LAB University of Applied Sciences

Google Cloud

Serverless-alustat ohjelmistokehityksessä

LAB-ammattikorkeakoulu
Tradenomi (AMK)
2023
Tapio Vaherkylä

Tiivistelmä

Tekijä(t)	Julkaisun laji	Valmistumisaika
Vaherkylä, Tapio	Opinnäytetyö, AMK	2023
	Sivumäärä	
	43	

Työn nimi

Google Cloud

Serverless-alustat ohjelmistokehityksessä

Tutkinto ja koulutusala

Tradenomi (AMK)

Tiivistelmä

Opinnäytetyössä tutkittiin ohjelmistojen julkaisuun soveltuvia Google Cloudin serverless-alustoja, selvitettiin alustojen välisiä eroja ja niiden soveltuvuutta ohjelmistojen julkaisuun. Googlen ohjeet ja kirjallisuus käsittelevät suurimmaksi osaksi yksittäisiä alustoja, ilman alustojen välistä vertailua ja koostetta alustoista.

Työ toteutettiin tutkimuksellisena opinnäytetyönä perehtymällä Google Cloudia ja serverless-alustoja käsittelevään kirjallisuuteen, tutkimuksiin, Googlen ohjeisiin ja internet lähteisiin.

Tutkimuksellisen työn tuloksena syntyi kirjallinen raportti, jota voidaan hyödyntää Google Cloudin serverless-alustan valinnassa ohjelmistokehityksen näkökulmasta. Työn lopputuloksena saadaan syvällisempää teoriapohjaista tietoa Google Cloudista, serverless-alustoista ja niiden käytettävyydestä ohjelmistojen julkaisussa.

Asiasanat

Google Cloud, serverless-alustat, IaaS, CaaS, PaaS, FaaS, BaaS

Abstract

Author(s)	Type of Publication	Published
Vaherkylä, Tapio	Thesis, UAS	2023
	Number of Pages	
	43	

Title of Publication

Google Cloud

Serverless platforms in software development

Degree and field of study

Bachelor of Business Administration (UAS)

Abstract

The purpose of this thesis was to investigate serverless platforms suitable for publishing software in Google Cloud, identify the differences between the platforms and their suitability for software publishing. Google's guides and literature mostly focus on individual platforms without comparing them or compiling information about them.

The work was carried out as a research thesis by studying literature, research, Google's instructions, and internet sources related to Google Cloud and serverless platforms.

As a result, a written report was produced that can be used to select a serverless platform on Google Cloud from the perspective of software development. The final outcome of the work is a deeper theoretical understanding of Google Cloud, serverless platforms, and their usability for software publishing.

Keywords

Google Cloud, serverless platforms, IaaS, CaaS, PaaS, FaaS, BaaS

Sisällys

SANASTO

AI	*Aritficial intelligence*, tekoäly
API	*Application interface*, ohjelmointi rajapinta
B2 WAN	B2 *wide area network*, julkinen ohjelmallisesti määritetty verkko
B4 WAN	B4 *wide area network*, yksityinen ohjelmallisesti määritetty verkko
BaaS	*Backend as a Service*, backend palveluna
BigTable	NoSQL-tietokanta
Borg	Klusterinhallintajärjestelmä
CaaS	*Container as a Service*, containerit palveluna
CLI	*Command line interface*, komentoliittymä
Cloud Firestore	NoSQL-tietokanta
Cloud Spanner	Hallinnoitu horisontaalisesti skaalautuva relaatiotietokanta
Cloud Storage	Ei -strukturoidun datan tallennusjärjestelmä
Compute Engine	Virtuaalipalvelin
Colossus	Klusteritason tiedostojärjestelmä
CPU	*Central Processing Unit*, prosessori
FaaS	*Functions as a Service*, funktiot palveluna
GFS	*Google File System*, edellinen versio Googlen tiedostojärjestelmästä
gRPC	gRPC *Remote Procedure Calls*, avoimen lähdekoodin RPC-kirjasto ja protokolla
HTTP	*Hypertext Transfer Protocol*, verkkoprotokolla
HTTP/2	*Hypertext Transfer Protocol*, uudistettu versio HTTP-verkkoprotokollasta
IoT	*Internet of Things*, esineiden internet

Google Cloud	Googlen pilvipalvelu
IaaS	*Infrastructure as Service*, infrastruktuuri palveluna
Kubernetes	Avoimen lähdekoodin containerien hallintajärjestelmä
Knative	Kubernetes-ympäristön laajennos
ML	*Machine Learning*, koneoppiminen
NFS	*Network File System*, verkkolevyjärjestelmä
Node	Virtuaalinen tai fyysinen kone kubernetes podien ajamiseen
On-premises	Omassa palvelinsalissa suoritettava ohjelmisto
Opensource	Avoin lähdekoodi
PaaS	*Platform as a Service*, alusta palveluna
POD	Joukko containereita, joita ajetaan Kubernetes Clusterissa
RAID	*Redundant array of Independent Disks*, tallennuksen virtualisointitekniikka
SaaS	*Software as a Service*, ohjelmiston toimittaminen palveluna
Sandbox	Eristetty ajonaikainen ympäristö
SDK	*Software Development Kit*, kokoelma ohjelmistokehityksen työkaluja
SDN-WAN	*Software defined network*, ohjelmallisesti määritetty verkko
TLS	SSL-salausprotokollan seuraaja

1 Johdanto

1.1 Tausta

Pilvipalveluita tarjoavien yritysten serverless-ympäristöt mahdollistavat sovellusten julkaisun ja ylläpidon kustannustehokkaasti vähentämällä infrastruktuurin kustannuksia, käyttökustannuksia ja henkilöstökustannuksia (Google 2021). Pilvipalveluyritykset tarjoavat myös monia muita pilvipalvelupohjaisia palveluita, joiden avulla yritykselle voidaan toteuttaa uusia liiketoimintaa hyödyttäviä ratkaisuja. Siirtymä pilvipalveluiden hyödyntämiseen yrityskäytössä tulee kiihtymään entisestään lähivuosina. Vuoteen 2025 mennessä 51 % yritysten IT -menoista kohdistuu pilvipalveluihin ja 65,9 % sovelluskehityksen menoista kohdistuu pilvipalvelu-teknologioihin. Hybridi- ja multicloud-pilvipalveluiden tarjonta madaltaa edelleen pilvipalveluiden käyttöönoton kynnystä. (Gartner 2022.)

Opinnäytetyön aihe valikoitui oman työn kautta tulleesta tarpeesta selvittää Google Cloudin palveluiden sisäistä rakennetta ja palveluiden käytettävyyttä pilvipalvelupohjaisen serverless-sovelluksen julkaisussa. Aiheen valintaan vaikutti myös henkilökohtainen kiinnostus pilvipalveluihin, pilvipalvelupohjaiseen sovelluskehitykseen ja pilvipalveluiden laajamittaiseen hyödyntämiseen sovelluskehityksessä.

Google Cloud on Googlen tarjoama pilvipalveluympäristö, joka koostuu yli 150 tuotteesta (Google 2022a). Googlen tarjoaman palvelukokonaisuuden avulla voidaan toteuttaa tietoturvallisia pilvipalvelusovelluksia ja palveluita tai siirtää olemassa olevia sovelluksia ja palveluita Googlen pilvipalveluun.

1.2 Työn tarkoitus, tavoite ja rajaukset

Opinnäytetyön tarkoituksena on luoda tekninen yleiskuva Google Cloudin perusrakenteista ja tutkia Google Cloudin serverless-palveluita teknisestä näkökulmasta sekä helpottaa serverless-palveluiden valintaprosessia. Työssä pyritään vastaamaan kysymyksiin eri Google Cloudin serverless-palveluiden soveltuvuudesta sovelluksen julkaisuun, mitä palveluiden käyttö edellyttää ja mitä pitää ottaa huomioon eri serverless-palveluissa. Vastaukset tutkimuskysymyksiin pyritään löytämään tutkimalla Google Cloud -pilvipalvelua sekä tutustumalla Googlen aineistoihin ja oppaisiin, kirjallisuuteen ja tutkimuksiin.

Työn teoriaosuuden ylätason tavoitteena on avata pilvipalvelun määritelmää, ominaispiirteitä, käyttöönottomalleja ja palvelumalleja. Työn tarkempi tavoite on tutkia Google Cloud -pilvipalvelun perusrakenteita ja ohjelmiston julkaisuun soveltuvia serverless-palveluita.

Opinnäytetyö rajataan ylimmällä tasolla pilvipalveluihin teoreettisen viitekehyksen muodostamiseksi. Työ rajataan tarkemmin käsittelemään sovelluksen julkaisuun soveltuvia Google Cloudin serverless-palveluita, niiden rakennetta, erityispiirteitä ja huomioitavia seikkoja. Tarvittaessa viitataan Google Cloudin verkko-, tallennus-, tietokanta-, identiteetti-, hallinta- ja monitorointipalveluihin sekä hinnoitteluun. Hybridi- ja multicloud-ratkaisut rajataan työn ulkopuolelle. Vertailua ei tehdä muiden yritysten tarjoamiin pilvipalveluihin tai avoimen lähdekoodin pilvipalveluihin. Google Cloudin serverless-palveluiden hyödyntämistä käsitellään teknologian näkökulmasta, kantaa ei oteta eettisiin näkökulmiin, ympäristövaikutuksiin, palveluiden kustannuksiin, ohjelmointikielien eroihin, hyviin ohjelmointikäytäntöihin ja tietoturvaan.

1.3 Tutkimusmenetelmä ja rakenne

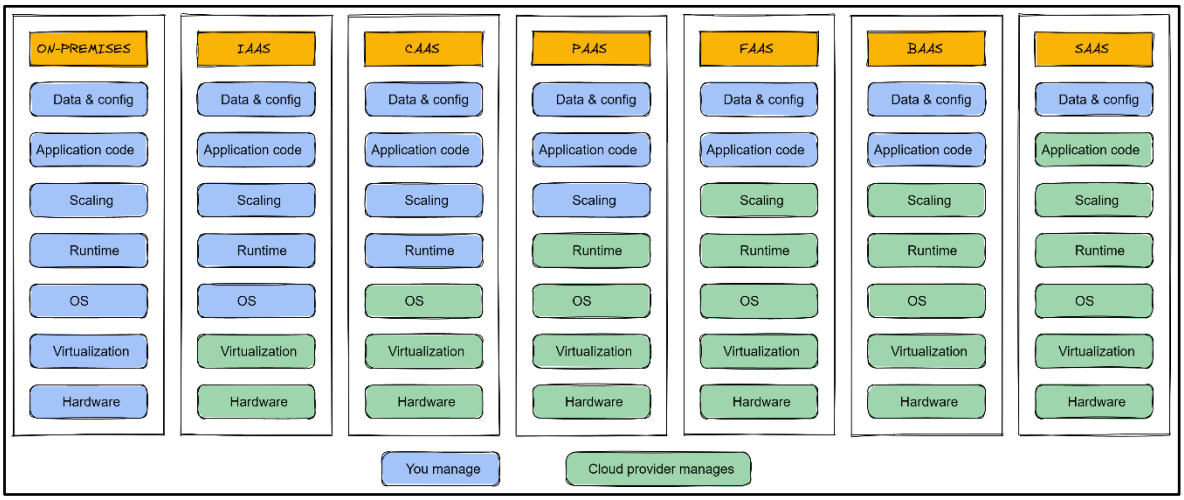
Työ toteutetaan tutkimuksellisena opinnäytetyönä, jonka tavoitteena on tuottaa serverless-alustojen valintaprosessia helpottavaa ohjeistusta. Tutkimuksen lopputuloksena muodostuu itse raportti. Tutkimuksellinen opinnäytetyö koostuu perehtymisestä ohjelmistojen julkaisuun soveltuviin Google Cloudin serverless-alustoihin. Kirjallisuuskatsauksessa tutkitaan yleisesti Google Cloudin perusrakenteita ja kohdennetaan tutkimus Google Cloudin serverless-alustoihin.

Työ koostuu kuudesta luvusta. Luvussa yksi kerrotaan työn taustasta, tarkoituksesta, tavoitteista ja rajauksista. Luvuissa 2–5 muodostetaan teoria- ja tietoperusta Google Cloudin perusrakenteista ja ohjelmistojen julkaisuun soveltuvista serverless-alustoista. Luvussa kuusi esitetään yhteenveto ja pohdinta. Yhteenvedossa tarkastellaan työn tuloksia, arvioidaan asetettujen tavoitteiden saavuttamista ja esitetään jatkotutkimuskohteita.

2 Pilvipalvelut

Pilvipalvelulle on kehitetty erilaisia määritelmiä. Pilvipalvelu on verkon välityksellä tarvittaessa kaikkialta saavutettavissa oleva, konfiguroitava ja helposti käyttöönotettava jaettu tietojenkäsittelyresurssi (Mell & Grance 2011, 6). Pilvipalvelu voidaan myös määritellä automaattisesti skaalautuvaksi, helposti ja nopeasti saatavilla olevaksi tietojenkäsittelyresurssiksi, joka on käytettävissä tarvittaessa. Kustannukset perustuvat resurssien käyttöön. (Harkut 2018.)

Pilvipalveluilla on viisi ominaispiirrettä: itsepalveluperiaate, pääsy palveluihin eri päätelaitteilla, jaettujen resurssien käyttö, joustava skaalautuvuus ja käytön tarkka mittaaminen (Mell & Grance 2011, 6). Pilvipalveluiden käyttöönottomalleina voidaan tunnistaa yksityinen-, yhteisöllinen-, julkinen- ja hybridipilvi. Pilvipalvelut ovat yleisimmin jaettu Software as a Service (SaaS)-, Platform as a Service (PaaS)- ja Infrastructure as a Service (IaaS)-palvelumalleihin. (Mell & Grance 2011, 6–7.) Kuvassa 1 tuodaan esille serverless-palvelumallien sekä on-premises-palvelinten ja SaaS-palvelumallin hallinnoinnin jakautumisen erot käyttäjän ja pilvipalvelutarjoajan välillä.



Kuva 1. Palvelumallien hallintatasot (mukailtu Vergadia 2022)

On-premises

On-premises-mallissa käyttäjä hallinnoi kaikkea datan ja laitteiston välillä. Palvelin voi sijaita fyysisesti omassa palvelinsalissa, johon käyttäjällä on pääsy. Käyttäjä hallinnoi dataa, sovelluksen koodia, skaalausta, ajonaikaista ympäristöä, verkkoa, palvelimia ja

käyttöjärjestelmää. Tämä on perinteinen tapa järjestää palvelinten sijoittelu ja hallinnointi. (Vergadia 2022.)

IaaS

Infrastructure as a Service-palvelumallissa palveluntarjoaja tarjoaa perustavaa laatua olevia tietojenkäsittelyresursseja palveluna, joiden avulla käyttäjä voi julkaista ja ajaa ohjelmistoja halutun käyttöjärjestelmän päällä. IaaS-palvelumallissa käyttäjä voi hallita ja konfiguroida dataa, sovelluksen koodia, skaalausta, ajonaikaista ympäristöä ja käyttöjärjestelmää. IaaS-palvelun käyttäjä ei hallinnoi varsinaista taustalla toimivaa pilvipalvelun infrastruktuuria, kuten virtualisointia ja laitteistoa. Palveluntarjoajan vastuulla on taata resurssien turvallisuus, toimivuus ja pilvipalvelun infrastruktuurin ylläpito. (Mell & Grance 2011, 7; Treat 2019.)

CaaS

Container as a Service -palvelumallissa palvelun käyttäjä voi julkaista sovelluksia containerissa halutussa ajonaikaisessa ympäristössä. Käyttäjä voi hallinnoida ja konfiguroida dataa, sovelluksen koodia, skaalausta, ajonaikaista ympäristöä. Palveluntarjoaja hallinnoi käyttöjärjestelmää, virtualisointia ja laitteistoa. (Treat 2019.)

PaaS

Platform as a Service -palvelumallissa käyttäjä julkaisee sovelluksia containerissa hyvin pitkälle hallinnoidussa ympäristössä etukäteen määritellyssä ajonaikaisessa ympäristössä. Käyttäjä hallinnoi dataa, sovelluksen koodia ja skaalausta. Palveluntarjoaja hallinnoi ajonaikaista ympäristöä, käyttöjärjestelmää, virtualisointia ja laitteistoa. (Mell & Grance 2011, 6–7; Treat 2019.)

FaaS

Function as a Service -palvelumalli on erittäin pitkälle hallinnoitu palveluntarjoajan toimesta. Käyttäjä voi julkaista sovelluksia pilvipalvelufunktioina. Käyttäjä hallinnoi dataa ja sovelluksen koodia. Palveluntarjoaja vastaa skaalauksesta, ajonaikaisesta ympäristöstä, käyttöjärjestelmästä, virtualisoinnista ja laitteistosta. (Treat 2019.)

