

**Technisches Konzept**

**Projektname: FISI-Wetterstation**

**Version:** 0.1

**Status:** Entwurf

**Datum:** 07.07.2017

**Autor: Kruppa Alexander**

Inhaltsverzeichnis

[Inhaltsverzeichnis 2](#_Toc487621437)

[1. Einführung und Ziele / Fachliche Anforderungen 3](#_Toc487621438)

[1.1 Aufgabenstellung 3](#_Toc487621439)

[1.2 Qualitätsziele 3](#_Toc487621440)

[1.3 Stakeholder 3](#_Toc487621441)

[2. Randbedingungen 4](#_Toc487621442)

[2.1 Technische Randbedingungen 4](#_Toc487621443)

[3. Kontext 4](#_Toc487621444)

[3.1 Externe Schnittstellen 5](#_Toc487621445)

[3.1.1 Externe Schnittstelle 1 5](#_Toc487621446)

[3.1.2 Externe Schnittstelle 1 5](#_Toc487621447)

[4. Lösungsstrategie 6](#_Toc487621448)

[5. Bausteinsicht 6](#_Toc487621449)

[5.1 Ebene 1 7](#_Toc487621450)

[5.1.1 Anwendung (Whitebox-Beschreibung) 7](#_Toc487621451)

[6. Laufzeitsicht 8](#_Toc487621452)

[7. Datensicht 9](#_Toc487621453)

[8. Verteilungssicht 10](#_Toc487621454)

[9. Konzepte 11](#_Toc487621455)

[9.1 Frameworks und Entwurfsmuster 11](#_Toc487621456)

[9.1.1 Frameworks 11](#_Toc487621457)

[9.2 Persistenz 11](#_Toc487621458)

[9.3 Benutzungsoberfläche 11](#_Toc487621459)

[10. Qualitätsszenarien 11](#_Toc487621460)

[10.1 Bewertungsszenarien 11](#_Toc487621461)

[11. Glossar 12](#_Toc487621462)

# Einführung und Ziele / Fachliche Anforderungen

Zum 14.07.2017 wird ein den Anwenderansprüchen, in folgendem Dokument definiert, konformes System übergeben mit dem er die Daten aus der Wetterstation in Echtzeit und mit einem Graphen über einen Definierten Zeitraum einsehen kann.

## Aufgabenstellung

Analyse der mitgebrachten Hardware des Kunden und Darstellung der Möglichen Funktionsweisen.

Konzeption einer Vorgehensweise zur bewälltigung der Angeforderten Funktionen.

Erstellung des Codes im Backend des Systems zur Abrufung, Verarbeitung und Speicherung der gelieferten Daten aus der Wetterstation.

Designen und Programmieren der Anwendungsoberfläche.

Integration der Daten aus der SQL Datenbank in die Anwendung.

## Qualitätsziele

* Echtzeitansicht der gelieferten Daten aus der Wetterstation
* Einstellmöglichkeit der Warntemperaturen, Aktualisierungsrate und Einheit in der die Temperatur dargestellt wird.
* Speicherung der Daten 1x pro Minute
* Selbsterklärendes Design der Desktopanwendung. Keine Anleitung benötigt.
* Aktualisierung der Graphen in Anwenderdefinierten abständen.

## Stakeholder

Die folgende Tabelle führt Stakeholder auf, die in Projekten relevant sein könn(t)en.

|  |  |  |
| --- | --- | --- |
| **Name** | **Rolle** |  |
| Tajedini Behnam,  Dr.-Ing. Jeschke Matthias | Auftraggeber, Anwender | Initiatoren und Auftraggeber des Projektes. Abnehmer des Fachkonzepts. Abnehmer des fertigen Projekts. Ansprechpartner für Fragen an Kunden |
| Kruppa Alexander | Projektleiter | Ansprechpartner für Fragen der Projektmitglieder. Verteiler der Arbeitspakete. Ansprechpartner der Auftraggeber. Vertreter des Projekts nach außen. |
| Bartelsen Luca  Koppehele Sascha  Ungar Michael | Projektmitglieder | Mitarbeiter im Projekt. Zuständig und Verantwortlich für die Ihnen zugeteilten Arbeitspakete. |

# Randbedingungen

Vollendung bis zum 14.07.2017.

