**Software Architectuur Document (SAD)**

Project: OS3 Alarmsysteem

Project team: OS31 – Groep 5

Teamleden: Hovestadt, Lucas L.J.

Jong, Bastiaan B.S. de

Kusters, Melvin M.

Smits, Wesley W.W.T.M.

Opdrachtgever: Franssen, Michael M.G.J., Fontys Hogescholen

Versie: 0.3

Versiedatum: 28 oktober 2019

Status: Deliverable

# Documenthistorie

|  |  |  |  |
| --- | --- | --- | --- |
| Versie | Wijzigingen | Auteur | Datum |
| 0.1 | Opzet document | Nico Kuijpers | 14-12-2017 |
| 0.2 | Documentstructuur aangepast. Componentendiagram, klassendiagram en sequentiediagram toegevoegd. | Melvin Kusters, Wesley Smits | 28-10-2019 |
| 0.3 | Beschrijving toegevoegd aan componentendiagram, klassendiagram en sequentiediagram. | Melvin Kusters | 28-10-2019 |

Inhoud

[Documenthistorie 2](#_Toc23180065)

[H1 Inleiding 4](#_Toc23180066)

[H2 Componentendiagram 5](#_Toc23180067)

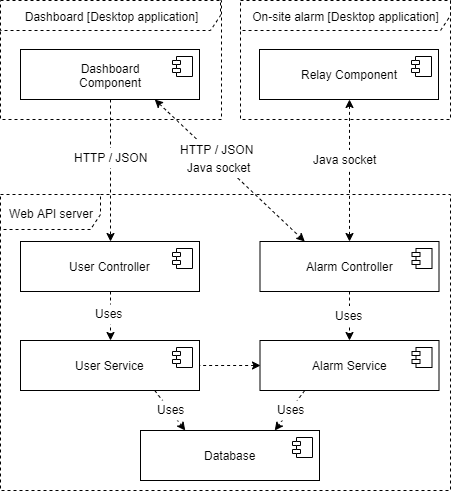
[H3 Klassendiagram en sequentiediagram 7](#_Toc23180068)

# H1 Inleiding

Het doel van het op te leveren gedistribueerde systeem betreft het faciliteren van beveiliging op een fysieke locatie in de vorm van een alarmsysteem, welke verschijnselen in de fysische wereld detecteert en o.b.v. specifieke criteria een waarschuwingssignaal afgeeft aan de gebruiker.

In dit document zijn schematische overzichten met gepaste beschrijvingen van de gelaagde softwarearchitectuur van het op te leveren systeem opgenomen.

# H2 Componentendiagram



In het bovenstaande componentendiagram is af te leiden uit welke componenten het systeem bestaat, en hoe ze tot elkaar in relatie staan. De componenten zijn gegroepeerd binnen subsystemen, die een losstaand domein markeren.

De verschillende componenten waaruit het systeem is opgemaakt zijn zelfstandige onderdelen die elk verantwoordelijk zijn voor een specifieke taak binnen het systeem.

Het **Dashboard Component** is verantwoordelijk voor het aanbieden van de user interface. Hieronder valt het mogelijk maken van interactie met het systeem door de gebruiker en het presenteren van ontvangen informatie. Het component maakt gebruik van zowel een stateful als stateless communicatiestrategie, namelijk:

* Het component verstuurt en ontvangt on-demand objecten van het type **User** en **Alarm** door endpoints aan te roepen op respectievelijk de **User Controller** en de **Alarm Controller.** De onderlinge communicatie vindt plaats over **HTTP**. De objectgegevens worden geserialiseerd in **JSON**.
* Het component ontvangt real-time sensorgegevens welke zijn ondergebracht in objecten van het type **Alarm** door een stateful verbinding tot stand te brengen met de **Alarm Controller** middels een **Java Socket**.

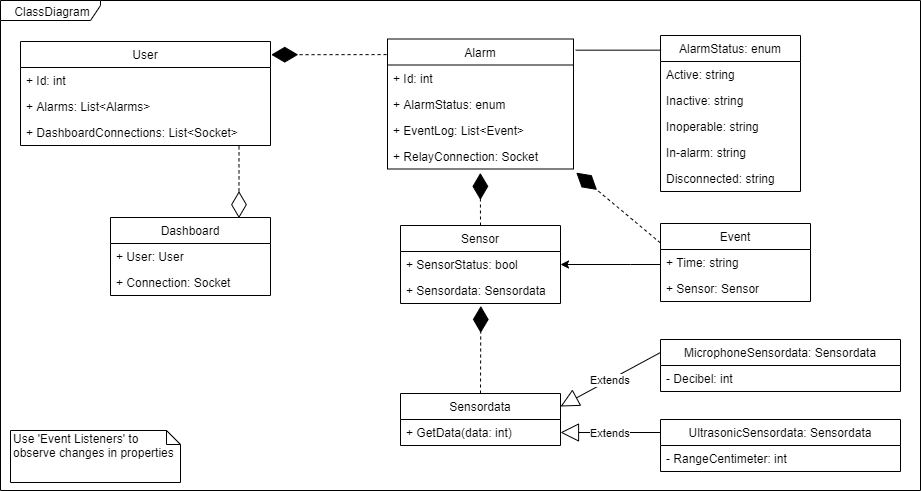
Het **Relay Component**  is verantwoordelijk voor het uitwisselen van informatie over het gekoppelde sensorsysteem tussen de Arduino (sensorsysteem, fysieke hardware opgenomen in component) en de **Alarm Controller** door een stateful verbinding tot stand te brengenmiddels een **Java Socket**.