BaaS

Back End as a Service -palvelumalli on erittäin pitkälle hallinnoitu. BaaS-palvelumalli mahdollistaa ohjelmiston erittäin nopean julkaisun. Käyttäjä hallinnoi dataa, konfiguraatiota ja sovelluksen koodia, pilvipalveluntarjoajan hallinnoidessa skaalauksen, ajonaikaisen ympäristön, käyttöjärjestelmän, virtualisoinnin ja laitteiston. Käyttäjä voi keskittyä varsinaisen sovelluksen kehittämiseen. (Treat 2019.)

SaaS

SaaS-palvelumallissa käyttäjät hyödyntävät palveluntarjoajan pilvipalvelun päällä ajettavia sovelluksia. Palveluntarjoaja vastaa pilvipalvelun infrastruktuurista, verkosta, palvelimista, käyttöjärjestelmästä, tallennustilasta ja sovelluksen ominaisuuksista. Käyttäjä hallinnoi käytössään olevaa dataa ja voi konfiguroida käyttämänsä sovelluksen asetuksia. (Mell & Grance 2011, 6.)

3 Google Cloud

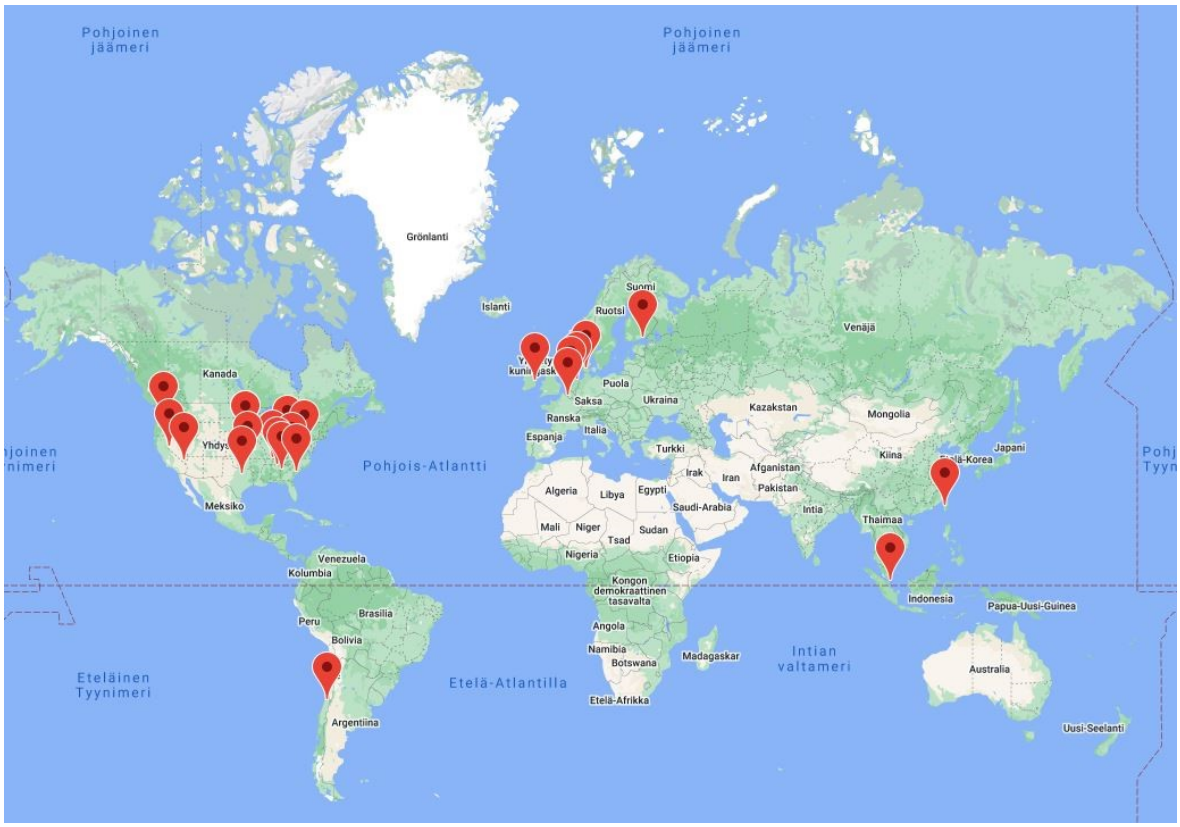
3.1 Yleiskuvaus

Google Cloud on kokoelma Googlen pilvipalvelupohjaisia tietojenkäsittelyn tuotteita ja palveluja, ja ne ovat käytettävissä yli 200 maassa ja alueella. Pilvipalvelun komponentit koostuvat fyysisistä laitteista ja virtuaalisista resursseista, joista muodostettuja palveluja hyödyntävät Googlen omat palvelut ja Google Cloudin käyttäjät verkon yli hajautettujen palvelinkeskusten kautta. (Google 2022b.) Pilvipalveluilta vaaditaan skaalautuvuutta, laajennettavuutta, hallittavuutta ja sopeutuvuutta (f5 2010).

Google Cloudissa näihin vaatimuksiin on vastattu virtualisoimalla palvelinkeskusten palvelimia, abstrahoimalla tallennuksen fyysisen rautatason monimutkaisuutta, virtualisoimalla palvelinkeskusten sisäverkko, palvelinkeskukset toisiinsa yhdistävä verkko sekä ulospäin tarjottavaa verkkoa (Edge Network, Espresso). Näiden perustavaa laatua olevien vaatimusten toteuttamisen myötä saavutetaan erittäin nopea datan tiedonsiirto, replikointi, tallennus ja tarjonta loppukäyttäjille.

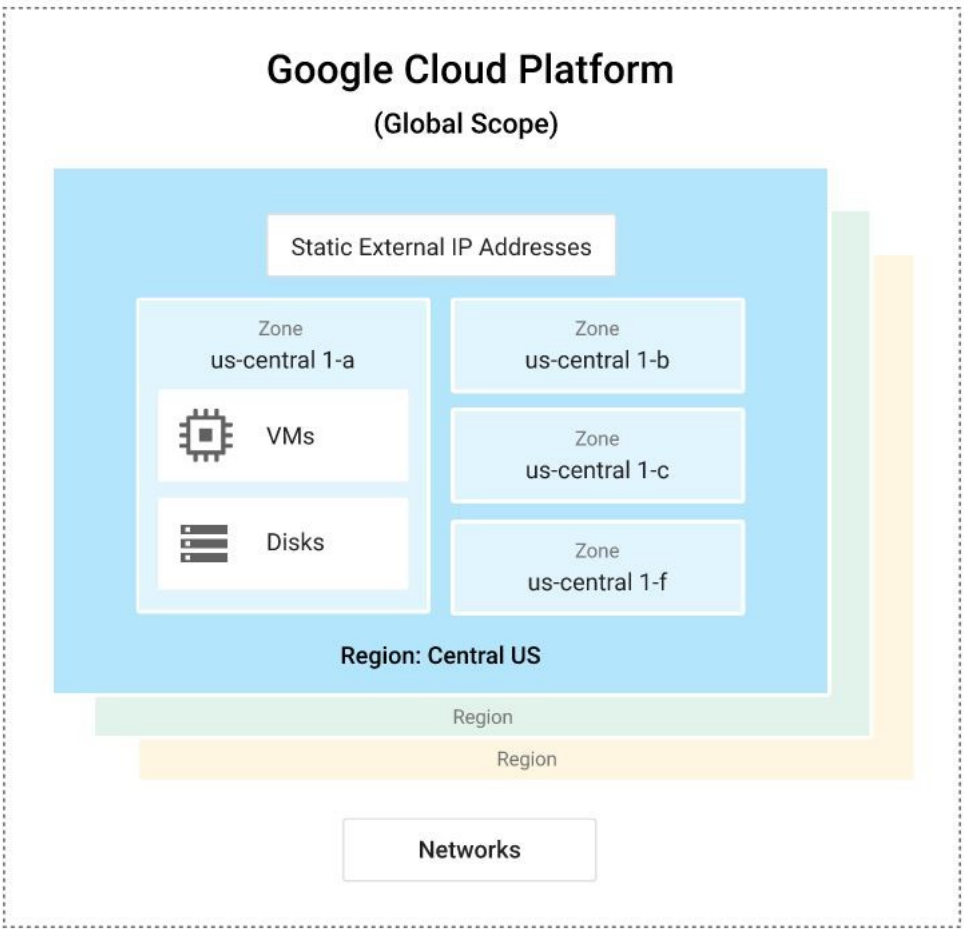
3.2 Palvelinkeskukset

Google Cloudissa on tällä hetkellä 22 palvelinkeskusta hajautettuna maantieteellisesti eri alueille (Kuva 2). Palvelinkeskukset on yhdistetty toisiinsa globaalin fyysisen verkkoinfrastruktuurin avulla.



Kuva 2. Google Cloud -palvelinkeskukset (Google 2022b)

Googlen palvelinkeskukset sijaitsevat Aasiassa, Australiassa, Euroopassa, Pohjois-Amerikassa ja Etelä-Amerikassa. Google Cloud koostuu loogisista itsenäisistä maantieteellisistä alueista (*regions*). Alueet koostuvat toisistaan eristetyistä loogisista vyöhykkeistä (*zones*). Loogisen abstraktion avulla Googlen palveluita voidaan jakaa virtuaalisesti (Kuva 3). (Google 2022c.)

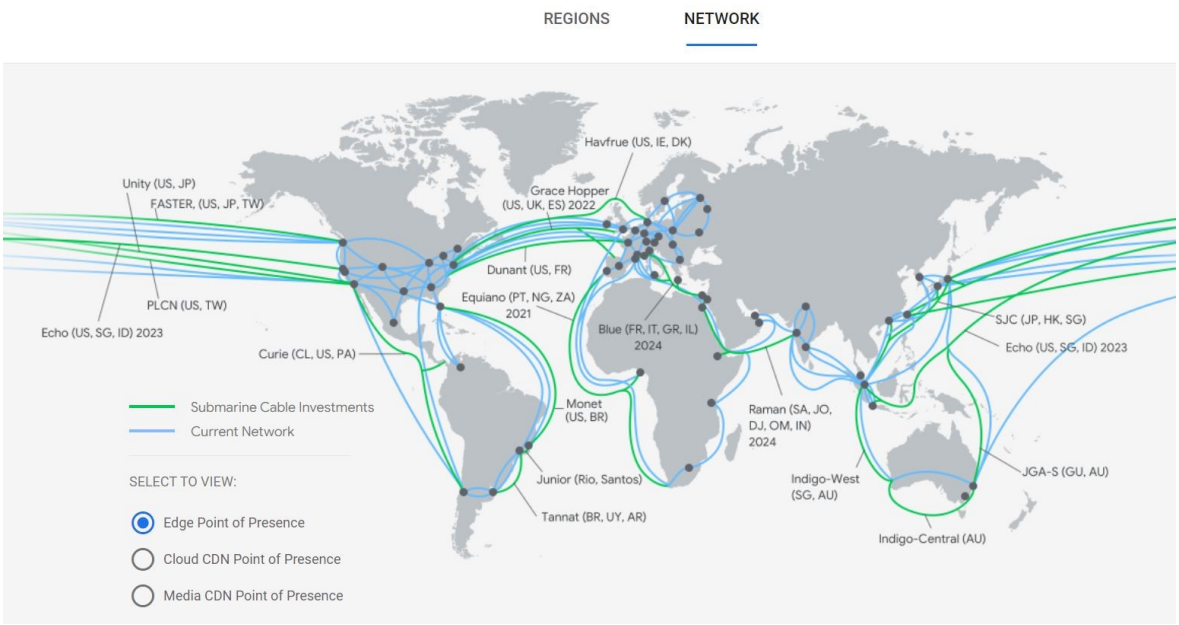


Kuva 3. Google Cloudin verkon, alueiden ja vyöhykkeiden infrastruktuuri (Google 2022d)

Datakeskusten, alueiden ja vyöhykkeiden hajautuksella voidaan taata pilvipalvelujen saavutettavuus häiriötilanteiden yhteydessä ja vähennetään palveluiden viivettä loppukäyttäjille tarjoamalla palvelua fyysisesti loppukäyttäjän läheltä. Pilvipalvelua hyödyntävän kehittäjän on otettava huomioon hajautuksen vaikutukset eri resurssien käytössä ja yhdistämisessä toisiinsa. (Google 2022d.)

3.3 Verkkoinfrastruktuuri

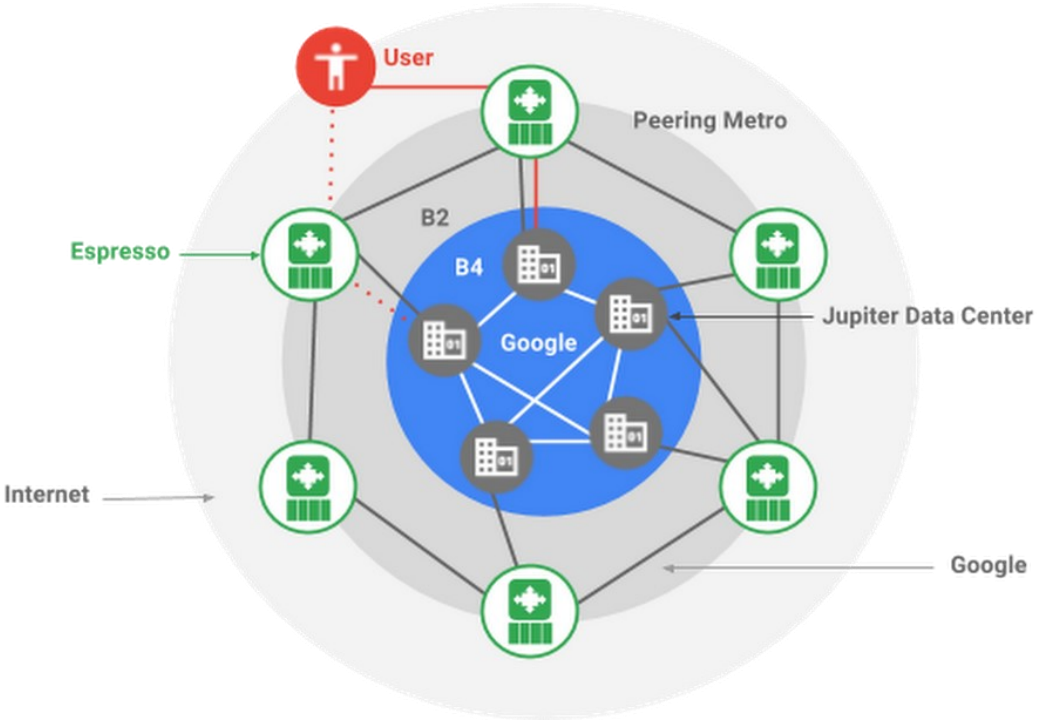
Google Cloudin verkkoinfrastruktuuri koostuu fyysisestä verkosta, palvelinkeskusten sisäisestä verkosta, yksityisestä SDN-WAN-verkosta sekä julkisesta SDN-WAN-verkosta. Valokuitukaapeleita hyödyntäen on luotu fyysinen palvelinkeskukset toisiinsa yhdistävä verkko (Kuva 4). (Vahdat 2017; Google. 2022q.)



Kuva 4. Google-valokuituverkko (Google 2022q)

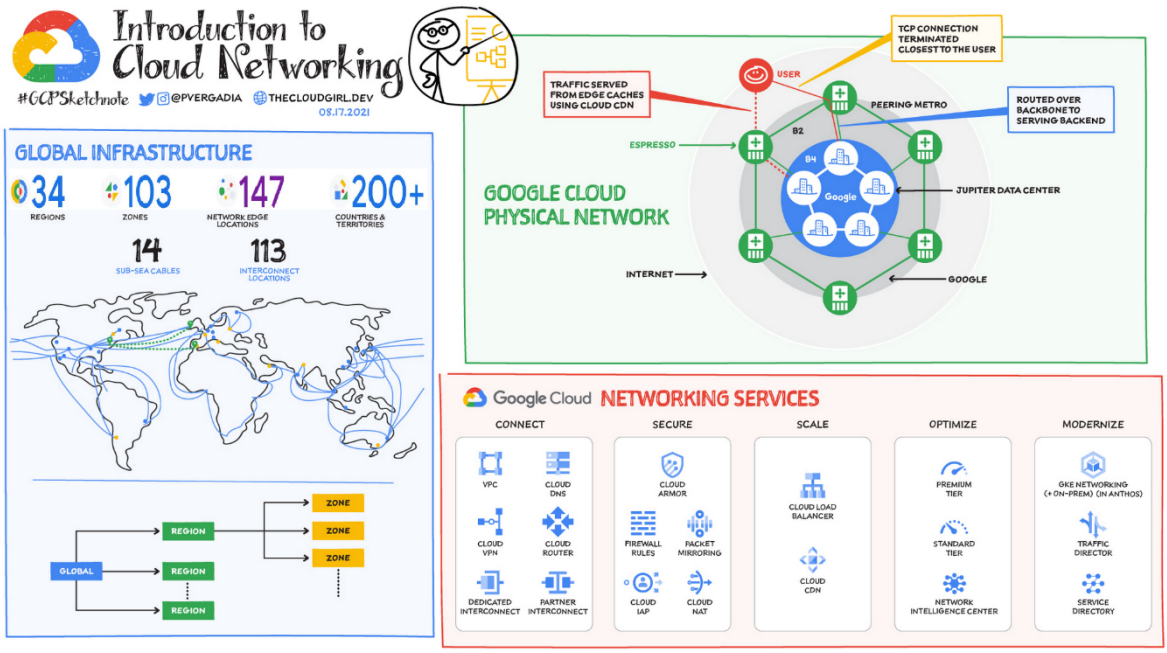
Palvelinkeskuksen sisäinen Jupiter fabric -verkko on OpenFlow-protokollaan perustuva ohjelmallisesti määritelty verkko (Singh ym. 2016, 3). Jupiter fabric -verkko hyödyntää monivaiheista Clos-piirikytkentäverkkoa. Clos-topologian avulla halvemmista ja fyysisesti pienemmistä kytkimistä muodostetaan suurempi looginen kytkin. Loogisista kytkimistä muodostetaan Jupiter fabric -verkko. Jupiter fabric -verkko yhdistää yhden palvelinkeskuksen kaikki palvelimet toisiinsa 1.3 Petabit/s nopeudella. (Vahdat 2022.)

