TAU INF202 Software Engineering

Individuelles Projekt

Architekturspezifikation

Version: 05.05.2023

Plant Management System

Verantwortliche/r:

Mahmutcan İlhandağ, e190503018@stud.tau.edu.tr

Oğuzhan Topal, e190503001@stud.tau.edu.tr

Stakeholder: DI. Ömer Karacan, omer.karacan@tau.edu.tr

Dokumentenverwaltung

Dokument-Historie

Version	Status *)	Datum	Verantwortlicher	Änderungsgrund
v1.0	Entwurf	05.05.2023	Mahmutcan İlhandağ Oğuzhan Topal	Dokument erstellt wurde.

**) Sofern im Projekt nicht anders vereinbart, sind folgende Statusbezeichnungen zu verwenden*
(in obiger Tabelle und am Deckblatt):
Dokument-Status: Entwurf / in Review / freigegeben (abgegeben)

Dokument wurde mit folgenden Tools erstellt:
Microsoft Office Word

Inhaltsverzeichnis

1. Einleitung

In diesem Dokument erläutern wir die architektonische Struktur des Plant Management System-Projekts. Wir werden zeigen, dass die zuvor im Pflichtenheft ermittelten Use Cases in dieser architektonischen Planung erfüllt werden. Wir zeigen und erklären die Datenbankstruktur.

In *Architekturüberblick*, werden wir den Aufbau unseres Projektes mit Schaubildern skizzieren und erläutern.

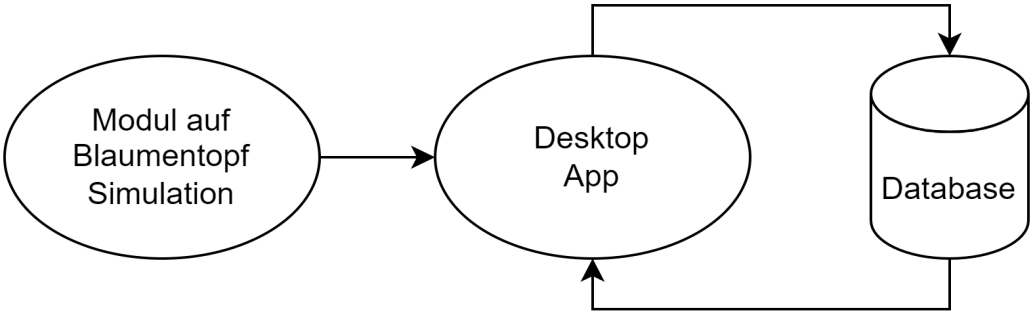
In *Beschreibung von Kontroller Klassen*, erklären wir die Funktionen der Klassen, die die Ausführung der Programme steuern, aus denen unser Projekt besteht.

In *Rückverfolgbarkeit der Anforderungen*, zeigen wir, welche Use Cases die von uns beschriebenen Funktionen im Pflichtenheft erfüllen.

In *Beschreibung der DB-Zugriffsschicht*, erläutern wir den Aufbau von Datenbank, die die zu speichernden Daten enthält.

