DD2482 AUTOMATED SOFTWARE TESTING AND DEVOPS

IoT fleet management with Thingsboard.io

An overview from a DevOps perspective

Fabian Segatz
segatz@kth.se

May 23, 2022

1 Introduction

In their mobility report from November 2021, Ericsson estimates that by 2027, there will be 30.2 billion interconnected internet of things (IoT) devices. Meaning nearly every object around us: traffic lights, thermostats, electric meters, fitness trackers, water pumps, cars, elevators, and even gym vests will be connected to the internet [Ericsson, 2021].

Large amounts of IoT devices - usually referred to as fleets - are often managed by one operator. I.e. Stockholm Exergi, a local energy company in Stockholm, operates a fleet of about 9000 energy monitoring devices [Andersson, 2018]. It is rather obvious, that smart solutions for monitoring and maintaining a such number of devices are needed.

3 of the 5 most valuable companies in the world (Microsoft, Alphabet & Amazon) have realized the potential of this market and each offer an **IoT device management tool**, included in their cloud services. Other than that, there are also open-source tools such as Thingsboard available, that enable companies to have a scalable management of their IoT fleet and allows a DevOps style development, without being dependant on an external cloud provider.

This essay targets to present the features and capabilities that Thingsboard offers, to help you improve from a DevOps perspective. Particularly, features that help to reduce the time-to-market of IoT products, accelerate the time-to-resolution in case of error, and additionally give you relevant data for advertising your product.

2 Thingsboard

Before discussing the advantages that arise from using Thingsboard, let's have a brief look at its architecture. Figure 1 is adapted from the official Thingsboard documentation and gives a good overview of the interconnected parts.

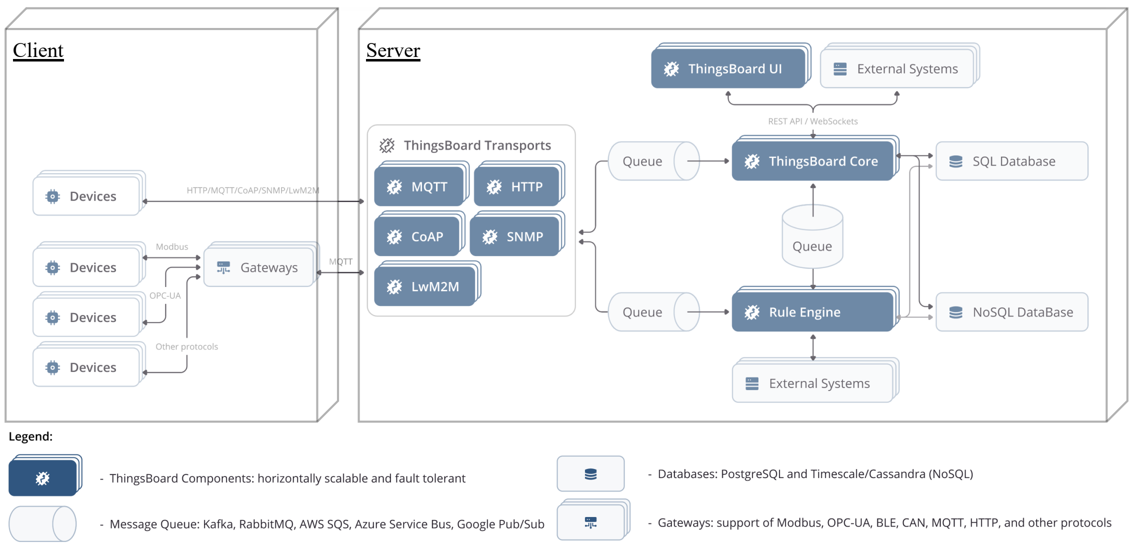

Figure 1: Architecture of Thingsboard [Thingsboard, 2022c]

The left side represents the client-side. There can be unlimited individual devices that are connected to one Thingsboard server individually or through a gateway. Each device or gateway gets its individual login credential in form of an access token that is used to verify the connection. Many different transport protocols can be used of which HTTP, MQTT, and CoAP are the most common with IoT devices.

The right side represents the server-side. The **Thingsboard UI** can be used to administrate the server or to interface with devices through dashboards. The **Thingsboard Core** is the main program, which is taking care of the communication with the devices, the UI, and additional databases. On top of that, it is also controlling the **Rule Engine**, which is an easy-to-use framework for building event-based workflows. It allows generating server-side applications, that interact with devices. Those rules can be triggered by messages from devices or a large variety of other events.

It is important to mention, that Thingsboard, does not offer its own firmware or software for the clients. However, their documentation is very extensive and gives good examples of how to connect any device via the typical communication protocols.

