Microservice Tradeoffs & Misunderstandings

Dependencies

Some Typical Problems in Larger Organizations

Did you ever suffer from “dependencies”, no matter of “technical” or “people” dependencies?

“They upgraded the lib to a new version and it creates trouble”

“Currently blocked because the framework has a bug”

“Couldn’t finish the story because the other team’s API was not ready”

“Gosh this sucks!!!”

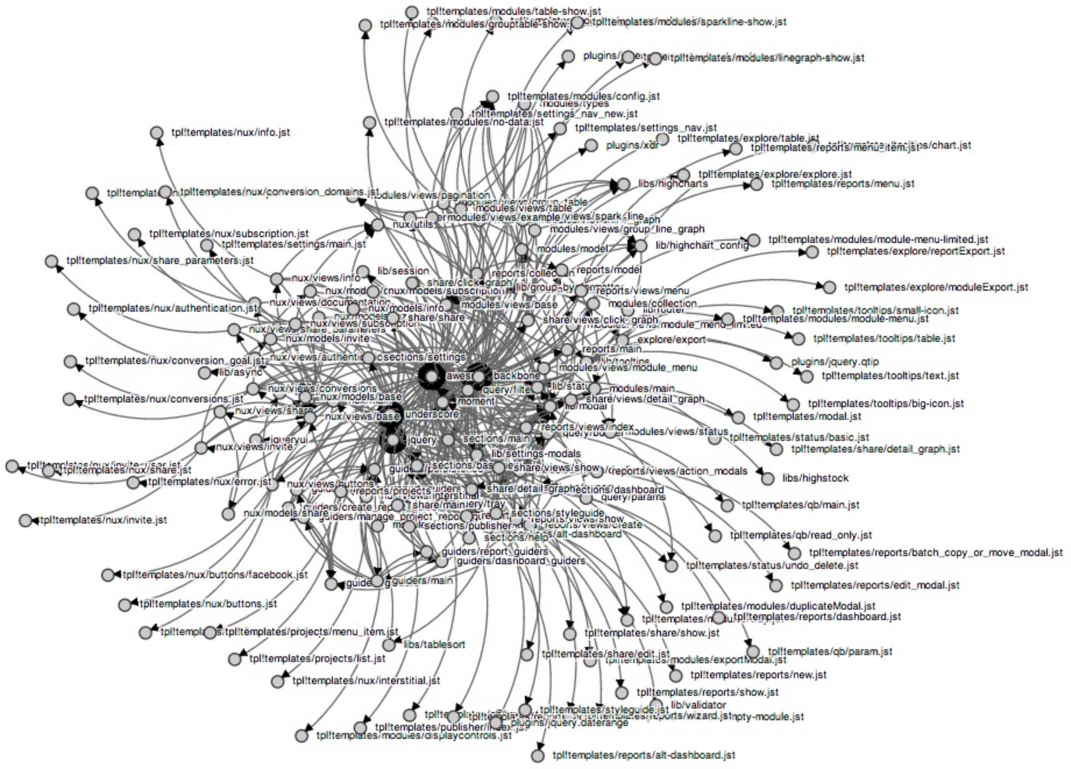
“Cannot continue because the build breaks for some other component”

“The Ops team is still working on the deployment, we need to wait”

Dependencies

Impact of Size of Services

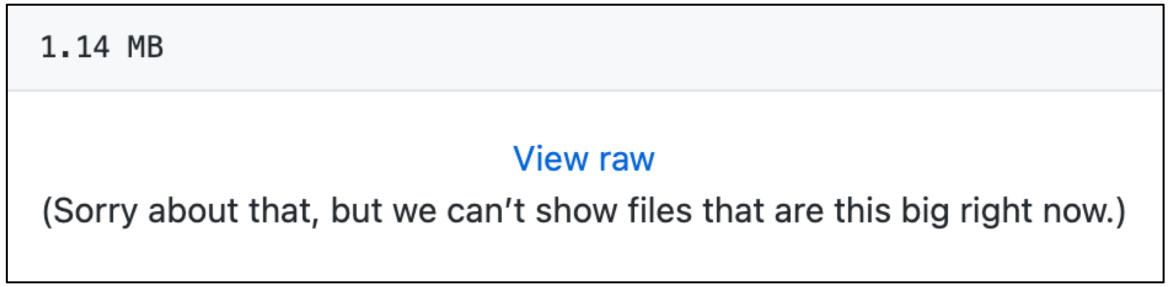
- On-Premise solution “transformed” to Cloud
- Strategy: 1 service for each 1-2 DB tables
- Result: 200+ Microservice – as small as possible
- To deploy anything, need to deploy everything
- JSON file defines all service dependencies
- Needed to get the solution deployed
- New job: maintain JSON dependency file



