Ontwikkeling van prototype leerlingen- en klimaatregistratiesysteem voor scholen

Afstudeerverslag

Student : Rob Kleinee
Studentnummer : 11062150
Onderwijsinstelling : Haagse Hogeschool Delft
Opleiding : Technische Informatica
Afstudeerperiode : 09-02-2015 t/m 05-06-2015
Begeleider : Tony Andrioli
Tweede examinator : Adri Pronk

Bedrijf : Alflex Technologies
Afdeling : Development
Bedrijfsbegeleider : Thomas Wagenveld
Opdrachtgever : Fred Lucas

Datum : 03-06-2015
Versie : 1.0

Versiebeheer

Versienr	Datum	Aanpassingen
0.1	28-03-2015	Basisconcept
0.2	02-05-2015	Tweede concept. Aanpassingen naar aanleiding van feedback Tony Andrioli. Het hoofdstuk Construction Fase 2 geschreven.
1.0	03-06-2015	Feedback van Tony Andrioli en Adri Pronk verwerkt. Het verslag verder afgeschreven.

Reviewers

Versienr	Datum	Reviewer
0.1	13-04-2015	Tony Andrioli
0.2	06-05-2015	Tony Andrioli, Adri Pronk

Gerelateerde documenten

Ref	Auteur	Titel
B	Rob Kleinee	Afstudeerverslag bijlagenboek

Referaat

Rob Kleinee, “Ontwikkeling van prototype leerlingen- en klimaatregistratiesysteem voor scholen”. Afstudeerverslag opleiding Technische Informatica, Haagse Hogeschool Delft, 2015.

In dit afstudeerverslag wordt het proces beschreven dat heeft plaatsgevonden bij de uitvoering van een afstudeeropdracht bij het bedrijf Alflex Technologies, in de periode van februari 2015 t/m juni 2015.

De opdracht betreft het ontwikkelen van een prototype/demonstratiemodel van een systeem, waarmee de aanwezigheid van leerlingen wordt geregistreerd en waarmee de kwaliteit van het klimaat binnen een klaslokaal inzichtelijk wordt gemaakt. Het prototype is ontwikkeld om inzichtelijk te maken wat de mogelijkheden zijn en om interesse te wekken bij scholen, om zo het eventueel toekomstige product te vermarkten.

Voor het ontwikkelen van het prototype is er onderzoek gedaan naar technieken waarmee een persoon geïdentificeerd kan worden. Er is een RFID-reader gebruikt, waarmee een leerling zich kan aanmelden door middel van een tag. Door middel van de ontwikkelde meetstations, die via radio frequentie (RF) communiceren, wordt het klimaat in een klaslokaal gemeten. Om de aanwezigheid van de leerlingen en het klimaat binnen het klaslokaal inzichtelijk te maken, is er een Android applicatie ontwikkeld.

Descriptoren:

- Studentenregistratie
- Persoonsidentificatie
- Klimaatmetingen
- Embedded systemen
- Smartphone applicatie
- Android

Voorwoord

Ter afronding van de opleiding Technische Informatica aan de Haagse Hogeschool in Delft is door mij een afstudeertraject doorlopen. Hiervoor is een afstudeeropdracht uitgevoerd bij het bedrijf Alflex Technologies, waarmee ik wil aantonen dat ik in staat ben tot het zelfstandig uitvoeren van een softwareontwikkelingstraject op Hbo-niveau.

Aan de basis van het afstudeertraject ligt een afstudeeropdracht die is opgezet door Alflex. De opdracht betreft het ontwikkelen van een prototype/demonstratiemodel van een systeem, waarmee de aanwezigheid van leerlingen wordt geregistreerd en waarmee de kwaliteit van het klimaat binnen een klaslokaal inzichtelijk wordt gemaakt.

Dit afstudeerverslag beschrijft het doorlopen proces tijdens de uitvoering van de afstudeeropdracht. Hierbij wordt ingegaan op de verschillende werkzaamheden en gemaakte keuzes. Het verslag is primair geschreven voor mijn beoordelaars. Echter is het verslag ook te lezen door ieder ander geïnteresseerde, mits diegene enige kennis bezit van softwareontwikkeling en het volgen van een softwareontwikkelingstraject.

Ik wil van de gelegenheid gebruik maken om mijn collega's bij Alflex te bedanken voor een leuke en leerzame tijd. In het bijzonder wil ik Thomas Wagenveld en Fred Lucas bedanken voor de begeleiding tijdens de uitvoering van de opdracht. Tot slot wil ik Tony Andrioli en Adri Pronk bedanken voor de begeleiding vanuit de opleiding en de goede tussentijdse feedback op mijn verslag.

Rob Kleinee,
Delfgauw, 2 juni 2015

Samenvatting

Ter afronding van de studie Technische Informatica aan de Haagse Hogeschool in Delft, is er door de afstudeerder Rob Kleinee een afstudeeropdracht uitgevoerd bij het bedrijf Alflex Technologies. Aan de basis van de opdracht lag het probleem dat docenten op scholen vaak veel tijd kwijt zijn aan ondersteunende, administratieve processen zoals aanwezigheidsregistratie van leerlingen. Daarnaast is er doorgaans nauwelijks inzicht in het klimaat binnen een klaslokaal, dat vaak van onvoldoende kwaliteit is.

Als antwoord hier op is tijdens de afstudeeropdracht een prototype van een systeem ontwikkeld, waarmee studenten zich kunnen aan- en afmelden en waarmee de aanwezigheid van de studenten en de kwaliteit van het klimaat binnen het klaslokaal inzichtelijk wordt gemaakt door middel van een smartphone applicatie. Het systeem bestaat daarbij uit de volgende subsystemen: meetstations voor het meten van de klimaatwaardes, een basisstation waarbij leerlingen zich kunnen aanmelden en een Android applicatie voor het inzichtelijk maken van de gegevens.

Bij aanvang van het project zijn verschillende methodieken tegen elkaar afgewogen, om zo tot een keuze te komen tot het gebruik van Open Up bij de uitvoering van het project. Door deze keuze is het project opgedeeld in de vier fases van Open Up: de inception fase, de elaboration fase, de construction fase en de transition fase.

De inception fase bestond uit één iteratie waarin de eisen aan het systeem zijn achterhaald, er een keuze is gemaakt voor de Nordic nRF51 ontwikkelkit als hardware voor de stations en de Eclipse ontwikkelomgeving is opgezet voor het schrijven van de software voor de stations in C++.

De elaboration fase was opgedeeld in twee iteraties. In de eerste iteratie is er een analyse uitgevoerd naar identificatietechnieken en is er een keus gemaakt voor het gebruik van RFID, zodat elke leerling zich kan aanmelden door het gebruik van een RFID-tag. In de tweede iteratie van de elaboration fase is er een analyse uitgevoerd naar de beschikbare protocollen voor de communicatie tussen de stations en naar de smartphone. Er is een keus gemaakt voor het gebruik van het ANT-protocol voor de communicatie tussen de stations en het gebruik van Bluetooth Smart voor de communicatie tussen het basisstation en de smartphone applicatie.

De construction fase bestond uit drie iteraties. In de eerste iteratie is het grootste gedeelte van de software voor het basisstation ontwikkeld, zodat dataopslag op een SD-kaart mogelijk werd en een RFID-reader gebruikt kon worden. In de tweede iteratie is de software voor de meetstations ontwikkeld, die de gemeten temperatuur, luchtvochtigheid en CO2 doorsturen naar het basisstation. In de derde iteratie is een Android applicatie ontwikkeld, waarmee de aanwezigheid van leerlingen kan worden opgezocht, evenals het huidige klimaat en de historie van de klimaatwaardes.

In de transition fase zijn alle systeemtests uit de construction fase nog eens uitgevoerd, voor een test van het complete systeem. Er is een gebruikersacceptatietest uitgevoerd met de opdrachtgever en het systeem is opgeleverd met alle gemaakte documentatie.

Er kan worden geconcludeerd dat de doelstelling, zoals deze aan het begin van het project is geformuleerd, grotendeels is behaald. Het prototype dat is gerealiseerd gedurende het project geeft een goed beeld van de mogelijkheden van een systeem voor leerling- en klimaatregistratie. Door tijdsgebrek is het niet gelukt om alle systeemeisen te implementeren, echter is aan de meeste systeemeisen voldaan, wat tot een werkend product heeft geleid.

Inhoudsopgave

Figuren

Tabellen

1 Inleiding

Ter afronding van de studie Technische Informatica aan de Haagse Hogeschool in Delft is er een afstudeeropdracht uitgevoerd bij het bedrijf Alflex Technologies. Aan de basis van deze opdracht lag het probleem dat docenten op scholen vaak veel tijd kwijt zijn aan ondersteunende, administratieve processen zoals onder andere aanwezigheidsregistratie van leerlingen. Daarnaast is het klimaat in een klaslokaal vaak van onvoldoende kwaliteit, wat kan resulteren in hoofdpijn en vermoeidheid. Dit is vaak te wijten aan onvoldoende inzicht in het klimaat binnen het klaslokaal.

Het doel van de afstudeeropdracht is om een prototype te ontwikkelen van een systeem waarmee studenten zich kunnen aan- en afmelden en waarmee de aanwezigheid van de studenten en de kwaliteit van het klimaat binnen het klaslokaal inzichtelijk wordt gemaakt. De opdrachtgever kan dit prototype vervolgens gebruiken om interesse te wekken bij scholen en zo te onderzoeken of er interesse is in een dergelijk product.

Dit afstudeerverslag beschrijft het doorlopen proces tijdens de uitvoering van de afstudeeropdracht. Hierbij wordt ingegaan op de verschillende werkzaamheden en gemaakte keuzes. Er worden delen behandeld van andere documenten die zijn geschreven gedurende de uitvoering van de opdracht. Daarbij wordt verwezen naar de volledige documenten die te vinden zijn in de bijlage, die dienen als extra informatie. De bijlagen zijn te vinden in een apart bijlagenboek.

Aan het begin van het verslag worden achtereenvolgens het afstudeerbedrijf, de opdrachtomschrijving, de aanpak en de planning besproken. Daaropvolgend wordt het proces besproken gedurende de uitvoering van de opdracht, waarbij de hoofdstukken zijn ingedeeld naar de fases van de gekozen ontwikkelmethodiek. Het verslag wordt afgesloten met een conclusie en aanbevelingen, gevolgd door een projectevaluatie.

2 Alflex Technologies

Dit hoofdstuk beschrijft globaal het bedrijf waar de afstudeeropdracht is doorlopen, zodat de verdere informatie over de afstudeeropdracht in dit verslag in een betere context is te plaatsen.

2.1 Algemene informatie Alflex

Alflex Technologies is een profit organisatie, gespecialiseerd in het bieden van maatwerk oplossingen op het gebied van elektronica, thermodynamische systemen, procesbesturing en meet- en regeltechniek. Daarnaast is Alflex Technologies business partner van een aantal fabrikanten, waarvan onder andere Minco Products, Matsuo Electric en Ditel.

Alflex Technologies is gestart in 1991 en telt op het moment rond de vijftien medewerkers. Er is een tweedeling in activiteiten te onderscheiden. Enerzijds worden er thermische oplossingen verkocht aan andere bedrijven, voor zowel koelen als verwarmen. Anderzijds wordt op aanvraag van klanten op elektronicagebied maatwerk ontwikkeld. Dit kunnen compleet werkende producten zijn, of een onderdeel van een groter product. Deze producten worden geheel binnen Alflex ontwikkeld, waarbij zowel de hardware als de software wordt gerealiseerd. Daarbij staat innovatie, betrouwbaarheid, een kortere time-to-market en duurzaamheid centraal.

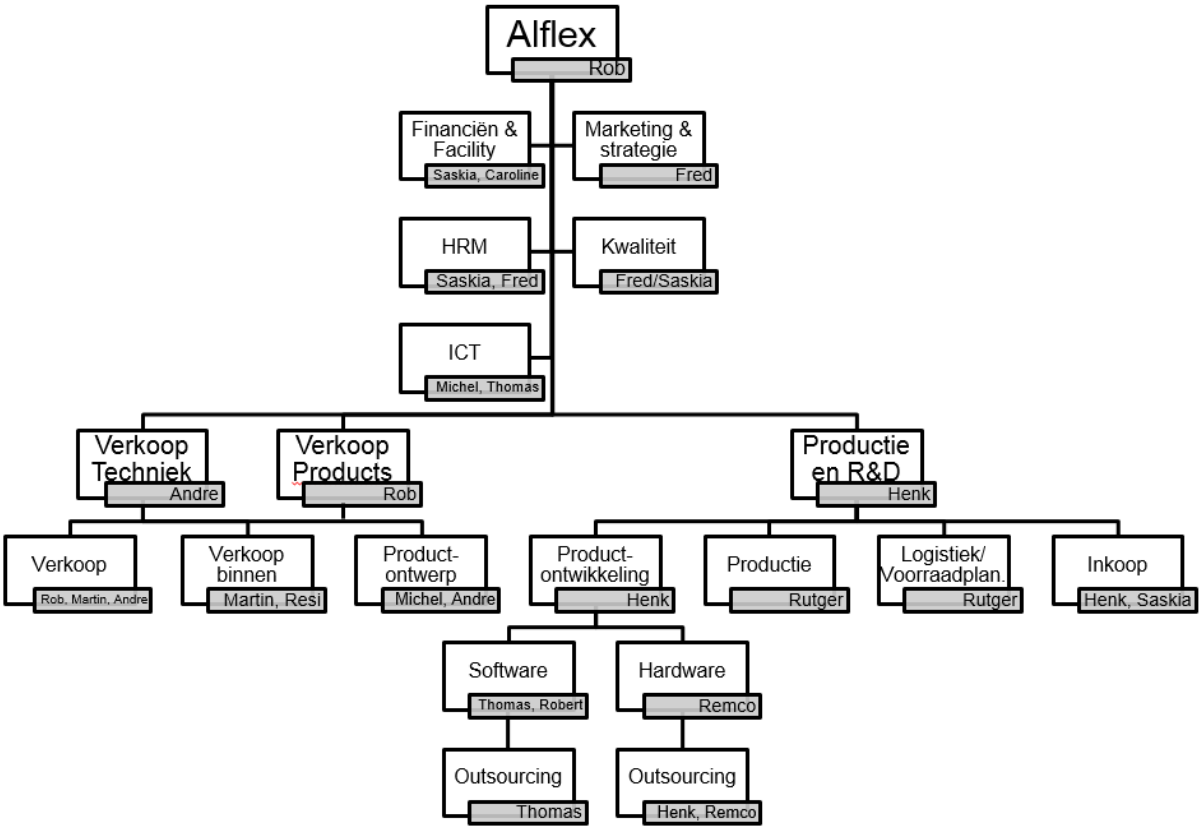
Ondertussen heeft Alflex een breed portfolio opgebouwd aan gerealiseerde oplossingen. Zo zijn er producten ontwikkeld voor verschillende soorten markten op het gebied van milieu, defensie, medisch, industrieel en offshore. Zo is er bijvoorbeeld een product ontwikkeld voor een biologische gewasbestuiving in de kas met onnatuurlijke lichtomstandigheden voor hommels, een product dat dient als anticorrosie oplossing voor windmolens en schepen, en een product voor de koeling en behoud van donororganen. [1]

2.2 Organisatiestructuur

Alflex is te typeren als een lijn-staf organisatie. Er is een hiërarchie te onderscheiden, maar toch is de organisatie relatief plat. Een organogram is te zien in Figuur 1. Opvallend is dat veel werknemers verschillende functies vervullen binnen het bedrijf. Zo is bijvoorbeeld Fred Lucas verantwoordelijk voor de onderwerpen *HRM, Marketing & Strategie* en *Kwaliteit.*

Onder de directeur, Rob Vogel, is een vijftal ondersteunende functies te onderscheiden. Vervolgens zijn er drie hoofdafdelingen: *Verkoop Techniek*, *Verkoop Products* en *Productie en R&D*. De afdeling *Verkoop Products* verkoopt maatwerkproducten die worden gemaakt op aanvraag van de klant. Deze producten worden vervolgens op de afdeling *Productie en R&D* gerealiseerd, zowel op het vlak van de hardware als de software. Op de afdeling *Verkoop Techniek* worden de thermische oplossingen verkocht, waarbij een deel bij externe partijen wordt ingekocht, het andere deel wordt binnen *Productie en R&D* ontwikkeld.

De sfeer binnen het bedrijf is informeel en ontspannen. Er wordt via korte lijnen overleg gepleegd, zodat de informatie meteen bij de juiste persoon terecht komt. Voor het inwinnen van informatie lopen de werknemers bij elkaar langs of spreken elkaar aan op de wandelgangen.



Figuur 1: Organogram Alflex [2]

2.3 Rol van afstudeerder binnen Alflex

De afstudeerder is geplaatst op de afdeling *Productie en R&D*, specifiek op de subafdeling *Software*. Hier wordt de software ontwikkeld voor de te leveren producten. Over het algemeen worden deze producten op aanvraag van klanten gemaakt. Echter is er binnen Alflex ook het idee ontstaan om zelf producten te ontwikkelen en onder eigen naam in de markt te zetten. Vanuit dit idee is ook de afstudeeropdracht ontstaan. Deze opdracht wordt niet uitgevoerd in opdracht van een klant van Alflex, maar de opdracht wordt uitgevoerd in opdracht van Alflex zelf.

De opdrachtgever binnen Alflex is Fred Lucas. Hij bepaalt de eisen aan het systeem en keurt het eindresultaat. Daarbij vervult hij de functie *Marketing & Strategie*.

3 Opdrachtomschrijving

Voorafgaand aan het afstudeerproject is er een afstudeerplan geschreven, waarin onder andere de probleemstelling, de doelstelling en globaal de uit te voeren werkzaamheden voor de opdracht beschreven staan (bijlage A). Bij aanvang van het project zijn er opnieuw gesprekken gevoerd met de opdrachtgever om zo te achterhalen of de opdracht nog steeds overeenkomt, zoals deze destijds is geformuleerd in het afstudeerplan. Dit bleek nog steeds het geval.

Vervolgens is er een plan van aanpak opgesteld voor het project. Het plan van aanpak is een belangrijk document, omdat dit beschrijft wat de opdracht inhoudt en hoe de opdracht wordt aangepakt. Alle belanghebbende partijen, in dit geval de opdrachtgever, de begeleider en de afstudeerder, zijn hierdoor op de hoogte van de aanpak van het project. Ook biedt het houvast en structuur bij de uitvoering van het project.

Een gedeelte van het document komt overeen met het afstudeerplan, aangezien de opdracht niet is veranderd. Echter is het plan van aanpak uitgebreider en gedetailleerder. Zo wordt er beschreven voor welke methodiek er is gekozen, waarbij meerdere methodieken zijn afgewogen. Aan de hand van deze methodiek is er een planning opgesteld. De gekozen methodiek, Open Up, schrijft onder andere een projectplan, een iteratie plan, een risicolijst en een Vision document voor. Deze documenten zijn niet als aparte documenten geschreven, maar de belangrijke elementen van deze documenten zijn opgenomen in het plan van aanpak. De keuze hiervoor wordt verderop in dit verslag toegelicht.

Het plan is voorgelegd aan de opdrachtgever en na de gegeven goedkeuring was er een goede bodem om verder te gaan met het project. In dit hoofdstuk wordt een gedeelte uit het plan van aanpak beschreven. Deze kennis is benodigd voor het lezen van de rest van het verslag. Het volledige plan van aanpak is te vinden in bijlage B.

3.1 Probleemstelling

Scholen zijn in eerste instantie gericht op het opleiden van leerlingen door middel van het aanbieden van theoretische kennis en het ontwikkelen van praktische vaardigheden. Docenten zijn echter ook veel tijd kwijt aan ondersteunende, administratieve processen zoals onder andere leerlingenregistratie. Er moet worden bijgehouden welke leerlingen er op welk tijdstip aanwezig zijn en zo nodig moeten er maatregelen worden getroffen, zoals het inlichten van de ouders bij langdurige afwezigheid. Door deze taken houdt de docent minder tijd over voor het daadwerkelijk overdragen van kennis.

Daarnaast is het klimaat in een klaslokaal vaak van onvoldoende kwaliteit. Zo kan er sprake zijn van een onaangename temperatuur of onvoldoende ventilatie, wat kan resulteren in hoofdpijn en vermoeidheid. Dit is vaak te wijten aan onvoldoende inzicht in het klimaat binnen de klas.

3.2 Doelstelling

De doelstelling van de opdracht is om een prototype/demonstratiemodel van een systeem te ontwikkelen, waarmee leerlingen zich kunnen aan- en afmelden en waarmee de aanwezigheid van de leerlingen en de kwaliteit van het klimaat binnen het klaslokaal inzichtelijk worden gemaakt voor een docent. Met dit demonstratiemodel wordt er inzichtelijk gemaakt wat de mogelijkheden zijn en het demonstratiemodel kan worden gebruikt om de interesse te wekken bij scholen, om zo het toekomstige product te vermarkten.

3.3 Opdrachtbeschrijving

Als antwoord op het probleem dat docenten te veel tijd kwijt zijn aan ondersteunende processen zoals onder andere leerlingenregistratie en er vaak te weinig inzicht is in het klimaat binnen het lokaal, is bij Alflex het idee ontstaan om één systeem te ontwikkelen voor scholen voor zowel de aanwezigheidsregistratie van leerlingen als de registratie van het klimaat binnen een klaslokaal. Voor beide functionaliteiten hoeft een school dan maar één enkel systeem aan te schaffen.

Een leerling moet zich elektronisch kunnen aan- en afmelden bij binnenkomst en verlaten van het klaslokaal. Dit moet kunnen gebeuren aan de hand van een apparaatje bij ingang van het klaslokaal, in het vervolg van dit verslag het *basisstation* genoemd. Hiervoor zal worden onderzocht welke techniek zich het beste leent voor het registreren van een leerling door middel van een kaart of een tag. Daarnaast moeten er in het lokaal verschillende metingen worden verricht omtrent het klimaat. Deze metingen moeten worden verricht door apparaatjes die in het lokaal op de juiste positie kunnen worden opgehangen, in het vervolg van het verslag de *meetstations* genoemd.

De docent moet op een smartphone kunnen raadplegen welke leerlingen er aanwezig zijn en welke leerlingen er missen. Afhankelijk van de gekozen techniek om een leerling te registreren, zal er een bepaald soort tag of kaart moeten worden gekoppeld aan een concrete leerling. Deze informatie moet worden opgeslagen op het basisstation. Hiervoor zal een desktopapplicatie of een mobiele applicatie moeten worden ontwikkeld om de tags te koppelen aan een leerling en de data in te laden op het basisstation. Ook moet de docent via een smartphone het klimaat in het klaslokaal kunnen monitoren. In het meest gunstigste geval moet er geautomatiseerd worden ingespeeld op de klimaatwaardes.

Dit systeem moet in eerste instantie zelfstandig kunnen opereren binnen de school, zonder dat er een verbinding naar de buitenwereld benodigd is. Wel moet er een mogelijkheid komen dit systeem uit te breiden met een optionele webserver, waar de gegevens op een duidelijke wijze kunnen worden gepresenteerd en eventueel geconfigureerd. Een visie voor de toekomst is dat de web applicatie vervolgens integreert met bestaande administratiesystemen binnen scholen.

3.4 Projectgrenzen

De focus van de afstudeeropdracht zal liggen op het systeem dat werkend is zonder webserver en het systeem zal een prototype zijn van het daadwerkelijke systeem. Er hoeft nog geen koppeling plaats te vinden met een database met roosterinformatie, maar het systeem zal werken met eenmalig ingevoerde versimpelde roosterdata dat enkel dient voor demonstratiedoeleinden van het prototype.

Indien er echter genoeg tijd blijkt te zijn zal er een prototype van het systeem worden ontwikkeld dat draait op een webserver, maar dit zal niet worden ingepland in de initiële planning. Het toekomstperspectief is dat de web applicatie geïntegreerd is met bestaande administratieve software binnen de school. Dit valt zeker buiten de scope van dit project.

Het geautomatiseerd opslaan van aanwezigheid van leerlingen kan een aantal vragen oproepen, op het gebied van privacy en bewaartermijnen. Onderzoek naar deze vragen valt buiten de scope van het project.

3.5 Business Case

Op dit moment bestaan er al systemen waarmee de aanwezigheid van leerlingen wordt geregistreerd, echter worden deze systemen nog niet veel toegepast op scholen. Ook bestaan er al systemen om het klimaat binnen een bepaalde ruimte te meten en inzichtelijk te maken. Alflex wil echter een combinatie van deze systemen maken en wil zichzelf ook meer gaan richten op deze

markt. Het uiteindelijke doel is om een goedkope oplossing te maken, om deze laag in de markt te zetten.

Door het ontwikkelen van een prototype van zo'n systeem, kan met dit prototype interesse worden gewekt bij scholen en kan er worden onderzocht of er ook daadwerkelijk vraag is naar zo'n systeem binnen de scholengemeenschap. Indien blijkt dat er behoefte is aan zo'n systeem, kan het prototype verder worden ontwikkeld naar een volwaardig product om deze vervolgens aan de man te brengen. Er kan dan worden gedacht aan integratie met bestaande administratieve software binnen scholen, of het bijhouden waar leerlingen zich precies bevinden binnen scholen. Ook zou dezelfde kaart/tag gebruikt kunnen worden voor het openen van kluisjes en het koppelen met andere apparaten.

De kosten voor het prototype liggen relatief laag. De opdracht wordt uitgevoerd door de afstudeerder gedurende zeventien weken, met een stagevergoeding per maand. Daarnaast zal er hardware moeten worden aangeschaft om het prototype te realiseren. De prijs van deze hardware zal binnen de enkele honderden euro's liggen.

De baten van het systeem zijn lastiger te bepalen. Het prototype kan worden ingezet om interesse te wekken bij scholen en om zo te kijken of er markt is voor zo'n systeem. Dit zal dus niet direct geld opleveren voor Alflex. Indien blijkt dat er een markt is voor het systeem, kan het prototype verder worden ontwikkeld tot een volwaardig systeem. Dat systeem kan dan wel worden verkocht en zal dan ook geld opleveren.

3.6 Op te leveren programmatuur

Het systeem kan opgedeeld worden in verschillende deelsystemen, waarbij voor elk deelsysteem programmatuur moet worden geschreven. De op te leveren programmatuur is als volgt:

- Programmatuur voor het basisstation
- Programmatuur voor de meetstations
- Programmatuur voor een desktopapplicatie en/of een Android applicatie met als doel het koppelen van tags/kaarten aan een leerling
- Een Android applicatie met als doel het uitlezen van de gegevens van het basisstation en de meetstations
- Indien er nog tijd over is een prototype van een applicatie die draait op een webserver, waarop alle gegevens worden opgeslagen en inzichtelijk worden gemaakt

3.7 Projectorganisatie

Het project wordt uitgevoerd door één persoon, namelijk de afstudeerder Rob Kleinee. Daarnaast zijn er nog vier personen betrokken bij het project: twee personen vanuit Alflex en twee personen vanuit de opleiding.

De twee personen vanuit Alflex zijn Thomas Wagenveld en Fred Lucas. Thomas Wagenveld fungeert als bedrijfsmentor, waarbij hij aanspreekpunt is vanuit de opleiding en de afstudeerder begeleidt met de opdracht. Fred Lucas is de opdrachtgever van de opdracht.

De twee personen vanuit de opleiding zijn Tony Andrioli en Adri Pronk. Tony Andrioli fungeert als begeleider bij de opdracht vanuit de opleiding. Adri Pronk is de tweede examinator. Tony Andrioli en Adri Pronk zullen beide de afstudeerder beoordelen.

4 Aanpak

Om structuur aan het project te geven is er gewerkt volgens een van te voren gespecificeerde methodiek met een bijbehorende planning. Een methodiek geeft houvast bij het uitvoeren van een project en schrijft een bepaalde aanpak voor met bijbehorende documenten. De indeling van de rest van dit verslag is gebaseerd op de gekozen methodiek. Dit hoofdstuk beschrijft deze keuze, de globale indeling van het project en de bijbehorende planning. Ook deze informatie komt uit het plan van aanpak.

4.1 Keuze ontwikkelmethodiek

Een project kan doorlopen worden volgens verschillende methodieken zoals RUP, SCRUM, XP en vele anderen. In deze paragraaf wordt beschreven welke methodiek er gebruikt werd en wordt de keuze hiervoor toegelicht.

Kenmerken project/eisen aan methodiek

Aan de hand van de kenmerken van het project en rekening houdend met eisen vanuit Alflex en de studie, zijn de volgende eigenschappen opgesteld waaraan de te kiezen ontwikkelmethodiek moet voldoen. Deze eigenschappen zijn vervolgens getoetst op verschillende methodieken. De opgestelde eigenschappen zijn als volgt:

- De methodiek moet geschikt zijn voor softwareontwikkeling.
- De methodiek moet ruimte bieden voor verrichten van onderzoek/analyse, aangezien gedurende dit project er verschillende technieken moeten worden geanalyseerd.
- Eisen moeten gedurende het project enigszins kunnen veranderen, naar mate er meer inzicht ontstaat in de verschillende deelsystemen. Ook kan een conclusie van een bepaalde analyse leiden tot veranderende systeemeisen.
- De methodiek bevat iteraties. Zo kunnen de verschillende deelsystemen iteratief ontwikkeld worden en zou er elke iteratie opnieuw gekeken kunnen worden naar de systeemeisen voor dat deelsysteem en kan er elke iteratie opnieuw een (korte) analyse worden uitgevoerd.
- (Relatief goed) solo uit te voeren.
- De methodiek moet ruimte bieden voor het schrijven van documentatie en het maken van softwareontwerpen. Vanuit zowel de studie als Alflex wordt er enige documentatie verwacht.
- Het is handig als er al enige ervaring aanwezig is met de methodiek.

Uiteenzetting methodieken

Aan de hand van de hierboven beschreven eigenschappen zijn er een aantal ontwikkelmethodieken onderzocht en bekeken. De verschillende methodieken zijn daarbij wel beperkt tot de meest bekende en degene waarvan al enige ervaring en/of kennis aanwezig was. Methodieken als DSDM en AUP zijn daardoor achterwege gelaten. De onderzochte methodieken zijn als volgt: Waterval, RUP, Open Up, SCRUM en XP.

Legenda vergelijking methodieken	
+	De methodiek bevat deze eigenschap
~	De methodiek bevat deels deze eigenschap
-	De methodiek bevat deze eigenschap niet of nauwelijks

Eigenschappen	Waterval	RUP	Open Up	SCRUM	XP
Geschikt voor softwareontwikkeling	+	+	+	+	+
Ruimte bieden voor onderzoek/analyse	+	+	+	~	~
Eisen moeten enigszins kunnen veranderen	-	+	+	+	+
Methodiek bevat iteraties	-	+	+	+	+
(Relatief goed) solo uit te voeren	+	-	~	~	-
Tijd voor schrijven documentatie en maken ontwerpen	+	+	+	~	-
Eigen ervaring met de methodiek	~	+	~	+	-

Tabel 1: Uiteenzetting methodieken

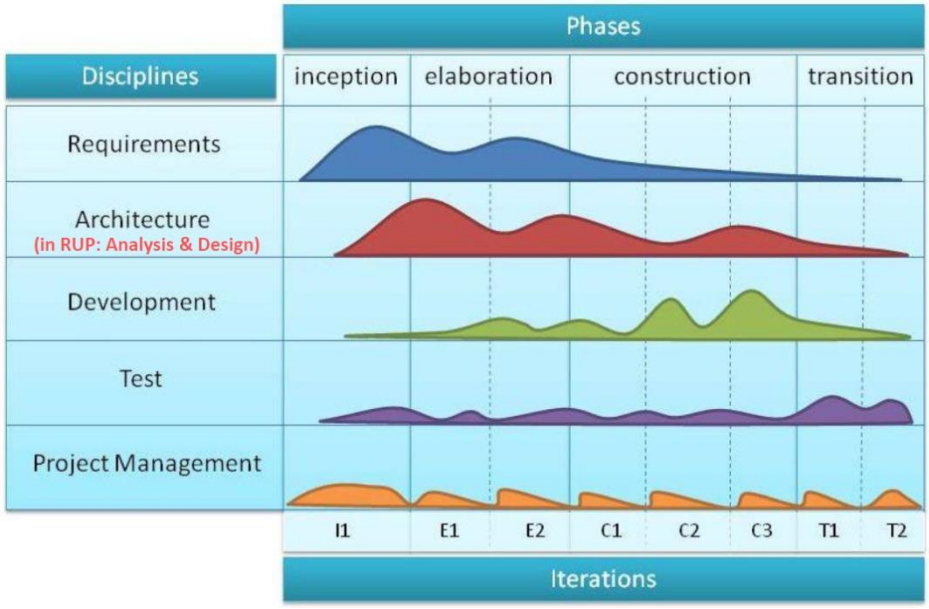
- De watervalmethodiek is relatief eenvoudig te plannen en te beheren, aangezien er een recht pad wordt bewandeld zonder iteraties. Door de afwezigheid van iteraties is er geen ruimte meer voor veranderende systeemeisen. Ook kan niet iteratief een nieuwe analyse worden gedaan voor een ander deelsysteem. Deze methodiek valt dus af. [3]
- RUP is een iteratieve ontwikkelmethodiek, waarbij een grote ruimte is voor het doen van analyse en onderzoek. Deelsystemen kunnen in verschillende iteraties worden doorlopen, waarbij opnieuw een analyse kan worden uitgevoerd. De methodiek staat open voor veranderende systeemeisen en geeft veel ruimte voor het inplannen van documentatie en het maken van ontwerpen. RUP is in beginsel echter vooral gericht op grote ontwikkelteams en op het bijhouden van veel documentatie rondom het projectmanagement, wat het complex maakt om het te beheersen. [4]
- Open Up lijkt erg sterk op RUP, aangezien RUP de bodem is voor Open Up. Echter heeft Open Up ook wat meer “behendigheid” overgenomen uit de Agile-methodieken. Zo worden de vier fases uit RUP doorlopen, maar zijn er per fase maar vijf in plaats van negen disciplines. Er wordt aangenomen dat de business modeling al is uitgevoerd en er is minder documentatie aanwezig voor het beheren van het project zelf. De methodiek is meer gespitst op kleinere ontwikkelteams. [5]
- SCRUM is een iteratieve en incrementele agile-methodiek, wat het erg geschikt maakt voor sterk wisselende of nog onbekende systeemeisen. Het is minder geschikt voor het uitvoeren van veel onderzoek, maar richt zich meer op het incrementeel ontwikkelen en opleveren van het softwareproduct. Hierbij wordt er weinig documentatie geschreven en ligt de nadruk erg op het opereren in kleinere maar nauw samenwerkende teams, waarbij er duidelijk vastgestelde rollen aanwezig zijn. Bij het individueel uitvoeren van SCRUM kan er een conflict in rollen voorkomen en de vergaderingen, waar SCRUM sterk op leunt, zijn enkel uitvoerbaar met meerdere personen. [6]
- XP is net als SCRUM een iteratieve en incrementele agile-methodiek. Veel SCRUM kenmerken zijn ook op XP van toepassing. XP streeft daarnaast nog meer naar eenvoud, waarbij er nog minder documentatie aanwezig is en het ontwerp van het softwaresysteem zo simpel mogelijk moet worden gehouden, waarbij niet zozeer gelet wordt op toekomstige uitbreidingen. Binnen XP is het gebruikelijk dat er met twee man tegelijkertijd achter dezelfde computer wordt geprogrammeerd. [7]

Keuze voor methodiek
De watervalmethodiek valt af doordat er geen aanwezigheid is van iteraties, waardoor er geheel niet kan worden ingespeeld op veranderende systeemeisen. XP valt af doordat de nadruk daar wel erg ligt op eenvoud en het schrijven van zo min mogelijk documentatie. SCRUM is op zich een goede kandidaat, maar op het gebied van documentatie staat het erg open voor een eigen invulling en wordt er geen vaste documentatie voorgeschreven. Ook is de methodiek erg gericht op een nauwe samenwerking in een team, met de verschillende rollen binnen het team, waardoor het lastiger is om deze methodiek individueel uit te voeren.

