Lehrstuhl für Computergestützte Modellierung und Simulation
TUM School of Engineering and Design
Technische Universität München

Integration von Sensordaten in moderne Digital Twin Plattformen

Wissenschaftliche Arbeit zur Erlangung des Grades

Bachelor of Science (B.Sc.)

an der TUM School of Engineering and Design
der Technischen Universität München.

Betreut von Prof. Dr.-Ing. André Borrmann
M.Sc. Sebastian Esser
Dr.-Ing. Simon Vilgertshofer

Eingereicht von Daniel Förster [redacted]
e-Mail: daniel.foerster@tum.de

Eingereicht am 01. Dezember 2023

Abstract

The Digital Twin (DT) is gaining more and more interest in the construction and building industry. The concept could achieve ecological, economic and social improvements, particularly for the operation of the building in the context of facility management. Several platforms already exist for developing and operating a DT. Thus this thesis identifies specific requirements during building operation for a DT platform and examines how these are covered by a selected platform (Autodesk Tandem). Therefore a prototype of a DT is being developed, where the generation and distribution of real-time data is achieved using a self-developed solution.

Zusammenfassung

Das Konzept des Digital Twin (DT) spielt auch im Bau- und Gebäudesektor eine immer größere Rolle. Vor allem für den Betrieb des Gebäudes könnten durch das Konzept im Kontext des Facility Managements Verbesserungen in ökologischer, ökonomischer und sozialer Hinsicht erreicht werden. Es existieren bereits einige Plattformen, welche das Erstellen und Betreiben eines DT ermöglichen. In dieser Arbeit werden daher für den Gebäudebetrieb spezifische Anforderungen an eine DT-Plattform herausgearbeitet und deren Abdeckung durch eine ausgewählten Plattform (Autodesk Tandem) untersucht. Im Zuge dessen wird auch prototypisch ein DT umgesetzt, wobei die Generierung und Verteilung von Echtzeitdaten mittels einer eigens entwickelten Lösung anvisiert wird.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungen

2D	Zweidimensional
AI	Artificial Intelligence
AIM	Asset Information Model
API	Application Programming Interface
BIM	Building Information Modeling
CAFM	Computer Aided Facility Management
CDE	Common Data Environment
CO2	Kohlenstoff-Dioxid
DBMS	Datenbankmanagement-System
DT	Digital Twin
FM	Facility Management
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
IFC	Industry Foundation Classes
IoT	Internet of Things
JSON	JavaScript Object Notation
MQTT	Message Queuing Telemetry Transport
PIR-Senor	Pyroelectric Infrared Sensor
RDF	Resource Description Framework
REST	Representational State Transfer
URL	Uniform Resource Locator
UWB	Ultra-wideband
VOC	Volatile Organic Compounds
WWW	World Wide Web

Kapitel 1

Einleitung

1.1 Motivation

Menschen verbringen ca. 87 Prozent ihrer Zeit innerhalb von Gebäuden (KLEPEIS et al., 2001). Daher ist es nicht verwunderlich, dass Faktoren, wie die Luftqualität in Gebäuden einen massiven Einfluss auf die allgemeine Gesundheit einer Gesellschaft haben (U.S. ENVIRONMENTAL PROTECTION AGENCY, 2023). In Deutschland werden außerdem fast 40 Prozent des gesamten Energieverbrauchs durch den Betrieb von Gebäuden verursacht (BUNDESMINISTERIUM FÜR WIRTSCHAFT UND KLIMASCHUTZ, 2023).

Vor dem Hintergrund dieser Tatsachen lässt sich erkennen, dass mit einer Optimierung des Gebäudebetriebs nicht nur große Mengen an Energie eingespart werden können, sondern auch die Lebensqualität und Gesundheit der Nutzer:innen eines Gebäudes zum Positiven beeinflusst werden kann. Eine vielversprechende Methode stellt in diesem Kontext das Konzept des Digital Twin (DT) dar. Dabei handelt es sich um die digitale Spiegelung der Realität. Durch die Ausstattung eines Gebäudes mit einer Vielzahl von Sensoren können verschiedenste Parameter innerhalb des Gebäudes beobachtet und aufgezeichnet werden. Durch das ständige Generieren und Sammeln von Daten wird eine große Menge an Informationen an einem zentralen Speicherort, dem DT, gesammelt. Mit wachsender Datenmenge können so immer zuverlässigere Erkenntnisse gewonnen und dadurch auch gegebenenfalls eine Optimierung des Gebäudebetriebs ermöglicht werden.

Diese Menge an Daten muss jedoch in einer geeigneten Art und Weise strukturiert und zugänglich gemacht werden. Vor allem für Technologien zur Analyse von großen Datenmengen, wie beispielsweise Artificial Intelligence (AI), ist allerdings die Qualität und Homogenität der Daten von entscheidender Bedeutung. Bei der strukturierten und einheitlichen Datenhaltung von Informationen über ein Gebäude oder ein Bauwerk allgemein, ist Building Information Modeling (BIM) der aktuelle Stand der Technik in Planung und Bau. Um einen DT eines Gebäudes zu verwirklichen, kann daher auf bereits funktionierende Konzepte aus dem BIM zurückgegriffen werden.

Die beschriebene Situation zeigt, dass zwar zur Erfassung modellbasierter Daten mit der Methode des BIM eine umfassende Möglichkeit zur digitalen Beschreibung von Bauwerken besteht, aber deren Fortführung nicht umfänglich genutzt wird. Gleichzeitig gibt es Unterschiede in den Gründen zur Datenerhebung. BIM ist eher feststehende Planungsgrundlage, die iterativ bestimmt wurde, die Sensoren hingegen liefern ständig neue Informationen. Daher wird die Verbindung dieser beiden unterschiedlichen Ansätze in einer DT-Plattform anvisiert.

1.2 Zielsetzung und Forschungsfragen

Zentrale Zielsetzung dieser Bachelorarbeit ist eine prototypische Umsetzung eines Digitalen Zwillings von Räumen des Lehrstuhls für Computergestützte Modellierung und Simulation. Diese Räumlichkeiten befinden sich innerhalb des Innenstadtcampus der Technischen Universität München in der Arcisstraße 21. Zusätzlich soll untersucht werden, wie relevante Daten für den Betrieb eines Universitätsgebäudes in eine DT-Plattform integriert werden können. Dabei sollen sowohl sich zeitlich verändernde Daten, als auch Sensordaten innerhalb des Gebäudes berücksichtigt werden.

Um diese Aufgabenstellung erfüllen zu können, müssen folgende Fragestellungen beantwortet werden:

Forschungsfrage 1: Welche Anforderungen gibt es allgemein an eine DT-Plattform und in welchem Maße deckt eine ausgewählte Plattform (Autodesk Tandem) zusätzlich die spezifischen Anforderungen während des Gebäudebetriebs ab?

Forschungsfrage 2: Welche Schritte sind notwendig um sowohl sich zeitlich verändernde Daten, als auch Sensordaten in eine DT-Plattform zu integrieren?

Forschungsfrage 3: Welches Vorgehen eignet sich für die Generierung und Verteilung von Sensordaten innerhalb eines Gebäudes?

1.3 Aufbau der Arbeit

Zunächst wird im zweiten Kapitel ein Überblick über den aktuellen Stand der Technik in Bezug auf die Thematik gegeben. Im dritten Kapitel wird die Methodik des anschließenden Teils erläutert. Im vierten Kapitel werden die DT-Plattformen allgemein und eine bestimmte Plattform (Autodesk Tandem) genauer analysiert. Im fünften Kapitel wird die Vorgehensweise zur Generierung von Sensordaten beschrieben. In Kapitel 6 erfolgt eine Zusammenfassung der Ergebnisse, um in Kapitel 7 ein Fazit zu ziehen und einen Ausblick auf die Zukunft zu geben.

Kapitel 2

Stand der Technik

In diesem Kapitel wird der aktuelle Stand der Technik in Bezug auf die Integration von Sensordaten in moderne DT-Plattformen dargestellt. Für das Verständnis dieser Thematik sind Grundlagen und einige konkrete Konzepte von BIM relevant. Diese werden im ersten Unterkapitel erläutert. Daraufhin wird das Konzept des DT im Gebäudekontext dargestellt. Im letzten Unterkapitel wird ein kurzer Überblick über Internet of Things (IoT) zur Generierung und Verteilung von Sensordaten innerhalb eines Gebäudes gegeben.

2.1 Building Information Modeling (BIM)

2.1.1 Allgemein

Bei Planung, Bau und Betrieb eines Bauwerks arbeiten zahlreiche Personen aus unterschiedlichen Fachbereichen zusammen. Die herkömmliche Methode beruht auf Plänen in digitaler oder teils sogar noch in Papierform. Zweidimensional (2D) und in Grundrisse, Ansichten, Schnitte und Detailzeichnungen unterteilt, werden so die notwendigen Informationen über das Bauwerk festgehalten und zwischen den beteiligten Personen ausgetauscht. Allerdings gehen bei jeder Übergabe zwischen den Beteiligten hierbei zahlreiche Informationen verloren (siehe Abb. 2.1). Laut BORRMANN et al., 2021, S.2 besteht bei der konventionellen Arbeitsweise ohne BIM beispielsweise ein erhebliches Problem darin, dass Simulationen, Analysen und Berechnungen nicht direkt durchgeführt werden können, sondern in den jeweiligen Softwaretools die relevanten Informationen neu eingegeben werden müssen. Auch die Übergabe von Bauplänen an den Bauherrn nach Vollendung des Bauwerks ist demnach schwierig. Hierbei ist von ihm ein hoher Aufwand zu betreiben, um die erforderlichen Informationen für den Betrieb des Bauwerks zu extrahieren und in ein Facility-Management-System zu integrieren.

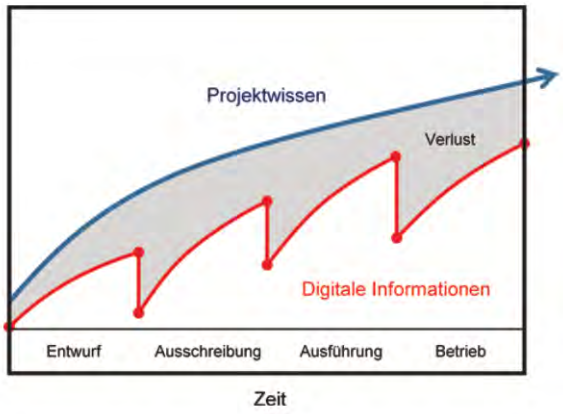

Abbildung 2.1: Informationsverlust bei Projektbearbeitung ohne BIM (BORRMANN et al., 2021, S.3)

BIM zielt darauf ab den Informationsverlust, der bei konventionellen Methoden auftritt, so gering wie möglich zu halten oder sogar komplett zu vermeiden. Als BIM bezeichnet man die konsistente und durchgängige Nutzung von Informationen über ein Bauwerk über dessen gesamten Lebenszyklus hinweg (siehe Abb. 2.2). Sämtliche relevanten digitalen Informationen, welche während der Planungs-, Bau-, und Nutzungsphase anfallen, werden innerhalb eines bzw. mehrerer Modelle gespeichert.

Abbildung 2.2: BIM im Lebenszyklus eines Bauwerks (BORRMANN et al., 2021, S.5)

Bei diesem Modell spricht man von einem Building Information Model. Dabei handelt es sich nicht nur um ein dreidimensionales Abbild der Geometrie der einzelnen Elemente des Bauwerks. Es enthält zudem nicht-physische Elemente, wie Räume und Zonen und eine gewisse Semantik. Unter diesem Begriff versteht man in der Informatik die Bedeutung von Daten oder Informationen. Bezogen auf ein Modell eines Bauwerks bedeutet dies, dass

alle enthaltenen Objekte Instanzen einer bestimmten Objektklasse sind. (z.B. Wand, Tür, Fenster, etc.)

