Rapport Leertaak 1 - BSS

United Nations Weather Data Management Institute   
Weergegevens Applicatie

**INV2B – Groep 5**  
Mark Nijboer, Kevin Haitsma   
en Rick van der Poel

Rapport Leertaak 1

**Datum:**   
25-09-2015

**Auteurs:**   
Mark Nijboer, Kevin Haitsma en Rick van der Poel

**Opdrachtgever:**   
UNWDMI

**Instituut:**  
Hanzehogeschool Groningen

#### Versiebeheer

|  |  |  |  |
| --- | --- | --- | --- |
| **Versie** | **Datum** | **Auteur** | **Beschrijving** |
| 0.1 | 23-09-2015 | Rick van der Poel | Indeling en begin verslag |
| 0.2 | 24-09-2015 | Rick van der Poel | Inleiding, probleemstelling, testsystemen |
| 0.3 | 25-09-2015 | Rick van der Poel | Samenvatting, verklarende woordenlijst, applicatie onderdelen |
| 0.4 | 25-09-2015 | Rick van der Poel | Samenvatting, stresstest grafieken |
|  |  |  |  |
|  |  |  |  |

Samenvatting

In dit rapport vindt u informatie over de proef- demoapplicatie die ontwikkeld is om te kijken hoe een grote workload aan gegevens verwerkt kunnen worden en waar problemen zich voor doen bij het verwerken van deze gegevens. De resultaten die voortkomen uit deze demoapplicatie zullen worden meegenomen bij de bouw van de uiteindelijke applicatie.

De demoapplicatie zal een groot aantal gegevens die binnenkomen in clusters moeten kunnen verwerken en daarna opslaan in een database. De gegevensstroom hiervoor is enorm en het doel is dan ook te kijken waar de bottleneck zit bij het verwerken van de gegevens. Daarnaast zal er ook een correctie gedaan moeten worden op gegevens die niet volledig of juist zijn binnengekomen.

De gegevens worden aangevoerd in XML een opmaaktaal die het als “platte tekst” aanlevert. Deze data zal vervolgens bruikbaar gemaakt worden, gecorrigeerd worden en vervolgens klaargemaakt worden voor de database. Om dit allemaal te kunnen doen wordt de applicatie in Java geschreven en wordt het programma in meerdere klasse verdeeld volgens de OO-methode.

Elke klasse heeft een eigen taak en samen functioneren deze klassen als applicatie. Hiervoor zijn de volgende klassen gemaakt:

- een klasse voor het initialiseren van de applicatie  
- een klasse voor het opzetten van de connectie met de cliënt  
- een klasse voor het verwerken van de XML-data  
- een klasse voor het corrigeren van de data  
- een klasse voor het verzamelen van de gegevens voor de database en het  
- en een klasse voor het invoeren van de gegevens in de database

Vervolgens is de applicatie getest om te kijken hoe het reageert op de grote stroom van gegevens. Hieruit is gebleken dat… [insert more text here]

Inhoudsopgave

[Samenvatting 3](#_Toc430791301)

[1. Inleiding 6](#_Toc430791302)

[2. Probleemstelling 7](#_Toc430791303)

[3. Applicatie onderdelen 8](#_Toc430791304)

[4. Stresstest 9](#_Toc430791305)

[4.1 Specificaties testsystemen 9](#_Toc430791306)

[4.2 Verwerkingssnelheden 9](#_Toc430791307)

[4.3 Bottleneck 9](#_Toc430791308)

[5. Conclusie 10](#_Toc430791309)

Verklarende woordenlijst

**Cliënt** – De cliënt is het programma van de UNWDMI dat de weergegevens doorstuurt naar de applicatie.

**Server** – De server is de applicatie waarmee de gegevens worden ontvangen en worden verwerkt.

**XML** – Extensible Markup Language is de opmaaktaal waarin de cliënt de weergegevens aanlevert in de vorm van platte tekst.

**UNWDMI** – Staat voor United Nations Weather Data Management Institute.

**Bottleneck** – Is de beperking in de applicatie waardoor het niet optimaal kan functioneren.

1. Inleiding

Voor u ligt het rapport waarin verslag wordt gedaan over de proef- demonstratie applicatie dat in ontwikkeling is voor de UNWDMI (United Nations Weather Data Management Institute). Dit rapport moet inzicht geven over de opbouw en prestaties van de demo-applicatie om zo als solide basis te dienen voor het uiteindelijke programma dat later ontwikkeld zou worden.

Het rapport worden een aantal belangrijke zakken naar voren gebracht:

Als eerst wordt er in hoofdstuk 2 gekeken wat nou precies de probleemstelling is en waarom de applicatie gemaakt is. Hierdoor wordt er een duidelijk beeld gecreëerd waaraan de applicatie moet voldoen.

Vervolgens wordt uitgelegd in hoofdstuk 3 uit welke onderdelen de applicatie bestaat en wat de functie is van elk van deze onderdelen.

Daarnaast is de applicatie natuurlijk ook uitvoerig getest. Deze test resultaten worden in hoofdstuk 4 besproken.

Uiteindelijk worden alle bevindingen op een rijtje gezet als conclusie in hoofdstuk 5.

2. Probleemstelling

De UNWDMI heeft een groot aantal weerstations over de wereld die elk een grote hoeveelheid aan gegevens binnen halen. Op deze gegevens worden tegelijkertijd zware selecties uitgevoerd om relevante data te onderscheiden. Aan ons is de taak een om uiteindelijk een systeem te bouwen dat deze gegevens kan verwerken en waar uiteindelijk een service aan gekoppeld kan worden. Ter voorbereiding op de bouw wordt er echter eerst een proef- en demonstratie systeem gebouwd met een relationele database.