De twee methodieken die het meest geschikt lijken voor dit project zijn RUP en Open UP. Bij beide is er genoeg ruimte voor het uitvoeren van een analyse en wordt het project iteratief uitgevoerd. Binnen elke fase is het gebruikelijk documentatie en ontwerpen op te leveren. Open Up heeft de voorkeur doordat deze niet zozeer gericht is op een groot ontwikkelteam, wat het geschikter maakt om het project individueel uit te voeren. Ook is deze methodiek minder groot en complex, waar bij RUP weer te veel (onnodige) documentatie wordt opgeleverd. De uiteindelijke keuze is dus Open Up geworden.

4.2 Projectuitvoering aan de hand van Open Up

De methodiek Open Up bestaat, net zoals RUP, uit vier fases: de inception fase, de elaboration fase, de construction fase en de transition fase. Binnen deze fases worden vijf verschillende disciplines doorlopen. [8] De fases kunnen vervolgens ook weer iteratief worden doorlopen. Dit geheel wordt duidelijk gemaakt in Figuur 2. Volgens bron [5] zijn er nog twee disciplines te vinden in Open Up, namelijk *deployment* en *environment*. De eerstgenoemde is niet van belang voor dit project, omdat het product niet daadwerkelijk in gebruik gaat worden genomen, aangezien het nog een prototype betreft. Laatstgenoemde betreft de installatie van de benodigde tools en het vaststellen hoe het proces precies gaat verlopen. Deze discipline wordt voor nu ondergebracht onder de discipline *Project Management*, zodat in de basis nog steeds Figuur 2 wordt gebruikt.



Figuur 2: Verloop Open Up [8]

Het project zal dus ingedeeld worden naar de vier fases van Open Up, waarbij binnen elke fase de verschillende disciplines worden doorlopen. De hoeveelheid tijd dat per fase wordt besteed aan een discipline, verschilt per fase. In de bovenstaande figuur is een indicatie gegeven van de hoeveelheid tijd van een discipline per fase, dit kan in de praktijk echter anders uitvallen. Een planning van het

project aan de hand van Open Up, is te vinden in een later hoofdstuk. Wat nu volgt is een globale beschrijving van het verloop van het project.

In de inception fase zal het plan van aanpak worden opgesteld, inclusief planning, business case en risicolijst. In Open Up is het namelijk gebruikelijk om vroeg na te denken over mogelijke risico's van het project en om deze risico's zo snel mogelijk te verhelpen. In deze fase zal ook een groot deel van de eisen aan het systeem worden opgesteld en wordt de ontwikkelomgeving opgezet.

In de elaboration fase en de construction fase zal iets worden afgeweken van de gebruikelijke activiteiten binnen deze fases, zoals Open Up het voorschrijft. Gebruikelijk is om het grootste gedeelte van het ontwerp af te hebben in de elaboration fase. Echter bestaat het te maken systeem uit verschillende deelsystemen, waar voor elk van de deelsystemen een apart ontwerp zal worden gemaakt. Het ontwerp voor de smartphone applicatie zal een totaal andere insteek hebben dan het ontwerp voor een van de embedded applicaties. Ook zal bij het maken van een ontwerp er eerst kennis moeten zijn van de al beschikbare softwarecomponenten en API's voor de omgeving waarop de software komt te draaien. Er is voor gekozen om in de elaboration fase enkel een globaal ontwerp te maken dat de samenhang tussen de deelsystemen weergeeft. Verder zal de focus liggen op het analyseren van de te gebruiken technieken en communicatieprotocollen. Zo zal hier worden bekeken welke techniek er gebruikt gaat worden voor de identificatie van de leerlingen en zal worden onderzocht hoe er gecommuniceerd gaat worden tussen de stations en van het basisstation naar de smartphone.

De construction fase kan dan vervolgens opgedeeld worden in meerdere iteraties, waarbij in elke iteratie een deelsysteem wordt ontwikkeld. In zo'n iteratie worden dan de gedetailleerdere analyses uitgevoerd, specifiek voor het subsysteem dat in die iteratie ontwikkeld wordt. Er kan dan onder andere gekeken worden welke softwarecomponenten er al beschikbaar zijn en hoe die gebruikt kunnen worden. Met deze kennis kan vervolgens een ontwerp worden gemaakt voor dat subsysteem, wordt de software geschreven en daaropvolgend getest. Tot slot wordt er in de transition fase een gebruikerstest uitgevoerd en wordt het systeem opgeleverd.

4.3 Op te leveren documenten

De op te leveren documenten volgen uit de keuze voor Open Up. Binnen Open Up is het gebruikelijk om meerdere documenten te schrijven, gericht op de verschillende disciplines. In vergelijking met RUP wordt er doorgaans minder gedocumenteerd, maar nog steeds zijn er veel documenten gespecificeerd. Omdat dit een relatief gezien klein project is en er maar één uitvoerend persoon is, is er voor gekozen de verschillende documenten per discipline samen te nemen tot één document. Dit zorgt voor een kleiner aantal documenten, maar waarin wel de belangrijkste elementen uit de originele documenten worden behandeld. Zo worden bijvoorbeeld het *Use-Case* document en het *System-Wide Requirements* document samen genomen tot het Requirements document.

Documenten

- Plan van aanpak inclusief risicolijst
- Requirements document
- Analyse document
- Design document
- Testdocument

4.4 Risicoanalyse

Er zitten meerdere risico's verbonden aan dit project. Door middel van de risk list van Open Up is er een risicoanalyse uitgevoerd. Door inzicht te verkrijgen in deze risico's en door op voorhand maatregelen tegen deze risico's te nemen is de kans van slagen groter. De risico's zijn daarbij

geordend op prioriteit, die is berekend aan de hand van de kans van voorkomen en de impact. Twee risico's zijn te zien in Tabel 2. De volledige lijst met risico's is te vinden in het plan van aanpak.

Nr	Beschrijving	Kans van voorkomen (in %)	Impact (0 – 10)	Prioriteit (kans x impact)	Maatregel
3	Er is een risico dat het project niet af komt binnen de gestelde deadline, zodat niet het gehele product kan worden opgeleverd.	50	10	500	Door bij aanvang van het project een planning op te stellen, kan de tijd per activiteit worden ingepland, zodat er inzicht is in de hoeveelheid beschikbare tijd per activiteit. Daarnaast zullen de eisen aan het systeem worden geprioriteerd, zodat de belangrijkste eisen als eerste zullen worden geïmplementeerd.
4	De levertijd van de benodigde hardware zou lang kunnen zijn, waardoor er niet gewerkt kan worden totdat de hardware er is en zo de planning in het geding komt.	50	6	300	In een vroegtijdig stadium bepalen welke hardware er precies benodigd is en deze vervolgens zo snel mogelijk aanschaffen.

Tabel 2: Voorbeeld erkend risico

4.5 De planning

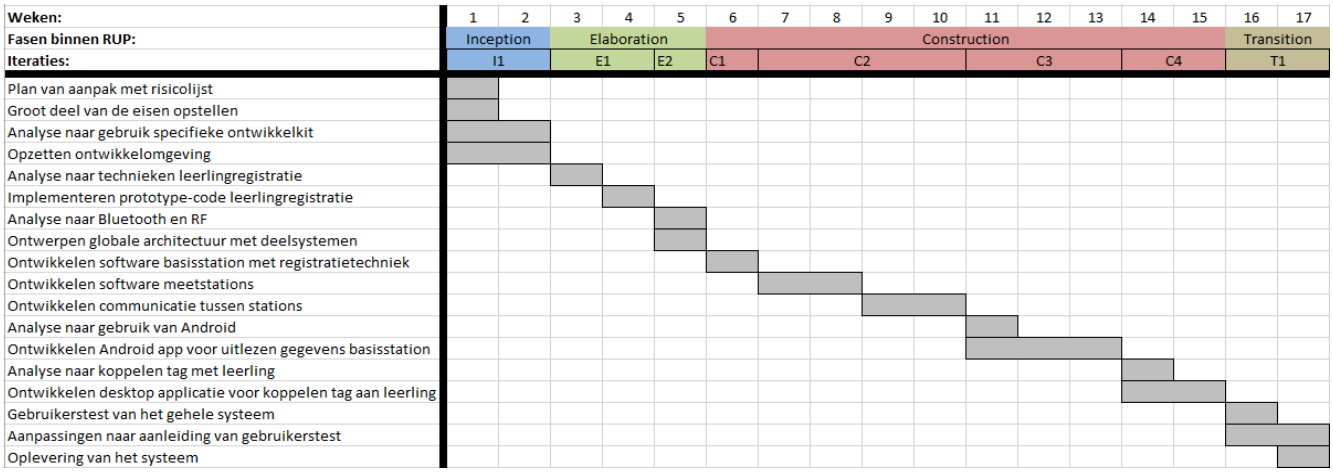
Aan de hand van de Open Up methodiek is er een globale planning van de iteraties opgesteld voor het project. Bij het maken van de planning is rekening gehouden met de risico's die zijn opgesteld in het plan van aanpak. Zoals gebruikelijk is bij Open Up, is de planning zo ingedeeld dat de grootste risico's zo veel mogelijk aan het begin van het project worden aangepakt. Zo zorgt risico nummer 4 uit Tabel 2 ervoor de analyse naar de te gebruiken ontwikkelkit al plaats vindt in de inception fase.

De inception fase bestaat uit één iteratie. De elaboration fase is opgedeeld in twee iteraties. In de eerste iteratie ligt de focus op de analyse naar identificatie-technieken voor leerlingregistratie en in de tweede iteratie ligt de focus op de analyse naar de te gebruiken communicatieprotocollen. De construction fase is zodanig ingedeeld, dat er in elke iteratie een deelsysteem wordt ontwikkeld. Deze deelsystemen komen voort uit de eerder beschreven op te leveren programmatuur. De transition fase bestaat uit één iteratie, waarin het product opgeleverd wordt.

In Tabel 3 is de globale planning te zien van het project en in Figuur 3 is deze planning vertaald naar een strokenplanning, voor een meer visuele weergave. Aan het eind van iedere iteratie is er een gedetailleerdere planning opgesteld voor de eerstvolgende iteratie. Deze planningen komen gedurende dit verslag aan bod.

Iteratie	Activiteiten	Geplande data
Inception Fase		
I1	Plan van aanpak met risicolijst	Week 1
	Groot deel van de eisen opstellen	Week 1
	Analyse naar gebruik specifieke ontwikkelkit	Week 1 / 2
	Opzetten ontwikkelomgeving	Week 1 / 2
Elaboration Fase		
E1	Analyse naar technieken leerlingregistratie	Week 3
	Implementeren prototype-code leerlingregistratie	Week 4
E2	Analyse naar Bluetooth en RF	Week 5
	Ontwerpen globale architectuur met deelsystemen	Week 5
Construction Fase		
C1	Ontwikkelen* software basisstation met registratietechniek	Week 6
C2	Ontwikkelen software meetstations	Week 7 / 8
	Ontwikkelen communicatie tussen stations	Week 9 / 10
C3	Analyse naar gebruik van Android	Week 11
	Ontwikkelen Android app voor uitlezen gegevens basisstation	Week 11 / 12 / 13
C4	Analyse naar koppelen tag met leerling	Week 14
	Ontwikkelen desktop applicatie voor koppelen tag aan leerling	Week 14 / 15
Transition Fase		
T1	Gebruikerstest van het gehele systeem	Week 16
	Aanpassingen naar aanleiding van gebruikerstest	Week 16 / 17
	Oplevering van het systeem	Week 17
*Ontwikkelen = ontwerpen, implementeren en testen		

Tabel 3: Planning van het project



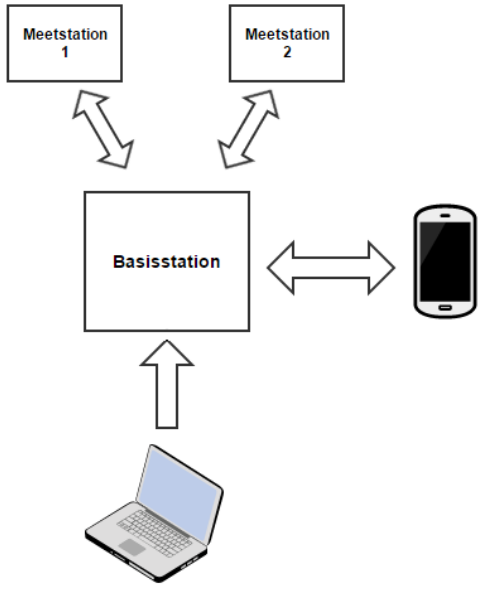
Figuur 3: Strokenplanning van het project

5 Inception fase

Het project is verlopen volgens de verschillende fases van de Open Up methodiek. In dit rapport worden de activiteiten en keuzes per fase besproken. In dit hoofdstuk wordt het proces gedurende de inception fase besproken. Deze fase bestond uit één iteratie waarin de eisen aan het systeem zijn achterhaald, er een keuze is gemaakt voor een ontwikkelkit en de ontwikkelomgeving is opgezet.

5.1 Indeling van het systeem

Uit de opdrachtomschrijving komt naar voren dat het te maken systeem bestaat uit verschillende onderdelen: een basisstation, de meetstations, een smartphone applicatie en een desktopapplicatie. Ter verduidelijk is Figuur 4 gemaakt. Op dit moment was nog onduidelijk wat de precieze eisen aan de onderdelen zijn en hoe deze onderdelen met elkaar moeten communiceren. Het doel was om dit gedurende de inception fase en elaboration fase duidelijk te krijgen.



Figuur 4: Eerste indeling van het systeem

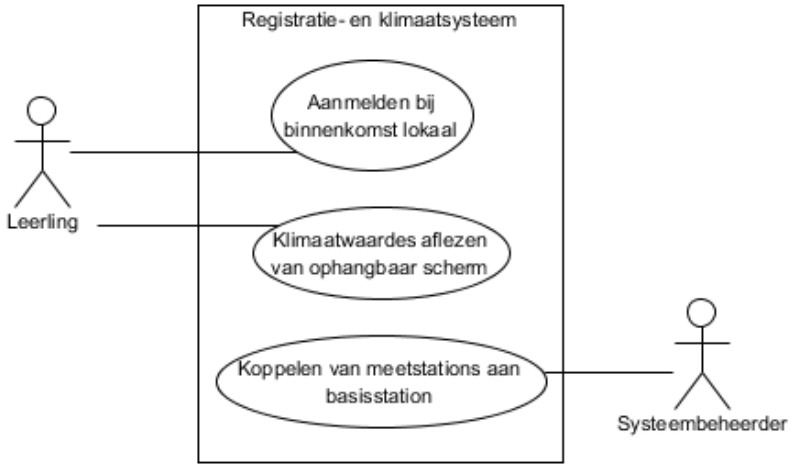
5.2 Achterhalen van de eisen

In de inception fase is een groot gedeelte van de eisen aan het gehele systeem achterhaald. Deze eisen zijn opgesteld aan de hand van gesprekken met de opdrachtgever. Hierbij zijn vragen aan de opdrachtgever opgesteld en voorgelegd. Daarnaast is er ook meegedacht met mogelijke eisen aan het systeem. Doordat de opdrachtgever geen kennis op technisch vlak heeft, zijn er ook gesprekken gevoerd met de begeleider, die op dit vlak wel kennis heeft en zo ook nog de nodige (niet-functionele) eisen stelde. De opdrachtgever heeft een rijke hoeveelheid aan ideeën over waar het systeem in de toekomst naar uit kan groeien. Het is echter door de beperkte tijd niet mogelijk om al deze ideeën uit te werken en daardoor was het belangrijk om een duidelijke scope vast te stellen voor dit project. Binnen deze scope zijn de eisen van het prototype opgesteld. Toch is er ook een aantal eisen opgesteld aan het systeem die waarschijnlijk niet meer haalbaar zijn binnen dit project, zodat beter bekend is waar het systeem uiteindelijk naar toe moet groeien. Met het ontwerp van het systeem kan dan wel rekening worden gehouden met toekomstige uitbreidingen.

Bij het verwoorden van de eisen is het systeem opgedeeld in de volgende deelsystemen: het registratie- en klimaatsysteem, de mobiele applicatie, een desktopapplicatie en een webapplicatie. In dit stadium van het project waren nog niet alle eisen aan de subsystemen geheel duidelijk. De eisen

zijn dan ook iteratief aangepast gedurende het project. In latere iteraties is er steeds weer een gesprek gevoerd met de opdrachtgever om de eisen aan een bepaald subsysteem te herzien. Door nieuw opgebouwde kennis kwamen er weer nieuwe eisen aan het licht, die in een vroeger stadium over het hoofd waren gezien. Al deze eisen zijn beschreven in een requirements document. Het gehele document is te vinden in bijlage C. In deze paragraaf worden enkele delen uit dit document toegelicht, om een indruk te geven hoe de eisen zijn verwoord.

5.2.1 Use-cases



Figuur 5: Use-case diagram - Mobiele Applicatie

In het requirements document staan use-cases met bijbehorende use-case scenario's beschreven, opgedeeld per subsysteem. De use-cases zijn gemaakt om zo de functionele eisen aan het systeem vast te leggen, uit het oogpunt van de gebruiker. Hierbij wordt een bepaald scenario uitgewerkt voor een bepaald type gebruiker. Deze use-cases zijn opgesteld aan de hand van de eerdergenoemde gesprekken met de opdrachtgever. Vervolgens zijn ze voorgelegd aan de opdrachtgever, waarna er weer aanpassingen werden gedaan.

De use-case diagrammen en use-case scenario's zijn opgesteld volgens de UML-notatie. [9] In Figuur 5 is een voorbeeld van een van de use-case diagrammen gegeven en twee bijbehorende use-case scenario's zijn te vinden in onderstaande tabellen. De complete lijst van de use-cases is te vinden in het requirements document.

Aanmelden bij binnenkomst lokaal	
Actoren	Leerling
Aannamen	Leerling bezit een kaart/tag
Beschrijving	Op het moment dat de leerling een klaslokaal binnenkomt, zal diegene zichzelf identificeren met een kaart/tag. Het systeem maakt kenbaar dat de leerling is aangemeld.
Uitzonderingen	[1] De gebruikte kaart/tag is niet bekend bij het systeem. Het systeem maakt kenbaar dat identificatie is mislukt. [2] De leerling hoort op dit moment niet in het lokaal te zijn. Het systeem maakt kenbaar dat identificatie is mislukt. [3] De leerling is al aangemeld. Het systeem maakt kenbaar dat de leerling al is aangemeld.
Resultaat	De leerling is geregistreerd als aanwezig.

Tabel 4: Use-case scenario - Aanmelden bij binnenkomst lokaal

Koppelen van meetstations aan basisstation	
Actoren	Systeembeheerder
Aannamen	In bezit van ten minste een basisstation en drie meetstations.
Beschrijving	De systeembeheerder koppelt het eerste meetstation aan het basisstation. De systeembeheerder koppelt het tweede meetstation aan het basisstation. De systeembeheerder koppelt het derde meetstation aan het eerste meetstation.
Uitzonderingen	-
Resultaat	De twee direct aan het basisstation gekoppelde meetstations versturen de gemeten klimaatwaardes door naar het basistation. Het derde meetstation stuurt de gemeten klimaatwaardes door naar het tweede klimaatstation, dat vervolgens deze informatie weer doorstuurt naar het basisstation.

Tabel 5: Use-case scenario - Koppelen van meetstations aan basisstation

5.2.2 Pakket van eisen

De eerder beschreven use-cases zijn vervolgens vertaald naar de functionele eisen van het systeem. Bepaalde use-cases zijn daarbij uitgewerkt tot meerdere afzonderlijke functionele eisen. Daarnaast zijn er ook niet-functionele eisen opgesteld waar het systeem aan moet voldoen, opgedeeld in een aantal categorieën. Al deze eisen zijn opgedeeld naar een bepaalde prioriteit. Zo werd duidelijk welke eisen het eerst zouden worden geïmplementeerd en welke eisen alleen aan bod zouden komen als er tijd voor zou zijn.

Prioritering

Voor het indelen van de eisen is gebruik gemaakt van de MoSCoW-methode [10]. De eisen zijn naar de volgende prioriteiten ingedeeld:

- Must haves: deze eisen moeten zeker in het eindresultaat voorkomen.
- Should haves: deze eisen zijn zeer gewenst, maar zonder is het systeem nog wel bruikbaar.
- Could haves: deze eisen komen alleen aan bod indien er nog tijd over is.
- Won't haves: deze eisen zullen in dit project niet aan bod komen, maar kunnen wel interessant zijn voor de toekomst.

Categorisatie

Met Open Up is het gebruikelijk de eisen aan het systeem onder te verdelen in verschillende categorieën, volgens het FURPS+ model. [5] De volgende categorieën zijn te onderscheiden:

- Functionality (functioneel)
- Usability (bruikbaarheid)
- Reliability (betrouwbaarheid)
- Performance (prestaties)
- Supportability (onderhoudbaarheid/aanpasbaarheid)

Optioneel:

- Design constraints (ontwerpeisen)
- Implementation (implementatie)
- Interface
- Physical (eisen aan fysieke apparaat)

In de volgende tabellen zijn enkele functionele en niet-functionele eisen gegeven. De volledige lijst met eisen is te vinden in het requirements document.

Functionele eisen registratie- en klimaatsysteem

Nr	Beschrijving eis	Prioriteit
F1.1	Een leerling moet zich kunnen identificeren en zo aanmelden bij het systeem door middel van een kaart en/of tag.	Must
F1.2	Informatie over welke leerling zich op welk tijdstip heeft aangemeld, moet worden opgeslagen op het basisstation.	Must
F1.3	Demonstratiedata met de lesblokken die gedurende de dag op een tijdstip worden gegeven in het lokaal, moet kunnen worden ingeladen op het basisstation.	Must
F1.4	Demonstratiedata met informatie over de leerlingen, gekoppeld aan een tag-id en een lesblok, moet kunnen worden ingeladen op het basisstation.	Must
F1.5	De temperatuur moet worden gemeten door een meetstation.	Must
F1.6	De koolstofdioxide moet worden gemeten door een meetstation.	Must
F1.7	De luchtvochtigheid moet worden gemeten door een meetstation.	Must
F1.8	De gemeten klimaatwaardes moeten door een meetstation worden doorgestuurd naar het basisstation.	Must
F1.9	Elk halfuur moet de huidige status van de gemeten klimaatwaardes (F1.5, F1.6, F1.7) worden opgeslagen op het basisstation.	Must
F1.10	De meetstations moeten direct of indirect (via een ander meetstation) kunnen worden gekoppeld aan het basisstation.	Must
F1.11	De huidige klimaatwaardes (F1.5, F1.6, F1.7) moeten worden getoond op een scherm aan de muur.	Could
F1.12	Er moet geautomatiseerd worden ingespeeld op de gemeten klimaatwaardes, door het regelen van de thermostaat of het openen van een raam.	Won't
F1.13	De gemeten klimaatwaardes (F1.5, F1.6, F1.7) moeten worden verstuurd naar een centrale webserver.	Won't
F1.14	Informatie over welke leerling zich op welk tijdstip heeft aangemeld, moet worden verstuurd naar een centrale webserver.	Won't

Tabel 6: Functionele eisen - Registratie- en klimaatsysteem

Betrouwbaarheid

Nr	Beschrijving eis	Prioriteit
B1	Bij stroomuitval of uitzetten van het registratie- en klimaatsysteem moet de opgeslagen data (aanwezige/afwezige leerlingen en klimaatregistratie) bewaard blijven.	Must

Tabel 7: Eisen betrouwbaarheid

Implementatie

Nr	Beschrijving eis	Prioriteit
I1	Het registratie- en klimaatsysteem moet gebruik maken van een Nordic nRF51 SoC.	Should
I2	De mobiele applicatie moet worden ontwikkeld voor Android	Must
I3	De mobiele applicatie moet kunnen draaien op Android versie 4.3 en hoger	Must
I4	De mobiele applicatie moet communiceren met het basisstation door middel van Bluetooth.	Must
I5	De software voor het registratie- en klimaatsysteem moet worden geïmplementeerd in een Eclipse ontwikkelomgeving, met de GCC-compiler.	Must

Tabel 8: Eisen Implementatie

5.3 Keuze voor specifieke ontwikkelkit

Een van de eisen aan het registratie- en klimaatsysteem is dat er gebruik moet worden gemaakt van een nRF51 SoC (System on Chip), in de basis een Cortex-M0. Het idee is dat verschillende losse stations met elkaar moeten kunnen communiceren om zo data uit te wisselen en er moet gecommuniceerd kunnen worden met een smartphone. De nRF51 bezit standaard functionaliteit voor een aantal RF-technieken, terwijl daar zeer weinig energie voor benodigd is. Dit zorgt ervoor dat de losse stations in potentie gebruik zouden kunnen maken van een batterijtje en ze daarop een langere tijd kunnen werken.

Om software te kunnen schrijven voor de nRF51 is een ontwikkelkit nodig. Dit is de hardware waarop de SoC is bevestigd, waarbij de pinnen van de SoC zijn verbonden met connectoren. Er is gezocht naar een ontwikkelkit op de officiële website van de fabrikant van de SoC en op websites die ontwikkelkits verkopen. Het bleek dat er verschillende ontwikkelkits beschikbaar zijn met daarop een nRF51 SoC, zodat er een keus moest worden gemaakt voor het gebruik van één van deze kits.

Er werden drie veel gebruikte kits gevonden: de nRF51 DK, de RFduino en de nRF51822-mKIT. Daarnaast zijn er een aantal andere ontwikkelkits gevonden met daarop een nRF51 SoC, maar die zijn een stuk minder bekend en er is dan ook minder documentatie aanwezig. Daarom is er voor gekozen de keuze beperkt te houden tot de drie veel gebruikte kits. Niet iedere ontwikkelkit ondersteunt dezelfde RF-technieken en elke kit heeft zijn eigen voor- en nadelen.

Om tot een keus te komen zijn er een tweetal eisen opgesteld waar de ontwikkelkit aan moet voldoen. Vervolgens zijn de communicatiemogelijkheden van de nRF51 SoC onderzocht en is onderzocht wat de mogelijkheden zijn van de ontwikkelkits met betrekking tot deze communicatiemogelijkheden. De ontwikkelkits zijn uiteengezet, met elk hun voor- en nadelen en tot slot is er een keuze gemaakt. Een uitgebreide beschrijving hiervan is te vinden in het analyse document, te vinden in bijlage D. Hier blijft het beperkt tot een beknoptere uiteenzetting.

5.3.1 Eisen aan ontwikkelkit

- Voor het basisstation moet er tegelijkertijd communicatie met de meetstations en met een smartphone kunnen plaatsvinden.
- De code die wordt geschreven voor de ontwikkelkit moet ook nog bruikbaar zijn indien het prototype in de toekomst verder wordt uitgewerkt.

5.3.2 Communicatiemogelijkheden van de nRF51 SoC

De nRF51 SoC is er in twee varianten: nRF51422 en nRF51822. De eerstgenoemde bezit standaard de communicatiemogelijkheden ANT+, Bluetooth Smart en het Gazell 2.4 Ghz protocol. De laatstgenoemde ondersteunt Bluetooth Smart en het Gazell 2.4 Ghz protocol. [11]

Bluetooth Smart

Bluetooth smart (voorheen Bluetooth Low Energy genoemd) is een technologie waarmee tussen verschillende apparaten kan worden gecommuniceerd, terwijl daarbij zeer weinig energie nodig is. Bluetooth Smart wordt gebruikt in smartphones en in apparaatjes zoals fitness sensoren, horloges en lampen. Het protocol wordt ondersteund door Android 4.3 of hoger en door IOS 5 of hoger.

Apparaten die over bluetooth communiceren kunnen verschillende rollen innemen: [12]

- Broadcaster: In deze rol zal het apparaat data adverteren naar de omgeving, zodat iedereen deze data kan oppikken. Er zal geen connectie opgezet worden met een ander apparaat.
- Observer: In deze rol is het apparaat het tegenovergestelde van een broadcaster. Het vangt data op van adverterende apparaten in de omgeving.

- Pheripheral: In deze rol zal het apparaat niet zelf een connectie initiëren, maar wacht het totdat het wordt aangesproken door een ander apparaat. Een voorbeeld is een temperatuursensor die kan worden uitgelezen door een smartphone.
- Central: In deze rol is het apparaat het centrale aanspreekpunt voor andere bluetooth apparaten die fungeren als pheripheral. Als central zal het apparaat een connectie initiëren naar meerdere pheripherals.

ANT
Het ANT-protocol is een tegenhanger van Bluetooth Smart. Het protocol wordt vooral gebruikt in combinatie met fitness gadgets. Ook met dit protocol kan er tussen apparaten worden gecommuniceerd, terwijl er zeer weinig energie voor nodig is. Het protocol is een gemakkelijker protocol dan Bluetooth Smart en ondersteunt meer typologieën dan Bluetooth Smart. Verder zijn de mogelijkheden nagenoeg gelijk. ANT wordt ondersteund in Android 3.1 of hoger. iPhone biedt nog geen standaard ondersteuning voor ANT, maar er zijn dongles beschikbaar als opzetstuk zodat er gewerkt kan worden met ANT. [13] Dit is dus echter wel minder gebruiksvriendelijk dan Bluetooth Smart. Bij het ontwikkelen van een nieuw product kan er op dit moment voor de communicatie met een smartphone beter gekozen worden voor Bluetooth Smart. Daarmee worden namelijk alle nieuwe Android smartphones en iPhones ondersteund.

Nordic Gazell 2.4 GHz protocol
Gazell is een protocol van Nordic, de fabrikant van de nRF51, voor het opzetten van een draadloze verbinding tussen één host en maximaal acht devices die ook dit protocol ondersteunen. Het nadeel van dit protocol is dat het zeer slecht gedocumenteerd is en er weinig voorbeelden van zijn te vinden.

Softdevices
Nordic heeft de verschillende Bluetooth en ANT protocol stacks ondergebracht in *Softdevices*. Deze binaire bestanden kunnen worden gedownload en op de chip worden geflashed, waarna ze door de programmeur gebruikt kunnen worden. Elke softdevice heeft andere eigenschappen en de twee verschillende SoC's nRF51422 en nRF51822 bieden niet beide ondersteuning voor alle softdevices. De ondersteuning is te zien in Tabel 9. [11]

Type SoC	**Softdevices**				
	S110	*S120*	*S130*	*S210*	*S310*
nRF51422	+	+	+	+	+
nRF51822	+	+	+	-	-

Tabel 9: Ondersteuning softdevices per SoC

Eigenschappen van de softdevices:
- S110: Bluetooth Smart als peripheral rol
- S120: Bluetooth Smart central rol
- S130: Bluetooth Smart met alle verschillende Bluetooth rollen, waarbij het mogelijk is tegelijkertijd zowel te fungeren als central en als peripheral
- S210: ANT protocol stack
- S310: Zowel Bluetooth als de ANT protocol stack

5.3.3 Keuze voor ontwikkelkit
De nRF51822-mKIT had als voordeel dat deze kit ondersteuning biedt aan de mbed ontwikkelomgeving; een online ontwikkelomgeving met uitgebreide libraries voor embedded projecten. [14] Echter biedt de nieuwere nRF51 DK ook ondersteuning voor deze ontwikkelomgeving, waardoor de nRF51822-mKIT is achterhaald. Verder biedt de nRF51822-mKIT geen meerwaarde tegenover de nRF51 DK.

Het voordeel van de RFduino, een kleine en goedkope Arduino met de nRF51822 SoC, is dat applicaties snel kunnen worden ontwikkeld door het gebruik van de bijbehorende libraries. Dit vergemakkelijkt en versnelt het ontwikkelproces enorm. Het nadeel hiervan is dat er minder vrijheid is bij het ontwikkelen van applicaties, aangezien men dan gebonden zit aan de beperkingen van de libraries. Ook kan de code dan niet zomaar geport worden naar een ander bord met een nRF51 SoC. Indien vanuit Alflex in de toekomst het prototype dan verder ontwikkeld wordt en hiervoor binnen het bedrijf een eigen bord wordt ontwikkeld met een nRF51 SoC, is de geschreven code niet zomaar te gebruiken. [15]

Een ander nadeel is dat enkel de S110 SoftDevice wordt ondersteund door de RFduino, waardoor de RFduino geen centrale rol kan vervullen bij Bluetooth communicatie en er geen ANT-protocol aanwezig is. Hierdoor kan de RFduino geen centrale rol vervullen bij de communicatie met de meetstations. Er is wel een vorm van het Gazell-protocol aanwezig, maar dit protocol is slecht gedocumenteerd en het is onduidelijk of dit protocol gelijktijdig met Bluetooth gebruikt kan worden.

De nRF51 DK lijkt een betere keuze, aangezien eerdergenoemde nadelen van de RFduino niet aan de orde zijn bij de nRF51 DK. Met deze kit is er het meeste vrijheid bij het ontwikkelen van een applicatie en de geschreven code zou relatief gemakkelijk kunnen worden geport naar een ander bord met een nRF51 SoC. Alle softdevices worden ondersteund, waardoor er volledige vrijheid is in het gebruik van de juiste RF-techniek. Echter is er wel minder snel werkende code te produceren doordat er dichter op de hardware wordt geprogrammeerd en er geen gebruik kan worden gemaakt van uitgebreide libraries. [11] Er is gekozen voor het gebruik van de nRF51 DK.

