Lab9: Microservices
Microservice Modeling and Communication Styles
T-302-HONN

Sindri Þór Guðmundsson
Ottó Ernir Kristinsson
28.10.2024

Modelling

1. Af hverju viljum við hafa lágt coupling á milli service-a?

   Því minna coupled sem tvö eða fleiri service eru því auðveldara er að viðhalda og breyta hverju service án þess að hafa áhrif á hin. Því minna coupling sem við höfum því nær komumst við markmiðinu að hafa hvert service indepenedently changed, deployed og released.

2. Af hverju viljum við hafa hátt cohesion innan hvers service?

   Hátt cohesion innan service þýðir að kóðinn innan hvers service er mjög tengdur og líklega ekki skyldur kóða í öðrum services. Þetta hjálpar okkur með að halda services loosely coupled við hver aðra.

3. Af hverju er content coupling sérstaklega “problematic” form af coupling?

   Content coupling er þéttasta tegundin af coupling þegar services teygja sig í gögn eða breyta þei hjá öðrum services. Þetta veldur því að það er mjög erfitt að vita hvort staða ganga sé rétt og gerir það að verkum að breytingar á strúktúr og gögnum er líklegt til að brjóta virkni.

4. Útskýrið hvernig það er einnig hægt að modela services eftir:
   a. Volatility

      Að modela services eftir Volatility þýðir að skipt er upp í services eftir því hversu líklegur kóði er til að breytast frekar en eftir domain boundaries.

   b. Data & Security

      Að modela services eftir Data & Security þýðir að skipt er upp í services eftir því hversu viðkvæm gögn eru og hversu mikið þarf að hugsa að öryggi þeirra. Viljum og þurfum kannksi ekki mjög strangar öryggisráðstafanir allstaðar í domain-inu og þá getur verið betra að modela eftir Data & Security.

   c. Technology

      Að modela services eftir Technology þýðir að það er tæknin sem skiptir mestu máli þegar skipt er upp í services. Mögulega þarf einhver virkni sérstaka tækni sem gerir það að góðu og cohesive service, sérstaklega ef enginn annar hluti kerfisins stólar á sömu tækni.

Communication Styles

1. Útskýrið asynchronous samskipti

   asynchronous samskipti eru blocking að því leyti að service kallar á annað og bíður eftir svari, það getur ekki haldið áfram þangað til svar berst.
   Asynchronous samskipti aftur á móti eru non-blocking þar sem service kallar á annað og heldur áfram að vinna þar til svar berst, ef svar berst yfir höfuð.

2. Úskýrið hvernig event driven architecture er náttúrulega asynchronous

   Event driven architecture er byggt á því að service “publishi” event til annara service-a, bíður ekki eftir svari og heldur áfram að vinna. Þetta er skilgreiningin á asynchronous samskiptastíl sem þýðir að event driven architecture er náttúrulega asynchronous.

3. Útfærið einfalt event-driven architecture kerfi
   notaðist við docker til að setja upp rabbit mq, en pip installaði pika eins og í linkaða tutorialinu.