De **User Controller** is verantwoordelijk voor het beheren en aanbieden van de gebruikers van het systeem. Het component maakt gebruik van de **User Service** om business logic aan te roepen.

De **Alarm Controller** is verantwoordelijk voor het beheren en aanbieden van de alarmen die zijn opgenomen in het systeem. Het component maakt gebruik van de **Alarm Service** om business logic aan te roepen.

De componenten **User Service** en **Alarm Service** bieden beide interfaces aan die de business logic van hun entiteitstype toegankelijk maken. Voor het persisten van gegevens wordt een onderliggende MySQL-database gebruikt.

# H3 Klassendiagram en sequentiediagram



Het bovenstaande klassendiagram bevat de entiteitstypes met bijbehorende eigenschappen die worden gebruikt door het systeem, en beschrijft diens relatie tot elkaar.

**Dashboard** heeft 0 of 1 **User**.

**User** heeft 0 of \* **Alarmen**.

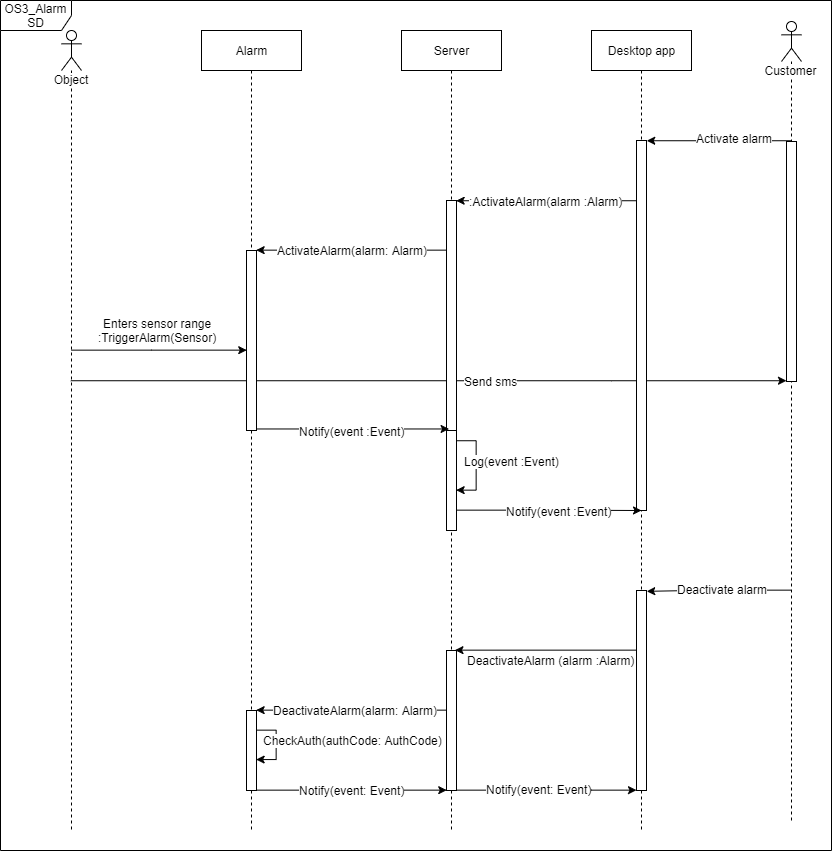
**Alarm** heeft een opgesomd attribuut genaamd “AlarmStatus” van het type **AlarmStatus**.

**Alarm** heeft 0 of \* **Sensors**.

**Alarm** heeft 0 of \* **Events** die worden gecreëerd door een **Sensor**.

**Sensordata** wordt uitgebreid door **MicrophoneSensordata** en **UltrasonicSensordata**.

*Initiëel onderzoek heeft geresulteerd in een mogelijke oplossing voor het observeren van veranderingen in objecteigenschappen t.b.v. de real-time communicatie (zie linkeronderhoek van bovenstaande figuur).*



Het bovenstaande sequentiediagram laat op chronologische wijze het gevolg van handelingen zien die worden geïnitieerd door een gebruiker van het systeem. Het gevolg is in bovenstaande figuur opgenomen in de vorm van methodes die worden uitgevoerd binnen het systeem.

De eerste sequentie betreft het activeren van een alarm door de gebruiker. De tweede sequentie betreft het deactiveren van een alarm door de gebruiker.