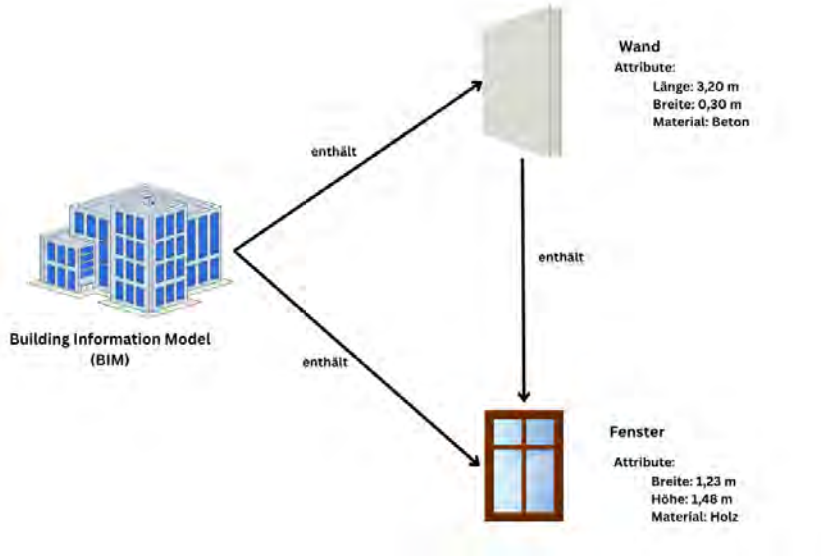

Abbildung 2.3: Aufbau eines Building Information Model

Wie bereits erwähnt und wie in Abb. 2.3 zu erkennen ist, handelt es sich bei BIM um eine objektorientierte Modellierung. Die Objekte des Modells sind Instanzen einer bestimmten Objektklasse bzw. einer Entität. Diese Objekte/ Instanzen haben bestimmte festgelegte Attribute, die sie beschreiben. Dies können beispielsweise geometrische Abmessungen oder Materialkennwerte sein. Mithilfe von benutzerdefinierten Eigenschaftssätzen können den Objekten auch zusätzliche Eigenschaften zugewiesen werden. Außerdem stehen die einzelnen Objekte in Beziehung zueinander und sind durch Relationen und Assoziationen verbunden. So kann beispielsweise wie in Abb. 2.3 eine Wand ein Fenster enthalten. Daher handelt es sich hierbei nicht nur um hierarchische Informationen, sondern um eine hochvernetzte Datenstruktur. Mithilfe der räumlichen Aggregationshierarchie können die Objekte auch Räumen innerhalb des Gebäudes zugewiesen werden. Industry Foundation Classes (IFC) enthalten die beschriebene Struktur und bilden die Grundlage zur offenen Zusammenarbeit mit BIM. IFC wird von verschieden Programmen unterstützt und wird zum Dateiaustausch zwischen den Programmen und den unterschiedlichen bei Planung, Bau und Betrieb beteiligten Personen genutzt.

In Abb. 2.2 wird bei der Betriebsphase bzw. der Bewirtschaftung eines Gebäudes u.a. Facility Managment genannt.

2.1.2 Computer Aided Facility Management (CAFM) als Teil des BIM-Lebenszyklus

Der Begriff Facility Management (FM) ist sehr weitreichend und es existieren unterschiedlichste Definitionen. In der im Jahr 2017 veröffentlichten internationalen Norm ISO 41011 wird der Begiff sinngemäß wie folgt definiert: *FM ist eine organisatorische Aufgabe, wel-*

che die Menschen, den Ort und die ablaufenden Vorgänge innerhalb der gebauten Umwelt berücksichtigt, mit dem Ziel, die Lebensqualität der Menschen und die Produktivität zu verbessern (»ISO 41011: Facility Management - Vocabulary«, 2017). FM ist ein umfassender und interdisziplinärer Ansatz zur Verwaltung und Optimierung von Gebäuden und Anlagen, der durch den Einsatz moderner Technologien ständig weiterentwickelt wird und eine zentrale Rolle bei der Sicherstellung der Funktionalität, Sicherheit und Nachhaltigkeit von Gebäuden spielt. FM beschäftigt sich also im Grunde mit allen Vorgängen und Gegebenheiten, die innerhalb der Nutzungsphase eines Gebäudes stattfinden. Somit fällt auch das Erstellen und Pflegen eines DT eines Gebäudes, welches sich in der Nutzungsphase befindet, in den Fachbereich des FM.

Im Zuge der digitalen Transformation gewinnen innovative Technologien wie das IoT, BIM und DT zunehmend an Bedeutung im FM. Diese Technologien ermöglichen eine verbesserte Datenerfassung, Analyse und Visualisierung von Gebäudedaten und tragen dazu bei, Prozesse zu optimieren, Ressourcen effizienter zu nutzen und Kosten zu reduzieren. Darüber hinaus tragen sie zur Verbesserung der Nachhaltigkeit und Umweltverträglichkeit von Gebäuden bei (BOSCH & DECKERT, 2023).

Wenn eine Anwendungssoftware verwendet wird, um die Prozesse des FM umfassend zu unterstützen spricht man allgemein von Computer Aided Facility Management (CAFM).

In der GEFMA-Richtinie (»GEFMA 400: Computer Aided Facility Management CAFM – Begriffsbestimmungen, Leistungsmerkmale«, 2021) werden die CAFM-Kernanwendungen definiert. Dazu gehören u.a.:

- Flächenmanagement
- Instandhaltungsmanagement
- Inventarmanagement
- Reinigungsmanagement
- Raum- und Asset-Reservierung
- Schließanlagenmanagement
- Umzugsmanagement
- Vermietungsmanagement
- Energiecontrolling
- BIM-Datenverarbeitung
- Vertragsmanagement
- Workplace Management

Diese Liste ist bei weitem nicht vollständig. CAFM deckt somit ein sehr weitreichendes Leistungsspektrum ab. Es ist sehr unwahrscheinlich, dass ein einziges Produkt das gesamte Spektrum abdecken kann. Somit ist die Interoperapilität zwischen den unterschiedlichen Produkten, welche im CAFM eingesetzt werden von hoher Bedeutung. Um diese Interoperabilität zu gewährleisten, muss für die Betriebsphase eine Common Data Environment (CDE) aufgebaut werden (AENGENVOORT & KRÄMER, 2021, S.638) (»ISO DIN 19650-1«, 2019) .

2.1.3 Common Data Environment (CDE)

Um die bereits angesprochene Zusammenarbeit von verschiedenen Personen bzw. Teilbereichen in den Prozessen von Planung, Bau und Betrieb umzusetzen, wird eine CDE benötigt. Dabei handelt es sich um eine gemeinsame Datenumgebung, in der Informationen über das Bauwerk oder ablaufende Prozesse ausgetauscht werden können. Die praktische Erfahrung hat gezeigt, dass die Speicherung aller Informationen innerhalb eines Modells nicht zu empfehlen ist, denn sowohl die parallele Bearbeitung von Planungsaufgaben, als auch die Klärung der Verantwortung von Planungsfehlern gestaltet sich hierbei schwieriger (PREIDEL et al., 2021, S.336). Daher setzen Richtlinien, wie die ISO EN DIN 19650-1 (»ISO DIN 19650-1«, 2019) einen kollaborativen Ansatz um. In diesem stellt die Föderation von lose zusammenhängenden Fachmodellen das Gesamtmodell dar (vgl. Abb. 2.4).

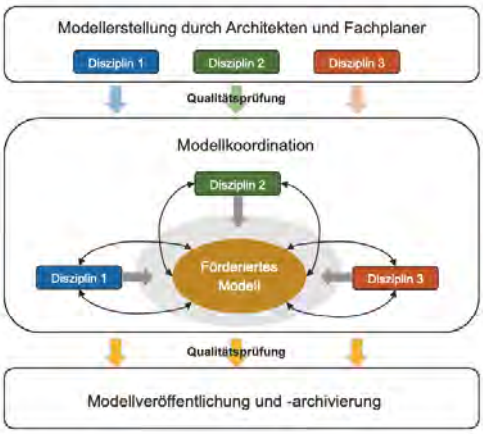

Abbildung 2.4: Fachmodellbasierte Zusammenarbeit mittels eines föderierten Modells (PREIDEL et al., 2021, S.343)

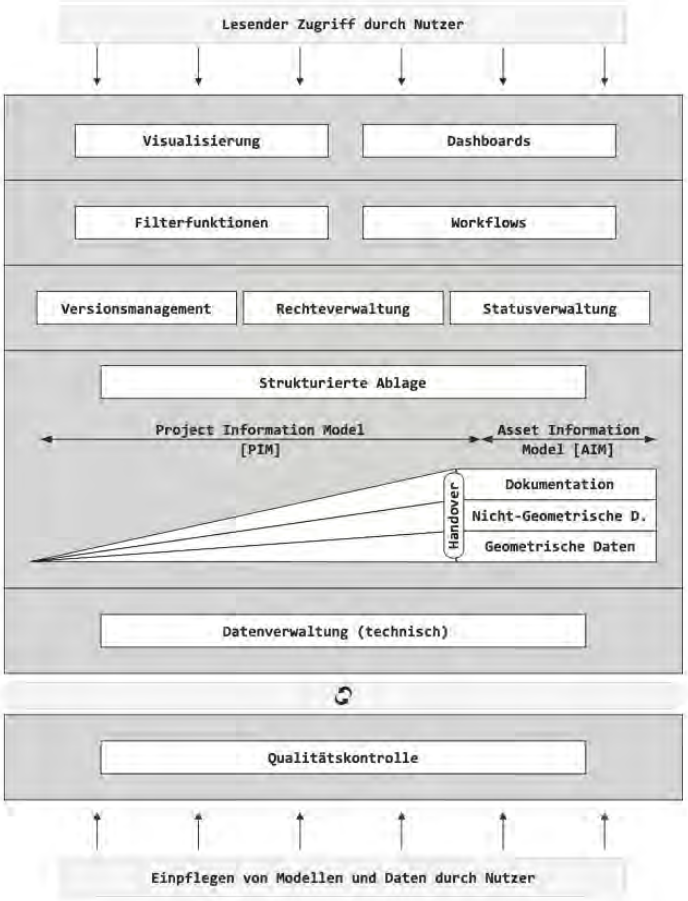

Abbildung 2.5: Typische technische Komponenten einer CDE (PREIDEL et al., 2021, S.343)

In Abbildung 2.5 sind die typischen technischen Komponenten einer CDE dargestellt. Dabei werden zunächst Modelle und Daten von den Nutzer:innen eingepflegt. Das zentrale Element der CDE bildet die Datenverwaltung bzw. das Daten-Repository. Darin werden die gesamten Informationen abgespeichert. Diese sollten für alle Projektbeteiligten zugänglich sein. Somit sind Cloud-basierte Lösungen von Vorteil, da Nutzer:innen über das Internet ortsunabhängig auf die Daten zugreifen können. Ein weiterer zentraler Aspekt ist die strukturierte Ablage der Daten. Während Planung und Bau werden alle Informationen innerhalb eines Project Information Model (PIM), während des Betriebs innerhalb eines Asset Information Model (AIM) gespeichert. Mit dem Übergang von Planung und Bau zur Betriebsphase wechselt der Fokus auf die Betrachtung des Gebäudes als Asset (Vermögenswert). Man spricht deshalb während der Betriebsphase von einem AIM (KRÄMER et al., 2022, S.115). Die Daten der Modelle sind dabei in Dokumentation, Geometrische Daten und Nicht-Geometrische Daten unterteilt. Die Datenstruktur kann zusätzlich durch Klassifizierungssysteme unterstützt werden. In der »DIN EN ISO 12006-2«, 2020 wird zu der Notwendigkeit einer Klassifizierung folgendes geäußert: *„Bei der Bauwerksinformationsmodellierung handelt es sich um den Austausch aller Arten von Information im Verlauf des Projektzeitplans zwischen Beteiligten und Anwendungen. [...] Damit dieser Austausch erfolgreich ist, ist ein vollständiger und konsistenter Ansatz an die Bauobjektklassifizierung im Projekt und zwischen Projekten erforderlich. “* In der Norm wurde daher eine Struktur

für Klassifizierungssysteme für die Baubranche definiert. Durch eine Rechteverwaltung wird sichergestellt, dass nur die wirklich dazu berechtigten Personen Eigentums-, Zugriffs- und Änderungsrechte auf gewisse Daten erhalten. Bei Änderungen von Informationen innerhalb der CDE wird immer eine neue Version abgespeichert. Zusätzlich wird der Änderungsverlauf in einem Journal gesichert. So können mögliche Fehler während der Prozesse detektiert und durch Zurückgehen auf eine frühere Version behoben werden. Mithilfe von Filterfunktionen können Nutzer:innen die für sie relevanten Informationen gezielt herausfiltern und somit nutzbar machen. Die Darstellung erfolgt durch Visualisierungen und Dashboards. (PREIDEL et al., 2021, S.343)

2.1.4 BIM-CAFM-Integration

Nach der Fertigstellung eines Bauwerks sollen im Idealfall sämtliche, während Planung und Bau angefallenen, Informationen an den Bauherren übergeben werden (Handover). Für die Umsetzung dieses Handovers gibt es drei unterschiedliche Szenarien: der Einmalige FM-Handover, die Bidirektionale Synchronisation oder das Integrierte Asset-Information-System (AENGENVOORT & KRÄMER, 2021, S.638ff).

