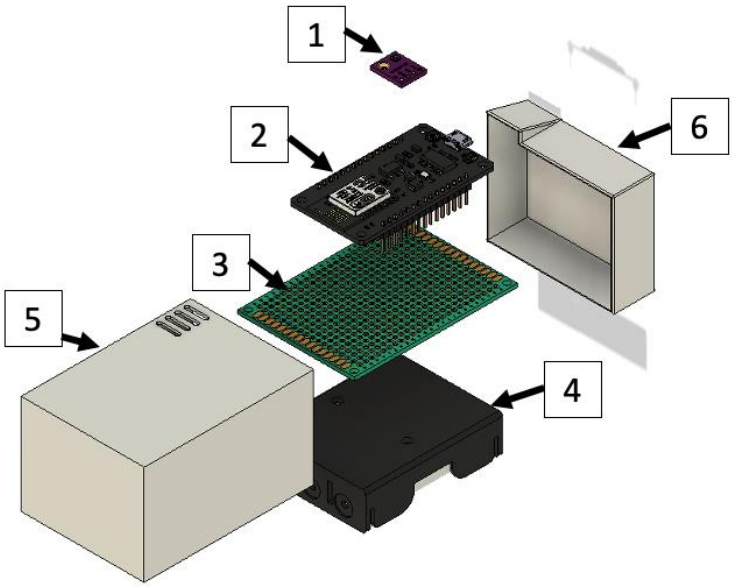
FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Wetterstation

Hausarbeit

im Modul 73116 –Programmierung und Informationsverarbeitung



vorgelegt von

David Wergen	*Mat.-Nr. 3115266*	*Industrial Engineering*
Matthias Simons	*Mat.-Nr. 3104576*	*Industrial Engineering*
Fabian Meyer	*Mat.-Nr. 3125420*	*Industrial Engineering*

Januar 2021

Inhaltsverzeichnis

Abbildungsverzeichnis

Projektinitialisierung

Problemstellung

Heutzutage verlassen wir uns blind auf die Wettervorhersage aus dem Internet oder dem Fernsehprogramm. Wie präzise diese Vorhersage letztlich ist, wird von kaum jemanden nachvollzogen.

Daher soll im Rahmen dieses Projektes eine Wetterdatenaufzeichnung vorgenommen werden und mit vorhergesagten Daten verglichen werden.

Zieldefinition

Über einen vordefinierten Zeitraum sollen lokale Wetterdaten erhoben werden.

Da das Projektteam aus drei Mitgliedern besteht soll an jedem Wohnort eines Mitgliedes des Projekteams eine Erhebung der Wetterdaten stattfinden (vgl. Abbildung 1 Standorte der Wetterstationen). Bei den aufzuzeichnenden Wetterdaten handelt es sich um die Temperatur, die Luftfeuchtigkeit und den Luftdruck. Diese Parameter sollen mit den entsprechenden Vorhersagen von lokal nahgelegenen Wetterstationen verglichen werden.

Als Ergebnis dieses Projektes sollen in verschiedenen Abbildungen die generierten Messwerte mit den vorhersagten Daten über einen bestimmten Zeitraum verglichen werden und eine mittlere Temperaturabweichung berechnet werden.

Projektteam

David Wergen, Wohnort: Monschau-Mützenich
Matthias Simons, Wohnort: Euskirchen
Fabian Meyer, Wohnort: Nideggen-Schmidt

Abbildung 1 Standorte der Wetterstationen

Entwicklung

Die Entwicklung des Gesamtsystems gliedert sich in die Entwicklung der Wetterstation mit der dazugehörigen Software und dem Programm zur Datenverarbeitung.

Entwicklung Wetterstation

Zur strukturierten Entwicklung der Wetterstation wird zunächst eine Funktionsstruktur erstellt. Auf Basis dieser Funktionsstruktur werden die Anforderungen an die Wetterstation abgeleitet, auf dessen Basis später die Lösungsfindung erfolgen kann.

Funktionsstruktur der Wetterstation

Die in Abbildung 2 dargestellte Funktionsstruktur soll aufzeigen, welche Kernfunktionen unsere Wetterstation erfüllen muss.

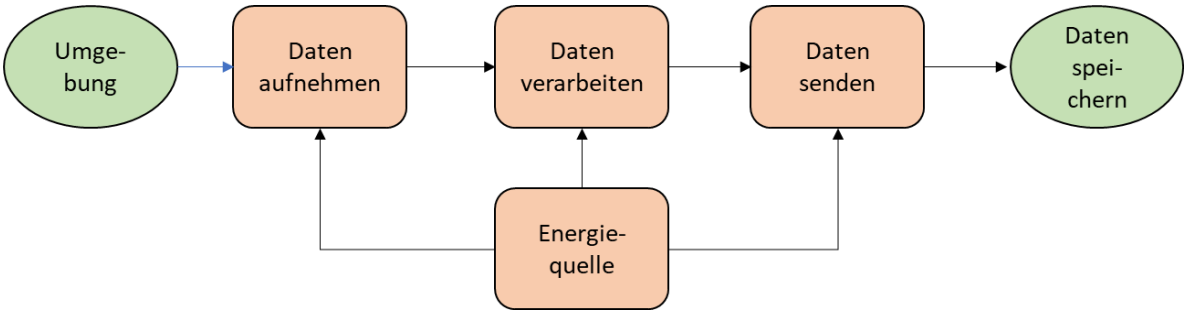


Abbildung 2 Funktionsstruktur

Zunächst muss die Wetterstation in die Umgebung eingebunden werden. Dies ist keine Funktion aber zum besseren Verständnis hier mit aufgeführt. Aus der Umgebung muss die Wetterstation Daten aufnehmen. Anschließend müssen diese verarbeitet und zum Speichern gesendet werden. Alle diese Funktionen werden zum Durchführen der Prozesse von einer Energiequelle versorgt. Am Ende ist die Datensicherung zu finden. Auch diese ist keine Funktion der Wetterstation aber ebenfalls zum besseren Verständnis angeführt.

Anforderungen an die Wetterstation

Aus der Funktionsstruktur der Wetterstation ergeben sich Anforderungen, die bei der Entwicklung berücksichtigt werden müssen. Die Anforderungen werden in der Anforderungsliste (Abbildung 3) gesammelt.

