Pilvepõhiste rakenduste arhitektuurid

Pelle Jakovits

Mai 2025, Tartu

Sisukord

- Pilvepõhised rakendused (*Cloud Native*)
- Kaheteist tegurit Pilvepõhiste rakenduste disainis
- Pilvepõhiste rakenduste arhitektuurimustrid

Pilvepõhised rakendused

- Pilvepõhine - *Cloud-native*
- Mikroteenused (ja nano) + Konteinerid + DevOps
- Ei liiguta rakendust pilve
  - Ehitame rakenduse pilves, pilves üles seadmiseks
    - Algusest peale pilves juurutamiseks

Pilvepõhisuse definitsioon

- Cloud Native Computing Foundation definitsioon:

Cloud-native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds.

- Selle lähenemisviisi aluseks on konteinerid, teenuse võrgud, mikroteenused, deklaratiivsed API-d, jms.
- Need tehnikad võimaldavad disainida lahtiselt-ühendatud süsteeme, mis on vastupidavad, skaleeruvad, juhitavad ja jälgitavad
- Kombineerides pilvepõhise arhitektuuri, automatiseerimise ja DevOPS metodoloogia on võimalik arendajatel minimaalse vaevaga läbi viia pidevaid ning etteaimatava mõjuga muudatusi

Pilvepõhise arhitektuuri omadused

- Ilma OS'ta arhitektuur
- Õige mahuga ressursid - Right-sized capacity
- Pidev juurutus - *Continuous delivery*
- Iseseisvus - Independence
- Automatiseeritud skaleeritavus - Automated scalability
- Kiire taastumine - Rapid recovery

Kaksteist tegurit

- *Twelve-Factor Application*
  - https://12factor.net/
- **Parimad tavad**, mida tuleks jälgida **pilvepõhiste** rakenduste disainimisel
  - Veebirakendused
  - Veebiteenused
  - Software-as-a-Service (Saas) rakendused

Tegurid

1. **Koodibaas**
   - Iga mikroteenuse jaoks üks koodibaas, mis on salvestatud oma hoidlas.
   - Versioonikontrolliga jälgituna saab seda juurutada mitmesse keskkonda (testimise, arenduse, tootmis).
2. **Sõltuvused**
   - Iga mikroteenus eraldab ja pakendab oma sõltuvused, hõlmates muudatusi kogu süsteemi mõjutamata.
3. **Konfiguratsioonid**
   - Konfiguratsiooniteave eraldatakse mikroteenusest ja muudetakse konfiguratsioonihaldustööriista kaudu väljaspool koodi.
   - Sama juurutus võib õige konfiguratsiooni korral levida erinevates keskkondades.
4. **Tugiteenused**
   - Abiressursid (andmehoidla kaustad, vahemälud, sõnumivahendajad, järjekorrad) tuleks avaldada adresseeritava URL-i kaudu.
   - See seob ressursi rakendusest lahti, võimaldades seda jooksvalt välja vahetada.

Tegurid

5. **Ehitamine, väljalaskmine, käivitamine**
   - Iga juurutus peaks olema eraldatud ehitamise (build), väljalaske (release) ja käivitamise (run) etappide vahel
   - Igaüks on märgistatud unikaalse ID-ga ja toetab tagasipööramise võimalust
   - Kaasaegsed CI/CD süsteemid aitavad seda põhimõtet tagada
6. **Protsessid**
   - Iga mikroteenus peaks töötama oma protsessis, mis on isoleeritud teistest töötavatest teenustest
7. **Portide sidumine**
   - Iga mikroteenus peaks olema iseseisev ning selle liidesed ja funktsionaalsus on avatud oma pordis
   - See tagab isolatsiooni teistest mikroteenustest
8. **Samaaegsus**
   - Kui võimsust on vaja suurendada, skaleerige teenuseid horisontaalselt mitme identse koopiana, mitte suurendades ühe eksemplari võimsust.