Note: the picture is just an example taken from http://www.mikeperham.com/2016/02/09/kill-your-dependencies/

Dependencies

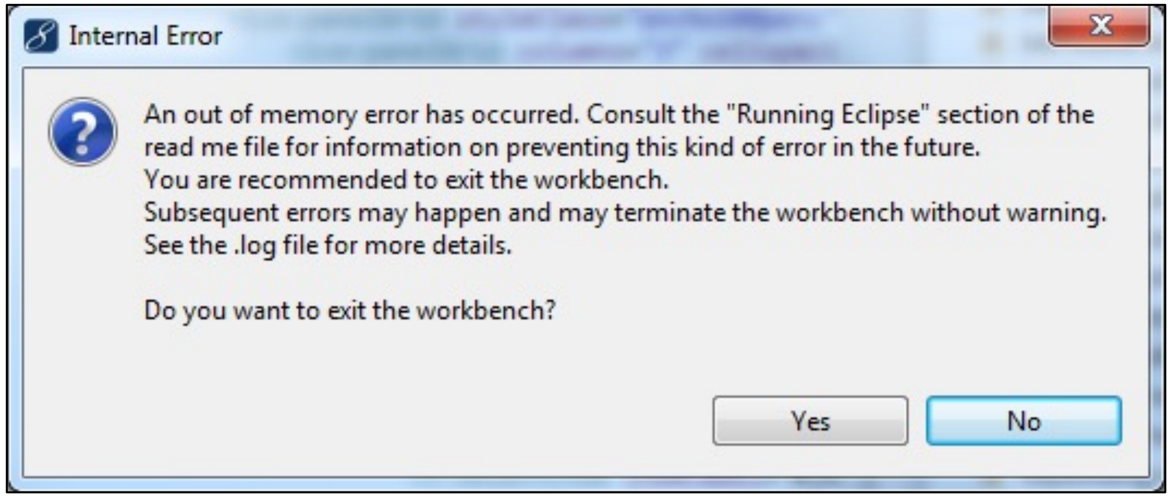
Impact of Re-Use and Coupling



1.14 MB

View raw

(Sorry about that, but we can't show files that are this big right now.)



Internal Error

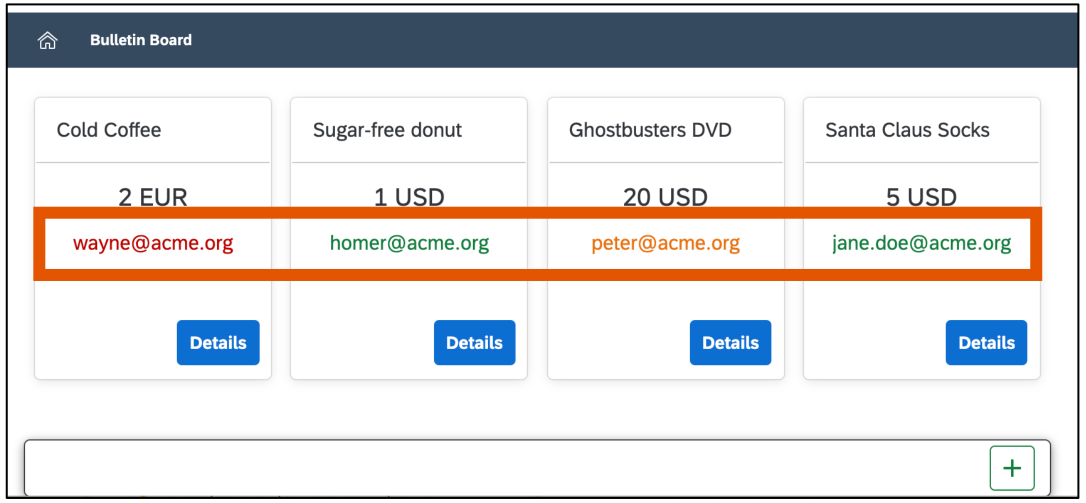
An out of memory error has occurred. Consult the "Running Eclipse" section of the read me file for information on preventing this kind of error in the future.
You are recommended to exit the workbench.
Subsequent errors may happen and may terminate the workbench without warning.
See the .log file for more details.