Beim einmaligen FM-Handover werden vor allem die relevanten alphanumerischen Daten direkt in die CAFM-Datenbank übertragen. Zur grafischen Darstellung werden die Modelle einmalig in klassische 2D-CAD-Pläne umgewandelt und über den Lebenszyklus fortgeführt. Bei CAFM-Systemen, die die bidirektionale Synchronisation verwenden, erfolgt der Datenaustausch über eine zumeist proprietäre Schnittstelle. In das in Planung und Bau genutzte Autorenwerkzeug muss ein Plug-in integriert werden. Dieses überträgt Änderungen der Informationen zwischen CAFM-System und dem BIM-Autorenwerkzeug. Beim dritten Szenario ist das CAFM-System integtraler Bestandteil eines Asset-Informations-System (vgl. Abb. 2.6). Hierbei wird in der Regel ein BIM-Modellserver verwendet, in dem die (BIM)-Modelle mithilfe eines Datenbankmanagement-System (DBMS) zur Verfügung gestellt werden. Bei diesem Server-basierten Ansatz spielt es für die Nutzer:innen keine Rolle, aus welchem System oder welcher Datenquelle sie Informationen benötigen. Über eine einheitliche Integrations- und Abfrageschicht können sie an alle Informationen gelangen oder neue Informationen abspeichern. Daher wird dieser Ansatz auch als verteilte Datenhaltung nach dem Linked-Data-Prinzip bezeichnet.

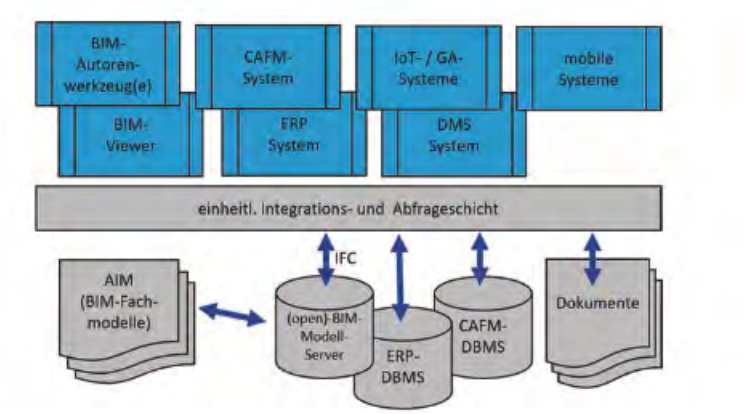

Abbildung 2.6: Integriertes Asset-Information-System (AENGENVOORT & KRÄMER, 2021, S.642)

Bei diesem Ansatz enthält das Integrationssystem selbst keine Daten, sondern nur Metadaten, um die zur Anfragebeantwortung benötigten Datenquellen ausfindig zu machen und abzufragen (KRÄMER et al., 2022, s.278). Dies eignet sich besonders gut für die Integration von sehr heterogenen Datenquellen. Beispielsweise wurde bei dem Projekt BIM-FM die Datenhaltung für zwei Bestandsgebäude (St. Hedwig Krankenhaus Berlin, Verbändehaus Berlin) untersucht (KRÄMER et al., 2018). Dazu wurden mehrere IFC-Modelle mit kommerziellen CAFM-Systemen und Punktwolken verknüpft. Diese Punktwolken wurden mittels 3D-Laserscans der Gebäude generiert. Es wurden verschiedene Datenbanktechnologien genutzt. Diese wurden mittels der Semantic-Web Technologie Resource Description Framework (RDF) verknüpft und gemeinsam abgefragt.

2.1.5 Linked Data

Das Linked Data Prinzip stammt ursprünglich aus dem World Wide Web (WWW) und dem Semantic Web (BERNERS-LEE et al., 2001). Im WWW können Menschen mittels Hyperlinks zueinander gehörenden Informationen durch das Netz folgen. Beim Semantic Web sind diese Verbindungen zwischen den Datenpunkten so beschaffen bzw. mit zusätzlichen Informationen oder Bedeutungen versehen, dass es auch Maschinen möglich ist, das Netz zu lesen. Ein Beispiel für diese Semantic Web Technologie ist das bereits erwähnte RDF.

Unter dem Begriff RDF versteht man einen graphenbasierten Ansatz zum Speichern von Informationen. Er enthält grundsätzlich einen dreiteiligen Datensatz, bestehend aus Subjekt, Prädikat, Objekt (vgl. Abb. 2.7). Bezogen auf ein Gebäude könnte solch ein Triplet beispielsweise lauten: $Wand3 \rightarrow istverbundenmit \rightarrow Wand4$

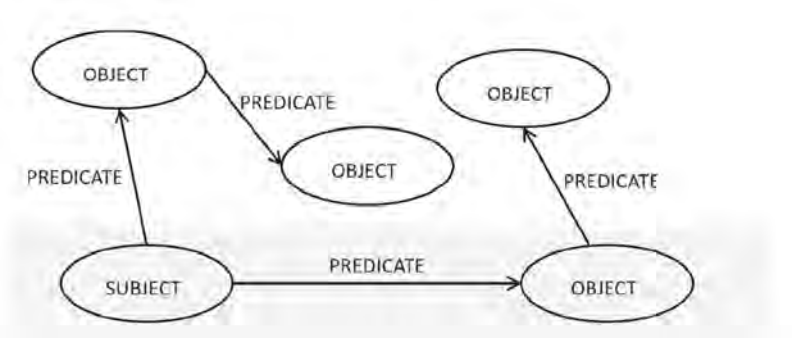

Abbildung 2.7: Schema des Resource Description Framework(RDF) (PAUWELS et al., 2022, S.163)

Sensordaten bzw. dynamische Daten sollten nicht direkt in das Bauwerksmodell oder das RDF integriert werden, da sie aus vielen einzelnen Datenpunkten bestehen. Eine solche Datendichte lässt sich nicht in die monolithische Datenstruktur eines BIM oder IFC-Modells sowie in Wissensgraphen integrieren (PAUWELS et al., 2022, S.179). Die Sensoren selbst, sowie deren Zusammenhang mit dem Bauwerk hingegen sollten durchaus dort gespeichert werden. In dem RDF werden somit die Metadaten gespeichert, welche die Verknüpfung der Daten und somit eine Abfrage derselben ermöglicht. Die Sensordaten selbst werden typischerweise in Datenbanken abgespeichert, die für Zeitreihendaten ausgelegt sind.

Auch laut BORRMANN et al., 2022 ist der Linked Data-Ansatz für das Konzept des DT sehr relevant. Die Herausforderung besteht laut ihnen hierbei darin, die einzelnen Ontologien und ihre Objekte korrekt zu verknüpfen.

DONKERS et al., 2023 analysieren in ihrem Artikel das Potential von Semantic Web Technologien für Preprocessing der Daten. Sie kommen zu dem Schluss, dass Data Preprocessing eine zeitaufwendige Angelegenheit ist und Expertenwissen benötigt. Semantic Web stellt eine Möglichkeit dar, diesen Vorgang zu optimieren.

2.2 Digital Twin (DT)

Das Konzept des DT wurde erstmals im Jahre 2003 im Kontext von Produkt Lebenszyklus Management erwähnt (GRIEVES, 2015). Bekanntheit erlangte das Konzept im Jahre 2012 als die NASA den DT einführte, um Raketen und andere Fluggeräte zuverlässiger und sicherer konstruieren zu können (GLAESSGEN & STARGEL, 2012). Durch den DT konnten sie das Verhalten ihrer Fluggeräte in Extremsituationen besser simulieren und somit Optimierungen an der Konstruktion vornehmen.

Das Konzept des DT erhält auch in der Baubranche seit einigen Jahren immer mehr Aufmerksamkeit. Die Definitionen dieses Begriffs gehen teils sehr weit auseinander und sind auch zwischen den Fachbereichen sehr unterschiedlich.

In Anlehnung an BRILAKIS et al., 2020 wird im Rahmen dieser Forschungsarbeit der Begriff folgendermaßen definiert: Bei einem DT handelt es sich um die digitale Nachbildung eines physischen Bauwerks. Was der DT enthält und in welcher Art und Weise er das Bauwerk darstellt, hängt immer mit den Zielen zusammen, weshalb der DT entwickelt wird. Um den aktuellen Zustand des realen Objekts darzustellen, sollten die Daten regelmäßig aktualisiert werden. Ein DT sollte standardisiert sein und dennoch erweiterbar, in der Lage, zentrale Anwendungsfälle direkt zu erfüllen und spezielle Anwendungsfälle mit Erweiterungen zu unterstützen. Er sollte cloud- und rechenfreundlich sein, sowie skalierbar und überprüfbar.

Laut BORRMANN et al., 2022 sind die DT-Technologien eine natürliche Weiterentwicklung der BIM-Technologien, da beide Konzepte auf klar definierten Datenstrukturen basieren. Somit sind auch die bereits bewährten Datenmodelle, wie IFC weiterhin relevant. Zusätzlich ist jedoch eine flexiblere Kombination mit Ontologien kleineren Umfangs für DT unerlässlich, ebenso wie ein viel feinerer Datenzugriff, der den konventionellen dateibasierten Datenaustausch ersetzen muss.

Beispielsweise kann während der Nutzungsphase eines Gebäudes durch Sensoren beobachtet und aufgezeichnet werden, wie sich diverse Parameter inner- oder außerhalb des Gebäudes verhalten. Dadurch kann Real Time Monitoring und Real Time Visualtisation ermöglicht werden.

2.2.1 Struktur eines DT

LU et al., 2019 haben eine Systemarchitektur für einen dynamischen DT entwickelt. Diese besteht aus fünf verschiedenen Schichten (Layer): "Data Acquisition Layer", "Transmission Layer", "Digital Modelling and Data Complementary Layer", „Data/Model Integration Layer“und „Application Layer“(vgl. Abb. 2.8).

Dabei werden im "Data Acquisition Layer"Daten von verschiedenen Sensoren und Geräten generiert und Über das "Transmission Layer"weitergeleitet. Im "Digital Modelling and Data Complementary Layer"treffen dann die Daten zusammen. Im „Data/Model Integration Layer“können die Daten analysiert und ausgewertet werden, um dann im „Application Layer“einem konkreten Anwendungsfall zu dienen.

