2023

ATP | Project- en Testplan

SLAAPKAMER COMFORT-REGELSYSTEEM

AHMET SERDAR ÇANAK

Datum:	14-4-2024
Cursuscode:	TCTI-VKATP-21
Naam:	Ahmet Serdar Çanak
Studierichting:	Technische Informatica
Specialisatie:	Embedded Systems Engineering
Studentennummer:	1760039

Inhoudsopgave

1. Inleiding

In het kader van de voortschrijdende ontwikkelingen in slimme thuisautomatisering, wordt in dit verslag een “Slaapkamer Comfort-Regelsysteem” geïntroduceerd en geanalyseerd. Dit prototypesysteem heeft als doel de leefomstandigheden in een slaapkamer te optimaliseren, waarbij gebruik wordt gemaakt van twee sensoren, namelijk een lichtsensor en een temperatuursensor, in combinatie met twee actuatoren: een generieke rolluikmotor en koelunit.

Deze rapportage belicht niet alleen de conceptuele en technische aspecten van het Regelsysteem, maar benadrukt tevens de cruciale rol van kwaliteitseisen, testen en validatie in het ontwikkelingsproces. Het gepresenteerde testplan is ontworpen om de effectiviteit, prestaties en betrouwbaarheid van het systeem te beoordelen. Het hoofddoel van deze testen is om te verzekeren dat het regelsysteem in staat is om het comfort van de gebruikers te handhaven.

In de aankomende secties zal het regelsysteem worden toegelicht met bijbehorende componenten, zullen de architectuurschetsen worden gepresenteerd en besproken, zal het testplan in detail worden behandeld. Tot slot heb ik verwijzing naar de datasheets van de desbetreffende componenten genoteerd.

2. Regelsysteem

Slaapkamer Comfort-Regelsysteem:

Het Slaapkamer Comfort-Regelsysteem is een automatiseringssysteem ontworpen om het comfort en welzijn in de slaapkamer te verbeteren. Het systeem maakt gebruik van twee belangrijke subsystemen voor temperatuurregeling en ochtendwekkerfunctionaliteit.

1. Temperatuurregeling met Ventilator en Temperatuursensor:

Dit systeem is gericht op het handhaven van een aangename slaapomgeving door temperatuurvariaties te beheren. Het bevat de volgende componenten:

- **Temperatuursensor**: Een temperatuursensor meet voortdurend de omgevingstemperatuur in de slaapkamer.
- **Koelunit**: Een koelunit is geïntegreerd in het systeem en wordt geactiveerd als de temperatuur in de slaapkamer boven een vooraf ingestelde drempel stijgt.
- **Automatische Aanpassing**: Als de omgevingstemperatuur weer binnen het gewenste bereik valt, zal de koelunit automatisch worden uitgeschakeld om onnodig energieverbruik te voorkomen.

2. Ochtendwekker met Zonlichtdetectie en Rolluikmotor:

Dit systeem fungeert als een intelligente ochtendwekker en is gebaseerd op de gecombineerde functionaliteit van een lichtsensor en een rolluikmotor. Het werkt als volgt:

- **Tijdinstelling**: De gebruiker kan een specifieke tijd instellen waarop ze wakker willen worden.
- **Lichtsensor**: Een lichtsensor detecteert de hoeveelheid omgevingslicht bij zonsopgang.
- **Alarmactivering**: Een uur voor de door de gebruiker ingestelde tijdstip, wordt het alarm geactiveerd. Als er voldoende daglicht aanwezig is, zorgt het systeem ervoor dat de rolluiken automatisch worden geopend met behulp van de rolluikmotor indien de rolluiken gesloten waren.
- **Ontwaken in Natuurlijk Licht**: De combinatie van het alarm en het openen van de rolluiken zorgt ervoor dat de gebruiker ontwaakt in een kamer die wordt verlicht door natuurlijk daglicht, wat het ontwaken aangenamer maakt.

2.1. Sensoren

- **Lichtsensor**: Als lichtsensor heb ik gekozen voor de BH1750, omdat het aan te sturen is via I^2C en omdat het voldoet aan de eis om lichtintensiteit te kunnen meten.
- **Temperatuursensor**: Als temperatuursensor heb ik gekozen voor de DHT11, omdat het aan te sturen is via een data-pin en het voldoet aan de eis om de omgevingstemperatuur te meten.