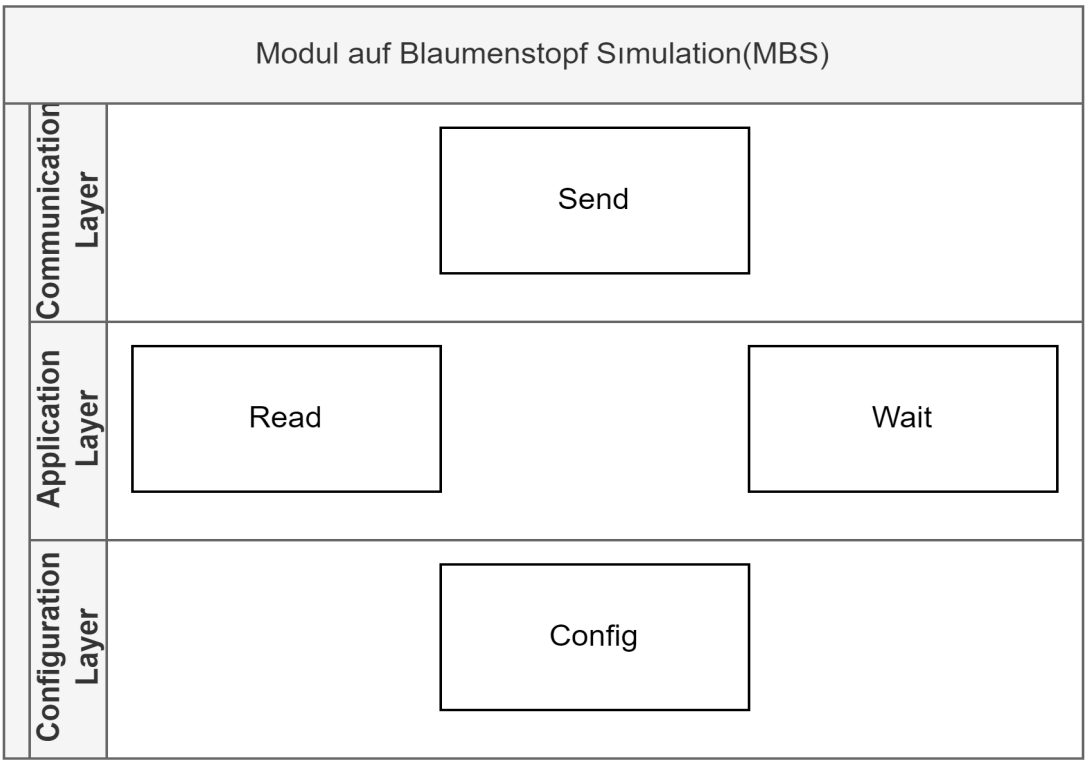
2. Architektur Überblick

Wie wir bereits im Pflichtenheft erwähnt haben, besteht unser Plant Management System-Projekt aus zwei Teilprogrammen. Dies sind Desktop App (DA) und Modul auf Blumentopf Simulation (MBS). Diese beiden Programme werden zwei getrennte Strukturen haben und miteinander kommunizieren.