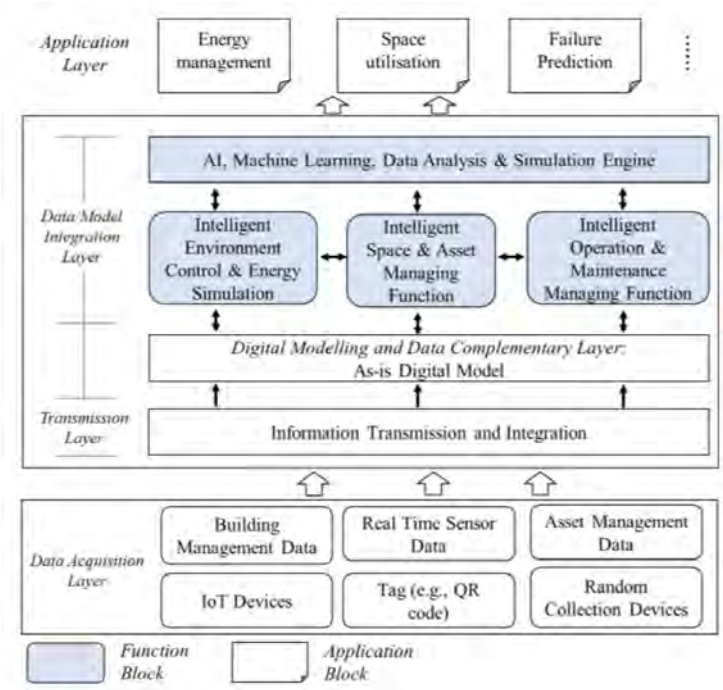

Abbildung 2.8: Systemarchitektur eines DT auf Gebäudeebene (LU et al., 2019, S.70)

2.2.2 Entwicklung von BIM zu DT

Es gibt zahlreiche Forschungsarbeiten, die sich mit DT im Gebäudesektor beschäftigen. DENG et al., 2021 haben versucht durch eine systematische Übersichtsarbeit einen Überblick zu geben. Dazu haben sie 123 verschiedene wissenschaftliche Artikel analysiert und die dort vorgestellten Konzepte eingeordnet. Um die Einordung der in den Artikeln vorgestellten Konzepte durchzuführen, haben sie fünf verschiedene Entwicklungsstufen herausgearbeitet (vgl. Abb. 2.9). Diese gliedern laut ihnen die Evolution ausgehend von BIM hin zu einem DT.

Studien, welche sich laut diesem Bewertungsschema auf der ersten Stufe befinden, nutzen die BIM-Methode als Werkzeug zur statischen 3D Visualisierung. BIM wird genutzt um die Informationsübergabe zwischen den verschiedenen Parteien zu verbessern, die im gesamten Lebenszyklus eines Bauwerks zusammenarbeiten.

Auf der zweiten Stufe fungiert das statische BIM-Modell als Datengrundlage, um darauf aufbauend Simulationen durchzuführen. Dadurch kann der Konstruktionsprozess oder das Verhalten des Gebäudes während der Nutzungsphase analysiert werden. (z.B. Energiebedarf, Thermische Simulationen, etc.)

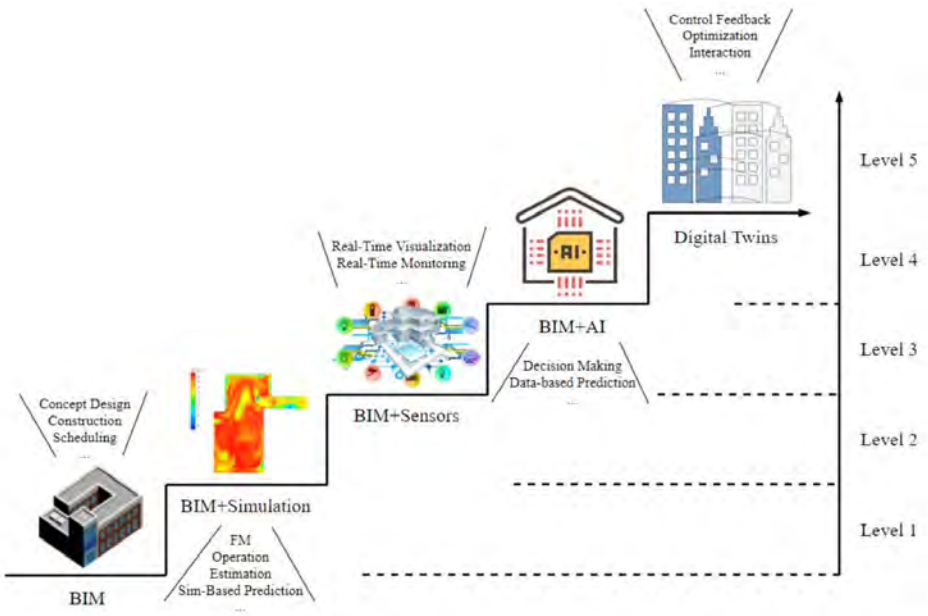

Abbildung 2.9: Entwicklung von BIM zu Digital Twins in der gebauten Umwelt (DENG et al., 2021, S.61)

Studien des dritten Levels beginnen damit das digitale Gebäudemodell mit Echtzeitdaten zu kombinieren und ermöglichen dadurch verschiedene Gegebenheiten der gebauten Umwelt zu beobachten und zu beeinflussen. So kann etwa das Innenraumklima beobachtet und gesteuert werden, um damit das thermische Wohlbefinden der Personen im Gebäude zu beeinflussen.

Auf der vierten Stufe werden die vorliegenden Daten mittels AI ausgewertet, um darauf fundierte Entscheidungen und Vorhersagen treffen zu können.

Erst bei der letzten Stufe handelt es sich laut ihrer Charakterisierung um einen vollständigen DT. Dieser ist in der Lage direkt auf die Parameter innerhalb des Gebäudes automatisch zu reagieren und eröffnet dadurch Optimierungsmöglichkeiten.

2.2.3 DT einer Campusumgebung

BÄCKLUND et al., 2022 stellen in ihrem Paper die ersten Schritte hin zur Implementierung eines DT einer Campusumgebung dar. In ihrem Fall lag der Campus einer Universität in Schweden vor. Die Pläne der Räumlichkeiten waren veraltet und lagen teils nur in Papierform vor. Sie unterteilten die Umsetzung des DT in vier Phasen: Voruntersuchungen, Sensor Installation, Generierung des Modells und Anreicherung des Modells mit Daten.

Die Voruntersuchungen sind laut ihnen ein interaktiver Prozess, in dem die Stakeholder zusammenkommen. Sie tauschen dabei ihre Ideen aus, inwiefern ein DT die Campusumgebung verbessern kann. Im Rahmen eines Workshops wurden hier die relevanten Umgebungsparameter festgelegt: Raumbelegung, CO2, Temperatur, Anzahl an Personen und in manchen Räumen auch die Lautstärke. Im Rahmen der Senorinstallation wurden die Räume und innerhalb derer die bestmöglichen Positionen für die Sensoren festge-

legt. Das Modell wurde mittels Laser-Scans generiert. Bei diesem Vorgang entsteht eine 3D-Punktwolke der Umgebung, welche danach weiter modifiziert wird, um daraus ein vollständiges 3D-Modell zu erstellen. Allerdings handelt es sich hierbei nur um eine geometrische Repräsentation der Wirklichkeit. Daher muss im letzten Schritt dieses 3D-Modell noch mit Daten angereichert werden.

Im Rahmen dieser Arbeit liegt der Fokus auf einem DT eines Gebäudes, welches sich in der Nutzungsphase befindet. Das Betreiben eines Gebäudes fällt in den Fachbereich des Facility Management.

2.2.4 DT-Plattformen

Grundsätzlich ist es schwierig einen umfassenden Überblick über die aktuelle Lage von DT-Plattformen auf dem Markt zu geben. Denn aufgrund der hohen Popularität und Aktualität des Themas kommen derzeit ständig neue Produkte auf den Markt. Dennoch folgt nun eine Auflistung einiger DT-Plattformen:

1. **Autodesk Tandem**: Autodesk ist ein US-amerikanischer Softwarehersteller, gegründet im Jahre 1982 (AUTODESK, INC, 2023). Mit Produkten, wie etwa AutoCAD und Revit sind sie im Bausektor seit Jahrzehnten stark vertreten. Im Jahre 2021 wurde von ihnen Autodesk Tandem veröffentlicht. Dabei handelt es sich um eine cloud-basierte DT-Plattform. Diese wird in Kapitel 4 detaillierter betrachtet.

2. **Microsoft Azure Digital Twins**: Azure Digital Twins ist eine Plattform, die die Erstellung von Zwillingsgraphen auf der Grundlage von digitalen Modellen ganzer Umgebungen ermöglicht. Bei einer solchen Umgebung kann es sich beispielsweise um Gebäude, Fabriken, Energieversorgungsnetze, Eisenbahnstrecken oder Stadien handeln. Sogar ganze Städte lassen sich modellieren. Diese digitalen Modelle liefern wichtige Einblicke, um Produkte zu verbessern, Vorgänge zu optimieren, Kosten zu senken und die Benutzerfreundlichkeit zu verbessern (MICROSOFT, INC, 2023a). Allerding ist als großer Nachteil von AZ zu nennen, dass man ein BIM nicht direkt in die Plattform importieren kann. In der Plattform werden Modelle mithilfe einer eigenen Sprache, der Digital Twin Definition Language (DTDL) definiert. Um ein BIM zu importieren muss man dieses Modell zuerst in DTDL konvertieren (MICROSOFT, INC, 2023b). Dabei gehen allerdings zahlreiche Informationen über das Bauwerk verloren.

3. **ACCA usBIM**: ACCA, gegründet im Jahre 1989, ist ein Hersteller von Software und Dienstleistungen für Architektur, Ingenieurwesen und Konstruktion (ACCA SOFTWARE S.P.A., 2023). Sie sind vor allem bei der Transformation von BIM zu openBIM eine der aktivsten Softwarehersteller (BUILDINGSMART, 2022). Mit usBIM stellen sie auch eine DT-Software zur Verfügung. Auch bei dieser handelt es sich um eine cloud-basierte DT-Plattform.

4. **Bentley iTwin**: Bei Bentley iTwin handelt es sich um eine Plattform, zur Entwicklung von SaaS-Lösungen (Software as a Service) für Planung, Bau und Betrieb von Infrastrukturanlagen (BENTLEY SYSTEMS INC., 2023). Der Fokus liegt hier also mehr auf der Entwicklung von Anwendungen für Infrastrukturprojekte und nicht konkret auf der Entwicklung und dem Betrieb eines DT eines Gebäudes.

5. **Nemetschek dTwin**: Auch die Nemetschek Group, ein deutscher Softwareanbieter, zu deren Produkten unter anderem Allplan zählt, arbeitet mit dTwin an einer DT-Software (DTWIN, 2023). Für diese steht derzeit allerdings nur eine Demo-Version auf Anfrage zur Verfügung (Stand: Oktober 2023).

6. **AWS IoT TwinMaker**: Amazon Web Services (AWS) bietet mit dem AWS IoT TwinMaker auch eine Plattform an, um DT-Lösungen umzusetzen.

7. **Eclipse Digital Twin** ist eine open-source Lösung zur Umsetzung von DT-Lösungen vor allem für den Industrie- und Produktionsbereich.

LEHNER et al., 2022 haben bereits ein Bewertungsschema entwickelt, um DT-Plattformen anhand verschiedener Anforderungen miteinander zu vergleichen. Ihre Betrachtung findet vor allem aus Sicht der Industrie und Produktion statt. Dennoch sind im Grunde alle Anforderungen für einen DT im Bau- und Gebäudesektor dennoch relevant, da es viele Überschneidungen in den Anforderungen gibt.