3 Key features

Using Thingsboard or any other IoT device management platform should definitely be considered for any IoT project. There are many use cases, that can elevate the quality of your projects by the means of DevOps. I found that the most relevant use cases can be pooled into four categories (key features): Monitoring, Remote control, Server-side applications, Scalability. In the following sections, we'll look into each key feature and discuss in which ways they can help to improve your life as a developer and increase the satisfaction of your customers.

Relevant topics are typed in bold letter in the following chapter. To get a quick overview over discussed topics, it can help to skim through this section briefly before reading it in detail.

3.1 Monitoring

In modern software engineering, it is understood that **software testing** is its own discipline and not something that is only applied ad hoc when software is not working. Testing needs a lot of expertise, planning and mostly go side-by-side with every step of the software engineering process. In DevOps, where software life cycles usually goes beyond the initial delivery, it is of great benefit, when testing can be continued even after deployment.

Thingsboard can be used as a central point to collect test results, general monitoring data to evaluate the system's performance, as well as sensor data measured by the embedded IoT device. This data can then simply be stored in a server-sided database and then be used for **long-term analysis**. This is valuable i.e. to the marketing department, which can use the data for advertisement. Other than that, also the work of business analysts is dependent on such data.

Just as relevant as long-term data collection is the feature of **real-time monitoring and visualization**. Thingsboard offers an advanced tool to create dashboards as part of its Thingsboard UI component. Figure 2 shows the dashboard from a smart metering demo example. Dashboards can not only be used internally but also made public in case live telemetry needs to be accessible to end-users.

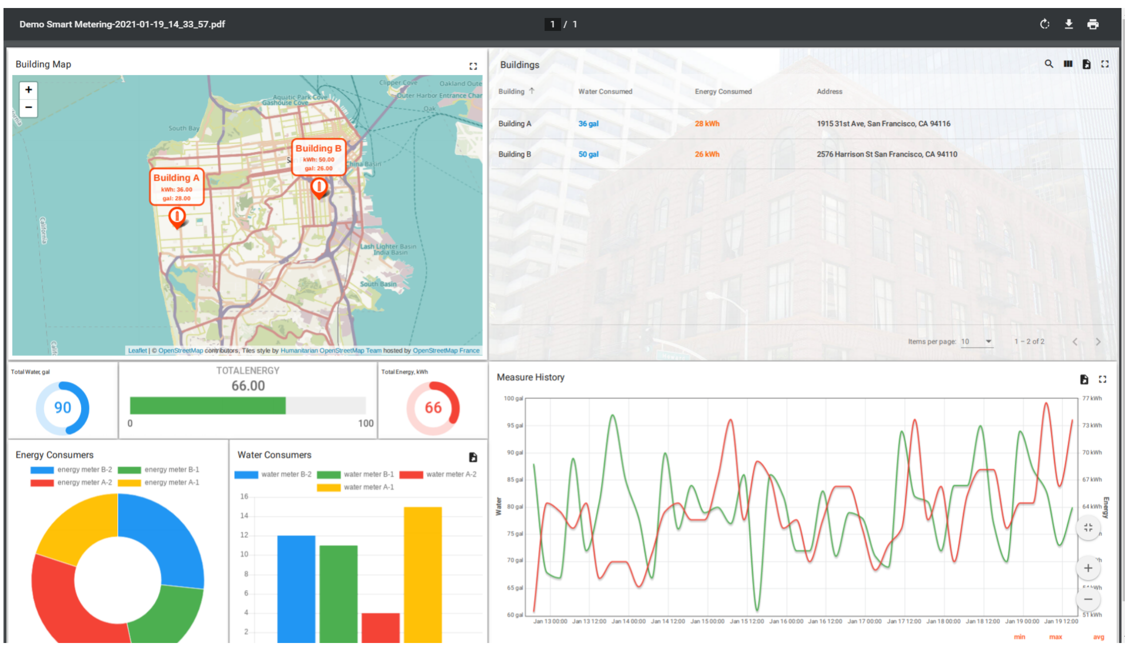

Figure 2: Smart metering demo dashboard [Thingsboard, 2022d]

Ultimately, Thingsboard also allows feeding log data to an **automated pipeline process**, which is executed by the rules engine. This is useful when data needs to be processed automatically. This will be elaborated on further in section 3.3, where server-side applications are discussed.

3.2 Remote control

Dashboards can not only be used for data logging and visualization but also to remotely control IoT devices. It is important to mention, that every interface to control a device, needs to be specifically implemented on the client-side. A typical DevOps example use-case for this would be a control panel for toggling **feature flags**.

This remote control capability can even get used for **software and firmware deployment** on connected devices. This is very important to ensure a short meantime-to-resolution. Software bugs can easily be fixed remotely, which is effectively reducing downtime, as well as travel- or transportation cost. Moreover, it's relevant from a **security** perspective. Vulnerabilities can be patched easily without needing to access the concerned devices physically.