Kuva 5 havainnollistaa Googlen palvelinkeskusten B4- ja B2-verkon sekä internetin verkkotopologian pääpiirteet. Ohjelmallisesti määritetyllä yksityisellä B4 WAN -verkolla yhdistetään kaikkien palvelinkeskusten palvelimet ja palvelut toisiinsa. (Vahdat 2022.) Myös B4-verkko perustuu OpenFlow-protokollaan (Singh ym. 2016, 3). B4-verkon kaistanleveys mahdollistaa datan replikoinnin reaaliajassa kahden eri datakeskuksen välillä (Vahdat 2017). Julkisella ohjelmallisesti määritetyllä B2 WAN -verkolla tarjotaan ulkopuolisille käyttäjille pääsy Googlen verkkoon ja palveluihin peering-menetelmällä (Mukherjee ym. 2020). B2-verkko hyödyntää osittain ohjelmallisesti määritettyä Espresso-verkkoa, joka reitittää B2-verkon liikennettä internetin muihin verkkoihin peering-menetelmällä (Vahdat 2017).



Kuva 5. Google Cloudin verkon rakenne (Vahdat 2017)

Google Cloudin verkkopalvelujen avulla kehittäjille tarjotaan verkkoon liittymisen palvelut, verkon tietoturvapalvelut, verkon skaalauspalvelut, optimointipalvelut ja modernisointipalvelut. Kuvassa 6 on koosteena Google Cloudin verkko, alueet, vyöhykkeet ja verkon palvelut. (Vergadia 2021a.)



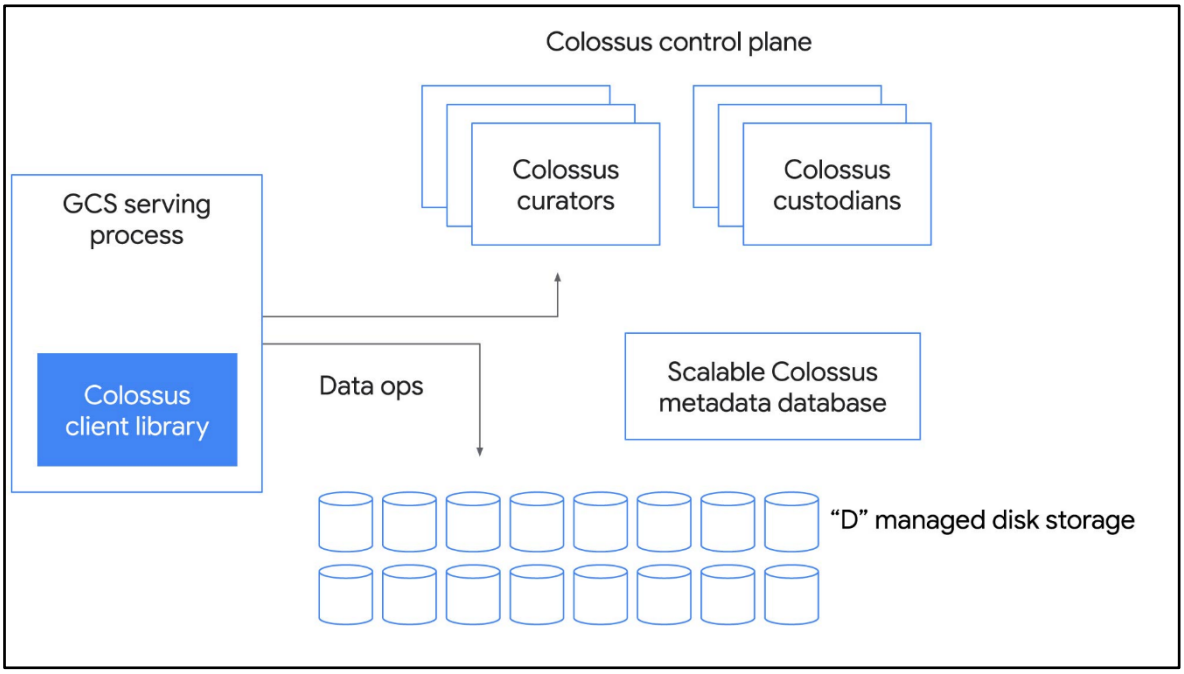
Kuva 6. Google Cloudin verkot, alueet, vyöhykkeet ja palvelut (mukailtu Vergadia 2021a)

3.4 Skaalautuva tallennusjärjestelmä

Google Cloud Storage -tallennusjärjestelmä (GCS) on runkona kaikille Googlen tallennuspalveluille. Pilvipalvelun tallennuspalveluiden tulee skaalautua automaattisesti tarpeen mukaan. Google Cloudin tallennusjärjestelmä koostuu klusteritason tiedostojärjestelmästä Colossuksesta, globaalista skaalautuvasta Spanner-relaatiotietokannasta ja skaalautuvasta työ- ja resurssiajastimesta Borgista. (Hildebrand & Sereneyi 2021.)

3.4.1 Colossus klusteritason tiedostojärjestelmä

Colossus pohjautuu Google File System -tiedostojärjestelmään (GFS). Colossuksen avulla hallinnoidaan, varastoidaan ja tarjotaan pääsy dataan kuormasta ja datamäärän kasvusta riippumatta. Colossus on hajautettu ja replikoitu tiedostojärjestelmä. Google Cloud Storagen palveluprosesseja ja sovelluksia ohjataan Colossuksen client-kirjastolla. Client-kirjasto tallentaa datan ”D”-tiedostopalvelimille, verkkoon liitetyille levyille ja samanaikaisesti kirjasto käyttää suoraan Colossuksen metadata-palvelun kontrollitason edunvalvojia, Curatoreita, tiedosto-operaatioissa tallentaen tiedostojen metadataa BigTable NoSQL -tietokantaan. Kontrollitason Custodian taustatallennushallinnoijat huolehtivat datan saatavuudesta, oikeellisuudesta, datan tasauksesta levyillä ja tarvittaessa RAIDin uudelleen rakentamisesta (Kuva 7). (Hildebrand & Sereneyi 2021.)



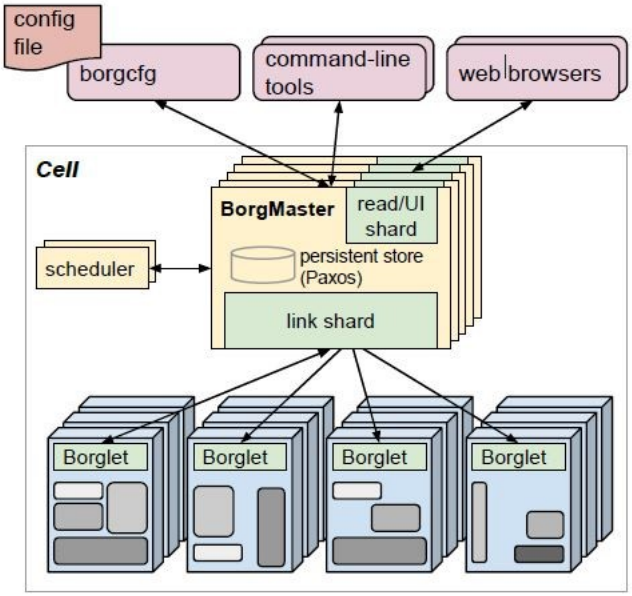
Kuva 7. Colossus-tiedostojärjestelmän kontrollitaso (Hildebrand & Sereneyi 2021)

3.4.2 Google Cloud Spanner

Spanner on Googlen täysin hallinnoitu horisontaalisesti skaalautuva ja globaalisti yhdenmukainen relaatiotietokanta. Google Cloud Storage tallentaa Spanneriin kaiken tallennettavan datan oikeuksiin ja sijaintiin liittyvän metadatan. Spanner käyttää itse sisäisesti hajautettua ja replikoitua Colossus-tiedostojärjestelmää. (Vergadia 2021b.)

3.4.3 Borg-klusterien hallintajärjestelmä

Fyysisten ja loogisten virtuaalipalvelimien ja niistä muodostettavien klusterien hallintaan käytetään Borg-klusterinhallintajärjestelmää (Kuva 8). Borg pystyy hallinnoimaan useita klustereita, joihin kuuluu kymmeniä tuhansia palvelimia. Palvelimilla ajetaan tuhansia sovelluksia ja satojatuhansia töitä. Borgin avulla hallinnoidaan tietojenkäsittelyoperaatioiden lisäksi tallennusoperaatioita. Borg jakaa suoritettavat sovellukset ja työt, ajastaa niiden suorituksen, käynnistää ja tarvittaessa uudelleen käynnistää ajettavia sovelluksia ja töitä. Suoritettava työ koostuu yhdestä tai useammasta tehtävästä. Työ suoritetaan Borg-solussa, joka koostuu useammasta palvelimesta. (Verma ym. 2015; Hildebrand & Sereneyi 2021.)

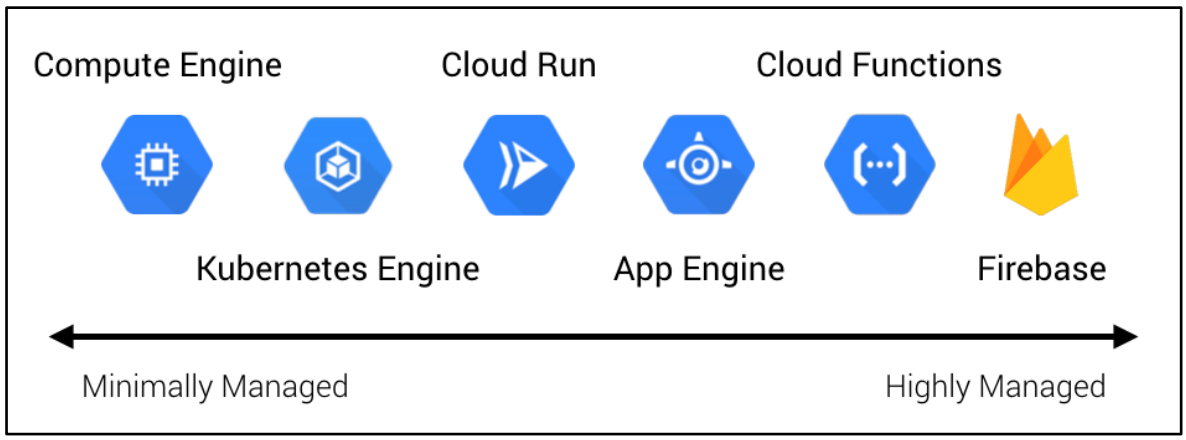


Kuva 8. Borg-klusterinhallintajärjestelmän rakenne (Verma ym. 2015)

4 Google Cloudin serverless-palvelut sovellusalustoina

4.1 Serverless-alustat ja hallinnan abstraktiotaso

Google Cloudin serverless-alustoja ovat Google App Engine (PaaS), Google Cloud Functions (FaaS), Google Cloud Run (CaaS), Google Firebase (BaaS) ja Google Kubernetes Engine, joka sijoittuu Compute Enginen (IaaS) ja Cloud Runin (CaaS) välille. Osa Google Cloud Platformin palveluista on erittäin pitkälle hallinnoituja, osan ollessa vähemmän hallinnoituja (Kuva 9). (Treat 2019.)



Kuva 9. Google Cloud Serverless -palveluiden hallinnointitaso (Treat 2019)

Hallinnointiin käytettävä aika tulee ottaa huomioon varsinaiseen ohjelmiston kehitystyöhön käytettävän ajan lisäksi. Pilvipalvelun tarjoajan toimesta erittäin pitkälle hallinnoiduilla alustoilla varsinaisen ohjelmiston kehittämiseen jää enemmän aikaa. (Treat 2019.)

4.2 Google Kubernetes Engine

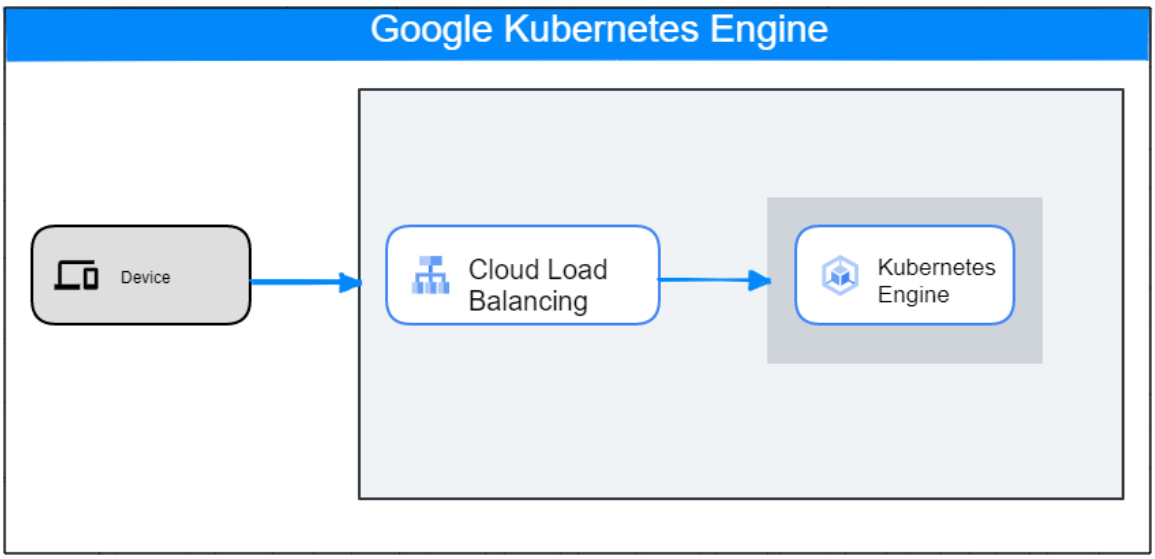
Kubernetes Engine sijoittuu IaaS:n ja CaaS:n välimaastoon ja se pohjautuu avoimen lähdekoodin Kubernetes-säiliöiden hallintaympäristöön. Kubernetes Enginen avulla voidaan julkaista, hallinnoida ja skaalata container-pohjaisia sovelluksia. (Treat 2019; Google 2022g.)

4.2.1 Kubernetes-ympäristö

Google Kubernetes Engine voidaan määritellä serverless-versiona Autopilot-toiminnon avulla. Autopilot-toiminto poistaa alla olevan infrastruktuurin tarkan määrittelyn. Autopilot-toiminnon avulla voidaan käyttää testattuja ja kovennettuja Kubernetes Engine -ympäristöjä

ja estää käyttäjän tekemät virheelliset määritykset. Kubernetes Enginessä on mahdollista konfiguroida tarvittava kokonaisuus manuaalisesti Standard-ympäristössä. Standard-ympäristöä käytettäessä menetetään Autopilotin serverless-määrityksen hyödyt. (Wheatley 2021, Google 2022f.)