Laut Definition ist ein DT die digitale Nachbildung eines physischen Bauwerks. Dabei ist es u.a. wichtig die aktuellen Zustände des realen Gebäudes zu repräsentieren. IoT ist ein Ansatz (bzw. Konzept) zur Generierung und Verteilung von Sensordaten innerhalb eines Gebäudes. Dadurch können Echzeitdaten bereitgestellt werden.

2.3 Internet of Things (IoT) und Sensorik

Der Begriff IoT wurde von Kevin Ashton im Jahre 1999 erstmals erwähnt. Die ersten diesem Konzept zugehörigen Erwähnungen reichen bis in das Jahr 1968 zurück (KROKER, 2018). IoT umfasst verschiedene Technologien, die es ermöglichen physische und virtuelle Gegenstände (Maschinen, Computer, Sensoren) durch das Internet miteinander zu verknüpfen (BABEL, 2023).

Innerhalb eines Gebäudes ermöglichen IoT-Geräte und Sensoren die kontinuierliche Überwachung und Erfassung von verschiedensten Parametern wie Temperatur, Luftfeuchtigkeit, Beleuchtung, Energieverbrauch und Präsenz von Personen. Diese Daten fließen in Echtzeit in den DT ein, wodurch eine präzise und aktuelle Abbildung des Gebäudezustands ermöglicht wird.

2.3.1 Sensorik zur Präsenzerkennung von Personen

Um festzustellen, ob sich Personen in einem Raum befinden gibt es verschiedene mögliche Ansätze. PEDERSEN et al., 2017 haben durch Kombination von verschiedenen Sensoren eine Genauigkeit von 98% in einem einzelnen Raum erreicht. Sie nutzten dazu folgende Sensoren bzw. Messwerte:

- Lufttemperatur
- Relative Luftfeuchtigkeit
- Absolute Luftfeuchtigkeit
- Kohlenstoff-Dioxid (CO2)
- Volatile Organic Compounds (VOC)
- Pyroelectric Infrared Sensor (PIR-Senor)
- Geräusche (Lautstärke)

Bei VOC handelt es sich um flüchtige organische Verbindungen in der Luft. Diese können beispielsweise aus Reinigungsmitteln oder Baustoffen stammen. Aber auch natürlich Menschen geben über ihren Atem diese Stoffe an die Luft ab.

Bei dem PIR-Senor handelt es sich um einen Bewegungssensor, wie er in den meisten automatischen Lichtschaltungen integriert ist. Dieser Sensor nimmt Bewegungen mithilfe der Infrarot-Technologie wahr. Zusätzlich zu dieser weit verbreiteten und bereits länger verwendeten Sensortechnologie zur Erkennung von Personen, gibt es neuere Sensoren, die mit Ultra-wideband (UWB) arbeiten. Die UWB Technologie ist besonders gut dafür geeignet, um innerhalb einer Umgebung menschliche Aktivität bzw. Präsenz festzustellen (THULLIER et al., 2022).

2.3.2 Protokolle zur Datenübertragung

Grundsätzlich stehen verschiedene Ansätze zur Verfügung, um Informationen zwischen Sensoren und zentralen Komponenten (Server, Datenbanken, etc) auszutauschen. Einige ausgewählte Technologien werden nachfolgend dargestellt.

Message Queuing Telemetry Transport(MQTT)

Bei Message Queuing Telemetry Transport (MQTT) handelt es sich um ein leichtgewichtiges Kommunikationsprotokoll zur Datenübertragung innerhalb von Netzwerken. MQTT zeichnet sich vor allem durch seine kleine Datengröße und den geringen Stromverbrauch aus (SHILPA et al., 2022). Es beruht auf dem Client-Server und dem publish-subscribe-Prinzip. Bei dieser Art der Datenübertragung veröffentlicht (published) ein Gerät unter

einem bestimmten Topic Daten. In Abb. 2.10 ist eine beispielhafte Datenübermittlung der Temperatur innerhalb eines Netzwerks dargestellt. Bei diesem Beispiel published ein MQTT-Client unter dem Topic Temperatur den Wert 25 °C. Diese Nachricht verschickt er an den MQTT-Broker, der als eine Art Server fungiert. Über diesen zentralen Knotenpunkt laufen alle Datenströme. Wenn ein weiterer MQTT-Client nun die von dem anderen Gerät versendeten Daten erhalten möchte, muss dieser den topic temperatur abonnieren (subscriben) und erhält dadurch die Temperatur von 25 °C. Um sicherzustellen, dass auch nur dazu berechtigte Personen oder Geräte die Daten erhalten, können zum Abonnieren eines Topics ein Benutzername und ein Passwort abgefragt werden.

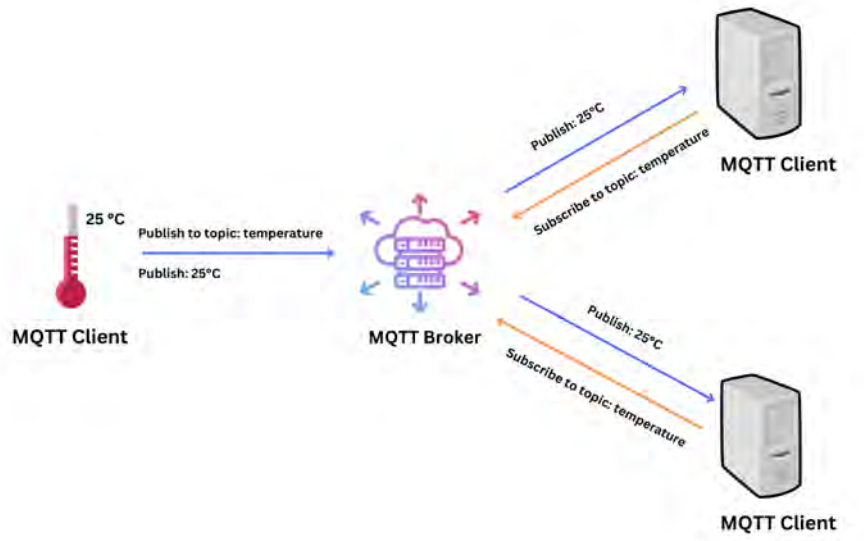

Abbildung 2.10: Schematischer Aufbau von MQTT

Hypertext Transfer Protocol(HTTP)

Das Hypertext Transfer Protocol (HTTP) stammt aus dem WWW und ist dabei eines der zentralen Protokolle. Das WWW allgemein beruht auf dem Client-Server-Prinzip. Wenn ein Client mit einem Server Daten austauschen möchte, stellt er dazu ein HTTP-Request (Anfrage). Um Informationen zu erhalten, stellt er einen GET-Request, um Informationen an den Server zu versenden, stellt er einen POST-Request. Der Server antwortet auf diesen Request mit einem Response.

2.3.3 Framework

CHAMARI et al., 2023 haben in ihrem Artikel ein Framework erarbeitet für das Management von IoT-Daten, die innerhalb eines Gebäudes generiert wurden (vgl. Abb. 2.11). Sie haben dabei drei verschiedene IoT Sensor Nodes verwendet um die aktuellen Parameter innerhalb eines Gebäudes zu messen. Zwei dieser Geräte waren fertige, auf dem Markt erhältliche Sensor Nodes. Die dritte haben sie auf der Grundlage eines ESP32 selbst entwickelt. Die beiden gekauften Sensor Nodes stellten über ihre jeweilige Plattform die Daten zur Verfügung. Die von ihnen selbst entwickelte Lösung verschickte innerhalb eines

lokalen Netzwerks die Daten. Alle drei sendeten per MQTT ihre Daten an einen MQTT-Broker (Mosquitto). Die Herausforderung war in ihrem Fall die Heterogenität der, durch die Sensor Nodes generierten, Daten. Daher erhielt ein Nachrichtenprozessor nach dem MQTT-Broker die Daten, um sie zur anschließenden Speicherung oder Weiterverwendung in eine einheitliche Form zu bringen.

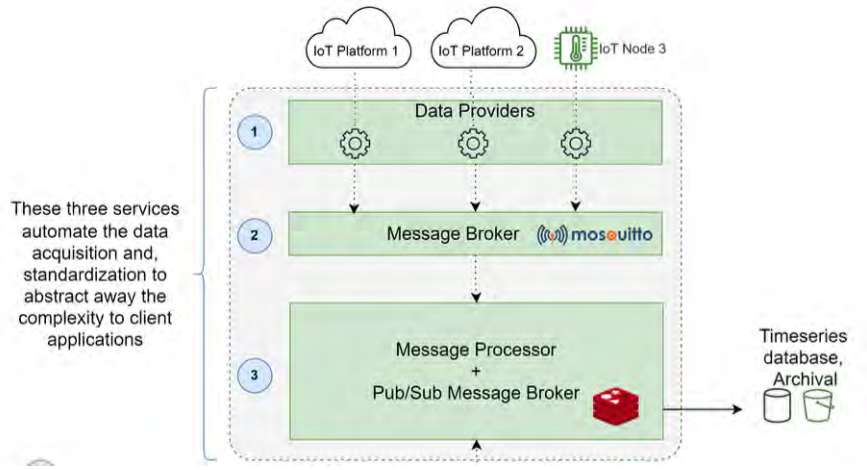

Abbildung 2.11: Framework für IoT Data Management (CHAMARI et al., 2023)

Dies ist nur ein Teil ihres gesamten Frameworks. Nach den beschriebenen Schritten konnte über eine API auch direkt auf die Daten des Nachrichtenprozessors zugegriffen werden.

2.4 Zusammenfassung

Mit BIM existieren bereits vor allem für Planung und Bau gut funktionierende Konzepte, die eine einheitliche und strukturierte Datenhaltung ermöglichen. In der Betriebsphase sind allerdings noch Lücken vorhanden. Das Konzept des DT stellt eine vielversprechende Möglichkeit dar, um die BIM-Methode auf den Betrieb eines Gebäudes zu erweitern (vgl. Abb. 2.12). Die aus Planung und Bau stammenden Modelle werden dazu in den DT importiert und dort mittels des Linked-Data-Ansatzes mit den Betriebsdaten verknüpft. Mithilfe von IoT und Sensorik lassen sich Echtzeitdaten sammeln und in den DT übertragen. Zusätzlich können auch andere, aus dem FM stammende, Daten, wie etwa Raumbelegungen, Reperaturhinweise, Wartungstermine, etc. in dem DT gespeichert werden.