2.2. Actuatoren

- **Rolluikmotor**: Aangezien er geen rolluikmotoren zijn, heb ik gekozen voor een generiek apparaat als rolluikmotor wat aan en uitgezet kan worden via een data-pin.
- **Koelunit**: Aangezien er geen koelunits zijn, heb ik gekozen voor een generiek aparaat als koelunit wat aan en uitgezet kan worden via een data-pin.

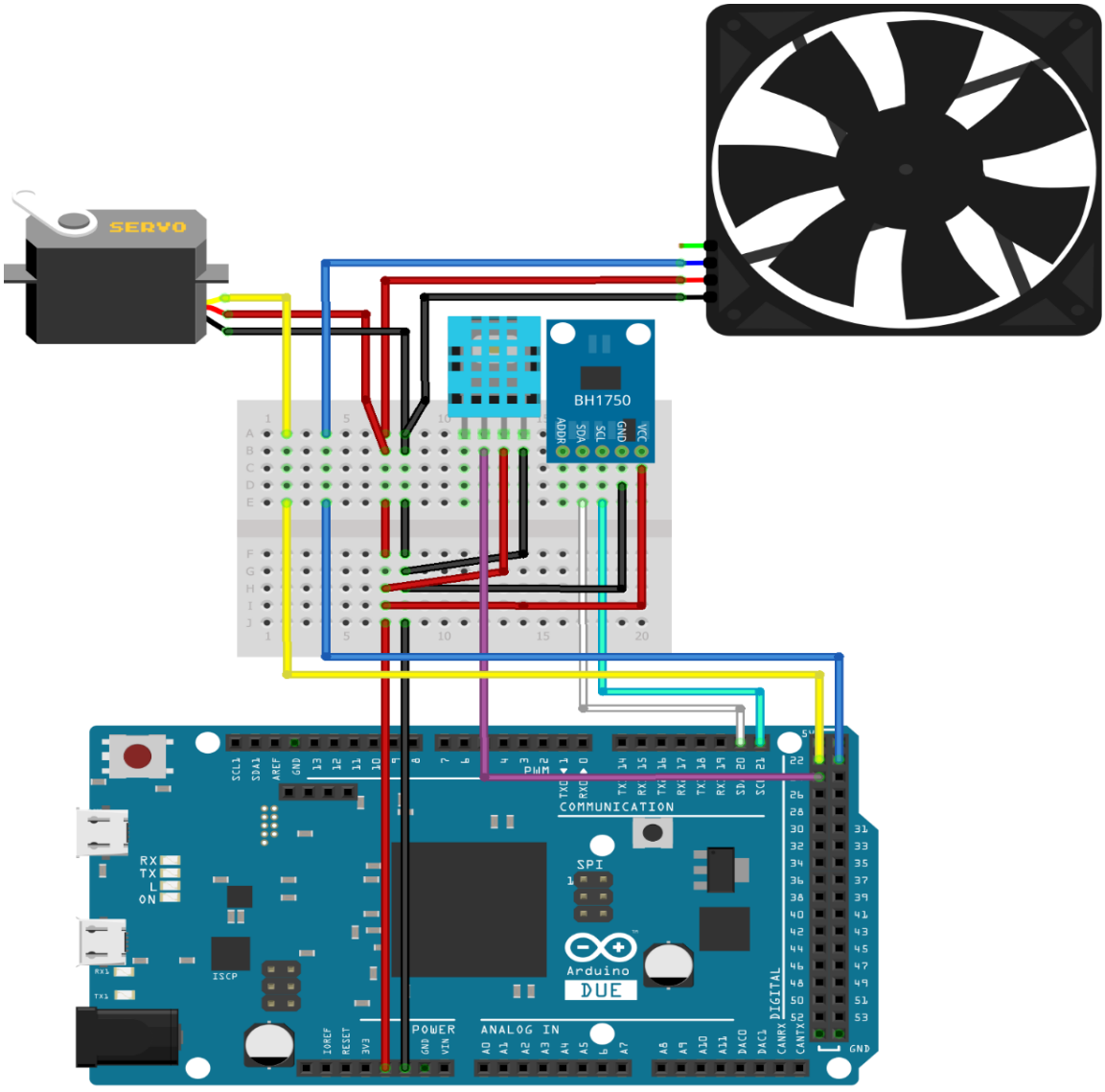
3. Architectuurschetsen

3.1. Hardware

In de Hardware-schets afgebeeld in figuur 1, is een Arduino Due microcontroller gebruikt in combinatie met een aantal sensoren en actuatoren om een geautomatiseerd systeem te realiseren. De Arduino Due fungeert als het brein van het systeem en stelt gebruikers in staat om verschillende aspecten van hun omgeving te meten en te controleren.