Anforderungsliste: Wetterstation						
Nr.	Beschreibung	Bezeichnung	Zahlenwerte			Einheit
			min	exakt	max	
1	Datenaufnahme	Temperatur-, Luftfeuchtigkeit und Luftdrucksensor		1		Stück
2	Datenverarbeitung	Microcontroller				
3	Datenübertragung	Netzwerkverbindung (Wlan)				
4	Datensicherung	Datenbank (Cloud)				
5	Schnittstellen	Kommunikation zwischen Microcontroller und Sensor				
6	Energieversorgung	Autark	14			Tage
7	Energieverbrauch	möglichst gering				
8	Schutz vor Umwelteinfluss	Gehäuse		1		Stück

Abbildung 3 Anforderungsliste Wetterstation

Die wichtigsten Anforderungen an die Wetterstation sind die Nutzung von Sensoren zur Aufnahme von Temperatur, Luftfeuchtigkeit und Luftdruck. Des Weiteren soll zur Verarbeitung der Messwerte ein Microcontroller genutzt werden, der fähig ist die Daten über eine Netzwerkverbindung zu übertragen. Die Datensicherung soll über eine Cloud erfolgen, damit die Daten jederzeit und von überall zugängig sind. Aufgrund der Außennutzung muss das empfindliche Messsystem durch ein Gehäuse vor Umwelteinflüssen geschützt werden und mindestens für 14 Tage autark, ohne externe Stromversorgung Messungen aufnehmen können.

Entwicklung der Software für den Microcontroller

Für die Strukturierte Entwicklung der Software der Wetterstation, wurde der Programmablauf in einem BPMN Modell dargestellt. Es beschreibt die zeitliche Abfolge der Prozesse, sowie die Ereignisse zu Beginn und zum Abschluss des Prozesses.

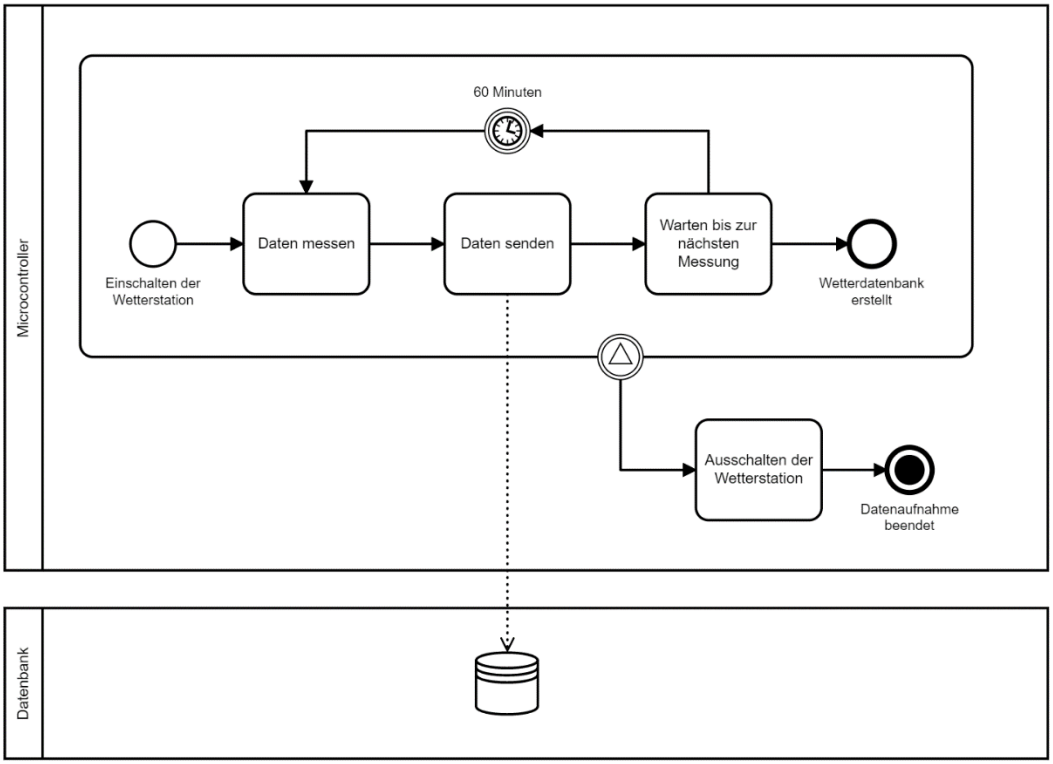


Abbildung 4 BPMN-Model der Wetterstation

Der Prozess beginnt mit dem Einschalten der Wetterstation. Nach dem Einschalten werden die Daten (Temperatur, Luftfeuchtigkeit, Umgebungsdruck) gemessen werden. Anschließend werden die gemessenen Daten in eine Datenbank gesendet. Nach dem Senden der Daten wartet der Prozess 60 min auf eine erneute Messung. Tritt der Fall ein, dass die Wetterstation ausgeschaltet wird, wird der gesamte Prozess gestoppt und die Datenaufnahme ist beendet.

Zur Auswertung der gemessenen Daten soll ein Programm entwickelt werden, dass die Messdaten aus der Datenbank ausliest. Zusätzlich sollen prognostizierte Daten aus einer Datenbank ausgelesen werden. Die Daten sollen verarbeitet und graphisch visualisiert werden. Die Programmstruktur und der Ablauf ist schematisch in Abbildung 5 in Form eines BPMN Modells dargestellt. Die aus der Funktionsstruktur resultierenden Anforderungen an die Datenverarbeitung sind in der Anforderungsliste Abbildung 6 dargestellt.

