3e Bachelor Informatica
Academiejaar 2015-2016

Faculteit Ingenieurswetenschappen en Architectuur
Valentin Vaerwijckweg 1 - 9000 Gent

Verkeerscentrum

Verslag voor bachelorproef (sprint 2)

Groep 2 Mike BRANTS
Tobias VAN DER PULST
Thomas VANDE WEGHE
Simon VERMEERSCH

Inhoudsopgave

Hoofdstuk 1: Behoefteanalyse

Het doel van dit project is het opzetten van een database voor de verkeersgegevens van bepaalde trajecten in Gent. In de eerste plaats worden gegevens opgehaald van bekende verkeersinformatie-databronnen. Daarna zal de opgehaalde data in een database opgeslagen worden. De bedoeling is om op deze manier de informatie van verschillende databronnen kwalitatief met elkaar te vergelijken.

1.1 Beschrijving project

Het Mobiliteitsbedrijf van de stad Gent is sinds 2014 bezig met het opzetten van een regionaal verkeerscentrum. Het is de bedoeling dat op termijn het verkeer in de regio constant gemonitord wordt, op semi-automatische basis op normale werkdagen en bemand tijdens piekmomenten en evenementen. Tijdens de week is het de bedoeling dat onverwachte incidenten, calamiteiten of significante verhogingen van de reistijden automatisch gesignaleerd worden aan de verantwoordelijke, die dan de nodige acties kan ondernemen. De gegevens zouden ook constant beschikbaar zijn voor het publiek via een website, sociale media en open data. Op die manier kunnen mensen de beste route en het beste moment kiezen om hun verplaatsingen te maken in de regio.

1.2 Functionaliteiten

1.2.1 Basis

1. Ophalen van kwalitatieve en vergelijkbare data bij verschillende bronnen
2. Opgehaalde data opslaan in database

1.2.2 Extra

1. Real-time overzicht van de verkeersdrukte op vooraf vastgelegde trajecten
2. Analyse en kwaliteitscontrole op basis van opgehaalde data
3. Aanbieden van gegevens aan externen via REST API
4. Platform gelinkt met sociale media om snelle communicatie aan te bieden
5. Meldingen genereren wanneer reistijden overschreden worden
6. Bepalen van de oorzaak van een vertraging

1.3 Use Case

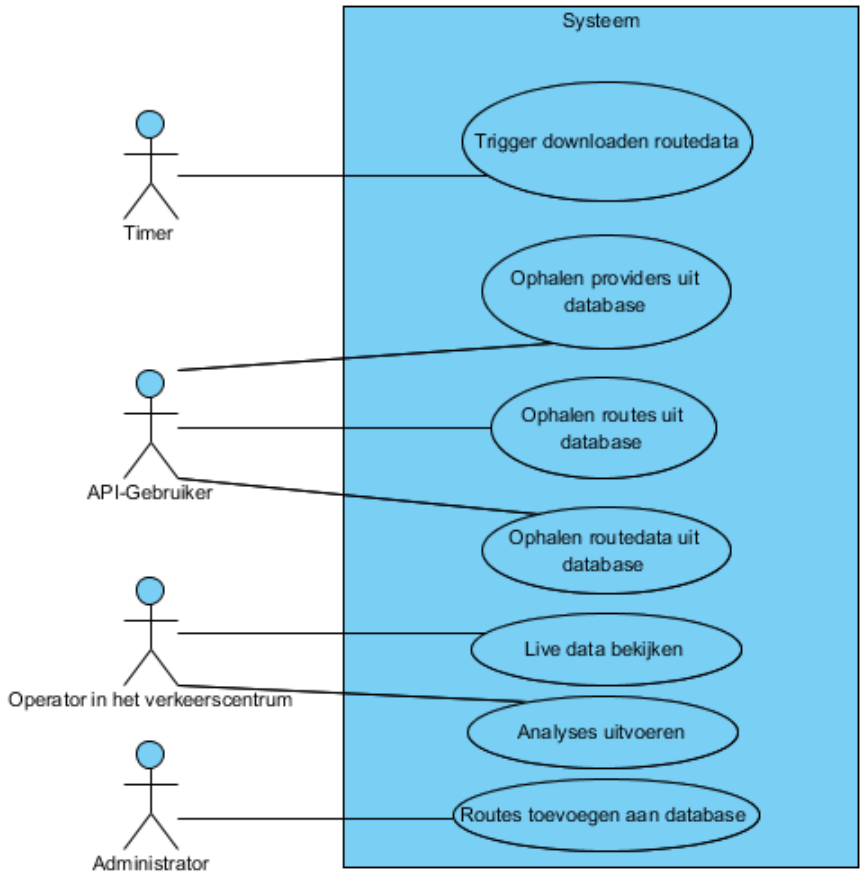

Diagram 1 Algemeen

Er zijn vier actoren aan het werk. In de eerste plaats is er een timer, deze zal een trigger sturen naar het programma zodat data afkomstig van de verschillende providers opgehaald wordt. Verder is er nog een API-gebruiker, dit is een persoon die data kan ophalen uit de database gebruikmakend van de REST API. Een andere actor is de operator in het verkeerscentrum, hij zal via het dashboard de verkeersdata monitoren en de mogelijkheid hebben om analyses uit te voeren. Als laatste is er nog een administrator, hij kan routes toevoegen aan de database zodat ook van deze routes data worden opgehaald.