De schets omvat de aansluiting van een koelunit, een rolluikmotor, een BH1750-lichtsensor en een DHT11-temperatuur- en vochtigheidssensor. De koelunit, rolluikmotor en DHT11-sensoren zijn aangesloten via digitale-pinnen voor gegevensoverdracht en de BH1750-lichtsensor maakt gebruik van de I²C-communicatie-interface voor gegevensoverdracht.

De schets is bedoeld om een praktische demonstratie te bieden van de integratie van verschillende sensoren en actuatoren met een Arduino-microcontroller, waardoor gebruikers in staat worden gesteld om de omgevingsvariabelen te bewaken en te beïnvloeden.



Figuur 1: Hardware schets

3.2. Software

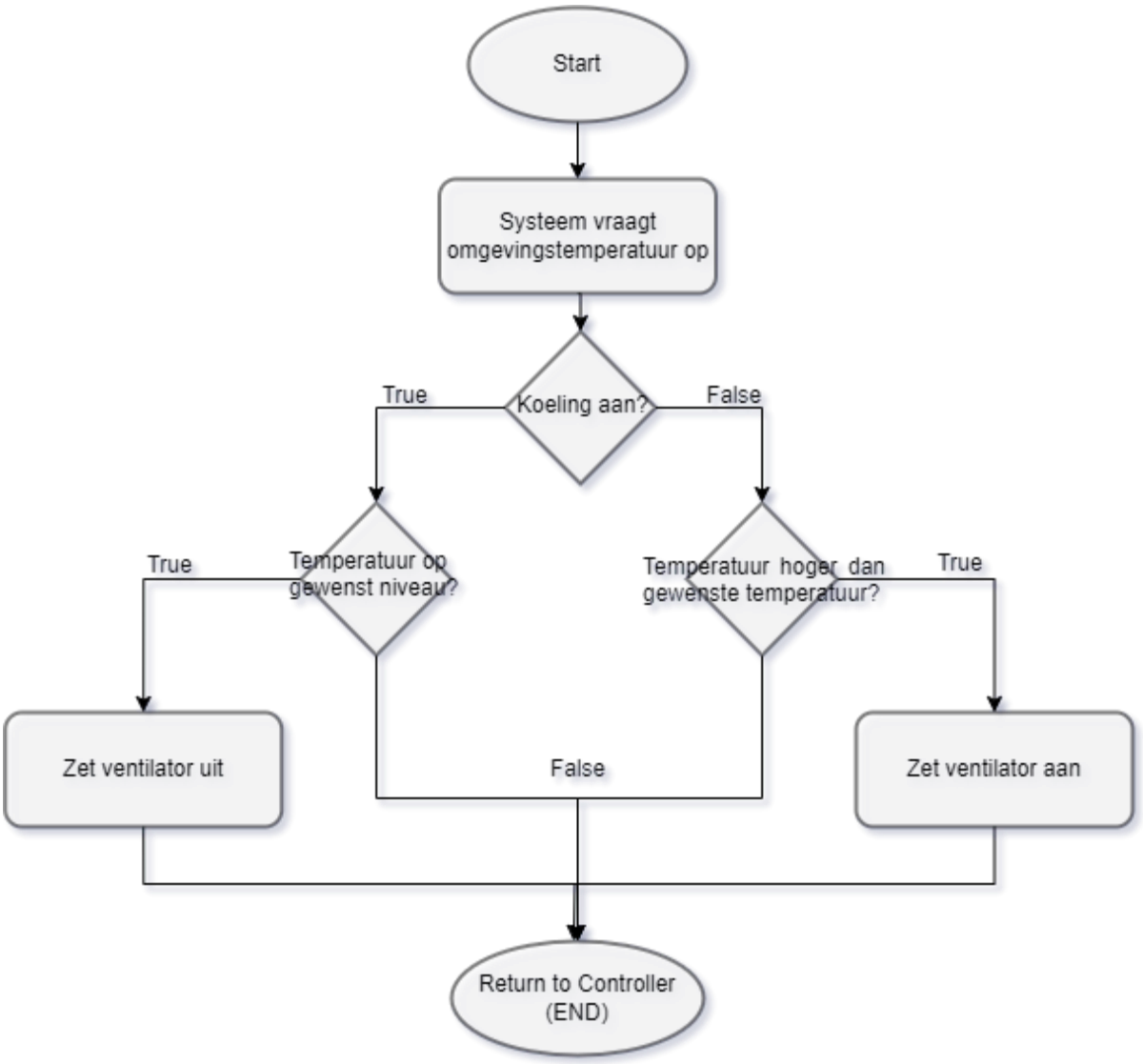
3.2.1. Flowcharts

De onderstaande flowcharts geven weer hoe de flow van de subsystemen eruit zullen zien.