Tegurid

9. **Ajutisi teenuse koopiad peaks kasutama ühekordselt**
   - Eelistage kiiret käivitamist, et suurendada skaleerimise võimalusi, ja graatsilist taaskäivitamist, et süsteem õigesse olekusse jätta.
   - Teenused peaksid olema võimelised graatsiliselt seiskuma
   - Dockeri konteinerid koos orkestraatoriga vastavad sellele nõudele oma olemuselt.
10. **Arendus, testimise ja tootmiskeskkondade sarnasus**
   - Hoidke keskkonnad kogu rakenduse elutsükli jooksul võimalikult sarnased
   - Konteinerite kasutuselevõtt aitab keskkondi sarnasena hoida (lihtne teisaldada)
11. **Logimine**
   - Käsitlege mikroteenuste loodud logisid sündmuste voogudena.
   - Koguge logiandmeid kesksesse andmekaevandamise/logihaldustööriistadesse
12. **Haldusprotsessid (Administreerimine)**
   - Käivitage haldustoiminguid (nagu andmete puhastamine või arvutusanalüüs) ühekordsete, sõltumatute protsessidena
   - Administreerimiseks vajalikud tööriistad peaksid olema pakendatud kaasa tavafunktsionaalsusega

PILVEPÕHISTE RAKENDUSTE ARHITEKTUURI MUSTRID

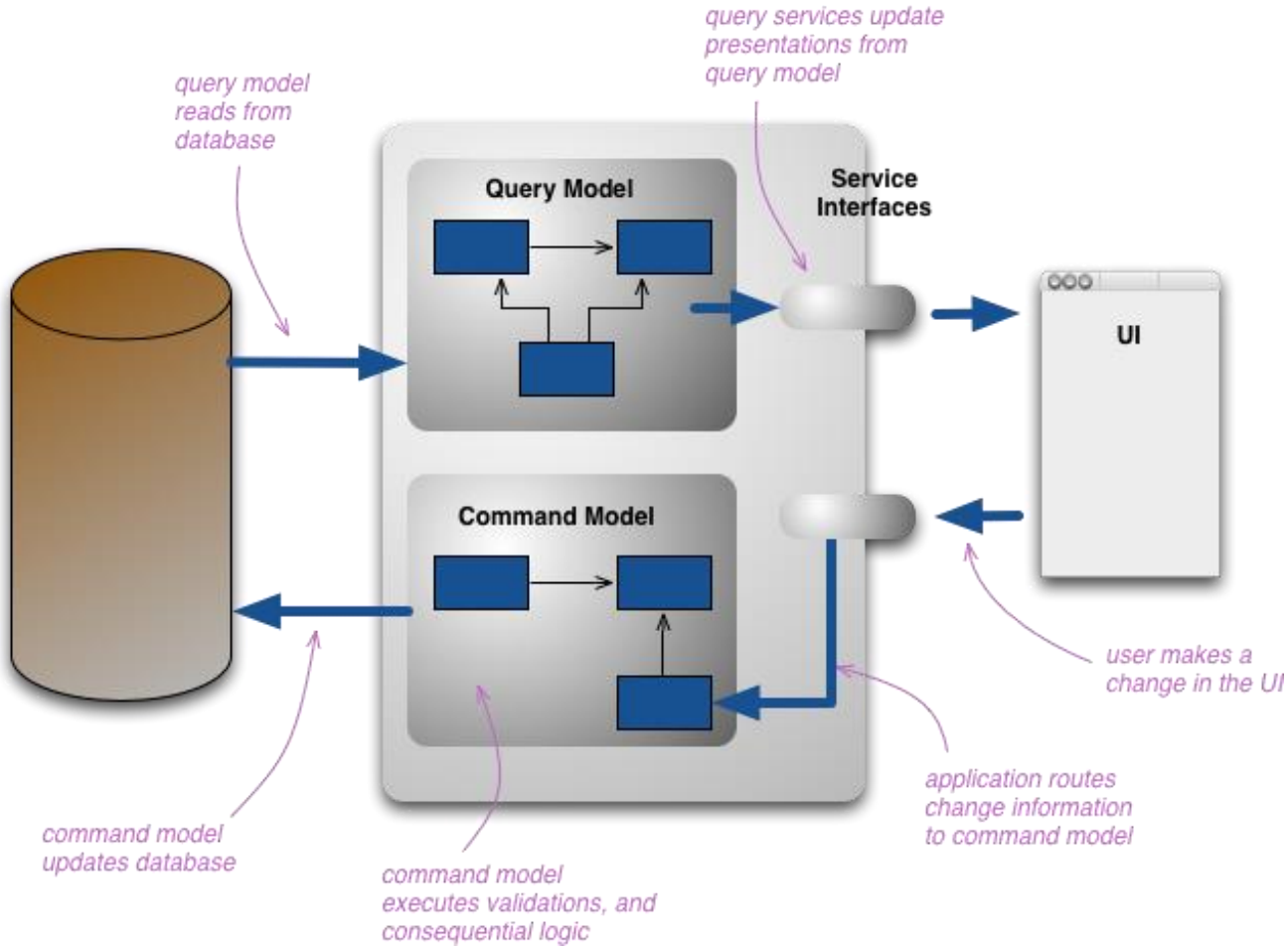
Pilvepõhiste rakenduste arhitektuurimustrid

