootstrap.com/
[7] Chart.js: Open source HTML5 Charts
http://www.chartjs.org/
[8] A. Rubio *et al.*, "Internet of Things (IoT): Wireless Diagnostics Solutions", ALBA-CELLS Synchrotron, presented at ICALEPS17, Barcelona, Spain, Oct 2017, paper THSH203, this conference