Do you want to exit the workbench?

- A central common service class (e.g. Product)
- More than 30,000 lines of code!
- Cannot open in IDE (out of memory)
- Reason: Maximize re-use
  - one service serves "all"
- Reduce duplication
  - dedicated team to "govern" it
- Team under pressure - becomes "bottleneck"

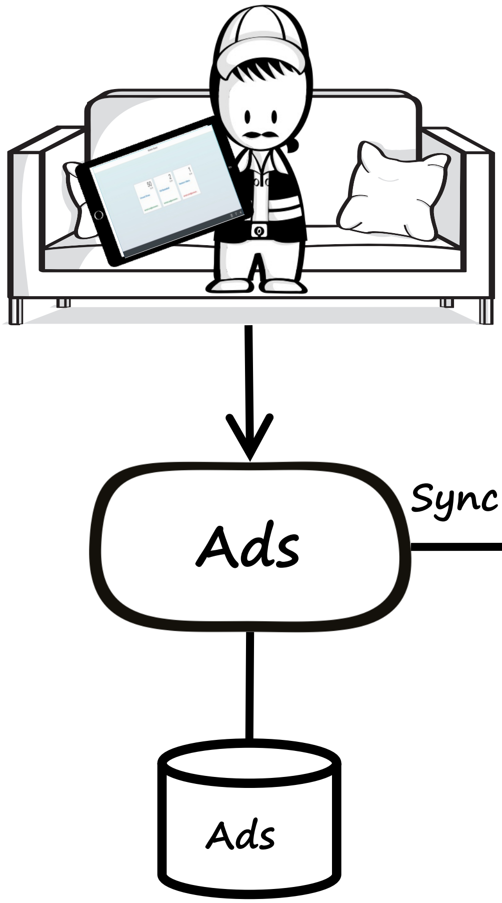