- Cloud Dynamic DNS
- Use of Content Delivery Networks (CDN)
- CQRS

CQRS - Command and Query Responsibility Segregation

- Andmebaaside kirjutamis- ja lugemisloogika eraldamine erinevat tüüpi operatsioonideks
  - Päringud andmetele juurdepääsuks/lugemiseks
    - `Select customer, count(*) from transactions group by customer`
  - Andmete oleku muutmise toimingud
    - `Submit_new_transation(customer, amount)`
- Andmete kirjutamis- ja lugemistoimingute jagamine erinevate sõlmede vahel
  - Andmeid muudetakse ainult põhisõlmes – parandades järjepidevust
    - See sünkroniseerib muudetud andmed kõigi teiste sõlmedega
  - Andmeid saab lugeda mis tahes sõlmest – parandades lugemise skaleeritavust
- Andmebaaside eraldamine
  - Lugemiseks ja kirjutamiseks erinevate andmebaasitehnoloogiate kasutamine
  - Mõnikord kasutatakse väga kõrge kirjutamisjõudluse saavutamiseks

CQRS-mudel ühe andmebaasiga



https://martinfowler.com/bliki/CQRS.html

High Available (HA) Postgres deployment

Pgpool

- Manages connections between clients and cluster
- Load balancing
- Routes writes and reads