With the rise in popularity and the increase of IoT usage, there is a growing need to develop new methods to cope with security concerns. The DevOps philosophy answers the security topic in a novel way: Since there is no possibility to know every vulnerability from the beginning and create a perfectly secure system, it is rather important to resolve security issues with minimal downtime to maximize customer satisfaction. Remote deployment allows to patch weak points in the fastest way possible.

One main difficulty when developing embedded IoT products is, that they are mostly used in a very specific context. Thus, it is not possible to just create a product and then sell it to multiple clients. Instead, there is often a baseline product that must be adjusted to the individual client's needs. With Thingsboard, it is possible to reduce the time-to-deployment, because products can be deployed with their core functionality immediately and then later be adjusted through remote deployment.

3.3 Server-side applications

Modern software products are often delivered as **software as a service (SaaS)**. This means that clients don't need to operate the software themselves. Instead, they pay a usage subscription fee to the operator, which is often the developer of the software. The operator needs to make sure, that the product is working at any time and adjust it according to the customer's individual needs. Thingsboard is the base point for extending a company's product line to SaaS products. An example use case would be online image classification. Embedded computers usually don't have the computational power to do even slightly complicated calculations in real-time. Most of the time they are optimized on energy consumption. Luckily, Thingsboard's rule engine allows to set up server-sided pipelines that can be triggered by incoming telemetry messages from devices. Thus, one could create a pipeline, that gets triggered by incoming image data. The image is then processed by a server-sided classification algorithm before the result gets sent back to the device for further use.

A typical pipeline created with the rules engine is sketched in figure 3. There are many predefined nodes that can be combined via an easy graphical user interface. Special functionality can be coded in JavaScript into script nodes (see nodes B and D).

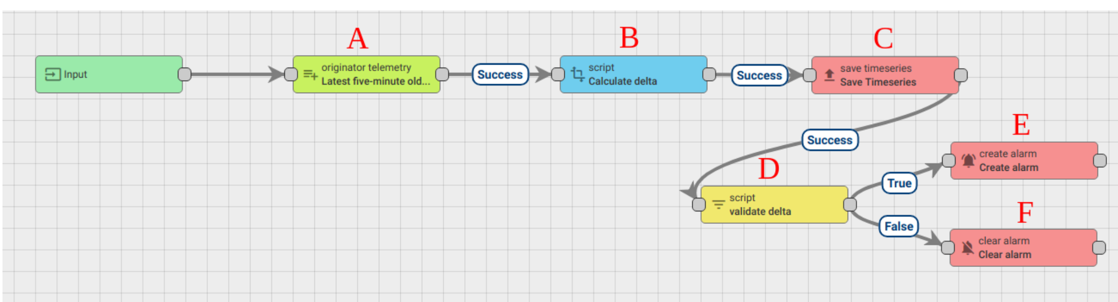

Figure 3: Example pipeline created with Thingsboard's rule engine [Thingsboard, 2022b]

Server-sided applications can be interesting to companies that specialised in IoT for multiple reasons. Again security concerns can be a driving force behind storing and **running business logic server-sided**. Especially with mainstream products, that are sold in large amounts, measurements need to be taken to prevent reverse engineering and copying of the products by malicious agents.

Other than that it can also help reduce the production cost for the devices. Instead of integrating storage hardware on the IoT device itself, Thingsboard allows using **online storage** on servers instead.

Relevant is also the use-case for **automatic anomaly detection**. Monitoring data and analyzing it manually is great, but sometimes it just has to go faster. "Machine learning techniques enable the development of anomaly detection algorithms that are non-parametric, adaptive to changes in the characteristics of normal behaviour" [Omar et al., 2013]. Such as previously stated, are embedded IoT devices not so well suited for running machine learning applications, thus anomaly detection must be run server-sided. It is of great advantage when malicious behaviour can be detected. Only then it is possible to automatically take predefined countermeasures, like disabling the attacked device or informing the operator to decide on individual actions.

In a way, this possibility to react automatically to an event and send out status messages to the operator, can be described as a type of **software bot** - another classic DevOps concept. Software bots can be used in many useful ways. I.e. are IoT devices often used to monitor a process. Via the Thingsboard rule engine it is easy to set up a alarm to notify operators or end-users when anything relevant happens during the process.

Even other devices could be notified, which can then i.e. be triggered to start a new process automatically themselves. Thingsboard allows operating and maintaining **relationships** between interconnected fleets of IoT devices easily. I.e. when two devices are dependent on each other and one of them fails, the rules engine allows to easily modify the relationship to replace the broken device with a new one, without the need to change firmware on the still-working device - again effectively reducing the meantime-to-resolution.