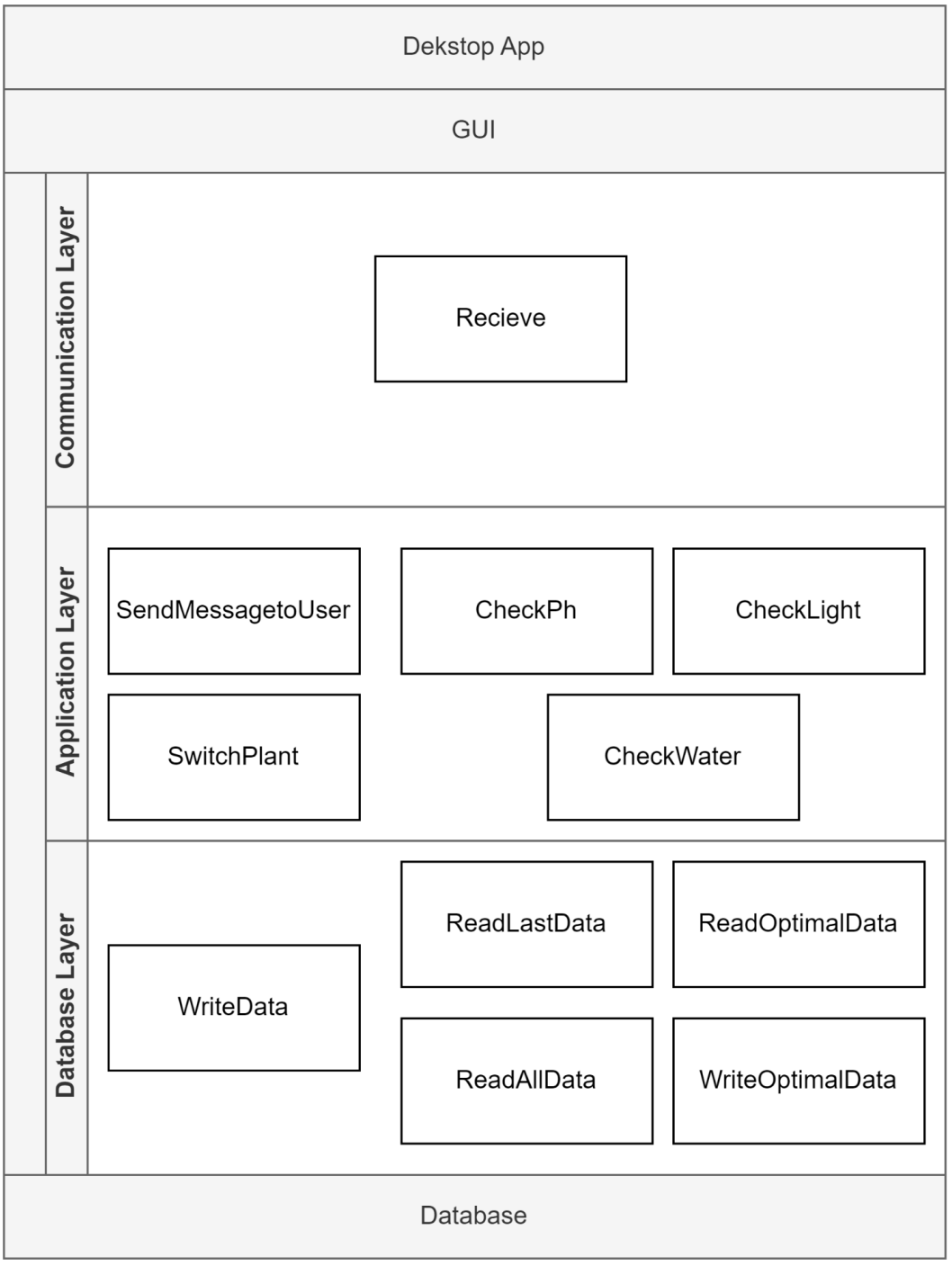
2.1 Modul auf Blumentopf Simulation

Wir haben das MBS-Programm so geplant, dass es aus 3 Schichten besteht. Diese Schichten sind Configuration, Application und Communication. Die Konfiguration Schicht kümmert sich um das Lesen der benutzerdefinierten Einstellungen und das Ausführen des Programms gemäß diesen Einstellungen. (Zum Beispiel: Wartezeit für Datenübertragung) Application Schicht umfasst das Lesen von Daten aus txt-Dateien der Anwendungsschicht und das Warten auf das Programm zwischen zwei Übermittlungen. Die Kommunikationsschicht ist für den Datenaustausch zwischen MBS und DA zuständig.