Repmgrd

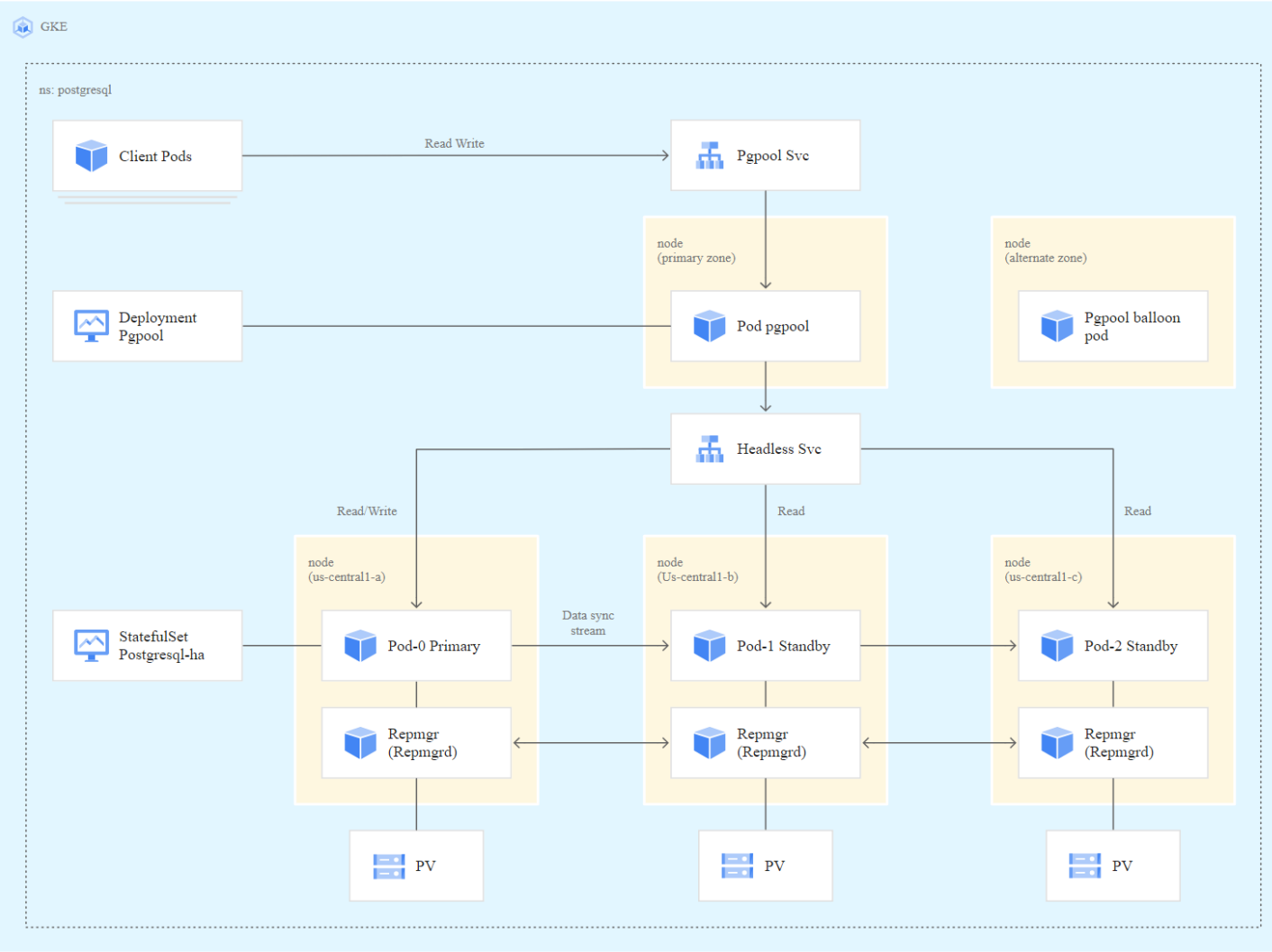
- Replikatsioonihaldur
- Replitseerib andmebaasi sisu

Pod-0 Primary

- Peamine SQL-i lõpp-punkt
- Kõik kirjutamispäringud suunatakse siia

Pod-X Standby

- Valmis üle võtma, kui esmane sõlm ebaõnnestub
- Lugemis koopia
- Lugemisi saab jaotada kõigi sõlmede vahel



Route 53

- Pilve DNS - *Domain Name Service*
- Võimaldab hallata:
  - Domeeninimede hankimist
  - Domeeni nimede lahendamist IP aadressiks
  - Päringute marsruutimist (nt. latentsuse, Geograafilise asukoha põhjal)
- Pilve DNS võimaldab kontrolli selle üle, kuidas kasutajate päringud ja liiklus jagatakse pilve regioonide ja asutuse enda süsteemide vahel

Sisu edastamise võrgustikud (CDN)