Website

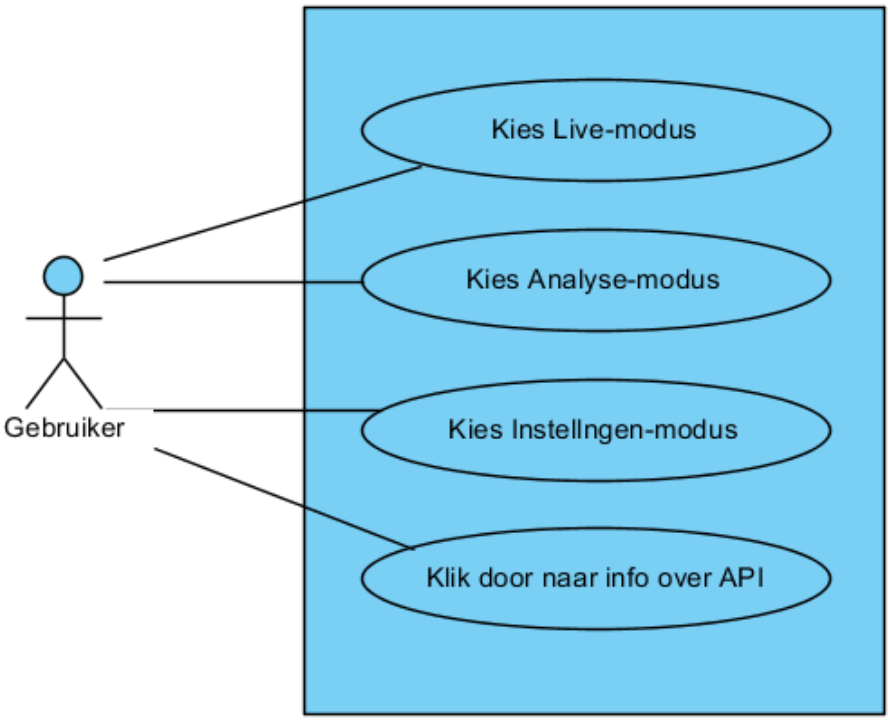

Diagram 2 Index van site

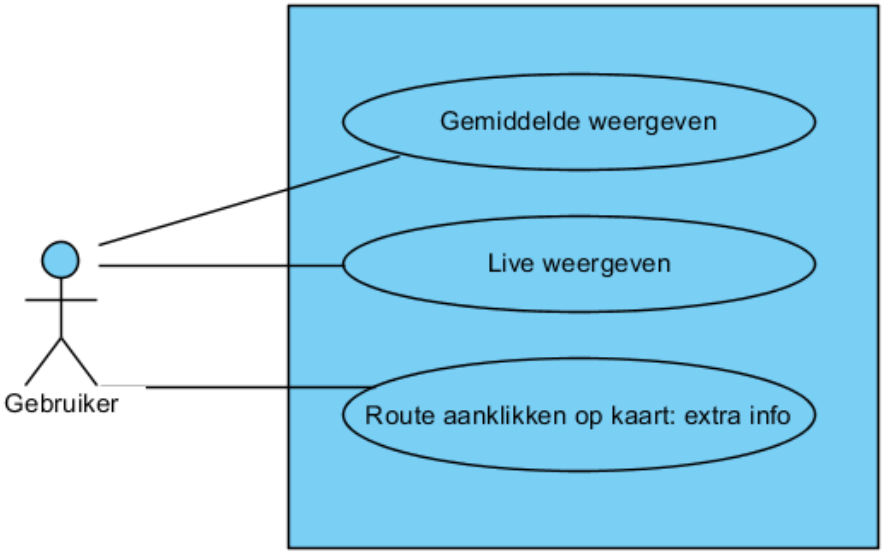

Diagram 3 Live-pagina

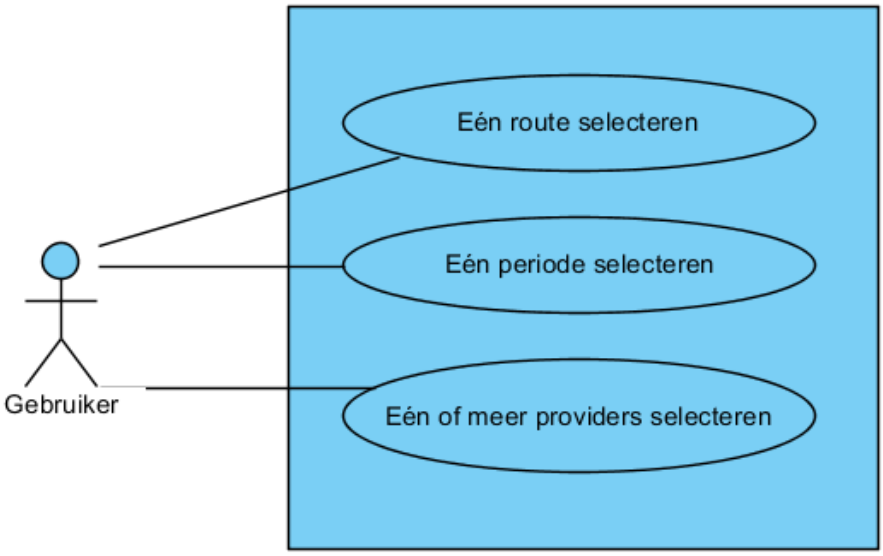

Diagram 4 Algemene analyse

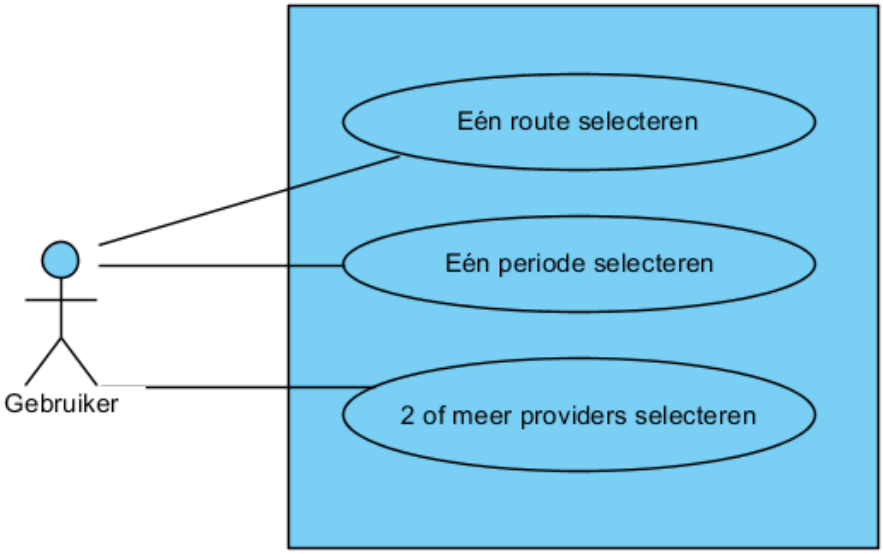

Diagram 5 Databronnen vergelijken

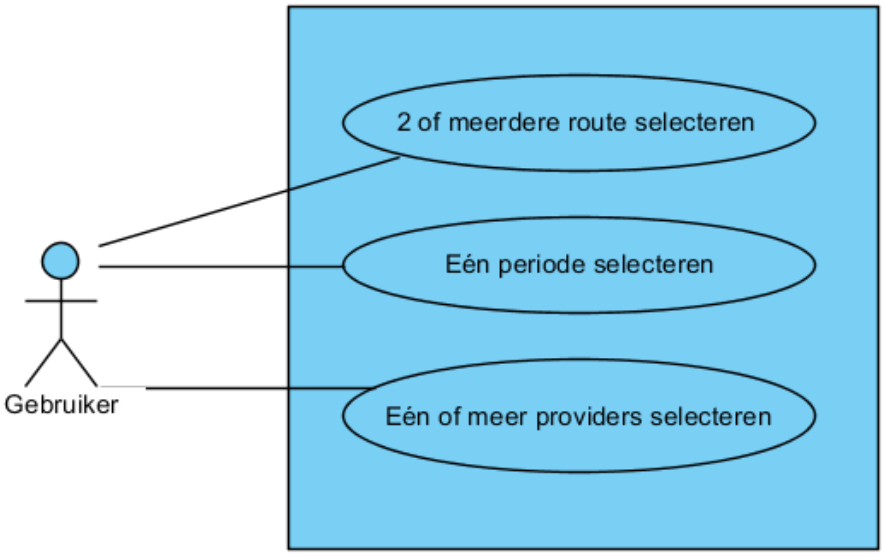

Diagram 6 Routes vergelijken

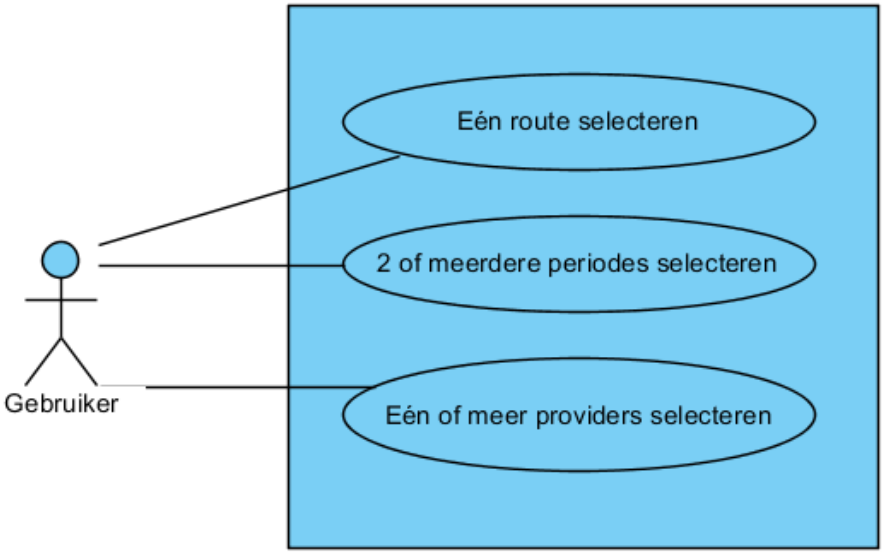

Diagram 7 Periodes vergelijken

1.4 Functieanalyse van de omgeving

1. Gebruikers
   (a) Ontwikkelaar
   (b) Administrator
   (c) Operator in het verkeerscentrum
2. Doelstellingen
   De doelstellingen representeren de product backlog en ze bevatten de taken die het systeem moet kunnen.
   (a) Basisfunctionaliteit
      i. Data ophalen uit meerdere bronnen
         A. Google Maps
         B. Here
         C. Waze
         D. TomTom
         E. Coyote
      ii. Databank creëren en opvullen met opgehaalde data
   (b) API met verschillende parameters
      i. Periode
      ii. Traject
      iii. Databron
      iv. Vertraging
   (c) Dashboard voor analyse van de verkeerssituaties
      i. Grafische opbouw van de GUI
      ii. Grafieken/Tabellen genereren
      iii. Grafische weergave op kaart
      iv. Ophalen data aan de hand van API
      v. Kwaliteitscontrole van de verschillende databronnen

Hoofdstuk 2: Ontwerp

2.1 Functioneel ontwerp

Er zijn drie types gebruikers in het systeem. Enerzijds zijn er de API-gebruikers, zij hebben de mogelijkheid om data uit de API op te vragen en eventueel verder te verwerken. Anderzijds zijn er de operatoren in het verkeerscentrum, zij hebben de mogelijkheid om via het dashboard live data te monitoren en analyses uit te voeren. Om af te sluiten is er een administrator, deze gebruiker kan routes toevoegen aan het systeem.

2.2 Technisch ontwerp

2.2.1 Hardware

De gevraagde applicatie is geprogrammeerd in Java, dit laat toe om op alle besturingssystemen te draaien zolang deze Java ondersteunen. Er zijn dan ook geen eisen aan de hardware voor deze applicatie, enkel softwarepakketten zijn vereist voor het draaien van deze software. Er is steeds een actieve internetverbinding vereist om de data van de databronnen te ontvangen.

2.2.2 Pakketten

Dit project wordt uitgewerkt in Java met behulp van het Java EE (Enterprise Edition) *framework*. Dit *framework* omvat verschillende technologieën die worden gebruikt in deze applicatie.

1. JavaServer Faces (JSF) 2.2
2. Context and Dependency Injection for Java 1.1
3. Enterprise JavaBeans (EJB) 3.2
4. Java Persistence API (JPA) 2.1
5. Java Transaction API (JTA) 1.2
6. Java API for RESTful Web Services (JAX-RS) 2.0

Dit wordt aangevuld met Glassfish voorzieningen.

1. Java Naming and Directory Interface (JNDI)
2. Java Database Connection (JDBC) connection pools

Hieronder zal elk deel uitgebreid behandeld worden.

JSF

Deze technologie laat toe webpagina's te genereren op basis van data in de applicatie. Deze data worden aangeleverd via *backing beans*. Over de jaren is JSF uitgebreid met AJAX ondersteuning alsook luisteraars bij datawijzigingen. Dit alles laat toe een zeer flexibele, *responsive* interface te maken voor de gebruikers.

Contexts and Dependency Injection

Alle beans verwachten zekere diensten waarvan zijzelf ook afhankelijk zijn. Zo wordt bij de Data Access Beans (DAO) gebruik gemaakt van de EntityManager voor interactie met de databank. Bij de TrafficDataDownloader wordt dan weer de context van de applicatieserver verwacht om andere beans op te vragen. Al deze diensten zijn niet de verantwoordelijkheid van deze individuele beans maar van de applicatieserver. Deze laatste gedraagt zich als *injector* en zal alle vereiste *services* (*dependencies*) injecteren in de beans aan de hand van annotaties en objecttypes.