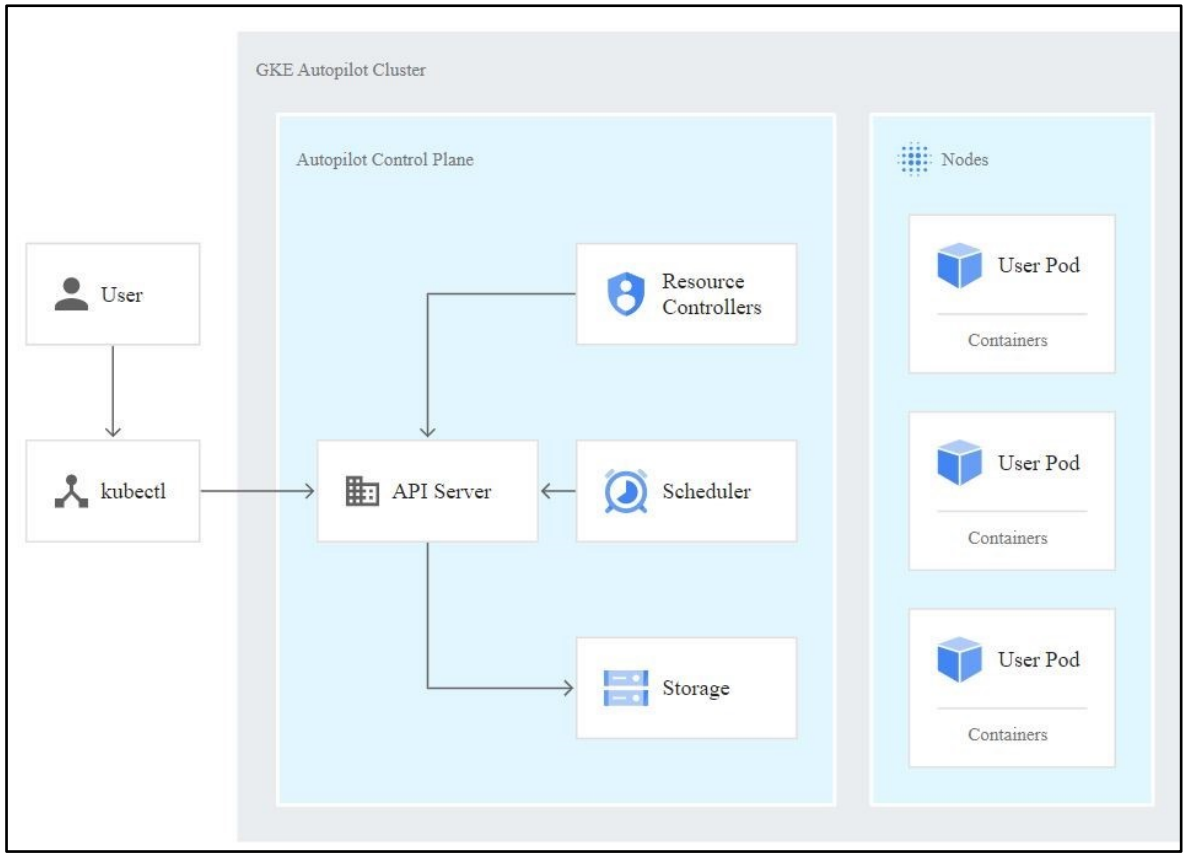
GKE hyödyntää Google Cloudin kuormantasausta, joka hoitaa sovelluksen skaalauksen lisäämällä horisontaalisesti klusteriin Compute Engine -virtuaalipalvelimia hallinnoituun instanssiryhmään tai poistamalla virtuaalipalvelimia ryhmästä kuorman mukaan (Kuva 10). Kuormantasaus myös reitittää liikenteen lähimmälle virtuaalipalvelimelle ja tarvittaessa poistaa korruptoituneita virtuaalipalvelininstansseja. GKE-klusterilla voi hyödyntää Virtual Private Cloud -verkkoa ja pysyviä massamuisteja sekä muita Google Cloudin palveluita. (Treat 2019; Google 2022g.)



Kuva 10. Kubernetes Enginen kuormantasaus (mukailtu Google 2022h)

4.2.2 Kubernetes Enginen rakenne ja integraatiot

GKE hallinnoi Autopilot-klusteria, joka koostuu yhdestä tai useammasta Control Planesta ja klusteriin kuuluvasta Node-poolista. Klusterin Control Planeja ajetaan useammalla palvelimella ja klusterilla on yleensä useita Nodeja Noodi-poolissa. Näin taataan korkean käytettävyyden klusterikokonaisuus ja suoritettavan sovelluksen vikasietoisuus. Control Plane koostuu klusterin API-serveristä, Cloud Controller Managerista, Resource Controllerista, Schedulerista ja Storagesta (Kuva 11). (Google 2022h.)

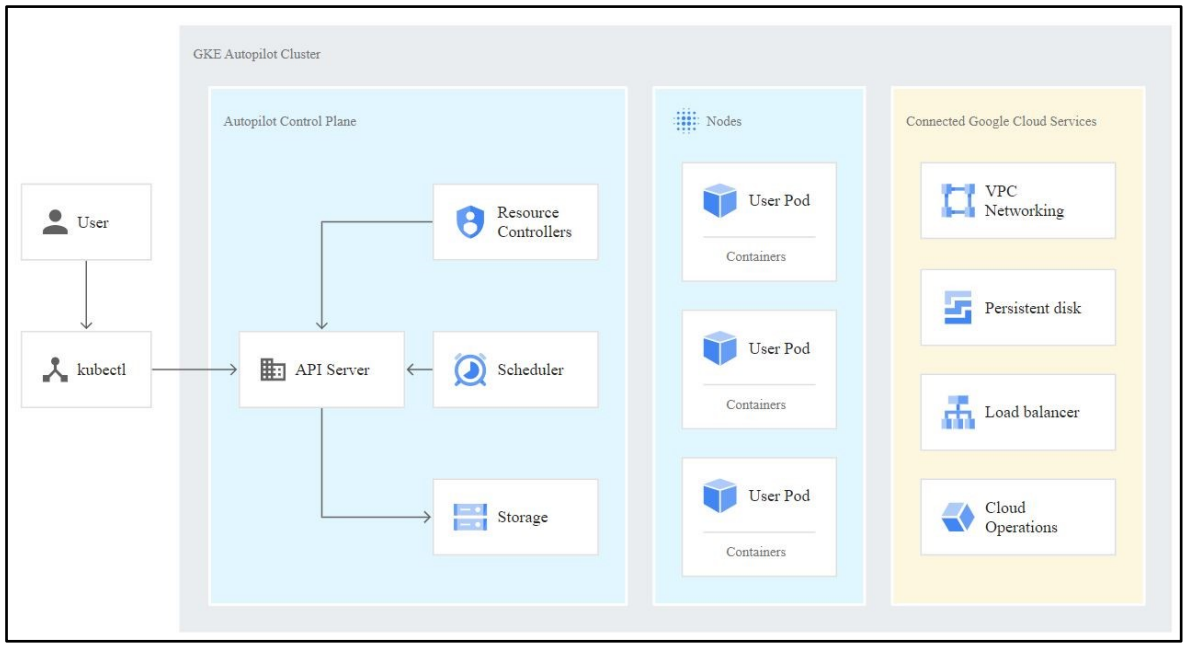


Kuva 11. Kubernetes Enginen rakenne (Google 2022h)

Control Plane päivittää tarvittaessa automaattisesti käyttämänsä Kubernetes- ja Node versiot ja hoitaa järjestelmän skaalauksen GKE:n autoscaler-toiminnolla Node-tasolla lisäämällä tai poistamalla Noodeja. GKE käyttää kuormantasaukseen myös horisontaalista ja vertikaalista skaalausta. Horisontaalisesta Podien skaalausta käytetään, kun kuorma ylittää CPU:lle tai muistin käytölle asetetut raja-arvot. (Google 2022x.) Vertikaalista Podien skaalausta käytetään määrittämään suositeltuja raja-arvoja CPU:n ja muistin käytölle. Control Plane vastaa työn elinkaaren hallinnasta, työn suorituksesta, työn suorituspaikasta ja milloin työ suoritetaan. API-serveri hoitaa klusterin sisäisten komponenttien välisen liikennöinnin, kubectl-komentorivikehotteen ja Google Cloud Consolen liikenteen klusterille. Resurssi -kontrollerin kautta monitoroidaan kaikkia komponentteja ja varmistetaan komponenttien haluttu tila. Node-poolissa on yksi tai useampi Node-virtuaalipalvelin. Node koostuu yleensä vähintään yhdestä Kubernetes Podista, jonka sisällä ajetaan yhtä tai useampaa säiliötä. Node odottaa API-serveriltä suoritettavaa työtä, suorittaa työn ja raportoi työn statuksen Control Planelle. Scheduler seuraa Storagea uusien töiden varalta ja jakaa klusterin kuormaa Node-poolin Nodeille ja Pod-säiliöille toteutettavaksi. (Sangapu ym. 2021; Google

2022h.) Useampia containereita voidaan käyttää yhden Podin sisällä, jos Podin säiliöt tarvitsevat jaettuja resursseja. Podin sisällä olevat containerit jakavat verkon, IP-osoitteen, IP-osoiteavaruuden, kokoelman portteja ja tallennustilan. Näin voidaan julkaista ilman sovelluksen konfiguraatiomuutoksia useita instansseja samasta sovelluksesta, eri instansseja eri sovelluksista tai kokonaan eri Noodeja. (Sullivan 2019.)

Google Kubernetes Engine voi hyödyntää Googlen muita palveluja Google Cloud API:n avulla määrittelemällä Workload Identityn. Kaikki Podit, joita ajetaan Kubernetes-palvelutunnuksella, autentikoituvat automaattisesti Google-palvelutunnukseen ja -palveluun. Cloud SQL:n hyödyntäminen on ainoa poikkeus. Cloud SQL vaatii Cloud SQL Proxyn määrittämisen palvelun käyttämiseksi (Kuva 12). (Google 2022h.)



Kuva 12. Kubernetes Enginen ympäristö ja integraatiot (Google 2022h)

4.3 Google Cloud Run (CaaS)

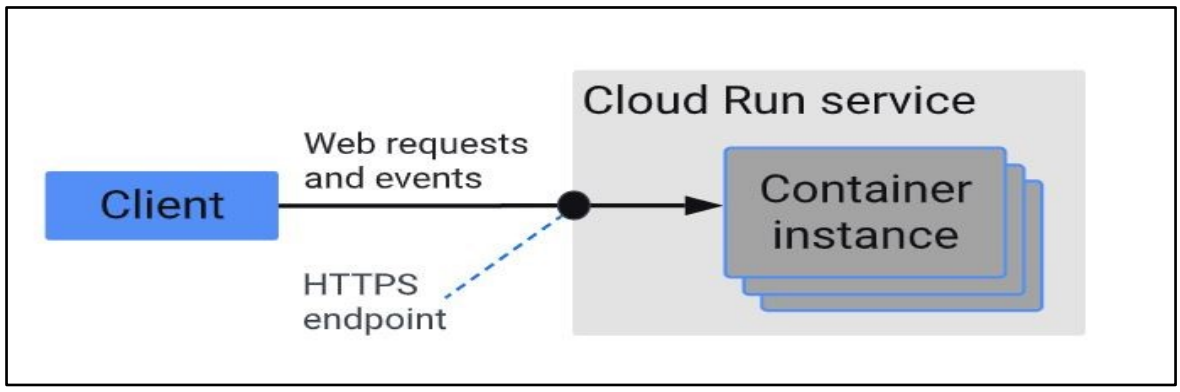
Cloud Run on täysin hallinnoitu container-teknologiaan perustuva Container as a Service -alusta. Cloud Runilla voidaan julkaista sovelluksia Cloud Run Services- ja Cloud Run Jobs -alustoilla. Google Cloud Runin avulla voidaan toteuttaa www-sivustoja, web-sovelluksia, API-rajapintoja, mikropalveluita ja striimattavan datan prosessointia Cloud Run Services -alustalla. Työjonojen tai tehtävien prosessointi voidaan toteuttaa Cloud Run Jobs -alustalla. (Google 2022i.)

4.3.1 Cloud Run -ympäristö

Cloud Run Services

Cloud Run Services -alustan avulla voi julkaista millä tahansa container-teknologiaa tukevalla ohjelmointikielellä toteutetun sovelluksen, joka vastaa web-pyyntöihin tai tapahtumiin. Lisäksi Cloud Runilla voidaan julkaista Go, Java, Node, Python, Kotlin, Ruby ja .NET Core -kielten lähdekoodista automaattisesti rakennettava container. (Google 2022i; Google 2022j.)

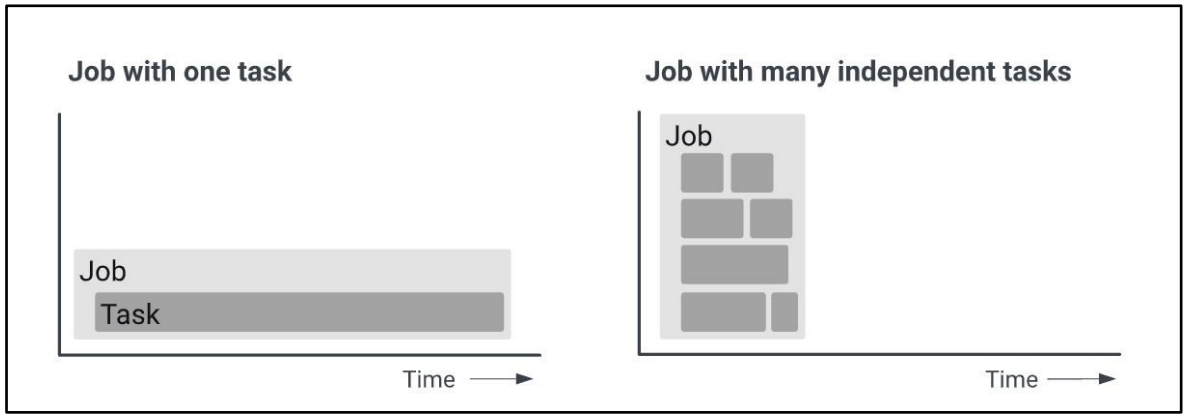
Cloud Run tarjoaa automaattisesti jokaiselle palvelulle HTTPS-päätepisteen, TLS-terminoinnin, tukee WebSocket-, HTTP/2- ja gRPC-liikennettä. Kysynnän kasvaessa Cloud Run skaalaa palvelun tuhanteen container-instanssiin ja kysynnän vähetessä poistaa tarpeettomat containerit (*scale to zero*) (Kuva 13). Container-instanssien määrää voidaan tarvittaessa lisätä kiintiötä kasvattamalla. Cloud Runissa voi myös määritellä minimi-instanssien lukumäärän sovelluksen toiminnan vaatimusten mukaan. Cloud Runissa on sisäänrakennettu liikenteen ohjaus viimeisimmälle versiolle, edelliselle versiolle tai liikenne voidaan jakaa eri versioille samanaikaisesti. Cloud Run mahdollistaa sovelluksen julkaisun yksityisenä tai julkisena. Cloud Run voi hyödyntää tiedostojen tallennukseen käyttömuistipohjaista tiedostojärjestelmää, Cloud Storagea ja NFS-verkkolevyjärjestelmää. (Google 2022i; Google 2022j.)



Kuva 13. Google Cloud Run Servicen rakenne (Google 2022i)

Cloud Run Jobs

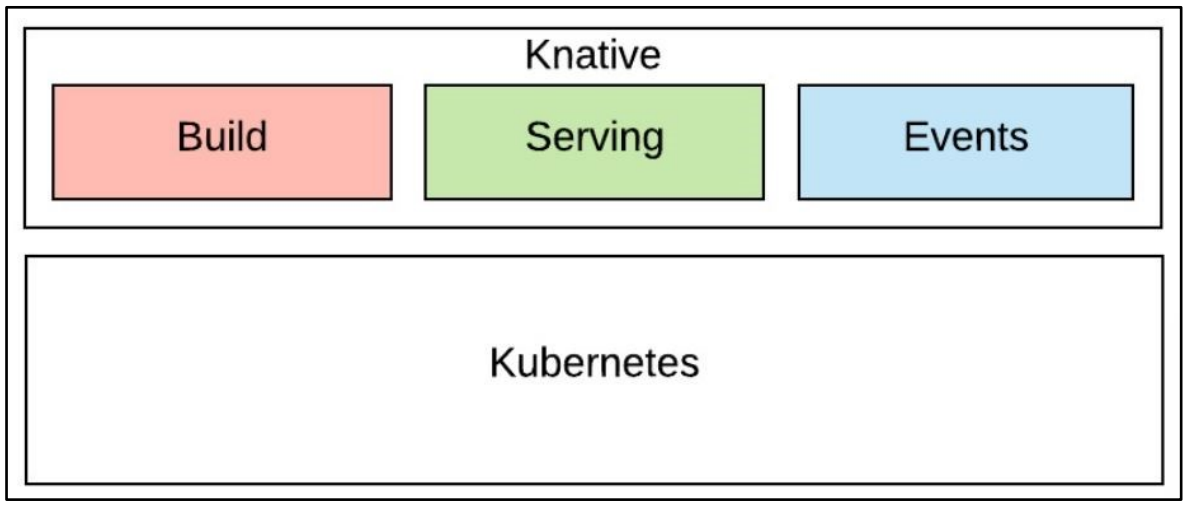
Cloud Run Jobs -alustalla voidaan suorittaa töitä, joilla on alku ja loppu. Cloud Run Jobs voi suorittaa töitä peräkkäisenä tai rinnakkaisena suorituksena (Kuva 14). Rinnakkaisen suorituksen avulla jokaiselle työlle luodaan oma container-instanssi. (Google 2022i.)