Bulletin Board Application

Scenario Overview

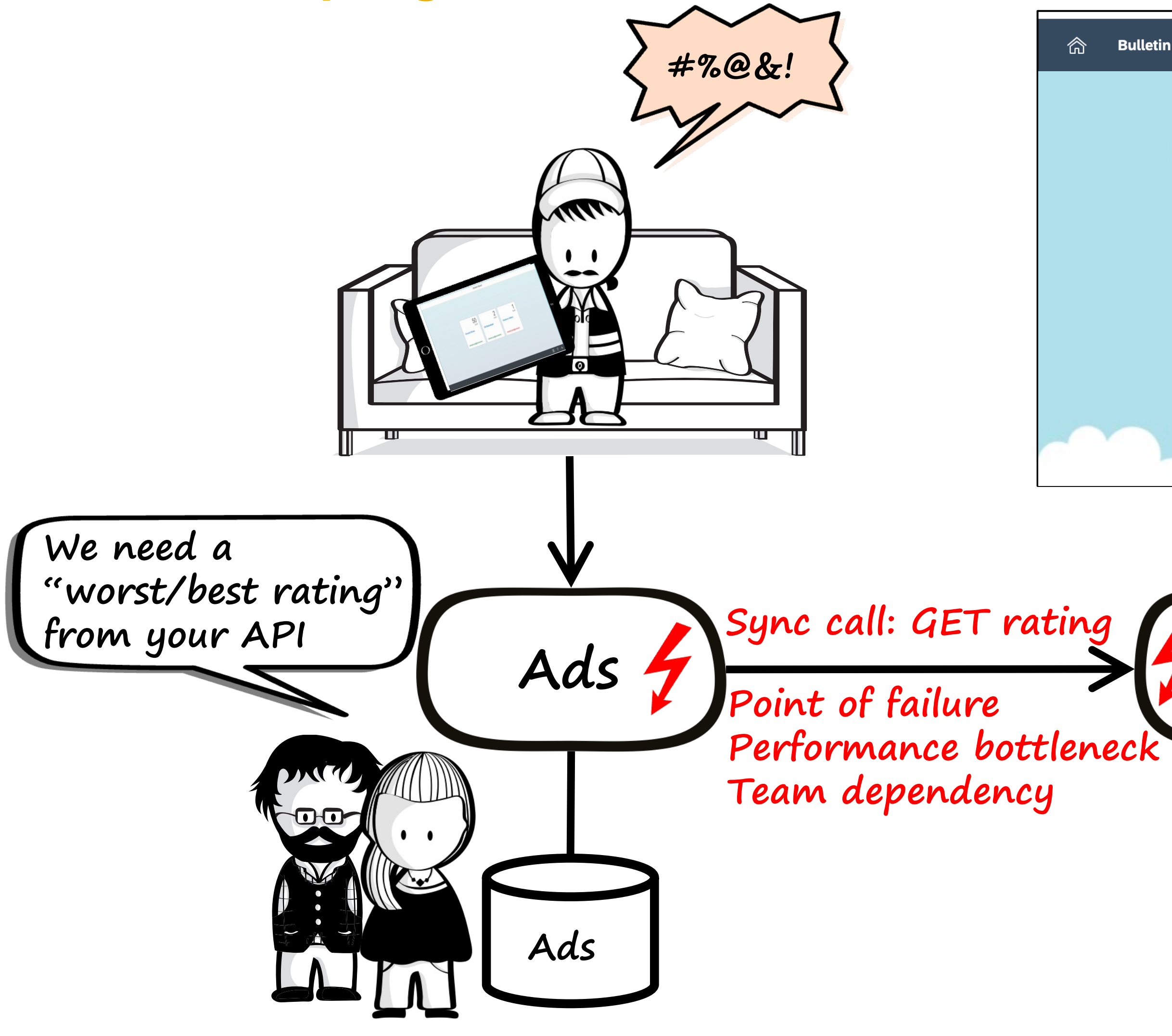


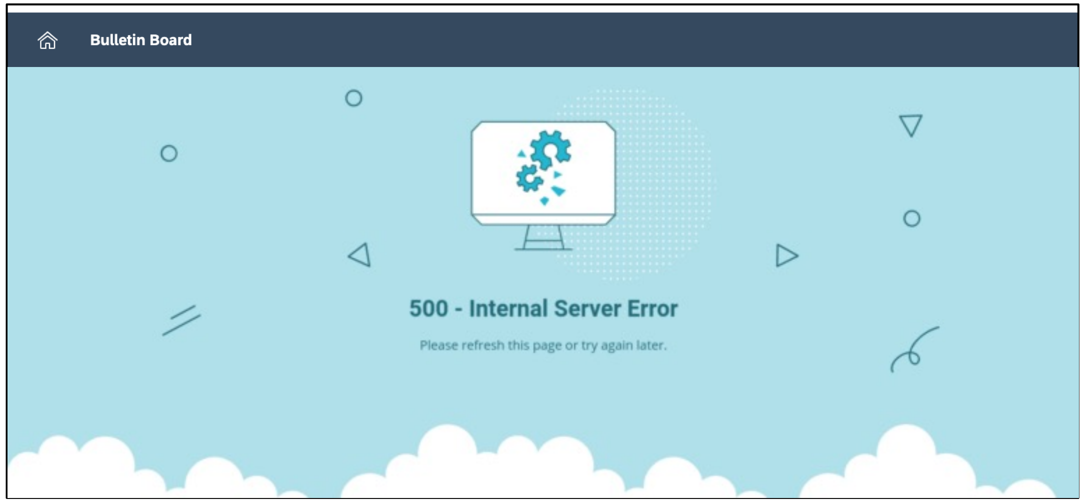
Bulletin Board

Cold Coffee	Sugar-free donut	Ghostbusters DVD	Santa Claus Socks
2 EUR	1 USD	20 USD	5 USD
wayne@acme.org	homer@acme.org	peter@acme.org	jane.doe@acme.org
Details	Details	Details	Details