5.4 Opzetten van de ontwikkelomgeving voor de nRF51 DK

Voor de ontwikkeling van software voor een nRF51 DK kan er gebruik worden gemaakt van vier verschillende toolchains: KEIL, IAR, GCC en de toolchain van mbed. Een eis vanuit Alflex is dat er gebruik moet worden gemaakt van de gratis GCC toolchain in combinatie met Eclipse. KEIL en IAR brengen hoge licentiekosten met zich mee en bij het gebruik van mbed zit de verdere ontwikkeling vast aan de mbed ontwikkelomgeving met bijbehorende libraries. Bij de ontwikkeling van de software voor deze SoC kan gebruik worden gemaakt van de nRF51 SDK.

Helaas is de documentatie vanuit Nordic niet altijd even uitgebreid, zodat het opzetten van een ontwikkelomgeving met Eclipse en GCC voor de nRF51 een hoop tijd heeft gekost. Er is enkel officiële documentatie aanwezig voor het opzetten van het geheel in KEIL. Aan de hand van verschillende incomplete tutorials van derden is het uiteindelijk gelukt om een van de voorbeeldprogramma's te compileren en te flashen vanuit Eclipse. Vervolgens is er een eigen project gemaakt, waarbij de makefile van het voorbeeldprogramma is gemodificeerd.

Als ontwikkeltaal was er een keus te maken tussen C en C++. Bij Alflex wordt er voor ARM-processoren ontwikkeld in C++. Daarnaast kan de software in C++ beter worden gestructureerd, doordat er object georiënteerd kan worden gewerkt. De programmatuur is dan beter onderhoudbaar en uitbreidbaar. Er is dan ook gekozen voor het gebruik van C++ als ontwikkeltaal.

5.5 Evaluatie inception fase

Deze fase bestond uit één iteratie waarin de eisen aan het systeem zijn achterhaald, er een keuze is gemaakt voor een ontwikkelkit en de ontwikkelomgeving is opgezet.

Alle doelen die gesteld waren aan de inception fase zijn behaald. Het plan van aanpak is afgerond, zodat er een goede bodem aanwezig is voor de rest van het project. In grote lijnen zijn de systeemeisen achterhaald, zodat er voor dit moment genoeg duidelijkheid was over wat het systeem moet kunnen. Verder is er een keuze gemaakt voor een ontwikkelkit. De mogelijkheden van deze kit zijn bekeken en er is geconstateerd dat deze kit een goede basis biedt voor het te maken systeem. Er is vastgesteld dat er meerdere opties aanwezig zijn voor de communicatie tussen de stations en naar een smartphone, zodat er op dat punt een grote kans van slagen is. Een analyse naar de communicatieprotocollen en een keuze voor een van de protocollen kan worden gemaakt in de elaboration fase.

Ook is er een aantal risico's aangepakt uit de risicolijst die is opgesteld aan het begin van het project. Zo is risico nummer 4 uit Tabel 2 aangepakt door vroegtijdig de ontwikkelkits aan te schaffen. Ook risico nummer 3 is aangepakt, door een planning op te stellen en de requirements aan het systeem te prioriteren.

Er stonden twee weken gepland voor de inception fase, maar deze fase was een paar dagen eerder afgerond dan verwacht. De elaboration fase begon daardoor iets eerder dan gepland.

6 Elaboration fase 1: Analyse identificatietechnieken

De elaboration fase is opgedeeld in twee iteraties, waarvan de eerste in het teken staat van de analyse naar identificatietechnieken voor de leerlingenregistratie en de tweede iteratie in het teken staat van de analyse naar de communicatie tussen de stations en de smartphone. In dit hoofdstuk wordt het proces toegelicht dat is doorlopen gedurende de eerste iteratie. De planning voor deze iteratie is te zien in Figuur 6. De inception fase was eerder afgerond dan verwacht, dus deze iteratie begon eind week 2.

Figuur 6: Planning elaboration fase iteratie 1

6.1 Identificatietechnieken

Het is de bedoeling dat het te maken systeem een mogelijkheid bevat waarmee leerlingen zich kunnen aanmelden/identificeren bij het binnentreden van een klaslokaal, waarna geregistreerd wordt dat deze leerlingen aanwezig zijn. Voor de realisatie hiervan kan gebruik worden gemaakt van verschillende soorten technieken, elk met zijn eigen voor- en nadelen, zodat er een keus moest worden gemaakt in het gebruik van één van deze technieken.

Om tot een goede keuze te komen is er eerst onderzocht welke identificatietechnieken er gebruikt worden in bestaande systemen. Hierbij is gekeken naar systemen met een uiteenlopend doel: van systemen die dienen tot identificatie van producten in een supermarkt, tot systemen die dienen tot persoonsbewijzen. Ook is er onderzocht welke systemen er al ontwikkeld zijn voor identificatie van leerlingen in scholen, en welke identificatietechnieken er door die systemen gebruikt worden. Aan de hand van deze bronnen zijn de meest gebruikte technieken op een rijtje gezet en geanalyseerd. Vervolgens zijn er eisen gesteld aan de te gebruiken techniek. Een aantal eisen is daarbij voortgekomen uit de eerdere gesprekken met de opdrachtgever en andere eisen zijn voortgekomen uit de opgedane kennis van de onderzochte technieken. Bij het maken van de keuze is ook rekening gehouden met toekomstperspectieven van het te maken systeem. Tot slot zijn de technieken tegen elkaar afgewogen en is er een keus gemaakt. In het analyse document staan de onderzochte bestaande systemen beschreven, evenals de meest gebruikte technieken en de afweging van de technieken. In deze paragraaf wordt hier een beknoptere samenvatting van gegeven.

6.1.1 Zoeken van geschikte bronnen

Tijdens dit project zijn er verschillende soorten analyses uitgevoerd. Zo zijn de eerder besproken ontwikkelkits onderzocht, is er een analyse uitgevoerd naar verschillende identificatietechnieken en zijn de te gebruiken communicatieprotocollen onderzocht. Bij het zoeken naar de juiste informatie is gebruik gemaakt van verschillende soorten bronnen:

- Wikipedia is niet gebruikt als primaire bron, omdat er ook een hoop foutieve informatie op staat, maar is in sommige gevallen wel gebruikt als startpunt, doordat er een hoop globale informatie is te vinden. Uit deze informatie zijn vervolgens zoektermen gehaald, waarmee gezocht is naar gedetailleerdere informatie in betrouwbaardere bronnen.
- Indien beschikbaar is er gebruik gemaakt van officiële documentatie of de websites van fabrikanten van producten. Voor communicatieprotocollen is er informatie gehaald uit officiële specificaties van het protocol.

- Bepaalde websites richten zich op het verschaffen van informatie over één soort techniek, waarbij auteurs specialisten zijn op het betreffende gebied. Bij andere websites is de betrouwbaarheid minder hoog, maar staat bepaalde informatie duidelijker beschreven. De informatie van deze websites is ook gebruikt, in combinatie met officiëlere (maar vaak minder eenvoudig leesbare) bronnen.
- Via de Haagse Hogeschool is er gebruik gemaakt van verschillende databanken, waar artikelen zijn te vinden die normaal gesproken enkel tegen betaling zijn te lezen. Zo zijn er papers gebruikt waarin informatie wordt verschaft over een onderwerp, vaak geschreven door personen vanuit een academische instelling. De betrouwbaarheid van deze informatie is hoger dan die van reguliere websites.
- Tot slot is er gebruik gemaakt van de boeken en de sheets die zijn aangereikt vanuit de opleiding.

6.1.2 De meest gebruikte identificatietechnieken

Barcodes

Barcodes worden al jaar en dag gebruikt bij vooral het scannen van producten, maar ook personen kunnen er mee worden geïdentificeerd. Originele barcodes hebben een opslagcapaciteit van 20 karakters. Nieuwere vormen, bestaande uit 2D-patronen zoals de QR-code, kunnen tot wel 7000 karakters opslaan. Barcodes zijn goedkoop te produceren, echter is het scannen niet erg gebruiksvriendelijk, is een barcode eenvoudig na te maken en is een barcode snel beschadigd of vies. [16]

Magnetische strip

Een magnetische strip kaart is een kaart met aan een kant een magnetische strip, waarop data gezet kan worden door het wijzigen van het magnetische veld. Deze informatie kan vervolgens worden uitgelezen door de kaart langs een magneetkop te bewegen. Zo'n kaart wordt vaak gebruikt in creditcards, vervoersbewijzen of kaarten ter identificatie van een persoon als beveiligingsmaatregel bij binnentreden van een gebouw. De kaarten zijn goedkoop te produceren, maar slijten gedurende het gebruik. [17]

Smart card

Een smart card is een kaart met daarop een kleine microcontroller of een andere vorm van intelligentie, met een geheugenchip. Er zijn twee varianten: een variant met contactpunten en een contactloze variant. De eerste variant moet handmatig in een lezer worden gestoken, zodat de contactpunten juist worden uitgelijnd. De contactloze variant communiceert met de lezer via de RFID-techniek die verderop wordt besproken. De kaarten zijn in staat tot het vasthouden van een relatief grote hoeveelheid data, waarbij de data vaak versleuteld wordt opgeslagen en er authenticatie-mogelijkheden zijn als extra beveiliging. De kaart wordt dan ook veel gebruikt bij betalingen en identiteitskaarten. [18]

Biometrie

Door middel van biometrie kan een persoon worden geïdentificeerd. Daarbij zijn er verschillende soorten van biometrie, zoals identificatie van een oog, een vingerafdruk, gezichtsherkenning, stemherkenning en nog vele andere vormen. Deze vorm van identificatie wordt steeds meer gebruikt in bijvoorbeeld smartphones en laptops. Het voordeel is dat de persoon geen pasje of iets dergelijks bij zich hoeft te dragen. Het nadeel is dat zulke systemen vaak duur zijn en de identificatie van een persoon duurt relatief lang. [19]

RFID
Het hoofddoel van RFID is het identificeren of traceren van een persoon of object. Dit gebeurt aan de hand van een ID die staat opgeslagen op een RFID-tag. Sommige RFID-tags bezitten de mogelijkheid om een kleine hoeveelheid data op te slaan, zodat naast een ID ook een andere vorm van data kan worden opgeslagen als dat gewenst is. De tag communiceert via RF met een lezer van de tag, waarbij de afstand varieert van enkele centimeters tot wel 100 meter. Het ID van de tag is vervolgens in een achterliggend systeem gekoppeld aan de betreffende persoon of het object. Het lezen van een RFID-tag gaat zeer snel en er kunnen meerdere tags tegelijkertijd worden uitgelezen. Een RFID-tag in de vorm van een sleutelhanger of kaart gaat niet snel kapot en is veiliger dan bijvoorbeeld een barcode. Afgelopen jaren is de prijs van een RFID-tag gedaald, waardoor de tags die opereren op korte afstand erg goedkoop zijn geworden. [20] [21]

NFC
NFC bouwt voort op de RFID-techniek, waarbij NFC als een specialisatie van RFID kan worden gezien. Het opereert op dezelfde frequentie als een van de drie frequentiebanden van RFID en de standaarden zijn gebaseerd op RFID-standaarden. NFC is ontworpen als beveiligde vorm van RFID, waarbij data versleuteld wordt opgeslagen. Anders dan bij een RFID-tag meestal het geval is, wordt er niet alleen gebruik gemaakt van de ID van de chip, maar wordt er ook daadwerkelijk belangrijke informatie opgeslagen. In tegenstelling tot RFID kunnen twee NFC-chips twee kanten op communiceren en zo data uitwisselen. De maximale afstand tussen twee NFC-chips bedraagt zo'n 10 cm. De techniek zit verwerkt in de nieuwste smartphones. [22] [23]

6.1.3 Toelichting keuze voor RFID
Bij een aantal van de besproken technieken is het mogelijk om data op het draagbare medium op te slaan, waarbij in sommige gevallen de data kan worden versleuteld en er authenticatiemogelijkheden zijn. Indien er een draagbaar medium gebruikt zou worden voor identificatie van een leerling, hoeft er niet zozeer gevoelige data worden opgeslagen. De informatie over een leerling kan centraal worden opgeslagen, zodat het medium enkel een ID hoeft te bevatten die gekoppeld is aan een leerling. Aan de te gebruiken techniek wordt dus ook niet de eis gesteld dat er data versleuteld kan worden opgeslagen. Het draagbare medium moet echter niet gemakkelijk na te maken zijn, om zo fraude te voorkomen.

De volgende eisen zijn opgesteld aan de te gebruiken techniek waarmee een leerling zich moet kunnen aanmelden bij binnenkomst in een klaslokaal. Een aantal eisen is daarbij voortgekomen uit de eerdere gesprekken met de opdrachtgever en andere eisen zijn voortgekomen uit de opgedane kennis van de onderzochte technieken.

- De uit te voeren handeling moet snel en gemakkelijk zijn.
- Indien een leerling een tag/kaart bij zich moet dragen, moet deze kaart/tag duurzaam zijn en niet snel kapot gaan.
- De benodigde apparatuur moet het liefst zo goedkoop mogelijk zijn, zodat het voor een school niet te duur wordt om bij elk lokaal te implementeren. Goedkoop is een relatief begrip, daarom worden de technieken ten opzichte van elkaar afgewogen om te bepalen of ze goedkoop of juist duur zijn.
- Indien gebruik wordt gemaakt van een kaart/tag, moet deze kaart/tag niet gemakkelijk na te maken zijn, om fraude te voorkomen. Daarbij is wel rekening gehouden dat het product gemaakt wordt voor leerlingen. Ook fraudegevoelig is een relatief begrip en daarom worden de technieken ten opzichte van elkaar afgewogen om te bepalen of ze fraudegevoelig zijn.

Legenda vergelijking identificatie-technieken	
++	De techniek voldoet zeker aan deze eis
+	De techniek voldoet aan de eis
~	De techniek voldoet in zekere mate aan de eis
-	De techniek voldoet niet aan deze eis
- -	De techniek voldoet zeker niet aan deze eis

Eisen	Barcode	Magnetische strip	Smart Card	Biometrie	RFID	NFC
Handeling snel en gemakkelijk	~	~	~	-	++	+
Kaart/tag duurzaam	- -	~	+	N.V.T	++	++
Relatief goedkoop	++	+	~	-	+	+
Niet fraudegevoelig	- -	-	+	++	~	+

Tabel 10: Identificatie-technieken uiteengezet

Tabel 10 toont de eerdergenoemde eisen, uitgezet tegen de verschillende technieken. De barcode en de magnetische strip vallen af doordat deze enkel aan de eis voldoen dat het een goedkope oplossing is. De barcode is erg eenvoudig na te maken en is niet duurzaam. De magnetische strip biedt geen voordelen tegenover de andere technieken.

Bij de variant van de smart card met contactpunten is de uit te voeren handeling minder gebruiksvriendelijk dan met de contacloze variant. Laatstgenoemde komt erg overeen met een normale RFID-tag, maar er is een grotere opslag van data mogelijk, waarbij de data versleuteld kan worden opgeslagen en er authenticatie-mogelijkheden zijn. Zoals eerder is besproken is dat niet als eis gesteld aan de te gebruiken techniek. Verder biedt een smart card geen voordelen tegenover een normale RFID-tag en is een smart card iets duurder.

Biometrie is het minst fraudegevoelig, doordat bijvoorbeeld een vingerafdruk altijd uniek is en niet zomaar is na te maken. De te verrichte handeling is bij zulke systemen echter minder gemakkelijk dan de andere technieken. Ook is de benodigde apparatuur relatief duur. Het is een afweging tussen fraudegevoeligheid, en gemak en kosten. Het te maken systeem is bedoeld om laag in de markt te zetten en de te verrichte handeling moet zo eenvoudig mogelijk zijn. Biometrie is dan niet de beste oplossing.

RFID en NFC zijn beide goede kandidaten als identificatie-techniek voor het te maken systeem. De tags zijn erg duurzaam en de tags kunnen worden gelezen door een portemonnee heen, waardoor de handeling snel verricht kan worden. Ook zijn dit intussen goedkope technieken en kan er niet op een makkelijke wijze fraude mee gepleegd worden.

NFC biedt echter additionele functionaliteiten bovenop RFID, zoals betere encryptie van opgeslagen data, daadwerkelijke communicatie en data-overdracht tussen twee chips en het is geïntegreerd in de nieuwere smartphones. Laatstgenoemde functie zou interessant kunnen zijn. Een leerling zou zich dan kunnen aanmelden met zijn smartphone in plaats van met een kaart/tag. Een nadeel is dat niet alle leerlingen een smartphone met NFC-functionaliteit bezitten. Ook moet er dan een app ontwikkeld worden om het geheel werkend te krijgen en moet bij aanmelden eerst de app geopend worden, wat langer duurt dan enkel een kaart tegen de lezer houden. Indien een leerling een andere smartphone krijgt moet er een nieuwe koppeling plaatsvinden. Tot slot is met NFC een maximaal bereik van 10 cm aanwezig, terwijl met RFID langere afstanden mogelijk zijn. Een toekomstidee zou dan kunnen zijn dat een leerling niet meer handmatig een kaart hoeft te scannen, maar dat de tag/kaart automatisch wordt gelezen bij binnenlopen van een bepaald klaslokaal. Er is gekozen voor RFID als techniek voor het identificatiesysteem.

6.2 Werking van RFID

In de vorige paragraaf is de keuze beschreven voor het gebruik van RFID. Vervolgens is onderzocht hoe RFID werkt en welke vormen van RFID er bestaan, om zo tot de keuze te komen voor het gebruik van een specifieke RFID-reader met bijbehorende tags. De uitgebreide beschrijving hiervan is te vinden in het analyse document. In deze paragraaf blijft het beperkt tot een beknopte beschrijving, met enkel de benodigde kennis voor het volgen van de gemaakte keuze. Voor deze gedetailleerdere analyse naar RFID zijn ook nieuwe bronnen gebruikt naast degene die zijn gebruikt voor het verkrijgen van algemene informatie over RFID. [20] [24] [25] [21]

6.2.1 Architectuur RFID

Voor de identificatie aan de hand van een RFID-tag zijn drie componenten benodigd: De RFID-tag, een RFID-reader en een host.

- De tag bestaat uit een antenne en een chip. Elke tag heeft een eigen ID, wat de tag uniek maakt. De tag kan worden geïntegreerd in een object zoals een kaart of een sleutelhanger.
- De reader leest de informatie van een tag en stuurt deze informatie door naar de backend.
- De host kan een computer of bijvoorbeeld een microcontroller zijn die de reader aanstuurt. De verkregen informatie kan worden verwerkt of kan worden doorgestuurd naar een ander systeem.

6.2.2 Frequentiebereik

RFID werkt op drie verschillende frequenties, waarbij elke frequentie andere eigenschappen heeft. De gebruikte frequenties worden de Lage Frequentie (LF), de Hoge Frequentie (HF) en de Ultra Hoge Frequentie (UHF) genoemd.

Lage Frequentie

De frequenties die gebruikt worden bij LF RFID zijn 125 KHz en 134.2 KHz. LF wordt voornamelijk gebruikt bij communicatie met een RFID-tag op een paar centimeters afstand, afhankelijk van de gebruikte antenne. Er zijn krachtigere readers beschikbaar waarmee een afstand van een meter kan worden gehaald, deze zijn echter wel een stuk duurder. RFID-standaarden binnen deze frequentie zijn onder andere ISO 14223 en ISO/IEC 18000-2.

Hoge Frequentie

De gebruikte frequentie is 13.56 MHz. NFC werkt ook op deze frequentie en is compatible met een aantal RFID-standaarden. De tags zijn goedkoper te produceren dan bij LF, er kunnen meerdere tags tegelijkertijd gelezen worden en het bereik kan oplopen tot 50 cm, al ondersteunen de meeste readers een afstand tot 10 cm. RFID-standaarden binnen deze frequentie zijn onder andere ISO/IEC 18092, voor NFC, en de veelgebruikte ISO/IEC 14443 A en ISO/IEC 14443 voor MIFARE producten.

Ultra Hoge Frequentie

De frequenties die gebruikt worden bij UHF bevinden zich tussen de 300MHz en 3Ghz. Het voordeel van deze hoge frequentie is dat er een bereik van enkele meters haalbaar is met een passieve tag en zelfs honderden meters in combinatie met een actieve tag. UHF-systemen zijn doorgaans een stuk duurder dan de systemen op de andere twee frequenties. Er is één standaard voor UHF-RFID, namelijk de ISO 18000-6C standaard (UHF Gen 2).

6.2.3 Passieve vs actieve tags

Er kan een onderscheid worden gemaakt in passieve en actieve tags.

Passieve tags

Passieve tags bezitten zelf geen batterij om te kunnen opereren, maar gebruiken de energie uit het signaal dat is verstuurd door de reader. Hierdoor kan de tag voor lange tijd gebruikt worden. Echter beperkt dit wel de afstand tot de reader en is er beperkte mogelijkheid om data op te slaan. Deze tags kunnen al zo goedkoop zijn als tien eurocent per tag.

Actieve tags

Een actieve tag bezit wel een batterij om te kunnen opereren. Daardoor is er meer rekenkracht aanwezig en kan er meer data worden opgeslagen. De afstand tot een reader wordt vergroot ten opzichte van een passieve tag en kan oplopen tot honderden meters. Een actieve tag opereert over het algemeen dan ook in de UHF. De prijs van een actieve tag kan oplopen tot honderd euro.

Semi actieve tags

Dit type tag gebruikt ook een batterij, maar enkel voor doeleinden om bijvoorbeeld sensoren uit te lezen. Het terugsturen van een signaal naar de reader gebeurt nog wel op dezelfde manier als bij een passieve tag. Dit type tags zijn een stuk goedkoper dan actieve tags, maar hebben een minder groot bereik.

6.2.4 Keuze voor specifieke RFID-reader en bijbehorende tags

De mooiste oplossing voor het identificeren van een leerling zou zijn dat de tag die de leerling bij zich draagt automatisch wordt uitgelezen op het moment dat de leerling het klaslokaal binnen komt, net zoals de beveiligingspoortjes werken in een winkel. De theoretische afstand van een HF-reader kan oplopen tot 50 cm, echter in de praktijk valt dit meestal een stuk lager uit. Een UHF-reader zou geschikt kunnen zijn doordat met een passieve tag enkele meters kunnen worden overbrugt. Deze readers zijn een stuk duurder in vergelijking met HF RFID-readers (al snel 10 a 20 keer zo duur) en er is in overleg met de opdrachtgever dan ook voor gekozen om geen UHF-reader te gebruiken. Voor het prototype van het systeem volstaat het om een reader te gebruiken waarbij manueel een tag moet worden gehouden.

De tags van een HF-reader zijn over het algemeen goedkoper dan tags van een LF-reader. Een HF-reader is in zekere mate ook te gebruiken in combinatie met een NFC-tag, mits er dezelfde standaard wordt gebruikt. HF-tags kunnen ook worden uitgelezen door een smartphone met een NFC-chip. Dit kan het bijvoorbeeld mogelijk maken om tags te koppelen aan een leerling door middel van een smartphone. Door deze redenen is er gekozen voor het gebruik van een HF RFID-reader met bijbehorende tags. De tags kunnen passieve tags zijn, doordat er geen grote hoeveelheid data hoeft te worden opgeslagen, er geen grote afstanden worden overbrugd en deze tags goedkoop zijn.

MIFARE RFID tags zijn de meest gebruikte tags in combinatie met een HF RFID-reader. De MIFARE-technologie is gebaseerd op de ISO 14443 standaard. Er zijn meerdere categorieën tags, variërend van de MIFARE UltraLight, een tag met 64 bytes zonder beveiliging, tot de MIFARE Plus X, met 4KB aan opslag en verschillende encryptiemethodes. [26] Laatstgenoemde valt in de categorie van de smart cards. Een RFID-reader die overweg kan met de MIFARE-tags heeft dan ook de voorkeur.

Elke tag heeft van zichzelf een uniek opgeslagen ID, die niet is te veranderen. In het geval van het te maken systeem kan zo'n ID van een tag gekoppeld worden aan een leerling, zodat bij uitlezen van de tag bekend is welke leerling aanwezig is. Zoals eerder is beschreven zou zo'n tag kunnen worden nagemaakt met de juiste apparatuur en kennis, zodat er fraude gepleegd kan worden. Eenvoudig is dit echter niet en het betreft hier een systeem waar leerlingen zich kunnen aanmelden/afmelden en

geen systeem voor toegangscontrole of betalingen. De beveiliging speelt voor dit prototype dan ook niet een hele grote rol, waardoor er gewoon gebruik kan worden gemaakt van het ID van de tag. Indien de tag in de toekomst ook gebruikt gaat worden voor bijvoorbeeld het openen van kluisjes, zal er wel extra beveiliging benodigd zijn en dan zou er alsnog een contactloze smart card gebruikt kunnen worden.

In eerste instantie was de keus gemaakt voor de HF RFID reader/writer RC522, een zeer goedkope RFID-module die kan worden aangesproken via SPI en overweg kan met de meeste MIFARE-tags. Alleen was er bij Alflex al een HF RFID-reader aanwezig, namelijk de SL018, gebaseerd op een goedkope Chinese chip. De SL018 kan onder andere ook overweg met alle MIFARE-tags en is aan te spreken via I2C. De keus is uiteindelijk dan ook gevallen op de SL018.

6.3 Prototype-code RFID-reader

Na de keuze voor de SL018 is er prototype-code geschreven om te kunnen communiceren met deze reader. De reader is te benaderen via I2C. In de Nordic SDK is een driver opgenomen voor I2C, die met een paar wijzigingen is te gebruiken, wat het werken met I2C vergemakkelijkt. Door middel van een hoog signaal op een gekoppelde input pin laat de reader weten dat er een tag in de buurt van de reader wordt gehouden. Deze pin kan dus gepolld worden en vervolgens kan de tag worden uitgelezen door de juiste commando's naar de reader te schrijven. De reader bleek te werken, dus deze reader kon gebruikt worden voor het verdere project.

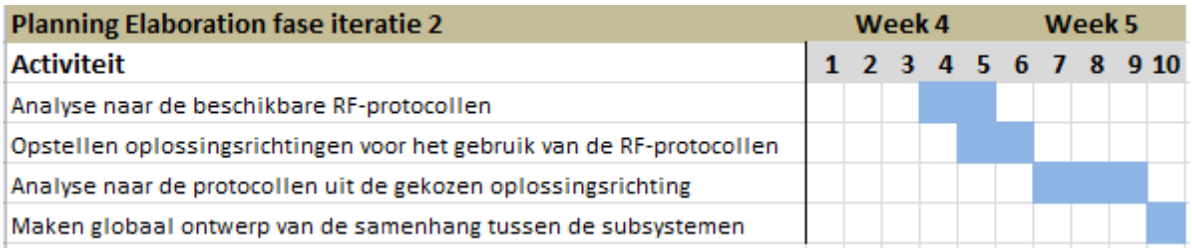
6.4 Evaluatie elaboration fase iteratie 1

Alle doelen die gesteld waren aan deze iteratie zijn behaald. Er is een analyse uitgevoerd naar bestaande identificatiesystemen en de daarin gebruikte technieken. Hiermee is weer één van de risico's uit de risicolijst aangepakt, namelijk risico nummer 1. Er is een keus gemaakt voor het gebruik van RFID als identificatietechniek en er is een specifieke reader gekozen. Tot slot is er prototype code geschreven om zo de werking van de reader te bevestigen.

Er stonden 2,5 week gepland voor deze iteratie, maar ook deze iteratie is een aantal dagen eerder afgerond dan verwacht.

7 Elaboration fase 2: Analyse communicatiemogelijkheden

In dit hoofdstuk wordt het proces toegelicht dat is doorlopen gedurende de tweede iteratie van de elaboration fase. De planning voor deze iteratie is te zien in Figuur 7.



Figuur 7: Planning elaboration fase iteratie 2

Het te maken systeem bestaat uit meerdere meetstations die hun klimaatmetingen doorsturen naar een basisstation. Met een smartphone is, door middel van een Bluetooth-verbinding met het basisstation, deze informatie uit te lezen. In deze iteratie is onderzocht hoe de communicatie tot stand kan worden gebracht tussen de stations en de smartphone.

7.1 Oplossingsrichtingen

De ontwikkelkit die voor de ontwikkeling wordt gebruikt, de nRF51 DK, kan overweg met alle softdevices en is dus in staat om te werken met ANT, Bluetooth Smart en het Gazell-protocol. Een korte beschrijving van deze protocollen en de verschillende softdevices is eerder gegeven in hoofdstuk 5.3.2. De communicatie tussen een meetstation en het basisstation, en de communicatie tussen het basisstation en een smartphone kan op verschillende manieren worden bewerkstelligd. Er zijn verschillende oplossingsrichtingen bedacht en voor elke oplossingsrichting is bepaald wat de voor- en nadelen zijn om zo tot een geschikte keus te komen. De voor- en nadelen zijn vergaard aan de hand van de eerder beschreven informatie, ervaringen van mensen op verschillende fora en door het experimenteren met de softdevices en de voorbeelden die zijn geleverd door Nordic. Wat volgt is een korte beschrijving van de oplossingsrichtingen met de keus voor één van deze oplossingen.

7.1.1 Gebruik van softdevice S130

Het S130-softdevice stelt het apparaat in staat om tegelijkertijd te fungeren als zowel een central en als een peripheral bij communicatie over Bluetooth Smart. Het basisstation kan dan de rol van peripheral aannemen bij de communicatie met de smartphone en de rol van een central bij de communicatie met de meetstations.

Voordelen	Nadelen
Gebruik van één enkel protocol voor alle communicatie is een consistente oplossing.	Het softdevice is nog in alpha-versie en is nog niet uitgebracht voor productie-doeleinden.
Er is kennis enkel nodig van Bluetooth Smart.	Er is weinig documentatie beschikbaar voor dit softdevice.
	Geen linkerscript en makefile aanwezig wat benodigd is voor ontwikkeling met GCC.
	Op fora veel problemen te vinden om het geheel werkend te krijgen onder GCC.

Tabel 11: Voor- en nadelen gebruik van S130

7.1.2 Gebruik van softdevice S110 en het Gazell-protocol

Het S110-softdevice stelt het apparaat in staat om te fungeren als een peripheral bij communicatie over Bluetooth Smart. Het basisstation kan dan de rol van peripheral aannemen bij de communicatie

met de smartphone. Het Gazell-protocol kan vervolgens gebruikt worden om te communiceren met de meetstations.

Voordelen	Nadelen
Het S110-softdevice is goed gedocumenteerd.	Het Gazell-protocol en Bluetooth kunnen niet tegelijkertijd worden gebruikt. Dynamisch wisselen kan wel, maar een bestaande Bluetooth-verbinding met de smartphone wordt dan verbroken bij het wisselen naar Gazell. Indien Gazell uitgezet wordt als er een smartphone is verbonden, is het niet mogelijk om bijvoorbeeld de nieuwste klimaatwaardes op te vragen en te tonen op de smartphone.
Er zijn meerdere voorbeeldapplicaties te vinden die gebruik maken van de S110-softdevice.	Er is vrijwel geen documentatie te vinden over het Gazell-protocol.
	Het Gazell-protocol is veel complexer om te implementeren dan ANT en Bluetooth Smart.

Tabel 12: Voor- en nadelen gebruik van S110 en Gazell

7.1.3 Gebruik van softdevice S310

Het S310-softdevice stelt het apparaat in staat om te communiceren door middel van het ANT-protocol en tegelijkertijd te fungeren als peripheral bij communicatie over Bluetooth Smart. Het basisstation kan dan de rol van peripheral aannemen bij de communicatie met de smartphone en over ANT communiceren met de meetstations.

Voordelen	Nadelen
ANT en Bluetooth kunnen correct tegelijkertijd naast elkaar werken. Collisions bij gebruik van dezelfde radio worden opgelost door het softdevice zelf.	Enkel de rol van peripheral is mogelijk binnen een Bluetooth-verbinding bij dit softdevice, de central-rol wordt niet ondersteund.
Het Bluetooth-protocol werkt hetzelfde als in het S110-softdevice en het gebruik is dan ook goed gedocumenteerd.	Het gebruik van het ANT-protocol is minder goed gedocumenteerd dan Bluetooth.
	Het gebruik van twee verschillende protocollen is niet consistent.

Tabel 13: Voor- en nadelen gebruik van S310

7.1.4 Gebruik van softdevice S120 en een aparte Bluetooth-module

Het S120-softdevice stelt het apparaat in staat om te fungeren als een central binnen een Bluetooth-verbinding. Het basisstation zou dan als central kunnen fungeren en zo communiceren met de meetstations over Bluetooth Smart, met gebruik van de interne Bluetooth-stack. Een aparte Bluetooth-module kan vervolgens gebruikt worden voor de communicatie met de smartphone.

Voordelen	Nadelen
Het S120-softdevice is goed gedocumenteerd.	Er is een aparte Bluetooth-module benodigd bovenop de al bestaande Bluetooth-stack.
Er zijn meerdere voorbeeldapplicaties te vinden die gebruik maken van de S120-softdevice.	Een extra Bluetooth-module kost meer energie.
Consistent in het gebruik van één protocol.	

Tabel 14: Gebruik van S120 en Bluetooth-module

7.1.5 Keuze voor oplossingsrichting

Het gebruik van het S130-softdevice leek in eerste instantie het meest voor de hand te liggen, aangezien dan alle communicatie kan verlopen via de interne Bluetooth-stack. Echter doordat dit softdevice nog in de alpha-versie is, is er weinig documentatie, werkt het nog niet goed samen met de GCC-toolchain en is het onbekend of het geheel stabiel draait.

Indien er gebruik wordt gemaakt van het S110-softdevice in combinatie met Gazell, is het niet mogelijk om de verbinding met een smartphone te behouden op het moment dat er gecommuniceerd moet worden met de meetstations. Daarnaast is het Gazell-protocol lastig te implementeren door afwezigheid van goede documentatie. De optie van een aparte Bluetooth-module is waarschijnlijk de meest gemakkelijke oplossing om het geheel werkend te krijgen, alleen gaat dit tegen de argumenten in voor het gebruik van een nRF51 SoC. De argumenten voor deze chip waren namelijk dat de chip zelf al mogelijkheden bevat voor RF, met een laag energieverbruik.