Nutzung der Unten genannten Systeme.

## Technische Randbedingungen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Hardware-Vorgaben | | | | |
| Wetterstation | | Tinkerforge Wettersation Starterkit mit Raspberry Pi | |
| Anwender PC | | | PC mit WLAN Zugang im selben Netz wie die Wetterstation | |
| Software-Vorgaben | | | | |
| Betriebssystem | | | Windows 7 | |
| Programmiervorgaben | | | | |
| Sprache | Java, PostgreSQL | | | |

# Kontext

Informationen über die aktuelle Helligkeit, Temperatur, Luftdruck und Luftfeuchtigkeit werden über drei Sensoren in der Wetterstation erhoben.

Diese Daten werden auf einem LCD-Display auf der Wetterstation und auf einer Desktopanwendung in Echtzeit angezeigt.

In der Desktopanwendung lassen sich ebenfalls Graphen über den Verlauf der Daten über die letzten 24 Stunden und über den heutigen Tag anzeigen.

Die Daten, welche in vom Anwender definierten Abständen in eine lokale SQL Datenbank geschrieben werden, speisen die Graphen mit Informationen.

Die Wetterstation ist über ein USB-Kabel an einen Raspberry Pi angeschlossen welcher die Daten per WLAN verteilt.

Der Abruf der Daten der Desktopanwendung erfolgt über JavaJDBC aus der lokalen SQL Datenbank

.

## Externe Schnittstellen

### Externe Schnittstelle 1

#### Identifikation der Schnittstelle

|  |  |
| --- | --- |
| Name / Bezeichnung der Schnittstelle | Wetterstation |
| Version |  |
| Typ der Schnittstelle | Protokollorientiert |
| Anforderungen an die Schnittstelle | Lieferung der Wetterdaten |
| Ablauf der Schnittstelle | Senden der Daten von der Wetterstation über den Raspberry Pi an den PC |

### Externe Schnittstelle 1

#### Identifikation der Schnittstelle

|  |  |
| --- | --- |
| Name / Bezeichnung der Schnittstelle | Datenbank |
| Version |  |
| Typ der Schnittstelle | Protokollorientiert |
| Anforderungen an die Schnittstelle | Speicherung der Wetterdaten und bereitstellung der Daten zum Abruf von der Anwendung |
| Ablauf der Schnittstelle | Gelieferte Daten werden in der Datenbank abgespeichert |

# Lösungsstrategie

Nach einer Analyse der Möglichkeiten der Wetterstation wurde mit Hilfe der Dokumentation des Geräts erarbeitet, wie die Daten in eine SQL Datenbank geschrieben werden können und auf dem Display dargestellt werden können.

Die Codierung des Backends wird von Luca Bartelsen durchgeführt. Das Frontend wird designt und programmiert von Sascha Koppehele. Die Schnittschtelle zwischen dem Frontend und Backend wird erarbeitet von Michael Ungar und in Zusammenarbeit mit Herrn Bartelsen und Herrn Koppehele implementiert. Herr Kruppa übernimmt die Arbeitseinteilung und Projektleitung. In Personalausfällen ist er als Springer verfügbar.

# Bausteinsicht

Aus der Wetterstation werden numerische Werte Ausgegeben welche über den Raspberry Pi per WLAN vom Anwender PC abgerufen werden. Auf dem PC werden diese Daten Ausgegeben und in eine SQL Datenbank abgespeichert.

In Der Wetterstation erheben drei Sensoren vier Messwerte:

* Ambient Light Bricklet 2.0 -> Helligkeit
* Humidity Bricklet -> Luftfeuchtigkeit
* Barometer Bricklet -> Luftdruck und Temperatur