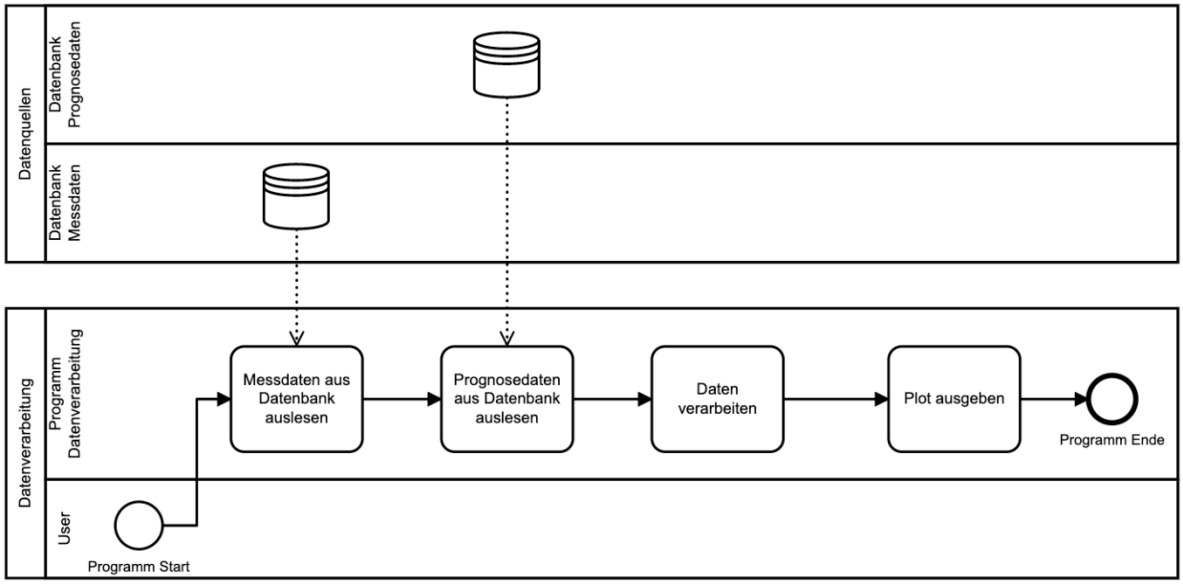


Abbildung 5 Funktionsstruktur der Datenverarbeitung

Anforderungsliste: Datenverarbeitung		
Nr.	Beschreibung	Bezeichnung
1	Prognosedaten	Nutzung von öffentlichen Archiven
2	Datenabfrage	Automatisierte Abfrage aus Datenbank
3	Datenverarbeitung	Ausgabe von visualisierten Daten
4	Programmiersprache	Python

Abbildung 6 Anforderungsliste an die Datenverarbeitung

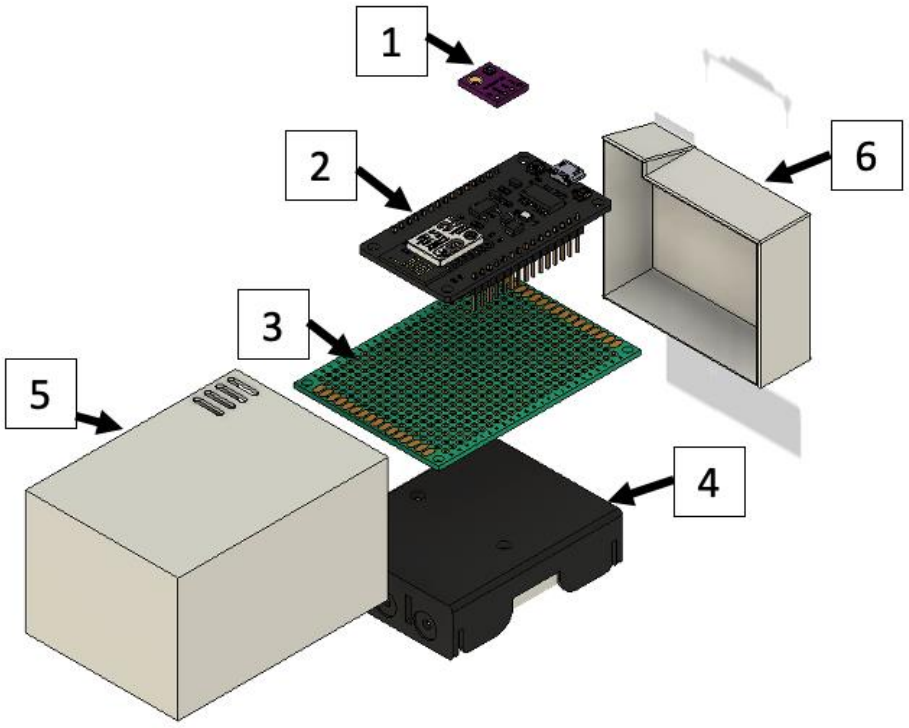
Die wichtigsten Anforderungen an die Datenverarbeitung sind, dass für die Prognosedaten öffentliche Datenbank Archive genutzt werden und die Abfrage automatisiert abläuft. Als Programmiersprache ist Python vorgegeben. Wichtig ist das eine ältere Python Version als 3.9 verwendet wird (am besten Version 3.8.7), da ansonsten die Meteostat Library (siehe Python Skript) nicht verwendet werden kann.

Lösungskonzept und Umsetzung

Auf Basis der benötigten Teilfunktionen der Wetterstation und den resultierenden Entwicklungsanforderungen wurden zunächst Lösungsmöglichkeiten gesammelt. Aus den gesammelten Lösungsmöglichkeiten wurde ein Lösungskonzept zur Erfüllung der Aufgabenstellung erstellt. Das Lösungskonzept orientiert sich an dem Ziel möglichst einfach und kostengünstig umsetzbar zu sein. Die Umsetzung gelang als kompaktes System mit elektro- und informationstechnischen Elementen.

Wetterstation

Die Wetterstation mit ihren einzelnen Komponenten und deren Anordnung sind in Abbildung 7 dargestellt.



Pos. Nr.	Benennung	Beschreibung	Menge	Material
1	BME 280	Sensor	1	-
2	ESP 8266	Mikro-Controller	1	-
3	Platine 5x7	Lochrasterplatine	1	-
4	Batteriehalter	-	1	-
5	Gehäuse	-	1	PLA
6	Deckel	-	1	PLA

Abbildung 7 Explosionszeichnung (oben) und Stückliste (unten) der Wetterstation

Es wurde sich für einen Sensor des Typs „BME 280“ (Pos. Nr. 1) entschieden, da dieser in kompakter Bauweise Temperatur, Luftfeuchtigkeit und Luftdruck messen kann. Als

Microcontroller wurde ein „ESP 8266“ (Pos. Nr. 2) gewählt, da dieser eine einfache Datenübertragung an eine Datenbank durch die Netzwerkverbindung über WLAN ermöglicht. Die Nutzung des Batteriehalters (Pos. Nr. 4) ermöglicht einen autarken Betrieb ohne externe Stromversorgung. Zum Schutz der empfindlichen Sensorik wurde ein Gehäuse mit Deckel (Pos. Nr. 5 & 6) konstruiert, damit das System vor Umwelteinflüssen wie z.B. Feuchtigkeit geschützt ist. Das Gehäuse wird im additiven Fertigungsverfahren „FLM“ hergestellt. Somit erfüllt das System Wetterstation die gestellten Anforderungen (Vergleich Abbildung 3).