Het gebruik van het S310-softdevice heeft ook zo zijn nadelen, zo is het gebruik van twee verschillende protocollen niet erg consistent en is het gebruik van het ANT-protocol minder goed gedocumenteerd. De lijkt echter wel de enige oplossing te zijn waarbij er tegelijkertijd met de smartphone en met de meetstations kan worden gecommuniceerd, zonder dat er een extra module benodigd is. Er is dan ook gekozen voor deze oplossingsrichting.

7.2 Analyse naar Bluetooth Smart

Bluetooth Smart wordt gebruikt voor de communicatie tussen het basisstation en de smartphone. Hiervoor wordt gebruik gemaakt van de Nordic SDK, die een API biedt naar het softdevice. De API is bestudeerd en het bleek dat er nog vrij veel kennis benodigd is van het Bluetooth Smart protocol om een verbinding op te zetten. Hiervoor moeten de hoger gelegen lagen binnen het protocol worden ingesteld. Daarom is er een analyse uitgevoerd naar de werking van deze lagen binnen het protocol. De lager gelegen lagen worden door het softdevice afgehandeld en deze zijn dus ook niet bestudeerd. De aanpak van de analyse is hetzelfde als de eerdere analyses. Er zijn bronnen gezocht [27] [28] [12] en de gevonden informatie is geformuleerd in het analyse document. In deze paragraaf blijft het beperkt tot een beknopte beschrijving.

7.2.1 Generic Access Profile (GAP)

De GAP-laag binnen de Bluetooth-stack is verantwoordelijk voor het beheren van de connecties tussen de apparaten. Hiermee wordt aangegeven hoe een apparaat zich bekend maakt aan de buitenwereld en op welke manier twee apparaten met elkaar verbonden moeten worden. Een apparaat neemt daarbij een bepaalde rol aan binnen GAP: broadcaster, observer, peripheral of central. Deze rollen zijn eerder besproken in hoofdstuk 5.3.2.

Een peripheral kan met een bepaald interval een advertentie-pakket versturen naar alle apparaten in de omgeving. Een central die aan het scannen is ontvangt zo'n pakket en kan vervolgens een connectie initiëren met de peripheral. Een peripheral kan enkel een verbinding hebben met één central, maar een central kan meerdere verbindingen hebben met verschillende peripherals.

7.2.2 Generic Attribute Profile (GATT)

De GATT-laag is verantwoordelijk voor het daadwerkelijk versturen van de data over de verbinding die is opgezet door de GAP-laag.

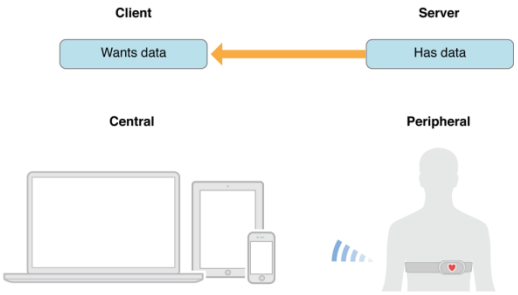
Rollen binnen GATT

Binnen de GATT-laag kunnen er twee rollen worden aangenomen:

- Client: In deze rol bezit het apparaat zelf geen data, maar communiceert het met een GATT-server om de daar opgeslagen data te lezen en/of te schrijven.

- Server: In deze rol bezit het apparaat opgeslagen data die toegankelijk wordt gemaakt voor een GATT-client.

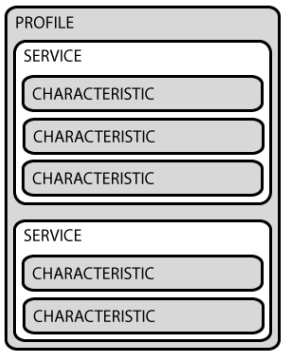
Het is het meest gebruikelijk dat een GAP-central fungeert als een GATT-client en dat een GAP-peripheral fungeert als een GATT-server. Dit hoeft echter niet altijd zo te zijn, de rollen op de twee lagen zijn gescheiden van elkaar. Dit wordt verduidelijkt in Figuur 8.



Figuur 8: Rollen binnen GAP/GATT [28]

Hiërarchie binnen GATT

De GATT-server structureert de data in een zogenoemde attribuut-tabel, waarin de daadwerkelijke data is opgeslagen in de attributen. De GATT-server implementeert een of meer Bluetooth profielen, waarbij elk profiel bestaat uit een of meer services, die weer bestaan uit een of meer characteristics. Een characteristic bestaat uiteindelijk weer uit de attributen. Deze structuur is verduidelijkt in Figuur 9.



Figuur 9: Structuur van Bluetooth Smart profiel [27]

7.3 Analyse naar ANT

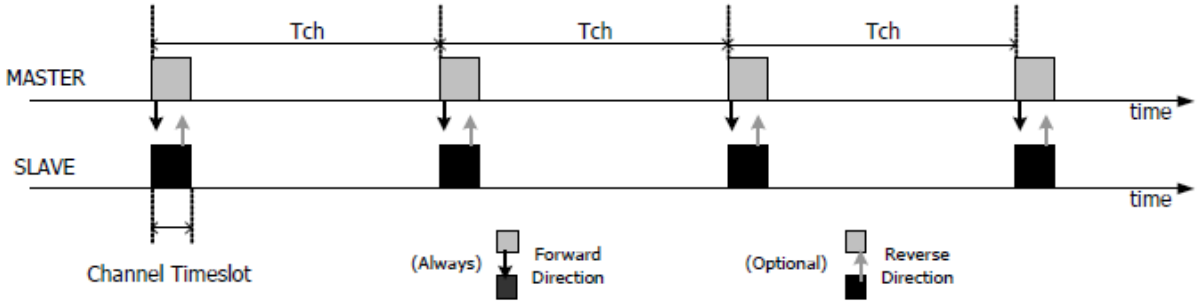
Om dezelfde reden als Bluetooth Smart is er kennis opgedaan van het ANT-protocol. Deze kennis is benodigd op het moment dat dit protocol geïmplementeerd gaat worden als communicatie tussen de meetstations en het basisstation. De volledige analyse is te lezen in het analyse document, in deze paragraaf blijft het beperkt tot de kennis die benodigd is voor een beter begrip van het verdere verslag. Voor de analyse zijn verschillende bronnen gebruikt. [29] [30] [31]

7.3.1 Communicatie over ANT

Een node (apparaat) binnen ANT kan fungeren als een master of als een slave. Een master initieert de verbinding, beheert het kanaal en verstuurt data naar de slave. De slave is voornamelijk de ontvanger van data en is verantwoordelijk voor de synchronisatie met de master. Een node kan tegelijkertijd zowel een master als een slave zijn, zodat de meest complexe netwerktopologieën mogelijk zijn (in tegenstelling tot Bluetooth Smart, waarmee enkel een ster-netwerk gevormd kan worden).

Twee nodes communiceren met elkaar over een zogenoemd kanaal. Er zijn verschillende communicatiemethoden, afhankelijk van het kanaaltype, hoe het kanaal is geconfigureerd, wat voor datatype er gebruikt wordt en welke kant de data op wordt verstuurd. Bij configuratie van een kanaal moeten er een vijftal parameters worden ingesteld.

Een master verzendt altijd een datapakket op het afgesproken tijdslot, waarop de slave optioneel kan reageren. Als de data ondertussen niet is vernieuwd wordt er toch een datapakket verstuurd, met daarin de oude data, voor de synchronisatie van het kanaal. Dit wordt verduidelijkt in Figuur 10.



Figuur 10: Communicatie over een ANT-kanaal [29]

7.3.2 Beheer van een kanaal

Binnen ANT zijn er drie verschillende manieren om een kanaal te gebruiken: onafhankelijke kanalen, gedeelde kanalen en een scan mode.

- *Onafhankelijk kanaal:* bij deze manier heeft het kanaal doorgaans één master en één slave. Een broadcast netwerk kan worden geïmplementeerd met één master en meerdere slaves op een onafhankelijk kanaal, echter kunnen de slaves dan niks terugsturen.
- *Gedeeld kanaal:* bij deze manier communiceert een node met meerdere andere nodes, waarbij de centrale node wordt geconfigureerd als master.
- *Scan mode:* bij deze manier ontvangt een slave node data van meerdere masters. De slave node verbruikt dan echter wel meer energie.

7.3.3 ANT+

ANT+ kan optioneel aan het ANT-protocol worden toegevoegd. ANT+ legt bepaalde kanaalparameters vast, evenals datastructuren en netwerksleutels. Door het vastleggen van deze standaarden worden losstaande producten in staat gesteld met elkaar te kunnen communiceren. Zo kunnen alle hartslagmeters voldoen aan dezelfde standaarden, zodat bijvoorbeeld een smart watch standaard kan werken met verschillende soorten hartslagmeters.

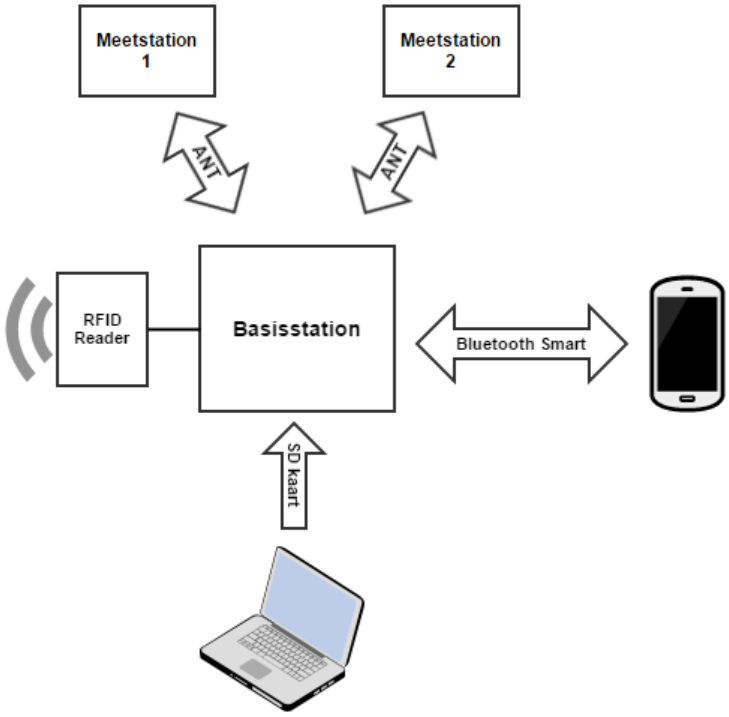
Het te maken systeem hoeft niet compatible te zijn met andere bestaande ANT-apparaten, zodat ANT+ dus ook niet toegepast zal worden.

7.4 Samenhang tussen de subsystemen

Aan het eind van de elaboration fase was duidelijk welke communicatietechnieken er gebruikt gaan worden tussen de subsystemen en welke identificatiemethode er gebruikt wordt voor het identificeren van de leerlingen. Figuur 4 uit de inception fase is daarom aangevuld met deze informatie, zodat de samenhang tussen de verschillende subsystemen is weergegeven in Figuur 11.

Het basisstation kan bij de ingang van een klaslokaal worden gehangen, zodat een persoon zich daar kan aanmelden door middel van de RFID-reader. De meetstations kunnen op een willekeurige plaats in het klaslokaal worden gehangen en sturen de gemeten waardes door naar het basisstation via het ANT-protocol. Met een smartphone kan via Bluetooth Smart een verbinding met het basisstation worden gemaakt om de gegevens uit te lezen.

In eerste instantie was er het plan om een desktopapplicatie te ontwikkelen, om daarmee via een seriële verbinding de demonstratiedata, van de lesblokken en studenten, in te laden op het basisstation. Later is er voor gekozen voor het overzetten van de data via een SD-kaart, in plaats van via een seriële verbinding. De keuze hiervoor wordt toegelicht in het volgende hoofdstuk.



Figuur 11: Samenhang tussen de subsystemen

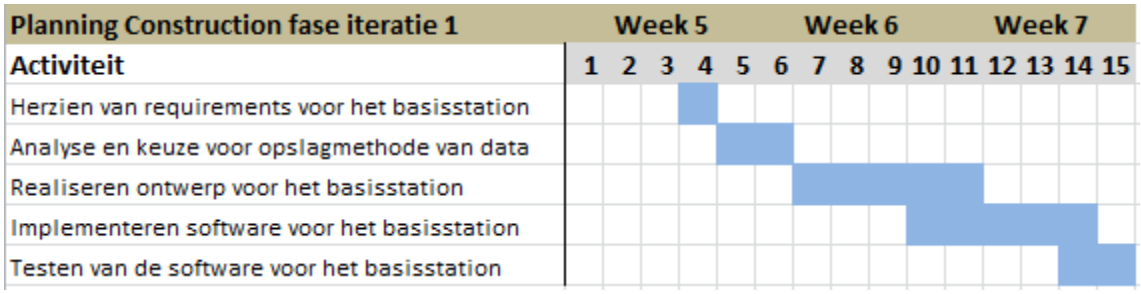
7.5 Evaluatie elaboration fase iteratie 2

Ook voor deze iteratie zijn de gestelde doelen behaald. Er is een keuze gemaakt uit de verschillende mogelijkheden tot communicatie, er is onderzocht hoe de communicatieprotocollen werken en de samenhang tussen de subsystemen is uitgedacht.

Op dit moment is er een einde gekomen aan de elaboration fase. De belangrijkste analyses zijn verricht en er zijn belangrijke keuzes gemaakt voor het gebruik van technieken in het verdere project. De grootste risico's uit de risicolijst zijn aangepakt en er is genoeg vertrouwen om de construction fase te starten.

8 Construction fase 1: Basisstation

De construction fase is opgedeeld in meerdere iteraties. In elke iteratie is er een subsysteem ontwikkeld, waarbij ook de benodigde veranderingen zijn doorgevoerd in de subsystemen uit eerdere iteraties. In dit hoofdstuk wordt het proces gedurende de eerste iteratie beschreven, waarin het grootste gedeelte van de software voor het basisstation is ontwikkeld. De planning voor deze iteratie is te zien in Figuur 12. In de globale planning was er enkel één week ingepland voor deze iteratie. Door nieuwe inzichten is besloten dat één week absoluut onhaalbaar zou zijn, zodat er meer tijd is ingepland.



Figuur 12: Planning construction fase iteratie 1

8.1 Verandering systeemeisen

Tijdens de afstudeerstage werd een prototype ontwikkeld van een systeem, dat mogelijk in de toekomst verder wordt ontwikkeld. Het was dus belangrijk om ook bepaalde ideeën over het toekomstige systeem helder te hebben, zodat hier rekening mee kon worden gehouden bij de ontwikkeling van het prototype.

Een van de eisen aan het (toekomstige) systeem was dat het systeem standalone moest kunnen werken, maar ook met een optionele webserver. In het eerste geval zou alle informatie, over welke groepen met leerlingen wanneer in het lokaal zouden moeten zijn, eenmalig manueel worden opgeslagen op het basisstation. In het laatste geval zou deze informatie centraal worden beheerd door een webserver. De bedoeling was om het prototype te ontwikkelen als een volledig werkende standalone versie.

Het bleek echter dat een standalone versie niet zou gaan werken. Op een middelbare school verandert het rooster van de leerlingen continue en vinden er regelmatig lokaalwijzigingen plaats. Zonder een centrale webserver zouden dan alle basisstations in een klaslokaal manueel geüpdatet moeten worden, wat niet te doen is. Het werd dan ook duidelijk dat een webserver niet meer optioneel is voor het toekomstige systeem, maar noodzakelijk is voor een juiste werking. Bij het toekomstige systeem zou dan geen data lokaal op een basisstation worden opgeslagen, maar zou het basisstation enkel dienen als doorgeefluik. De webserver beheert vervolgens alle data en heeft een koppeling met bestaande administratieve systemen binnen een school. De smartphone applicatie kan dan de data direct van de webserver halen. Wel zou er nog een verbinding met een basisstation gemaakt kunnen worden om bijvoorbeeld het huidige klimaat op te vragen of om bepaalde instellingen te veranderen op het station.

Het prototype moest wel nog steeds standalone kunnen werken. In samenspraak met de opdrachtgever werd besloten dat het prototype zou werken met statisch ingeladen demonstratiedata, waarbij eenmalig wordt opgegeven welke groepen met leerlingen er zijn en welke lesblokken er worden gevolgd. Ook wordt opgeslagen welk tag-ID er gekoppeld is aan welke leerling. Deze data is er enkel voor demonstratiedoeleinden, om zo wel tot een compleet werkend prototype

te komen, dat gebruikt kan worden als demonstratiemodel. Bij het prototype zal de smartphone applicatie nog alle data van het basisstation halen.

8.2 Opslag van de data

Standaard heeft de nRF51 DK geen mogelijkheden tot het persistent opslaan van data. Er is gekeken naar het gebruik van FLASH-geheugen, EEPROM en een SD-kaart. Alle drie de geheugentypes zijn benaderbaar via een protocol als I2C of SPI. Wijzigen van data in FLASH-geheugen of op een SD-kaart gaan vaak gepaard met het verwijderen van een groot stuk geheugenblok, terwijl bij EEPROM individuele bytes kunnen worden gewijzigd. Het nadeel van dit typen geheugen is dat een geheugengebied maar een beperkt aantal keer herschreven kan worden. Een SD-kaart lost dat probleem op door gebruik van *wear leveling*, waarbij data steeds op een andere plaats in het geheugen wordt geschreven. Bij FLASH en EEPROM moet dat op applicatieniveau worden afgehandeld. Een ander voordeel van een SD-kaart is dat er een bestandssysteem gebruikt kan worden, zodat de kaart op een computer kan worden uitgelezen.

Er is gekozen voor het gebruik van een SD-kaart voor de opslag van de data. Er is een open source FAT library gevonden die gebruikt kan worden, zodat er gewerkt kan worden met FAT als bestandssysteem. De demonstratiedata van groepen met leerlingen en de gegeven lesblokken kan dan gemakkelijk vanaf een computer op de SD-kaart worden gezet, zodat de data niet via een seriële verbinding naar het basisstation hoeft te worden overgezet.

Op de nRF51 DK is het niet mogelijk om de huidige tijd op te vragen, wat wel benodigd is bij het loggen van de aanmeldtijden van de leerlingen en bij het loggen van de klimaatwaardes. Als oplossing hiervoor is gebruik gemaakt van een externe RTC-module, gevoed met een batterijtje. Via het I2C-protocol is vervolgens de huidige tijd in te stellen en op te vragen.

8.2.1 Fysieke opslag

Het basisstation moet in staat zijn tot het opslaan en uitlezen van de volgende data:

- Eenmalig statisch ingeladen demonstratiedata: gedefinieerde lesblokken die gedurende de dag op een opgegeven tijdstip gegeven worden, met een groep leerlingen die dat lesblok moeten volgen. Zo is bekend welke leerling op welk tijdstip aanwezig hoort te zijn. Iedere leerling is vervolgens gekoppeld aan een RFID tag-id.
- Loggegevens van de datum en tijd dat een leerling zich heeft aangemeld.
- Loggegevens van de klimaatwaardes binnen het klaslokaal.

De demonstratiedata van de lesblokken en groepen met leerlingen, moet door middel van een desktopapplicatie op de SD-kaart kunnen worden gezet. De data is het snelst en gemakkelijkst uit te lezen door de microcontroller als de ruwe bytes worden opgeslagen en uitgelezen. Toch is er voor gekozen om deze data in ASCII-tekens op te slaan in een komma gescheiden bestand. Het voordeel hiervan is dat de data door een persoon kan worden gelezen. Indien blijkt dat er uiteindelijk toch niet genoeg tijd meer is om de desktopapplicatie te ontwikkelen, kan deze data nog manueel worden aangepast.

De loggegevens van de klimaatwaardes en aangemelde leerlingen worden door middel van een smartphone uitgelezen en hoeven dus ook niet leesbaar te zijn voor een persoon. Deze data wordt dan ook als binaire data opgeslagen in een bestand op de SD-kaart.

8.3 Realiseren van componentbord

Bij Alflex waren er al sockets aanwezig voor een SD-kaart, waar dan ook gebruik van is gemaakt. Zo'n socket kan niet zomaar verbonden worden met de ontwikkelkit, want het is geen kant en klare module met pinnen. De socket is dan ook gesoldeerd op een gaatjesbord, met de benodigde pull-up weerstanden en aansluitpinnen. Ook de RTC is hierop gesoldeerd, zodat er maar één bordje hoeft te worden verbonden met de ontwikkelkit.

8.4 Keuze voor geen RTOS

Bij het ontwikkelen van software voor embedded systemen, kan gebruik worden gemaakt van een *Real-Time Operating System (RTOS).* Een RTOS verdeelt de functies van de software in verschillende taken met een bepaalde prioriteit. Deze taken worden vervolgens afgewisseld aan de hand van een algoritme, waarbij gegarandeerd wordt dat taken met een hogere prioriteit ook sneller aan bod komen. [32] [33] Bij hard real-time systemen is het meestal noodzakelijk om een RTOS te gebruiken, omdat er dan absoluut geen deadline gemist mag worden door een taak. Als er geen RTOS wordt gebruikt, is het gebruikelijk vanuit de main-loop te pollen voor bepaalde events, waarbij geen garantie is dat een bepaalde functionaliteit binnen een opgegeven tijdsbestek aan bod komt.

Door de nRF51 wordt één RTOS officieel ondersteund, namelijk de RTX RTOS van KEIL. Er is verder weinig officiële documentatie te vinden voor het gebruik van een andere RTOS. Op het forum van Nordic is wel wat informatie gezet over dit onderwerp door medewerkers van Nordic. Daarop is te vinden dat andere RTOS, zoals Free-RTOS, niet standaard werken. [34] Wel zijn er een aantal niet-officiële RTOS beschikbaar van derde partijen, maar het is onduidelijk in hoeverre deze correct werken.

In de softdevices is standaard een schedule handling library opgenomen, die gebruikt kan worden om communicatie events van Bluetooth en ANT vanuit de achterliggende interrupts over te dragen naar de applicatie. Tijd kritische taken met betrekking tot de communicatie-protocollen worden op die manier door het softdevice afgehandeld, waarbij de garantie wordt gegeven dat deze taken binnen een opgegeven tijdseenheid uitgevoerd worden. [12] Hiervoor is geen aparte RTOS benodigd.

De voor- en nadelen voor het gebruik van de RTX RTOS zijn op een rijtje gezet, om zo tot een keuze te komen tot het wel of niet gebruiken van de RTOS.

Voordelen:

- Er kunnen tijd kritische taken worden gedefinieerd, waarbij er garantie is dat deze taken een opgegeven deadline halen.
- Een RTOS kan een duidelijker structuur geven aan het programma.

Nadelen:

- Er moet eerst kennis worden vergaard over de werking van de RTOS voordat deze gebruikt kan worden.
- De code kan er ook complexer door worden, zodat bij een kleinere applicatie het zonder RTOS overzichtelijker kan zijn.
- Op het Nordic forum zijn er problemen te vinden bij het gebruik van de RTX RTOS in combinatie met een softdevice en in combinatie met de GCC-compiler.

Er is voor gekozen om geen gebruik te maken van de RTX RTOS. De te ontwikkelen applicatie blijft een relatief kleine applicatie. De enige echt tijd kritische functionaliteiten binnen de applicatie, zijn de taken met betrekking tot de communicatie-protocollen. Voor deze taken kan gebruik worden gemaakt van de al aanwezige scheduler en de tijd kritische procedures worden intern afgehandeld door het softdevice zelf, zodat de applicatie daar niet verantwoordelijk voor is.

8.5 Eerste ontwerp basisstation

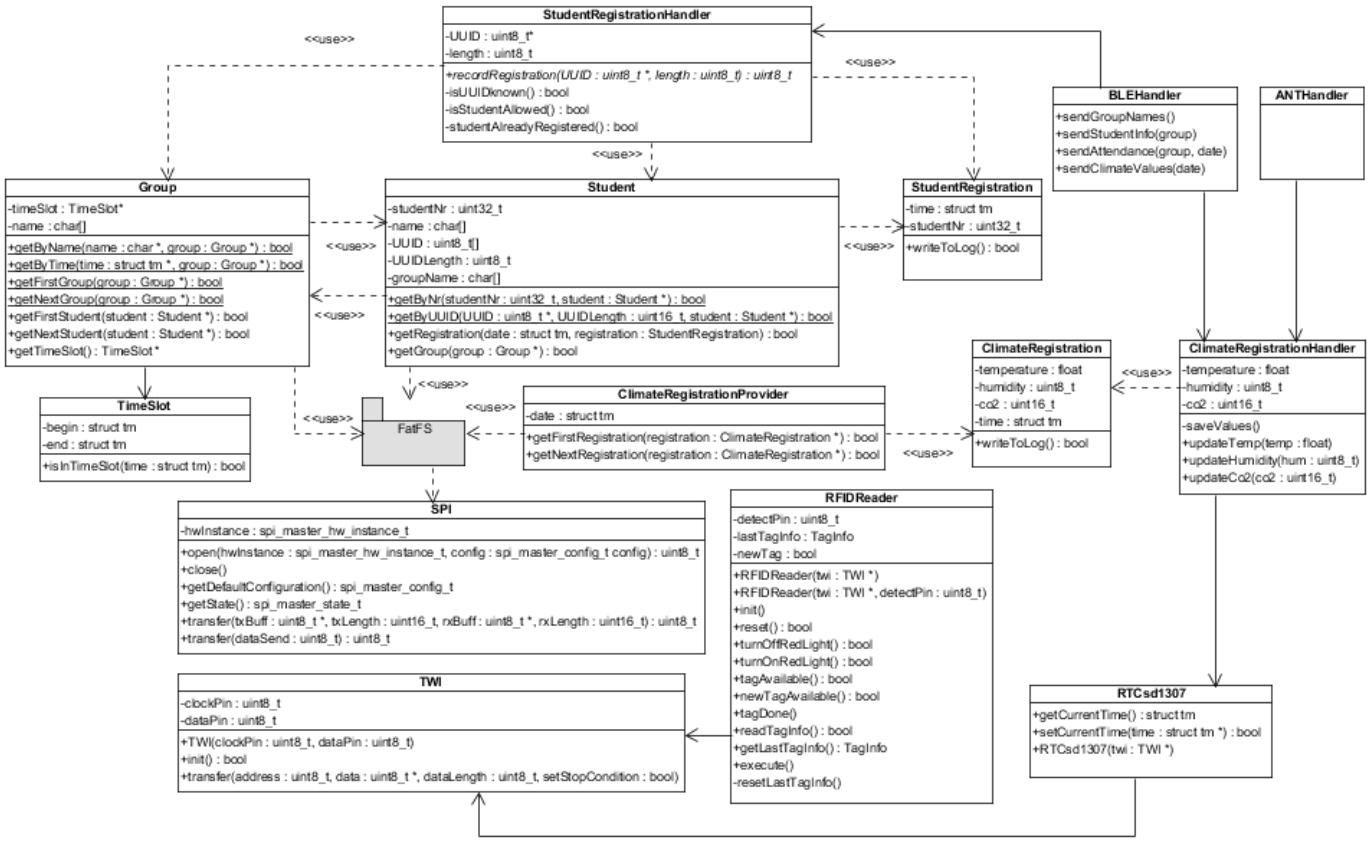
Zoals eerder is beschreven, is er voor gekozen de embedded software te ontwikkelen in C++. Voordat de code werd geschreven, is er eerst een ontwerp van de software gemaakt. Tijdens het implementeren van het ontwerp kwamen er fouten naar boven en bleken bepaalde delen van het ontwerp toch niet handig. Tijdens het implementeren van de code is het ontwerp dan ook regelmatig aangepast, om consistent te blijven met de code. Het ontwerp is vastgelegd door middel van UML-diagrammen [9] in een ontwerp document. Dit document beschrijft de architectuur en ontwerpen van de subsystemen, met keuzes voor bepaalde patronen en procedures. Ook worden belangrijke implementatiedetails besproken, die belangrijk zijn voor de verdere ontwikkeling van het systeem. Het ontwerp document is te vinden in bijlage E.

8.5.1 Best practices software embedded systemen

Embedded systemen beschikken doorgaans over relatief weinig geheugen en processorcapaciteit, waardoor hier zuinig mee om moet worden gegaan. Vanuit Alflex worden er dan ook een aantal best practices gehanteerd bij het schrijven van software voor embedded systemen. Het zijn geen harde eisen, maar over het algemeen zorgen ze voor stabielere software. Ook bij dit project is hiermee rekening gehouden. De best practices zijn van invloed op de gemaakte ontwerpen en constructies. Onderstaand volgt een opsomming van de best practices:

- Er dient geen gebruik te worden gemaakt van de C++ STL-libraries.
- Dynamisch geheugenallocatie dient zo veel mogelijk te worden beperkt en het liefst helemaal niet te worden gebruikt.
- Er dient geen gebruik te worden gemaakt van C++ exceptions.
- Er dient geen gebruik te worden gemaakt van real-time type checking.

8.5.2 Eerste opzet klassendiagram

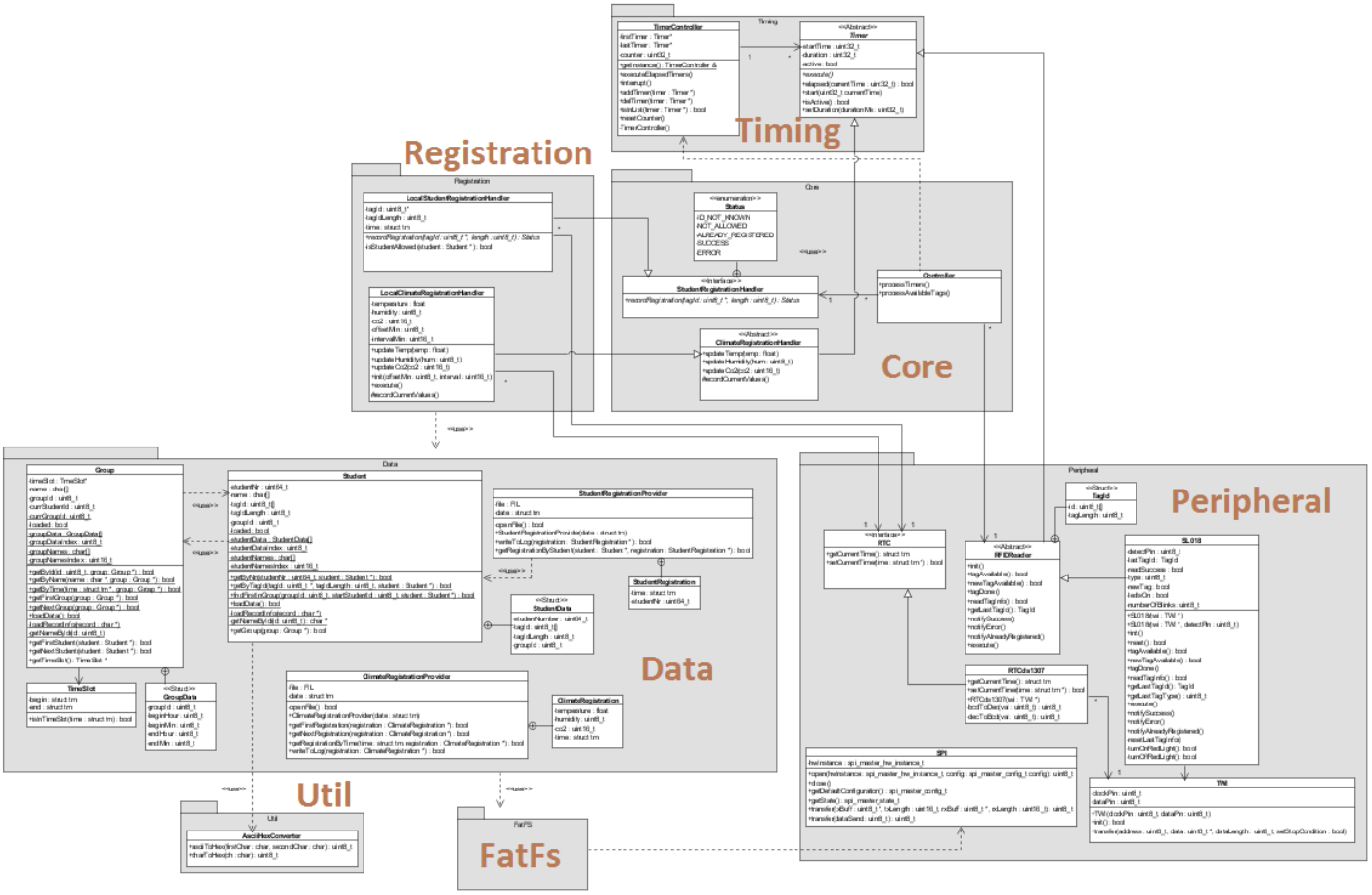


Figuur 13: Basisstation - eerste opzet klassendiagram

Om tot een eerste opzet van een klassendiagram te komen zijn er eerst conceptuele klassen verzameld, dus klassen die iets uit de werkelijkheid representeren. Voorbeelden hiervan zijn *Group, Student, RFIDReader* en *RTCsd1307*. Vervolgens zijn er operaties op deze klassen uitgedacht, zoals readTagInfo() voor de *RFIDReader.* Daarna zijn er klassen bedacht die niet iets uit de werkelijkheid representeren, maar benodigd zijn voor ondersteunende en uitvoerende taken binnen de software. Een voorbeeld is de klasse *StudentRegistrationHandler*, die verantwoordelijk is voor het verwerken van een gelezen RFID tag ID. Op deze manier is de eerste opzet van het klassendiagram opgebouwd, zoals deze is te zien in Figuur 13. Bij deze eerste opzet is nog geen gebruik gemaakt van interfaces of bepaalde design patterns, deze elementen zijn pas later toegepast voor meer structuur en onderhoudbaarheid.

8.5.3 Opdeling in packages

De eerste opzet van het klassendiagram is vervolgens uitgewerkt tot een groter klassendiagram, waarbij de klassen zijn ingedeeld in packages. Een UML klassendiagram voor het eerste ontwerp van het basisstation is te zien in Figuur 14. Dit diagram geeft een overzicht van de verschillende packages waaruit het ontwerp is opgebouwd, met de samenhang tussen deze packages. In het ontwerp document wordt elke package toegelicht. In dit verslag zal het beperkt blijven tot een aantal toelichtingen.