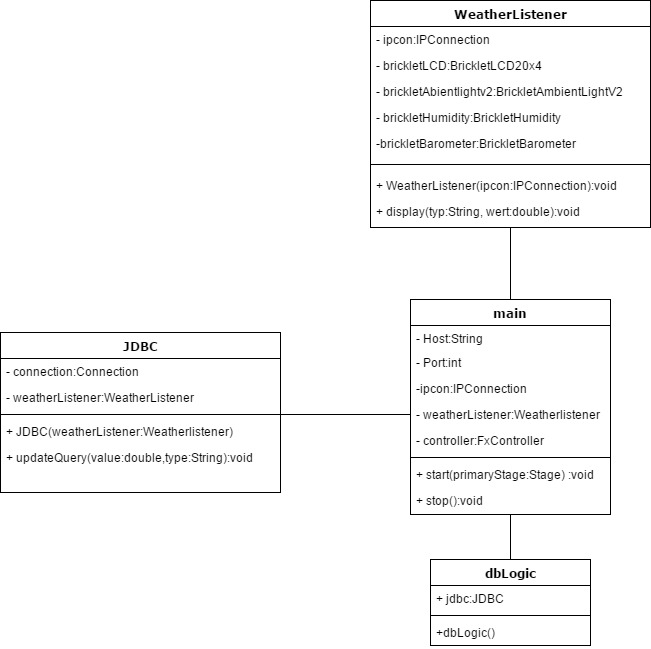
Die Messwerte werden auf dem eingebauten LCD-Display ausgegeben und an den Raspberry Pi gesendet.

In 1 Minuten abständen werden die Daten von dem Raspberry Pi an den PC gesendet.

## Ebene 1

### 

### Anwendung (Whitebox-Beschreibung)



# Laufzeitsicht

Messdaten

Per WLAN

Anwender PC

Messdaten

Per USB

Raspberry Pi

Wetterstation

Liest Daten ein

Frontend

SQL-Datenbank

Backend

Liest Daten ein

Und gibt aus

Speichert

Daten

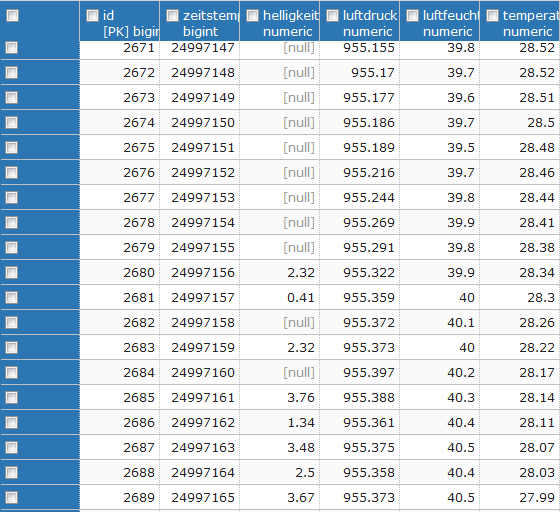
Sendet Daten und gibt auf Wetterstation aus

Wetterstation

# Datensicht

Es werden 1x pro Minute alle vier Werte: Temperatur, Luftdruck, Luftfeuchtigkeit und Helligkeit von der Wetterstation abgerufen und in einer SQL Datenbank auf dem Anwender PC gespeichert.

Die Daten bleiben in der Datenbank gespeichert und werden nicht gelöscht.



# Verteilungssicht

# Konzepte

## Frameworks und Entwurfsmuster

### Frameworks

* Java Runtime Environment
* PostGreSql

## Persistenz

Flüchtige Daten aus der Wetterstation werden Dauerhaft auf der Festplatte des PC’s in einer SQL Datenbank gespeichert

## Benutzungsoberfläche

Graphische Windows Anwendung.

Oberfläche nicht konfigurierbar.

# Qualitätsszenarien

## Bewertungsszenarien

* Auslöser (*stimulus*): Ein Benutzer startet seine Desktop Anwendung
* Quelle des Auslösers (*source*): Benutzer
* Umgebung (*environment*): System läuft unter Normallast. Datenbank ist erreichbar da sie zentral auf dem Anwender PC ist.
* Systembestandteil (*artifact*): Anwender PC,Raspberry Pi
* Antwort (*response*): Ab dem Start der Anwendung werden nun in ein Minuten Abständen die Messdaten vom Anwender PC vom Raspberry Pi abgerufen.
* Antwortmetrik (*response measure*): Die 4 Messdaten werden korrekt abgerufen und in die Datenbank gespeichert.

# 

# Glossar

Inhalt

Die wichtigsten Begriffe der Software-Architektur in alphabetischer Reihenfolge

|  |  |
| --- | --- |
| Begriff | Definition |
|  |  |
|  |  |