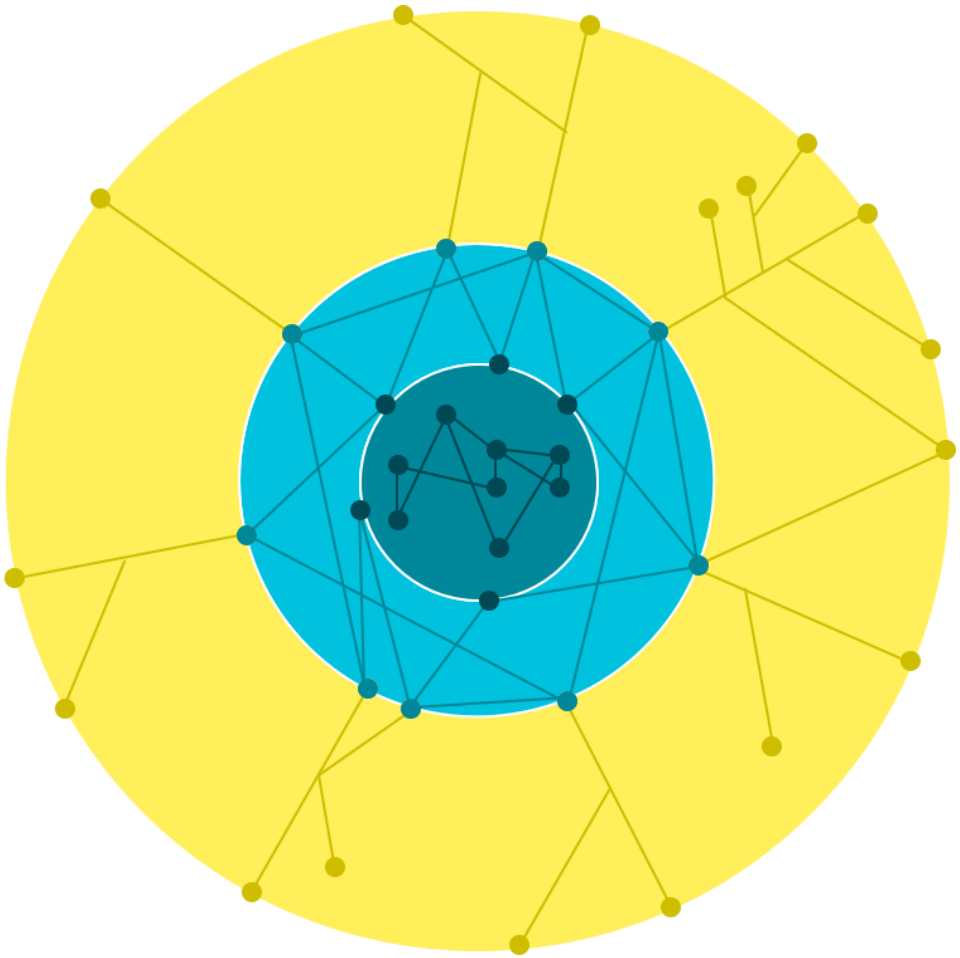
- *Content Delivery Networks*
- Ressursside kättesaadavaks tegemine eraldi "kihist"
  - Pildid, Videod, HTML, JS failid, jne.
- Kasutada ära ressursside ja lõppkasutajate geograafilise asukoha info
  - Ressursside kiire kohale toimetamine
  - Puhverdamine
  - Interneti liikluse vähendamine
- Teenus ei tagasta andmed, vaid viite andmeobjektile CDN's.
  - Lõppkasutaja (Klientrakendus, brauser) tõmbab andmed alla kõige lähemast asukohast

Google Äärevõrk

- Google globaalne infrastruktuur andmete ja pilve ressursside kohale toimetamiseks ja puhverdamiseks lõppkasutajatele
  - *Google Content Delivery Network*
- Kasutuses näiteks Youtube videote jaoks
- Puhverdamine (Caching)
- Interneti tasemel liikluse balansseerimine