Figuur 14: Klassendiagram basisstation eerste ontwerp

De software is opgebouwd uit verschillende packages om zo meer structuur te geven aan de code en daarmee de uitbreidbaarheid en aanpasbaarheid te verhogen. Er is getracht ervoor te zorgen om de cohesie binnen een package hoog te houden en de koppeling naar andere packages zo laag mogelijk. Dit om ervoor te zorgen dat bij een bepaalde verandering het aantal te veranderen packages zo laag mogelijk is. De cohesie binnen de *Registration* package is echter zeer laag en de koppeling met

andere packages is hoog. Toch is er voor gekozen de klassen in deze aparte package te stoppen en niet in de *Core* package, zodat deze package in de toekomst gemakkelijk is te vervangen voor een andere package, waarbij de klassen op een andere manier zijn geïmplementeerd en dan communiceren met een webserver. De *Core* package hoeft dan niet aangepast te worden. Dit wordt verderop in dit document gedetailleerder beschreven.

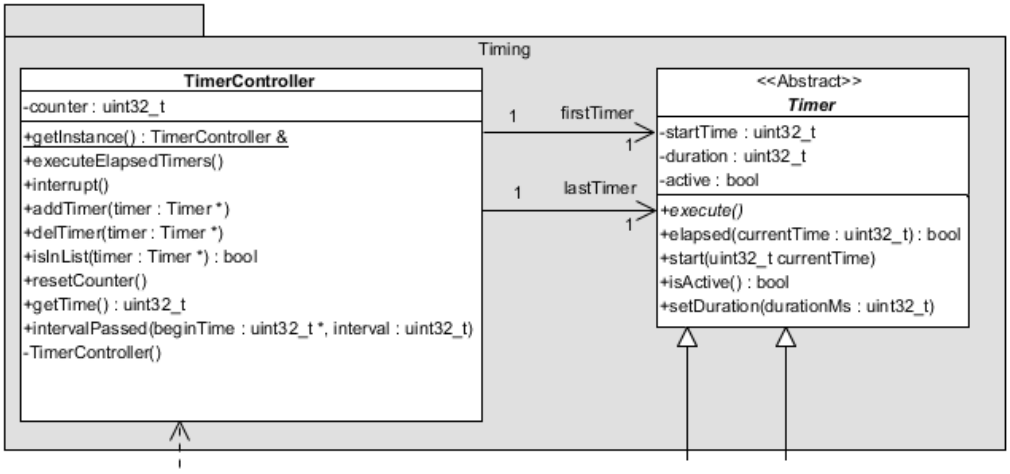
Tevens is er voor gezorgd dat bepaalde packages ook los zijn te gebruiken in andere projecten, zonder dat daar bijvoorbeeld de *Core* package voor benodigd is. Zo zouden de *Peripheral* package, met drivers voor de RFID-reader en de RTC, en de *Timing* package kunnen worden overgezet naar een ander project.

Bij de klassen waar in de toekomst verandering wordt verwacht, is er gebruik gemaakt van interfaces en abstracte klassen. Door het overerven van deze klassen kan er relatief eenvoudig een andere implementatie worden gebruikt. De andere klassen die afhangen van deze abstracte klassen en/of interfaces, hoeven op dat moment niet te veranderen. Hier is dus het principe "*klassen moeten afhankelijk zijn van abstracte klassen en niet van concrete klassen"* toegepast [35].

Timing package

De *Timing* package, te zien in Figuur 15, bestaat uit een *TimerController* en een abstracte klasse *Timer*. Het idee is dat subklassen van *Timer* zich kunnen aanmelden bij de *TimerController* en zo na een opgegeven tijdsinterval worden aangeroepen, waarbij de overridden methode *execute()* wordt aangeroepen. Het is daardoor geïmplementeerd als een soort observer pattern [36]. De lijst met timers is een dynamische lijst, geïmplementeerd als een linked list, waarbij een *Timer* zelf een node voorstelt van de lijst. Het voordeel hiervan is dat de *TimerController* geen groeiende lijst hoeft te implementeren, gebaseerd op dynamische geheugenallocatie. De *TimerController* is geïmplementeerd als een singleton, waardoor er maar één instantie van kan bestaan.

Met deze timer kunnen er bepaalde taken met een opgegeven tijdsinterval worden uitgevoerd. Dit zijn geen ontzettend tijd kritische taken, aangezien de daadwerkelijke uitvoering van een taak een aantal milliseconden kan schelen met het opgegeven tijdsinterval. Een voorbeeld van een taak zou kunnen zijn om een ledje te laten branden voor een tijdsbestek van één seconde.



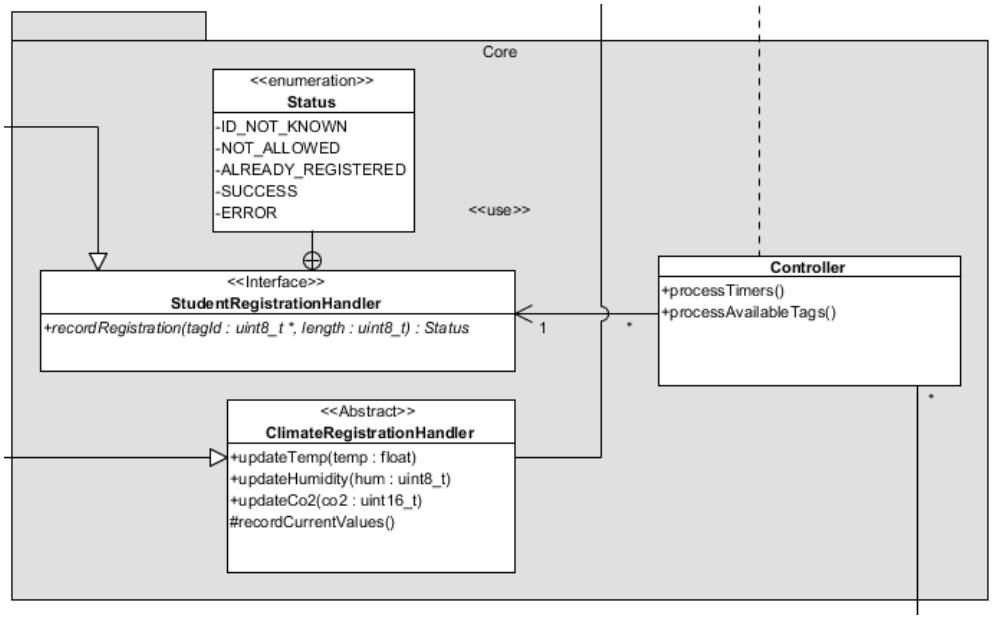
Figuur 15: Ontwerp basisstation - Timing package

Core package

De *Core* package bevat de *Controller* klasse, die zorgt voor het uitvoeren van de eerdergenoemde *TimerController* en voor het opvragen van informatie van de *RFIDReader* en het vastleggen van deze informatie door middel van een *StudentRegistrationHandler.* Met de *ClimateRegistrationHandler* kunnen de huidige klimaatwaardes worden vastgehouden en met een bepaald tijdsinterval worden

vastgelegd. De drie laatstgenoemde klassen zijn geïmplementeerd als abstracte klassen, omdat op deze plaatsen verandering wordt verwacht. Bij het prototype zullen de gegevens namelijk worden lokaal worden vastgelegd op een SD-kaart, maar in de toekomst zullen de gegevens waarschijnlijk worden verstuurd naar een server. Ook kan er een andere RFIDReader worden gebruikt, die op een langere afstand werkt.

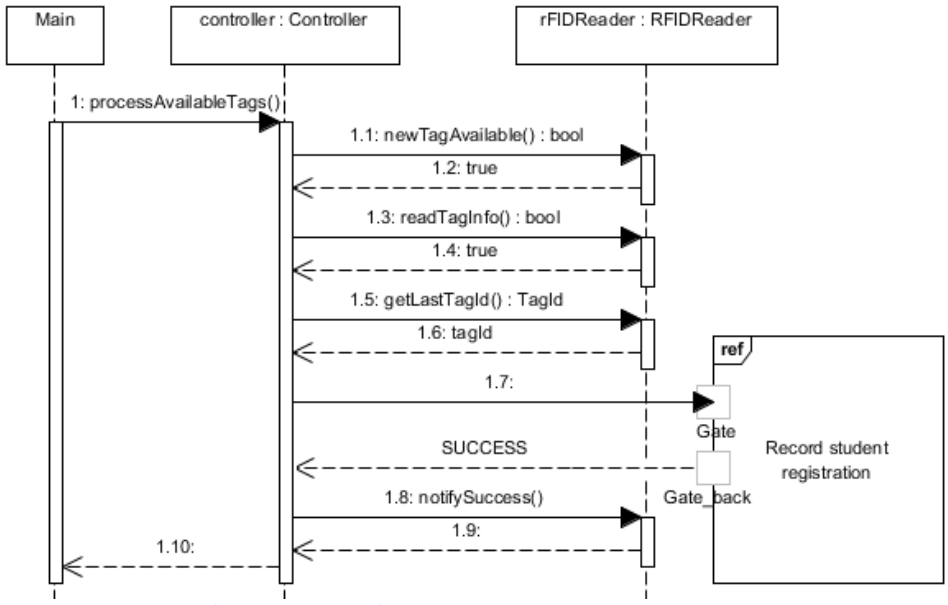
De *Registration* package implementeert de twee eerder genoemde registrationhandlers, waarbij de data lokaal wordt opgeslagen en ingelezen door middel van de *Data* package. Indien in de toekomst de data beheerd wordt door een webserver, kan de *Registration* package vervangen worden voor een andere package, met een andere implementatie van de registrationhandlers.



Figuur 16: Ontwerp basisstation - Core package

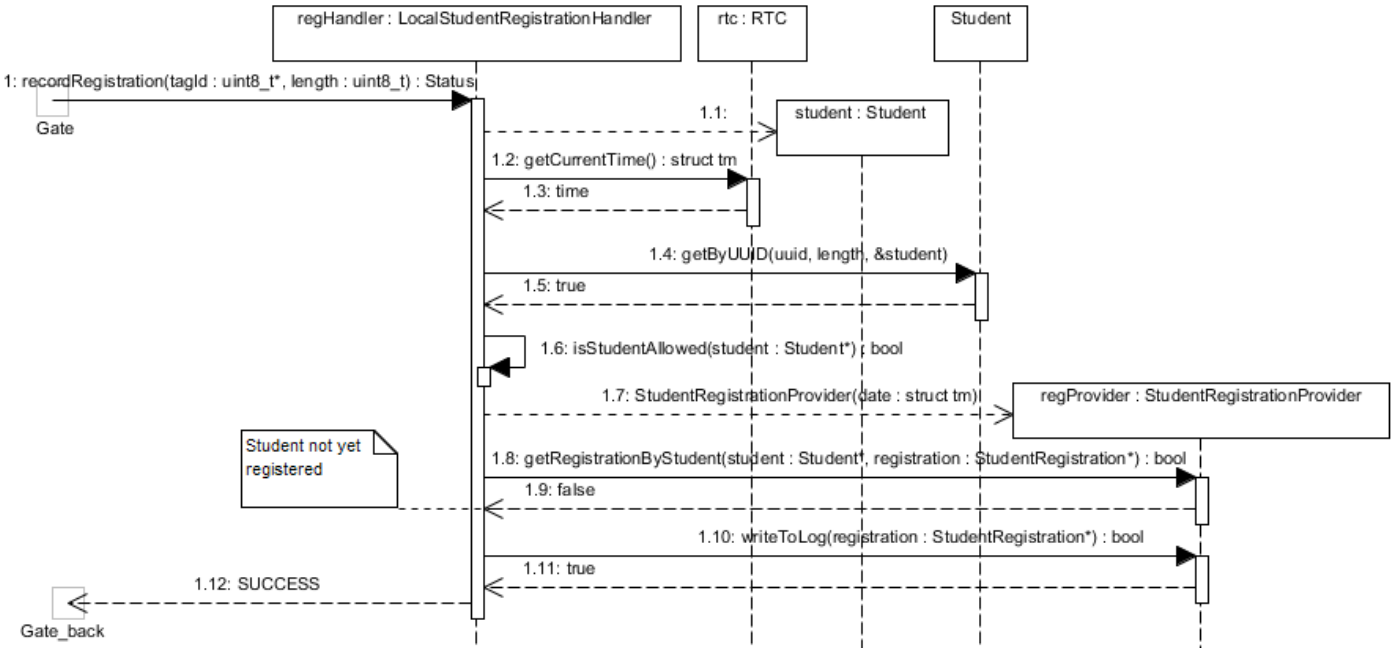
Verwerken van tags

Om bepaalde procedures binnen de software te verduidelijken, zijn er sequentiediagrammen gemaakt. Hiermee wordt verduidelijkt hoe bepaalde klassen samenwerken om zo een bepaalde taak te volbrengen. Twee sequentiediagrammen zullen hier toegelicht worden.



Figuur 17: Sequentiediagram - verwerken van tags

Figuur 17 toont een sequentiediagram, waarmee de opeenvolgende acties worden weergegeven bij het pollen naar een nieuwe beschikbare RFID-tag en de afhandeling daarvan. Na het opvragen van het tag-id, is er een gate te zien naar het sequentiediagram die getoond wordt in Figuur 18. Dit diagram toont dat de informatie over de student, gekoppeld aan het tag-id, wordt opgehaald. Vervolgens wordt gecontroleerd of de student wel op dit moment in het lokaal moet zijn en of de student niet al geregistreerd is. Tot slot wordt de informatie weggeschreven naar een log.



Figuur 18: Sequentiediagram - opslaan van studentregistratie

8.6 Communicatie met de SD-kaart

Bij de communicatie met de SD-kaart wordt er een bestandssysteem gebruikt, zodat de data kan worden bekeken en aangepast vanaf een computer. Het meest bekende bestandssysteem voor embedded applicaties is FatFs [37] en deze module is dan ook gebruikt. De module is zo opgebouwd dat het bestandssysteem los staat van de driver naar de SD-kaart. Voor een aantal microcontrollers is er een driver geschreven, maar niet voor de nRF51. Deze driver moest dus worden geport voordat de module gebruikt kon worden.

Uiteindelijk heeft de communicatie met de SD-kaart meer tijd gekost dan gepland. Elke SD-kaart werkt standaard met het SPI-protocol. Bij het porten van de driver voor de SD-kaart, is de code herschreven zodat het werkt met de SPI-library van de nRF51 SDK. De driver bleek niet te werken en de fout kon op verschillende plaatsen aanwezig zijn: bij de herschreven driver, bij de SPI-communicatie, of bij het soldeerwerk. Het was erg lastig achterhalen waar de fout zich precies bevond en uiteindelijk is er gebruik gemaakt van een Logic analyzer: een apparaatje met bijbehorende software, waarmee de signalen op de lijnen kunnen worden bekeken. In Figuur 19 is de werking te zien. Met behulp van de Logic Analyzer werd er een fout gevonden in zowel het soldeerwerk als in de SPI-instellingen.

Nadat de communicatie met de SD-kaart was geïmplementeerd, is er gewerkt aan de klassen voor het benaderen en opslaan van de gegevens. In eerste instantie was de applicatie zo opgezet, dat de demonstratiedata met groepen leerlingen en lesblokken steeds van de SD-kaart werd gelezen indien de informatie benodigd was. Hiervoor moesten verschillende bewerkingen worden gedaan, zoals het doorzoeken van mappen en bestanden naar de juiste groep en/of leerling. Het bleek dat deze operaties erg veel tijd in beslag namen, waardoor het opzoeken van de informatie over een leerling

in het ergste geval kon oplopen tot een seconde, wat onacceptabel is. Er ontstond het idee om bij opstarten van de microcontroller alle data over de groepen en leerlingen van de SD-kaart te lezen en op een efficiënte wijze op te slaan in het RAM-geheugen. Voor het prototype zou de grootte van de data gering blijven, zodat alle data in het aanwezige RAM-geheugen past. Het opzoeken van de informatie kost op deze manier minder dan een milliseconde, wat een grote vooruitgang is.

Figuur 19: Analyse met Logic Analyzer

8.7 Testen van de software

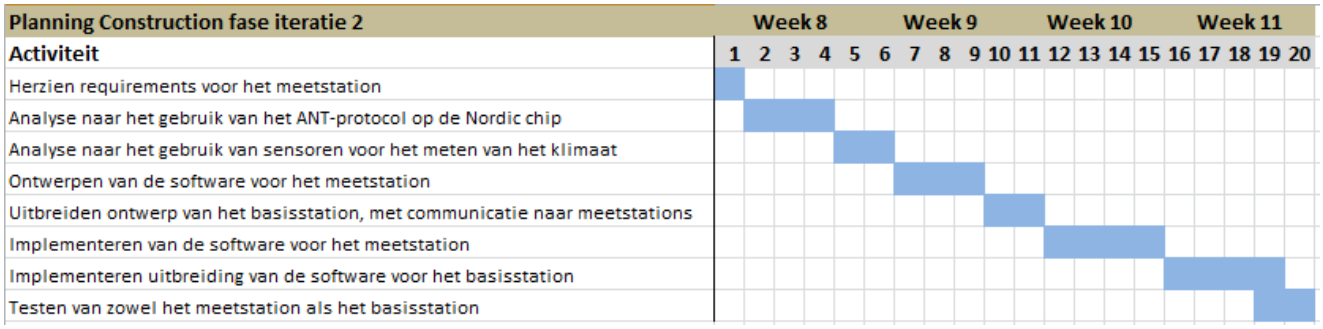
Gedurende de ontwikkeling van het basisstation, is de software regelmatig getest door middel van ontwikkeltests en integratietests. Er zijn testcases ontwikkeld die dienen als systeemtest, waarbij getest wordt of er aan de functionele eisen en niet-functionele eisen wordt voldaan. De testcases die ontwikkeld zijn voor deze iteratie, worden opnieuw gebruikt in de volgende iteraties, met aanvullende testcases. Een uitgebreidere beschrijving van welke tests er precies uitgevoerd zijn, met een beschrijving van de testcases, is te vinden in hoofdstuk 11.

8.8 Evaluatie construction fase iteratie 1

Achteraf gezien bleek het een goede keus om meer tijd in te plannen voor deze iteratie dan de oorspronkelijk ingeplande week. De 2,5 week waren hard nodig om tot een ontwerp te komen voor de software voor het basisstation en om dit ontwerp vervolgens te implementeren, waar veel tijd is gespendeerd aan de communicatie met de SD-kaart. Na deze iteratie is het grootste gedeelte van de software voor het basisstation af, zodat het mogelijk is om een RFID-tag te lezen en te verwerken aan de hand van de demonstratiedata van groepen met leerlingen en lesblokken. Ook kunnen klimaatmetingen worden gelogd in een bestand op de SD-kaart.

9 Construction fase 2: Meetstations en communicatie

De tweede iteratie van de construction fase stond in het teken van het ontwikkelen van de meetstations en het bewerkstelligen van de communicatie met het basisstation. Dit hoofdstuk beschrijft het doorlopen proces in deze iteratie. In Figuur 20 is de planning te zien die gemaakt is voor deze iteratie. Doordat er meer tijd was ingepland voor de vorige iteratie, begon deze iteratie een week later dan origineel gepland was.



Figuur 20: Planning construction fase iteratie 2

9.1 Eis aan koppelen van stations

Aan het begin van het project werd de eis gesteld dat het basisstation meetwaardes, van het klimaat binnen het klaslokaal, toegestuurd moet krijgen vanaf één of meer meetstations. Deze meetstations zouden dan direct met het basisstation communiceren. Voortkomend uit een gesprek met een collega ontstond echter het idee om het mogelijk te maken om meetstations aan elkaar te koppelen en dat tussenliggende stations fungeren als doorgeefluik. Op die manier zijn grotere afstanden te overbruggen, zodat het systeem in de toekomst ook zou kunnen werken in bijvoorbeeld een grote collegezaal op de Universiteit. Dit idee werd voorgelegd aan de opdrachtgever en die vond het een goed plan om dit idee door te voeren in het product, zodat dit is vastgelegd als een nieuwe systeemeis.

9.2 Analyse naar gebruik ANT-protocol op de nRF51 SoC

Zoals eerder beschreven is in de elaboration fase van het project een analyse uitgevoerd naar het ANT-protocol, omdat er voor gekozen is dit protocol te gebruiken als communicatieprotocol tussen de stations. In de tweede iteratie van de construction fase is vervolgens het gebruik van het softdevice onderzocht, dat toegang verschaft tot de ANT (en Bluetooth) protocol stack op de nRF51.

Het softdevice kan worden aangesproken door middel van een API in de nRF51 SDK. Deze uitgebreide API biedt toegang tot onder andere alle functionaliteiten van de ANT en Bluetooth stack. De API zelf is goed gedocumenteerd door middel van commentaar in Doxygen-formaat. Als startpunt is dat commentaar doorgelezen, om zo te achterhalen hoe de API gebruikt kan worden. Er is vanuit de fabrikant verder geen documentatie aanwezig voor het gebruik van ANT. De API is echter wel exact geïmplementeerd zoals staat beschreven in de officiële specificatie van het ANT-protocol [29], dus is deze specificatie gebruikt bij het instellen van de juiste parameters voor de ANT-kanalen. Daarnaast zijn er voorbeeldprogramma's meegeleverd, waarmee delen van de functionaliteit worden geïllustreerd. De code van deze programma's is uitvoerig bekeken en aangepast, om zo de precieze werking te achterhalen.

Het softdevice wordt als een apart binair bestand op de chip geflashed, waardoor de applicatie en het softdevice logisch gescheiden zijn. Vanuit de applicatie kan het softdevice worden aangesproken

door middel van supervisor calls (SVC-aanroepen), die te gebruiken zijn als normale functies. Met deze functies kunnen bijvoorbeeld de ANT-kanalen worden geconfigureerd en geopend. Terugkoppeling naar de applicatie vindt plaats door middel van een event gebaseerd systeem. Zo wordt er bij binnenkomende data een RX_EVENT gegenereerd voor een kanaal, waarna in de applicatie het datapakket binnengehaald kan worden.

9.3 Gebruik van sensoren voor klimaatmetingen

De meetstations moeten de temperatuur, luchtvochtigheid en CO2 kunnen meten, om zo inzicht te krijgen in de kwaliteit van het klimaat binnen een klaslokaal. Voor het uitvoeren van deze metingen zijn daarvoor bestemde sensoren nodig. Het kennisniveau van zowel de opdrachtgever als de afstudeerder was op dit gebied erg laag. Binnen Alflex zijn er echter meerdere experts aanwezig op dit gebied. Door middel van gesprekken met deze collega's is achterhaald welke sensoren er het best gebruikt kunnen worden en wat de correcte meetposities zijn binnen een ruimte. Het bleek dat alle drie de type klimaatmetingen meestal worden uitgevoerd op anderhalve meter boven de grond, op een plaats niet in de buurt van een raam of een deur. Alle drie de metingen kunnen dus verricht worden door één enkel meetstation, maar toch is er voor gekozen de meetstations modulair te maken. Dat wil zeggen dat een meetstation niet alle drie de metingen hoeft te verrichten. Hierdoor is het ook mogelijk om bijvoorbeeld de CO2 op een andere plaats te meten dan de temperatuur.

Op het gebied van klimaatsensoren is de SHT75 een veelgebruikte temperatuur- en luchtvochtigheidssensor die bij Alflex ook voor andere projecten wordt gebruikt. Deze sensor is dan ook gebruikt voor dit project. Luchtvochtigheidssensoren komen over het algemeen niet veel als losse sensoren voor, maar meestal in combinatie met een temperatuursensor. De reden hiervoor is dat de relatieve luchtvochtigheid afhankelijk is van de temperatuur.

Goede CO2-sensoren zijn duur en daarom werd er vanuit Alflex besloten om geen CO2-sensor aan te schaffen voor het te maken prototype. Door de afstudeerder is er aan de opdrachtgever als alternatief voorgesteld om de CO2-waardes dan maar te simuleren, zodat de rest van het systeem wel demonstreerbaar is. Indien het prototype in de toekomst daadwerkelijk verder uitgewerkt gaat worden, kan er alsnog een echte CO2-sensor worden gebruikt.

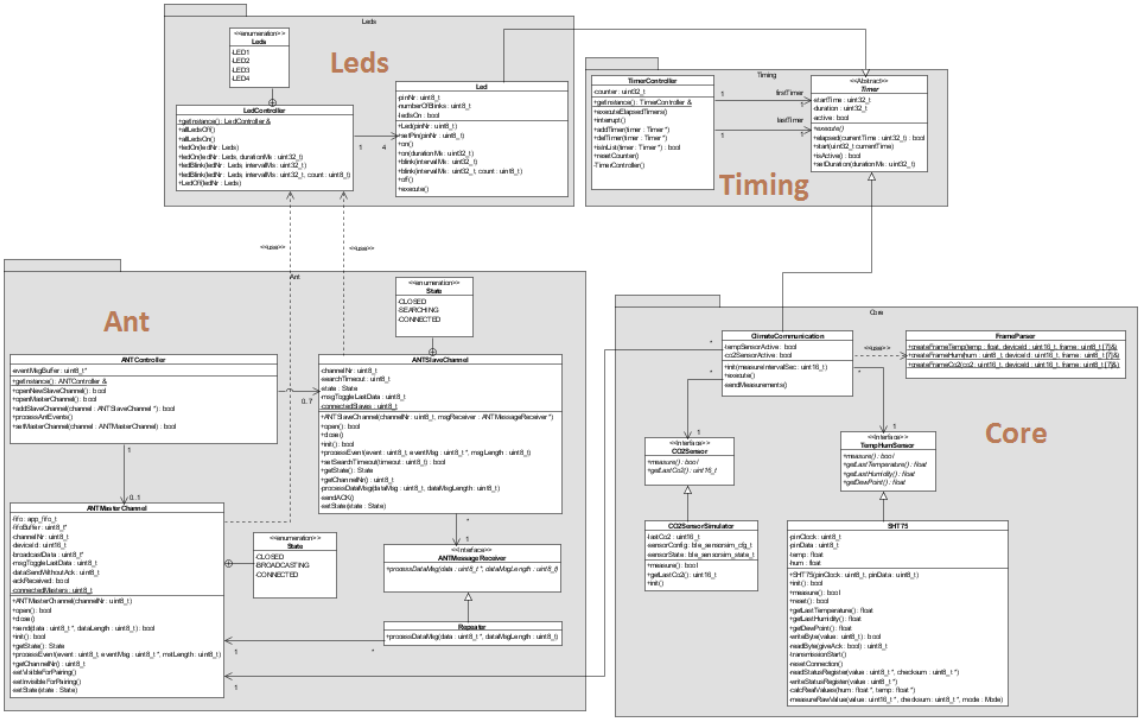
9.4 Ontwerp software meetstation

Net als voor de software voor het basisstation, is er ook voor de software voor een meetstation een ontwerp gemaakt door middel van UML-diagrammen. Om tot een eerste opzet van een klassendiagram te komen is dezelfde tactiek gehanteerd als bij het ontwerp voor het basisstation, zoals is te lezen in hoofdstuk 8.5.2. De procedure zal hier dan ook niet opnieuw toegelicht worden, maar in deze paragraaf zullen de belangrijkste elementen van het ontwerp worden besproken. Het gehele ontwerp is terug te vinden in het ontwerp document.

9.4.1 Opdeling in packages

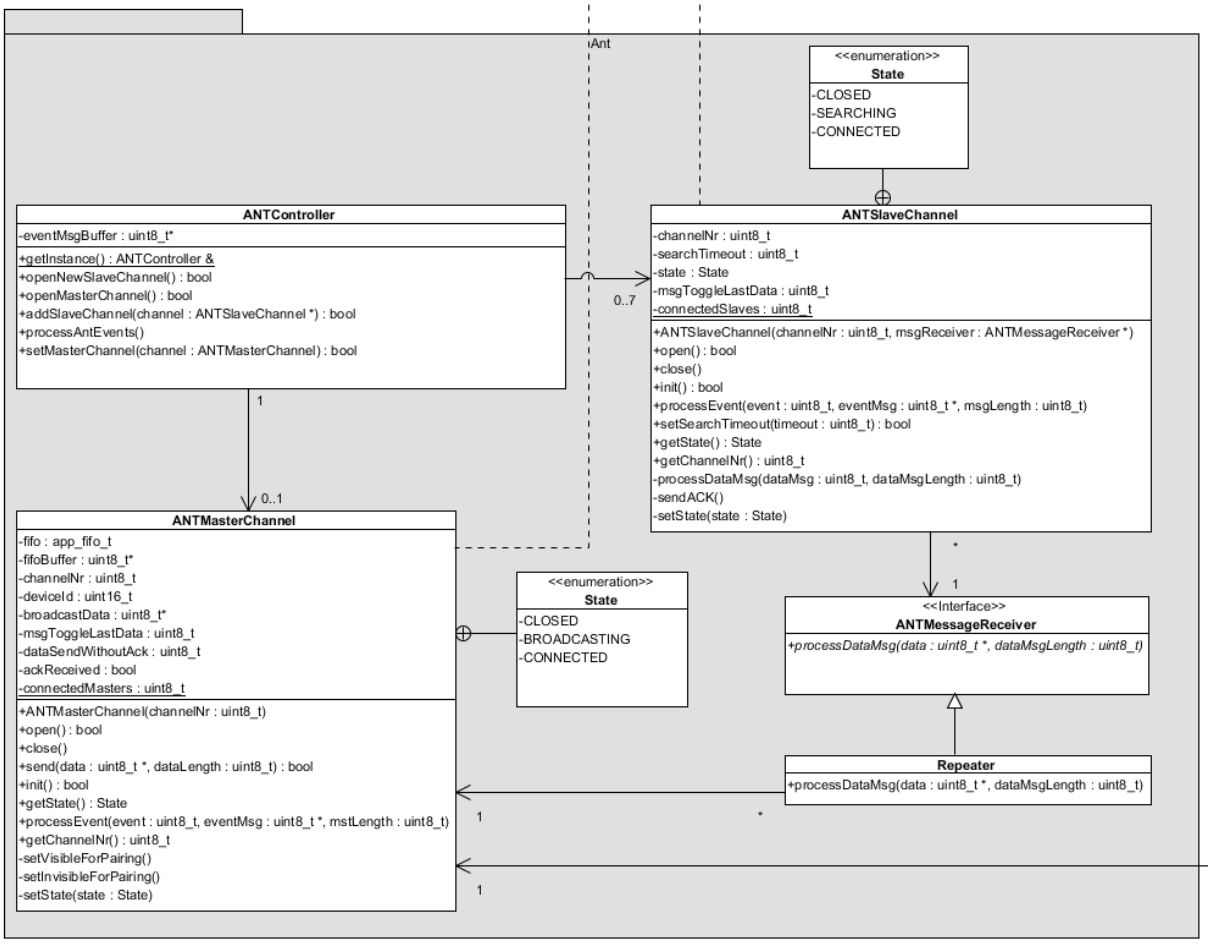
Ook de software voor het meetstation is opgedeeld in verschillende packages om zo meer structuur te geven aan de code en daarmee de uitbreidbaarheid en aanpasbaarheid te verhogen. Het klassendiagram van de software voor het meetstation is te zien in Figuur 21. Ook nu is getracht te zorgen voor een lage koppeling tussen de packages en een hoge cohesie binnen een package.

De *Timing* package is overgenomen van de software voor het basisstation. De *Leds* package bevat onder andere de klasse *LedController*, geïmplementeerd als een singleton, waarmee de leds kunnen worden aangestuurd op het hardware bordje. De klasse *Led* is daarbij een subklasse van *Timer* uit de *Timing* package, zodat een led kan knipperen of aangezet kan worden voor een bepaalde tijd. De *Core* package is verantwoordelijk voor de klimaatmetingen en het doorsturen van deze metingen door middel van de klasse *ANTMasterChannel* uit de *Ant* package.



Figuur 21: Klassendiagram meetstation

9.4.2 Ant package



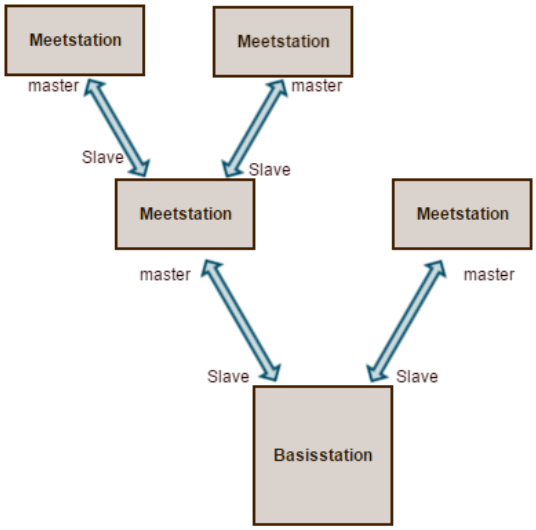
Figuur 22: Ontwerp meetstation - Ant package

De *Ant* package verzorgt de communicatie over het ANT-protocol. De klassen in deze package maken gebruik van de SVC-aanroepen naar het softdevice. De *ANTController* is verantwoordelijk voor het openen van de *ANTMasterChannel* en de *ANTSlaveChannels.* Binnen de applicatie is één master kanaal aanwezig, voor het verzenden van data naar een ander station, en zeven slave kanalen voor het ontvangen van data van maximaal zeven andere stations. De methode processAntEvents() van de *ANTController* moet steeds worden aangeroepen vanuit het hoofdprogramma. Door middel van deze methode wordt gecontroleerd of er events aanwezig zijn van de ANT stack en vervolgens worden deze events gedelegeerd naar het juiste object.

De *ANTSlaveChannel* stuurt binnenkomende data door naar een subklasse van de interface *ANTMessageReceiver*, die verantwoordelijk is voor het afhandelen van de data. Een mogelijkheid is om hiervoor een instantie van de subklasse *Repeater* te gebruiken, die binnenkomende data weer doorstuurt naar een ander station. Zo kan een station fungeren als een tussenliggend station. Een andere mogelijkheid is om een nieuwe subklasse te implementeren die de data op een andere manier afhandelt. Op deze manier is dit een losstaande package die ook in andere projecten is te gebruiken en waarbij door middel van overerving een eigen implementatie is te definiëren voor het afhandelen van binnenkomende data. Dit is zo ontworpen met het oog op het gebruik van deze package in de software voor het basisstation, waarover later meer beschreven wordt.

9.5 Communicatie over ANT

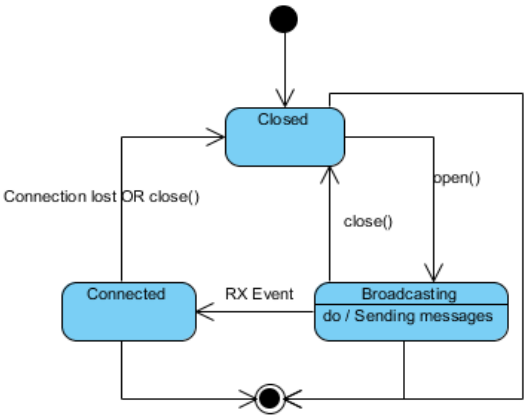
Zoals besproken in de vorige paragraaf heeft elk station maximaal zeven slave kanalen en één master kanaal. Het systeem bestaat uit één basisstation en maximaal tien meetstations. Een mogelijke topologie die hiermee te maken is, is te zien in Figuur 23.



