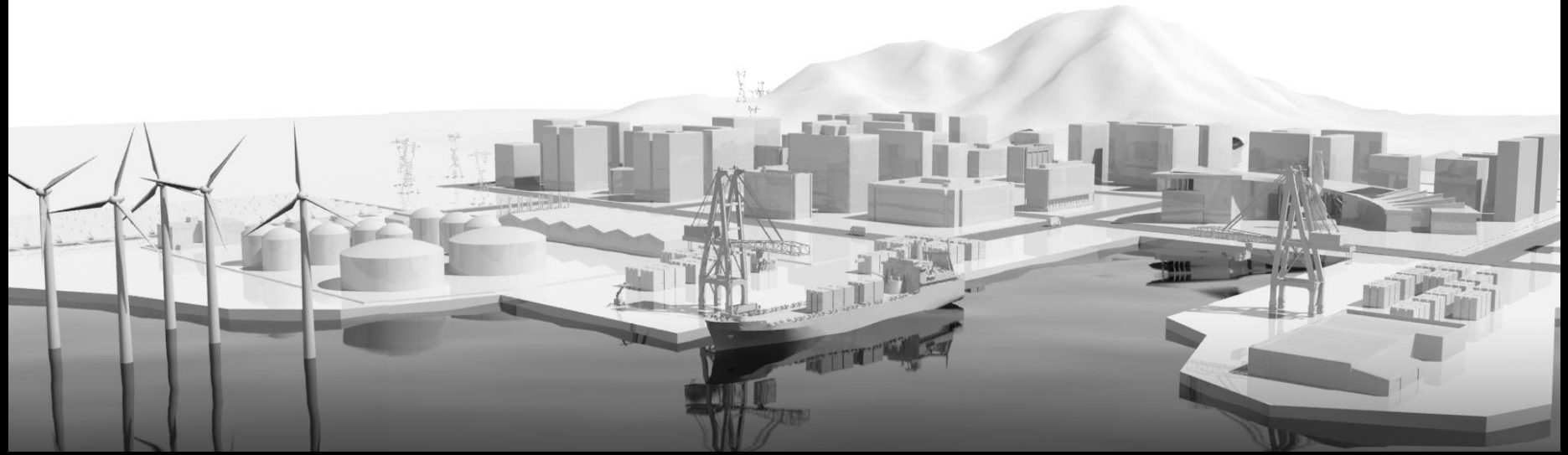
IoT-TICKET Architecture

Microservice Success Story



IoT-TICKET
About presenter

- *Jaakko Takaluoma 40+, Wapice Ltd.*
- *Location: Vaasa, the sunniest city in Finland*
- *Position: Lead architect, business solution segment*

- *Hobbies: fixing things, fishing, Judo*
  (note, no directly computer related hobbies !!!)

IoT-TICKET
About presentation

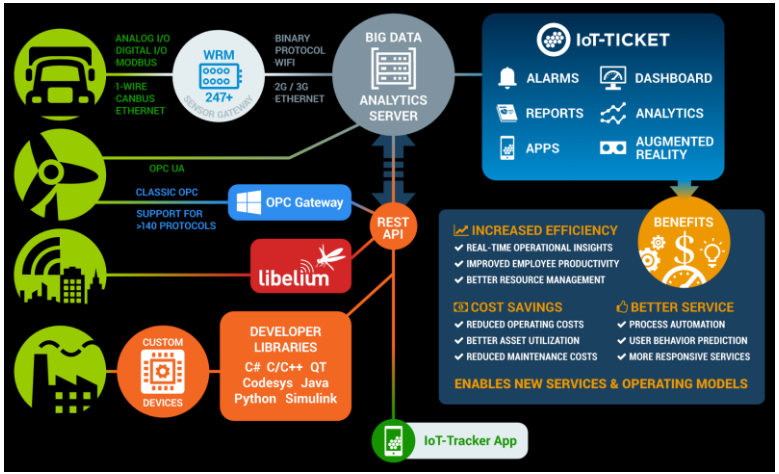
This presentation opens our experiences, mainly positive, about microservice architecture in context of our flagship product IoT-TICKET architecture renovation projects during 2015-2016

Developer perspective is included, the author was part of the development team during the transition to the new architecture both designing and implementing

IoT-TICKET
About the product

https://www.iot-ticket.com

- IoT-Ticket is the most comprehensive IoT product in the market today - It is the office suite of IoT
- Ask for a separate demo, or register and try it yourselves
- Technically speaking, it is cloud-able software, scales and is designed to store gigantic amount of data
- Behind the scenes, there are
  - databases (SQL, NoSQL)
  - message queues and streams
  - REST APIs
  - Java components
  - HTML5-JS-CSS SPA Web applications
  - and infrastructure to run these components



IoT-TICKET
Some history

https://www.iot-ticket.com

- The roots of the IoT-Ticket are over 10 years old
  - Year 2005, monolithic *application server* based architecture was considered top design
  - We developed with Java EE principles using the best practises available (incl. layered modular architecture with EJB service facade, utilized App server resources)