EJB

JavaBeans zijn door software beheerde modulaire bouwblokken. In deze beans wordt de *business logic* voor een Enterprise Applicatie verwerkt. De grootste kenmerken van deze beans zijn hun modulariteit, onafhankelijkheid van elkaar en schaalbaarheid.

Modulariteit
Iedere module (EJB) in het project is uitwisselbaar met een nieuwe module. Dit is aan te passen in een extern *properties*-bestand. Zo kan op ieder moment bijvoorbeeld een databron worden toegevoegd, een database worden vervangen door een andere of een nieuwe *web service* worden toegevoegd.

Onafhankelijkheid
Alle beans zijn onafhankelijk. Alle objecten die voorkomen in meerdere beans (zoals interfaces) zijn gebundeld in een gemeenschappelijke bibliotheek. Hierdoor zullen beans niets merken wanneer een andere bean wijzigt.

Uitbreidbaarheid
De modules zijn niet enkel onafhankelijk, maar hebben eveneens geen vaste relatie met de locatie waar ze werken. Zo kan een *database-bean* op een andere server staan dan de *analyser-bean*. De enige vereiste hiervoor is dat JNDI van de ene server gelinkt is aan de JNDI van de andere server. De beans zullen hun parameters en teruggeefwaardes steeds serialiseren en doorsturen naar de zogenaamde *remote bean*.

JPA

De Java variant voor Object Relational Mapping (ORM) laat toe de gegevens in een databank rechtstreeks af te beelden op objecten door middel van annotaties. Deze manier van interactie met de databank laat een zeer eenvoudige werking toe. Het zal echter niet de performantie van handmatige SQL-commando's evenaren.

JTA

Deze API start (zonder enige configuratie) steeds een transactie bij het aanroepen van een functie in een *managed bean*. Indien die functie een fout zou opwerpen zal een *rollback* gebeuren tot de toestand vlak voor de aanroep is bereikt. In deze applicatie wordt op deze API vertrouwd voor opslag van gegevens in de databank. Bij een *error* zal de opdracht voor dat interval niet worden uitgevoerd, maar zal de applicatie wel blijven werken.

JAX-RS

Deze API laat toe om services aan te bieden volgens het Representational State Transfer (REST) patroon. In deze applicatie wordt het gebruikt om de API uit te werken.

JNDI

Deze technologie laat toe data of objecten op te vragen via hun naam. Voor dit project werd de link naar de bronbestanden, de link naar de JDBC Connection Pool en de link naar alle beans opgenomen in JNDI.

JDBC Connection Pools

Een Connection Pool houdt een *cache* van connecties naar een welbepaalde databank bij en maakt deze beschikbaar aan de applicaties van de applicatieserver. Dit alles zorgt voor een hogere efficiëntie want de connecties worden behouden en herbruikt. Hiernaast wordt er ook een hogere veiligheid aangeboden, de connectieparameters zijn namelijk niet langer in de applicatie zelf aanwezig.

2.3 Software ontwerp

2.3.1 Databronnen

Google Maps

URL: https://developers.google.com/maps/documentation/distance-matrix/

Reistijden van Google Maps kunnen opgevraagd worden via de Google Maps Distance Matrix API. In de URL kunnen verschillende start- en eindpunten worden meegegeven. Er moet rekening gehouden worden dat er voor elke combinatie van start- en eindpunt een reistijd wordt opgenomen in het antwoord. Bij het opgeven van drie startpunten en drie eindpunten, zal het resultaat een 3X3-matrix zijn. Dit komt overeen met negen aanvragen. Er kunnen maximaal 10 start- en eindpunten worden opgegeven in één aanvraag.

Om aanvragen te doen naar de API, is er een unieke toegangssleutel nodig, die kan aangevraagd worden bij Google. Met de gratis sleutel zijn er 2500 aanvragen per dag mogelijk. Indien deze limiet overschreden is, wordt er 0,5 dollar (= 0,4548 euro) per 1000 extra elementen aangerekend. In de huidige situatie waarbij er van 34 routes elke vijf minuten data wordt opgevraagd en dit 18u per dag, is er nood aan 7300 aanvragen per dag. Indien er rekening gehouden wordt met extra tussenpunten zal het aantal aanvragen sterk oplopen. Het gratis model van Google zal niet volstaan. Er kan worden overgeschakeld op het Google Maps API's Premium Plan zodat er tot 100 000 aanvragen per dag gedaan kunnen worden.

Here

URL: https://developer.here.com/rest-apis/documentation/routing

Here stelt reistijden ter beschikking via zijn Routing API. In de URL kan je een route meegeven door de coördinaten in te stellen voor start- en eindpunt. Ook eventuele tussenpunten kunnen worden vermeld. Verder moet voor de toepassing die hier ontworpen wordt steeds aangegeven worden dat men de kortste route wil en dat men informatie baseert op het huidige verkeer. Op deze manier zal steeds actuele verkeersinformatie over een vaste route worden teruggeven.

Om de API van Here te kunnen gebruiken moeten er twee sleutels aangevraagd worden. De eerste 90 dagen kan dit gratis en mogen er tot 100 000 aanvragen per maand gedaan worden. In de huidige situatie waarbij er voor 34 routes elke vijf minuten data wordt opgevraagd en dit 18u per dag, is er nood aan 220 000 aanvragen per maand. Dit betekent dat er sowieso een betalende formule nodig is. Om tot 275 000 aanvragen per maand te kunnen doen, moet er gekozen worden voor het standaardplan dat 99 euro per maand kost.

TomTom

URL: http://developer.tomtom.com/products/onlinenavigation/onlinerouting

TomTom heeft een API waarbij je de verkeersinformatie van de routes kan opvragen waarna je een JSON-object terugkrijgt. Het gebruikte account is een evaluatieversie waarbij er een aantal beperkingen zijn. Eén van de beperkingen is dat er slechts vijf aanvragen per seconde kunnen gedaan worden en het is ook zo dat er op een hele dag in totaal slechts 1000 aanvragen mogen gedaan worden. Er zal sowieso moeten overgeschakeld worden naar een betalend plan, info daarover is niet te vinden op de site van TomTom. Er zal dus contact opgenomen moeten worden met de provider.

Coyote

URL: https://maps.coyotesystems.com/traffic/

Coyote stelt geen API ter beschikking, alle data wordt verkregen door de website te scrapen. Er zijn bijgevolg geen beperkingen voor het aantal mogelijke aanvragen. Enige voorwaarde is dat er een account ter beschikking is met de juiste routes op.

2.3.2 Structuur van de data (API)

URL: http://docs.verkeerscentrumgent.apiary.io/

De REST API geeft info terug uit de database, hier heb je de keuze uit drie opties: Routes, RouteData en Providers. Wanneer je hier aanvragen naar doet krijg je gegevens uit de database terug. In de komende secties wordt een korte toelichting gegeven per optie, gevolg door een voorbeeld van aanvraag en antwoord in JSON-formaat. **De versie in dit document is verouderd en onder constructie, voor recente informatie kan je gebruik maken van de URL.**

2.3.2.a Routes

De routes zijn de trajecten waar realtime data van opgeroepen wordt. Indien je geen parameters meegeeft zullen naam, id en tussenpunten teruggegeven worden. In de toekomst zal het ook mogelijk zijn om routes toe te voegen via de API.

Vraag

Patroon

GET http://verkeer-2.bp.tiwi.be/api/v2/routes/**{id}**

Parameter	Beschrijving
id	lijst van id's gescheiden door een komma, ook het woord *all* is toegestaan

Voorbeeld

GET http://verkeer-2.bp.tiwi.be/api/v2/routes/**1,2,3**

Antwoord

Voorbeeld


```
[
 {
    "name": "R4 Zelzate",
    "id": 1,
    "geolocations": [
    {
        "latitude": 51.192226,
        "name": "Zelzate",
        "longitude": 3.776342
    },
    {
        "latitude": 51.086447,
        "name": "Gent",
        "longitude": 3.672188
    }
   ]
 }
]
```

2.3.2.b RouteData

De RouteData is de opgehaalde data per route van de verschillende databronnen op verschillende tijdstippen. Je kan kiezen van welke routes je data wil terugkrijgen.

Data in een bepaald interval
De eerste mogelijkheid is om begin- en eindpunt van de periode in te stellen. Dit laatste is nuttig om data in een bepaald interval terug te krijgen.

Vraag

Patroon
GET http://verkeer-2.bp.tiwi.be/api/v2/routes/**{id}**/data/**{timeStart}**/**{timeEnd}**

Parameter	Beschrijving
id	lijst van id's gescheiden door een komma, ook het woord *all* is toegestaan
timeStart	begintijd van het interval waarvan je data wenst te ontvangen
timeEnd	eindtijd van het interval waarvan je data wenst te ontvangen, dit is optioneel

Voorbeeld
GET http://verkeer-2.bp.tiwi.be/api/v2/routes/**1,2,3**/data/**1456761535931**/**2456829774992**/

Antwoord

Voorbeeld


```
[
 {
    "data": [
      {
       "duration": 753,
       "distance": 14677,
       "provider": "GoogleMaps",
       "timestamp": "1456761535931"
      },
      {
       "duration": 681,
       "distance": 14685,
       "provider": "Here",
       "timestamp": "1456761535931"
      }
    ],
    "name": "R4 Zelzate",
    "id": 2,
    "geolocations": [
```

```
     {
        "latitude": 51.192226,
        "name": "Zelzate",
        "longitude": 3.776342
     },
     {
        "latitude": 51.086447,
        "name": "Gent",
        "longitude": 3.672188
     }
    ]
  }
 ]
```

Actuele informatie
Het is ook mogelijk om in plaats van twee tijdstippen enkel *current* op te geven, in dat geval wordt de actuele informatie getoond.