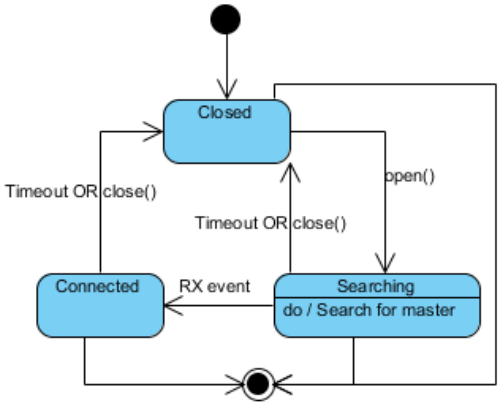
Figuur 23: Topologie stations

9.5.1 Werking van de ANT kanalen

Instanties van de klassen *ANTMasterChannel* en *ANTSlaveChannel* reageren anders op binnenkomende events en functieaanroepen, afhankelijk van de state waarin zij zich bevinden. Om dit te verduidelijken zijn er UML state-diagrammen gemaakt, waarmee gedrag in een bepaalde state wordt verduidelijkt. In Figuur 25 is de state diagram te zien van de *ANTMasterChannel*. Het kanaal kan worden geopend, waarna er continue berichten worden gestuurd naar de omgeving voor een bepaalde tijd. Als er een antwoord van een ander station binnen komt op hetzelfde kanaal, treedt er een RX event op en gaat het object in de connected state. Indien er voor een bepaalde tijd geen antwoord meer wordt ontvangen wordt de verbinding als verloren beschouwd en gaat het object naar de closed state. Het state diagram voor de *ANTSlaveChannel* is te zien in Figuur 24.



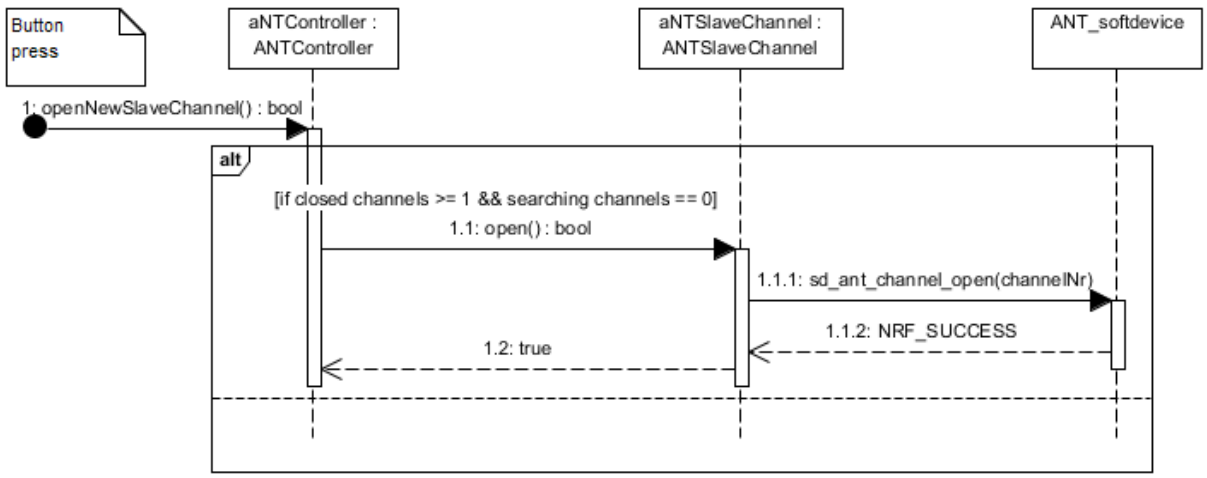
Figuur 25: State diagram ANTMasterChannel



Figuur 24: State diagram ANTSlaveChannel

9.5.2 Koppelen van stations

Een station moet weten met welk ander station er gecommuniceerd moet worden. In eerste instantie was bedacht om een netwerk te implementeren waarin elk station data doorstuurt naar alle omliggende stations en dat een station alle binnenkomende data ook weer doorstuurt, totdat de data uiteindelijk bij het basisstation terecht komt. Zo'n netwerk wordt ook wel een mesh-netwerk genoemd. Op deze manier hoeven twee stations niet manueel met elkaar worden verbonden, maar hoeft een station enkel aangezet te worden, waarna er een zelfregulerend netwerk ontstaat. De implementatie van zo'n netwerk bleek echter vrij complex en is niet zozeer nodig voor het te maken systeem, aangezien er in de meeste gevallen maar één of twee meetstations gebruikt zullen worden in een klaslokaal. Daarnaast is er dan nog het probleem dat stations niet moeten communiceren met stations uit andere lokalen. Dit idee is dan ook verworpen en vervolgens is er een manier bedacht om de stations manueel te koppelen, door het gebruik van de buttons op het bordje. Door op twee stations op een andere button te drukken, wordt er op het ene station een master kanaal geopend en op het andere station een slave kanaal. Door middel van ledjes op het bordje wordt er feedback gegeven aan de gebruiker over de status van een koppeling. Een slave kanaal kan enkel nog geopend worden als er nog een kanaal vrij is en als er niet al een ander kanaal aan het zoeken is. Deze procedure is verduidelijkt in Figuur 26.



Figuur 26: Sequentiediagram openen slave channel

Het ANT-protocol definieert een manier om aan te geven of een master kanaal al is verbonden. Hiervoor kan een zogenoemde pairing bit worden gezet. Een slave kanaal zal enkel zoeken naar een master kanaal waarbij het pairing bit nog niet is gezet.

9.5.3 Protocol over ANT

Bovenop het ANT-protocol is er nog een simpel protocol geïmplementeerd voor het versturen van de daadwerkelijke data. Elke klimaatmeting wordt verstuurd in een apart frame, zodat het protocol is uit te breiden als er in de toekomst een nieuw type klimaatmeting bij zou komen. Een frame bestaat altijd uit exact acht bytes, omdat met het ANT-protocol enkel datapakketten van exact acht bytes verstuurd kunnen worden.

Data ID	Data Type	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6

Tabel 15: Algemene opbouw protocol over ANT

Bovenstaand is te zien hoe de algemene opbouw is van een datapakket. De eerste byte geeft het Data ID aan. Dit geeft aan wat voor soort datapakket het is. De Data ID van een datapakket met een klimaatwaarde varieert echter steeds tussen 0 en 1, zodat de slave weet wanneer het een nieuw datapakket ontvangt en wanneer het nog een oud datapakket ontvangt. Het data type geeft aan om wat voor soort klimaatmeting het gaat. Het volledige protocol is beschreven in het ontwerp document.

9.6 Versturen klimaatmetingen

De meetstations meten met een opgegeven interval het klimaat en sturen deze gegevens vervolgens door. Daarbij is de software zo ontwikkeld dat er eerst wordt gecontroleerd welke sensoren er aangesloten zijn en met enkel die sensoren wordt er een meting verricht en wordt de meting doorgestuurd. Hierdoor kan elk meetstation dezelfde software bezitten, maar kan een gebruiker bepalen welke sensoren er worden aangesloten op het meetstation.

De SHT75 temperatuur en luchtvochtigheidssensor is te benaderen via een protocol dat erg lijkt op I2C, maar dat het net niet is. De eerder ontwikkelde *TWI* klasse voor de software voor het basisstation was dan ook niet te hergebruiken. De enige mogelijkheid tot communicatie was door dit protocol zelf te implementeren, waarbij op pinniveau de clock en data lijnen aangestuurd moeten worden. Hiervoor is gebruik gemaakt van de datasheet van de sensor en de voorbeeldcode voor een ander soort microcontroller, die beschikbaar is gesteld door de fabrikant van de sensor. De logic analyzer is hierbij weer gebruikt om de signalen op de lijnen te analyseren en zo te achterhalen of de sensor de verwachte data terugstuurt.

9.7 Uitbreiding software basisstation

Het basisstation moet ook in staat zijn om te communiceren over het ANT-protocol, zodat er gecommuniceerd kan worden met de meetstations. Daarnaast moeten de binnenkomende klimaatmetingen worden verwerkt. De software voor het basisstation uit de vorige iteratie is hiervoor uitgebreid.

De eerder besproken *Ant* en *Leds* packages zijn overgenomen uit de software voor de meetstations. Nu komt de keuze voor de eerder besproken interface *ANTMessageReceiver* tot zijn recht, omdat er door middel van een nieuwe overerving van deze interface een andere subklasse is geïmplementeerd voor het afhandelen van binnenkomende data, terwijl de *Ant* package niet aangepast hoeft te worden.

De klassen die verantwoordelijk zijn voor het verwerken van de binnenkomende klimaatwaardes waren in de vorige iteratie al geïmplementeerd. Echter door nieuwe inzichten en veranderende

systeemeisen zijn deze klassen aangepast. Doordat het aantal meetstations variabel is en doordat de meetstations niet altijd meer direct aan een basisstation gekoppeld zitten, weet het basisstation niet van hoeveel meetstations het data kan verwachten. Daarnaast hoeft niet elk meetstation elk type klimaatmeting te verrichten. Er moest een routine worden bedacht waarmee dynamisch bepaald kan worden wanneer het basisstation bijvoorbeeld alle temperatuurmetingen binnen heeft gekregen. Op dat moment kan bijvoorbeeld het gemiddelde worden berekend van alle temperatuurmetingen of kan de hoogste waarde worden genomen. Laatstgenoemde is vastgelegd als systeemeis, dus in dit geval wordt de hoogste waarde genomen. Vervolgens worden elk halfuur de dan huidige klimaatwaardes gelogd in een bestand op de SD-kaart.

De routine komt kortgezegd op het volgende neer. Elk meetstation stuurt met hetzelfde interval de gemeten waardes naar het basisstation. Per type klimaatmeting onthoudt het basisstation van welk meetstation de meting afkomstig is, door middel van een uniek nummer dat het meetstation meestuurt. Op het moment dat er opnieuw een meting binnen komt van hetzelfde meetstation, weet het basisstation dat in de tussentijd alle aanwezige meetstations hun waardes hebben doorgestuurd. Op deze manier kunnen er dynamisch meetstations worden bijgeplaatst of verwijderd, maar blijft het algoritme wel werken.

9.8 Evaluatie construction fase iteratie 2

De iteratie is met succes afgerond binnen het aantal geplande weken. De iteratie was zelfs een aantal dagen eerder afgerond, waardoor de volgende iteratie iets eerder kon beginnen. Aan het eind van deze iteratie was de software ontwikkeld voor de meetstations en konden de meetstations communiceren met het basisstation via het ANT-protocol, waarna de gegevens op het basisstation worden verwerkt. Hiervoor is ook de software voor het basisstation uitgebreid en deels aangepast.

9.9 Verandering in de planning voor iteratie 3

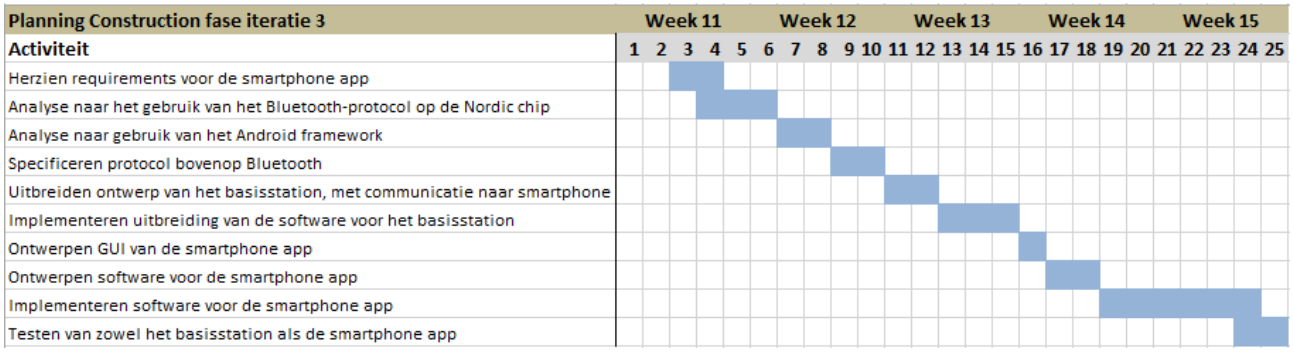
Aan het eind van de tweede iteratie van de construction fase is weer een gedetailleerdere planning gemaakt voor de daaropvolgende iteratie, de iteratie waarin de Android applicatie is ontwikkeld. Ondanks dat het project nog op schema lag volgens de globaal opgestelde planning, is er voor gekozen om meer tijd in te plannen voor het maken van de Android applicatie. Hier is voor gekozen omdat werd gedacht dat de drie weken die hiervoor gepland stonden in de globale planning niet genoeg zouden zijn om een volledig functionele Android applicatie te ontwikkelen.

De opdrachtgever heeft namelijk aangegeven dat de Android applicatie een belangrijk middel is om het product te presenteren. Deze applicatie is namelijk de enige visuele representatie van het product en moet daarom ook aantrekkelijk ogen. Er zal dus ook tijd gespendeerd worden aan het uiterlijk van de applicatie. Daarnaast is er in de globale planning geen rekening gehouden met het feit dat de software voor het basisstation ook nog uitgebreid moet worden, om via Bluetooth Smart te kunnen communiceren. Door nieuwe inzichten gedurende het project is bekend geworden dat dit meer werk gaat kosten dan gedacht. Daarom is er voor gekozen dat de derde iteratie de laatste iteratie is van de construction fase en vijf weken zal duren in plaats van drie weken.

Dat er meer tijd ingepland wordt voor de Android applicatie, had wel als gevolg dat er geen tijd meer zou zijn voor de desktopapplicatie, waarmee op een makkelijke wijze demonstratiedata kan worden geconfigureerd en op het basisstation kan worden gezet. Zoals is beschreven in hoofdstuk 8.2, is het nu echter ook mogelijk deze data handmatig te configureren door middel van een komma gescheiden bestand. Daar is destijds voor gekozen voor het geval er geen tijd meer zou zijn voor de desktopapplicatie, wat op dit moment een goede keuze blijkt. De verandering van de planning is aan de opdrachtgever voorgelegd en de opdrachtgever heeft ermee ingestemd.

10 Construction fase 3: Android applicatie

Zoals beschreven in het vorige hoofdstuk staat de derde iteratie van de construction fase in het teken van de ontwikkeling van een Android applicatie en is dit tevens de laatste iteratie van de construction fase. De planning voor deze iteratie is te zien in Figuur 27. In dit hoofdstuk wordt het doorlopen proces gedurende deze iteratie besproken.



Figuur 27: Planning construction fase iteratie 3

10.1 Detailleren requirements smartphone applicatie

Aan het begin van deze iteratie zijn er opnieuw gesprekken gevoerd met de opdrachtgever om zo de eerder vastgelegde (globale) eisen voor de smartphone applicatie te detailleren. Hiervoor zijn weer use cases opgesteld, die vervolgens geverifieerd zijn door de opdrachtgever, en aan de hand van deze use cases is er een lijst opgesteld met de systeemeisen, geprioriteerd volgens de MoSCoW-methode. Hiervoor is weer dezelfde procedure doorlopen zoals is beschreven in hoofdstuk 5.2, en de eisen zijn vastgelegd in het requirements document.

Twee van de niet-functionele eisen zijn dat de applicatie moet kunnen draaien op Android en communiceert met het basisstation door middel van Bluetooth Smart. Door middel van de applicatie moet inzichtelijk worden gemaakt welke leerlingen er aanwezig en afwezig waren op een bepaald tijdstip. Ook moet het huidige klimaat binnen het klaslokaal worden getoond en moet er een mogelijkheid zijn om de historie te bekijken van de klimaatwaardes gedurende een bepaalde dag, week of maand. Tot slot moet de tijd op het gekoppelde basisstation ingesteld kunnen worden.

De applicatie wordt gemaakt om het prototype demonstreerbaar te maken op bijvoorbeeld scholen, om zo interesse te kunnen wekken voor het product. Als het prototype verder wordt ontwikkeld zal de applicatie niet meer met het basisstation communiceren, maar met een centrale server. De opzet van de applicatie zal dan veranderen en ook zal de data dan op een andere manier worden gepresenteerd, aangezien er dan sprake is van een zeer grote hoeveelheid aan data. Voor nu is het genoeg dat de applicatie enkel een indruk geeft van de functionaliteiten en mogelijkheden van het product.

10.2 Analyse naar gebruik van Bluetooth Smart op de nRF51 SoC

In de vorige iteratie is er een analyse uitgevoerd naar het gebruik van het ANT-protocol door middel van het softdevice. In deze iteratie is er een soortgelijke analyse uitgevoerd, maar dan naar het gebruik van Bluetooth Smart. Vanuit de fabrikant Nordic is er, in tegenstelling tot ANT, voor Bluetooth Smart erg veel documentatie beschikbaar gesteld. Er zijn dan ook meerdere tutorials en andere documenten gebruikt bij het achterhalen van de werking van Bluetooth Smart door middel van het softdevice. Daarnaast is er veel voorbeeldcode beschikbaar en is de API goed beschreven.

In de basis kan het softdevice op dezelfde manier worden gebruikt als bij ANT, ook de communicatie met de Bluetooth stack is gebaseerd op SVC-aanroepen en terugkoppeling door middel van events. Echter zit het ANT-protocol een stuk eenvoudiger in elkaar en wordt de gebruiker geacht om zelf de communicatie en pairing procedure juist te implementeren. De configuratie en gebruik van Bluetooth Smart is daarentegen een stuk uitgebreider, omdat al deze zaken door de stack zelf worden afgehandeld.

Bluetooth Smart is, in tegenstelling tot de klassieke Bluetooth, niet ontworpen om grote hoeveelheden data te versturen. De indeling van profielen, services en characteristics is gericht op het af en toe verzenden van kleine hoeveelheden data van maximaal 20 bytes, zoals een hartslagmeetwaarde. Tussen het basisstation en de Android applicatie moet een relatief grote hoeveelheid data worden doorgestuurd, dus er moest een manier worden gevonden om dat te bereiken. Hiervoor zijn verschillende standaard Bluetooth Smart profielen onderzocht, maar geen van de standaard profielen is geschikt voor het beoogde doel. Bij de klassieke Bluetooth is er standaard al een profiel aanwezig, waarbij een UART-verbinding wordt gesimuleerd. Door middel van dat profiel (het SPP-profiel) kan er gecommuniceerd worden alsof er een UART wordt gebruikt en zo kunnen er grotere hoeveelheden data worden verstuurd. [38] Voor Bluetooth Smart is geen soortgelijk profiel standaard aanwezig. Nordic heeft echter een stuk code beschikbaar gesteld, waarin zo'n profiel zelf is geïmplementeerd. Hiervoor is één service gebruikt, genaamd de UART-service, met daarin twee characteristics: TX en RX. Door het lezen en schrijven van en naar de characteristics, wordt er een UART-verbinding gesimuleerd waarover steeds maximaal 20 bytes verstuurd kunnen worden.

Er is gekozen om van de Nordic UART-service gebruik te maken, omdat dat de snelste en meest flexibele manier is om data over te sturen. Het geeft de vrijheid om zelf een protocol te definiëren waarover gecommuniceerd kan worden. Op deze manier kunnen er gespecificeerde frames met data worden verstuurd, waarin een frame wordt opgedeeld in blokken van 20 bytes. Door gebruik te maken van de Nordic UART-service wordt er veel tijd bespaard, omdat er nu niet zelf een soortgelijk profiel geïmplementeerd hoeft te worden. De UART-service is beschikbaar gesteld in C. Deze service is vervolgens herschreven naar C++, opgedeeld in klassen, en geïntegreerd in het eigen project.

10.3 Analyse naar het Android framework

Android is een open source besturingssysteem voor mobiele telefoons en tablets, ontwikkelt door Google. Iedereen is vrij om applicaties te ontwikkelen voor Android. Een applicatie wordt geschreven met behulp van de Android SDK in de Java programmeertaal.

De afstudeerder had al enige ervaring met het ontwikkelen van applicaties voor Android in Java. Deze kennis was ondertussen wel wat verouderd, dus is er eerst opnieuw onderzocht hoe het Android framework in elkaar zit. Hiervoor is informatie gehaald uit het boek *Beginning Android 4 Application Development* [39] en van de officiële specificaties op de Android developers website. Het grootste gedeelte van een Android applicatie bestaat uit klassen die subklassen zijn van klassen uit het framework. Daarbij maken deze klassen weer gebruik van andere componenten uit het framework. Het is dus noodzakelijk om eerst de benodigde kennis te vergaren voordat een Android applicatie kan worden geschreven. Wat volgt is een zeer korte beschrijving van het Android framework. Deze kennis is benodigd voor een beter begrip van het verdere verslag.

Een Android applicatie bestaat in de basis uit de volgende vier typen componenten: [39]

- *Activities*: Een activity wordt gebruikt voor het weergeven van één scherm. Een activity bevat meerdere views, zoals een tekstveld of een afbeelding. Deze activities staan los van elkaar en kunnen afzonderlijk worden gestart. Vanaf Android 3.0 kan een activity bestaan uit meerdere *fragments*, wat in de basis een soort mini activities zijn.

- *Services*: Een service is een component dat op de achtergrond een langdurige taak uitvoert. Een service heeft geen gebruikersinterface en wordt aangestuurd vanuit een activity. Als een applicatie wordt afgesloten kan de service blijven bestaan. Een muziekspeler gebruikt bijvoorbeeld een service om de muziek op de achtergrond af te spelen, ongeacht in welk scherm de gebruiker zich bevindt.
- *Content providers:* Een content provider biedt een eenzijdige interface naar een achterliggende dataset. Deze data kan in een lokale database aanwezig zijn, maar ook ergens op een server. Een content provider kan vanuit meerdere applicaties gebruikt worden.
- *Broadcast receivers:* Een broadcast receiver is een component dat in staat is tot het ontvangen van broadcasts. Een broadcast is een bericht dat door iedere applicatie en door het systeem zelf verstuurd kan worden. De broadcast receivers die in dat bericht geïnteresseerd zijn vangen het bericht op en voeren daaropvolgend een handeling uit. Een voorbeeld is een broadcast die wordt verzonden door het systeem als het scherm uit gaat. Een applicatie zou dat bericht op kunnen vangen en zo kunnen bijhouden hoe vaak het scherm aan en uit gaat op een dag.

Nordic heeft de source code vrijgegeven van een Android applicatie, waarin dezelfde Bluetooth UART-service is geïmplementeerd als in de eerder besproken voorbeeldapplicatie voor de nRF51. Ook deze service is gebruikt, om zo te communiceren met de Bluetooth UART-service die draait op het basisstation.

10.4 Communicatie over Bluetooth Smart

Zoals eerder besproken wordt er een Bluetooth Smart profiel gebruikt waarmee een UART-verbinding wordt gesimuleerd tussen het basisstation en de Android applicatie. Er moest nog wel een protocol worden geïmplementeerd, waarover beide apparaten met elkaar praten, zodat de binnenkomende bytes geïnterpreteerd kunnen worden. Door middel van Bluetooth Smart kunnen er blokken data verstuurd worden van maximaal 20 bytes, waarbij wordt gegarandeerd dat de data aan komt. De data die van het basisstation naar de Android applicatie wordt verstuurd bedraagt echter soms meer dan 20 bytes. Daarom is er voor gekozen de data te verpakken in een frame met een vaste structuur. Dit frame wordt dan opgedeeld in blokken van 20 bytes verstuurd. De structuur van het frame is te zien in Tabel 16. Het volledige protocol met een uitleg van alle commando's en de opbouw van het datapakket is te vinden in het ontwerp rapport.

Commando	Aantal bytes datapakket	Datapakket	CRC-16

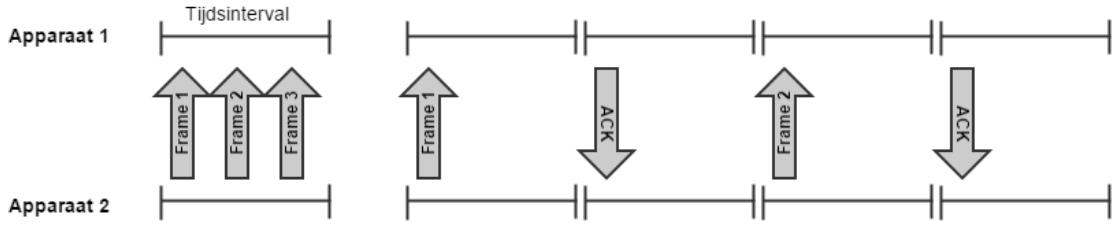
Tabel 16: Structuur frame over Bluetooth Smart

Ondanks dat gegarandeerd wordt dat de datablokken van 20 bytes correct aan komen, is er toch voor gekozen om een CRC over het frame te berekenen en mee te sturen. Door middel van de CRC wordt gecontroleerd of alle bytes correct zijn aangekomen. Doordat de frames opgedeeld worden in aparte blokken, deze blokken worden gebufferd en verstuurd, en na het versturen weer achter elkaar worden gezet, zou het kunnen voorkomen dat er ergens in dit proces iets fout gaat waardoor het frame corrupt raakt. Een corrupt frame kan door middel van de CRC worden gedetecteerd.

Om te bepalen hoe de uitwisseling van de data het best kan verlopen, is eerst onderzocht hoe de maximale doorvoersnelheid kan worden bereikt over de gesimuleerde UART-service. Hiervoor is op het basisstation het voorbeeldprogramma met de Nordic UART-service gebruikt, in combinatie met de voorbeeldapplicatie voor Android. Hiermee is geanalyseerd hoe lang het duurt om een vaste hoeveelheid bytes over te sturen, waarbij de configuratieparameters steeds zijn aangepast. Zo werd achterhaald dat Bluetooth Smart een vast tijdsinterval gebruikt om de datablokken over te sturen. Afhankelijk van de mogelijkheden van het apparaat dat fungeert als de central, is het mogelijk om één tot zes datablokken te verzenden binnen één tijdsinterval. Ook het minimaal mogelijke tijdsinterval is afhankelijk van de mogelijkheden van de central en bedraagt in het gunstigste geval

7.5 ms. De maximale doorvoersnelheid kan dus per type smartphone verschillen. De smartphone waarop de ontwikkeling plaatsvond, ondersteunt een tijdsinterval van minimaal 30 ms.

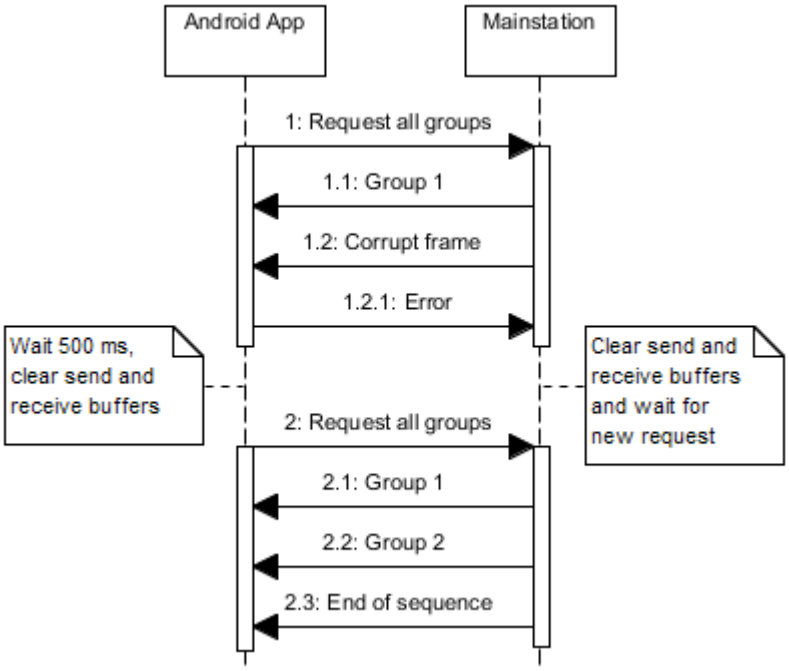
In eerste instantie was bedacht om na elk verstuurd frame een acknowledge terug te sturen, om zo aan te geven dat het volgende frame verstuurd kan worden. Echter was nu duidelijk dat er ook meerdere datablokken verstuurd kunnen worden binnen één tijdsinterval. In het geval een frame binnen één datablok past, kunnen er dus meerdere frames verstuurd worden binnen een tijdsinterval. Het antwoord van de andere partij vindt echter altijd plaatst op het eerstvolgende tijdsinterval. Door het versturen van een acknowledge na elk frame is dus het voordeel weg om meerdere frames te kunnen versturen binnen één tijdsinterval, waardoor de doorvoersnelheid drastisch omlaag gaat. Dit wordt verduidelijkt in Figuur 28.



Figuur 28: Twee opties voor versturen van frames

Door het nieuwe inzicht is er voor gekozen om geen acknowledge meer te sturen na elk ontvangen frame, maar enkel een error terug te sturen in het geval er iets mis is gegaan. In het geval de Android applicatie dan bijvoorbeeld alle groepen opvraagt, verstuurt het basisstation meerdere frames na elkaar, met in elk frame de data voor één groep. Dit is weergegeven in Figuur 29. Als er iets mis gaat gedurende de communicatie begint de aanvraag opnieuw. Hierdoor is geen complexe error afhandeling nodig om enkel de mislukte frames opnieuw te verzenden. Het nadeel is dat dit de communicatie enorm vertraagd, echter zal het niet vaak voorkomen dat frames corrupt raken, aangezien Bluetooth Smart garandeert dat de datablokken correct aankomen.

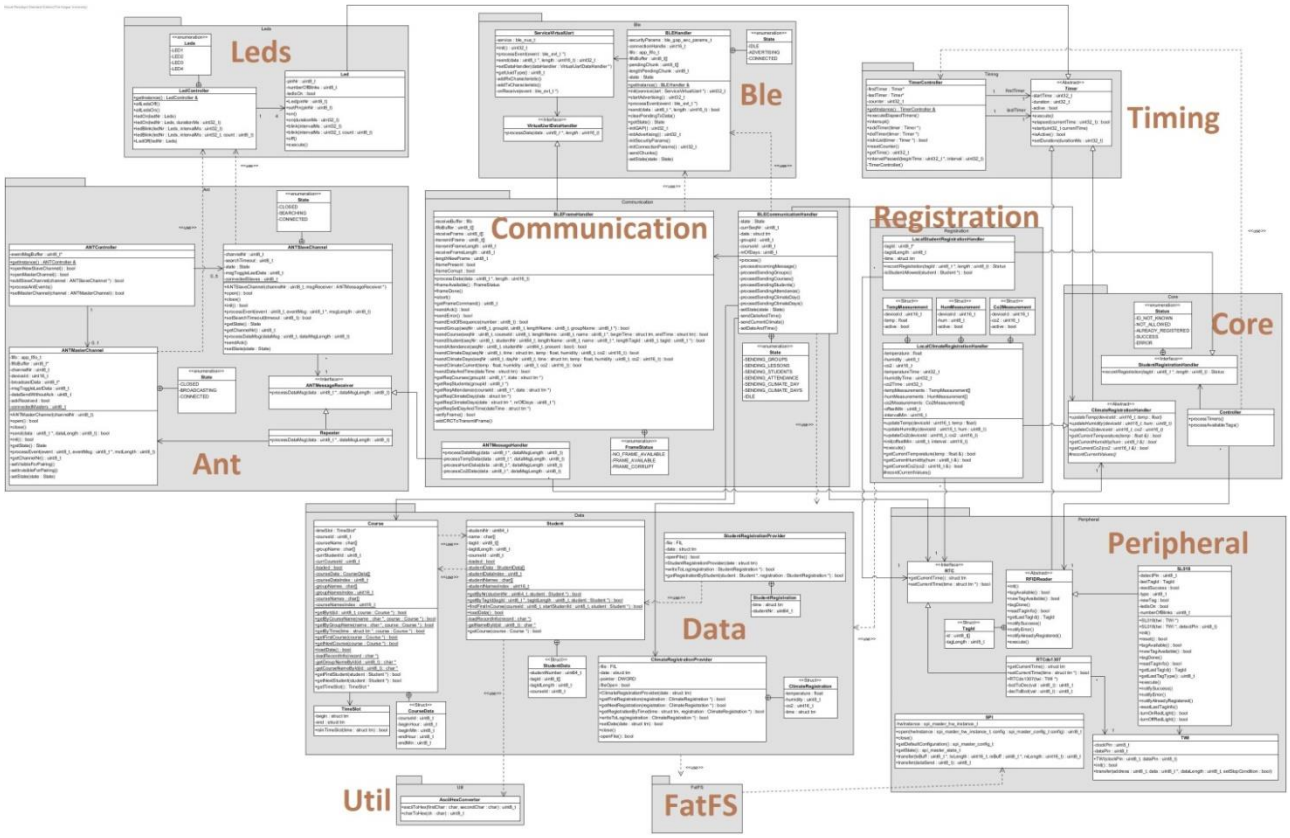
Er zijn meerdere diagrammen gemaakt zoals het diagram uit Figuur 29, om de communicatie tussen het basisstation en de Android app weer te geven, waarbij elk diagram een bepaalde situatie duidelijk maakt. Hiervoor zijn UML sequentie diagrammen gebruikt. Normaal gesproken worden deze diagrammen gebruikt om aanroepen tussen objecten weer te geven, maar in dit geval is het diagram gebruikt voor een ander doeleinde, namelijk het verduidelijken van de communicatie. De diagrammen zijn te vinden in het ontwerp rapport.



Figuur 29: Communicatie tussen basisstation en Android app

10.5 Software basisstation uitgebreid met Bluetooth communicatie

Het ontwerp van de software voor het basisstation zoals deze is gemaakt in de eerste iteratie van de construction fase (hoofdstuk 8), is uitgebreid in de tweede iteratie voor de communicatie met de meetstations, en is in deze derde iteratie opnieuw uitgebreid met functionaliteit voor de communicatie met de smartphone. In het vorige hoofdstuk is te lezen dat de eerder beschreven packages *Ant* en *Leds* zijn overgenomen van de software voor de meetstations en dat er een *Communication* package was bijgekomen. In deze laatste iteratie is er een *Ble* package bijgekomen en is de *Communication* package uitgebreid. De overige packages zijn hetzelfde gebleven. De definitieve versie van het klassendiagram voor de software van het basisstation is te zien in Figuur 30.