Bulletin Board Application
Problem: Coupling



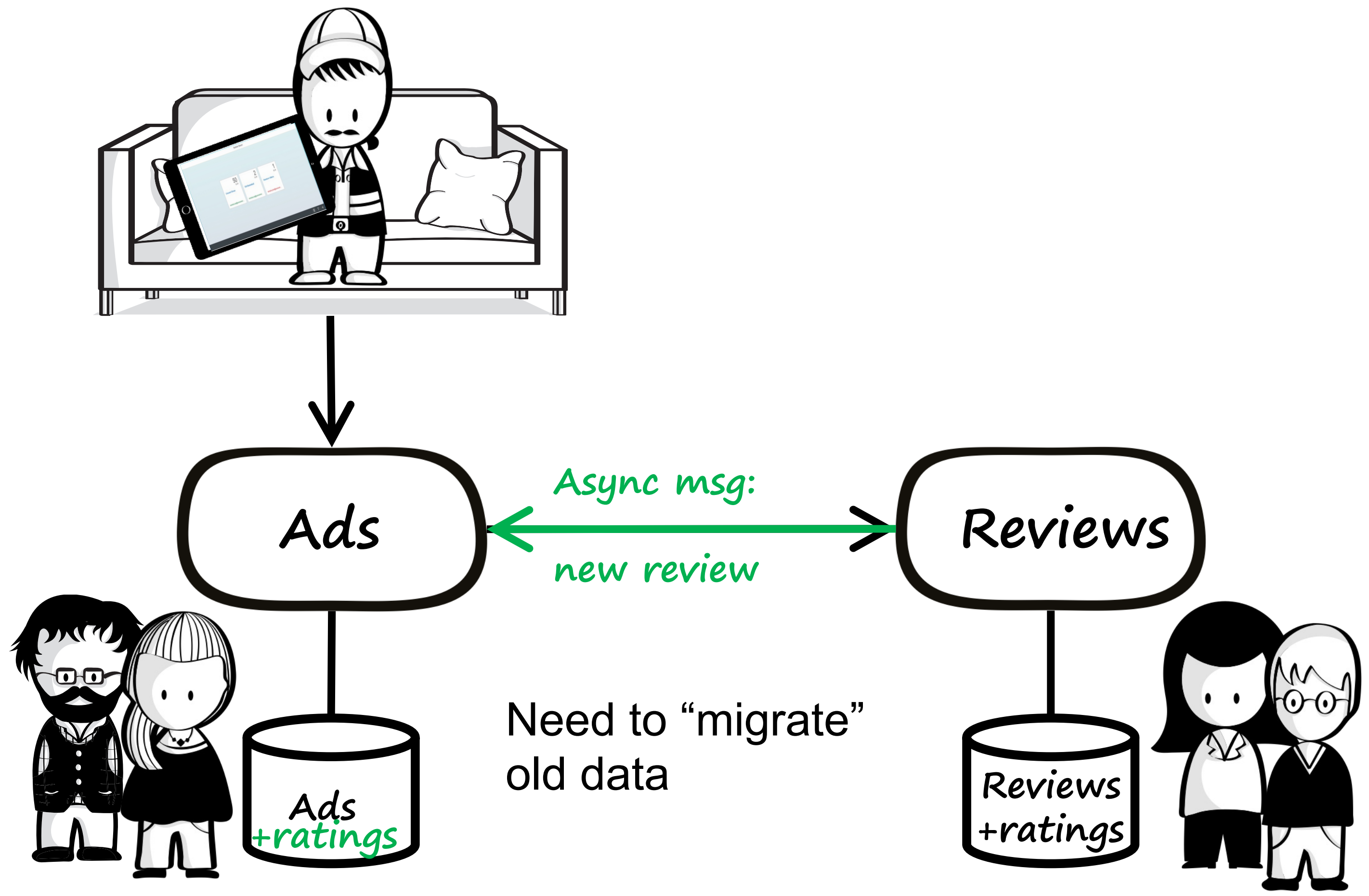


500 - Internal Server Error

Please refresh this page or try again later.

