Improving the performance of Taranta: analysis of memory requests and implementation of the solution

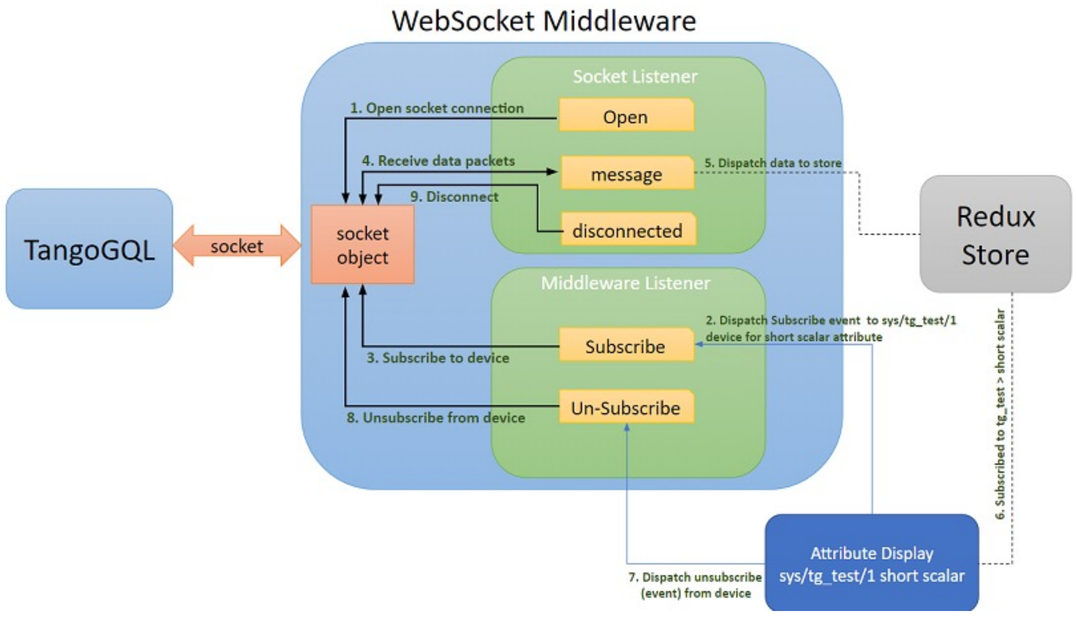
Matteo Canzari, Hélder Ribeiro, Ajaykumar Dubey, Athos Georgiou, Valentina Alberti, Yimeng Li, Vincent Hardion, Mikel Eguiraun, Johan Forsberg

ABSTRACT

Taranta is a software suite for generating graphical interfaces for Tango Controls software, currently adopted by MaxIV for scientific experiment usage, SKA during the current construction phase for the development of engineering interfaces for device debugging, and other institutions. A key feature of Taranta is the ability to create customizable dashboards without writing code, making it easy to create and share views among users by linking the dashboards to their own tango devices. However, due to the simplicity and capabilities of Taranta's widgets, more and more users are creating complex dashboards, which can cause client-side resource problems. Through an analysis of dashboards, we have found that excessive memory requests are generated by a large amount of data. In this article, we report on the process we believe will help us solve this performance issue. Starting with an analysis of the existing architecture, the issues encountered, and performance tests, we identify the causes of these problems. We then study a new architecture exploiting all the potential of the Javascript framework React on which Taranta is built, before moving on to implementation of the solution.

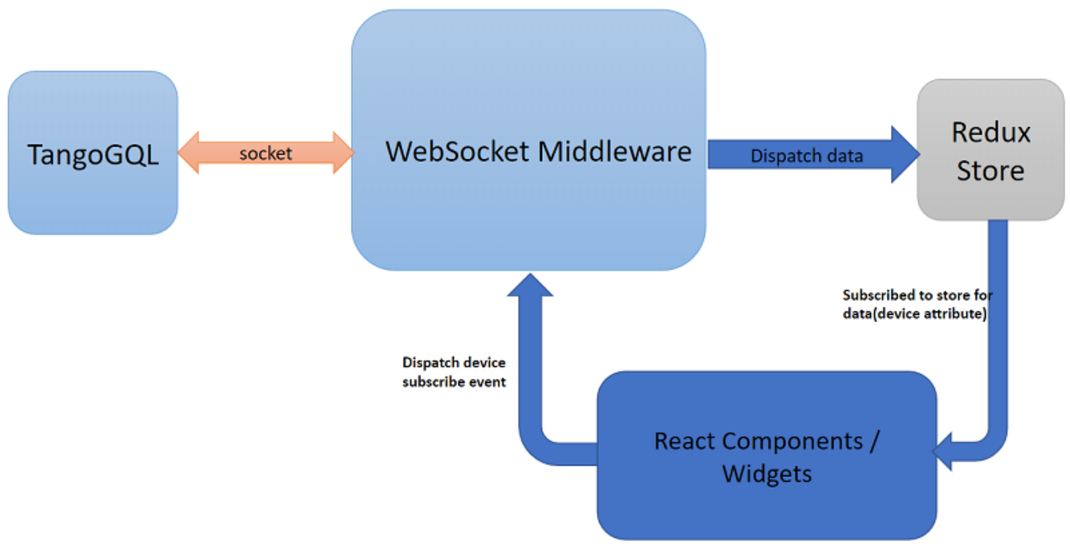
Problem impact

Taranta users experienced a slowdown and eventual halt in dashboard updates due to excessive widgets, leading to increased browser RAM usage. Lack of a runtime notification mechanism misled users about data updates from Tango. Severe slowdowns froze dashboards without providing user feedback. Unresponsive commands caused uncertainty about system functionality. Escalating browser RAM usage posed a risk of system slowdowns and crashes. Promptly addressing this issue was crucial to restore user confidence in Taranta, especially for dashboards with many widgets. Developers were aware of performance limitations but prioritized new feature development. Community reports highlighted increasing software adoption for complex use cases, showcasing Taranta's growing utility and user confidence.