3.2.1.1. Temperatuurregeling-subsysteem

Het Temperatuurregeling-subsysteem vraagt als allereerste de huidige omgevingstemperatuur op. Vervolgens wordt gecontroleerd of het koelsysteem reeds actief is. Als dit niet het geval is, wordt gekeken of de omgevingstemperatuur hoger is dan de gewenste temperatuur. Als dat het geval is, wordt het koelsysteem geactiveerd. Als dit niet het geval is, blijft de huidige toestand gehandhaafd.

Als het koelsysteem al actief is, wordt gecontroleerd of de omgevingstemperatuur de gewenste temperatuur heeft bereikt. Als dat niet het geval is, blijft de koelunit draaien om de gewenste temperatuur te handhaven. Zodra de gewenste temperatuur is bereikt, wordt de koelunit uitgeschakeld om energie te besparen.

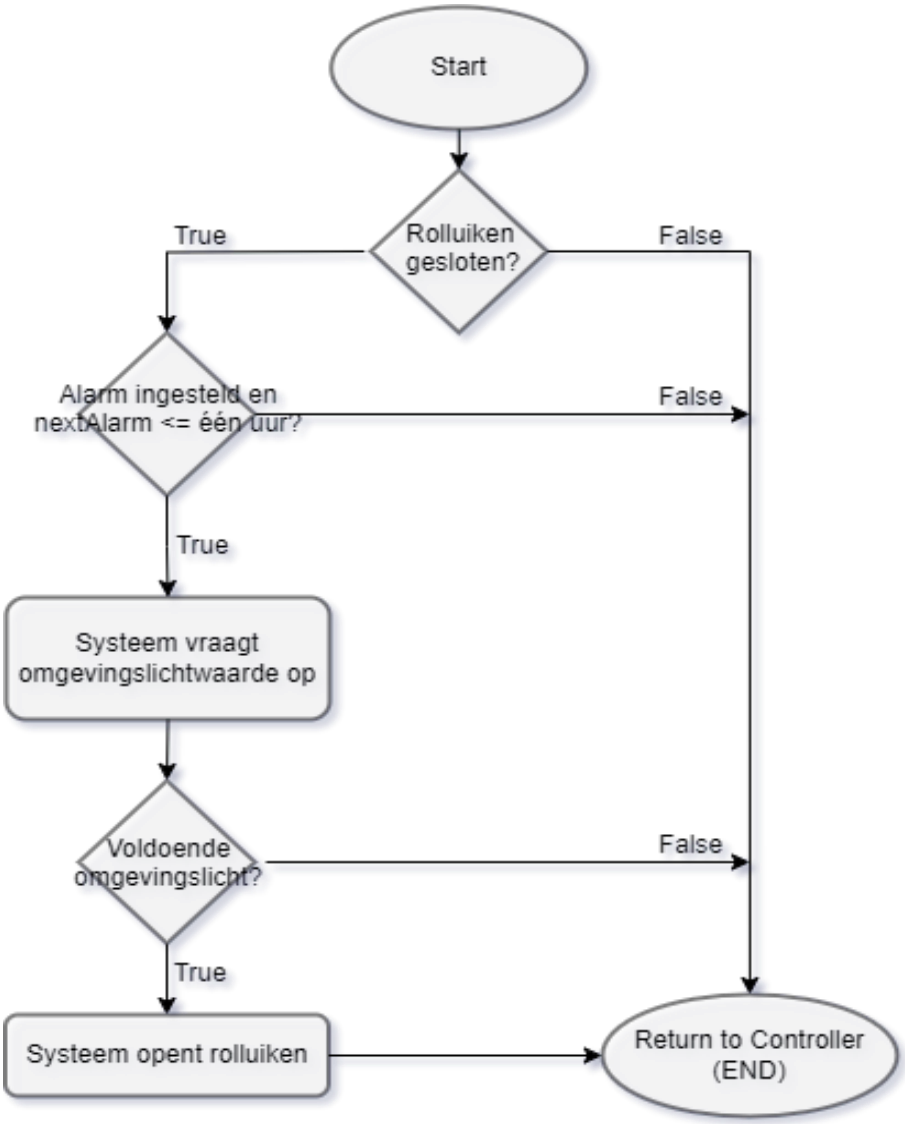


Figuur 2: Temperatuurregeling-subsysteem

3.2.1.2. Ochtendwekker-subsysteem

Het Ochtendwekker-subsysteem begint met een controle of het rolluik is gesloten. Nadat is geverifieerd dat het rolluik gesloten is, kijkt het systeem naar de ingestelde wekkertijd. Als er een alarm is ingesteld dat binnen het komende uur moet afgaan, wordt het proces voortgezet.