Bulletin Board Application

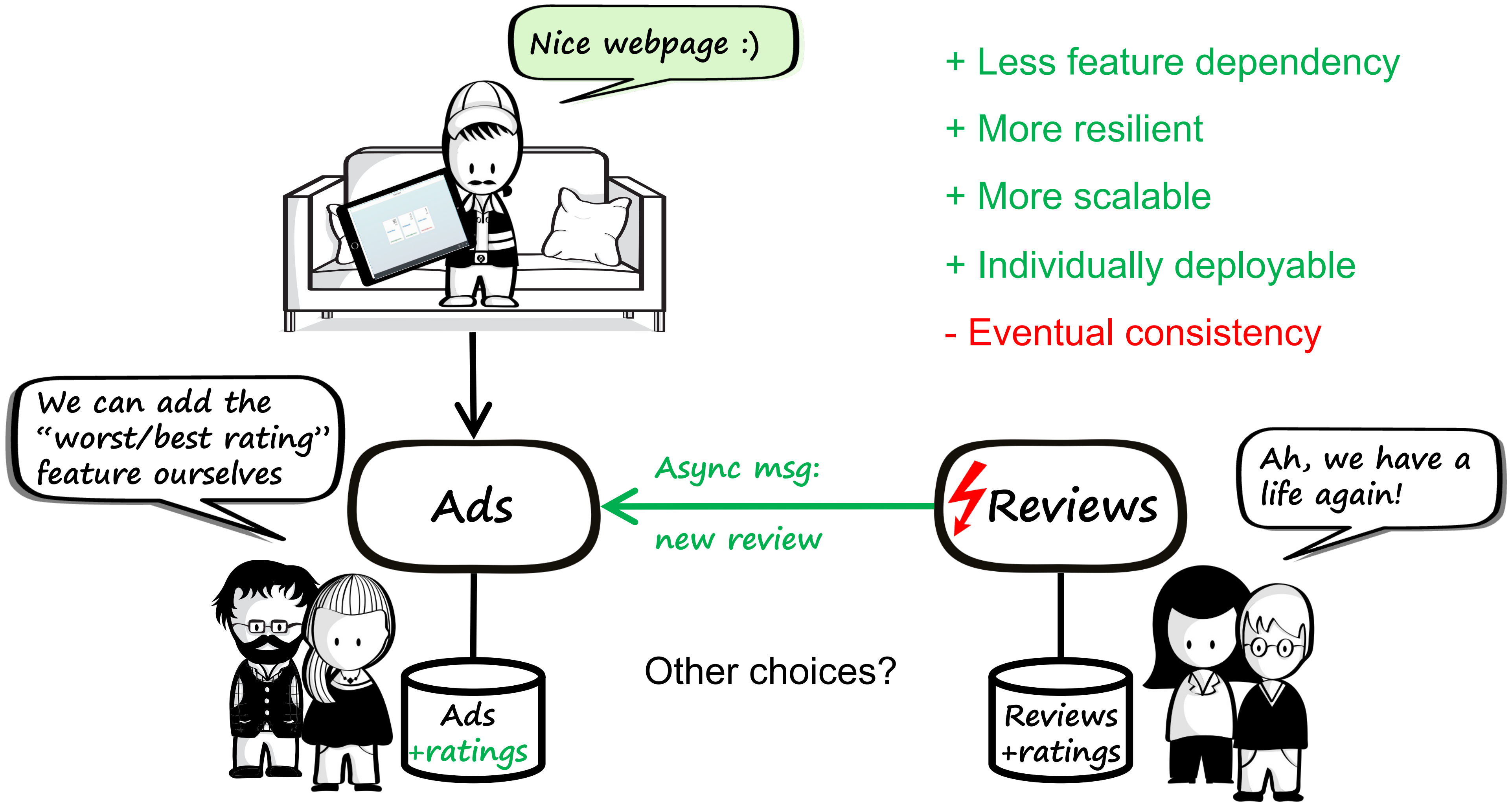
Solution: Decoupling



Need to “migrate”
old data

Bulletin Board Application

Decoupling: Advantages


Nice webpage :)
We can add the "worst/best rating" feature ourselves
Ads
Async msg:
new review
Reviews
Ah, we have a life again!
Ads +ratings
Reviews +ratings