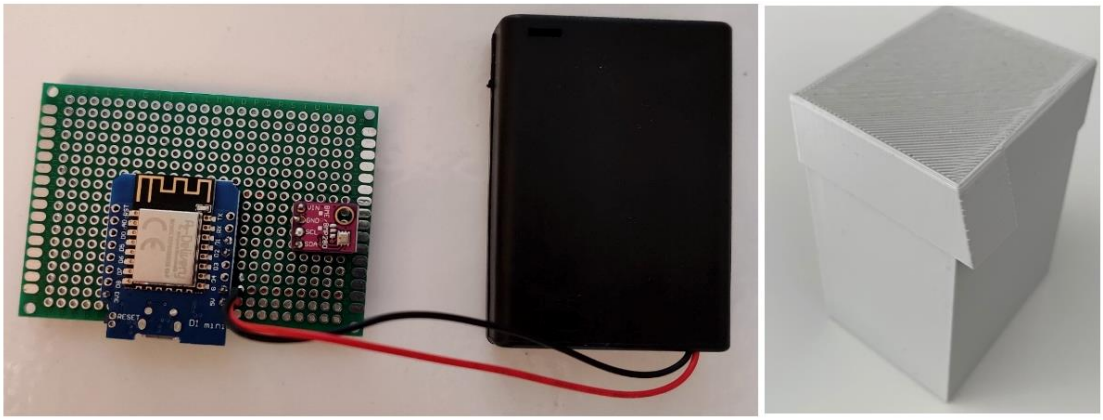


Abbildung 8 Anordnung der elektrotechnischen Komponenten auf der Lochrasterplatine (links) und Gehäuse (rechts)

Die Anordnung der elektrotechnischen Komponenten erfolgt, wie in Abbildung 8 zu erkennen, auf einer Lochrasterplatine (Pos. Nr. 3). Ebenfalls zu erkennen ist das additiv gefertigte Gehäuse.

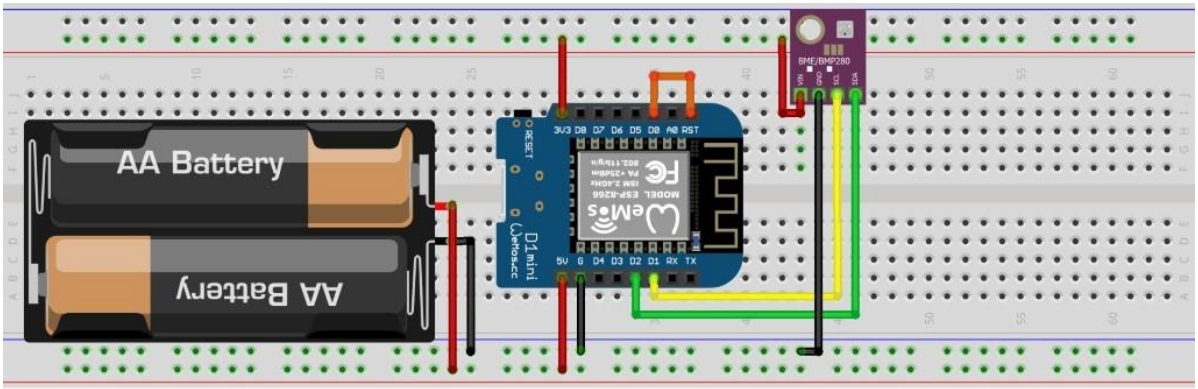


Abbildung 9 Schaltplan

Der Schaltplan der elektrotechnischen Komponenten ist in Abbildung 9 dargestellt. Die Batteriehalterung mit Platz für zwei AA Batterien dient nur der Visualisierung, in Realität wurde eine Batteriehalterung für drei AA Batterien verwendet.

Die Programmierung des Microcontrollers „ESP 8266“ basiert auf der Programmiersprache Arduino/C++. Durch sogenanntes „flashen“ des Microcontrollers ist es möglich die Programmiersprache Micropython zu verwenden. Die wichtigsten Elemente des Programms werden im Folgenden kurz beschrieben. Das komplette Programm befindet sich im Anhang unter „boot.py“.

Abbildung 10 Programmablauf Microcontroller

```
44  while True:
45    try:
46      #Messwerte auslesen
47      bme = BME280.BME280(i2c=i2c)
48      temp = bme.temperature
49      hum = bme.humidity
50      pres = bme.pressure
51      #Messwerte in Dictionary speichern
52      sensore_readings = {"value1": temp, "value2": hum, "value3": pres}
53
54      #mit IFTTT API verbinden und Request mit Messwerten senden
55      request_headers = {"Content-Type": "application/json"}
56      request = urequests.post(
57      "http://maker.ifttt.com/trigger/"+ort+"/with/key/"+api_key,
58      json=sensore_readings,
59      headers = request_headers)
60      request.close()
61
62      # Deep Sleep
63      rtc = machine.RTC()
64      rtc.irq(trigger=rtc.ALARM0, wake=machine.DEEPSLEEP)
65
66      if machine.reset_cause() == machine.DEEPSLEEP_RESET:
67
68        print('woke from a deep sleep')
69      rtc.alarm(rtc.ALARM0, 3600000)
70      machine.deepsleep()
71
72    except OSError as e:
73      print("Failed to read/publish sensor readings.")
```

Abbildung 11 Ausschnitt aus dem Programm boot.py

Der Programmablauf ist in Abbildung 10 dargestellt. Einzelne Funktionen sollen mit Abbildung 11 verdeutlicht werden. Dort ist ein Ausschnitt aus dem Programm „boot.py“ zu sehen. Zunächst wird eine Netzwerkverbindung mit dem Microcontroller über WLAN hergestellt. Daraufhin wird mit einer „while“ Schleife (Siehe Abbildung 11) der Sensor ausgelesen (Zeile 47-50), eine Verbindung zur IFTTT-API hergestellt und die Messwerte werden an den Server

gesendet (Zeile 55-60) und mit der IFTTT-API in die Datenbank in der Cloud geschrieben. Der Sensor wird anschließend in den sogenannten „Deepsleep“ versetzt (Zeile 63-70). Der „Deepsleep“ dient dazu den Energieverbrauch der Wetterstation zu minimieren in dem der Sensor zwischen den Messungen in einen Ruhezustand (Dauer: 60 Minuten) versetzt wird.