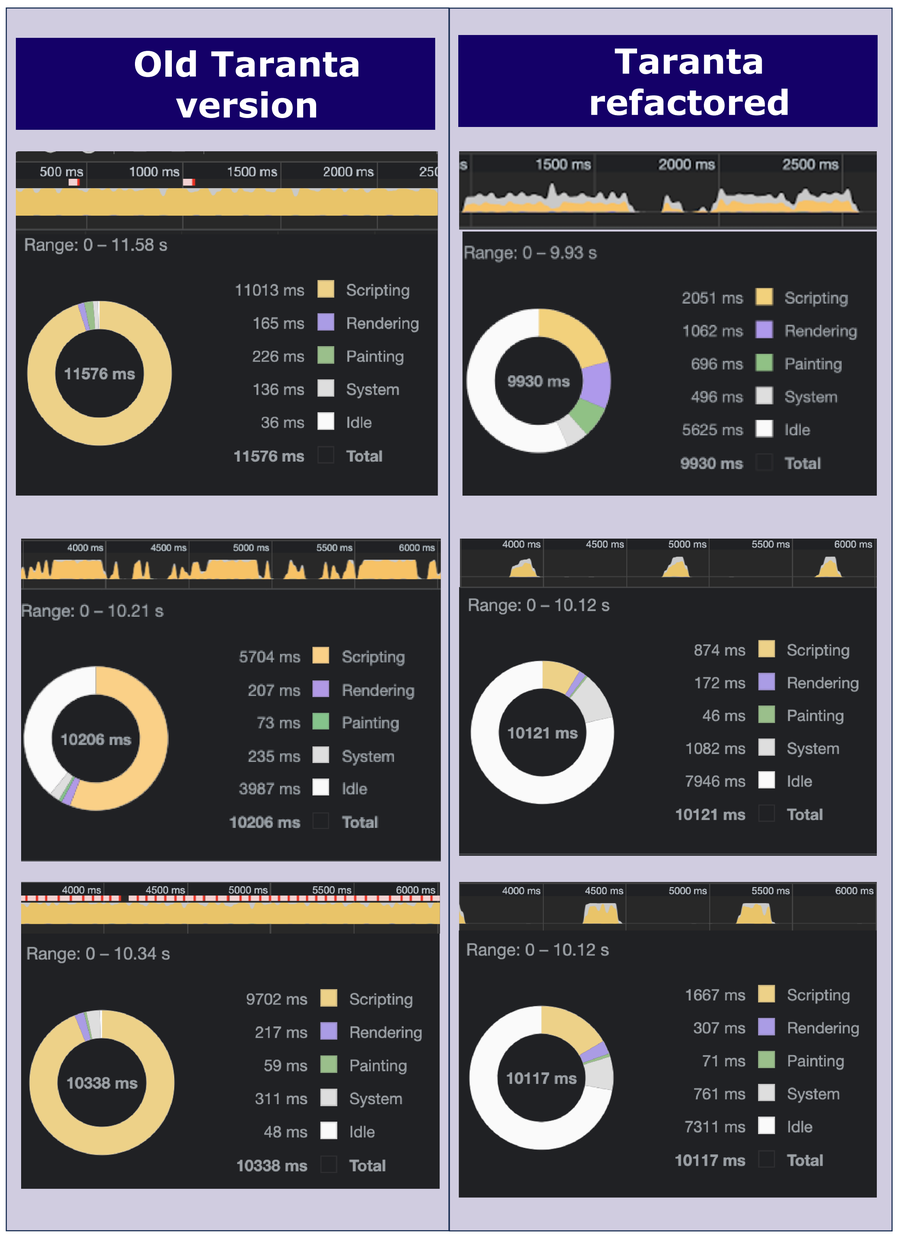
Solution and implementation

To improve architecture, focused on optimizing component-data interaction. Redesigned to trigger components only with relevant data, preventing unnecessary renders. Decoupled components from Tango APIs, enhancing modifiability and extensibility using an internal store and middleware for data subscriptions.



Data population, subscription, and unsubscription are managed by the Redux middleware. Middleware extends the store's dispatch method, enabling additional code execution and logic before actions reach the reducer. The communication with TangoGQL is facilitated by a specific middleware named websocketmiddleware, responsible for subscribing, unsubscribing, and populating the store with the requested data.

Benchmark result



INAF ISTITUTO NAZIONALE DI ASTROFISICA

TARANTA TANGO ON WEB

Contact me: matteo.canzari@inaf.it