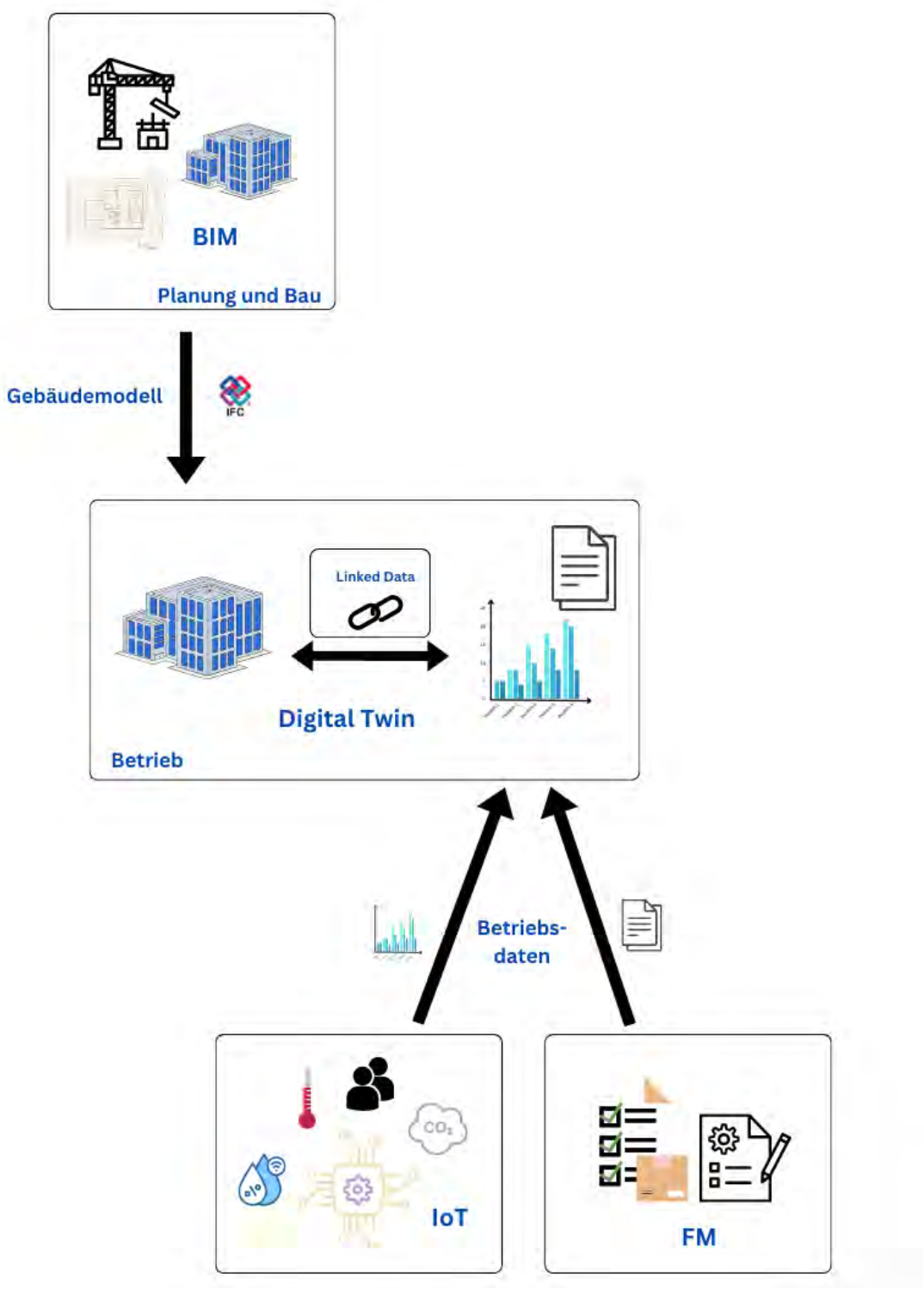

Abbildung 2.12: Schematischer Aufbau eines DT

Mit einigen von den in 2.2.4 vorgestellten Plattformen existieren bereits Lösungen, um einen DT eines Gebäudes zu entwicklen und zu betreiben. Doch es wurde bisher nicht untersucht, wie die Integration dieser betriebsspezifischen Daten in eine konkrete DT-Plattform umgesetzt werden kann.

Kapitel 3

Methodik

Der Fokus dieser Arbeit liegt auf der prototypischen Umsetzung eines Digitalen Zwillings von Räumen des Lehrstuhls für Computergestützte Modellierung und Simulation. Diese Räumlichkeiten befinden sich innerhalb des Innenstadtcampus der Technischen Universität München in der Arcisstraße 21.

Da es sich bei den vorliegenden Gebäuden um Altbauten handelt, war die Datenlage hinsichtlich der bereits vorliegenden BIM-Modelle teils mangelhaft. Daher musste zunächst eine Strategie entwickelt werden, wie aus den verschiedenen vorliegenden Modellen die relevanten Informationen extrahiert und in einem Modell zusammengeführt werden können.

Aufgrund der Tatsache, dass der Digitale Zwilling nur von einigen Räumen erstellt werden sollte, wurden in Bezug auf die Modellierung nur diese Räumlichkeiten betrachtet. Es hätte auch die Möglichkeit bestanden, die gesamten vorliegenden BIM-Modelle in ein großes Modell mit mehreren Gebäuden zusammenzuführen. Mithilfe von Raumzuweisungen und Filtermöglichkeiten der DT-Plattformen, wäre so auch eine gute Darstellung der Räumlichkeiten möglich gewesen. Allerdings hätte dies aufgrund der Inhomogenität der vorliegenden Daten zu erheblichen Komplikationen und einem damit verbundenen zeitlichen Mehraufwand geführt. Daher musste diese Möglichkeit leider verworfen werden. Schließlich wurde ein Modell verwendet, welches bereits vorlag und mittels eines Laserscans generiert wurde.

Der nachfolgende Teil dieser Arbeit ist in zwei Kapitel unterteilt. In Kapitel 4 werden zunächst einige DT-Plattformen verglichen. Dazu werden in 3.1 die Anforderungen an die Plattformen geschildert. Der Vergleich der Plattformen erfolgt nur auf Grundlage der allgemeinen Anforderungen. Lediglich im Fall von Autodesk Tandem werden auch die branchenspezifischen Anforderungen betrachtet und die Plattform anhand derer analysiert. Daraufhin wird das Einpflegen von sich zeitlich verändernden Daten, wie beispielsweise der Raumbelegung durch Lehrstühle, in Autodesk Tandem untersucht. Außerdem wird die Vorgehensweise zur Integration von Sensordaten geschildert.

Allerdings müssen die im vorherigen Kapitel behandelten Sensordaten auch generiert und an die DT-Plattform versandt werden. Daher wird in Kapitel 5 die Vorgehensweise zum Aufbau eines Sensor Netzwerks beschrieben. Dazu wurde als zentraler Knotenpunkt auf einem Raspberry Pi die Plattform openHAB eingerichtet. An diese werden Sensordaten von einer selbstentwickelten Sensor Node übermittelt. Die Entwicklung der Hardware und Software dieser Sensor Node wird ebenfalls geschildert.

Im Anschluss daran findet eine Zusammenfassung der Ergebnisse aus den Kapiteln 4 und 5 statt. Zuletzt wird darauf aufbauend ein Fazit gezogen und ein Ausblick in die Zukunft zu weiterer Forschung in dieser Thematik gegeben.

3.1 Anforderungen an eine DT-Plattform

In Anlehnung an LEHNER et al., 2022 werden folgende allgemeine Anforderungen an eine DT-Plattform gestellt:

1. **Bidirektinonale Synchronisation**: Um die bidrektionale Synchronisation zu gewährleisten, muss der DT die Bidirektionale-Kommunikation zwischen sich und den Geräten herstellen.

2. **Konvergenz**: Konvergenz bedeutet im Fall einer DT-Plattform, dass der reale und der digitale Zwilling stets übereinstimmen müssen. Sollte dies nicht der Fall sein, sollte dies erkannt und behoben werden können. Da die DT-Plattform unter Umständen einen kritischen realen Prozess überwacht, muss diese auch verifiziert und validiert werden können. Mittels historischer Daten vor der Implementierung des DT könnten so Schwellwerte und Reaktionen validiert und verifiziert werden. In manchen Fällen könnte es von Nöten sein, dass der DT wirklich in Echtzeit auf Gegebenheiten reagiert. Doch im vorliegenden Fall, bei dem nur einige Raumwerte ermittelt werden, ist solch ein Echtzeitverhalten nicht von Nöten.

3. **Automatisierungsprotokolle**: Bei Automatisierungsprotokollen handelt es sich um Kommuntikationsprotokolle, welche automatisierte Reaktionen auf Gegebenheiten ermöglichen (z.B. MTConnect, Open Platform Communications Unified Architecture, Siemens S7, Beckhoff Automation Device Specification, Rockwell D1).

4. **Plattforminteroperabilität**: Um die Plattforminteroperabilität vollends zu gewährleisten, muss eine Plattform standardisierte Schnittstellen bieten, um 1) DT-Daten und Strukturinformationen abzurufen und 2) Änderungen an DTs für externe Dienste vorzunehmen.

5. **Systeminteroperabilität**: Um die Systeminteroperabilität zu gewährleisten, muss die Plattform Unterstützung bieten, um verschiedene Geräten über ihre DTs zu verbinden und auf der Grundlage dieser Verbindungen automatische Interaktionen zu implementieren.

6. **Strukturierte Modellierung und Visualisierung**: Die Plattform ermöglicht es Fachleuten, physikalische Geräte, Typen und Daten auf strukturierte Weise zu modellieren und bietet eine grafische Schnittstelle.

7. **Verbindungs- und Datensicherheit**: Um die Verbindungs- und Datensicherheit sicherzustellen muss die Plattform eine Form von Authentifizierungs- und Autorisierungsmechanismus beim Aufbau einer Verbindung zwischen einem DT und

einem physischen System sowie eine sichere lokale Verbindung zwischen den Geräten bieten. Außerdem sollte sie Möglichkeiten zur Verschlüsselung der an den DT gesendeten Daten und der auf dem DT selbst gespeicherten Daten bieten.

8. **Wiederverwendbarkeit**: Die Plattform sollte benutzerdefinierte und vorgefertigte Komponenten bieten, die zwischen Projekten wiederverwendet werden können.
9. **Kontinuierliche Integration (CI) und Kontinuierliche Bereitstellung (CD)** beinhalten die fortlaufende Einarbeitung von Änderungen in einen DT. Nach einer positiven Validierung durch die Qualitätssicherung werden die Änderungen implementiert. Dies sollte realisierbar sein ohne den DT zu stoppen.
10. **Bereitstellung**: DTs können im Sinne der Bereitstellung an verschiedenen Orten betrieben werden. Entweder direkt auf einem Gerät vor Ort oder als cloud-Lösung.

Ferner existieren durch die Besonderheiten im Betrieb eines Gebäudes weitere branchenspezifische Anforderungen an eine DT-Plattform. Die Mindestanforderungen an eine DT-Plattform aus sicht des Gebäudebetriebs wären, dass sie dazu in der Lage ist, sowohl die Geometrie des Gebäudes, sowie dessen aktuellen Zustand in Form von Echtzeitdaten darzustellen.

Wie in Kapitel 2 erläutert wurde, spielen beim Betreiben eines DT eines Gebäudes verschiedene Elemente mit ein. Als Datengrundlage, von Planung und Bau stammend, wird ein BIM, bestehend aus mehreren föderierten Fachmodellen in die Betriebsphase übergeben. Zusätzlich werden während des Betriebs des Gebäudes weitere Daten generiert. Dabei sollte eine CDE beibehelaten, bzw. für den Betrieb des Gebäudes geschaffen werden. In 2.1.3 wurden die dazu gehörigen Grunglagen vermittelt und die typischen technischen Komponenten einer CDE vorgestellt.

Da es sich bei einem DT eines Gebäudes in erster Linie um ein Konglomerat von Daten aus heterogenen Quellen handelt. Genauer gesagt um die Kombination von Gebäudedaten, welche aus dem BIM stammen und Zustandsdaten, welche während des Betrieb generiert und aktualisiert werden. Durch Zusammenführen dieser beiden Datenpakete erhält der DT seinen wahren Wert. Denn er stellt als ein zentraler Sammelpunkt aller Informationen über ein Gebäude ein Informationslexikon dar. Durch eine zentrale Abfrage- und Integrationsschicht, wie sie in 2.1.4 als Teil eines Integrierten Asset-Information-System beschrieben wurde, werden die Daten Nutzer:innen und anderen Systemen zugänglich gemacht.

Zusammenfassend ist also der DT ein zentraler Sammelort für alle möglichen Daten und wird ständig auf dem neuesten Stand gehalten. Er bietet Zugang zu den Daten für Nutzer:innen und Systeme, die am FM beteiligt sind. Doch in 2.2.2 wurde auch erläutert, dass die Funktionsweisen eines DT über dies hinausgehen. Durch Analysewerkzeuge und Algorithmen können potenzielle Verbesserungsmöglichkeiten der Prozesse im FM ausgemacht werden. Außerdem kann der DT sogar durch automatische Reaktionen direkt auf Gegebenheiten in dem Gebäude reagieren und zu einem sicheren, nachhaltigeren und effizienteren Gebäudebetrieb beitragen.