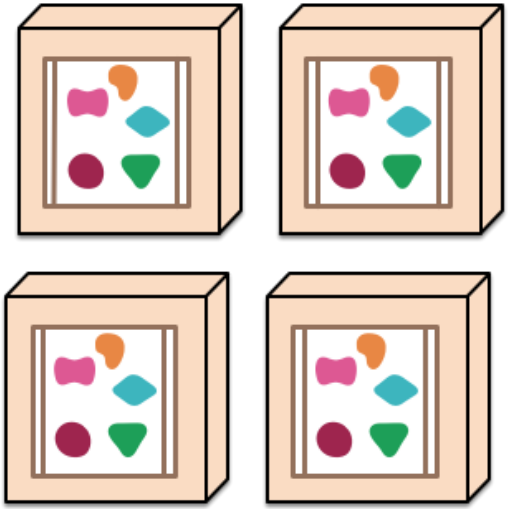
- At the end of 2014 we decided to start changing the *existing software* architecture to microservice architecture...

IoT-TICKET
monolith vs microservice

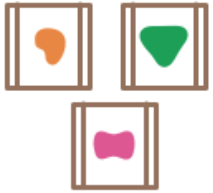
The picture is borrowed from
https://martinfowler.com/articles/microservices.html

A monolithic application puts all its functionality into a single process...

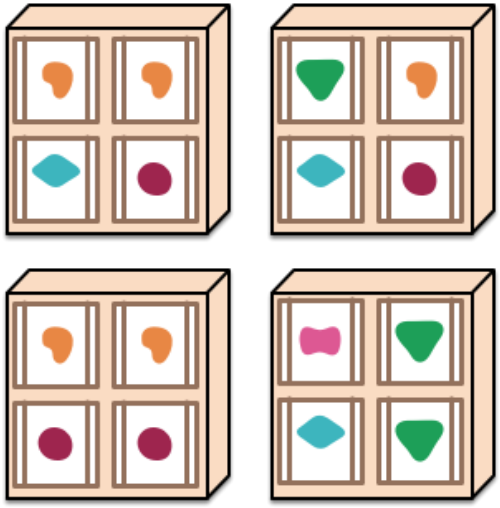
... and scales by replicating the monolith on multiple servers



A microservices architecture puts each element of functionality into a separate service...

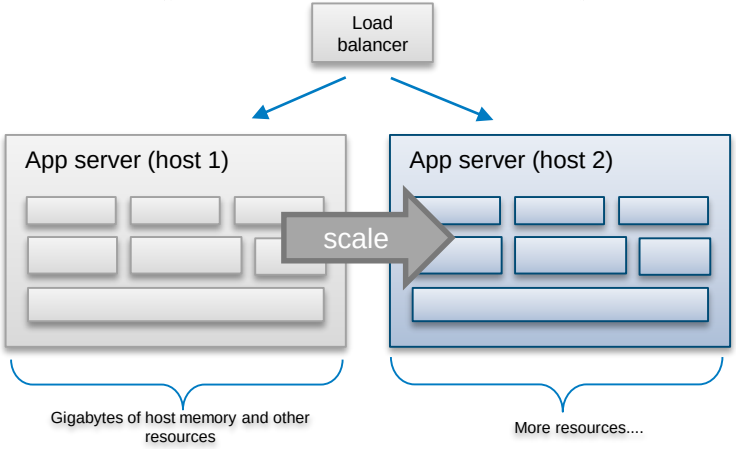


... and scales by distributing these services across servers, replicating as needed.



IoT-TICKET
Monolithic architecture

- Very difficult to scale apps efficiently
- Cloud support is restricted
- Eventually developers were not happy with the Java EE stack
  - It was too heavy – debugging was partially impossible
  - Project structures became enormous (with Eclipse at least)
  - Changes made to code required restarts and waiting..
  - Web UI developers did not want to run the backend on their development environments



IoT-TICKET
microservice architecture

- Benefits
  - Cloud support is excellent
  - Smaller service components are easier to develop and debug
- Developers don't necessarily want to be categorized as UI developers or Server-side developers – Full-stack developer suits better
  - Formerly it happened, that developers simply cannot fix bugs or develop new features on the other side of the system (server-side or UI)
- This is frustrating and waste of human resources

- With microservice architecture the developer starts only minimum set of services and runs only the interesting services in IDE with debugger
- This is both efficient and also helps people understand the system as a whole

IoT-TICKET
Results

- We have run the new architecture for almost a year now
- Good things
  - The system has become more stable
  - We have been able to fix problems in the system behind the scenes without users noticing
  - We have been able to scale critical parts on demand
  - We have been able to get younger developers produce better quality than before
  - Developers are happier in general
- Challenges
  - System setup is more complex
  - ..on the other hand once it is done, things go smoother than before
  - Logging and metrics

Wapice

Technology Partner

Software. Electronics. Innovation.

www.wapice.com