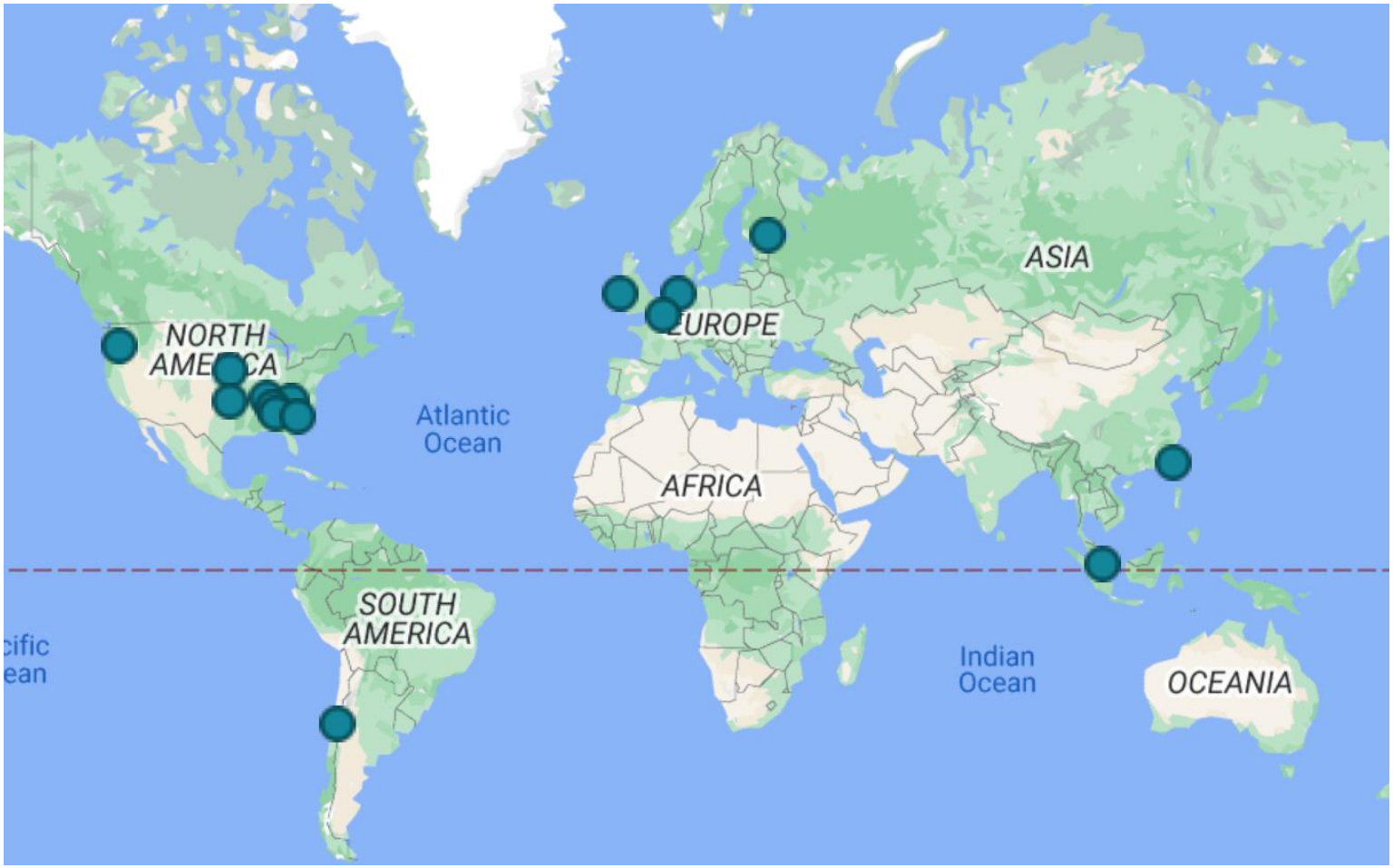
Google Äärevõrk

- Data Centers
  - Google andmekeskused
- Edge Points of Presence
  - Suuremate linnade juures
- Edge Nodes
  - Kohalike teenusepakkujate andmekeskustes



https://cloud.google.com/blog/products/networking/understanding-google-cloud-network-edge-points