Vraag

Patroon

GET http://verkeer-2.bp.tiwi.be/api/v2/routes/**{id}**/data/current/

Parameter	Beschrijving
id	lijst van id's gescheiden door een komma, ook het woord *all* is toegestaan

Voorbeeld

GET http://verkeer-2.bp.tiwi.be/api/v2/routes/**1,2,3**/data/current/

Antwoord

Voorbeeld


```
[
 {
    "data": [
      {
       "duration": 753,
       "distance": 14677,
       "provider": "GoogleMaps",
       "timestamp": "1456761535931"
      },
      {
       "duration": 681,
       "distance": 14685,
       "provider": "Here",
       "timestamp": "1456761535931"
      }
    ],
```

```
    "name": "R4 Zelzate",
    "id": 2,
    "geolocations": [
     {
       "latitude": 51.192226,
       "name": "Zelzate",
       "longitude": 3.776342
     },
     {
       "latitude": 51.086447,
       "name": "Gent",
       "longitude": 3.672188
     }
    ]
  }
]
```

2.3.2.c Providers

Het is mogelijk om via de RESTAPI alle databronnen op te vragen, dit kan handig zijn om in bijvoorbeeld RouteData te gebruiken als filter.

Vraag

Patroon

GET http://verkeer-2.bp.tiwi.be/api/v2/providers

Voorbeeld

GET http://verkeer-2.bp.tiwi.be/api/v2/providers

Antwoord

Voorbeeld

```
[
 "Here",
 "GoogleMaps"
]
```

Filters

Er zijn 2 types filters die je kan meegeven op het einde van je URL: *fields* en *provider*. In *fields* kan je beslissen wat weergegeven moet worden in het antwoord en bij *provider* kan je ervoor kiezen om enkel data van bepaalde databronnen weer te geven.

Voorbeeld

?fields=route.name,route.id,route.geolocations&provider=GoogleMaps,Here

---

2.3.3 Databank

De database bestaat uit drie tabellen. RouteData is op termijn de grootste tabel, hierin worden alle opgehaalde gegevens bewaard. In Routes staan alle trajecten waarvan data zullen worden opgehaald. Deze trajecten bestaan uit GeoLocations die het traject bepalen.

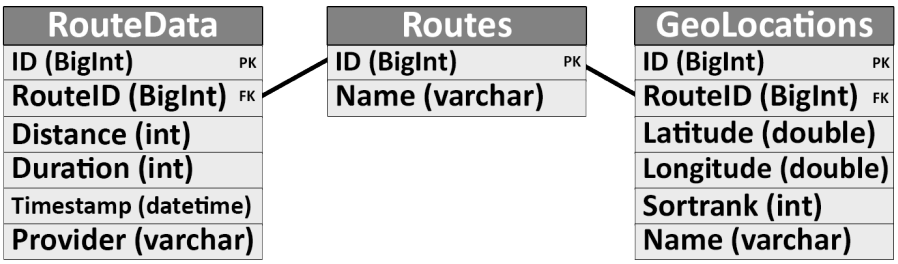

Diagram 8 Databank

2.3.4 Klassendiagram

In dit onderdeel vindt u drie klassendiagrammen, al kan je het in dit geval ook interfacediagrammen noemen. Om te beginnen is er een diagram voor de basiscomponenten, de meest elementaire klassen in het systeem. Deze klassen vormen ook de database. Hierna komt het diagram van het gegevensbeheer, deze bevat de samenwerking van klassen die data ophalen en verwerken. Als laatste, maar daarom niet minder belangrijk, vindt u de BeanFactory. Dit diagram bevat slechts één klasse die de klassen uit diagram 3 zal beheren.

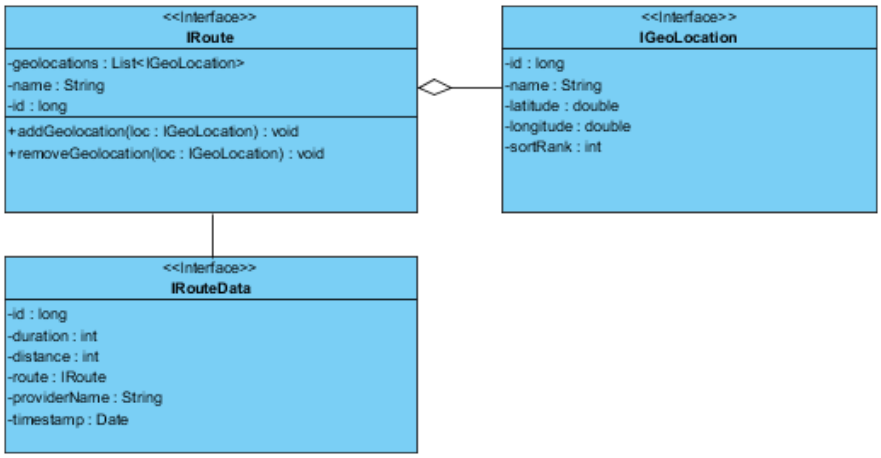

Diagram 9 Basiscomponenten/Database

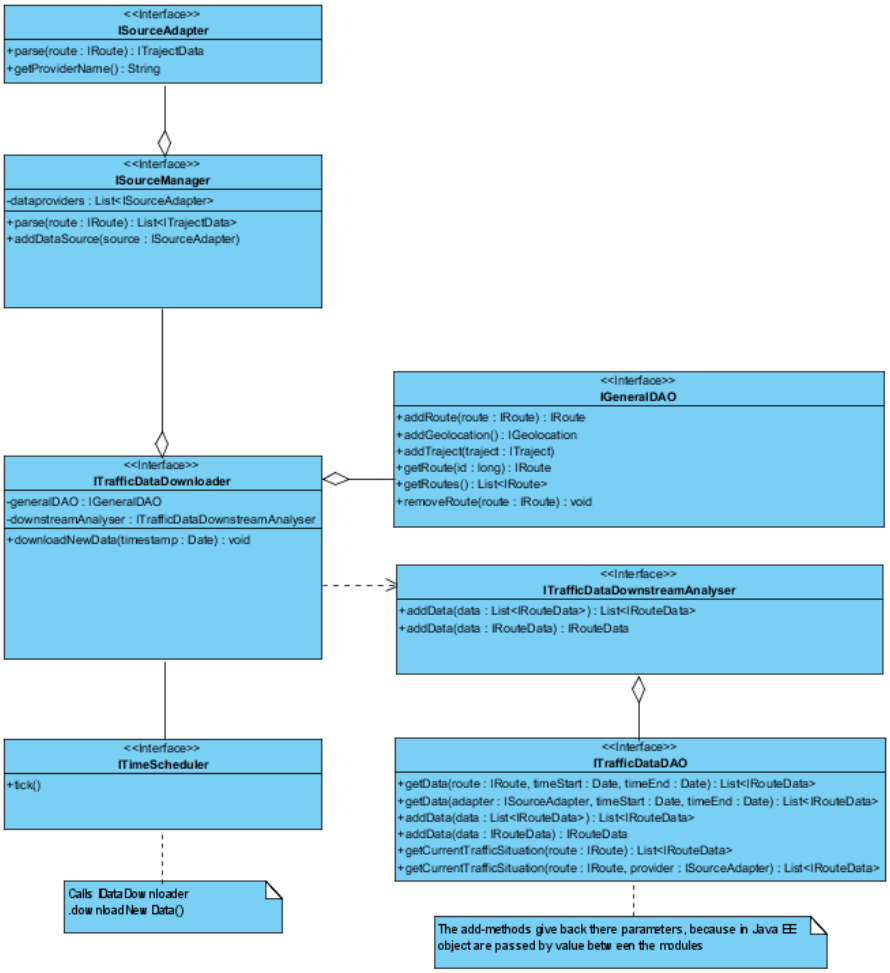

Diagram 10 Gegevensbeheer

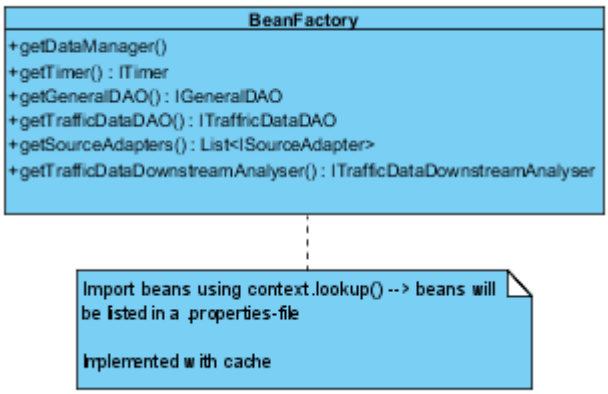

Diagram 11 BeanFactory

2.3.5 Verantwoordelijkheid per klasse

Klasse	Verantwoordelijkheid
Route	bevat informatie over een route
GeoLocation	bevat informatie over een locatie
RouteData	bevat verkeersinformatie van één route, één databron en dit op één bepaald moment in de tijd
[Object]Entity	deze klassen vertegenwoordigen de drie bovenstaande klassen zodat ze in de database kunnen opgeslagen worden
[Provider]SourceAdapter	omzetten van data, aangeboden door een provider, naar RouteData voor één Route
SourceManager	beheert alle adapters en zorgt ervoor dat de verschillende RouteData voor één Route bij elke adapter worden opgehaald
TimerScheduler	automatische triggering voor ophalen nieuwe data, het interval van deze timer is wijzigbaar op tijdstippen naar keuze
TrafficDataDownloader	beheren van alle routes en geef commando om RouteData op te halen door aan de SourceManager die per opgegeven route alle adapters zal aanspreken
TrafficDataDownstreamAnalyser	data afkomstig van SourceAdapters controleren op correctheid en nadien verdere acties ondernemen indien nodig
BeanFactory	deze klasse zal dependency injection vertegenwoordigen in alle klassen, deze klasse wordt altijd aangesproken om tussen beans te communiceren
GeneralDAO	verbinding tussen core en database, zorgt voor de opslag van Route en GeoLocation in de database
TrafficDataDAO	zorgt voor de opslag van RouteData in de database
DataProvider	data aan de RESTAPI doorgeven

2.3.6 Line Of Business

De volledige applicatie streeft naar de richtlijnen van een Line Of Business applicatie.

