BIJLAGE SYSTEMEN OP HOOFDLIJNEN

Binnen de scope en de bijbehorende opties vallen drie systemen die momenteel volledig onafhankelijk van elkaar functioneren met uitzondering van data-uitwisseling. Hieronder worden de drie systemen op hoofdlijnen geïntroduceerd ten behoeve van de marktconsultatie.

1. DRIS

1.1. Functie

Het DRIS-systeem is het centrale systeem van OV-data. In het DRIS-systeem worden alle geplande en actuele reisgegevens van alle stad- en streekvervoerders ontvangen. Het systeem integreert deze tot de complete haltegebaseerde datastroom ten behoeve van DRIS-borden op straat en andere afnemers.

Daarnaast faciliteert met systeem het beheren en monitoren van de dataketen van de vervoerder tot en met de DRIS-borden op straat door middel van een beheerinterface, en worden diverse externe rapportages gefaciliteerd.

1.2. Koppelvlakken

Input

- NeTEx (planning)
- KV6 (actuele voertuigberichten)
- KV15 (vrij teksten)
- KV17 (mutaties)
- SIRI-VM/ET (GVB-variant)

Output

- KV7 (planning op halteniveau)
- KV8 (actualiteit op halteniveau)
- KV7/8turbo (Non-XML, geschikt voor grote hoeveelheden haltes)
- Data t.b.v. rapportages (realisatie en kwaliteitsindicatoren)
- Systeemloggings

1.3. Architectuur

De applicatie is ontwikkeld in JAVA uitgevoerd in de vorm van meerdere microservices. Deze microservices stellen veelal API’s beschikbaar, en kunnen op verschillende manieren aan het werk worden gezet, afhankelijk van de functie die de microservice binnen de applicatie uitvoert.

De microservices zijn in principe stateless geïmplementeerd. Dit houdt in dat er geen toestand wordt bijgehouden in de services zelf. In plaats daarvan is de toestand gevat in de call naar de service, of wordt deze opgehaald uit (calls naar) andere microservices (die op zichzelf overigens weer stateless zijn).

1.4. Softwarecomponenten

Ubuntu Server

- Ubuntu Server minimal dient als basis voor het cluster.

etcd

- etcd is een gedistribueerde key-value store die wordt gebruikt binnen een Kubernetes cluster als opslag voor de status van het cluster. Zo worden alle resources en hun revisies erin opgeslagen.

Calico

- Calico verzorgt de netwerkverbindingen binnen het Kubernetes cluster.

Kubernetes

- Kubernetes is het container management systeem.

Metrics server

- De Metrics Server wordt gebruikt om de Metrics API beschikbaar te maken binnen een Kubernetes cluster. Deze API kan gebruikt worden om resource-gebruik binnen het cluster op te vragen.

Docker Private Registry

- Docker Registry wordt gebruikt als opslagplaats voor docker images van klanten en ACC ICT die worden gebruikt binnen een Kubernetes cluster.

Monitoring

- Prometheus wordt gebruikt als loggingfaciliteit voor metrics van nodes binnen Kubernetes clusters. Een Prometheus installatie op Kubernetes bestaat uit Prometheus, Grafana en MySQL.

Grafana

- Grafana wordt ingezet als webfrontend voor de metrics. Standaard levert onze blueprint de Prometheus datasource en twee dashboards mee voor Kubernetes en Docker monitoring.

MySQL

- MySQL is voor storage van Grafana en sessiebeheer.

Prometheus

- Prometheus verzamelt de metrics van de in Kubernetes ingebouwde metricsserver.

Graylog

- Componenten die worden gebruikt voor logaggregatie van containers die op het Kubernetes cluster draaien. Een Graylog installatie op Kubernetes bestaat uit Elasticsearch, Filebeat, Journalbeat, MongoDB en Graylog.

MetalLB

- Kubernetes heeft standaard geen ingebouwde loadbalancer en gaat ervan uit dat deze door Cloud providers beschikbaar wordt gesteld. MetalLB implementeert een Layer 2 en 3 (BGP) LoadBalancer voor on-premise omgevingen. Het maakt het mogelijk om bij services het type LoadBalancer op te geven.

PostgreSQL

- Postgres draait in een cluster met twee database nodes en een arbiter voor quorum. Met de repmgr software doen we fail-overs. We gebruiken keepalived met een custom checkscript voor het beheer van het adres waar de clients op verbinden.
- Alle drie de servers uit het cluster draaien op een andere locatie en zijn dus geografisch gescheiden.

Failover

- Repmgr monitored en managed de failover. Daarna volgt het IP adres door keepalived met behulp van onderstaand checkscript.

1.5. Hosting

De applicatie draait in een gevirtualiseerde omgeving, waarbij zowel voor de productie- als acceptatie-omgeving twee instanties draaien en op twee locaties. Toegang doormiddel van VPN gefaciliteerd.