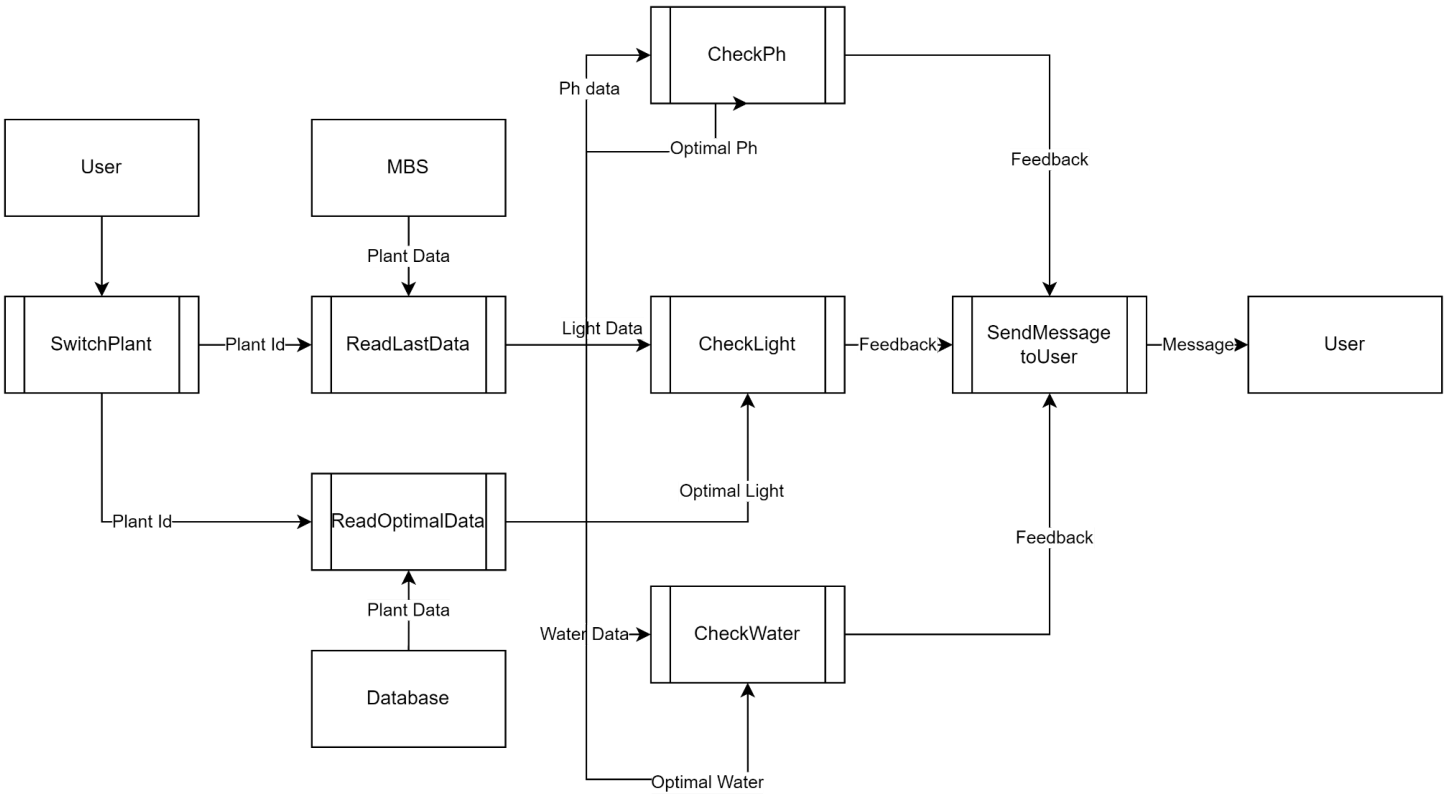
2.2 Desktop App

Die Desktop-App besteht aus GUI, Kommunikationsschicht, Anwendungsschicht, Datenbankschicht und Datenbankschichten. Das GUI-Programm dieser Schichten stellt die Benutzerkommunikation bereit. Communication Layer verantwortlich für die Kommunikation zwischen Desktop App und MBS. Die Anwendungsschicht ist für die Ausführung der Hauptfunktionen der App verantwortlich. Die Datenbankschicht ist für die Datenübertragung zwischen dem DA und der Datenbank verantwortlich. In der Datenbank werden Daten gespeichert, die von der Anwendung getrennt gehalten werden müssen.



2.3 Data Flow

In diesem Abschnitt haben wir den interfunktionalen Weg aufgezeigt, dem die Daten während des Betriebs unserer Anwendung folgen werden.



3. Beschreibung der Kontroller Klassen

3.1 Modul auf Blumentopf Simulation

- **Communication Layer**
  Send Package Funktion bereitet ein Datenpaket vor, das die Daten und die MBS-ID enthält, und sendet dieses Paket an den DA.
- **Application Layer**
  Wait Funktion wartet eine bestimmte Zeit.
- **Configuration Layer**
  Config Funktion liest die Einstellungen aus einer Konfigurationsdatei und nimmt die notwendigen Anpassungen im Programm vor und weist die Daten den entsprechenden Variablen zu.

3.2 Desktop App

- **Communication Layer**
  Recieve Funktion liest das von MBS gesendete Datenpaket.
- **Application Layer**
  SendMessagetoUser Funktion erstellt als Ergebnis der Steuerungen die erforderliche Nachricht und sendet sie an die GUI zur Anzeige für den Benutzer.
  CheckPH Funktion überprüft die Eignung des pH-Wertes der Pflanze.
  CheckLight Funktion überprüft die Eignung des Licht-Wertes der Pflanze.
  CheckWater Funktion überprüft die Eignung des Water-Wertes der Pflanze.
  SwitchPlant Funktion ändert den Kontext der Plant, in DA ausgeführt wird.
- **Database Layer**
  WriteData Funktion zeichnet die reelle Daten aus der Anlage in die Datenbank auf.
  ReadLastData Funktion liest die zuletzt erfassten reelle Daten der jeweiligen Anlage aus der Datenbank.
  ReadAllData Funktion liest alle reelle Daten der jeweiligen Anlage aus der Datenbank.
  ReadOptimalData Funktion liest optimale Daten der jeweiligen Anlage aus der Datenbank.
  WriteOptimalData Funktion schreibt neue optimale Werte für die jeweiligen Anlage.