Flexibel en Uitbreidbaar

Door gebruik te maken van Java EE, waarin de gehele applicatie in verschillende modules wordt opgedeeld, kunnen nieuwe modules eenvoudig afzonderlijk worden gecreëerd en worden toegevoegd.

Onderhoudbaarheid

Dit analysedocument bevat alle nodige informatie over de klassen en hun onderlinge relaties na sprint 1. Op het einde van de ontwikkelingsperiode zal een documentatie worden voorzien met alle nodige informatie voor andere ontwikkelaars die de applicatie zouden willen wijzigen of uitbreiden.

Testbaarheid

De verschillende componenten werden getest gebruikmakend van unittests en integratietests.

Late Binding

Java EE biedt de mogelijkheid om een applicatie op te delen in verschillende modules die afzonderlijk van elkaar kunnen worden gecompileerd. Er werden twee DAO's voorzien zodat de algemene data over routes en de verkeersinformatie over de routes kunnen worden opgeslagen in twee verschillende databases. Zo zal na de ontwikkelingsperiode de verkeersinformatie waarschijnlijk worden opgeslagen in een NoSQL-database, omdat de hoeveelheid data enorm groot zal worden.

Parallelle ontwikkeling

Door opdeling in modules, die Java EE aanbiedt, kunnen programmeurs afzonderlijk van elkaar code implementeren.

Losse koppeling van objecten

Modules kunnen eenvoudig worden ontkoppeld en worden vervangen door een andere. De BeanFactory, die de module-objecten aanbiedt, wordt eenvoudig geconfigureerd in een propertiesbestand. Zo kan eenvoudig een nieuwe DAO worden toegevoegd door de link in het configuratiebestand te wijzigen naar een andere DAO. Zo kan bijvoorbeeld de manier van opslaan van data eenvoudig worden gewijzigd van een SQL-database naar een NoSQL-database.

Crosscutting concerns (Logging)

De manier van logging kan eenvoudig worden gewijzigd door het hierboven beschreven principe van losse koppeling. Voor logging werd ook een module voorzien die kan worden gewijzigd door de link aan te passen in het propertiesbestand bij de BeanFactory.

2.3.7 Gegevensstroomdiagram

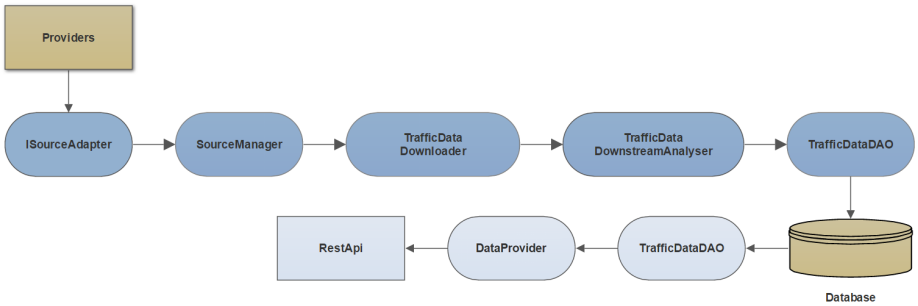

Diagram 12 Gegevensstroomdiagram

Downstream

De data van de verschillende databronnen worden opgehaald met behulp van de SourceAdapters, per databron bestaat er een adapterklasse die de ISourceAdapter interface implementeert. De verschillende adapters bevinden zich in de SourceManager-klasse. Vanuit de TrafficDataDownloader wordt data per route aangevraagd, de SourceManager zal deze aanvragen doorsturen aan elke adapter en de ontvangen data per adapter teruggeven aan de TrafficDataDownloader. Vervolgens passeren de data ook nog de TrafficDataDownstreamAnalyser die controleert of de data geldig zijn en eventueel meldingen genereert. Om af te sluiten worden de data doorgegeven aan de TrafficDataDAO, deze klasse zorgt ervoor dat de data in de database terechtkomen.

Upstream

De mogelijkheid bestaat om via API-aanvragen data uit de database te halen. In de eerste plaats gebeurt dit via de TrafficDataDAO die contact heeft met de database, via de DataProvider wordt een bepaalde aanvraag gedaan aan de TrafficDataDAO. De RestAPI zorgt er dan voor dat de *return*-waardes van de DataProvider in JSON-formaat kunnen worden opgeroepen.

2.3.8 Bestandsstructuur

Het project is opgedeeld in verschillende beans die hieronder worden behandeld.

Logger

Deze bean start tegelijk met de server en maakt logging naar een bestand (log.txt) mogelijk.

TimerScheduler

Deze bean start tegelijk met de server en stuurt triggers volgens het patroon dat opgegeven wordt aan de bean in een *properties*-bestand. De triggers roepen op om data te downloaden. Deze timer wordt gecreëerd via Java EE TimerServices.

TrafficDataDownloader

De downloader staat in voor de connectie tussen de adapters en de DAO. Deze bean wordt getriggerd door de TimerScheduler.

TrafficDataDownstreamAnalyser

Deze *analyser* staat in voor generatie van meldingen bij verkeersproblemen en controleren op geldigheid van nieuwe data. De nieuwe data zal via deze klassen kunnen worden toegevoegd aan de databank.

XXXSourceAdapter

Databronnen leveren nieuwe RouteData-objecten aan de applicatie. Deze data worden verkregen via API-aanroepen. Er zijn voorlopig al vier SourceAdapters: Here, Google Maps, Coyote en TomTom.

GeneralDAO

De GeneralDAO is verantwoordelijk voor het beheer van de routes die door de applicatie in de gaten worden gehouden. Deze worden gedefinieerd door een opeenvolging van geolocaties. De data wordt verpakt door GeoLocationEntity en RouteEntity om deze compatibel te maken met de achterliggende databank.

TrafficDataDAO

De TrafficDataDAO is verantwoordelijk voor het beheer van data die betrekking heeft op de verkeerssituatie op een bepaald moment. De data wordt verpakt door RouteDataEntity om deze compatibel te maken met de achterliggende databank.

GeneralDAONoDB/TrafficDataDAONoDB

Dit zijn twee dummy-objecten die gebruikt kunnen worden voor tests zonder de echte databank te gebruiken.

RESTAPI

Deze bean representeert de REST-service van het project. De REST-service onttrekt data uit het systeem en geeft deze vervolgens aan de gebruiker in JSON-formaat.

Hoofdstuk 3: Testplan

3.1 Core

3.1.1 Sortering geolocaties

In één route zitten meerdere geolocaties om te verzekeren dat het gevolgde pad correct is. Om zeker te zijn dat de sortering van de locaties correct gebeurt, werd een unittest geschreven.

3.1.2 GeoLocation

In de klasse GeoLocation bestaan 2 variabelen om de coördinaten te bepalen namelijk *latitude* en *longitude*. Deze moeten binnen bepaalde grenswaarden liggen, bijgevolg werd hiervoor een exceptie met bijkomende unittest geschreven.

3.1.3 SourceAdapter

Een adapter zal de data afkomstig van een bepaalde provider, in dit geval Here, aan de hand van een gevraagde route ophalen. Deze data wordt dan in een RouteData-object opgeslagen en via bovenliggende klassen weggeschreven naar de database. Het testen van het wegschrijven naar de database zelf gebeurt in de TrafficDataDownloader.

De URL en API-Key zijn terug te vinden in het SourceAdapter.properties-bestand, waar ze ook aangepast kunnen worden. Om na te gaan of er een exceptie wordt opgegooid bij een route waar de provider geen data van heeft, is een unittest beschikbaar in de bijhorende SourceAdapter-bean. Bij alle mogelijke problemen dienen excepties gelogd te worden.

De testplannen voor adapters verlopen analoog met enkele uitzonderingen.

1. Ideaal Scenario
   (a) Data wordt opgehaald via URL, API-key en geolocaties van de route
   (b) Afstand en reistijd worden uit de data (JSON-Object) gehaald
   (c) Afstand en reistijd worden in het RouteData object opgeslagen
   (d) RouteData-object wordt teruggegeven

2. Mogelijke problemen
    (a) URL / API-Key werkt niet meer
        i. Testen door gebruik te maken van een dummy .properties-bestand
    (b) Internetconnectie werkt niet
        i. Testen door internet uit te schakelen
    (c) Geen data voor Route beschikbaar (door incorrecte geolocaties)
        i. Testen door gebruik te maken van de Unittest in de Bean van de desbetreffende SourceAdapter

3.1.4 TimerScheduler

Er is een dummy TimerScheduler.properties-bestand aanwezig in de Realisatie-map, dit kan je gebruiken om onderstaande checklist te doorlopen.