Enkele kengetallen ten aanzien van de hosting:

- Omvang hosting: 72 vCPU, 552 GB RAM, 4000 GB Enterprise storage
- Actueel datavolume internet in: 95ste percentiel 11 Mbit/s, max 45 Mbit/s
- Actueel datavolume internet uit: 95ste percentiel 7 Mbit/s, max 12 Mbit/s
- Beschikbare bandbreedte LAN: 1 Gbit/s
- Actuele beschikbaarheid omgeving: 99.99% (24/7)

2. CDD

2.1. Functie

De Centrale Distributieserver DRIS (CDD) beoogt het aansturen van DRIS-borden op straat in technische zin efficiënter te maken dan mogelijk is met de bestaande koppelvlakken 7 en 8. Deze efficiëntie wordt bereikt doordat DRIS-borden zelf het ophalen van informatie initiëren (dus minder beheer) en doordat er sprake is van een uniform beheerkoppelvlak wat uniform (en daarmee efficiënt) beheer mogelijk maakt.

Momenteel wordt CDD gevoed vanuit de DRIS-server op basis van koppelvlak 7 en 8. Op termijn zal dit waarschijnlijk koppelvlak 7/8turbo worden, of een systeemkoppeling.

2.2. Koppelvlakken

Input

- KV7 (planning op halteniveau)
- KV8 (actualiteit op halteniveau)
- PSA-bestand (via NDOV-loket)

Output

- Open DRIS-koppelvlak(ken): aanmelden, berichten en beheer

2.3. Architectuur

Het CDD maakt, net als de DRIS-server, gebruik van verschillende services. Iedere service voert een specifieke taak uit. Communicatie tussen deze services vindt plaats met behulp van berichten(events) die via een MQTT-broker uitgewisseld worden tussen de verschillende services. Services genereren als output een bericht op een topic, die vervolgens door alle processen die op het topic zijn geabonneerd verder worden afgehandeld.

2.4. Softwarecomponenten

Het raamwerk van de CDD-applicatie is een ander raamwerk dan dat van de DRIS-server, en is gebouwd op het .Net Core 2.2 framework. Voor de berichtcommunicatie tussen de verschillende services wordt de MQTT-broker versie 4.1.1 van EMX-Q gebruikt. De database is MariaDB versie 10.4. De web interfaces worden aangeboden via Apache 2.4.41. Dit alles wordt gehost op virtuele linux servers met als operating systeem Ubuntu 18.04 (of gelijksoortig.). Microsoft Visual Studio 2019 wordt gebruikt om de applicatie te ontwikkelen.

Gebruikte (externe) libraries zijn:

- Log4Net, open source logging framework (Apache license 2.0)
- MQTTnet (MIT License)
- Microsoft.Extentions (Apache License, Version 2.0)
- Quartz (Apache License, Version 2.0)

2.5. Hosting

De applicatie draait in een gevirtualiseerde omgeving, waarbij zowel voor de productie- als acceptatie-omgeving twee instanties draaien en op twee locaties. Toegang doormiddel van VPN gefaciliteerd.

Enkele kengetallen ten aanzien van de hosting:

- Omvang hosting: 56 vCPU, 264 GB RAM, 1120 GB Enterprise storage
- Beschikbare internet bandbreedte: 25Mbit/s (synchroon)
- Actuele beschikbaarheid omgeving: 99.99% (24/7)

3. Ladekast

3.1. Functie

Een van de hoofdfuncties van OV-Data is het ontvangen, integreren/creëren en distribueren van reisinformatiegerelateerde datastromen en datasets. Veel van deze datastromen worden gebruikt in meerdere systemen en datasets worden opgeslagen voor toekomstig gebruik. De ladekast is een opslagsysteem waarbij iedere dataset (een “la”) door afnemende systemen en afnemers kan worden geraadpleegd.

De gegevens worden beschikbaar gesteld via en web interface en API, daarnaast is er ook nog een FTP-omgeving voor de distributie van operationele datasets.

3.2. Datasets

Leveranciers leveren datasets via https aan. Dataset bestaan uit: xml files, json files, binaire files, etc. Inhoudelijk gaat het om:

- Koppelvlak 7 Turbo calendar
- Koppelvlak 7 Turbo planning
- KV8turbo_passtimes
- KV8turbo_generalmessages
- Userstop lastseen
- Concessie
- Concessie shapefile
- Lijnen
- Lijnen shapefile
- Zones
- Haltes
- PSA (CSV en XML)
- Meetboek
- Bezetting
- Incidenten

3.3. Architectuur

De basis is input via kanalen > centrale opslag > output via kanalen.

- Input kanalen in principe via https achter een firewall en TLS . Paar aanleverkanalen achter een VPN over http,
- Centrale server is op basis van een EBS die de verschillende api’s serviced. Database is een gerepliceerde mongoDB.
- Output over https, 2FA voor de front-end en TLS voor system 2 system toegang.

3.4. Softwarecomponenten

- Back-end: Mule EBS, Java componenten
- Front-end: AngularJs, NgNix web server.

3.5. Hosting

De ladekast draait in een gevirtualiseerde omgeving, en kent een productiesysteem en een back-upsysteem. Toegang doormiddel van VPN gefaciliteerd.

De FTP-omgeving draait in een gevirtualiseerde omgeving. Toegang doormiddel van VPN gefaciliteerd.

Enkele kengetallen ten aanzien van de hosting:

Omvang hosting ladekast: 16 vCPU, 82 GB RAM, 3 TB Storage
Omvang hosting FTP-omgeving: 2 vCPU, 2 GB RAM, 120 GB Enterprise storage
Totaal actueel uploadvolume: ~ 5 GB per dag upload
Totaal actueel downloadvolume: < 500MB per dag
Verwachting groei: upload 100%, download 1000%
Actuele beschikbaarheid omgeving: 99.99% (24/7)