Het demo-systeem zal aan enige eisen moeten voldoen ter voorbereiding op het uiteindelijke systeem:

- Het programma moet in Java J2SE 1.5.0 of hoger geschreven worden  
- Gebruik van een Postgres of MySQL relationele database  
- Referentiele integriteit **moet** afgedwongen worden  
- Gegevens moeten ontvangen worden in XML-formaat uit meerdere bronnen   
 met gebruik van een enkele socket.   
- Er moet een correctie op de weergegevens uitgevoerd worden  
- De weergegevens moeten opgeslagen worden  
- Er moet een voorbeeld query op een selectie van de weergegevens gedemonstreerd kunnen worden.

Het systeem moet naast de bovenstaande functionaliteiten ook bestand zijn tegen de hoog stroom aan gegevens die de vele weerstations zullen aanvoeren. Hiervoor zal het systeem uitvoerig getest moeten worden of het een hoge stroom van gegevens aan kan. Daarnaast moeten ook de grenzen van het systeem naar voren komen zodat het duidelijk is waar de bottleneck van het systeem zich bevindt.

3. Applicatie onderdelen

De applicatie is opgebouwd uit meerdere onderdelen die elk een eigen functie hebben om te voldoen aan de eisen van de opdrachtgever.

**Main**  
Dit is de hoofdklasse van de applicatie waarin de applicatie wordt gestart. Deze klasse zorgt ervoor dat alle benodigde onderdelen gestart worden en er een connectie tot stand wordt gebracht.

**ClientConnection**Deze klasse zorgt dat er een verbinding tot stand komt met de cliënt en dat de weerdata van de cliënt wordt ontvangen. Daarna roept deze klasse de XML-parser aan om de weerdata te verwerken.

**XMLparser**  
De XMLparser klasse zorgt er voor dat de XML-data die wordt geleverd door de cliënt ingelezen wordt en aangepast wordt zodat het in een database gezet kan worden.

**DataCorrection**Deze klasse is voor het controleren van de gegevens en zo nodig deze gegevens te corrigeren wanneer het niet goed is doorgekomen. De correctie wordt doormiddel van een gewogen gemiddelde gedaan die wordt bepaald door voorgaande gegevens van het zelfde station.

**DatabaseQueue**De DatabaseQueue klasse verzameld alle verwerkte XML-data zodat dit gezamenlijk in één query verwerkt kan worden in de database.

**DatabaseInterval**De databaseInterval klasse zorgt er voor dat verzamelde gegevens in de query pas na elke seconde in de database worden gestopt. Hierdoor kunnen we de datastroom naar de MySQL database beperken zodat het niet overbelast wordt door het aantal connecties.

4. Stresstest

Om te kijken of de applicatie gereed is voor dagelijks gebruik, is de applicatie een stresstest ondergaan. In dit hoofdstuk leest u waaruit de testsystemen bestonden, waarop getest is, hoe de applicatie door de test is gekomen en wat de bottleneck van de applicatie is.

## 4.1 Specificaties testsystemen

|  |  |  |
| --- | --- | --- |
| **Cliënt & Server Systeem 1 – Desktop (Rick)** | | |
| Besturingssysteem | CPU | Geheugen |
| Windows 10 | Intel Core I7-4770K 3.50 GHz | 8GB RAM 2133Mhz |
| **Cliënt & Server Systeem 2 – Notebook (Mark)** | | |
| **Besturingssysteem** | **CPU** | **Geheugen** |
| Ubuntu | Intel Core I7-4712HQ 2.3 GHz | 16GB RAM 1600Mhz |
| **Cliënt & Server Systeem 3 – Notebook (Kevin)** | | |
| **Besturingssysteem** | **CPU** | **Geheugen** |
| Windows 8 | Intel Core I5-4210U 1,7 GHz | 8GB RAM 1333Mhz |
| **Cliënt & Server Systeem 4 – Notebook (Rick)** | | |
| **Besturingssysteem** | **CPU** | **Geheugen** |
| Windows 10 | Intel Core I7-4720HQ 2.6 GHz | 16GB RAM 1600Mhz |

De applicatie is op elk systeem lokaal getest om de data transfer zo hoog mogelijk te houden. Dit is gedaan om te kijken welke snelheden de applicatie aan kan met de maximale doorvoer snelheid zonder de limiterende snelheidsfactor van het internet of bijvoorbeeld satelliet.

## 4.2 Verwerkingssnelheden

De applicatie is op de verschillende systemen getest en hieruit zijn de volgende metingen naar voren gekomen:

|  |  |
| --- | --- |
| **Systeem 1 - Desktop** | |
| Max. verwerking per sec | 800 |
| Verlies bij maximale doorvoer van 800 clusters | 0 |

|  |  |
| --- | --- |
| **Systeem 2 - Notebook** | |
| Max. verwerking per sec | 650 |
| Verlies bij maximale doorvoer van 800 clusters | 150 |

|  |  |
| --- | --- |
| **Systeem 3 - Notebook** | |
| Max. verwerking per sec | 600 |
| Verlies bij maximale doorvoer van 800 clusters | 200 |

|  |  |
| --- | --- |
| **Systeem 4 - Notebook** | |
| Max. verwerking per sec | 700 |
| Verlies bij maximale doorvoer van 800 clusters | 100 |

## 4.3 Bottleneck

5. Conclusie