+ Less feature dependency
+ More resilient
+ More scalable
+ Individually deployable
- Eventual consistency

Other choices?

"One of the great tradeoffs in microservices is that you have much higher levels of duplication than you would ever tolerate in previous architectures, but part of that is to get to that level of decoupling that allows you to move faster."

Neal Ford, architect and director at Thoughtworks, book author and speaker

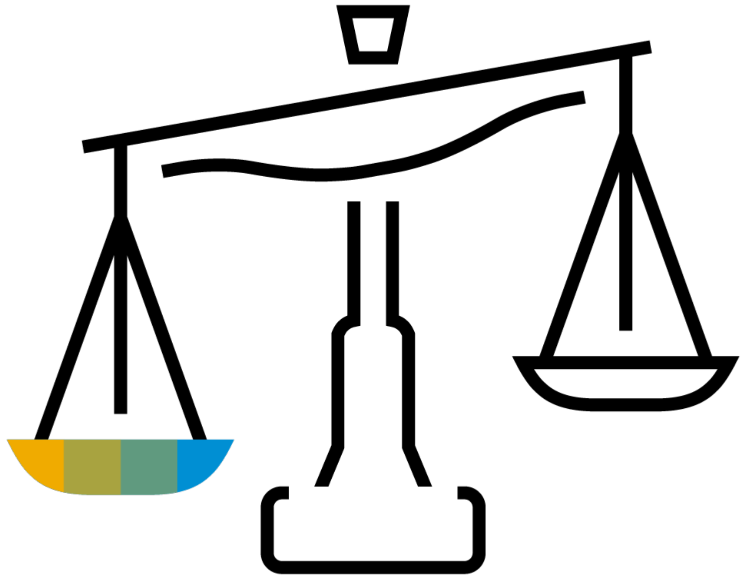
“This is one of the fundamental misunderstandings people have: they want high degrees of re-use and also a highly decoupled architecture. You cannot have both: re-use is coupling. So part of decoupling things means stopping re-using so much.”