Die IFTT-API dient als Schnittstelle zwischen Microcontroller und der Datenbank. IFTTT ist ein Dienstanbieter, mit dem es möglich ist, verschiedene Anwendungen mit bedingten Verknüpfungen miteinander zu verbinden. Die Anweisungen erfolgen nach dem „if this then that“ Prinzip. Der „this“ Teil ist der sogenannte Trigger und der „that“ Teil die Aktion. In diesem Fall ist der Trigger der „request“ und die Aktion das Schreiben der Messwerte (sensore_readings) in eine Google Sheets Datei. Google Sheets dient dabei als Datenbank, die aus Tabellen für jeden Ort besteht (Abbildung 12). Als Cloud wird Google Drive genutzt.

	A	B	C	D	E
1	Date	Location	Temperatur	Humidity	Pressure
2	December 30, 2020 at 01:12AM	Euskirchen	5.67C	69.30%	1009.50hPa
3	December 30, 2020 at 02:12AM	Euskirchen	4.36C	70.58%	1009.94hPa
4	December 30, 2020 at 03:12AM	Euskirchen	4.19C	80.19%	1012.75hPa
5	December 30, 2020 at 04:12AM	Euskirchen	2.87C	78.07%	1012.52hPa
6	December 30, 2020 at 05:12AM	Euskirchen	3.17C	75.33%	1011.66hPa
7	December 30, 2020 at 06:12AM	Euskirchen	3.71C	73.45%	1012.10hPa
8	December 30, 2020 at 07:12AM	Euskirchen	3.78C	74.09%	1013.44hPa
9	December 30, 2020 at 08:12AM	Euskirchen	3.39C	74.45%	1014.39hPa
10	December 30, 2020 at 09:12AM	Euskirchen	3.24C	75.67%	1016.12hPa
11	December 30, 2020 at 10:12AM	Euskirchen	2.85C	70.70%	1015.53hPa
12	December 30, 2020 at 11:12AM	Euskirchen	3.39C	65.17%	1014.12hPa
13	December 30, 2020 at 12:12PM	Euskirchen	4.56C	63.16%	1012.83hPa
14	December 30, 2020 at 01:13PM	Euskirchen	5.45C	58.55%	1011.43hPa

Abbildung 12 Ausschnitt der Google Sheets Datei

Aufgrund des Umfangs des Programms wird die Funktionsweise mithilfe von Abbildung 13 dargestellt und wichtige Programmbausteine werden im Folgenden, mit Ausschnitten aus dem „main.py“ Programm erklärt. Das komplette Programm befindet sich im Anhang.

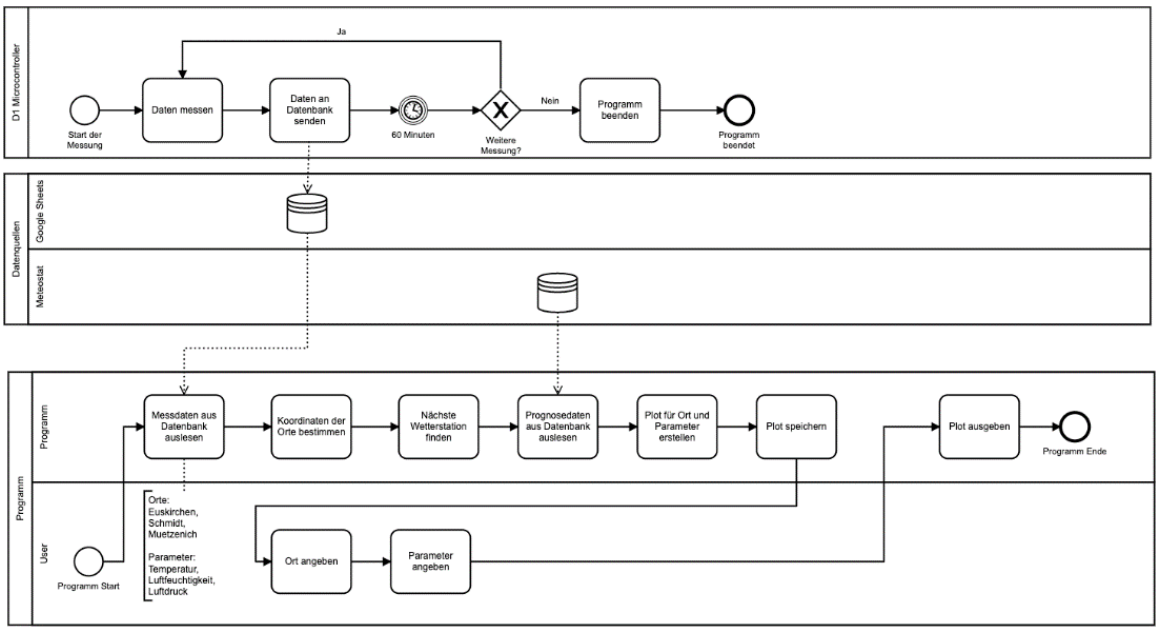


Abbildung 13 Funktionsweise und Programmablauf in Anlehnung an BPMN

Messdaten aus Datenbank auslesen

Im vorherigen Kapitel wurde beschrieben, dass die Messwerte in eine Google Sheets Datei geschrieben werden. Um die Datei auszulesen ist es notwendig die Google Sheets API zu nutzen.

```
13 scope = ["https://spreadsheets.google.com/feeds",
14          'https://www.googleapis.com/auth/spreadsheets',
15          "https://www.googleapis.com/auth/drive.file",
16          "https://www.googleapis.com/auth/drive"]
17
18 creds = ServiceAccountCredentials.from_json_keyfile_name("WetterstationIT.json", scope)
19 client = gspread.authorize(creds)
```

Abbildung 14 Initialisierung der Google Sheets API

```python
#gemessene Daten auslesen
def get_measurement_data(location):
    sheet = client.open(location).sheet1  # Open the spreadhseet
    data = sheet.get_all_records()  # Get a list of all records
    data = pd.DataFrame(data)
    data["Date"] = pd.to_datetime(data["Date"])
    data = data.set_index(keys = data["Date"])
    data = data[["Temperatur", "Humidity", "Pressure"]]
    data = data.rename(columns={"Temperatur": temp, "Humidity": rhum, "Pressure": pres})
    # clear data from units
    data[temp] = data[temp].str.strip("C")
    data[rhum] = data[rhum].str.strip("%")
    data[pres] = data[pres].str.strip("hPa")
    data = data.astype("float")
    return data
```

Abbildung 15 Auslesen der Messwerte aus Google Sheets

Zunächst muss die Google Sheets API über Google Developers initialisiert werden (Abbildung 14). Anschließend kann eine Verbindung hergestellt werden. Mit der Funktion get_measurement_data (Abbildung 15) wird die Google Sheets Datei ausgelesen und die Daten in ein Pandas Dataframe geschrieben, bereinigt und ausgegeben.