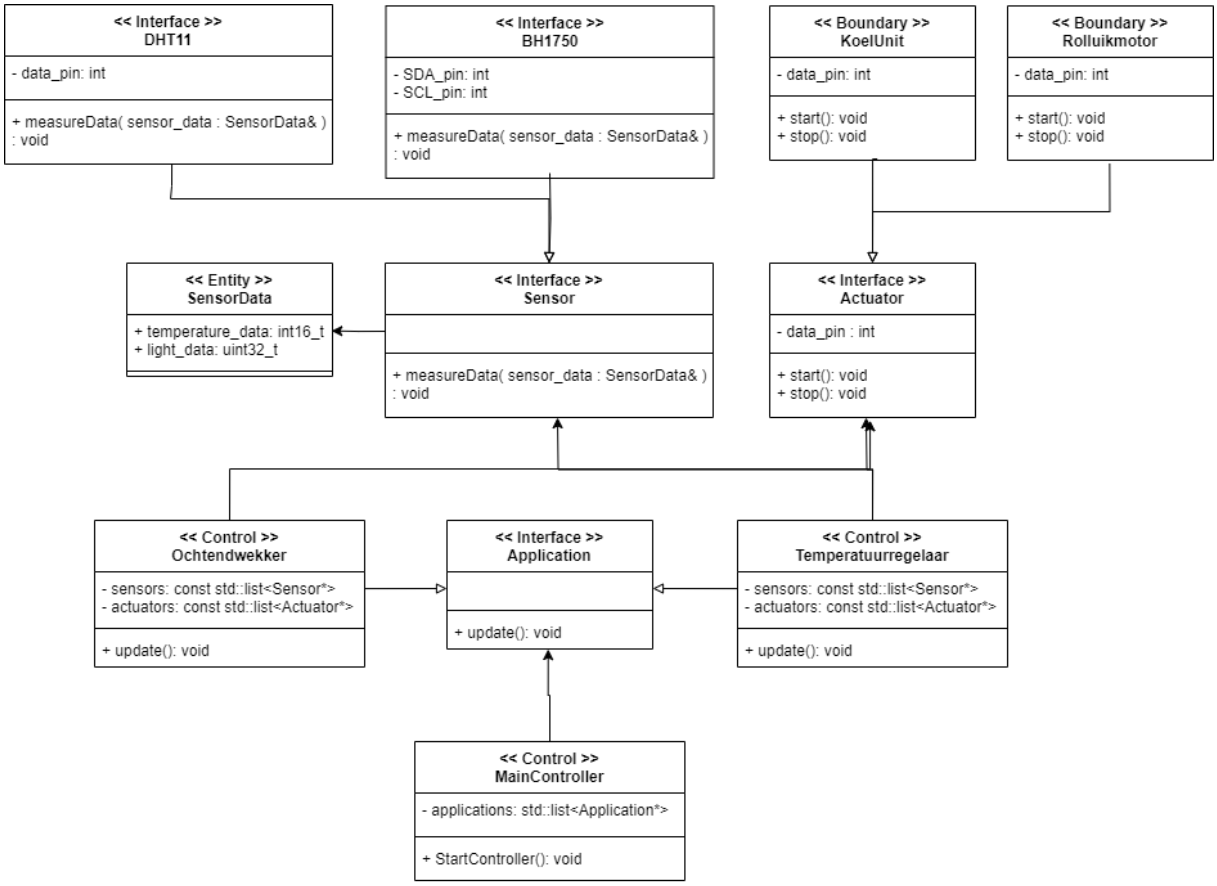
Op dit punt roept het systeem de lichtsensor om de actuele omgevingslichtwaarde te verkrijgen. Met deze informatie wordt gecontroleerd of de huidige lichtomstandigheden voldoende zijn om op een natuurlijk wijze wakker te worden. Indien er voldoende omgevingslicht is, initieert het systeem het openen van het rolluik. Door dit te doen, wordt de ruimte gevuld met natuurlijk zonlicht, waardoor het mogelijk wordt voor de gebruiker om aangenaam en geleidelijk wakker te worden.