Kuva 14. Google Cloud Run Jobin menetelmät (Google 2022i)

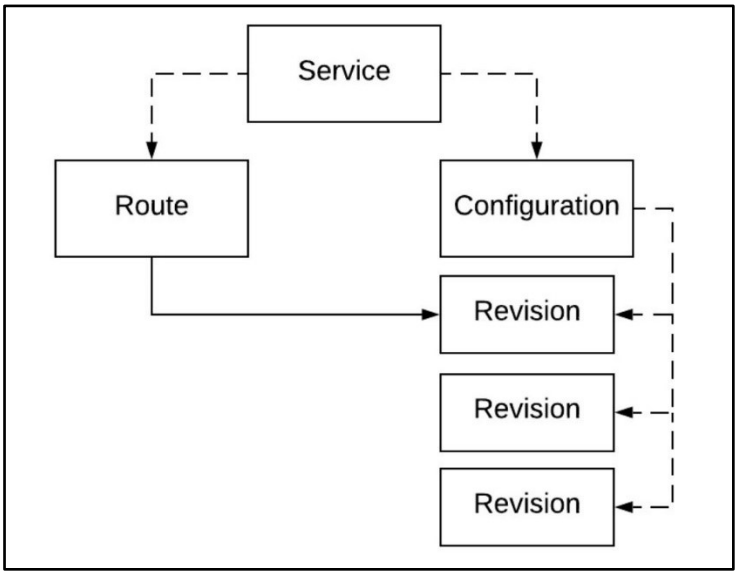
4.3.2 Cloud Runin rakenne ja integraatiot

Google Cloud Run on rakennettu suoraan Google Borgin, skaalautuvan container-infrastruktuurin päälle. Borg mahdollistaa erittäin nopean resurssien skaalauksen Cloud Runin kuorman kasvaessa. (Venema 2021). Ylemmällä abstraktiotasolla Cloud Run käyttää avoimen lähdekoodin Knative-ympäristöä Kubernetesin säiliönhallintajärjestelmän päällä. Knativen avulla voidaan rakentaa, jakaa, ajaa ja hallinnoida Kubernetes-säiliönhallintajärjestelmän sovelluksia. Knativen Build-toiminnon tai Google Buildin avulla voidaan rakentaa container Cloud Run -alustalle. Knativen Serving -toiminto hallinnoi containerin työkuorman, elinkaaren, reitittää liikenteen yhdelle tai useammalle revisiolle (*traffic splitting*), hallinnoi konfiguraatiota ja revisioita (Kuva 15).



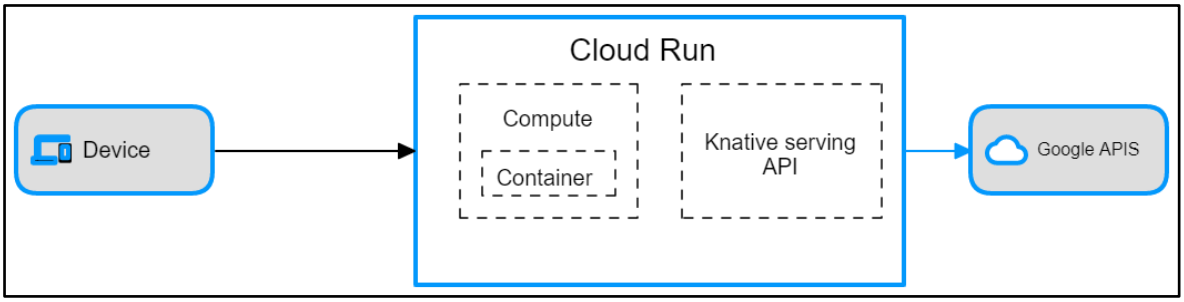
Kuva 15. Knativen rakenne (Sangapu ym. 2021)

Julkaistun ohjelmiston versionhallinta on toteutettu Serving-toiminnon hallinnoimilla revisioilla. Hallinnoitujen revisioiden avulla voidaan tarvittaessa palata aikaisempaan versioon ohjelmasta (Kuva 16).



Kuva 16. Cloud Run -version hallinta (Sangapu 2021)

Knative Eventsin avulla Cloud Run voi käyttää HTTPS-, gRPC-, Pub/SUB-, Schedule-, Asynchronous Cloud Tasks- ja Webhook targets-triggereitä toimintojen käynnistämiseksi ja ohjaamiseksi. (Sangapu ym. 2021.) Google Cloud Run voi hyödyntää Googlen muita palveluja Google APIn rajapintojen avulla Knative Serving APIn kautta (Kuva 17) (Rose 2020).



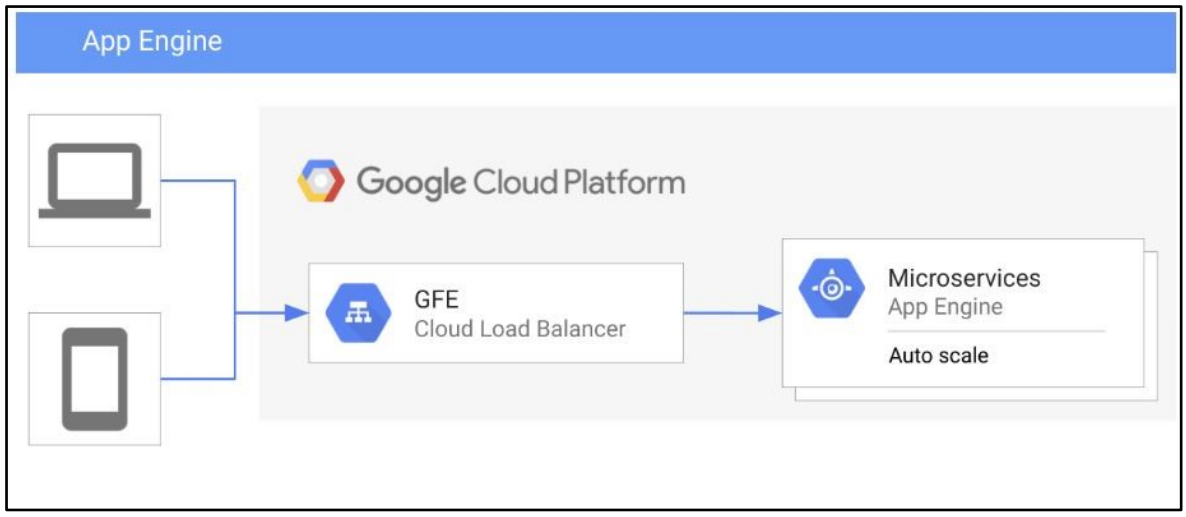
Kuva 17. Cloud Run -integraatiot (mukailtu Rose 2020, 160)

4.4 Google App Engine (PaaS)

App Engine on Googlen Platform as a Service -alusta, jonka avulla voidaan toteuttaa skaalautuvia web-sovelluksia, mobiilisovellusten backendejä ja HTTP/HTTPS API -rajapintoja. App Enginen avulla kehittäjä voi keskittyä sovelluksen business-logiikan kehittämiseen, alustan hoitaessa skaalauksen ja hallinnoinnin automaattisesti. (Treat 2019; Rose 2020.)

4.4.1 App Engine -ympäristö

Google Front End tarjoaa App Enginellä toteutetuille web-sovelluksille IP-pohjaisen sovelluksen hostauksen, Denial of Service -suojauksen, verkkoliikenteen TLS-terminoinnin, DNS-nimipalvelun ja sallii vain HTTP/HTTPS-protokollien liikenteen sisäänpäin. Kaikki sisään tuleva liikenne välitetään Google Front Endin kautta App Enginelle, sovellukselle tulevien verkkoprotokollien rajaamiseksi. GAE:ssa on myös automaattisesti skaalautuva Cloud Load Balancer -kuormantasaaja, jonka avulla liikennettä ohjataan useille taustaprosesseille HTTP/HTTPS-protokollien välityksellä (Kuva 18). Google App Enginestä on valittavissa Standard ja Flexible -versiot, jotka jakavat pääosin samat ominaisuudet, joitakin poikkeuksia lukuun ottamatta. (Rose 2020; Google 2022e.)



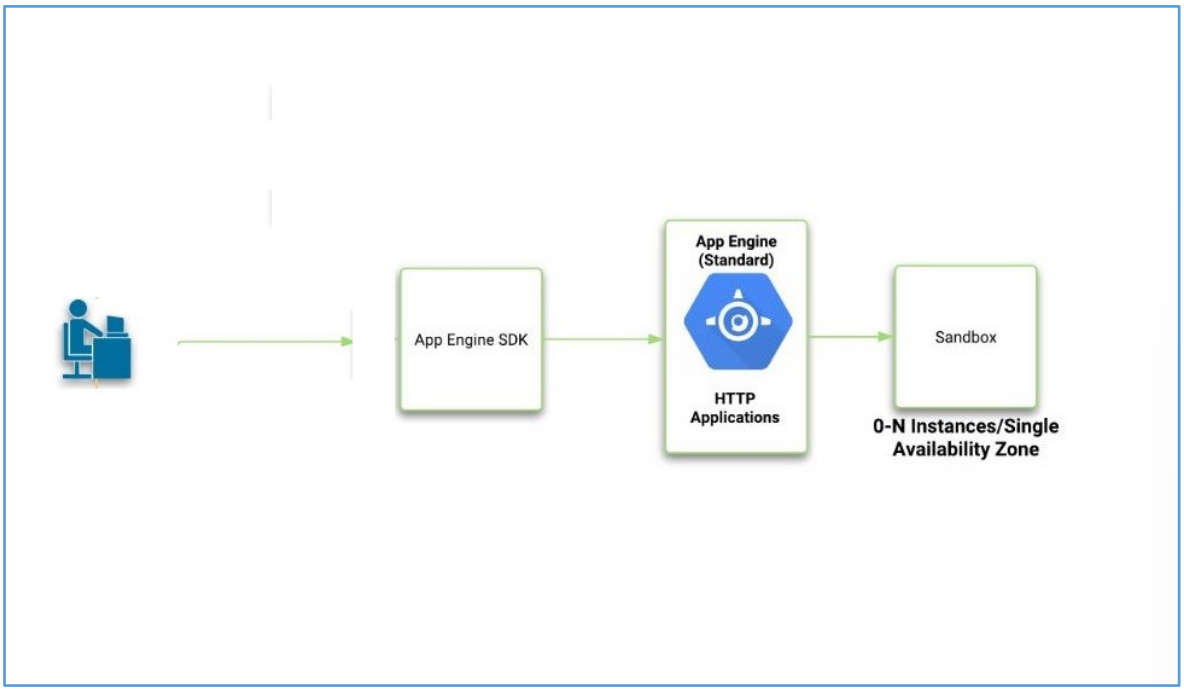
Kuva 18. Google App Engine -rakenne (Rose 2020)

Standard

Standard-ympäristössä sovellus ajetaan etukäteen konfiguroidussa container-instanssissa ja tarvittaessa Standard-ympäristö skaalautuu automaattisesti nollaan aktiiviseen

instanssiin. Täysin dynaaminen resurssien skaalautuminen on kustannuksiltaan edullisempaa kuin Flexible-ympäristöstä koituvat kustannukset.

Standard-ympäristön instanssin käynnistymisaika ja sovelluksen käyttöönotto vie joitakin sekunteja ja suorituksen aikaista container-instanssia ajetaan eristetyssä hiekkalaatikossa (*sandbox*). Etukäteen konfiguroidun containerin ja sandbox-tilan vuoksi Standard-ympäristössä on käytössä vain rajallinen määrä ohjelmistokehityskirjaston (SDK) ohjelmointikieliä (Kuva 19). Tällä hetkellä tuettuina App Engine SDK:n tukemat ohjelmointikielet ovat Node.js, Java, Ruby, Go, Python ja PHP. (Rose 2020; Google 2022e.)



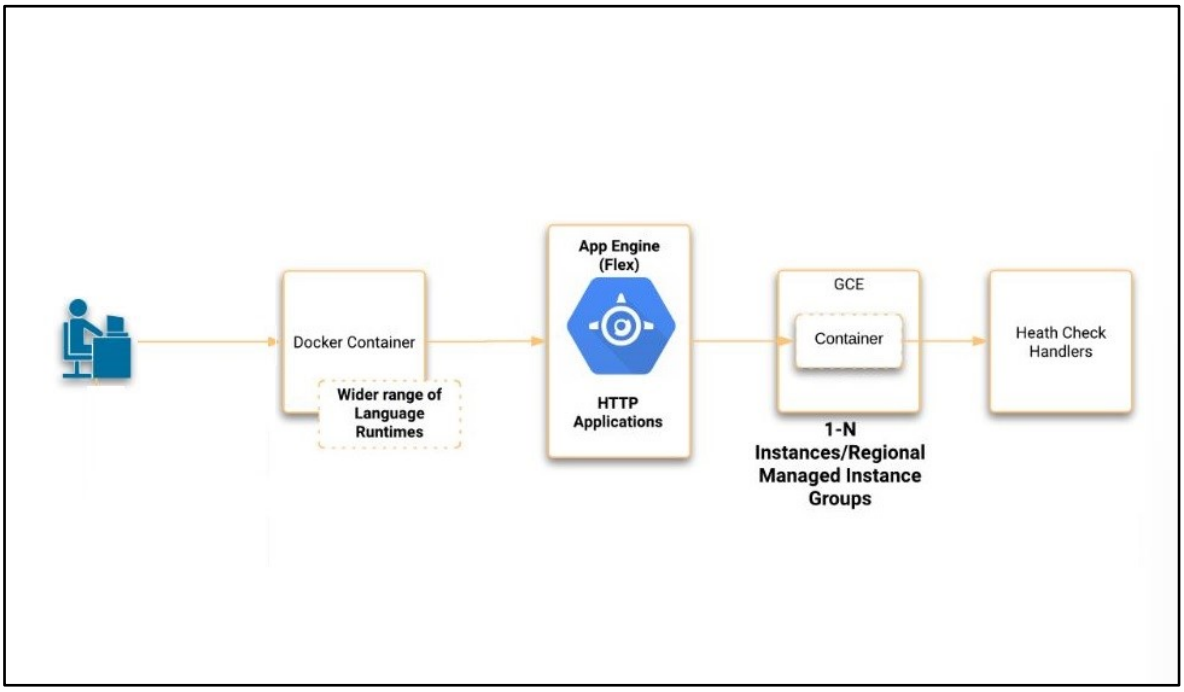
Kuva 19. Google App Engine Standardin ajonaikainen ympäristö ja skaalautuvuus (mukailtu Rose 2020)

Flexible

Flexible-ympäristössä on olemassa aina yksi aktiivinen instanssi ja se ei skaalaudu nollaan. Yhden aktiivisen instanssin myötä, Flexible-ympäristö on lähtökohtaisesti kustannuksiltaan kalliimpi kuin Standard-ympäristö.

Flexible-ympäristön käynnistymisaika ja sovelluksen käyttöönotto vie joitakin minuutteja Google Cloud Enginen kylmäkäynnistyksen ja containerin käynnistyksen myötä. Flexible-ympäristön suorituksen aikaisia instansseja ajetaan Google Compute Enginen virtuaalikoneista muodostetuista alueellisesti hallinnoiduissa instanssiryhmissä (*Managed Instance*

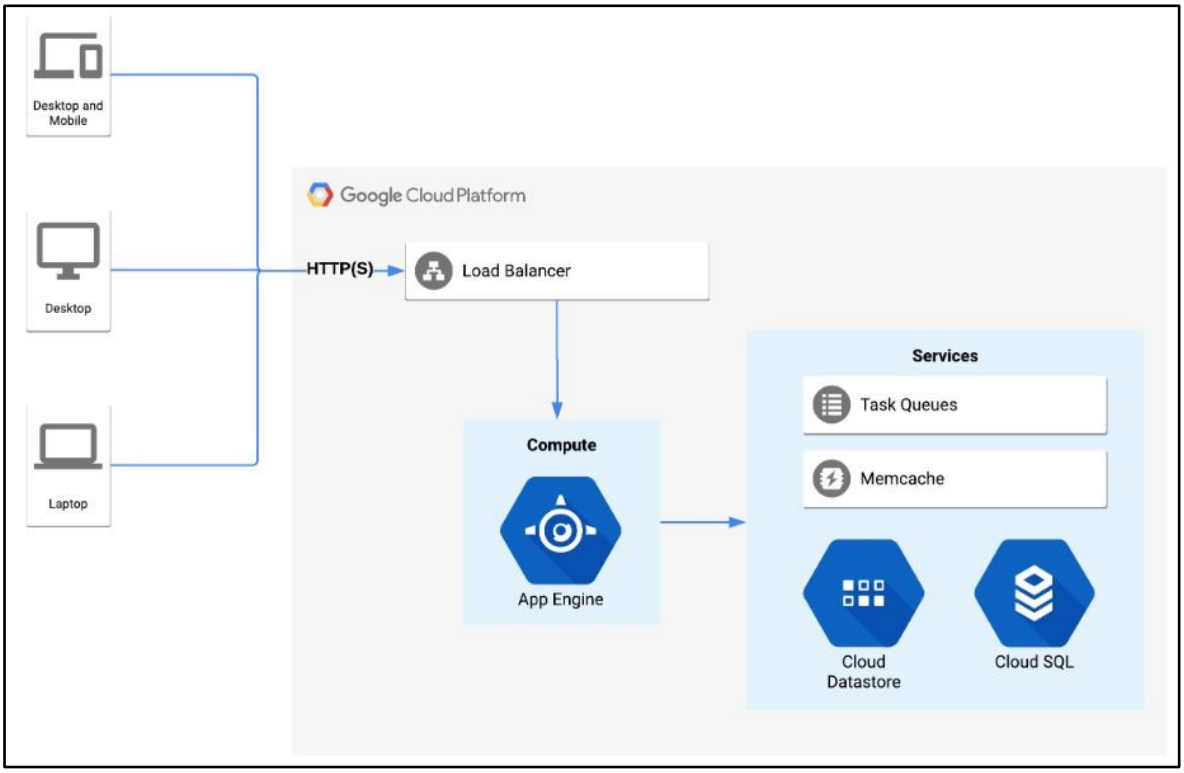
Goup). Hallinnoitujen MIG-ryhmien avulla mahdollistetaan Flexible-ympäristön automaattinen skaalautuvuus (Kuva 20). Flexible-ympäristössä suoraan tuettuja ohjelmointikieliä ovat Python, Java, Node.js, Go, Ruby, PHP ja .NET. Myös muiden ohjelmointikielten käyttö ajonaikaisessa ympäristössä on mahdollista kustomoidun Docker-imagen avulla. (Rose 2020; Google 2022e.)