Prognosedaten aus Datenbank auslesen

Die Prognosedaten stammen aus der Klima- und Wetterdatenbanken Meteostat auf die über die Meteostat API zugegriffen werden kann. Um die Wetterdaten auszulesen sind mehrere Schritte erforderlich.

```python
def get_coordinates(location):
    geolocator = Nominatim(user_agent="Wetterstation")
    location = geolocator.geocode(location)
    coordinates = (location.latitude, location.longitude)
    return coordinates
```

Abbildung 16 Funktion get_coordinates

Zunächst werden vom Ort die Koordinaten bestimmt. Dies geschieht durch das Modul „geopy.geolocator“ mit dem es Möglich ist sich die Koordinaten von Orten und Adressen ausgeben zu lassen (Abbildung 16).

```python
#nächste Wetterstation von Koordinaten
def get_closest_weather_station(lat, lon, start, end):
    stations = Stations()
    stations = stations.nearby(lat, lon)
    stations = stations.inventory('hourly', (start, end))
    station = stations.fetch(1)
    return station
```

Abbildung 17 Funktion get_closest_weather_station

Mit der Longitude und Latidude der Koordinaten und dem gewünschten Zeitraum kann nun die nächste Wetterstation bestimmt werden (Abbildung 17).

```
51 #Prognostizierte Daten auslesen und zeitlich eingrenzen
52 def get_prognosed_data(station, start, end):
53     data = Hourly(station, start=start, end=end)
54     data = data.normalize()
55     data = data.interpolate()
56     data = data.fetch()
57     data = data[["temp", "rhum", "pres"]]
58     data = data.rename(columns={"temp": temp, "rhum": rhum, "pres": pres})
59     data = data.loc[data.index >= start]
60     data = data.loc[data.index <= end]
61     return data
```

Abbildung 18 Funktion get_prognosed_data

Anschließend werden die Wetterdaten für die nächste Wetterstation und dem gewünschten Zeitraum ausgelesen, bereinigt und ausgegeben.

Plot für Ort und Parameter erstellen

Mit den vorher beschriebenen Funktionen werden für jeden Ort und jeden Parameter die prognostizierten und gemessenen Daten visualisiert und gespeichert. Die Visualisierung erfolgt über das Modul Matplotlib.pyplot.

Plot ausgeben

Um die in der vorherigen Funktion gespeicherten Plots anzuzeigen, wird der User über die Input Funktion aufgefordert den gewünschten Ort und Parameter einzugeben. Anschließend wird der gewünschte Plot über das Modul Matplotlib.image angezeigt.

Ergebnisse und Auswertung

Das Ergebnis des Entwicklungsprojektes ist eine Wetterstation, die in der Lage ist Temperatur- Luftfeuchtigkeits- und Luftdruckmessungen vorzunehmen. Des Weiteren wurde eine Datenverarbeitungssoftware entwickelt, die mehrere Datenbanken ausliest. Zum einen wird eine Datenbank, in der die durch die Wetterstation aufgenommenen Messwerte gespeichert sind und zum anderen eine Datenbank mit öffentlichen Klima- und Wetterdaten ausgelesen. Diese Daten werden anschließend durch die Software bereinigt und visualisiert. Die Bedienung des Programms und die Ergebnisse der Visualisierung werden im Folgenden vorgestellt.

Bedienung

Das Programm „main.py“ wird (empfohlen als Jupyter Notebook) geöffnet und ausgeführt. Der Bediener wird aufgefordert eine Eingabe zum gewünschten Ort (Euskirchen, Nideggen-Schmidt oder Mützenich) und zum gewünschten Parameter (Temperatur, Luftfeuchtigkeit und Luftdruck) zu treffen. Hierbei ist darauf zu achten, dass die Abfrage exakt wie angezeigt (sprich mit Einheiten) eingegeben wird. Daraufhin wird ein Plot zum gewünschten Ort und Parameter ausgegeben. Alle Plots werden in dem Ordner „figures“, der automatisch bei erstmaliger Ausführung des Programms im selben Pfad erstellt wird, gespeichert.

Ergebnisse

Die Ergebnisse der Visualisierung werden nachfolgend am Beispiel des Standortes Euskirchen vorgestellt (Stand 15.01.2021).