Figuur 3: Ochtendwekker-subsysteem

3.2.2. Klassen diagram

Zoals te zien in de klassen diagram afgebeeld in figuur 4, zal de “hardware” aansturing gedaan worden met interfaces. De code omtrent deze interfaces zullen geschreven worden in C++ en de kamer zal worden gesimuleerd in Python en dit allemaal zal worden gekoppeld met een binding zoals PyBind11.



Figuur 4: Klassen diagram C++ code met alle verschillende klassen, bijbehorende relaties en functies.

4. Kwaliteitscriteria

In de wereld van informatietechnologie is het leveren van kwalitatief hoogstaande software essentieel voor het succes van elk project. Normen zoals ISO 25010 [1] stellen criteria vast voor verschillende kwaliteitsaspecten, zoals functionaliteit, betrouwbaarheid en bruikbaarheid. Deze criteria zijn van cruciaal belang om de algehele kwaliteit van een IT-oplossing te waarborgen. Door deze normen te volgen, kunnen organisaties risico's verminderen, de tevredenheid van gebruikers verbeteren en de algehele efficiëntie van het project vergroten. Kortom, het integreren van kwaliteitscriteria is een sleutelpraktijk om het succes van IT-projecten te verzekeren.

Om de codekwaliteit te garanderen heb ik alle aspecten van ISO 25010 geprioriteerd, aangezien het regelsysteem relatief simpel systeem is, hebben sommige aspecten van ISO 25010 een ~~lage prioriteit~~ en zijn dus weggelaten, terwijl andere aspecten een hoge prioriteit hebben:

- Functionele Geschiktheid:
    - Functionele Volledigheid: De mate waarin de set functies alle gespecificeerde taken en gebruikersdoelen omvat.
    - Functionele Correctheid: Het vermogen van de software om de juiste resultaten te leveren.
    - Functionele geschiktheid: Mate waarin de functies het uitvoeren van gespecificeerde taken en doelstellingen vergemakkelijken.
- Betrouwbaarheid:
    - Volwassenheid: De frequentie van het falen van een product.
    - Beschikbaarheid: Mate waarin een systeem, product of onderdeel operationeel en toegankelijk is wanneer het nodig is voor gebruik.
- Onderhoudbaarheid:
    - Testbaarheid: Mate van effectiviteit en efficiëntie waarmee testcriteria kunnen worden vastgesteld voor een systeem, product of onderdeel, en tests kunnen worden uitgevoerd om te bepalen of aan die criteria is voldaan.

4.1 Belang van kwaliteitscriteria

Op basis van de prioriteiten heb ik de volgende relatieve belangstellingen toegewezen die ik nader zal toelichten.

Kwaliteitscriterium	Relatieve belangstelling (%)
1. Functionele Geschiktheid	40
2. Prestatie-efficiëntie	0
3. Compatibiliteit	0
4. Gebruiksvriendelijkheid	0
5. Betrouwbaarheid	30
6. Veiligheid	0
7. Onderhoudbaarheid	30
8. Overdraagbaarheid	0

- 1. Functionele Geschiktheid: Het systeem moet aan de vooraf gestelde eisen voldoen van de gebruiker anders kan je het project eigenlijk gelijk afschrijven.
- 2. Prestatie-efficiëntie: Voor dit systeem is de optimalisatie van code niet van belang aangezien het een relatief simpel systeem is.
- 3. Compatibiliteit: Dit product heeft geen predescessors en zal geen successors krijgen waar het rekening mee zal moeten houden.
- 4. Gebruiksvriendelijkheid: De gebruiker hoeft dit product niet al te makkelijk te kunnen gebruiken aangezien het product grotendeels geautomatiseerd is.
- 5. Betrouwbaarheid: Het product moet eigenlijk ten alle tijden beschikbaar zijn als het gedeployed is anders heb je niet echt veel aan het systeem.
- 6. Veiligheid: Het systeem beschikt zich niet over belangrijke data waardoor deze criteria niet echt van belang is.
- 7. Onderhoudbaarheid: Het is belangrijk in elk systeem om modulaire, herbruikbare, aanpasbare en testbare code te bevatten en daarom heeft het ook de hoogste prioriteit gekregen.
- 8. Overdraagbaarheid: Aangezien het product een one off product is hoeft de code niet overdraagbaar te zijn en heeft het dus ook geen prioriteiten gekregen.