Kapitel 4

Vergleich von DT-Plattformen und Analyse von Autodesk Tandem

In diesem Kapitel findet zunächst ein Vergleich von vier ausgewählten DT-Plattformen statt. Daraufhin wird Autodesk Tandem in seinen Grundzügen beschrieben und die prototypische Integration von verschiedenen Daten untersucht.

4.1 Vergleich von ausgewählten DT-Plattformen

Zum Vergleich wurden folgende DTPlattformen herangezogen:

1. Microsoft Azure Digital Twins (AZ)
2. Autodesk Tandem (AT)
3. Amazon Web Services (AWS)
4. Eclipse (EC)

Im folgenden werden die vier Plattformen anhand der, in 3.1 vorgestellten, Anforderungen miteinander verglichen. Dazu wurde der Vergleich von LEHNER et al., 2022 um eine Spalte für Autodesk Tandem erweitert und diese Plattform dementsprechend eingeordnet (vgl. Tab. 4.1).

Tabelle 4.1: Abdeckung der Anforderungen durch ausgewählte DT-Plattformen (LEHNER et al., 2022, S.57f)

Qualitätsmerkmale	Anforderungen	Bewertungsschema	AZ *	AT	AWS *	EC *
Die funktionale Eignung gibt an, inwieweit ein Produkt oder System die angegebenen und implizierten funktionalen Anforderungen erfüllt, wenn es unter festgelegten Bedingungen eingesetzt wird.	Bidirektionale Synchronisation	🟢 DT stellt automatisch die Bidirektionale-Kommunikation zwischen sich und den Geräten her 🟡 Kombination von manuellem und automatischem Verbindungsaufbau 🔴 Manueller Aufbau einer Verbindung für jedes Gerät	🟡	🟡	🟡	🟡
	Konvergenz	🟢 Die Plattform bietet automatische Unterstützung für die Gewährleistung der Konvergenz zwischen dem DT und seinem physischen Gegenstück 🟡 Die Plattform bietet eine Umgebung zur manuellen Umsetzung der Konvergenz 🔴 Keine Unterstützung für Konvergenz	🔴	🔴	🔴	🔴
	Verifikation und Validierung	🟢 Die Plattform bietet eine Testsuite für bestimmte Funktionen, oder es können Anforderungen definiert und ausgeführt werden, bevor der DT bereitgestellt wird. 🟡 Ungültige Strukturen werden automatisch erkannt (z. B. Senden von Daten von einem Sensor, für den es keinen entsprechenden DT gibt) 🔴 Die Plattform bietet keine Testmöglichkeiten	🟡	🔴	🟢	🔴
Die Performance bezieht sich auf die Reaktionsfähigkeit und die Effizienz der Ressourcennutzung.	Echtzeitverhalten	🟢 Die Plattform bietet eine harte Echtzeitschnittstelle am Edge und die Möglichkeit, weiche Echtzeitbedingungen in der Cloud zu messen und zu bewerten. 🟡 Plattform bietet nur harte Echtzeitschnittstelle am Edge 🔴 Keine (harte oder weiche) Echtzeitunterstützung	🟡	🟡	🟡	🟡
Kompatibilität ist das Ausmaß, in dem ein System den Austausch von Informationen mit anderen Netzen unterstützt.	Automatisierungsprotokolle	🟢 DT unterstützt mindestens zwei weit verbreitete Protokolle, z.B. MTConnect, Open Platform Communications Unified Architecture, Siemens S7, Beckhoff Automation Device Specification oder Rockwell D1 🟡 Eines dieser Protokolle wird unterstützt 🔴 Keine spezifische Unterstützung für Automatisierungsprotokolle	🔴	🔴	🟡	🔴
	Plattforminteroperabilität	🟢 Plattform bietet standardisierte Schnittstellen, um 1) DT-Daten und Strukturinformationen abzurufen und 2) Änderungen an DTs für externe Dienste vorzunehmen 🟡 Nur eine dieser Optionen unterstützt 🔴 Keine dieser Optionen unterstützt	🟢	🟢	🟢	🟢
	Systeminteroperabilität	🟢 Plattform bietet Unterstützung für die Definition von Verbindungen zwischen verschiedenen Geräten über ihre DTs und für die automatische Implementierung von Interaktionen auf der Grundlage dieser Verbindungen 🟡 Nur einer dieser Aspekte verfügbar 🔴 Keiner dieser Aspekte verfügbar	🟡	🟡	🟡	🔴
Die Benutzerfreundlichkeit beschreibt das Ausmaß, in dem die Benutzerein System nutzen können, um ihre Ziele mit Effektivität und Effizienz zu erreichen.	Strukturierte Modellierung und Visualisierung	🟢 Die Plattform ermöglicht es Fachleuten, physikalische Geräte, Typen und Daten strukturiert zu modellieren und bietet eine grafische Schnittstelle 🟡 Plattform bietet Modellierungsfunktionen ohne grafische Benutzeroberfläche 🔴 Die Plattform unterstützt keine DT-Modellierung	🟢	🟢	🔴	🟢
Die Sicherheit bestimmt den Grad, in dem Daten und Verbindungen so gesichert sind, dass keine unberechtigten Zugriffe erfolgen können.	Verbindungs- und Datensicherheit	🟢 Die Plattform bietet irgendeine Form von Authentifizierungs- und Autorisierungsmechanismus beim Aufbau einer Verbindung zwischen einem DT und einem physischen System und einer sicheren lokalen Verbindung zwischen Geräten; die Plattform bietet Möglichkeiten zur Verschlüsselung von Daten, die an den DT gesendet werden, und von Daten, die auf dem DT selbst gespeichert sind 🟡 Die Plattform unterstützt nur eine der vorgenannten Möglichkeiten 🔴 Keine Unterstützung für Sicherheit	🟢	🟡	🟢	🟢

Qualitätsmerkmale	Anforderungen	Bewertungsschema	AZ *	AT	AWS *	EC *
Wartungsfreundlichkeit ist die Fähigkeit, sich an Änderungen der Umgebung und der Anforderungen an ein Softwareprodukt anzupassen.	Modifizierbarkeit	DT kann während der Laufzeit geändert werden DT kann geändert werden, während er inaktiv ist DT kann nach der Bereitstellung nicht mehr geändert werden				
	Wiederverwendbarkeit	Plattform bietet benutzerdefinierte und vorgefertigte Komponenten, die zwischen Projekten wiederverwendet werden können Komponenten können innerhalb eines Projekts wiederverwendet werden Komponenten sind nicht wiederverwendbar				
Portabilität beschreibt die Effizienz, mit der Hardware- und Software-Artefakte zwischen zwei Systemen übertragen werden können.	Kontinuierliche Integration (CI) und Kontinuierliche Bereitstellung (CD)	Plattform bietet komplette CI/CD-Pipeline oder Integration CI oder CD wird angeboten oder kann integriert werden Keine Unterstützung oder Integration				
	Bereitstellung	Plattform bietet Vor-Ort- und Cloud-native Lösungen für die Bereitstellung Nur Vor-Ort- oder Cloud-native Lösungen für die Bereitstellung verfügbar Nicht zutreffend				
			* Einordnung nach Lehner et al.,2022			

4.2 Analyse von Autodesk Tandem

Im folgenden wird zunächst die allgemeine Funktionsweise von Autodesk Tandem analysiert und bewertet. Daraufhin wird die protoypische Integration von betriebsspezifischen Daten untersucht. Dazu werden sowohl sich zeitlich verändernde, als auch Sensordaten betrachtet.

4.2.1 Allgemein

Es gibt zwei unterstützte Dateiformate zum Import von Gebäudemodellen in die Plattform. Entweder kann eine Revit-Datei oder eine IFC-Datei hochgeladen werden. Die Möglichkeit zum Import von IFC-Dateien ist positiv zu bewerten, da die Interoperabilität auch von Produkten außerhalb der Softwarelandschaft von Autodesk gegeben ist. Beim Import von Revit-Dateien ist dennoch mit einem geringeren Informationsverlust zu rechnen. Denn beim Austausch über IFC gehen parametrische Abhängigkeiten teils verloren (BORRMANN et al., 2021, S.143).

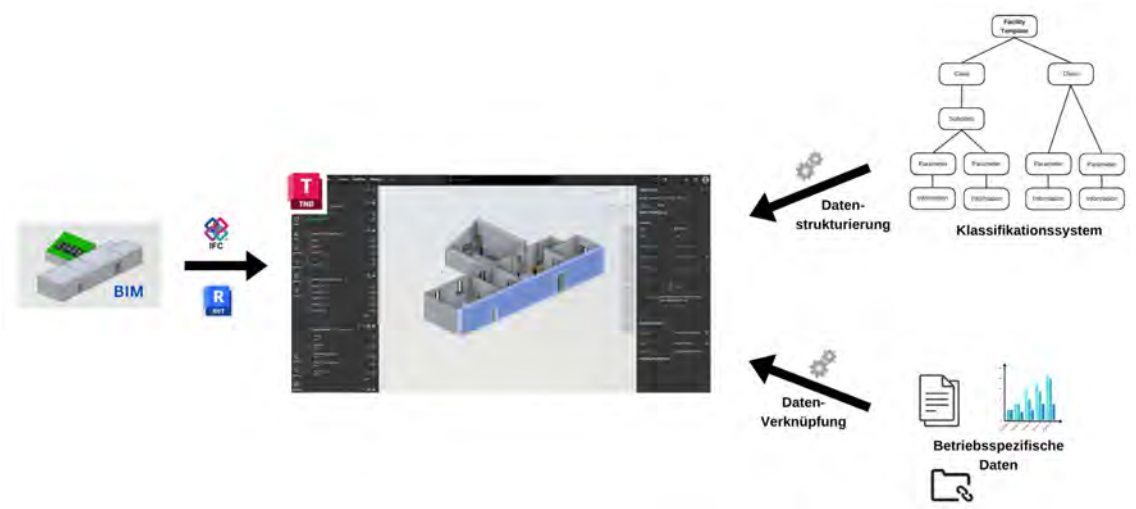

Abbildung 4.1: Grundlegende Funktionsweise von Tandem

Nach dem Import der Modelle können die Daten mittels eines Klassifikationssystems strukturiert werden. Außerdem können Betriebsspezifische Daten mit den Modellen verknüpft werden (vgl. Abb. 4.1).

Die Modelle werden einer Facility hinzugefügt. Jede Facility ist eine digitale Repräsentation eines Gebäudes (DT), welches sich gerade entweder in der Planungs- und Bau- oder in der Betriebsphase befindet. In Autodesk Tandem können mehrere Facilities erstellt und parallel betrieben werden. In Autodesk Tandem kann auf eine Facility ein Facility Template angewendet werden. Dabei handelt es sich um eine Vorlage, welche dazu genutzt wird, um bei Erstellen einer neuen Facility direkt ein Klassifikationssystem festzulegen. Dies kann durch die Nutzer:innen erstellt und modifiziert werden. Es können auch vordefinierte Standardklassifikationen (allerdings bisher nur auf Standard der USA) genutzt werden.