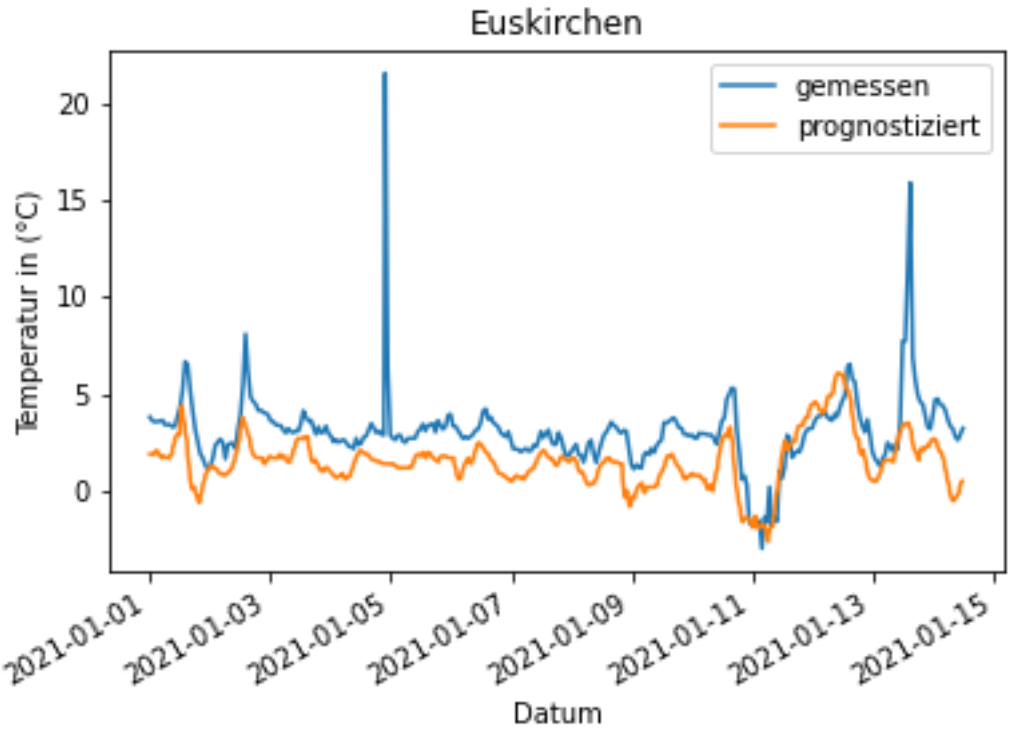


Abbildung 19 Verlauf der Temperatur am Standort Euskirchen

In Abbildung 19 ist der Temperaturverlauf am Standort Euskirchen dargestellt. Es ist zu erkennen, dass der Temperaturverlauf der gemessenen und prognostizierten Daten einen ähnlichen Verlauf vorweist. Allerdings ist der gemessene Temperaturverlauf vertikal und horizontal leicht verschoben. Am 05.01.2021 und am 13.01.2021 sind Ausreißer zu erkennen. Diese Ergebnisse sind vermutlich durch Messfehler verschuldet.

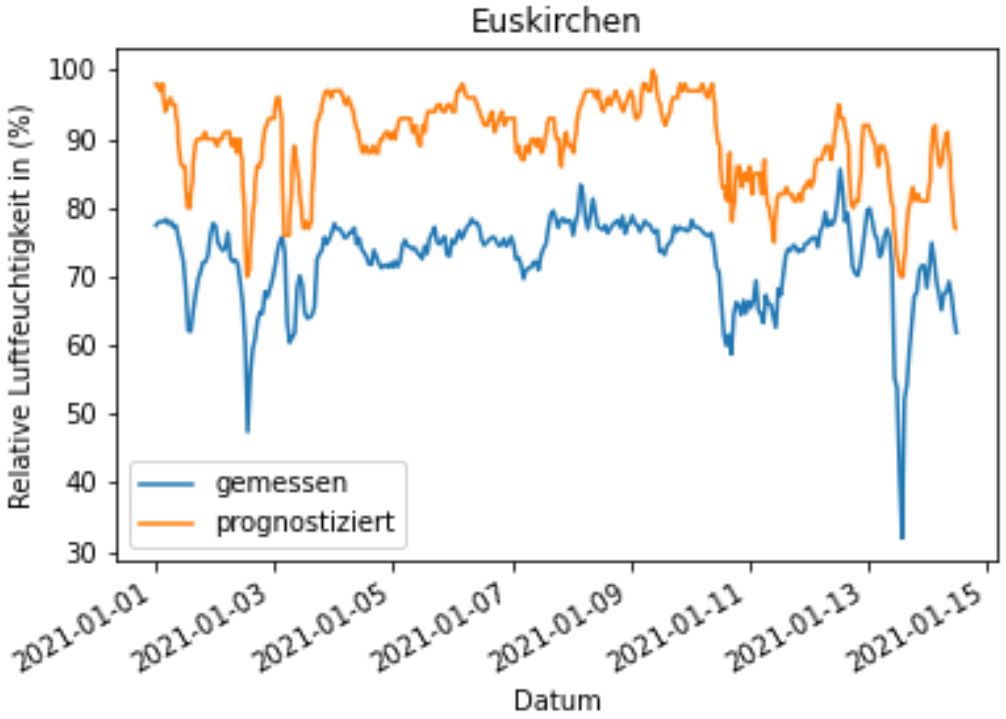


Abbildung 20 Verlauf der relativen Luftfeuchtigkeit am Standort Euskirchen

In Abbildung 20 ist der Verlauf der relativen Luftfeuchtigkeit am Standort Euskirchen dargestellt. Ähnlich wie beim Temperaturverlauf weisen die gemessenen und prognostizierten Daten einen ähnlichen Verlauf auf. Der gemessene Verlauf weist wieder eine vertikale und horizontale Verschiebung auf.