3.4 Scalability

Maintaining a fleet of IoT devices does not exclusively mean making sure that existing functionality is preserved. It can also include developing new applications, updating devices to perform better, or increasing the number of maintained devices. There is a necessity to improve the **scalability** of IoT solutions and Thingsboard is the perfect tool for that.

First of all, one organization can use one central Thingsboard server to operate different IoT fleets **isolated** from one another. So-called tenant profiles allow to have individual rule engines and even run isolated Thingsboard cores. This is important to prevent unwanted interference between different fleets, while still maintaining easy administration.

When additional devices shall be added to a fleet Thingsboard offers two features. Device profiles can be described to create a **common set of rules and permissions** which new devices can inherit from. Together with the second feature of **automatic provisioning**, this can greatly accelerate the process of integrating new devices.

Another advantage when using a central IoT fleet management tool is, that it allows taking **actions on many devices at the same time**. I.e patch a security update, send a notification to a device, or disable a complete fleet in emergency situations.

Last but not least, since Thingsboard is nothing else but a simple server application, we can of course also **run it in the cloud**. Thus, processing power and storage size can be flexibly adjusted according to the current demand.

4 Conclusion

Already today there are many companies, that professionally implemented Thingsboard into their products. Use cases span a variety of different areas, such as smart energy, precision farming, smart buildings, telecommunication, and industry 4.0 [Thingsboard, 2022a].

Most references described Thingsboard as a flexible and easy-to-learn tool, which is visually appealing. Many highlighted the great customer support, provided by the Thingsboard team. Other than that, it was often mentioned, that using their tool greatly helped with reducing the time-to-market. Two references said explicitly, that this was because Thingsboard could be used instead of developing an own IoT platform management tool.

Besides all the credit for the product itself, some references highlighted how Thingsboard was used for archiving DevOps goals. So were among others Thingsboard's scaling capabilities praised. The possibility of deploying applications server-sided was seen as an achievement that makes delivery easier. One customer, who specializes in industry 4.0 stated, stated that Thingsboard's monitoring feature helped them to improve their performance process. Defects can be qualified faster and production managers can plan their orders better.

As it is open-source, Thingsboard can be tested for free. When advanced support is required later in the process, one can subscribe to their professional edition for a monthly fee.

To round up, DevOps is not practiced much in embedded software development as certain characteristics of the embedded domain present obstacles to DevOps adoption. In the web domain, however, applying DevOps is common practice [Lwakatare et al., 2016]. IoT, as the exact intersection between those two domains, allows applying a subset of DevOps principles. Thingsboard is an easily accessible tool, that helps achieve this, without large cost overhead. It provides solutions to many key problems that arise when developing or maintaining IoT fleets. In this essay I gave an overview of a few relevant use cases, but I can just recommend visiting their website `www.thingsboard.io` to dig further into there platform.

References

[Andersson, 2018] Andersson, A. (2018). Stockholm exergi och telia i samarbete för effektivare energisystem i smarta fastigheter - stockholm exergi. `https://www.stockholmexergi.se/nyheter/stockholm-exergi-och-telia-i-samarbete-for-effektivare-energisystem-i-smarta-fastigheter/`. Accessed: May 17, 2022.

[Ericsson, 2021] Ericsson (2021). Broadband iot still rising as 2g and 3g continue to decline. *Ericsson Mobility Report November 2021*, page 18.

[Lwakatare et al., 2016] Lwakatare, L. E., Karvonen, T., Sauvola, T., Kuvaja, P., Olsson, H., Bosch, J., and Oivo, M. (2016). Towards devops in the embedded systems domain: Why is it so hard? In *Proceedings of the 49th Annual Hawaii International Conference on System Sciences*.

[Omar et al., 2013] Omar, S., Ngadi, M., Jebur, H., and Benqdara, S. (2013). Machine learning techniques for anomaly detection: An overview. *International Journal of Computer Applications*, 79.

[Thingsboard, 2022a] Thingsboard (2022a). Industries. `https://thingsboard.io/industries/`. Accessed: May 22, 2022.

[Thingsboard, 2022b] Thingsboard (2022b). Telemetry data validation. `https://thingsboard.io/docs/user-guide/rule-engine-2-0/tutorials/telemetry-delta-validation/`. Accessed: May 19, 2022.

[Thingsboard, 2022c] Thingsboard (2022c). What is thingsboard? `https://thingsboard.io/docs/getting-started-guides/what-is-thingsboard/`. Accessed: May 17, 2022.

[Thingsboard, 2022d] Thingsboard (2022d). Working with iot dashboards. `https://thingsboard.io/docs/user-guide/dashboards/`. Accessed: May 19, 2022.