4. Rückverfolgbarkeit der Anforderungen

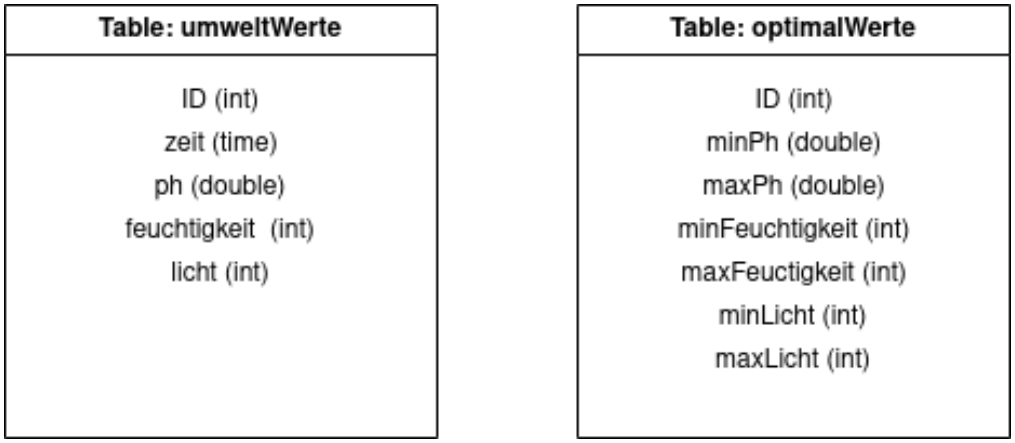
Use Case	Funktion
• **/MBS-1/** Das Modul sollte in der Lage sein, Daten aus 3 verschiedenen Dateien zu lesen (3 Textdateien, die Licht-, pH- und Feuchtigkeitssensoren darstellen).	ReadDaten
• **/MBS-2/** Das Modul muss terminiert werden. Er soll in bestimmten Zeitabständen eine Verbindung zum Client aufbauen und Daten senden. (Im realen Szenario wird dieses Zeitintervall mit 6 Stunden angenommen, in der Simulation wird es jedoch auf eine repräsentativ kurze Zeit eingestellt.)	Wait, Config, SendPackage
• **/MBS-3/** Das Modul sollte die ihm zugewiesene ID aus einer ID-Datei in dem Folder lesen, in dem es arbeitet, und diese ID zu den gesendeten Daten hinzufügen.	Config
• **/MBS-4/** Das Modul liest die Einstellungen aus einer Konfigurationsdatei, die die erforderlichen Einstellungen enthält. Kommunikation usw. lernt die notwendigen Informationen aus dieser Datei.	Config
• **/DA-1/** Die Desktop App muss sich in regelmäßigen Abständen mit MBS verbinden und Daten auslesen können.	Recieve
• **/DA-2/** Die Desktop App sollte die gelesenen Daten in der Datenbank speichern.	WriteData
• **/DA-3/** Optimale Werte für die Topfpflanze sollten mit dem Desktop-APP-Programm in der Datenbank erfasst und bearbeitet werden.	ReadOptimalData, WriteOptimalData

• **/DA-4/** Die Desktop App soll die folgenden Situationen prüfen, indem sie die aktuell in die Datenbank hochgeladenen Daten mit den optimalen Daten vergleicht und die für die relevante Situation ermittelte Meldung auf dem Bildschirm ausgibt.
  ○ Ist das Umgebungslicht im Bereich, den es haben sollte?
  ○ Ist die Erde der Pflanze feucht genug?
  ○ Ist der pH-Wert des Bodens angemessen?
	CheckPH, CheckLight, CheckWater, SendMessagetoUser

• **/DA-5/** Die Desktop App sollte in der Lage sein, die historischen Informationen aus der Datenbank zu lesen und sie dem Benutzer auf Anfrage anzuzeigen.	ReadAllData
• **/DA-6/** Eine Desktop-Anwendung sollte in der Lage sein, mit mehr als einem MBS zu kommunizieren.	SwitchPlant
• **/DB-1/** Die Daten werden in einer externen SQL-Datenbank gespeichert.	WriteData
• **/DB-2/** Werte von MBS und gewünschte optimale Werte werden in zwei getrennten Tabellen gehalten.	Database Layer
• **/DB-3/** Daten sollten zwischen dem Modul und der Anwendung in Paketen übertragen werden, die ID, Licht, Feuchtigkeit und pH enthalten.	Send, Receive

5. Beschreibung der DB-Zugriffsschicht(Daten-Modelle)

Die Datenbank besteht aus zwei Tabellen, in denen die tatsächlichen Werte und die optimalen Werte gespeichert werden. Auf die Realwertetabelle wird von den Funktionen WriteData, ReadAllData und ReadLastData aus der Datenbankschicht des DA zugegriffen. Die Funktionen WriteOptimalData und ReadOptimalData greifen auf die Optimalwertetabelle aus der Datenbankschicht des DA zu.



.