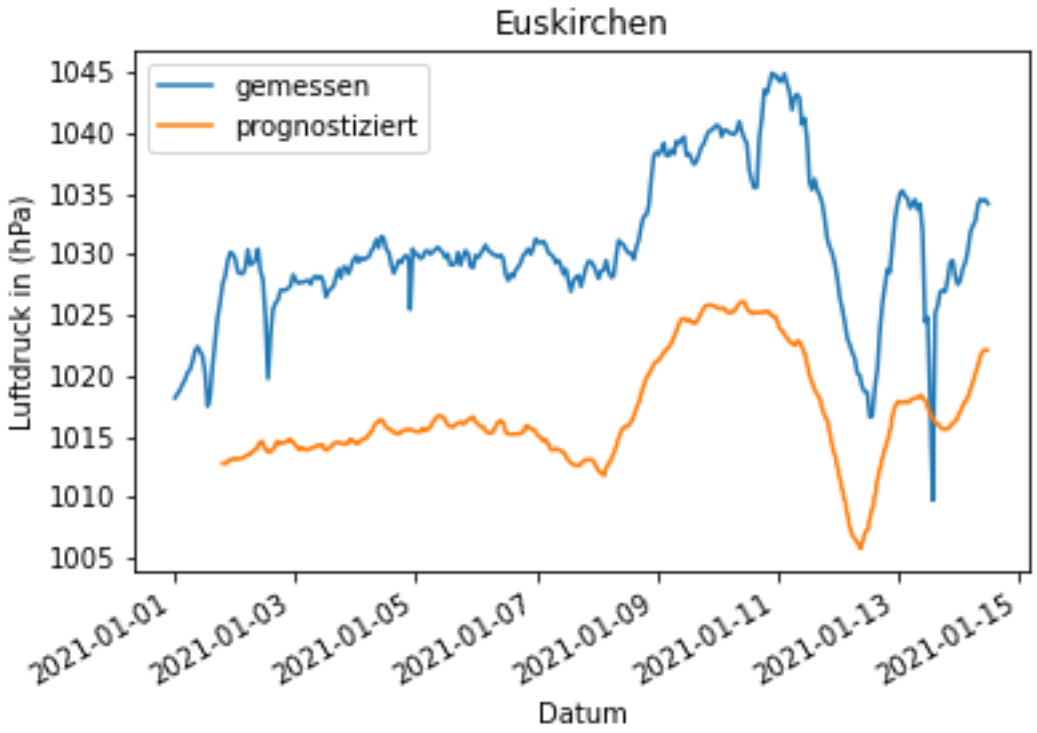


Abbildung 21 Verlauf des Luftdrucks am Standort Euskirchen

In Abbildung 21 – Verlauf des Luftdrucks am Standort Euskirchen – lässt sich ebenfalls ein ähnlicher Verlauf der gemessenen und prognostizierten Daten feststellen. Genauso wie in den beiden vorherigen Fällen ist die gemessene Kurve horizontal und vertikal verschoben. Gründe für die vertikale Verschiebungen können Messfehler und für die horizontale Verschiebung zeitliche Abweichungen in der Datenübertragung sein. Die genannten Gründe werden in der Fehlerdiskussion näher beschrieben.

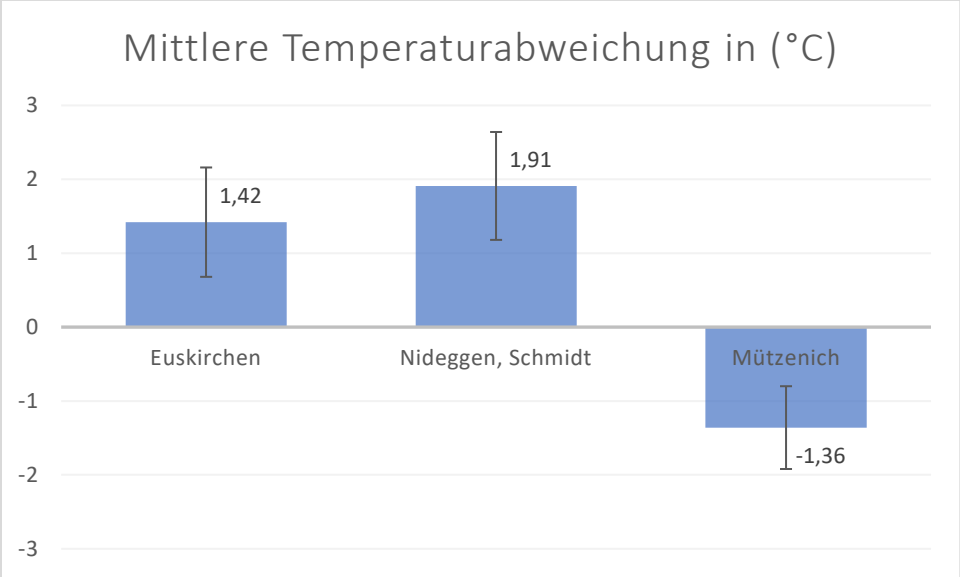


Abbildung 22 Mittlere Temperaturabweichung

Die mittlere Temperaturabweichung zwischen gemessener und prognostizierter Temperatur ist für die einzelnen Standorte in Abbildung 22 dargestellt. Die mittleren Abweichungen liegen in jedem Fall in der gleichen Größenordnung. Bis auf den Standort Mützenich liegt in jedem Fall eine positive Abweichung vor.

Fehlerdiskussion

Bei den von der Wetterstation aufgenommenen Werten ist davon auszugehen, dass diese nicht ganz korrekt und damit fehlerbehaftet sind. Zum einen war es sehr schwierig die Wetterstation optimal zu positionieren. Sie musste vor zu viel Regen geschützt werden und gleichzeitig musste es möglich sein, eine WLAN-Verbindung aufzubauen. Beides kann dazu geführt haben, dass der Abstand zu einer Wärmequelle (Bsp. dem Haus) nicht ausreichend war und so die Temperaturmessung beeinflusst hat. Zum anderen ist es möglich, dass Feuchtigkeit in das selbstgedruckte (nicht 100% dichte) Gehäuse eingedrungen ist und die Feuchtigkeitsmessung verfälscht wurde.

Ein weiterer zu beachtendem Punkt ist, dass es nicht möglich war den verwendeten Sensor zu kalibrieren. Es musste also darauf vertraut werden, dass die Messung des Sensors stimmt. Da die Wetterstation selbst zusammengebaut und gelötet wurde, ist es ebenfalls möglich, dass schlechte Lötstellen oder Verbindungen dazu geführt haben, dass der Sensor kein 100%iges Ergebnis liefert.

Es kann ebenfalls zu unterschiedlichen Ergebnissen der Wetterstation und den Daten aus der Onlinedatenbank kommen, da es nicht möglich immer zu denselben Uhrzeiten zu messen. Die „Deepsleep-Funktion“ des Microcontrollers zeigte Abweichungen von bis zu 30 min auf, sodass sich die Messzeitpunkte der Wetterstation im Vergleich zu den Onlinedaten immer wieder verschoben haben. Diese zeitlichen Verschiebungen können, beim Aufnehmen von Wetterdaten, bereits zu unterschieden führen.

Fazit

Abschließend lässt sich sagen, dass die Wettervorhersagen nicht exakt mit den gemessenen Daten übereinstimmen. Betrachtet man exemplarisch Abbildung 22 ist eine mittlere Temperaturabweichung von bis zu 1,91°C festzustellen. Abweichungen in ähnlicher Größenordnung lassen sich auch für die Vorhersage der Luftfeuchtigkeit und der Vorhersage des Luftdruckes beobachten.

Dadurch, dass die genutzten Wettervorhersagen für ganze Dörfer oder Städte gültig sind, ist mit einer lokalen Abweichung zu rechnen. Außerdem müssen die im Kapitel Fehlerdiskussion aufgeführten Fehlerquellen berücksichtigt werden.

Um präzisere Rückschlüsse ziehen zu können, müssen die Messungen über einen längeren Zeitraum fortgeführt werden. Die aktuellen Messungen wurden ausschließlich bei winterlichen Witterungsbedingungen durchgeführt. Wie gut die Qualität von einer Wettervorhersage bei beispielsweise hoch sommerlichen Bedingungen ist, lässt sich aktuell nur abschätzen.

Basierend auf den bisherigen Erkenntnissen lässt sich die Güte einer Wettervorhersage als gut beschreiben, da es keine signifikanten Abweichungen von den prognostizierten Daten gibt.