Er wordt een lijn geprint in de log van de server na afloop van een interval, op deze manier kan de werking gecontroleerd worden. Je neemt ook best een timer bij de hand ter controle.

1. Test met verschillende intervallen
    (a) Werkt dit voor een interval van 2 minuten?
    (b) Werkt dit voor een interval van 5 minuten?
    (c) Werkt dit voor een interval van 10 minuten?
2. Test of tijdens het draaien van de applicatie het interval correct aangepast wordt wanneer gevraagd
    (a) Wordt de overgang gemaakt van 2 minuten naar 5 minuten?
    (b) Wordt de overgang gemaakt van 5 minuten naar 10 minuten?
    (c) Wordt de overgang gemaakt van 10 minuten naar 2 minuten?

3.1.5 TrafficDataDownloader - DownstreamAnalyser - DAO

Bij het testen van deze modules wordt gebruik gemaakt van een DummySourceAdapter.

De data die in DummySourceAdapter gegenereerd wordt ziet eruit zoals in onderstaande tabel. De teller is dus het volgnummer voor het interval, startend bij nul. De duration wordt aan de hand van de teller berekend.

teller	RouteID	Distance	Duration
0	1	1000	100
1	1	1000	130
2	1	1000	160
3	1	1000	190
4	1	1000	220
5	1	1000	250
6	1	1000	280
7	1	1000	310
8	1	1000	340
9	1	1000	370
0	2	2000	200
1	2	2000	230
2	2	2000	260
3	2	2000	290
4	2	2000	320
5	2	2000	350
6	2	2000	380
7	2	2000	410
8	2	2000	440
9	2	2000	470

In de TimerScheduler wordt bij elk interval een methode in TrafficDataDownloader getriggerd waarbij de data voor alle routes opgehaald wordt voor alle adapters. In dit geval is er dus slechts één adapter. TrafficDataDownstreamAnalyser is een klasse die de data zal doorgeven aan de TrafficDataDAO die op zijn beurt de RouteData in de database zal wegschrijven. Er worden twee routes gebruikt voor deze test, deze hebben id 1 en 2.

Indien bij het uitvoeren van deze test bovenstaande data na tien intervallen ook in de database teruggevonden kan worden met correcte timestamp, dan betekent dit dat de data correct doorgegeven wordt van module naar module.

3.1.6 DataProvider

Om in de DataProvider alle functies te testen wordt gebruik gemaakt van de DBUnit-*library*. Aan de hand van dummydata wordt de correctheid van de *output* gecontroleerd.

3.2 RESTAPI

3.2.1 DataProvider - RESTAPI

Om de DataProvider en bijhorende RESTAPI te testen wordt gebruik gemaakt van dummydata die je in de tabel hieronder kan terugvinden. Hierbij aansluitend wordt de systeemklok van de applicatie ingesteld op 8 april 2016 om 14u25.

ID	Distance	Duration	Provider	RouteID	Timestamp
1	1345	140	GoogleMaps	1	8/04/2016 14:23
2	1350	142	Here	1	8/04/2016 14:23
3	1355	141	TomTom	1	8/04/2016 14:23
4	1360	143	Coyote	1	8/04/2016 14:23
5	1345	148	GoogleMaps	1	8/04/2016 14:18
6	1350	149	Here	1	8/04/2016 14:18
7	1355	162	TomTom	1	8/04/2016 14:18
8	1345	152	GoogleMaps	1	8/04/2016 14:13
9	1350	153	Here	1	8/04/2016 14:13
10	1355	154	TomTom	1	8/04/2016 14:13
11	1360	161	Coyote	1	8/04/2016 14:13
12	1345	153	GoogleMaps	1	8/04/2016 14:08
13	1350	155	Here	1	8/04/2016 14:08
14	1355	157	TomTom	1	8/04/2016 14:08
15	1360	165	Coyote	1	8/04/2016 14:08
12	1345	153	GoogleMaps	1	8/04/2016 13:23
13	1350	155	Here	1	8/04/2016 13:23
14	1355	157	TomTom	1	8/04/2016 13:23
15	1360	165	Coyote	1	8/04/2016 13:23
12	1345	135	GoogleMaps	1	8/04/2016 6:23
13	1350	137	Here	1	8/04/2016 6:23
14	1355	136	TomTom	1	8/04/2016 6:23
15	1360	134	Coyote	1	8/04/2016 6:23
16	1345	132	GoogleMaps	1	8/04/2016 6:18
17	1355	131	TomTom	1	8/04/2016 6:18
18	1360	136	Coyote	1	8/04/2016 6:18
19	1345	135	GoogleMaps	1	7/04/2016 6:23
20	1350	134	Here	1	7/04/2016 6:23
21	1355	133	TomTom	1	7/04/2016 6:23
22	1360	136	Coyote	1	7/04/2016 6:23
23	1345	133	GoogleMaps	1	7/04/2016 6:18
24	1355	133	TomTom	1	7/04/2016 6:18
25	1360	135	Coyote	1	7/04/2016 6:18

Aan de hand van deze data kan handmatig berekend worden wat de output moet zijn bij de verschillende REST-aanvragen. Voorlopig werd slechts één voorbeeld uitgewerkt. Hierna kan men na het invoeren van de URL nakijken of het resultaat overeenkomt met de hieronder handmatig samengestelde JSON-objecten. In de JSON-objecten staat naast bepaalde onderverdelingen de desbetreffende functie in DataProvider.

Bij het berekenen van het *delaylevel* wordt gebruik gemaakt van tresholds, deze zijn als volgt: 1 (maximaal 2 minuten vertraging), 2 (4 minuten), 3 (8 minuten), 4 (20 minuten) en 5 (meer dan 20 minuten).

URL: http://verkeer-2.bp.tiwi.be/api/v2/routes/1/days?providers=
providers=GoogleMaps,Here,TomTom,Coyote&avgstart=1460121838000
&avgend=1460125438000&start=1460121838000&end=1460125438000

JSON:


```
[

    {
    "id": 1,
    "name": "R4 Zelzate",
    "geolocations": [
        {
         "name": "Zelzate",
         "latitude": 51.192226,
         "longitude": 3.776342
        },
        {
         "name": "Gent",
         "latitude": 51.086447,
         "longitude": 3.672188
        }
    ],
    "currentDuration": 142, //getCurrentDuration
    "currentVelocity": 10, //getCurrentVelocity
    "optimalDuration": 134, //getOptimalDuration
    "optimalVelocity": 10, //getOptimalVelocity
    "avgDuration": 148, //getAvgDuration
    "avgVelocity": 9, //getAvgVelocity
    "trend": 7, //getTrend
    "recentData": { //getRecentData
        "duration": {
            "name": "recentData 2",
            "description": "This data are the durations over the
                            last hour for route 1",
            "data": {
                "1460124538000": 158,
                "1460124838000": 155,
                "1460125138000": 153,
                "1460125438000": 152
            }
        }
    },
    "currentDelayLevel": 3, //getCurrentDelayLevel
    "distance": 1353, //getDistance
    "data": { //getData
        "friday": {
            "duration": {
```

```
                "name": "durations friday 1",
                "description": "This data are the durations on a
                                monday for route 1",
                "data": {
                    "1460096338000": 133,
                    "1460096638000": 135,
                    "1460124538000": 158,
                    "1460124838000": 155,
                    "1460125138000": 153,
                    "1460125438000": 152
                }
            },
            "velocity": {
                "name": "velocities friday 1",
                "description": "This data are the velocities on a
                                monday for route 1",
                "data": {
                    "1460096338000": 10,
                    "1460096638000": 10,
                    "1460124538000": 9,
                    "1460124838000": 9,
                    "1460125138000": 9,
                    "1460125438000": 9
                }
            }
        }
    }
]
```

3.3 Website

Het webgedeelte van de applicatie wordt volledig getest via Integration Testing. De website, waarvan de URL terug te vinden is in de handleiding, doet dienst als testomgeving.

3.3.1 Startpagina

Op de startpagina zijn er drie navigatiemogelijkheden. In bijlage staat een checklist waarin de werking van elke functionaliteit kan worden gecontroleerd.

3.3.2 Live

Deze pagina bestaat uit twee delen, namelijk de kaart en de tabel. Op deze pagina zijn er twee mogelijkheden waar hierna dieper wordt op ingegaan: de live-modus zelf en een weergave van de gemiddelden. In bijlage staat een checklist waarin de werking van elke functionaliteit kan worden gecontroleerd.

Live - Gemiddeld

Het plan voor het testen van live en gemiddeld is identiek aan elkaar, de checklist voor dit deel moet bijgevolg tweemaal doorlopen worden.

3.3.3 Analyse

Er zijn een aantal verschillende analyses die uitgevoerd kunnen worden, deze verlopen grotendeels analoog. Voorlopig wordt enkel een specifiek plan opgesteld voor databronnen vergelijken.

Databronnen vergelijken

In het stappenplan en in tweede instantie, de sidebar, worden parameters meegegeven aan de analyse. Deze parameters worden pas na het indienen van het formulier gezet in de URL en kunnen bijgevolg ook dan pas nagekeken worden. In bijlage staat een checklist waarin de werking van elke functionaliteit kan worden gecontroleerd.