5. Testen

Voor dit project moet één unit test, één integratietest en één systeemtest worden uitgevoerd. Alleen deze testen worden dan ook in dit testplan verder uitgewerkt en beschreven.

5.1. Unit Test

Het doel van een unit test is om losse hardware of software componenten te testen. In mijn geval wil ik één van mijn subsystemen testen op 'functionele correctheid.

Ik heb er voor gekozen om de logica van mijn temperatuurregeling subsysteem te testen, omdat een deel van het grotere systeem niet zal functioneren als deze sub-module foutieve resultaten oplevert.

Slagingscriteria:

- De test is succesvol als de output van de functie overeenkomt met de flowchart in figuur 2.
    - Als de koelunit uit staat en de temperatuur hoger is dan gewenst, moet de ventilator aangaan (de functie retourneert '1').
    - Als de koelunit aan staat en de temperatuur gelijk is aan de gewenste temperatuur, moet de ventilator uitgaan (de functie retourneert '0').
    - Als de koelunit uit staat en de temperatuur gelijk is aan de gewenste temperatuur, blijft het systeem in dezelfde staat (de functie retourneert 'None').
    - Als de koelunit aan staat en de temperatuur nog niet gelijk is aan de gewenste temperatuur, blijft het systeem in dezelfde staat (de functie retourneert 'None').

Testuitvoering:

1. De functie ontvangt drie waarden: de omgevingstemperatuur, de gewenste temperatuur en de status van de koelunit.
2. Vervolgens berekent de functie wat er moet gebeuren met de koelunit.
3. Na de berekening retourneert de functie de nieuwe gewenste staat voor de koelunit.

Dit proces wordt vervolgens tweemaal herhaald voor elke flow in de flowchart, wat resulteert in acht unit tests voor de desbetreffende sub-module.

Motivatie en potentiële risico's:

De voorgestelde unit test voor het temperatuurregelingssubsysteem van het slaapkamer comfortsysteem, waarbij de temperatuur wordt gereguleerd door een koelunit en het slaapcomfort wordt beïnvloed, is van essentieel belang om te voldoen aan de functionele correctheidseisen van de ISO 25010-norm. Dit systeem speelt een cruciale rol bij het waarborgen van het comfort en welzijn van de gebruikers tijdens de slaap.

Door de temperatuur nauwkeurig te regelen en te zorgen voor optimaal slaapcomfort, draagt dit subsysteem bij aan een goede nachtrust en algemeen welzijn. Een foutieve werking van dit subsysteem kan echter leiden tot ernstige negatieve gevolgen, zoals ongemak tijdens de slaap, verstoring van de slaappatronen en zelfs gezondheidsproblemen.

De unit test, die de functionaliteit van het temperatuurregelingssubsysteem valideert volgens de specificaties in de flowchart, is daarom van cruciaal belang. Het identificeert potentiële fouten in de logica van het systeem, zoals onjuiste reacties op temperatuurwensen. Het niet uitvoeren van deze test kan leiden tot een verminderde slaapkwaliteit en comfort voor de gebruikers, waardoor het systeem zijn beoogde doel niet kan bereiken en dus niet functioneel correct is.

5.2. Integratietest

1. Beschrijving van de test: Een integratie test wordt ingezet als je bijvoorbeeld hardware en/of software componenten in combinatie met elkaar wilt testen. In mijn geval wil ik het toepassen om de functionele geschiktheid van mijn hardware en software componenten te testen. Ik zal deze test uit gaan voeren na mijn unit tests aangezien de logica dan dus functioneel is en ik dus kan overstappen naar de volgende stap en dat is de logica daadwerkelijk integreren in het systeem.
2. Doel: Het doel van deze test is om te valideren dat de integratie van mijn subsystemen correct functioneert en dus ook de hardware aanstuurt.
3. Motivatie: Als de hardware correct wordt aangestuurd is het systeem dus eigenlijk zo goed als klaar om in zijn geheel getest te worden.
4. Testcriteria: De functies van de subsystemen moeten voldoen aan de eisen die afgebeeld zijn in de flowcharts.
5. Slagingscriteria: De test is geslaagd als de hardware componenten aan of uit gaan wanneer de softwarecomponenten dat van ze vereisen.
6. Risicoanalyse: Wanneer dit onderdeel niet goed wordt getest is er een kans dat de subsysteem de hardware niet aan kunnen sturen en dus ook de daadwerkelijke eisen van de gebruiker niet kunnen verrichten. In dit project is de schade van het risico niet een al te groot probleem aangezien de desbetreffende persoon alleen niet op een eerder tijdstip zal ontwaken en/of misschien zal gaan zweten in zijn slaap.