Neal Ford, architect and director at Thoughtworks, book author and speaker

Microservices: Key Takeaways

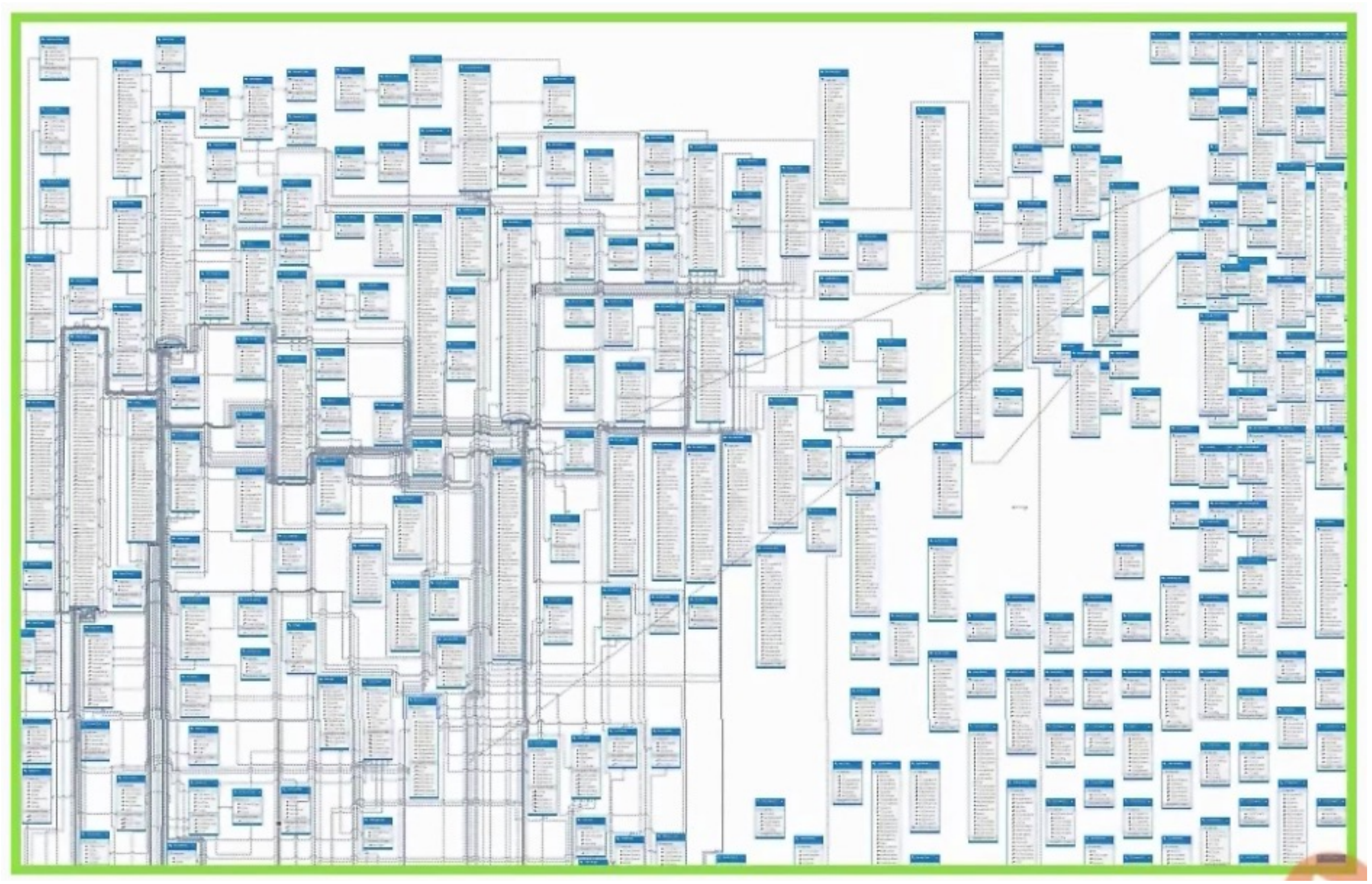
Tradeoff „Speed vs. Re-Use“

- Microservices favor **decoupling** over re-use to **increase speed**
- Re-using is not a bad thing!
  Stopping all re-use would be silly, too!
- Be aware of the tradeoff, **check which re-use decisions make us too slow!**

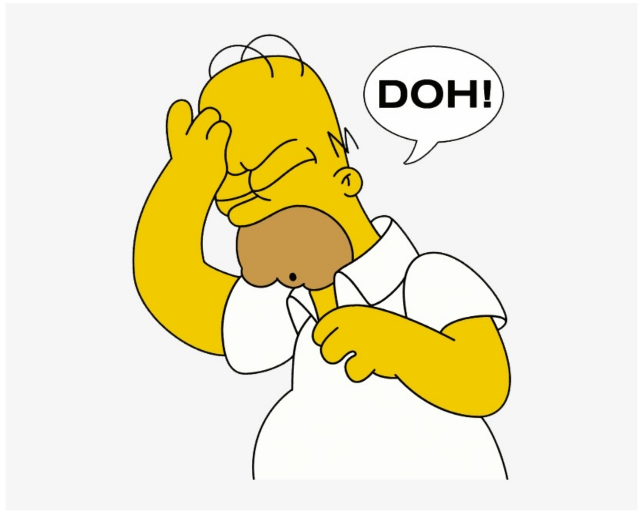


Time to market / speed of innovation is a fundamental business concern, so this is not a pure architecture / engineering decision

We Were Taught “All Duplication is Evil” for Decades



…but nobody taught us that this will be the result



Time to free up some brain memory & reset