Data Centers

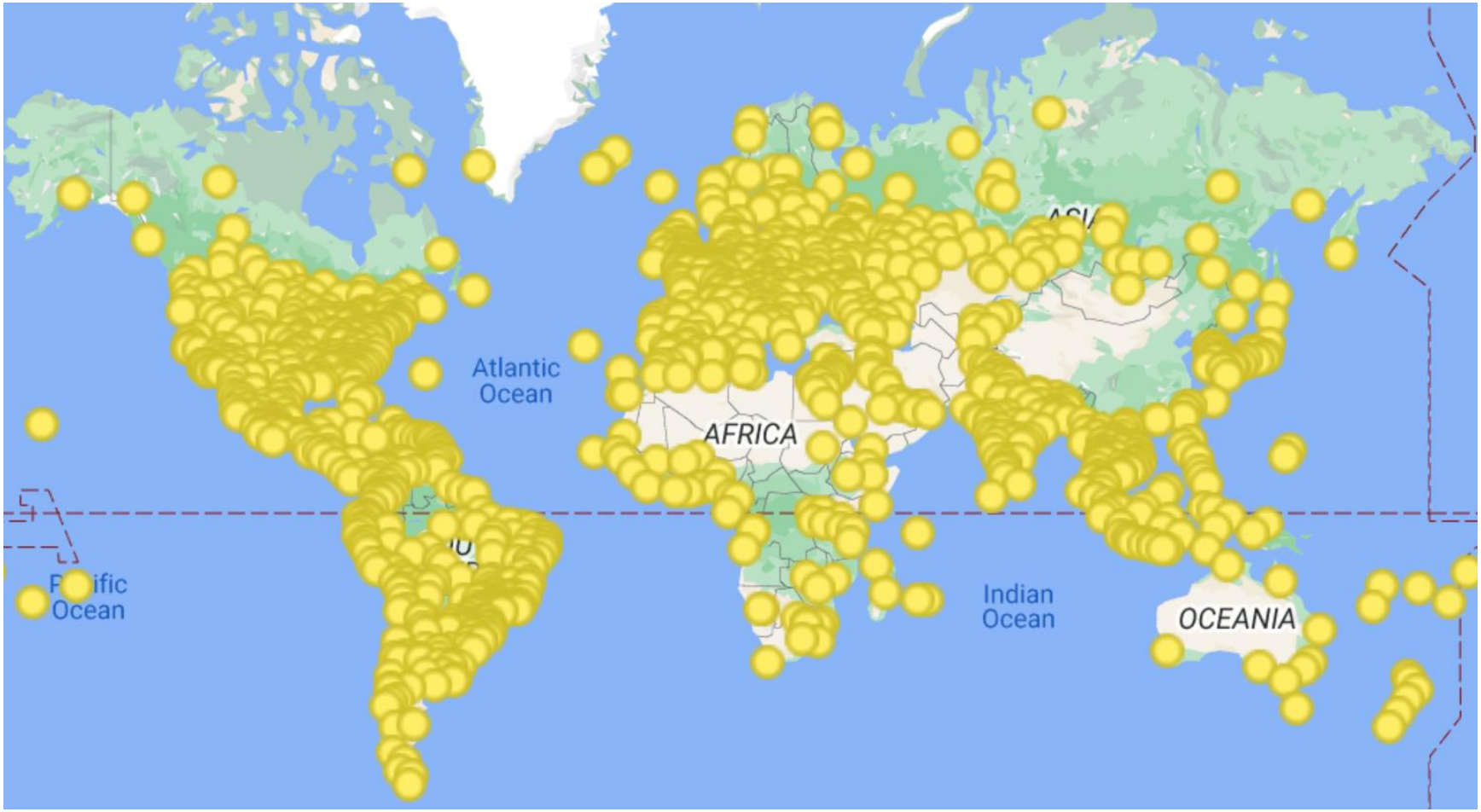


https://cloud.google.com/blog/products/networking/understanding-google-cloud-network-edge-points

Edge Points of Presence

https://cloud.google.com/blog/products/networking/understanding-google-cloud-network-edge-points

Edge Nodes



https://cloud.google.com/blog/products/networking/understanding-google-cloud-network-edge-points

Kokkuvõte

- Mikroteenuse ja pilvepõhised arhitektuurid lähevad järjest keerulisemaks
  - Hajuskomponentide arvu kasvuga suureneb keerukus
  - Vaja jagada haldamine erinevatesse kihtidesse

- Järgmine praktikum:
  - Arendame edasi oma mikroteenusepõhist rakendust pilvepõhiseks
- Järgmine loeng:
  - Jätkame hajus ja mikroteenuse arhitektuuride teemal
  - Vaatame näiteid pilveteenustest, mida siiani ei ole loengus olnud
  - Eksami korraldus ja konsultatsioon