5.3. Systeemtest

7. Beschrijving van de test: Een systeemtest wordt opgesteld om te valideren dat het systeem voldoet aan de opgestelde requirements. In mijn geval wil ik het systeem dan volledig testen.
8. Doel: Het doel van deze test is om te bepalen of het regelsysteem voldoet aan alle eisen en dus automatisch wordt geregeld.
9. Motivatie: Als het systeem deze test behaald is het klaar voor de Acceptance en Field tests.
10. Testcriteria: Het systeem moet voldoen aan de requirements.
11. Slagingscriteria: De test is gehaald als het regelsysteem voldoet aan alle requirements.
12. Risicoanalyse: Als de test niet gehaald wordt is het systeem dus niet klaar om geaccepteerd te worden door de gebruiker.

6. Decorators

Bij dit project zal er gebruikt gemaakt worden van één decorator: measure_time.

6.1. measure_time decorator

De decorator ‘measure_time’ meet de uitvoeringstijd van een functie en geeft deze informatie weer door het te loggen.

Door de decorator ‘measure_time’ toe te voegen, kunnen we de prestaties van verschillende functies binnen het systeem evalueren en eventuele knelpunten in de uitvoeringstijd identificeren.

Zonder de ‘measure_time’ decorator zouden we geen inzicht hebben in de uitvoeringstijd van de verschillende functies binnen het systeem. Dit zou het moeilijker maken om prestatieproblemen op te sporen en te optimaliseren en/of verbeteren.

```python
import time

def measure_time(func):
    def wrapper(*args, **kwargs):
        start_time = time.time()
        result = func(*args, **kwargs)
        print(f"Function '{func.__name__}' took {time.time() - start_time:.6f} seconds to execute.")
        return result
    return wrapper
```

Figuur 5: decorator measure_time.

7. Functional Programming

In dit project zullen twee principes uit functioneel programmeren worden toegepast: pure functies en hogere-orde functies.

7.1 Pure functies

Een belangrijk concept dat ik heb toegepast, is het gebruik van pure functies, zoals te zien bij de thermostaatLogic-functie in figuur 6. In deze functie accepteert thermostaatLogic de nodige parameters (actualTemp, desiredTemp, coolerOn) en retourneert een waarde op basis van die parameters. Door de afwezigheid van bijwerkingen op de interne toestand van een object, wordt thermostaatLogic beschouwd als een pure functie.

Door dergelijke functies te gebruiken, wordt de code voorspelbaar en gemakkelijk te begrijpen. Dit maakt het makkelijker om te testen en debuggen, en biedt voordelen zoals verbeterde parallelle uitvoering en verminderde complexiteit van het systeem.

```
1   def thermostaatLogic(actualTemp, desiredTemp, coolerOn):
2       if coolerOn:
3           if actualTemp <= desiredTemp - 1.0:
4               return "stop"
5       else:
6           if actualTemp >= desiredTemp + 1.0:
7               return "start"
8       return "none"
```

Figuur 6: pure functie thermostaatLogic.

7.2. Hogere-orde functies

In mijn code heb ik het principe van hogere-orde functies toegepast door gebruik te maken van een decorator genaamd ‘measure_time’ te zien in figuur 5. Deze decorator stelt me in staat om de uitvoeringstijd van verschillende functies te meten zonder de oorspronkelijke code van die functies te wijzigen. Hierdoor kan ik de prestaties van mijn code analyseren en optimaliseren zonder de kernlogica te verstoren.

Datasheets

[1] ISO, „ISO 25010,” [Online]. Available: https://iso25000.com/index.php/en/iso-25000-standards/iso-25010. [Geopend 9 November 2023].

[2] R. Semiconductor, „BH1750FVI : Sensor ICs,” November 2011. [Online]. Available: https://www.mouser.com/datasheet/2/348/bh1750fvi-e-186247.pdf. [Geopend 25 September 2023].

[3] L. Aosong Electronics Co., „DHT11-Temperature-Sensor.pdf,” [Online]. Available: https://components101.com/sites/default/files/component_datasheet/DHT11-Temperature-Sensor.pdf. [Geopend 25 September 2023].