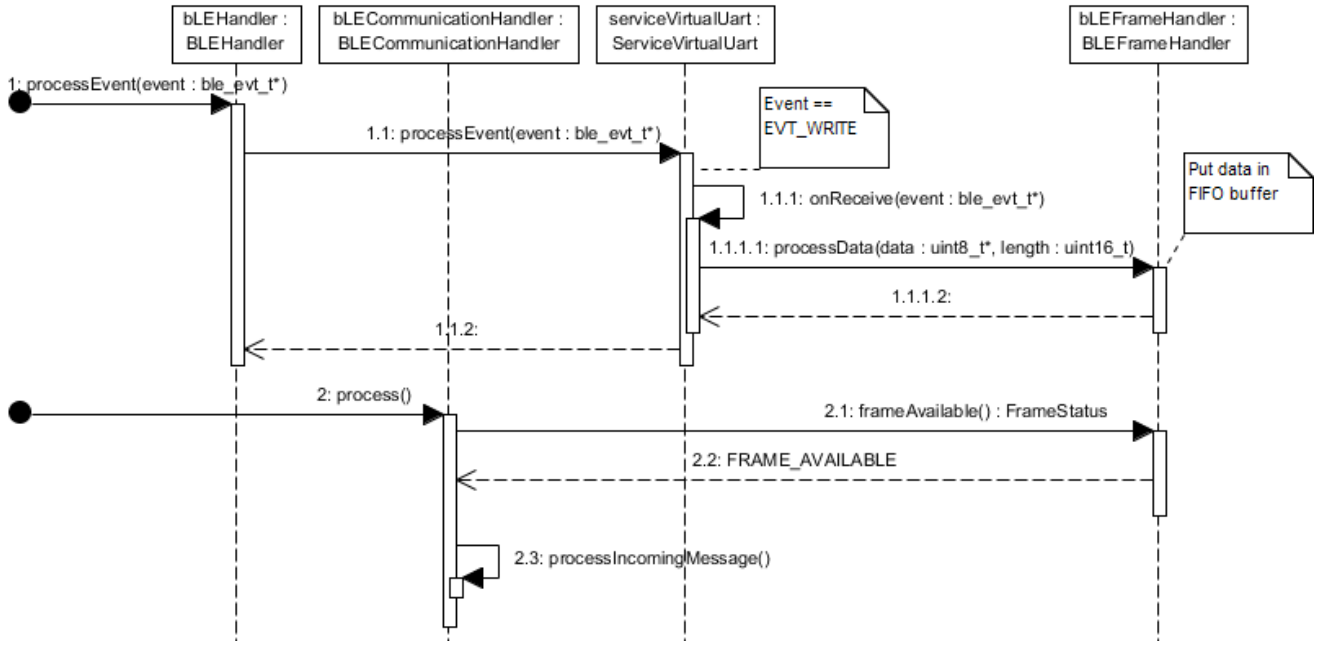
Figuur 30: Klassendiagram basisstation definitieve ontwerp

10.5.1 Verzenden en ontvangen van data over Bluetooth Smart

De *Ble* package is verantwoordelijk voor het versturen en ontvangen van data over Bluetooth Smart. Hiervoor is de herschreven code gebruikt van de eerder besproken Nordic Uart-service. De *Communication* package is uitgebreid met klassen die de communicatie over Bluetooth Smart regelen en de frames maken en parsen. Deze package heeft echter wel een hogere koppeling gekregen met andere packages, doordat er veel benodigde informatie wordt opgevraagd bij de andere packages.

Verzenden en ontvangen van data via Bluetooth Smart verloopt zowel via het hoofdprogramma als via events die gegenereerd worden door de Bluetooth stack. Deze events zijn geïmplementeerd in de vorm van interrupts met de hoogste prioriteit en onderbreken dus het hoofdprogramma. Het is doorgaans een goed uitgangspunt om zo min mogelijk activiteiten uit te voeren in een interrupt, zodat het hoofdprogramma niet te lang wordt onderbroken. Daarom wordt de binnenkomende data bij het afhandelen van het BLE event enkel in een fifo-buffer geplaatst. Vanuit het hoofdprogramma

wordt deze data vervolgens verwerkt. Dit is verduidelijkt in Figuur 31. De methode *processEvent()* wordt aangeroepen vanuit de event interrupt en de *process()* methode vanuit het hoofdprogramma.



Figuur 31: Sequentiediagram – Afhandelen binnenkomende BLE data

10.5.2 Afhandelen van binnenkomende aanvragen

Het basisstation reageert alleen op een aanvraag vanuit de smartphone. De smartphone zal altijd de uitwisseling van data initiëren. Het verwerken van sommige aanvragen duurt relatief kort, zoals het versturen van de huidig ingestelde datum en tijd. Andere aanvragen duren langer om te verwerken.

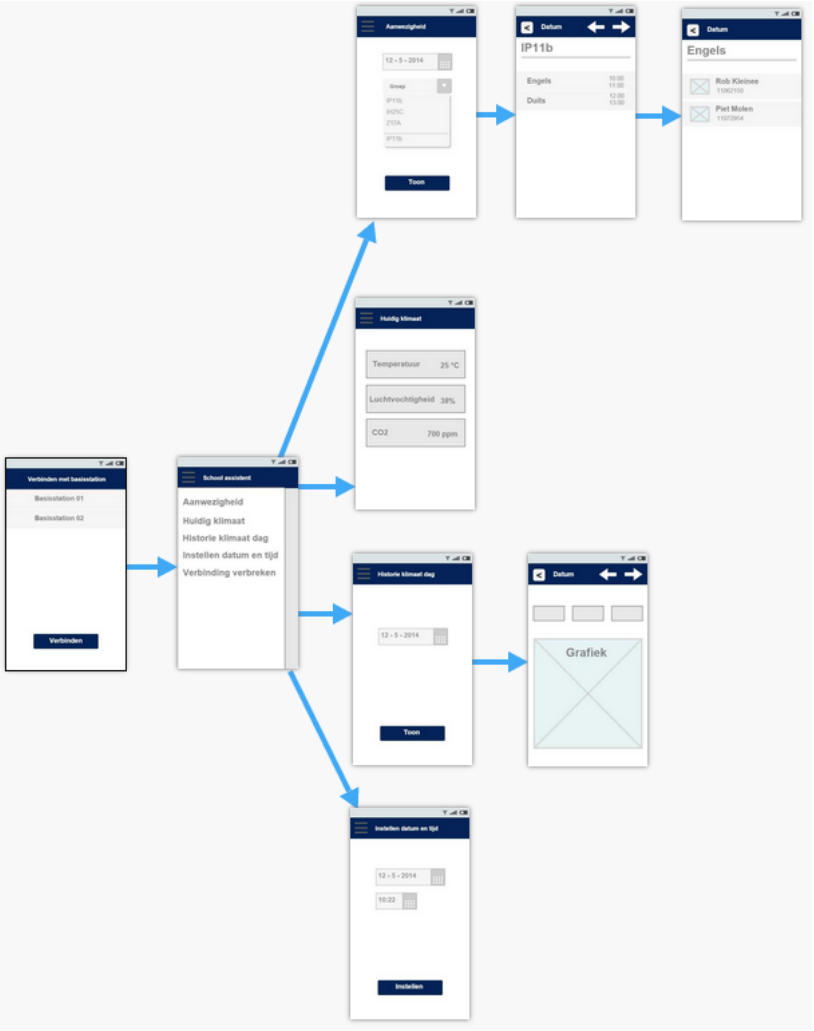
Er zijn veel tests uitgevoerd om te achterhalen wat de maximale verwerkingstijd is in het meest ongunstige geval. Dit is gedaan door alle aanvragen eens uit te voeren en daarbij te werken met veel data. Door het tijdelijk bijhouden van deze operaties met software timers kon worden achterhaald hoe lang elke aanvraag precies duurt. Vervolgens zijn deze timers gebruikt om nader te onderzoeken waar de vertraging precies zat, zoals het ophalen van de data van een SD-kaart of juist het versturen over Bluetooth Smart. Het bleek dat vooral de SD-kaart de grootste vertraging geeft.

Bij sommige aanvragen, zoals de aanvraag voor het verkrijgen van de aanwezigheid van alle leerlingen in een les, moet er zoveel informatie van de SD-kaart worden gehaald en worden teruggestuurd over Bluetooth Smart, dat de verwerking in het ongunstigste geval soms wel langer dan een seconde kan duren. De rest van het systeem moet echter ook blijven werken. Een van de niet-functionele eisen is dat een RFID tag die voor de lezer wordt gehouden, binnen een seconde afgehandeld moet zijn. Deze eis is niet altijd te halen als toevallig net door middel van de smartphone gecommuniceerd wordt met het basisstation. Als oplossing is bedacht om een softwaretimer bij te houden. Op het moment dat het verwerken van een aanvraag langer dan 200 ms duurt, wordt de verwerking gepauzeerd en wordt eerst de rest van het programma uitgevoerd. Daarna wordt het verwerken van de aanvraag weer vervolgd. Hierdoor blijft het gehele systeem operatief.

10.6 Ontwerp GUI van de Android applicatie

Alvorens met de ontwikkeling van de Android applicatie te beginnen, is er een flowdiagram gemaakt van de applicatie. Dit diagram toont globaal de indeling van elk scherm en de samenhang tussen deze schermen, waarbij de nadruk meer op de laatstgenoemde ligt. Het diagram geeft namelijk geen exact ontwerp van elk scherm, maar geeft meer een indicatie van de inhoud van een scherm. Het diagram is gemaakt om zo door middel van een visuele representatie tot een overeenstemming te komen met de opdrachtgever over de structuur van de applicatie. Het kost namelijk minder tijd om in dit stadium de structuur van de applicatie te wijzigen, dan wanneer de schermen al daadwerkelijk geïmplementeerd zijn. De feedback van de opdrachtgever is vervolgens weer verwerkt in het diagram. Zo was er eerst een design voorgesteld waarin veel de kleur rood terug kwam, echter prefereerde de opdrachtgever de kleuren uit het Alflex-logo.

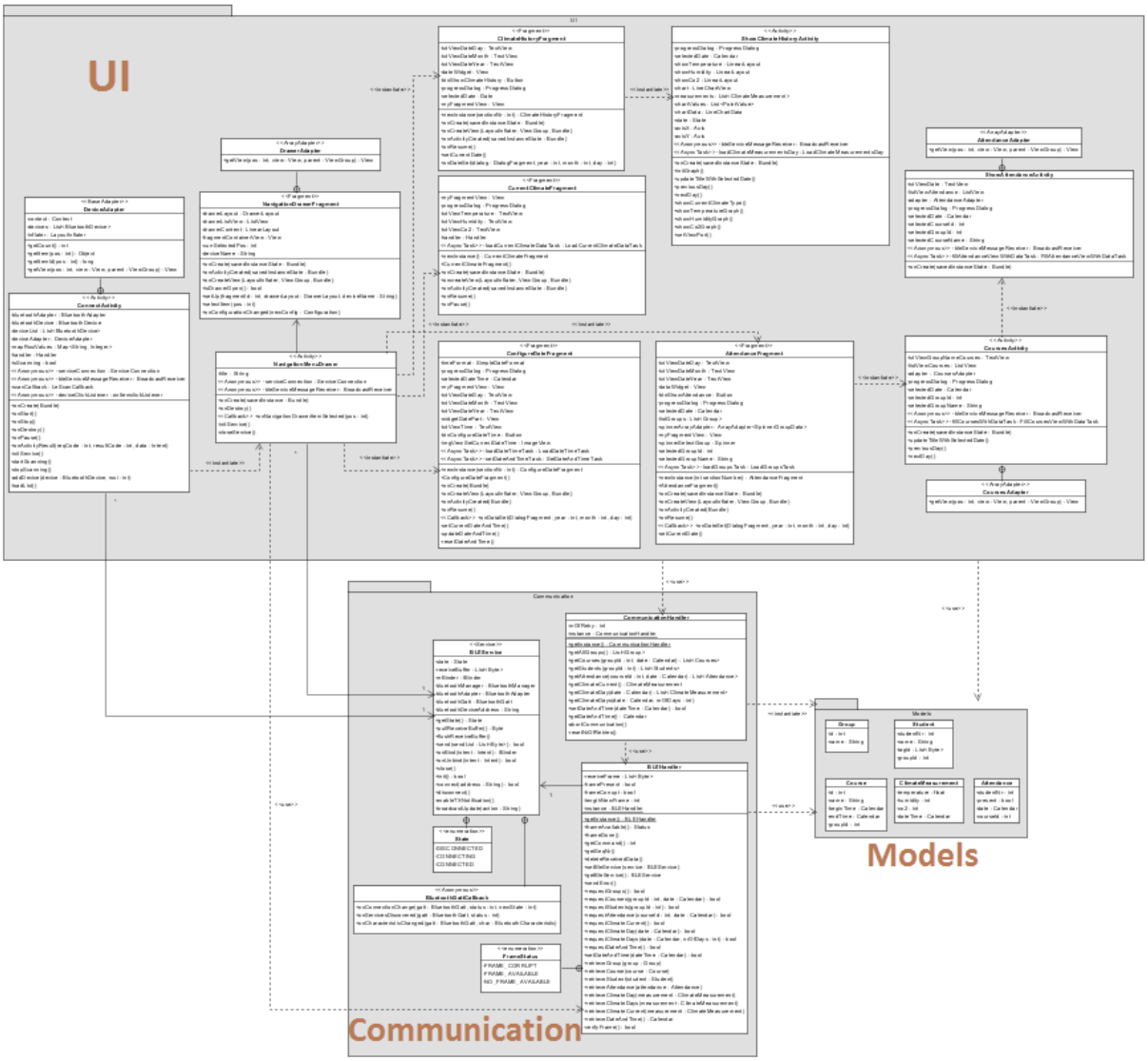
Het diagram is gemaakt aan de hand van de gespecificeerde systeemeisen en een eigen invulling van de afstudeerder. Daarbij is rekening gehouden met ontwerppatronen die worden voorgeschreven door de Android design specificatie. Zo is er voor gekozen om als menu een *navigation drawer* te gebruiken, dat tegenwoordig veel gebruikt wordt als een standaard menu in Android applicaties. Dat is een menu dat tevoorschijn komt aan de linkerkant van het scherm als de gebruiker daar met zijn vinger over veegt. De Android design specificatie schrijft een aantal richtlijnen voor bij het gebruik van de *navigation drawer*. Zo moet het menu alleen zijn op te roepen vanuit de hoger gelegen schermen en niet vanuit de dieper gelegen schermen. [40] Bij het maken van de applicatie is rekening gehouden met deze richtlijnen, zodat de applicatie vertrouwd aanvoelt bij de gebruiker, omdat het voldoet aan de standaard.



Figuur 32: Flowdiagram Android applicatie

10.7 Ontwerp van de Android applicatie

Nadat het flowdiagram van de Android applicatie heeft gezorgd voor een afstemming met de opdrachtgever over de indeling van de schermen, is er ook voor de Android applicatie een ontwerp gemaakt door middel van UML-diagrammen. De procedure hoe het ontwerp tot stand is gekomen is eerder in dit verslag al behandeld en is ook van toepassing op het ontwerp voor de Android applicatie. Deze procedure zal hier dan ook niet opnieuw toegelicht worden. Wel zijn er een aantal andere keuzes gemaakt bij het maken van het ontwerp, in vergelijking met de eerder gemaakte ontwerpen voor de andere subsystemen. Deze keuzes zullen hier worden toegelicht en er zal een indruk worden gegeven van het gemaakte ontwerp. Het gehele ontwerp is ook nu weer terug te vinden in het ontwerprapport.



Figuur 33: Klassendiagram Android applicatie

Het ontwerp van de software voor de Android applicatie is niet zo uitbreidbaar opgezet als het ontwerp voor de software van het basisstation en de meetstations. De applicatie is namelijk bedoeld als visuele representatie van het prototype. Indien in de toekomst verder wordt gewerkt aan het prototype, zal de Android applicatie zijn data niet meer van het basisstation halen, maar van een centrale server. De hoeveelheid data om weer te geven wordt daarmee ook veel groter, waardoor er een hele andere opzet van de applicatie benodigd is. Zo worden de verschillende groepen met leerlingen op dit moment in een lijst getoond, maar dat zal bij een grote hoeveelheid data niet meer

werken, zodat er iets van een zoekmogelijkheid moet komen. Hiervoor is het eerst nodig een heel datamodel uit te denken voor de opslag op de server. Dat valt echter buiten de scope van dit project. Daarom is voor dit ontwerp minder aandacht geschonken aan het gebruik van interfaces of abstracte klassen.

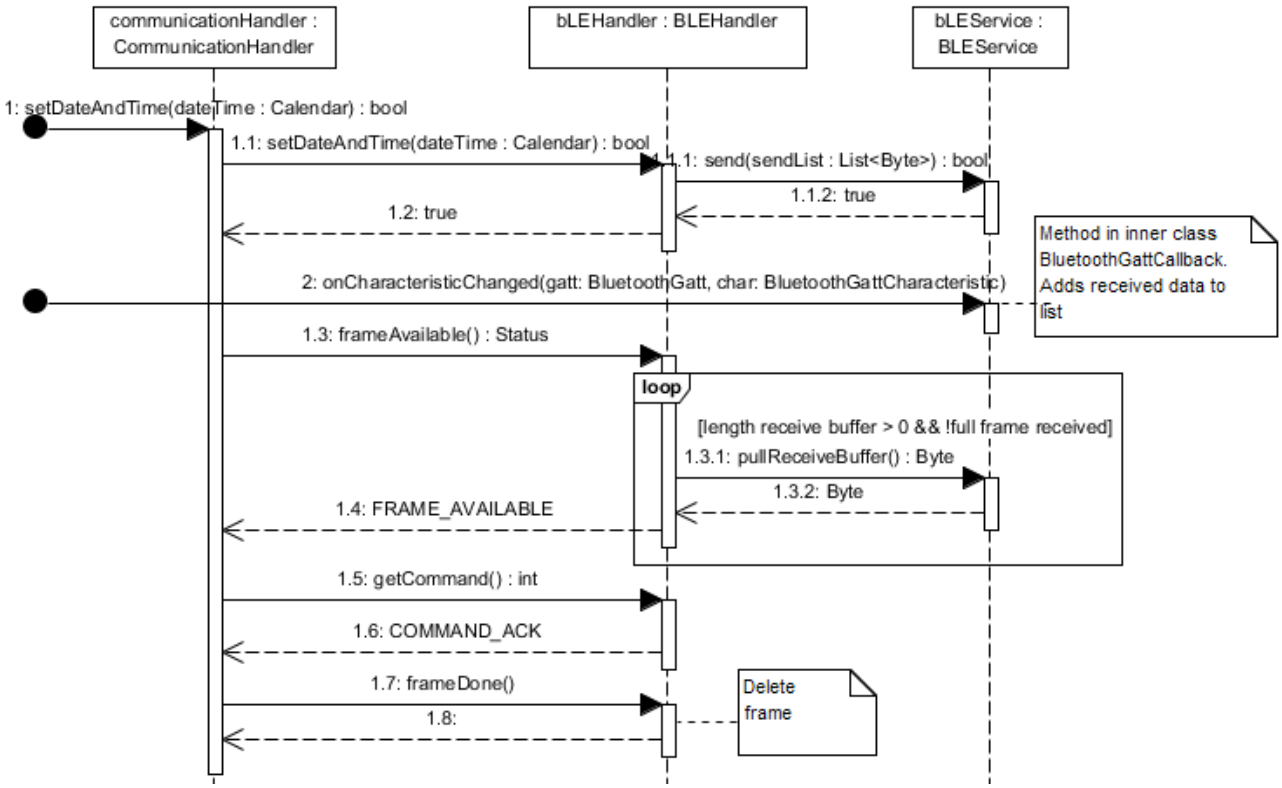
Het UML klassendiagram van de Android applicatie is te zien in Figuur 33. De applicatie is daarbij opgedeeld in drie packages: De *Ui* package, de *Communication* package en de *Models* package. In de *ui* package bevinden zich de klassen die subklassen zijn van een *Activity* of *Fragment* en zorgen voor het presenteren van de data op het scherm. Deze klassen gebruiken de klassen uit de *Communication* package om te communiceren met het basisstation via Bluetooth Smart. De gegevens die hiermee worden verkregen worden teruggegeven in de vorm van een lijst met objecten uit de model package.

Bij het maken van een applicatie voor Android wordt er veel gebruik gemaakt van het Android framework. Zo bestaat elk scherm uit een subklasse van de *Activity* klasse of de *Fragment* klasse. Indien al deze overervingen van klassen uit het Android framework zou worden weergegeven in het klassendiagram, wordt het diagram erg onoverzichtelijk. Daarom is er voor gekozen om stereotypes te gebruiken om aan te geven om wat voor soort klasse het gaat. Zo is bij een aantal klassen het stereotype *<<Activity>>* of *<<Service>>* te zien, wat in werkelijkheid dus subklassen zijn van de *Activity* of de *Service* klasse uit het Android framework. Daarnaast bestaat een Android applicatie doorgaans veel uit inner klassen en uit anonymous inner klassen. Laatstgenoemde is het creëren van een object, waarbij meteen method overloading plaatsvindt, zonder dat er eerst een subklasse hoeft te worden gedefinieerd. Dit wordt bijvoorbeeld veel gebruikt bij event listeners, om zo te reageren op het indrukken van een knop op het scherm. Er is geen officiële manier binnen UML om dit soort objecten weer te geven. Er is gekozen om dit soort objecten weer te geven als een member variabele van een klasse, met het *<<Anonymous>>* stereotype. Door het gebruik van stereotypes blijft het diagram overzichtelijk. Het diagram is nu niet volgens de officiële manier van UML gespecificeerd, maar in dit geval is de duidelijkheid en leesbaarheid van het diagram verkozen boven de officiële regels.

10.8 Communicatie met het basisstation

Figuur 34 toont het verloop bij het configureren van de datum en tijd op het basisstation. Er is te zien hoe de methode setDateAndTime() vanuit een andere thread wordt aangeroepen dan de methode onCharacteristicChanged(). Laatstgenoemde is een event, dat optreedt in de *BLEService* op het moment dat er nieuwe data binnenkomt vanuit het basisstation. Deze data wordt in een ontvangstlijst gezet. Vanuit de methode setDateAndTime() wordt vervolgens de frameAvailable() methode aangeroepen om te achterhalen of er al een volledig frame is binnengekomen. Hiervoor wordt de data uit de ontvangstlijst gehaald.

Het leek er echter op dat niet alle data correct werd ontvangen. Bij het uitlezen van de ontvangstbuffer bleken er vaak bytes te missen. In eerste instantie was het onbekend waar de fout kon zitten. Er is onderzocht of de data wel correct werd verstuurd vanuit het basisstation en dat bleek het geval. Vervolgens is gekeken of de datapakketten die binnen komen op de Android applicatie overeenkomen met de pakketten die zijn verstuurd. Ook dit bleek het geval, dus kon er geconcludeerd worden dat er tijdens de communicatie niets mis ging. De fout moest dus zitten in het gedeelte van de code waar de data in de ontvangstlijst wordt gezet of juist in het gedeelte van de code waar de data uit de ontvangstlijst wordt gehaald. Op deze manier is achterhaald dat het probleem bleek voort te komen uit een race condition. De ontvangstbuffer wordt namelijk gelijktijdig benaderd vanuit twee verschillende threads. Hierdoor is de uitkomst van de bewerking op de data ongedefinieerd en zo kwam het dat er data uit de ontvangstbuffer verdween.



Figuur 34: Configureren van de datum en tijd op het basisstation

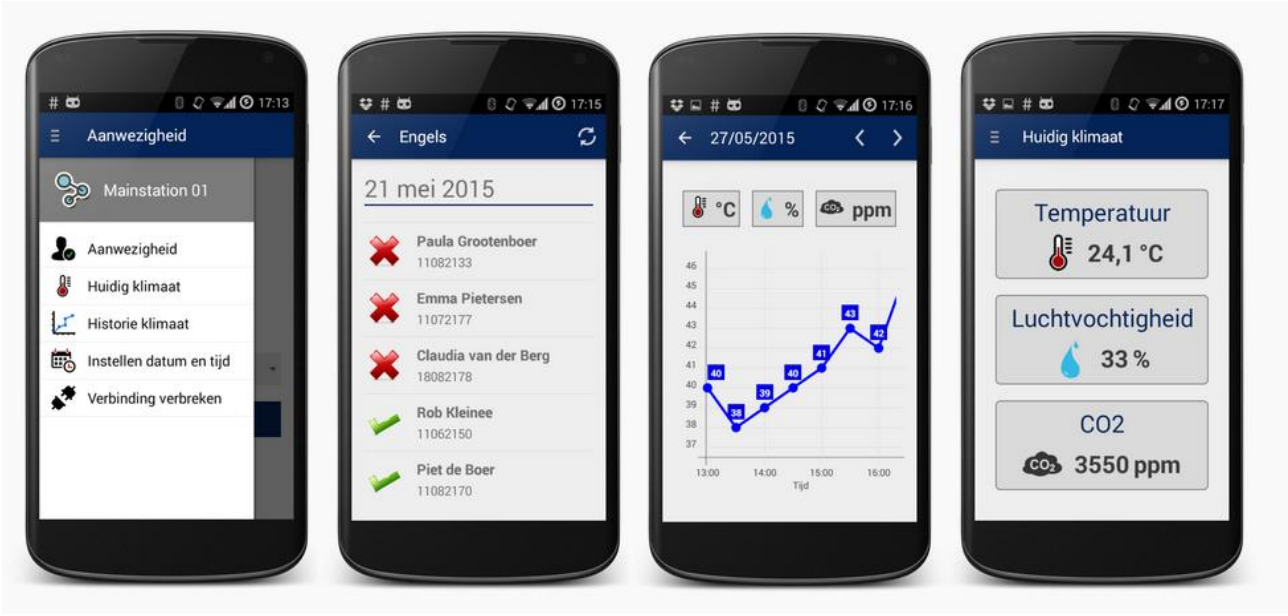
Om de race condition te voorkomen is er een mutex gebruikt in de vorm van het *synchronized* keyword in Java. Hierdoor wordt afgedwongen dat enkel één thread tegelijkertijd de ontvangstbuffer kan benaderen. Hiermee was het probleem verholpen en werd alle data correct ontvangen.

```
synchronized(receiveBuffer) {
    receiveBuffer.add(b);
}
```

Figuur 35: Code - Gebruik van het synchronized keyword

10.9 Screenshots van de Android applicatie

Om een indruk te geven van de Android applicatie is in Figuur 36 een aantal screenshots te zien van de applicatie. Dit is echter maar een gedeelte van alle schermen.



Figuur 36: Screenshots van de Android applicatie

10.10 Evaluatie construction fase iteratie 3

Achteraf gezien was het een goede keuze om meer tijd in te plannen voor deze iteratie. De vijf weken die uiteindelijk ingepland waren bleken hard nodig. Het basisstation is uitgebreid met de mogelijkheid om te communiceren via Bluetooth Smart en daarbij te dienen als peripheral. Vervolgens is er een Android applicatie ontwikkeld waarmee via Bluetooth Smart een verbinding met het basisstation kan worden gemaakt. Door middel van de Android applicatie wordt de data gepresenteerd met betrekking tot de aanwezigheid van studenten en de gemeten klimaatwaardes in het lokaal.

Zelfs nu er vijf weken waren ingepland voor deze iteratie, is het niet gelukt om alle systeemeisen aan de Android applicatie te implementeren. Drie van de eisen waren dat de historie van de klimaatwaardes in het lokaal getoond moet kunnen worden per dag, per week en per maand. Enkel eerstgenoemde is geïmplementeerd. Aan de andere twee eisen waren een lagere prioriteit toegekend en er bleek geen tijd meer te zijn om deze eisen te implementeren. Alle systeemeisen met een hogere prioriteit zijn echter wel geïmplementeerd, waardoor de applicatie wel volledig functioneel is.

11 Transition fase

De transition fase van het project bestond uit één korte iteratie van anderhalve week. De nadruk in deze fase lag op het opnieuw testen van het gehele systeem en het opleveren van het systeem. De planning voor deze fase is te zien in Figuur 37. Dit hoofdstuk beschrijft het proces van deze fase.

Planning Transition fase	Week 16					Week 17				
Activiteit	1	2	3	4	5	6	7	8	9	10
Testen van het gehele systeem										
Aanpassingen aan het systeem naar aanleiding van de tests										
Opleveren van het systeem										
Afronden afstudeerverslag										

Figuur 37: Planning transition fase

11.1 Het testen van de software

Het testen van de geschreven software heeft plaats gevonden gedurende het gehele project. Hoe het testen heeft plaatsgevonden wordt echter pas in dit hoofdstuk beschreven, zodat er een eenduidige plaats is waar deze informatie kan worden gevonden in dit verslag.

11.1.1 Het testplan

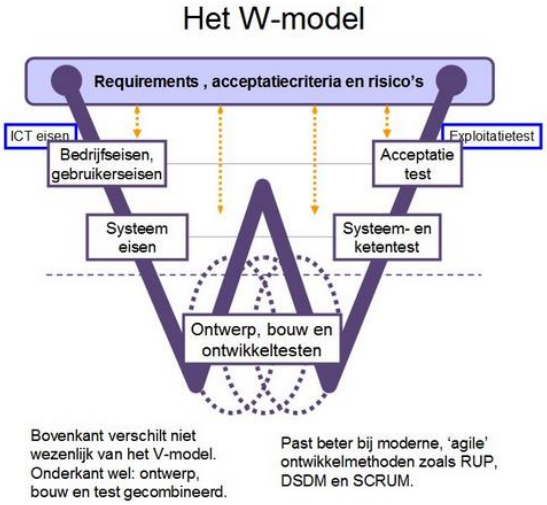
In de eerste iteratie van de construction fase is een testplan geschreven. Dit testplan maakt duidelijk op welke manier het systeem getest gaat worden. Daarin wordt de teststrategie beschreven met de daaruit voortvloeiende testsoorten en testtechnieken. Het testplan is terug te vinden in het test document, te vinden in bijlage F.

Als teststrategie is er een lichte versie van SmarTEST gebruikt. SmarTEST is een Nederlandse methode voor het testen van informatiesystemen. Deze teststrategie is gericht op agile ontwikkelingsmethodieken en zet zichzelf neer als een teststrategie voor RUP en SCRUM. [41] SmarTEST kan daarom goed gebruikt worden in combinatie met Open Up. Daarnaast was er door de afstudeerder al enige kennis opgedaan van deze testmethode gedurende de opleiding.

Eerst is er met behulp van het boek *SmarTEST, Slim testen van Informatiesystemen* [41] kennis vergaard over de testmethode. Vervolgens is een strategie uitgezet voor het testen van het systeem. SmarTEST specificeert een grote verscheidenheid aan testsoorten en testtechnieken, die bij moeten dragen aan een betere kwaliteit van de software. Echter niet alle technieken die staan gespecificeerd, moeten ook altijd toegepast worden. Het verschilt per project welke testtechnieken het meest geschikt zijn voor het beoordelen van de kwaliteit. Omdat dit een relatief klein project is en het systeem een prototype betreft, is er voor gekozen een kleine set van testsoorten en testtechnieken te gebruiken. Hiervoor is eerst onderzocht welke testsoorten en testtechnieken er allemaal zijn. De testsoorten zijn onder te verdelen in de ontwikkeltests, integratietests, systeemtests en de acceptatietests. Een testsoort kan uitgevoerd worden door middel van meerdere testtechnieken, variërend van inspecties tot een load-test. Al deze testtechnieken kunnen worden opgedeeld naar de volgende specificaties: blackbox testen, whitebox testen, glassbox testen, statisch testen en dynamisch testen. Een beschrijving van deze testsoorten en technieken is terug te vinden in het test document.

Veel testmethodes maken gebruik van het bekende *V-model*, dat oorspronkelijk is ontwikkeld voor de waterval ontwikkelingsmethodiek. SmarTest heeft dit model uitgebreid naar het *W-model*, waarin de ontwikkeling en de ontwikkeltesten in elkaar overlopen en niet meer als aparte fasen worden gezien. Hierdoor past het model beter bij een iteratieve aanpak. Aan het eind van een iteratie is het gebruikelijk systeemtests uit te voeren. Dit is verduidelijkt in Figuur 38.

Er is voor gekozen om bij het testen van het te maken systeem gebruik te maken van alle testsoorten. Tijdens de ontwikkeling van de software zijn er steeds ontwikkeltests en integratietests uitgevoerd, om zo de geschreven softwarecomponenten te testen, afzonderlijk en in combinatie met andere softwarecomponenten. Deze tests zijn vaak uitgevoerd met behulp van een debugger, om zo te testen hoe de code reageert op verschillende invoer en omstandigheden. Deze tests zijn echter niet beschreven in het testdocument. Daarnaast zijn er systeemtests en duurtests uitgevoerd, evenals een gebruikersacceptatietest. Deze tests zijn wel beschreven in het testdocument.



Figuur 38: Het W-model [41]

11.1.2 Systeemtesten gedurende het project

Aan het eind van iedere iteratie in de construction fase zijn er systeemtests uitgevoerd, waarmee is getest of het systeem voldoet aan de gespecificeerde eisen. Deze tests zijn uitgevoerd als whitebox, glassbox of blackbox testen. Voor elke test is een testcase beschreven, waarin staat beschreven welke eis wordt geverifieerd, wat de te volgen teststappen zijn en wat het verwachte resultaat is. Een van de testcases is te zien in Tabel 17. De overige testcases zijn te vinden in het test document.

Testcase TF4.0	**Koppelen van meetstations aan basisstation**
Beschrijving	De meetstations moeten direct of indirect (via een ander meetstation) kunnen worden gekoppeld aan het basisstation.
Referentie specificatie(s)	[D1] : F1.10
Voorwaarden	• In bezit van een basisstation en drie meetstations.
Teststappen	
1	Zet alle stations aan.
2	Laat het basisstation zoeken door op BUTTON 2 te drukken en laat meetstation 1 verbinden door op BUTTON 1 te drukken.
3	Laat het basisstation nogmaals zoeken door op BUTTON 2 te drukken en laat meetstation 2 verbinden door op BUTTON 1 te drukken.
4	Laat meetstation 1 zoeken door op BUTTON 2 te drukken en laat meetstation 3 verbinden door op BUTTON 1 te drukken.
Verwacht resultaat	
Tijdens het zoeken knippert LED 2 op het station dat zoekt en tijdens het verbinden knippert LED 1 op het station dat verbinding maakt. Binnen enkele seconden stopt het knipperen en blijven deze leds permanent branden, wat aangeeft dat de stations zijn gekoppeld. Eindresultaat: meetstations 1 en 2 zijn direct verbonden met het basisstation en meetstation 3 is verbonden aan meetstation 1.	