Kuva 20. Google App Engine Flexin ajonaikainen ympäristö ja skaalautuvuus (mukailtu Rose 2020)

4.4.2 App Enginen rakenne ja integraatiot

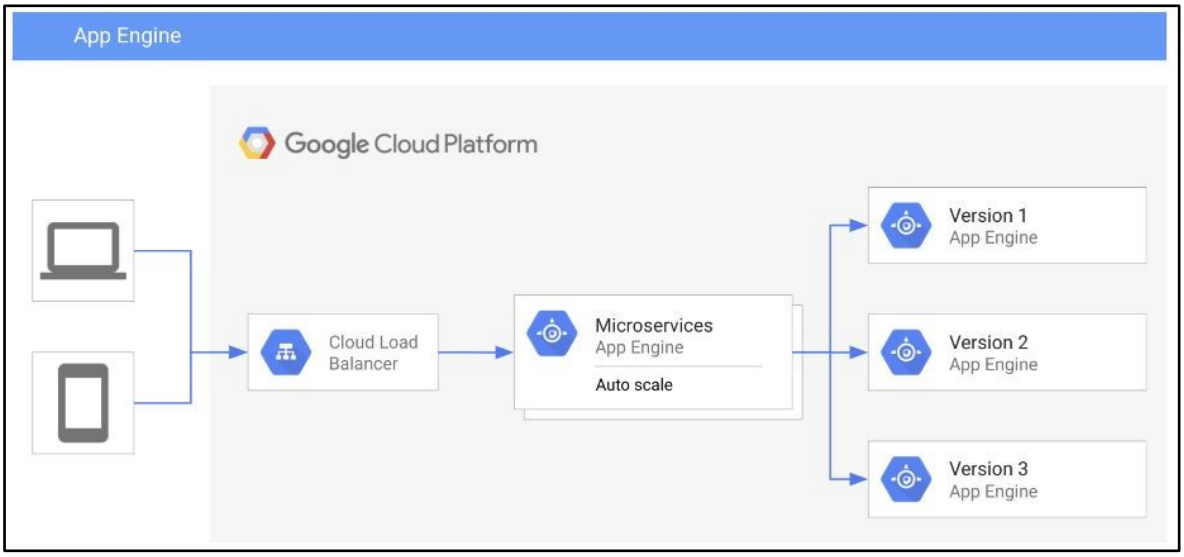
Google App Engine käyttää lukuisia Google Cloudin sisäisiä palveluja, jotta pilvinatiivin serverless-sovelluksen käyttöönotto on helppoa ja konfiguroinnin tarve on minimoitu. App Engineen ohjattu liikenne ohjataan Google Front Endin kuormantasaajalle, Load Balancerille. Load Balancerilta liikenne kulkee App Enginelle. App Engine käyttää Services-kerroksen Memcache- ja Task Queues -palveluita. Tietovarastona voidaan käyttää Cloud SQL -relaatiotietokantaa tai Cloud Datastore- tai Cloud Firestore NoSQL -tietokantoja (Kuva 21).



Kuva 21. Google App Enginen rakenne (Rose 2020)

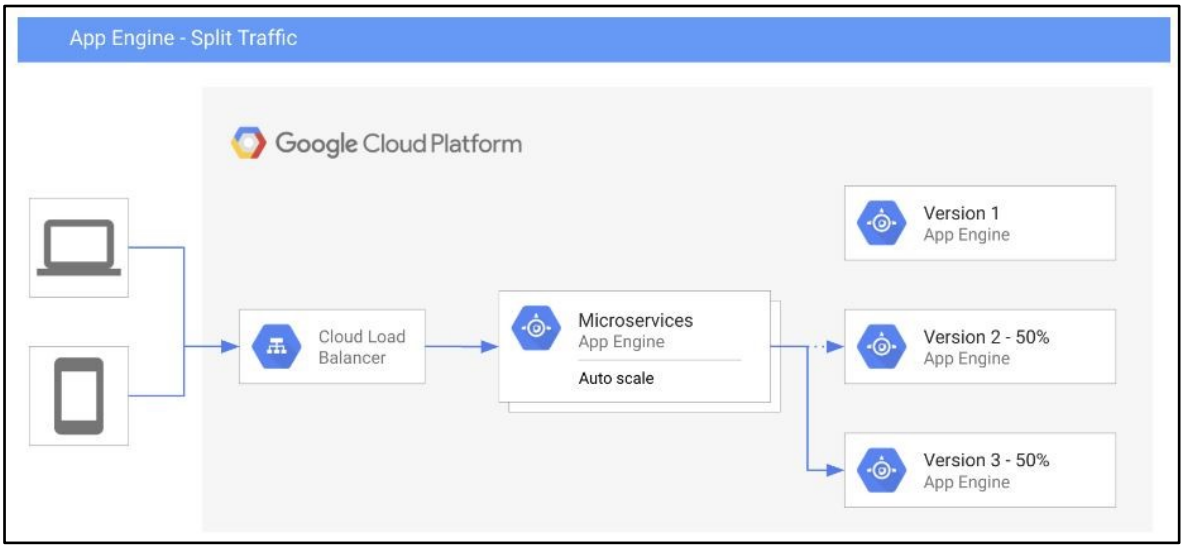
Service-kerroksen Task Queue:n avulla App Engine ylläpitää web-liikenteen nopeaa vastetta eristämällä web-liikenteen pyynnöt varsinaisesta prosessoinnista siirtämällä web-pyynnön prosessointia vaativat tehtävät Task Queuen Pull- tai Push -jonoille. Service-kerroksen Memcachen avulla App Engine muodostaa käyttömuistiin puskurin pysyvästä tietovarastosta pyyntöjen perusteella ja mahdollistaa nopean pääsyn lyhytaikaisesti tarvittavaan dataan. Sovelluksessa voidaan hyödyntää Google Cloudin muita palveluja Services-kerroksen tarjoaman Google Cloud API -integraation avulla. (Rose 2020.)

Google App Engine ylläpitää automaattisesti eri versioita sovelluksesta. Kehittäjän tehdessä uuden julkaisun, App Engine luo siitä automaattisesti uuden version. Jokaisella versiolla on oma URL-osoite ja versiot ovat nähtävillä samanaikaisesti. Lähtökohtaisesti kaikki liikenne ohjataan oletusversiolle. Julkaistu versio voidaan päivittää uudella versiolla tai tarvittaessa voidaan palata aikaisempaan julkaistuun versioon (Kuva 22). (Rose 2020.)



Kuva 22. Google App Engine -version hallinta (Rose 2020)

Liikenteen jakamisen avulla kehittäjät voivat helposti liikkua eri versioiden välillä. App Enginen liikenteen jaon avulla voidaan myös määrätä, kuinka paljon liikenteestä menee tietylle versiolle. Tätä ominaisuutta voidaan hyödyntää A/B-testauksessa kahden eri julkaistun version välillä. Liikenteen jako voidaan toteuttaa IP-osoitteen perusteella, keksien avulla tai satunnaisesti hyödyntäen IP-osoitteita ja keksejä (Kuva 23). (Rose 2020.)



Kuva 23. Google App Engine -liikenteen jakaminen (Rose 2020)

4.5 Google Cloud Functions (FaaS)

Cloud Functions on Functions as a Sevice -alusta, jonka avulla on mahdollista kehittää itsenäisiä, tiettyjä käyttötarkoitusta varten luotuja tapahtumaohjattuja pilvipalvelufunktioita. Cloud Functions -funktioiden avulla voidaan toteuttaa reaaliaikaisia tiedostojen käsittelytehtäviä, tapahtumaohjattuja työputkia, IoT-backendejä, webhookeja ja kolmannen osapuolen integraatioita APIen avulla. Funktioiden avulla voidaan kytkeä toisiinsa muita sovelluksia ja palveluita. (Rose 2020; Google 2022k.)

4.5.1 Cloud Functions -ympäristö

Cloud Functions -funktiot ovat automaattisesti skaalautuvia, tapahtumaohjattuja ja funktioiden käytössä ovat HTTP/HTTPS-endpoint- ja taustapalvelupohjaiset triggerit funktioiden aktivoimiseksi. Tietotuvallisen ympäristön varmistamiseksi funktiolle määritetään palvelutili ja suoritus tapahtuu palvelutilin alaisuudessa. Funktiot voivat olla julkisia, mutta funktiota kutsuva triggeri voidaan määritellä käyttämään autentikointia funktion suoritusympäristön ja siihen liittyvien pilvipalvelujen suojaamiseksi. Funktiot ovat luonteeltaan tilattomia, suoritus tapahtuu turvatussa ympäristössä ja funktiot eivät käytä jaettua muistia. Google Cloud Functions:sta on kaksi versiota Cloud Functions (1st gen) ja Cloud Functions (2nd gen). (Rose 2020; Google 2022k.)

4.5.2 Cloud Functionsin rakenne ja integraatiot

Google Cloud Functions (1st gen) -funktioita suoritetaan Googlen hallinnoidussa ympäristössä. Google ei avaa ensimmäisen sukupolven funktioiden hallinnoitua ympäristöä tarkemmin. Google Cloud Functions (2nd gen) -funktiot on rakennettu Cloud Runin päälle ja se hyödyntää Google Eventarc:in asynkronista tapahtumaohjattua arkkitehtuuri -palvelua. Ensimmäisen ja toisen sukupolven funktioita voidaan kirjoittaa Node-, Python-, Go-, Java-, .NET-, Ruby- ja PHP-kielillä. (Google 2022l; Google 2022m.) Taulukossa 1 on Cloud Functions -funktioiden käytössä olevat triggerit (Taulukko 1).

Cloud Functions (2nd gen)	**Cloud Functions (1st gen)**
HTTP/HTTPS	HTTP/HTTPS
Pub/Sub	Pub/Sub
Cloud Storage	Cloud Storage
Generalized Eventarc	Firestore
Eventarc:in tapahtumatyypit (Google-palvelut, Google SaaS sovellukset ja omat sovellukset), lisäksi yli 90 tapahtumalähdettä Cloud Audit Logista	Google Analytics for Firebase
	Firebase Realtime Database
	Firebase Authentication
	Firebase Remote Config

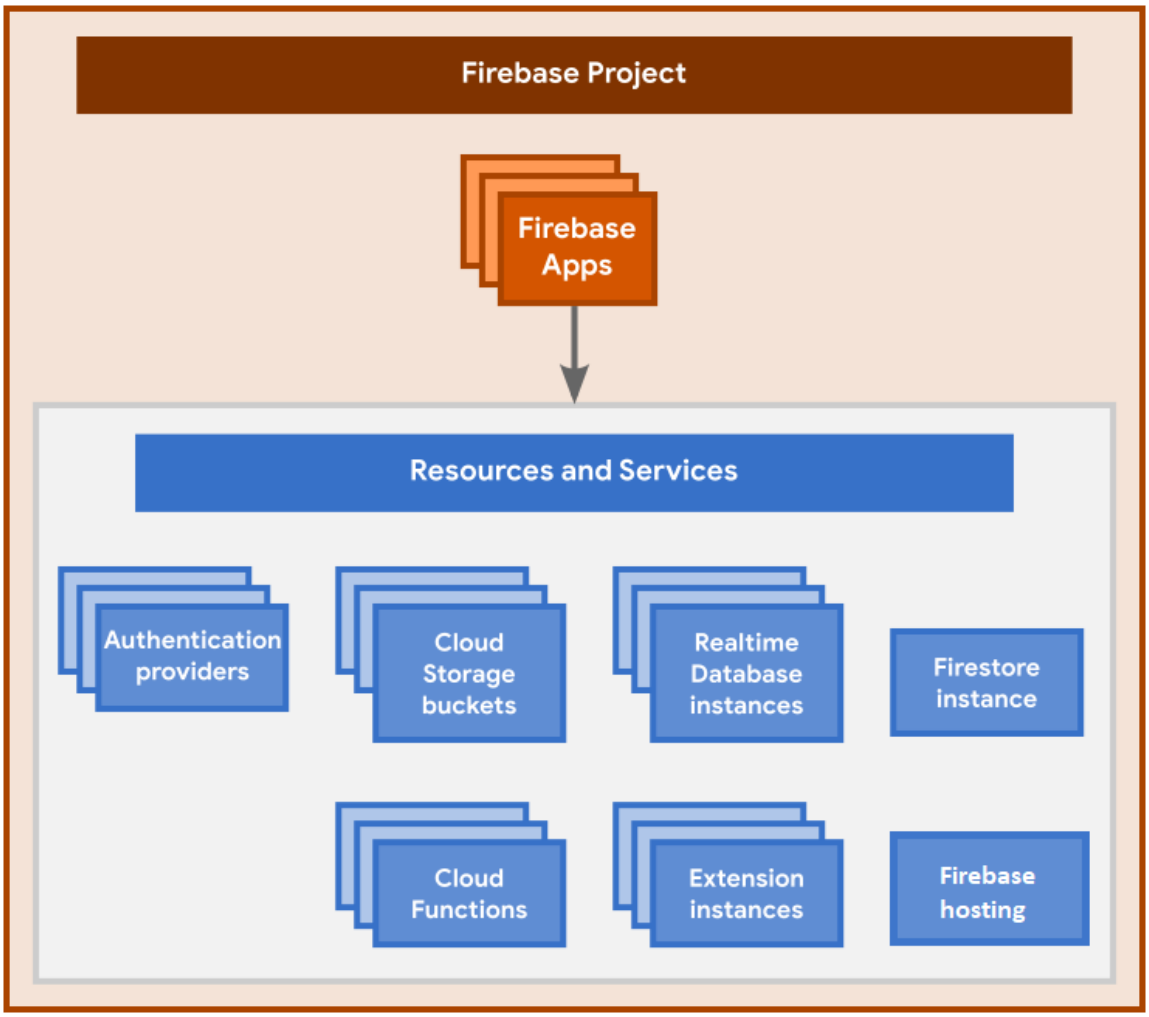
Taulukko 1. Cloud funktioiden triggerit (mukailtu Google 2022n)

4.6 Google Firebase (BaaS)

Firebase on Backend as a Service -alusta, jonka avulla voidaan toteuttaa mobiili-, web- ja pelisovelluksia. Firebase on automaattisesti skaalautuva alusta, johon on yhdistetty lukuisia Googlen palveluita. Firebase vapauttaa kehittäjät palvelintarpeen määrittelystä, palvelinten konfiguroinnista ja ylläpidosta. Kehittäjät voivat keskittyä sovelluksen toteutukseen. (Treat 2019.)

4.6.1 Firebase-ympäristö

Firebase koostuu projektista, joka voi sisältää yhden tai useamman sovelluksen. Projekti voi sisältää maksimissaan 30 sovellusta. Sovellukset voivat hyödyntää projektiin liitettyjä Google Cloudin Authentication providers-, Cloud Storage Buckets-, Realtime Database Instance-, Firestore Instance-, Cloud Function-, Extension Instances-, Firebase Hosting -taustapalveluja ja tuotteita (Kuva 24). (Google 2022p.) Google ei avaa tarkemmin mitä Googlen sisäisiä palveluja Firebase käyttää toteutuksessa.



Kuva 24. Firebase-projektin rakenne ja tuotteet (mukailtu Google 2022p).

Firebase-projektin luonnin yhteydessä luodaan taustalla Google Cloud -projekti. Tämä mahdollistaa Firebase-projektin hallinnoinnin Firebase Consolen, Google Cloud Consolen, Firebase CLI:n tai Google Cloud CLI:n kautta. Koska Firebase-projekti on Google Cloud -projekti, kaikki Google Cloudin APIt ja tuotteet ovat käytössä Firebase sovelluksella. (Google 2022p.)