Hoofdstuk 4: Gebruikshandleiding

4.1 Website

URL: https://http://verkeer-2.bp.tiwi.be/web/

Er zijn ook filmpjes terug te vinden waarin je wegwijs wordt gemaakt op de website, deze zijn terug te vinden op onderstaande URL.

URL: https://www.youtube.com/channel/UCfVianoeTHN38Ky9gZxGEmA

4.1.1 Live-pagina

URL: https://http://verkeer-2.bp.tiwi.be/web/live

De Live-pagina dient om een overzicht te bieden van de actuele verkeerssituatie op alle trajecten, zowel op kaart als in tabelvorm.

Door op de startpagina op Live te klikken komen we op de livepagina die onderverdeeld is in een kaartgedeelte (links) en een tabelgedeelte (rechts).

Op de kaart kan worden in- en uitgezoomd doormiddel van de + en knop, ofwel door te scrollen. Je kan de kaart ook eenvoudig verslepen.

Door op de kaart over een bepaald traject te hoveren, wordt dit traject gehighlight in de tabel. Wanneer je op een traject klikt, krijg je ook nog eens extra informatie over dit traject in een pop-up.

De kleur van het traject op de kaart komt overeen met het kleur van de actuele vertraging. Groen staat voor normaal verkeer en rood voor grote vertragingen.

De tabel aan de rechterkant bevat de volgende zaken voor elk traject:

1. De actuele reistijd in minuten
2. De actuele vertraging ten opzichte van de optimale reistijd in minuten
3. Een pijl die de trend van het verkeer aangeeft (stijgend of dalend)

Boven deze tabel loopt een timer die aangeeft wanneer er nieuwe data beschikbaar zullen zijn.

Door rechtsboven op de kaart te klikken op gemiddeld wordt er overgeschakeld op de gemiddelde modus. Deze modus werkt analoog als de livemodus maar geeft de gemiddelde verkeerssituatie voor elk traject terug. In de gemiddelde modus ontbreekt dus wel de pijl die de trend aangeeft.

Om terug te keren naar de livemodus kan op live geklikt worden. Door te klikken op het logo linksboven wordt er teruggekeerd naar de startpagina.

4.1.2 Analyse-pagina

URL: https://http://verkeer-2.bp.tiwi.be/web/analyse

Perioden vergelijken

Deze analyse kan worden gebruikt om data (van één of meerdere bronnen) te vergelijken tussen twee of meerdere periodes.

Door op de startpagina te kiezen voor Analyse en vervolgens door te klikken op 'Perioden vergelijken, komen we bij het stappenplan om de analyse op te stellen.

1. Eerst moeten de te vergelijken periodes worden ingesteld. Klik onder begin om de begindatum en het beginuur van de periode in te stellen via de kalender, bevestig met OK. Doe dit analoog voor de einddatum en einduur en klik daarna op volgende.

2. Vervolgens worden één of meerdere trajecten die je wenst te analyseren aangeduid in de lijst door erop te klikken. Wanneer je over een traject hovert, verschijnt er een afbeelding van dit traject. Als alle gewenste trajecten zijn aangeduid, klikt u op volgende.

3. Tenslotte kan je specifiëren welke dataproviders je wenst te selecteren voor het berekenen van de gemiddelde reistijden. Standaard staan alle providers aangeduid maar je kan manueel de providers die niet nodig zijn uitvinken. Klik tenslotte op analyseer.

Er verschijnt nu een grafiek waarop elke periode in een verschillend kleur wordt aangeduid. Door te scrollen kan er op de grafiek worden in- en uitgezoomd en wanneer u met de muis op de grafiek gaat staan, krijgt u de exacte data te zien. Indien u een bepaalde periode niet meer wenst weer te geven op de grafiek, kan u deze verwijderen door onder de grafiek de desbetreffende periode aan te klikken. Door nogmaals op deze periode te klikken, zal deze periode opnieuw op de grafiek verschijnen.

Door rechtsboven op tabel te klikken, kunnen we de data van deze analyse in tabelvorm bekijken.

U kan ook de aanvraag van deze analyse aanpassen door linksboven op wijzig aanvraag te klikken. In de sidebar die vervolgens verschijnt kan u de start- en einddata van de periodes, de gewenste trajecten en of de nodige dataproviders aanpassen. Indien de nodige aanpassingen zijn gebeurd, klikt u op analyseren om de gewenste grafiek en tabel te genereren.

Databronnen vergelijken

Deze analyse kan gebruikt worden om de data van twee of meerdere databronnen te vergelijken.

Door op de startpagina te kiezen voor analyse en vervolgens door te klikken op databronnen vergelijken, komen we bij het stappenplan om de analyse op te stellen.

1. Eerst moet de periode ingesteld worden waarover de dataproviders zullen worden vergeleken. Klik hiervoor op Nieuwe Periode en vul met behulp van de kalender de start- en einddatum van de periode in. Zorg ervoor dat de startdatum voor de einddatum komt natuurlijk en klik vervolgens op voeg toe.

   Doe dit voor elk van de te vergelijken periodes. Toegevoegde periodes kunnen aangepast of verwijderd worden met de knoppen naast de periode in de lijst. Als alle periodes correct zijn toegevoegd, klikt u op volgende.

2. Vervolgens worden één of meerdere trajecten die je wenst te analyseren aangeduid in de lijst door erop te klikken. Wanneer je over een traject hovert, verschijnt er een afbeelding van dit traject. Als alle gewenste trajecten zijn aangeduid, klikt u op volgende.

3. Tenslotte kan je specifiëren welke dataproviders je wenst te vergelijken met elkaar. Standaard worden deze allemaal opgenomen in de berekening maar door de ongewenste dataproviders uit te vinken, kan u dit vermijden. Klik tenslotte op analyseer.

Er verschijnt nu een grafiek waarop elke dataprovider in een verschillend kleur wordt aangeduid. Door te scrollen kan er op de grafiek worden in- en uitgezoomd en wanneer u met de muis op de grafiek gaat staan, krijgt u de exacte data te zien. Indien u een bepaalde dataprovider niet meer wenst weer te geven op de grafiek, kan u deze verwijderen door onder de grafiek de desbetreffende provider aan te klikken. Door nogmaals op deze provider te klikken, zal deze provider opnieuw op de grafiek verschijnen.

Door rechtsboven op tabel te klikken, kunnen we de data van deze analyse in tabelvorm bekijken.

U kan ook de aanvraag van deze analyse aanpassen door linksboven op wijzig aanvraag te klikken. In de sidebar die vervolgens verschijnt kan u de start- en einddatum van de periode, de gewenste trajecten en of de nodige dataproviders aanpassen. Indien de nodige aanpassingen zijn gebeurd, klikt u op analyseren om de gewenste grafiek en tabel te genereren.

Hoofdstuk 5: Installatiehandleiding

Vooraleer dit project kan worden uitgevoerd zullen enkele aanpassingen nodig zijn aan Glassfish. Volgend stappenplan moet doorlopen worden voor het project kan worden *gedeployed*, meer info over de stappen zijn terug te vinden in verdere onderverdelingen in deze handleiding.

1. Voer alle *fixes* door in Glassfish
2. Voeg de externe *libraries* toe aan Glassfish
3. Herstart Glassfish
4. Voeg JNDI *resources* toe
5. Voeg JDBC connectie toe
6. Start MySQL-server en zorg ervoor dat er een database 'verkeer2' bestaat die leeg is
7. Server
   (a) Lokaal
      i. Verifieer dat de Beans en SourceAdapter *property*-bestanden de applicatienaam voor hun JNDI naam hebben staan. Deze moeten voldoen aan volgend formaat: java:global/ApplicatieNaam/ContainerNaam/BeanNaam.
      ii. *Clean and build* allereerst VerkeerLib, hierna iedere bean en vervolgens de applicatie (Verkeer2) zelf
      iii. Vanuit Netbeans kan het project onmiddellijk *deployed* worden via de Java EE applicatie Verkeer2
   (b) Extern
      i. Verifieer dat de Beans en SourceAdapter property-bestanden de applicatienaam niet voor hun JNDI naam hebben staan. Deze moeten voldoen aan volgend formaat: java:global/ContainerNaam/BeanNaam
      ii. *Clean and build* allereerst VerkeerLib, hierna iedere bean en vervolgens de applicatie (Verkeer2) zelf
      iii. In de *adminconsole* van Glassfish: voeg iedere module toe onder de 'Applications' tab. *Deploy* steeds met alle *libraries* op te geven in het veld 'Libraries'. Voor de REST-applicatie: geef als *contextroot* 'api' op.
      iv. Herstart de applicatie

5.1 *Fixes* in Glassfish

Er dienen een aantal fixes uitgevoerd te worden in de *admin console* van de Glassfish Server.

Op deze pagina ga je naar Configurations - server.config - JVM Settings, daar selecteer je JVM Options. Hier dienen volgende JVM Options toegevoegd te worden:

-Dorg.omg.CORBA.ORBSingletonClass=com.sun.corba.se.impl.orb.
ORBSingleton

Deze fix is nodig sinds JDK 1.7u25 wegens het afwezig zijn van het com.sun.corba.ee-*package*. Deze kan vervangen worden door com.sun.corba.se.

-Duser.timezone=Europe/Brussels

Dit is nodig om de tijdszone van de applicatie in België te laten staan, stel dat de applicatie op een server in een andere tijdszone draait dan zal toch nog de Belgische tijdzone gebruikt worden.