Tabel 17: Voorbeeld testcase systeemtest

Iedere iteratie zijn er testcases geschreven voor het subsysteem dat werd ontwikkeld in die iteratie, zodat er iedere iteratie meer testcases bijkwamen. Een daaropvolgende iteratie werden de oude testcases opnieuw uitgevoerd, om zo vast te stellen dat die functionaliteiten ook nog steeds werkten. De resultaten van de testcases zijn vervolgens iedere iteratie vastgelegd in het testdocument, met eventuele opmerkingen. De resultaten van een drietal testcases zijn te zien in Tabel 18.

Testcase	Resultaat	Opmerkingen
TF1.0	X	Het kabeltje tussen de VDD en de SD-kaart bleek brak te zijn, waardoor de communicatie met de SD-kaart af en toe wegviel. Verholpen met een nieuw kabeltje.
TF2.0	V	-
TF3.0	V	-

Tabel 18: Een aantal resultaten van de systeemtests

11.1.3 Duurtesten

Om te testen of het systeem ook op de langere termijn correct blijft werken, zijn er duurtests uitgevoerd. Voor deze tests is dezelfde aanpak gehanteerd als voor de systeemtests. Er zijn weer testcases opgesteld, de testcases zijn uitgevoerd en de resultaten zijn weer vastgelegd in het test document. Een voorbeeld van een test is een test waarbij een basisstation werd gekoppeld aan twee meetstations. Vervolgens is getest of de stations na twee dagen nog steeds gekoppeld waren en of gedurende die twee dagen elk halfuur de klimaatmetingen zijn opgeslagen.

11.1.4 Gebruikersacceptatietest

In de transition fase van het project is een gebruikersacceptatietest uitgevoerd. Met een gebruikersacceptatietest test de opdrachtgever of het volledige product voldoet aan de gespecificeerde eisen en verwachtingen.

Een aantal van de eerder gespecificeerde systeemtests zijn blackbox-tests. Deze testcases zijn opnieuw gebruikt voor de gebruikersacceptatietest, maar nu zijn de testcases door de opdrachtgever zelf uitgevoerd. De resultaten en opmerkingen zijn vervolgens gedocumenteerd in het test document. Tot slot heeft de opdrachtgever het product zelf vrij getest. Ook de opmerkingen die hieruit naar voren kwamen zijn gedocumenteerd.

Uit de gebruikerstest kwamen geen grote problemen naar voren, enkel een aantal opmerkingen met betrekking tot de layout van de Android applicatie. Deze opmerkingen zijn gebruikt om de layout iets aan te passen. Het product voldeed verder aan de verwachtingen van de opdrachtgever.

11.2 Overdragen van het product aan de opdrachtgever

Aan het eind van het project is het systeem overgedragen aan de opdrachtgever. Helaas is het door tijdsgebrek niet mogelijk geweest om alle gespecificeerde systeemeisen te implementeren. Daarom is er in het test document een hoofdstuk opgenomen waarin beschreven staat welke systeemeisen wel zijn doorgevoerd in het systeem en welke niet. Hierdoor is het voor de opdrachtgever wel duidelijk wat het huidige systeem allemaal wel en niet kan.

Zoals eerder in het verslag is beschreven zijn alle eisen aan het systeem ingedeeld naar prioriteit, aan de hand van de MoSCoW-methode. De prioriteit *must have* betekent dat zonder deze eis het systeem niet kan werken. Voor dit systeem zijn bijna alle eisen met de prioriteit *must have* verwerkt, enkel de eisen voor de desktop applicatie niet, aangezien er geen tijd meer was om een desktopapplicatie te ontwikkelen. Als alternatief hierop is bedacht om de data op het basisstation in te laden door middel van een SD-kaart met komma gescheiden bestanden. Dit is eerder in het verslag uitgebreider beschreven. Hierdoor is de desktop applicatie niet per se benodigd, aangezien de data nu ook manueel kan worden aangepast, wat natuurlijk wel minder gebruiksvriendelijk is. Door dit alternatief is het systeem wel volledig werkbaar. Daarnaast zijn een drietal eisen met een lagere prioriteit niet geïmplementeerd. Deze eisen zijn te zien in Tabel 19. Tot slot zijn de eisen met de *won't have* prioriteit niet geïmplementeerd, maar deze eisen lagen dan ook buiten de scope van het project.

Nr	Beschrijving eis	Prioriteit	Gehaald
F1.11	De huidige klimaatwaardes (F1.5, F1.6, F1.7) moeten worden getoond op een scherm aan de muur.	Could	X
F2.4	Door middel van de mobiele applicatie moet de historie van de klimaatwaardes gedurende een opgegeven week binnen een lokaal kunnen worden weergegeven.	Should	X
F2.5	Door middel van de mobiele applicatie moet de historie van de klimaatwaardes gedurende een opgegeven maand binnen een lokaal kunnen worden weergegeven.	Should	X

Tabel 19: Niet geïmplementeerde eisen aan het systeem

Het plan was om alle geschreven code te voorzien van commentaar in Doxygen-formaat. Door middel van dit commentaar wordt van iedere functie beschreven wat de exacte in- en uitvoer is en wat de gebruiker van de functie kan verwachten. Hiermee wordt het voor een andere programmeur gemakkelijker om verder te werken aan de software. Delen van de geschreven software zijn voorzien van dit commentaar, echter was het door tijdsgebrek niet meer haalbaar om alle klassen en methodes van dit commentaar in Doxygen-formaat te voorzien. De code in de functies zelf is wel steeds van commentaar voorzien, om zo aan te geven hoe bepaalde procedures en algoritmes werken. Verder is het gehele ontwerp van alle geschreven software gedetailleerd beschreven in het ontwerp document, wat een andere programmeur van genoeg houvast voorziet om verder te kunnen werken aan het systeem.

Tot slot is alle geschreven documentatie en alle geschreven source-code overgedragen aan de opdrachtgever.

12 Conclusie en aanbevelingen

12.1 Conclusie

Bij aanvang van het project was de volgende doelstelling geformuleerd:

"De doelstelling van de opdracht is om een prototype/demonstratiemodel van een systeem te ontwikkelen, waarmee leerlingen zich kunnen aan- en afmelden en waarmee de aanwezigheid van de leerlingen en de kwaliteit van het klimaat binnen het klaslokaal inzichtelijk worden gemaakt voor een docent. Met dit demonstratiemodel wordt er inzichtelijk gemaakt wat de mogelijkheden zijn en het demonstratiemodel kan worden gebruikt om de interesse te wekken bij scholen, om zo het toekomstige product te vermarkten."

Er kan worden geconcludeerd dat deze doelstelling grotendeels is behaald. Het prototype dat is gerealiseerd gedurende het project geeft een goed beeld van de mogelijkheden van een systeem voor leerling- en klimaatregistratie. Zoals eerder is besproken is het door tijdsgebrek niet gelukt om alle systeemeisen te implementeren, echter is aan de meeste systeemeisen voldaan, wat tot een werkend product heeft geleid.

12.2 Aanbevelingen

Het systeem is gerealiseerd als een prototype. Dit prototype zou nog aangevuld kunnen worden met functionaliteiten waar door tijdsgebrek niet meer aan toe is gekomen. De volgende uitbreidingen kunnen nog worden toegepast:

- Op dit moment worden de CO2-waardes niet echt gemeten, maar worden de waardes gesimuleerd. Er zou alsnog een echte CO2-sensor gebruikt kunnen worden.
- Er zou een kastje gemaakt kunnen worden om de hardware in te stoppen, zodat het prototype mooier oogt.
- De Android applicatie kan nog worden uitgebreid met de twee systeemeisen waar geen tijd meer voor was, zodat alsnog de klimaatwaardes kunnen worden bekeken gedurende een week of een maand. Op dit moment kunnen de waardes enkel bekeken worden gedurende een dag.
- Er zou alsnog een desktopapplicatie ontwikkeld kunnen worden voor het configureren van demonstratiedata van lesblokken en leerlingen, zodat het niet manueel geconfigureerd hoeft te worden door middel van komma gescheiden bestanden.

Indien blijkt dat er interesse is in het systeem, zal het prototype fors uitgebreid moeten worden en zal het prototype op sommige punten gewijzigd moeten worden:

- Optimaliseren van het stroomverbruik van de meetstations, zodat deze stations langere tijd kunnen werken op een batterijtje.
- Het basisstation zal moeten communiceren met een centrale server. Op de server wordt dan alle data van de aanwezigheid van studenten en de gemeten klimaatwaardes opgeslagen, zodat deze data niet meer lokaal op het basisstation wordt opgeslagen.
- De server zal alle informatie moeten bevatten met betrekking tot de studenten, gekoppeld aan een RFID-tag, en de roosters van de groepen met studenten.
- Het integreren van de software op de server met bestaande administratieve systemen binnen scholen.
- Het ontwikkelen van een smartphone applicatie waarmee de data niet meer van een basisstation wordt gehaald, maar rechtstreeks van de server. Er moet worden nagedacht over het duidelijk presenteren van grote hoeveelheden data.
- Het gebruik van een RFID-reader op langere afstand kan worden onderzocht, zodat studenten niet meer handmatig hun RFID-tag hoeven te laten scannen.

13 Projectevaluatie

In dit hoofdstuk wordt het project als geheel geëvalueerd. Hierbij is het hoofdstuk opgedeeld in de productevaluatie, de procesevaluatie en de evaluatie van de aan te tonen competenties.

13.1 Productevaluatie

Gedurende het project is er een prototype gerealiseerd van een leerlingen- en klimaatregistratiesysteem voor scholen. Dit prototype zou bestaan uit een basisstation, meetstations, een smartphone applicatie en een desktopapplicatie. Door tijdgebrek is laatstgenoemde helaas komen te vervallen. Als alternatief kan de versimpelde roosterdata nu in het basisstation worden gezet door middel van komma gescheiden bestanden. Dit is wel een stuk minder gebruiksvriendelijk dan een desktopapplicatie, echter was dit gezien het tijdsbestek wel een goed alternatief. Verder zijn alle systeemeisen met een *must-have* prioriteit wel geïmplementeerd. Een aantal eerder benoemde systeemeisen met een lagere prioriteit is komen te vervallen.

Er moet benadrukt worden dat het gerealiseerde systeem wel echt een prototype is. Er wordt nu nog gewerkt met een versimpelde dataset, waarmee wordt aangegeven welke lesblokken met welke studenten er zijn. Ook wordt deze data, samen met de gelogde data van de aanwezigheidsregistratie en de klimaatregistratie, opgeslagen op het basisstation en niet op een centrale server. Het systeem moet fors worden uitgebreid voordat het daadwerkelijk in gebruik kan worden genomen op een school. Het geeft echter wel een goede indicatie van de mogelijkheden van het systeem, zodat hiermee interesse kan worden gewekt bij scholen.

De software voor alle subsystemen is herhaaldelijk getest door middel van verschillende testsoorten. Hiermee is er voor gezorgd dat het systeem stabiel draait en doet wat het moet doen volgens de opgestelde specificaties. De software voor het basisstation en de meetstations is uitbreidbaar en onderhoudbaar opgezet. Zo staat de architectuur van de software bijvoorbeeld open voor uitbreidingen met een andere RFID-reader of andere sensoren. Daarnaast is er rekening gehouden met uitbreidingen met betrekking tot het gebruik van een webserver. De software is daarbij opgedeeld in verschillende packages, waarbij sommige losstaande packages ook gebruikt kunnen worden in andere projecten. Er is bewust voor gekozen minder tijd te besteden aan het maken van een uitbreidbaar ontwerp voor de Android applicatie, omdat de applicatie toch grotendeels veranderd moet worden indien er grote hoeveelheden data getoond gaan worden. De applicatie is echter wel een goede visuele presentatie van het prototype. Door middel van een duidelijke interface kan er inzicht worden verkregen in de aanwezigheids- en klimaatregistratie.

Gedurende het project zijn er ook meerdere documenten gemaakt voor het bedrijf. Het plan van aanpak gaf een goede basis voor de start van het project. In het requirements document zijn overzichtelijk de eisen aan het systeem gespecificeerd, zodat er een overeenstemming met de opdrachtgever kon worden bereikt. Het analysedocument biedt een bron van informatie voor de belangrijkste gemaakte keuzes op het gebied van de gebruikte technieken en protocollen. Het verschaft de benodigde kennis waarom deze technieken en protocollen worden gebruikt en hoe deze technieken en protocollen werken. Het ontwerp document biedt een goed uitgangspunt voor een andere programmeur om verder te werken aan het systeem. Het ontwerp van het systeem is uitvoerig beschreven, zodat een volgende programmeur weet hoe de software aangepast en uitgebreid kan worden. Tot slot maakt het test document inzichtelijk hoe het systeem is getest en wordt aangetoond dat het systeem voldoet aan de gespecificeerde eisen. Ook is aangegeven welke systeemeisen wel en welke niet zijn gerealiseerd.

Door tijdsgebrek is helaas niet alle geschreven code meer van commentaar voorzien in Doxygen-formaat. Hierdoor is niet de exacte omschrijving van elke geschreven klasse en methode aanwezig,

wat wel erg handig is voor een andere programmeur bij het uitbreiden van de software. Dit wordt echter wel deels opgevangen door de beschrijving van het ontwerp van de software in het ontwerprapport en door het commentaar dat te vinden is in de code zelf.

13.2 Procesevaluatie

Gedurende het project is de Open Up methodiek gebruikt om structuur te geven aan de uitvoering van de opdracht. Er is gekozen om iets af te wijken van de oorspronkelijke aanpak van Open Up, door in de elaboration fase niet al het grootste gedeelte van het ontwerp af te hebben, wat wel gebruikelijk is. Deze aanpak zorgde voor een goede structuur en houvast bij de uitvoering van het project. Door het iteratieve karakter konden de subsystemen ieder in een eigen iteratie in de construction fase ontwikkeld worden, waarbij steeds de verschillende disciplines opnieuw doorlopen konden worden. De eisen werden verfijnd en er werd een ontwerp gemaakt voor het betreffende subsysteem. Daar voorafgaand is het grootste gedeelte van de analyses uitgevoerd in de elaboration fase, zodat de grootste projectrisico's vroeg werden aangepakt en bekend was welke technieken en protocollen er gebruikt zouden worden. Daarbij bood Open Up houvast bij het maken van verschillende documenten. Achteraf gezien was het een juiste keus om de Open Up methodiek op deze manier te gebruiken en bleek de methodiek goed te passen bij het karakter van dit project.

Zoals al eerder is aangegeven is het niet gelukt om alle systeemeisen te implementeren binnen de gestelde tijd, er is zelfs een heel subsysteem komen te vervallen, namelijk de desktopapplicatie met als doel het configureren van demonstratiedata van de lesblokken en studenten. Bij aanvang van het project was al duidelijk dat het krap zou worden het gehele systeem te maken in de beschikbare tijd, maar na het maken van de globale planning werd gedacht dat het wel haalbaar kon zijn. Het grootste gedeelte van het project verliep dan ook volgens planning en sommige iteraties waren zelfs eerder afgerond dan verwacht. Aan het begin van iedere iteratie is er een gedetailleerdere planning gemaakt voor de betreffende iteratie, om zo beter grip te krijgen op de beschikbare tijd. Dit bleek goed te werken, aangezien hiermee de prioriteiten aan de juiste werkzaamheden toegekend kon worden. Echter heeft de afstudeerder zich aan het begin van het project verkeken op de hoeveelheid werk voor de Android applicatie. Gedurende het project werd duidelijk dat er meer tijd benodigd was voor de Android applicatie dan de drie geplande weken. Hierdoor was er geen tijd meer voor de desktopapplicatie.

Het zou beter zijn geweest als er bij aanvang van het project al uitvoeriger was nagedacht over een eventueel back-up plan indien het project niet haalbaar zou zijn binnen de beschikbare tijd. Nu moest halverwege het project worden nagedacht over een alternatief voor de desktopapplicatie, terwijl dit ook bij aanvang al uitgedacht had kunnen worden. Dit is uiteindelijk wel recht gezet door de keuze om de demonstratiedata op het basisstation in te laden door middel van komma gescheiden bestanden op een SD-kaart, zodat deze data ook manueel kan worden geconfigureerd. Deze keus is gemaakt met de gedachte dat er misschien niet genoeg tijd zou zijn voor de desktopapplicatie. Daarnaast is de opdrachtgever tijdig op de hoogte gesteld van het inzicht dat er niet genoeg tijd zou zijn en is daarbij het alternatief aangedragen. Ook doordat steeds prioriteiten aan de systeemeisen zijn toegekend, was het duidelijk welke systeemeisen eerst geïmplementeerd moesten worden en welke later. Hierdoor zijn een aantal vervallen systeemeisen uiteindelijk de eisen met een lagere prioriteit.

13.3 Competenties

Gedurende de opleiding is er gewerkt aan meerdere competenties die bij het vakgebied van de Technische Informatica horen. Bij aanvang van dit project zijn er vijf competenties gekozen die in het bijzonder aangetoond werden bij de uitvoering van de opdracht. In deze paragraaf wordt aangetoond dat de gekozen competenties op een juiste manier zijn uitgevoerd.

13.3.1 A4 - Kiezen van een ontwikkelstrategie en ontwikkelmethodiek

Bij aanvang van het project is een ontwikkelmethodiek gekozen die gehanteerd zou worden bij de uitvoering van de opdracht. Hiervoor zijn er eerst eisen opgesteld aan de methodiek, aan de hand van de kenmerken van dit specifieke project. Zo werd bijvoorbeeld vastgesteld dat het project zich leent voor een iteratieve aanpak, zodat de deelsystemen waaruit het systeem bestaat iteratief ontwikkeld kunnen worden en dat de methodiek open moet staan voor het uitvoeren van analyse. Daarnaast is bij het opstellen van de eisen ook rekening gehouden met verwachtingen vanuit het bedrijf en de studie.

Nadat de eisen aan de methodiek helder waren zijn er meerdere methodieken geanalyseerd. Aan de hand van een tabel is duidelijk gemaakt in hoeverre elke onderzochte methodiek voldoet aan de gestelde eisen. Vervolgens zijn de voor- en nadelen van elke methodiek tegen elkaar afgewogen en is er een gemotiveerde keuze gemaakt voor de Open Up methodiek.

Er is voor gekozen iets af te wijken van de oorspronkelijke aanpak van de Open Up methodiek. In de elaboration fase is niet het grootste gedeelte van de architectuur ontworpen, zoals dat wel gebruikelijk is, maar het ontwerp is gedurende de iteraties in de construction fase tot stand gekomen. De motivatie hiervoor was dat het ontwerp van elk subsysteem een andere insteek zou hebben en hiervoor andere kennis benodigd zou zijn. Zo kon elk subsysteem in een andere iteratie ontwikkeld worden. Hiermee is de methodiek dus aangepast naar de kenmerken van het project.

13.3.2 A5 - Opstellen van systeemeisen

In het begin van het project is een groot gedeelte van de eisen aan het gehele systeem achterhaald. In dit stadium waren nog niet alle eisen even duidelijk en werden sommige eisen globaal opgesteld. In latere iteraties zijn de eisen aan elk subsysteem steeds herzien en gedetailleerd. Ook zijn er een aantal eisen opgesteld die waarschijnlijk niet meer haalbaar zouden zijn binnen dit project, maar zo wel kunnen dienen tot een beter begrip van waar het systeem naar toe moet groeien in de toekomst.

De eisen aan het systeem zijn opgesteld aan de hand van gesprekken met de opdrachtgever. Hierbij zijn ook vragen opgesteld en voorgelegd, en is er meegedacht met mogelijke eisen aan het systeem. Eerst zijn er use-cases met bijbehorende scenario's uitgewerkt, om zo de functionele eisen vast te leggen uit het oogpunt van de gebruiker. Nadat deze geverifieerd waren door de opdrachtgever zijn de use-cases vertaald naar de functionele eisen van het systeem. Daarnaast zijn er ook niet-functionele eisen opgesteld, opgedeeld volgens het FURPS+ model. Alle eisen zijn geprioriteerd volgens de MoSCoW-methode en gedocumenteerd in een apart requirements document. De eisen zijn daarbij zo opgeschreven dat ze verifieerbaar zijn, zodat ze later getest konden worden.

13.3.3 C8 - Ontwerpen van een technisch informatie systeem

Voorafgaand aan het schrijven van de software, is steeds eerst een ontwerp gemaakt van de software door middel van de UML-notatie. Tijdens het implementeren van de code zijn er vaak door nieuwe inzichten nog wel aanpassingen gedaan aan het ontwerp, waarbij is gezorgd voor de consistentie tussen de diagrammen en de code. Het ontwerp is iteratief tot stand gekomen, waarbij in de construction fase van het project iedere iteratie het ontwerp is gemaakt voor een nieuw deelsysteem. Daarbij is soms ook het ontwerp van een eerder subsysteem uitgebreid. Deze manier van het ontwerpen van de software paste goed bij de gekozen aanpak van het project.

Voor het vastleggen van het ontwerp is gebruik gemaakt van verschillende soorten UML-diagrammen, waarbij voornamelijk klassendiagrammen, sequentiediagrammen en statediagrammen zijn gebruikt. Hierbij is gezorgd voor consistentie tussen de verschillende diagrammen en de geschreven code. De diagrammen zijn grotendeels opgesteld volgens de officiële UML-specificaties. Echter is hier in sommige gevallen van afgeweken als dat de leesbaarheid en duidelijkheid ten goede kwam. Een voorbeeld hiervan is het gebruik van stereotypes om aan te geven dat een klasse een subklasse is uit het Android framework. Het ontwerp is overdraagbaar gemaakt door deze te documenteren in het ontwerp document.

Bij het maken van de klassendiagrammen is steeds nagedacht in hoeverre het subsysteem uitbreidbaar moet zijn. Hier is vervolgens op ingespeeld door het gebruik van abstracte klassen en interfaces op de punten waar uitbreiding of een aanpassing wordt verwacht. Ook is rekening gehouden met de koppeling en cohesie tussen packages en kunnen een aantal packages los functioneren, zodat ze ook te gebruiken zijn in andere projecten. Tot slot zijn bepaalde standaardproblemen opgelost door het gebruik van design patterns zoals een *observer* of een *singleton*.

13.3.4 D17 Testen van software systemen

Het testen van de software is uitgevoerd aan de hand van de SmarTest testmethode, die goed aansluit bij de keuze voor de Open Up methodiek bij de uitvoering van het project. Vervolgens is geanalyseerd welke testsoorten en testtechnieken SmarTest voorschrijft en is een keus gemaakt voor een aantal testsoorten en testtechnieken. Dit is gedocumenteerd in een testplan, waarin ook is beschreven hoe het testen aangepakt zou gaan worden.

Het testen van de software is iteratief aangepakt. Tijdens de ontwikkeling zijn er steeds ontwikkeltests en integratietests uitgevoerd en aan het eind van iedere iteratie in de construction fase werden er systeemtests uitgevoerd voor het ontwikkelde subsysteem en zijn de systeemtests herhaald van de eerdere iteraties. In de transition fase zijn er duurtests en een gebruikersacceptatietest uitgevoerd. De systeemtests zijn vastgelegd door middel van testcases, waarin de voorwaarden beschreven staan voor de test, evenals de teststappen en de verwachte resultaten. Elke testcase is gerelateerd aan een gespecificeerde eis uit het requirements document. De use-cases en de eisen aan het systeem dienden als basis voor de testcases. De resultaten van de tests zijn vervolgens gedocumenteerd in het test document.

13.3.5 H5 Professioneel werken: Zelfstandig werken

Het afstudeerproject is zelfstandig uitgevoerd. Bij aanvang is aan de hand van gesprekken met de opdrachtgever en begeleider een opdrachtomschrijving geformuleerd. De weg naar het uiteindelijke doel lag grotendeels open. Vervolgens is zelfstandig een methodiek gekozen en een daarbij behorende planning gemaakt waarin de werkzaamheden staan benoemd, en is het project uitgevoerd aan de hand van deze planning. Tijdens de uitvoering is de planning steeds in de gaten gehouden en zijn er veranderingen aangebracht in de planning indien dat nodig was. Zo werd op een gegeven moment duidelijk dat de Android applicatie meer tijd zou kosten dan gepland was. Er werd voorgesteld aan de opdrachtgever om de desktopapplicatie daarom te laten vervallen en in plaats daarvan is het alternatief aangedragen om de data in te laden via een komma gescheiden bestand.

De afstudeerder heeft steeds zelf het initiatief genomen om een gesprek te voeren met de opdrachtgever en begeleider op de momenten dat dat nodig was. Hiervoor werd een lijstje opgesteld met de te behandelen punten. Tijdens zulke gesprekken werden ook de systeemeisen achterhaald, waarbij ook werd meegedacht over de functies van het systeem. Tot slot is zelfstandig besloten voor welke delen van het systeem een analyse benodigd was en zijn er analyses uitgevoerd naar de te gebruiken technieken en protocollen. Hierbij was de vrijheid aanwezig in het maken van keuzes.

Verklarende woordenlijst

Alflex Technologies	Het bedrijf waar de afstudeeropdracht is uitgevoerd.
ANT	Een open standaard voor draadloze verbindingen tussen apparaten, meestal gebruikt in sensor-netwerken.
ANT+	Vaste standaarden bovenop ANT, voor comptabiliteit tussen verschillende producten.
Biometrie	Het vaststellen van meetbare eigenschappen van levende wezens.
Bluetooth Smart	Ook wel BLE of Bluetooth 4.0 genoemd. Een open standaard voor draadloze verbindingen tussen apparaten op korte afstand.
FatFS	Een klein FAT-bestandssysteem voor embedded applicaties.
Gazell	Een 2.4 Ghz protocol van Nordic.
I2C	Een synchrone seriële bus voor datacommunicatie tussen twee mediums.
Logic Analyzer	Een apparaatje met bijbehorende software, waarmee signalen op de lijnen kunnen worden geanalyseerd.
MoSCoW-methode	Een manier van prioriteiten stellen aan systeemeisen.
NFC	Near Field Communication, een contactloze communicatiemethode met RFID als basis.
Nordic Semiconductor	Een bedrijf dat gespecialiseerd is in het bieden van hardware oplossingen met geïntegreerde draadloze communicatie. Fabrikant van de nRF51 SoC.
Ontwikkelkit	Een hardwarebordje met een geïntegreerde microcontroller en verschillende GPIO-pinnen.
Ontwikkelmethodiek	De manier waarop het proces van het ontwikkelen van software wordt ingedeeld.
Open Up	Onwikkelmethodiek, afgeleid van RUP.
Race condition	Een gelijktijdige benadering van data vanuit twee verschillende threads, wat zorgt voor een ongedefinieerd resultaat.
RF	Radio Frequentie, de frequenties in het elektromagnetisch spectrum tussen 3 kHz en 300 GHz.
RF	Radio Frequency, radiogolven in het bereik van 3 kHz tot 300 GHz
RFID	Radio-frequency identification: identificatie door middel van radiogolven.
RFID-tag	Een kleine chip met data, uit te lezen via RF.
RTOS	Real-Time Operating system; verdeelt de functies van de software in verschillende taken met opgegeven prioriteit.
SDK	Software development kit; hulpmiddelen bij het ontwikkelen van software, zoals bestaande softwarecomponenten en documentatie.
SmarTest	Een testmethode die past bij methodieken als RUP of SCRUM.
SoC	System on Chip, een computerchip met daarop alle computercomponenten.
Softdevice	Binaire bestanden van Nordic waarin de protocol stacks voor Bluetooth Smart, ANT en Gazell zijn ondergebracht.
SPI	Serial Peripheral Interface, een synchrone seriële data link tussen ten minste twee mediums.
Toolchain	Een pakket van softwareontwikkelingtools.

Bibliografie

[1] Alflex, „Alflex,” Alflex Technologies, [Online]. Available: http://www.alflex-technologies.com/nl/company/about-alflex/. [Geopend 11 Februari 2015].

[2] F. Lucas, *Organigram Alflex,* Alflex, 2015.

[3] „What is Waterfall model- advantages, disadvantages and when to use it?,” Istqbexamcertification, [Online]. Available: http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/. [Geopend 9 Februari 2015].

[4] P. Kruchten, The Rational Unified Process An Introduction, Pearson Education, 2009.

[5] Eclipse, „Open Up,” Eclipse, [Online]. Available: http://epf.eclipse.org/wikis/openup/. [Geopend 9 Februari 2015].

[6] Scrum.nl, „What is scrum?,” [Online]. Available: http://www.scrum.nl/site/Wat-is-Scrum-agile-scrum. [Geopend 11 Maart 2014].

[7] D. Wells, „The Rules of Extreme Programming,” 1999. [Online]. Available: http://www.extremeprogramming.org/rules.html. [Geopend 9 Februari 2015].

[8] H. v. d. Bosch, *Sheets; Andere methoden van systeemontwikkeling,* HHS Delft.

[9] J. Warner en A. Kleppe, Paktisch UML, 2011.

[10] P. Abhijit, „MoSCoW Method for Requirements Prioritzation,” 20 Juni 2013. [Online]. Available: http://www.businessanalysis.in/2013/06/moscow-method-for-requirements-prioritization.html. [Geopend 11 Februari 2015].

[11] Nordic, „nRF51 Series SoC,” Nordic, [Online]. Available: https://www.nordicsemi.com/eng/Products/nRF51-Series-SoC. [Geopend 16 Februari 2015].

[12] Nordic Semiconductor, „Creating Bluetooth Low Energy Applications using nRF51822,” 2014.

[13] This is ant, „Ant+ in mobile,” [Online]. Available: http://www.thisisant.com/business/opportunities/mobile//?android-api/. [Geopend 16 Februari 2015].

[14] Nordic, „nRF51822-mKIT,” [Online]. Available: https://www.nordicsemi.com/eng/Products/Bluetooth-R-low-energy/nRF51822-mKIT. [Geopend 12 Februari 2015].

[15] RFdigital, „RFD22102 RFduino,” [Online]. Available: http://www.rfdigital.com/product/rfd22102-rfduino-dip/. [Geopend 12 Februari 2015].

[16] Atlasrfid, „RFID vs Barcode,” [Online]. Available: http://atlasrfid.com/jovix-education/auto-id-basics/rfid-vs-barcode/. [Geopend 19 Februari 2015].

[17] IBM100, „Magnetic Stripe Technology,” [Online]. Available: http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/magnetic/. [Geopend 19 Februari 2015].

[18] Smart Card Alliance, „About Smart Cards,” [Online]. Available: http://www.smartcardalliance.org/smart-cards-faq/. [Geopend 19 Februari 2015].

[19] Biometricsinstitute, „Types of Biometrics,” [Online]. Available: http://www.biometricsinstitute.org/pages/types-of-biometrics.html. [Geopend 19 Februari 2015].

[20] K. Bonsor en W. Felon, „How RFID works,” [Online]. Available: http://electronics.howstuffworks.com/gadgets/high-tech-gadgets/rfid5.htm. [Geopend 19 Februari 2015].

[21] B. Violino, „The basics of RFID Technology,” 16 Januari 2005. [Online]. Available: http://www.rfidjournal.com/articles/view?1337/. [Geopend 19 Februari 2015].

[22] J. Thrasher, „RFID vs NFC: What's the difference?,” 11 Oktober 2013. [Online]. Available:

http://blog.atlasrfidstore.com/rfid-vs-nfc. [Geopend 19 Februari 2015].

[23] N. Chandler, „What's the difference between RFID and NFC?," [Online]. Available: http://electronics.howstuffworks.com/difference-between-rfid-and-nfc.htm. [Geopend 19 Februari 2015].

[24] B. Fennani, H. Hamam en A. Dahmane, „RFID overview," Universiteit de Moncton Canada, 2011.

[25] Impinj, „RFID Technology Primer," [Online]. Available: http://www.impinj.com/resources/about-rfid/. [Geopend 25 Februari 2015].

[26] CoreRFID, „Mifare RFID Tags: A Short Buyer's Guide," 2008. [Online]. Available: http://www.corerfid.com/Files/067%20Mifare%20Tag%20Buyers%20Guide.pdf. [Geopend 24 Februari 2015].

[27] K. Townsend, „Introduction Bluetooth Smart," Adafruit, 20 Maart 2014. [Online]. Available: https://learn.adafruit.com/introduction-to-bluetooth-low-energy/introduction. [Geopend 3 Maart 2015].

[28] Laura, „Bluetooth Low Energy Introduction," 16 Januari 2014. [Online]. Available: http://mbientlab.com/blog/bluetooth-low-energy-introduction/. [Geopend 3 Maart 2015].

[29] ANT, „ANT Message Protocol and Usage," 2014.

[30] Engenuics, „ANT Radio System".

[31] S. Khssibi, H. Idoudi, A. van den Bossche, T. Val en A. L. Saidane, „Presentation and analysis of a new technology for low-power wireless sensor network," Universiteit van Toulouse, Frankrijk, 2013.

[32] N. Lethaby, „Why use a Real-Time Operating System in MCU applications," Texas Instruments Incorporated, Dallas, Texas, 2013.

[33] KEIL, „RTOS Advantages," [Online]. Available: http://www.keil.com/rl-arm/rtx_rtosadv.asp. [Geopend 13 Maart 2015].

[34] O. Morten, „FreeRTOS," 17 Januari 2014. [Online]. Available: https://devzone.nordicsemi.com/question/2197/freertos/. [Geopend 27 Maart 2015].

[35] R. C. Martin en M. Martin, Agile Principles, Patterns, and Practices in C#, Pearson Education, Inc, 2012.

[36] E. Gamma, R. Helm, R. Johnson en J. Vlissides, Design Patterns, Addison-Wesley, 2001.

[37] Chan, „FatFS - Generic FAT File System Module," 2015. [Online]. Available: http://elm-chan.org/fsw/ff/00index_e.html. [Geopend 11 Maart 2015].

[38] Bluetooth SIG, „Serial Port Profile," 2015. [Online]. Available: https://developer.bluetooth.org/TechnologyOverview/Pages/SPP.aspx. [Geopend 9 Mei 2015].

[39] W.-M. Lee, Beginning Android 4 Application Development, Canada: John Wiley & Sons, Inc., 2012.

[40] Android , „Navigation Drawer," [Online]. Available: https://developer.android.com/design/patterns/navigation-drawer.html. [Geopend 9 mei 2015].

[41] E. Bouman, SmarTEST, Slim testen van informatiesystemen, Den Haag: Sdu Uitgevers bv, 2008.