Firebase client SDK on saatavilla iOS-, Android-, Web-, C++ ja Unity-ympäristöille (Google 2022o). Firebase Admin SDK voidaan asentaa Node-, Java-, Python-, Go tai .NET ajonaikaista ympäristöä tukevalle palvelimelle. Palvelinpohjainen asennus mahdollistaa kriittisten toimintojen suorittamisen suojatussa palvelinympäristössä. Ohjelmallinen projektien luonti ja ylläpito sekä projekteihin liittyvien resurssien hallinnointi on mahdollista Firebase Management REST API:n avulla. (Google 2022p.)

4.6.2 Firebase Products ja Extensions

Firebase-projektille voidaan lisätä sovelluskehitykseen, monitorointiin ja julkaisuun sekä käyttäjien sitouttamiseen tarvittavat tuotteet Firebase Consolen kautta. Käyttöönotto ei vaadi asennusta kehitysympäristöön. Firebase-pohjaisten ohjelmistojen kehitystyössä voidaan käyttää taulukossa 2 kuvattuja Firebase-tuotteita. (Google 2022r.)

Firebase product	**Kuvaus**	**Firebase product**	**Kuvaus**
Cloud Firestore	NoSQL-tietokanta	App Check	Backend resurssien suojaus
Cloud Functions	Pilvipalvelu-funktiot	Authentication	Autentikointi
Hosting	Hostaus	Cloud Storage	Objektivarasto
Realtime Database	NoSQL-tietokanta	Crashlytics	Kaatumisten seuranta
Performance Monitoring	Monitorointi	Test Lab	Sovellustestaus
App Distribution	Sovelluksen jakelu	Google Analytics	Analytiikka
Machine Learning	Koneoppiminen	In-App Messaging	Viestintäpalvelu
A/B Testing	Tuote- ja markkinointitestaus	Cloud Messaging	Alustariippumaton pilvipalveluviestintä
Remote Config	Etäkonfigurointi	Dynamic Links	Dynaamiset linkit

Taulukko 2. Firebase Products -luettelo (Google 2022r)

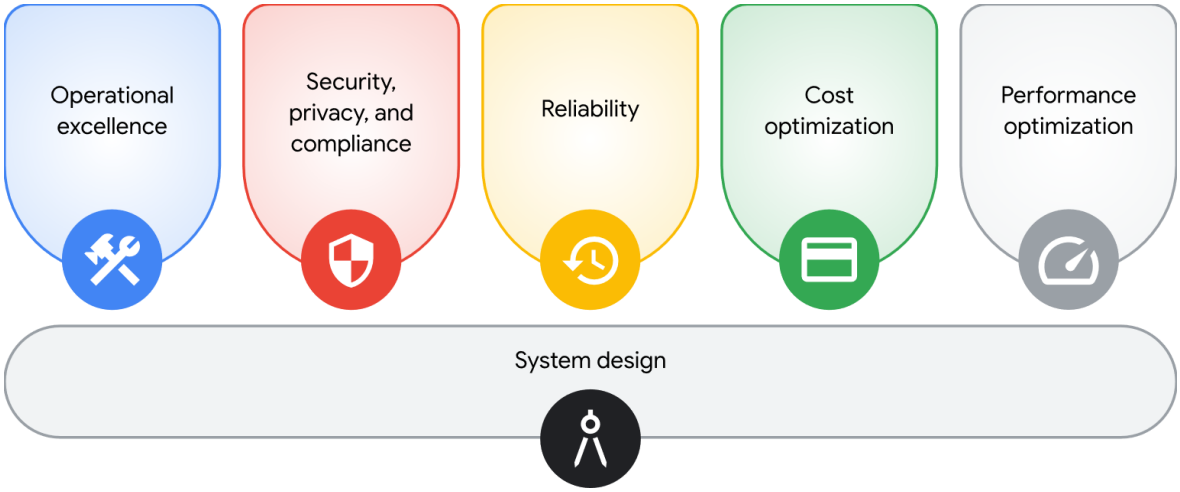
Firebase-projektin käyttöön voidaan tarvittaessa ottaa käyttöön laajennoksia (Firebase Extensions), joiden avulla voidaan toteuttaa tiettyä tarkoitusta varten lisätoimintoja. Firebase Extensionit asennetaan Firebase Consolen kautta. Käyttöönotto ei vaadi asennusta kehitysympäristöön. Extensionit ovat Googlen tai kolmannen osapuolen kehittämiä maksullisia tai ilmaisia laajennoksia.

5 Serverless-alustan valintaa ohjaavat tekijät

5.1 Google Cloud Architecture Frameworkin perusperiaatteiden hyödyntäminen

Ohjelmistot vaihtelevat yksinkertaisista sovelluksista ja rajapinnoista erittäin monimutkaisiin kokonaisuuksiin. Parhaan alustan valitseminen tietynlaiselle ohjelmistolle tiettyä käyttötarkoitusta varten on haastavaa serverless-alustojen tarjonnan ja ohjelmistojen vaatimusten myötä.

Arkkitehtuurin topologian kokonaisvaltaisen suunnittelun apuna voi käyttää Google Cloud Architecture Framework -kehystä, joka tarjoaa suosituksia ja parhaita käytäntöjä pilvipalvelupohjaisen ohjelmiston toteuttamiseksi. Architecture Framework on jaettu kuuteen kategoriaan (Kuva 25), jotka tulee huomioida pilvipalvelun arkkitehtuurin suunnittelussa. (Google 2022s.)



Kuva 25. Google Cloud Architecture Frameworkin kategoriat

Yksinkertainen arkkitehtuuri

Google Cloud Frameworkin System design -kategorian perusperiaatteet suosittelevat käyttämään yksinkertaista pilvipalvelun suunnittelumallia. Monimutkaisten arkkitehtuurien toteuttaminen on yleensä haastavaa ja ylläpito voi muuttua ajan myötä hankalaksi. Cloud Framework suosittelee käytettäväksi mahdollisimman pitkälle hallinnoitua serverless-alustaa. Näin vähennetään alustan konfigurointiin, ylläpitoon ja hallinnointiin kuluvaa aikaa sekä työmäärää. Pitkälle hallinnoitu serverless-alusta minimoi alustoihin liittyviä riskejä. (Google 2022t.)

Mikropalvelumalli

Sovelluksen arkkitehtuurin pilkkominen pienempiin komponentteihin ja palveluihin mahdollistaa komponenttien, funktioiden ja palvelujen päivittämisen toisistaan riippumattomasti. Mikropalvelumallin avulla voidaan luoda kohdennettuja tietoturvamenetelmiä, luotettavuusmäärittelyjä komponentti ja palvelu kerrallaan. Ositettu arkkitehtuurimalli helpottaa ohjelmiston monitorointia ja tarkan suorituskyvyn sekä hinnan seurannan. Toisaalta liiallinen ohjelmiston pilkkominen johtaa helposti erittäin monimutkaiseen rakenteeseen ja on ristiriidassa Sytem Design -kategorian perusperiaatteissa esitetyn yksinkertaisen suunnittelumallin kanssa. Arkkitehtuurin suunnittelun yhteydessä on tarkoin harkittava komponenttien, palvelujen ja funktioiden pilkkomista itsenäisiksi prosesseiksi. (Google 2022t.)

Tilaton arkkitehtuuri

Tilattoman arkkitehtuurin hyödyntämisen avulla ohjelmiston skaalautuvuutta ja luotettavuutta voidaan lisätä. Tilattomalla ohjelmistolla ei ole ylimääräisiä riippuvuuksia tehtävien suorittamiseksi tai ohjelmiston käynnistymiseen liittyen. Näin mahdollistetaan ohjelmiston nopea käynnistyminen ja skaalautuminen tarvittaessa. (Google 2022t.)

5.2 Serverless-alustojen ominaisuudet

Alustoilla on toisistaan poikkeavia piirteitä ja ominaisuuksia. Suurimmalla osalla serverless-alustoista voidaan julkaista sama ohjelmisto muuttamalla ohjelmiston toteutustapaa alustan asettamien vaatimusten mukaisesti. Eri alustojen ominaispiirteet tulee ottaa huomioon, jotta alustan ominaisuuksia voidaan hyödyntää täysimääräisesti, kyetään minimoimaan kehitykseen käytettävää aikaa, vähennetään kustannuksia ja saavutetaan pilvipalveluoperaattorin hallinnoinnista maksimaalinen hyöty. Kuvassa 26 on esitetty eri serverless-alustojen ohjelmiston hostaukseen liittyvät ominaisuudet.

Options	GKE	Cloud Run	App Engine flex	App Engine stand.	Cloud Functions	Firebase
Deployment format	Cluster	Application code or Container	Application code or container	Application code	Function	Application code
Custom URLs	✓	✓	✓	✓	✗	✓
Scale to zero	✗	✓	✗	✓	✓	✓
Free tier	✗	✓	✗	✓	✓	✓
Persistent disks	✓	✗	✗	✗	✗	✗
Custom system packages	✓	✓	✓	✗	✗	✗
Websockets	✓	✓	✓	✗	✗	✓
Run any language	✓	✓	✓	✗	✗	✗
Request timeout	None	60 min.	60 min.	1 min.	9 min.	1 min.
Background processes	✓	✓	✓	~	✗	●
GPU / TPU access	✓	✗	✗	✗	✗	✗
VPC connectivity	✓	✓	✓	✓	✓	●
Managed event driven support	✗	✓	✗	✗	✓	Use Pub/Sub

✓ Supported ✗ Not supported ~ Basic- ja manual scaling ● via Cloud functions

Kuva 26. Serverless-alustojen hosting ominaisuudet (mukailtu Google 2022v)

Alustojen ominaisuuksien perusteella voidaan määrittää ohjelmistolle soveltuvia käyttökohteita. Useimpia alustoja voidaan käyttää samankaltaisten ohjelmistojen ja toiminnallisuuksien toteuttamiseen. Ominaisuuksissa on myös merkittäviä eroja, jotka ohjaavat alustan valintaa.

Kubernetes Engine

Kubernetes Engine on ominaisuuksiltaan kattavin alusta. Kubernetes Engine soveltuu useita containereita käyttävän, pilvinatiivin tai mikropalvelupohjaisen ohjelmiston kehittämiseen. Kubernetes Engine hoitaa myös ohjelmiston skaalauksen ja kuormantasauksen automaattisesti. Kubernetes Enginen avulla voidaan myös toteuttaa ohjelmistoja, jotka käyttävät HTTP / HTTP2 tai gRPC:n lisäksi muita protokollia. (Treat 2019; Vergadia 2022.)

Kubernetes Engine soveltuu

- tilan säilyttäviin ohjelmistoihin
- tilattomiin ohjelmistoihin
- AI / ML -palveluihin
- skaalautuvaan datan käsittelyyn

- skaalautuviin pelialustoihin
- kuormansietokykyä vaativille ohjelmistoille (Treat 2019; Vergadia 2022).

Cloud Run

Cloud Run soveltuu pilvinatiivin ja mikropalvelupohjaisen ohjelmiston kehittämiseen. Cloud Runilla voidaan toteuttaa websivustoja, tapahtumapohjaisia palveluita, API -palveluita, datan prosessointipalveluita ja webhookkeja. (Treat 2019; Vergadia 2022.)

Cloud Run soveltuu

- tilattomiin ohjelmistoihin
- tapahtumapohjaisiin ohjelmistoihin ja järjestelmiin
- web-sivustojen toteutuksiin
- API -palveluihin
- ohjelmistoihin, jotka vaativat muokattua ajonaikaista ympäristöä ja ohjelmointikielten tukea
- dataa prosessoivien sovellusten ja webhookkien toteuttamiseen (Treat 2019; Vergadia 2022).

Cloud Run ei sovellu

- tilan säilyttäviin ohjelmistoihin
- yksityiskohtaiseen ympäristön ja infrastruktuurin hallintaa vaativaan ympäristöön (Treat 2019; Vergadia 2022).

App Engine

App Enginen avulla voidaan toteuttaa skaalautuvia web-sovelluksia, mobiilisovellusten backendejä ja HTTP/HTTPS API -rajapintoja. App Engine tarjoaa Flexible ja Standard versiot. (Treat 2019; Vergadia 2022.)

App Engine soveltuu

- tilattomiin ohjelmistoihin
- nopeasti kehitettäviin CRUD-sovelluksiin
- muutamasta palvelusta koostuviin ohjelmistoihin
- monimutkaisten APIen toteutuksiin (Treat 2019; Vergadia 2022).

App Engine ei sovellu

- tilan säilyttäviin ohjelmistoihin
- erikoistuneiden ohjelmistokehyksien avulla toteutettuihin ohjelmistoihin

- ohjelmistoille, jotka tarvitsevat HTTP / HTTPS:n lisäksi muita protokollia (Treat 2019; Vergadia 2022).

Cloud Functions

Cloud Functions on hyvä valinta tapahtumapohjaisten ohjelmistojen ja järjestelmien toteuttamiseen. Cloud Functions:ien avulla voidaan toteuttaa API-palveluita ja yhdistää eri alustoilla ajettavia ohjelmistoja ja järjestelmiä. Cloud Functions:sta on tarjolla Cloud Functions (1st gen) ja Cloud Functions (2nd gen) versiot. (Treat 2019; Vergadia 2022.)

Cloud Functions soveltuu

- tapahtumapohjaisten ohjelmistojen ja järjestelmien toteuttamiseen
- eri ohjelmistojen ja järjestelmien yhdistämiseen
- API-palveluiden toteuttamiseen
- webhookkien toteutukseen
- dataa prosessoivien järjestelmien tekoon (Treat 2019; Vergadia 2022).

Cloud Functions ei sovellu

- tilan säilyttävien järjestelmien toteuttamiseen
- monimutkaisten APIen toteuttamiseen
- järjestelmiin, jotka vaativat ympäristön tarkkaa hallintaa
- ohjelmistoihin, jotka vaativat muokattuja ajonaikaisia ympäristöjä tai muita ohjelmistoja (Treat 2019; Vergadia 2022).

Firebase

Firebase on hyvä valinta, jos pitää toteuttaa mobiili / HTML -pohjainen ohjelmisto ja halutaan keskittyä pääosin ohjelmointiin. Firebase soveltuu myös nopeasti kehitettävän ohjelmiston prototyypin toteutukseen. Firebase on erittäin pitkälle hallinnoitu alusta, jonka rajoituksia voidaan minimoida käyttämällä hyväksi muita Googlen serverless-alustoja ja API-palveluita. (Treat 2019; Vergadia 2022.)

Firebase soveltuu

- mobiilipohjaisten ohjelmistojen toteutukseen
- ohjelmistoprototyypin toteuttamiseen
- ohjelmistoille, joissa suurin osa logiikasta toteutetaan clientissä
- myös laajemmille ohjelmistoille hyödyntämällä muita serverless-alustoja ja API-palveluita (Treat 2019; Vergadia 2022).

Firebase ei sovellu

- ohjelmistoille, joiden toteutus vaatii alustan hallintaa
- container -pohjaisten ohjelmistojen julkaisuun (Treat 2019; Vergadia 2022).

5.3 Liiketoiminnan, kehitystyön, ylläpidon ja siirrettävyyden vaatimukset

Ohjelmistoa kehittävällä yrityksellä ja yrityksen asiakkaalla voi olla liiketoiminnan ajureita, teknologisia vaatimuksia, kehitystyöhön ja alustan hallintaan sekä ylläpitoon liittyviä vaatimuksia, jotka vaikuttavat serverless-alustan valintaan. Ohjelmiston kehittäjien osaamisalueet asettavat toisaalta omia vaatimuksia alustan ja teknologioiden suhteen.

Myös ohjelmistolla voi olla teknologisia, kehitystyöhön, ylläpitoon ja hallintaan liittyviä vaatimuksia, jotka tulee ottaa huomioon alustaa valittaessa. Valintaa tulee tarkastella myös ohjelmiston siirrettävyyden näkökulmasta.

5.3.1 Liiketoiminnan ajurit

Ohjelmistoa kehittävän yrityksen tulee ottaa huomioon serverless-alustasta ja alustan käyttämistä resursseista aiheutuvat kustannukset, jotta yritys voi tarjota asiakkaalle kilpailukykyisellä hinnalla oman tuotteen. Kustannuksia tulee optimoida ja käytettäviä resursseja minimoida, vaarantamatta ohjelmiston luotettavaa toimintaa, palvelun laatua, saavutettavuutta ja kykyä vastata tarvittaessa kuormituksen kasvuun. Hyödyntämällä automaattisesti toipuvaa serverless-alustaa ja ohjelmistoa voidaan säilyttää asiakkaan luottamus. (Google 2022y.)

Google tarjoaa laskureita, joiden avulla voidaan arvioida serverless-alustoista ja niiden käyttämien resurssien kustannuksia. Serverless-alustojen laskutusmalleina on käytettävien resurssien mukaan tapahtuva hinnoittelu (*pay as you go*) tai kiinteistä kustannuksista ja resurssien käytöstä muodostuva hinnoittelu (Vergadia 2022). GKE Autopilot, Cloud Run, Cloud Functions ja Firebase käyttävät resurssien mukaan tapahtuvaa hinnoittelua. App Engine käyttää kiinteän instanssikustannuksen ja käytettävien resurssien yhdistelmään pohjautuvaa hinnoittelua.