Verder dient ook volgend commando uitgevoerd te worden in de map waar Glassfish geïnstalleerd staat.

asadmin set server.ejb-container.property.disable-nonportable-jndi-names="true"

Deze fix is nodig om de v2 Vendor-Specific JNDI Names te *disablen*. Zonder deze fix kunnen 2 EJB's niet dezelfde *remote interface* hebben omdat hun JNDI naamgeving overlapt.

5.2 Externe *libraries* in Glassfish

Alle *libraries* die in de libs-map van het project staan moeten aan Glassfish toegevoegd worden. Verder dient ook nog de VerkeerLib toegevoegd te worden, deze kan je terugvinden in de VerkeerLib-map. Het toevoegen van de *libraries* in Glassfish doe je door de bestanden te plaatsen in volgende twee mappen binnen de Glassfish-map.

1. /glassfish/lib
2. /glassfish/domains/domain1/lib/applibs

5.3 Glassfish Resources

Hieronder zijn alle resources vermeld die worden gebruikt. Deze moet manueel worden toegevoegd in het adminstratorbeheer van Glassfish.

DataProvider

```
JNDI Name: resources/properties/DataProvider
Resource Type: java.util.Properties
Properties:
org.glassfish.resources.custom.factory.PropertiesFactory.fileName=
ABSOLUTE_LOCATION_TO_PROJECT/Realisatie/DataProvider.properties
```

TimerScheduler

```
JNDI Name: resources/properties/TimerScheduler
Resource Type: java.util.Properties
Properties:
org.glassfish.resources.custom.factory.PropertiesFactory.fileName=
ABSOLUTE_LOCATION_TO_PROJECT/Realisatie/TimerScheduler.properties
```

ThresholdManager

```
JNDI Name: resources/properties/ThresholdManager
Resource Type: java.util.Properties
Properties:
org.glassfish.resources.custom.factory.PropertiesFactory.fileName=
ABSOLUTE_LOCATION_TO_PROJECT/Realisatie/ThresholdManager.properties
```

GeoJsonProvider

```
JNDI Name: resources/properties/GeoJsonProvider
Resource Type: java.util.Properties
Properties:
org.glassfish.resources.custom.factory.PropertiesFactory.fileName=
ABSOLUTE_LOCATION_TO_PROJECT/Realisatie/GeoJsonProvider.properties
```

Beans

```
JNDI Name: resources/properties/Beans
Resource Type: java.util.Properties
Properties:
org.glassfish.resources.custom.factory.PropertiesFactory.fileName=
ABSOLUTE_LOCATION_TO_PROJECT/Realisatie/Beans.properties
```

SourceAdaptors

```
JNDI Name: resources/properties/SourceAdaptors
Resource Type: java.util.Properties
Properties:
org.glassfish.resources.custom.factory.PropertiesFactory.fileName=
ABSOLUTE_LOCATION_TO_PROJECT/Realisatie/SourceAdaptors.properties
```

WebSettings

```
JNDI Name: resources/properties/WebSettings
Resource Type: java.util.Properties
Properties:
org.glassfish.resources.custom.factory.PropertiesFactory.fileName=
ABSOLUTE_LOCATION_TO_PROJECT/Realisatie/WebSettings.properties
```

5.4 Glassfish JDBC

Hieronder staat de JDBC connection pool die gebruikt wordt in de applicatie. Deze moet manueel worden toegevoegd in het adminstratorbeheer van Glassfish.

Connection Pool

```
Pool Name: generalMysql
Resource Type:  javax.sql.DataSource
Database Driver Vendor: MySQL

additional properties
password: ****
databaseName: verkeer2
serverName: 127.0.0.1
user: ****
portNumber: 3306
url: jdbc:mysql://localhost:3306/verkeer2
```

Resources

```
JNDI Name: jdbc/verkeer/general
Pool Name: generalMysql
```

Hoofdstuk 6: Bijlage

6.1 Testplan

6.1.1 Startpagina

Live	
Wordt bij hovering de knop geaccentueerd?	
Word je bij klikken doorverwezen naar de Live-pagina?	
Analyses	
Wordt bij hovering de knop geaccentueerd?	
Word je bij klikken doorverwezen naar de Analyses-pagina?	
Instellingen	
Wordt bij hovering de knop geaccentueerd?	
Word je bij klikken doorverwezen naar de Instellingen-pagina?	

6.1.2 Live

Switch-knop tussen Live en Gemiddeld	
van Live naar Gemiddeld	
Wordt de gemiddelde data in de tabel weergegeven?	
Wordt de gemiddelde data op de kaart weergegeven?	
van Gemiddeld naar Live	
Wordt de live data in de tabel weergegeven?	
Wordt de live data op de kaart weergegeven?	
Keuzemogelijkheden in balk bovenaan	
Links	
Ga je bij het klikken op het logo naar de indexpagina?	
Rechts (knoppen overlopen van links naar rechts)	
Ga je bij het klikken op de eerste knop naar de Live-pagina?	
Ga je bij het klikken op de tweede knop naar de Analyse-pagina?	
Ga je bij het klikken op de derde knop naar de Instellingen-pagina?	
Ga je bij het klikken op de vierde knop naar de startpagina?	
Wordt bij het klikken op de vijfde knop de pagina vernieuwd?	
Ga je bij het klikken op de zesde knop naar het inlogscherm?	

Live-Gemiddeld

Switch rechtsbovenaan tabel	
Routes met vertraging	
Worden enkel routes met vertraging getoond op de kaart?	
Worden enkel routes met vertraging getoond in de tabel?	
Alle routes	
Worden alle routes getoond op de kaart?	
Worden alle routes getoond in de tabel?	
Kaart	
Zoomfunctie	
Werkt inzoomend scrollen?	
Werkt uitzoomend scrollen?	
Wordt er ingezoomd bij gebruik van de '+'-knop?	
Wordt er uitgezoomd bij gebruik van de '-'-knop?	
Routes	
Standaard	
Worden alle routes correct en met richting op kaart weergegeven? (document met alle routes zal meegeleverd worden)	
Hebben alle routes de correcte kleur (afhankelijk van de treshold)?	
Functionaliteit	
Hover over een route	
Wordt (indien nodig) gescrolld naar de juiste route in de tabel?	
Wordt de route geaccentueerd in de tabel?	
Verdwijnt de accentuering na stoppen met hover?	
Klik op een route	
Verschijnt er een pop-up scherm?	
Bevat de pop-up info van het laatste kwartier in een grafiek?	
Er is een pijl die de trend correct aangeeft?	
Het pop-up scherm verdwijnt na klikken op het kruisje?	
Verschijnt er andere informatie bij klikken op een andere route?	
Tabel	
Standaard	
Wordt bij elke route correcte reistijd weergegeven?	
Wordt bij elke route correcte vertraging weergegeven?	
Is er een scrollbar indien niet alle routes op één pagina passen?	
Verschijnt er relevante informatie in de pop-ups rechtsonderin?	
Wordt het interval in de balk bovenin correct weergeven?	
Wordt de data gepdate na afloop van het interval?	
Functionaliteit	
Hover over een route	
Wordt de route geaccentueerd in de tabel?	
Verdwijnt de accentuering na stoppen met hover?	
Klik op reistijd	
Wordt de tabel gesorteerd van grote naar kleine reistijd?	
2e klik: wordt de tabel omgekeerd gesorteerd?	
Klik op vertraging	
Wordt de tabel gesorteerd van grote naar kleine vertraging?	
2e klik: wordt de tabel omgekeerd gesorteerd?	

6.1.3 Analyse

Databronnen vergelijken

Stappenplan	
Periode instellen	
Verschijnt na afloop de startdatum in de URL?	
Verschijnt na afloop de einddatum in de URL?	
Word je verplicht een einddatum later dan de startdatum in te stellen?	
Route kiezen	
Word je verplicht maximaal één route te selecteren?	
Is het zo dat als je geen route selecteert, de knop 'Volgende' niet werkt?	
Verschijnt na afloop de Route-ID in de URL?	
Provider kiezen	
Word je verplicht minimaal één route te selecteren?	
Is het zo dat als je geen provier selecteert, de knop 'Volgende' niet werkt?	
Verschijnen na afloop de providers in de URL?	
Tijdens deze analyse klik je op een andere analyse	
Worden (indien mogelijk) de parameters behouden?	
Grafiek	
Wordt de correcte data weergegeven in de grafiek (nakijken met RESTAPI)?	
Grafiek	
Wordt de correcte data weergegeven in de tabel (nakijken met RESTAPI)?	
Sidebar	
Periode instellen	
Verschijnt na afloop de startdatum in de URL?	
Verschijnt na afloop de einddatum in de URL?	
Word je verplicht een einddatum later dan de startdatum in te stellen?	
Route kiezen	
Word je verplicht maximaal één route te selecteren?	
Is het zo dat als je geen route selecteert, de knop 'Volgende' niet werkt?	
Verschijnt na afloop de Route-ID in de URL?	
Provider kiezen	
Word je verplicht minimaal één route te selecteren?	
Is het zo dat als je geen provier selecteert, de knop 'Volgende' niet werkt?	
Verschijnen na afloop de providers in de URL?	
Klik op analyseer	
Worden de parameters in de URL aangepast?	
Wordt de correcte data weergegeven in de grafiek (nakijken met RESTAPI)?	
Wordt de correcte data weergegeven in de tabel (nakijken met RESTAPI)?	