5.3.2 Kehitystyön ajurit

Ohjelmiston kehitystyössä on pyrittävä minimoimaan virheiden tutkimiseen käytettävää aika, jolloin voidaan lisätä uusien ominaisuuksien kehittämiseen käytettävää aikaa. Tämän tavoitteen saavuttamiseksi automatisoidaan toistuva työ ja käytetään alan parhaita menetelmiä ja käytäntöjä.

Ohjelmistoprojektin toteuttavan tiimin koostuessa pääosin ohjelmoijista, kannattaa valita alusta, jonka hallinnointiin ja ylläpitoon ei tarvitse varata resursseja. Kehitystyö nopeutuu kehittäjien keskittyessä pelkästään koodiin. Sopivia alustoja ovat App Engine, Cloud Run, Cloud Functions ja Firebase. Kehitystiimin omatessa kokemusta Kubernetes-ympäristön määrittelystä sekä hallinnasta ja kehitettäessä kokonaan omaa alustaa, on Kubernetes selkeä valinta. (Google 2022y; Vergadia 2022.)

5.3.3 Ylläpidon ajurit

Ohjelmisto tulisi suunnitella siten, että se kykenee toipumaan automaattisesti virhetilanteista ja yksittäisen komponentin virheen vaikutukset ovat rajoitettuja. Ohjelmiston ylläpitoon liittyvä toistuva työ pitää automatisoida mahdollisimman pitkälle. Näin kyetään säästämään kustannuksia ja vapautetaan resursseja kehitystyöhön. (Google 2022y.)

Google Cloud Frameworkin mukaan tulisi käyttää mahdollisimman pitkälle hallinnoitua alustaa (Google 2022t). Kubernetes Engine tarjoaa laajimman hallinnan tason käyttöjärjestelmätasolta ylöspäin, mutta samalla menetetään osa hallinnan helppoudesta. Cloud Runin avulla hallintavastuu alkaa ajonaikaisesta ympäristöstä ylöspäin ja hallinnan määrä pienenee. App Enginessä hallinta on kehittäjien vastuulla skaalaustasosta ylöspäin, hallinnan määrän pienentyessä. Cloud Functionsin ja Firebasen hallinnointitaso alkaa ohjelmiston koodista ylöspäin. Taulukossa 3 on esitetty ylläpidon ja hallinnoinnin abstraktiotasot. (Vergadia 2022.)

Data & configurations	
Application code	Cloud Functions / Firebase
Scaling	App Engine
Runtime	Cloud Run
OS	Kubernetes Engine
Virtualization	Pilvipalvelun tarjoaja hallinnoi
Hardware	Pilvipalvelun tarjoaja hallinnoi

Taulukko 3. Hallinnoinnin ja ylläpidon abstraktiotaso

5.3.4 Ohjelmiston siirrettävyys

Ohjelmiston pilvipalveluntarjoajasidonnaisuus (*vendor lock-in*) voidaan välttää käyttämällä Open Source-pohjaisia alustaratkaisuja. Kubernetes Engine pohjautuu avoimen lähdekoodin Kubernetes-ohjelmistoon. Cloud Run perustuu avoimen lähdekoodin Knative-alustaan ja Kubernetes-ohjelmistoon. Kubernetes Enginessä ja Cloud Runissa ohjelmistot julkaistaan containerien avulla. Cloud Functions Framework pohjautuu avoimen lähdekoodin ohjelmointikehyksiin ja funktioita voidaan suorittaa Cloud Functions -palvelun alla, kehitysympäristössä, on-premises-palvelimilla, Cloud Run -ympäristössä ja muilla Knative-pohjaisilla alustoilla (Vergadia 2022; Google 2022u). Alusta- ja ohjelmointikielten sidonnaisuuksien myötä App Engine ja Firebase ovat enemmän sidoksissa palvelun tarjoajaan vaikeuttaen ohjelmiston siirtämistä toiselle alustalle tai kokonaisuudessaan toisen pilvipalveluntarjoajan alustalle.

6 Yhteenveto ja pohdinta

Opinnäytetyön tarkoituksena oli luoda tekninen yleiskuva Google Cloudin perusrakenteista, tutkia Google Cloudin serverless-palveluita ja helpottaa ohjelmiston julkaisuun soveltuvien serverless-palveluiden valintaprosessia. Työssä pyrittiin vastaamaan kysymyksiin eri Google Cloudin serverless-palveluiden soveltuvuudesta ohjelmistojen julkaisuun, mitä palveluiden käyttö edellyttää ja mitä pitää ottaa huomioon eri serverless-palveluissa. Tutkimuskysymyksien haettiin vastauksia perehtymällä Google Cloud -pilvipalveluun, Googlen aineistoihin ja oppaisiin, kirjallisuuteen ja tutkimuksiin. Työssä saatiin vastaukset asetettuihin tutkimuskysymyksiin ja asetetut tavoitteet saavutettiin.

Työssä nostettiin esille Google Cloud Frameworkin System Desing -kategoriasta yksinkertaisen arkkitehtuurin, mikropalvelumallin ja tilattoman arkkitehtuurin periaatteet ohjaamaan ohjelmiston suunnittelua ja alustan valintaa. Serverless-palveluiden tarkemman analyysin avulla luotiin kooste alustojen ominaisuuksista ja niiden valintaan vaikuttavista seikoista. Lisäksi työssä nostettiin esille liiketoiminnan, kehitystyön, ylläpidon ja ohjelmiston siirrettävyyden näkökulmat, jotka osaltaan toimivat ajureina ja vaikuttavat alustan valintaan.

Lopputyön raporttia voidaan käyttää apuna valittaessa ohjelmistolle soveltuvaa julkaisualustaa. Erittäin kattavilla ominaisuuksilla alustoista erottuu Kubernetes Engine, Firebase mobiilipohjaisten ohjelmistojen julkaisuun ja Cloud Functions pilvipalvelufunktioiden toteuttamiseen. Täysin yksiselitteistä vastausta oikeasta alustasta tietyssä käyttötapauksessa ei voida antaa, koska suurinta osaa Google Cloudin serverless-alustoista voidaan käyttää samankaltaisten ohjelmistojen julkaisuun alustojen ominaisuuksien eroista huolimatta. Alustojen omien ominaisuuksien asettaessa rajoituksia, voidaan hyödyntää Google Cloud APIa ja käyttää kaikkia Google Cloudin palveluita ja serverless-alustoja toiminnallisuuden lisäämiseksi. Google Cloud Frameworkin System Design -kategorian mukaisesti ohjelmiston toiminnallisuutta tulee pilkkoa, jonka seurauksena tiettyjä toimintoja kannattaa suorittaa toisella serverless-alustalla toteutetulla mikropalvelulla, pilvipalvelufunktiolla tai ohjelmalla. Ohjelmiston jakamisessa pienempiin osiin, tulee kuitenkin ottaa huomioon System Designin yksinkertaisen pilvipalvelun suunnittelumalli.

Google Cloudin serverless-alustojen osalta on löydettävissä useita jatkotutkimusaiheita. Eräänä jatkotutkimusaiheena olisi hyvä selvittää serverless-alustojen hinnan muodostuminen tietyt resurssit vaativan sovelluksen julkaisussa eri serverless-ympäristöissä ja kuormittamalla palveluita. Tämän tutkimuksen tulosten avulla voidaan yhdistää hintatutkimustieto serverless-alustojen ominaisuuksiin ja saada selkeä kuva kustannusten kehityksestä suhteessa alustojen ominaisuuksiin. Hintatietoa voitaisiin myös verrata Googlen palveluiden hintalaskureista saatuihin tuloksiin. Laajentamalla tutkimusta edelleen, selvitettäisiin

Google Cloudin käyttöä yrityksen kokonaisvaltaista pilvipalvelusiirtymää toteutettaessa, huomioiden pilvipalveluiden ominaisuudet ja valituista palveluista koituvat kustannukset.

Lähteet

f5. 2010. Controlling the Cloud: Requirements for Cloud Computing. F5 white paper. Viitattu 20.9.2022. Saatavissa https://www.f5.com/content/dam/f5/corp/global/pdf/white-papers/controlling-the-cloud-wp.pdf

Gartner. 2022. Gartner Says More Than Half of Enterprise IT Spending in Key Market Segments Will Shift to the Cloud by 2025. Viitattu 18.7.2022. Saatavissa https://www.gartner.com/en/newsroom/press-releases/2022-02-09-gartner-says-more-than-half-of-enterprise-it-spending

Google. 2021. Google Cloud. Serverless Architecture: What is Serverless? Viitattu 18.7.2022. Saatavissa https://cloud.google.com/serverless/whitepaper#serverless_for_application_development

Google. 2022a. Google Cloud. Dream, build, and transform with Google Cloud. Viitattu 1.8.2022. Saatavissa https://cloud.google.com/

Google. 2022b. Google palvelinkeskukset. Tutustu palvelinkeskusten sijainteihin. Viitattu 2.8.2022. Saatavissa https://www.google.com/about/datacenters/locations/

Google. 2022c. Google Cloud. Geography and regions. Viitattu 11.8.2022. Saatavissa https://cloud.google.com/docs/geography-and-regions

Google. 2022d. Google Cloud. Google Cloud overview. Viitattu 2.8.2022. Saatavissa https://cloud.google.com/docs/overview

Google. 2022e. Google Cloud. Choose an App Engine Environment. Viitattu 24.8.2022. Saatavissa https://cloud.google.com/appengine/docs/the-appengine-environments

Google. 2022f. Google Cloud. Autopilot overview. Viitattu 26.8.2022. Saatavissa https://cloud.google.com/kubernetes-engine/docs/concepts/autopilot-overview

Google. 2022g. Google Cloud. Load balancing and scaling. Viitattu 26.28.2022. Saatavissa https://cloud.google.com/compute/docs/load-balancing-and-autoscaling

Google. 2022h. Google Cloud. Autopilot Cluster Architecture. Viitattu 30.8.2022. Saatavissa https://cloud.google.com/kubernetes-engine/docs/concepts/autopilot-architecture

Google. 2022i. Google Cloud. What is Cloud Run. Viitattu 1.9.2022. Saatavissa https://cloud.google.com/run/docs/overview/what-is-cloud-run

Google. 2022j. Google Cloud. Deploying from source code. Viitattu 1.9.2022. Saatavissa https://cloud.google.com/run/docs/deploying-source-code

Google. 2022k. Google Cloud. Cloud Functions overview. Viitattu 8.9.2022. Saatavissa https://cloud.google.com/functions/docs/concepts/overview

Google. 2022l. Google Cloud. Cloud Functions version comparison. Viitattu 8.9.2022. Saatavissa https://cloud.google.com/functions/docs/concepts/version-comparison

Google 2022m. Google Cloud. Cloud Functions execution environment. Viitattu 8.9.2022. Saatavissa https://cloud.google.com/functions/docs/concepts/execution-environment

Google 2022n. Google Cloud. Cloud Functions triggers. Viitattu 8.9.2022. Saatavissa https://cloud.google.com/functions/docs/calling

Google 2022o. Firebase. Understand Firebase Projects. Viitattu 9.2.2023. Saatavissa https://firebase.google.com/docs/projects/learn-more?authuser=0&hl=en

Google 2022p. Firebase. Firebase Management API. Viitattu 9.2.2023. Saatavissa https://firebase.google.com/docs/reference/firebase-management/rest

Google. 2022q. Google Cloud. Meet our network. Viitattu 15.8.2022. Saatavissa https://cloud.google.com/about/locations#network

Google 2022r. Firebase. Make your app best it can be. Viitattu 8.2.2023. Saatavissa https://firebase.google.com/?gclid=Cj0KCQiAi8KfBhCuARIsADp-A55UKKlR-bRP_iYu01S08V3FgALAtUhj_uEDZwdghAmwj3pGtjhH3nm8aArZ5EALw_wcB&gclsrc=aw.ds

Google 2022s. Cloud Architecture Center. Google Cloud Architecture Framework. Viitattu 20.2.2023. Saatavissa https://cloud.google.com/architecture/framework

Google 2022t. Cloud Architecture Center. Core Principles of system design. Viitattu 20.2.2023. Saatavissa https://cloud.google.com/architecture/framework/system-design/principles

Google 2022u. Google Cloud. Functions Framework. Viitattu 1.4.2023. Saatavissa https://cloud.google.com/functions/docs/functions-framework

Google 2022v. Google Cloud. App hosting on Google Cloud. Viitattu 10.4.2023. Saatavissa https://cloud.google.com/hosting-options

Google. 2022x. Google Cloud. Horizontal Pod autoscaling. Viitattu 5.9.2022. Saatavissa https://cloud.google.com/kubernetes-engine/docs/concepts/horizontalpodautoscaler

Google. 2022y. Google Cloud. Cloud Architecture Center. Viitattu 14.4.2023. Saatavissa https://cloud.google.com/architecture/scalable-and-resilient-apps

Harkut, D. 2018. Cloud Computing. Technology and Practices. Viitattu 21.2.2023. Saatavissa https://www.intechopen.com/chapters/63602

Hildebrand, D. & Sereneyi, D. 2021. Google. Colossus under the hood: a peek into Google's scalable storage system. Viitattu 17.8.2022. Saatavissa https://cloud.google.com/blog/products/storage-data-transfer/a-peek-behind-colossus-googles-file-system

Mell, P & Grance, T. 2011. The NIST Definition of Cloud Computing. Special Publication 800-145. Viitattu 20.2.2023. Saatavissa DOI https://doi.org/10.6028/NIST.SP.800-145

Mukherjee, B., Tomkos, I., Tornatore, M., Winzer, P. & Zhao Y. 2020. Springer Handbook of Optical Networks. E-kirja. Cham: Springer Nature Switzerland AG. Google Play Books.

Rose, R. 2020. Hands-On Serverless Computing with Google Cloud. E-kirja. Birmingham: Packt Publishing Ltd. Google Play Books.

Sangapu, S., Panyam, D. & Marston, J. 2021. The Definitive Guide to Modernizing Applications on Google Cloud. E-kirja. Birmingham: Packt Publishing Ltd. Google Play Books.

Singh, A., Ong, J., Agarwal, A., Anderson, G., Armistead, A., Bannon, R., Boving, S., Desai, G., Felderman, B., Germano, P., Kanagala, A., Liu, H., Provost, J., Simmons, J.,Tanda, E., Wanderer, J., Hölzle, U., Stuart, S. & Vahdat, A. 2016. Jupiter rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network. 6-7. Viitattu 15.8.2022. Saatavissa DOI 10.1145/2975159

Sullivan, D. 2019. Official Google Cloud Certified Associate Cloud Engineer Study Guide. E-kirja. Indianapolis: John Wiley & Sons Inc. Google Play Books.

Treat, T. 2019. Serverless on GCP: A comprehensive Guide. DZone. Viitattu 24.8.2022. Saatavissa https://dzone.com/articles/serverless-on-gcp

Vahdat, A. & Koley, B. 2017. Google Cloud. Espresso makes Google cloud faster, more available and cost effective by extending SDN to the public internet. Google Cloud -blogi. Viitattu 18.8.2022. Saatavissa https://blog.google/products/google-cloud/making-google-cloud-faster-more-available-and-cost-effective-extending-sdn-public-internet-espresso/

Vahdat, A. 2022. Google Cloud. A look inside Google's Data Center Networks. Google Cloud -blogi. Viitattu 14.8.2022. Saatavissa https://cloud.google.com/blog/products/gcp/a-look-inside-googles-data-center-networks.

Venema, W. 2021. Building Serverless Applications with Google Cloud Run. E-kirja. Sebastopol: O'Reilly Media Inc. Google Play Books.

Vergadia, P. 2021a. Google Cloud Networking overview. Google Cloud -blogi. Viitattu 15.8.2022. Saatavissa https://cloud.google.com/blog/topics/developers-practitioners/google-cloud-networking-overview

Vergadia, P. 2021b. What is Cloud Spanner? Google Cloud -blogi. Viitattu 17.8.2022. Saatavissa https://cloud.google.com/blog/topics/developers-practitioners/what-cloud-spanner

Vergadia, P. 2022. Visualizing Google Cloud. E-kirja. Hoboken: John Wiley & Sons. Google Play Books.

Verma, A., Pedrosa, L., Korupolu, M., Oppenheimer, D., Tune, E. & Wilkes, J. 2015. Large-scale cluster management at Google with Borg. Viitattu 19.8.2022. Saatavissa DOI https://doi.org/10.1145/2741948.2741964

Wheatley, M. 2021. siliconANGLE. Google Kubernetes Engine goes serverless with new Autopilot Mode. Viitattu 26.8.2022. Saatavissa https://siliconangle.com/2021/02/24/google-kubernetes-engine-goes-serverless-new-autopilot-mode/