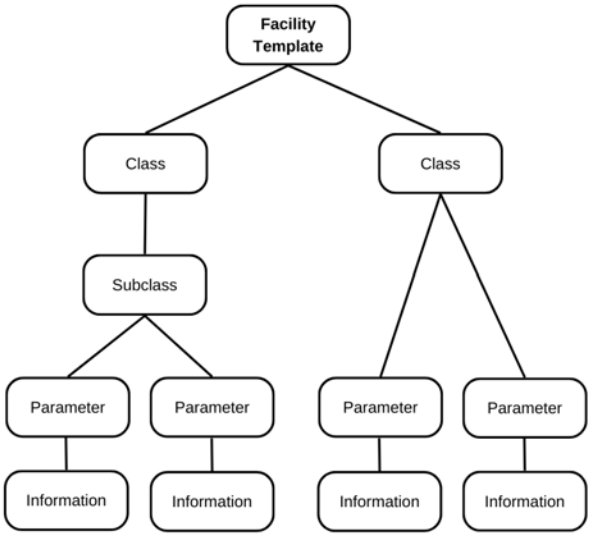

Abbildung 4.2: Struktur eines Facility Template

Wie in Abb. 4.2 zu erkennen ist, handelt es sich bei diesem Klassifikationssystem um ein hierarchisches System mit mehreren Ebenen, das das Konzept der Vererbung unterstützt. Es können also Eigenschaften bzw. Parameter von Eltern an ihre Kinder weitervererbt werden. Ein Parameter ist eine Property, welches eine spezifische Definition der Daten enthält, die mit einem Asset verknüpft sind. Dies können beispielsweise Name, Datentyp, Messwert, Einheit des Messwerts, Symbol, Genauigkeit, Beschreibung oder Kontext sein. Assets sind hier Objekte oder Elemente innerhalb einer Facility, welche im Kontext des FM betrachtet werden müssen, wie beispielsweise eine Tür, eine Pumpe oder ein Lüftungssystem. Asset Properties sind vom Nutzer in Autodesk Tandem erstellte Properties, welche innerhalb des AIM spezifiziert werden. Design Properties hingegen sind in Revit erstellte Properties, welche beim Import in Autodesk Tandem übernommen werden. Falls also Revit als BIM-Autorenwerkzeug während Planung und Bau verwendet wurde, können die hier definierten Eigenschaften und Informationen nahezu vollständig in den Betrieb übergeben und so erhalten werden.

Der Aufbau der Tandem Datenbank ist laut der Dokumentation (Github) wie eine dreidimensionale Matrix zu verstehen. Die Zeilen sind die Objekte im Modell, die Spalten deren Attribute und die dritte Dimension stellt die veränderten Werte der Property über die Zeit dar. Auf die Datenbank kann man derzeit über das Graphical User Interface (GUI) oder eine Representational State Transfer (REST) Application Programming Interface (API) zugreifen. Dabei handelt es sich um eine Schnittstelle zum Datenaustausch, die auf HTTP beruht. Der Produkt Roadmap ist zu entnehmen, dass derzeit die REST API verfügbar ist, als nächstes ein Javascript Software Development Kit(SDK) erscheinen wird und für die Zukunft Plug-Ins bzw. Erweiterungen geplant sind. Die Entwicklung der REST API ist noch nicht vollständig abgeschlossen, sie befindet sich noch in einer Testphase. Sie ist recht kompliziert zu bedienen und für neue Nutzer in der Handhabung nicht sehr intuitiv. Dies wird auch in der Dokumentation der API so beschrieben. Sie soll allerdings dadurch, dass sie nur als eine dünne Schicht über der Tandem Datenbank agiert, sehr effizient und effektiv sein (AUTODESK, 2023).

4.2.2 Bewertung

Die in 2.1.3 beschriebenen typischen technischen Komponenten einer CDE sind fast vollständig in Autodesk Tandem enthalten (vgl. Abb. 4.2).

Tabelle 4.2: Technische Komponenten von Tandem im Vergleich zu den typischen Komponenten einer CDE

Technische Komponente einer CDE	Autodesk Tandem
Datenverwaltung	✓
Strukturierte Ablage	✓
Klassifikationssystem	✓
Versionsmanagement	✓
Rechteverwaltung	✓
Statusverwaltung	✓
Filterfunktionen	✓
Workflows	X
Dashboards	✓
Visualisierung	✓

Tandem agiert als Cloud-basierte Plattform und ermöglicht den Nutzer:innen durch seine Datenverwaltung einen ortsunabhängigen Zugriff über das Internet. Alle Informationen des AIM werden durch eine strukturierte Ablage hier verwaltet und zusätzlich wird auch ein Klassifikationssystem verwendet. In Tandem wird eine Aufzeichnung von Änderungen zum Versionsmanagement durchgeführt, allerdings ist es nicht möglich auf frühere Versionen des DT zurück zu wechseln. Es können weitere Nutzer:innen zu einer Facility hinzugefügt und ihnen im Sinne der Rechteverwaltung unterschiedliche Rechte erteilt werden. Auch eine Art der Statusverwaltung ist enthalten, bei der überprüft werden kann, wie viele Assets in dem DT bereits mit Informationen versehen wurden. Mithilfe von Filterfunktionen können in Tandem benötigte Informationen herauskristallisiert werden. Dazu gibt es in Tandem verschiedenste Visualisierungsmöglichkeiten, wie in 4.2.4 bereits gezeigt wurde. Außerdem können auch Dashboards erstellt und gespeichert werden, welche die schnelle Darstellung von wichtigen Informationen ermöglichen.

Bei Tandem handelt es sich nach der Einstufung unter 2.1.4 um ein Integriertes Asset-Information-System. Es wird ein BIM-Modellserver verwendet, in dem die Modelle mithilfe eines DBMS zur Verfügung gestellt werden. Alle Informationen über das Bauwerk, auch Informationen aus dem Betrieb werden in diesem DBMS gespeichert. Über eine einheitliche Integrations- und Abfrageschicht können Nutzer:innen auf die Daten zugreifen. Es spielt für sie dabei keine Rolle, aus welchem System oder welcher Datenquelle sie Informationen benötigen.

Die DT-Plattform agiert als zentraler Knotenpunkt der Datenhaltung für alle am FM beteiligten Akteure und eingesetzten Systeme. Und sogar in Bezug auf das Erstellen und Pflegen eines DT von Baustellen gibt es erste Studien, die Tandem dazu als mögliche Software heranziehen (Reja & Varghese, 2022).

Allerdings sind an dieser Stelle auch Kritikpunkte zu äußern. Andere Programme und Systeme müssen mithilfe von Schnittstellen auf die Daten zugreifen können. Derzeit (Stand: November 2023) befinden sich die Schnittstellen noch in der Entwicklungsphase und können Änderungen unterworfen werden. Somit birgt es für andere Softwarehersteller

enorme Unsicherheiten, da sie nicht mit Sicherheit davon ausgehen können, dass der Datenaustausch einwandfrei funktioniert und dies auch nach Änderungen so bleiben wird.

Es sind für die Datenhaltung und den Datenaustausch während des Betrieb eines Gebäudes des Regelungen oder zumindest Absprachen zu treffen, um eine intakte CDE gewährleisten zu können.

Außerdem befindet sich die DT-PLattform derzeit lediglich auf der dritten Stufe der Evolutionsskala aus 2.2.2. Es werden die Modelle aus dem BIM mit Sensordaten verknüpft und visualisiert. Um auf die nächsthöheren Stufen zu gelangen müssten zunächst Analysetools und Algorithmen implementiert werden. Um auf die fünfte und letzte Stufe der Skala zu gelangen und somit den Status eines vollwertigen DT zu erreichen, müssste der DT dazu in der Lage sein, direkt auf Gegebenheiten innerhalb des Gebäudes automatisch reagieren zu können.

4.2.3 Integration von sich zeitlich verändernden Daten

Um die Integration von betriebsspezifischen Daten zu untersuchen, wird dies im folgenden anhand einiger konkreter Beispiele exemplarisch durchgeführt.

Für den Betrieb eines Universitätsgebäudes sind einige sich zeitlich verändernde Daten relevant. Wie beispielsweise die Raumbelegung durch Lehrstühle oder Schlüsselberechtigungen. Zur Integration dieser Daten wurde das Klassifizierungssystem um die notwendigen Klassen und Parameter erweitert. (vgl. Abb. 4.3)

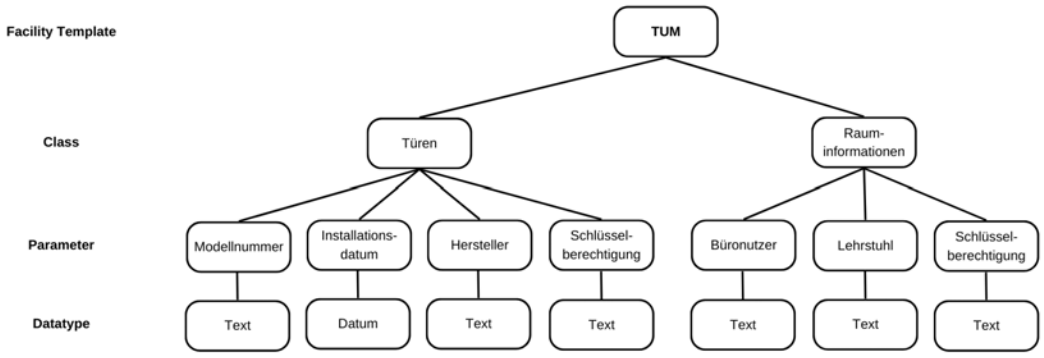

Abbildung 4.3: Struktur zur Integration von sich zeitlich verändernden Daten

In der TUM gibt es elektrische Türschlösser, welche durch Vorhalten eines Chips durch eine Person geöffnet werden können, falls diese dazu berechtigt ist. Um diese Funktionalität mit dem DT verknüpfen zu können, müssen nun diese Informationen an geeigneter Stelle

in das AIM integriert werden. Eine Möglichkeit wäre einen Parameter Schlüsselberechtigung zu erstellen und diesen im Facility Template der Klasse Tür hinzuzufügen. Nun kann über die GUI im 3D-Modell die jeweilige Tür ausgewählt und unter ihren Asset Properties der Parameter modifiziert werden. Beispielsweise könnte man hier eine Liste der Namen oder Positionen aufführen, welche berechtigt sind diese Tür zu öffnen (vgl. Abb. 4.4).

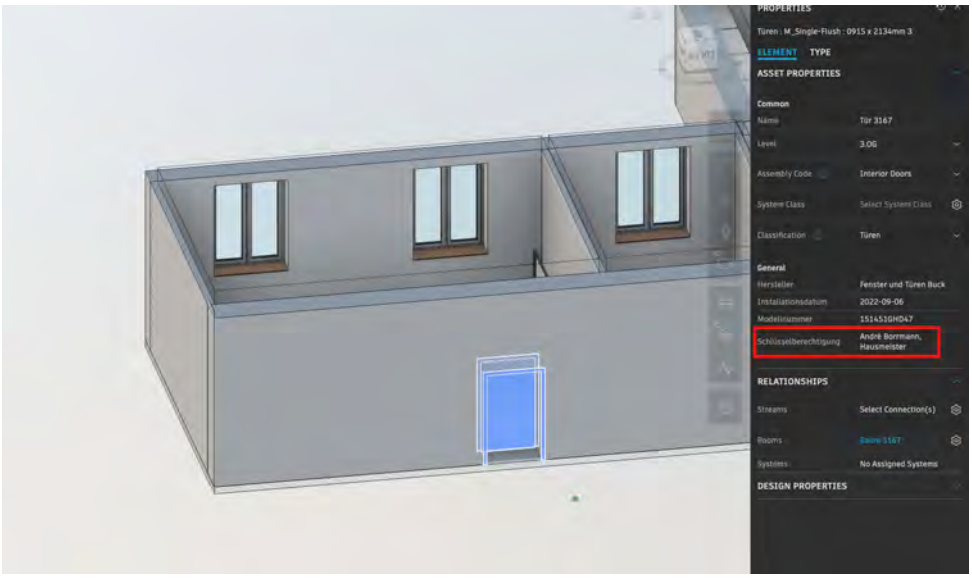

Abbildung 4.4: Asset Properties einer Tür in Autodesk Tandem

Ähnlich kann beim Thema Raumbelegung durch Lehrstühle verfahren werden. Denn es besteht die Möglichkeit Parameter nicht nur physischen Elementen hinzuzufügen, sondern auch beispielsweise Räumen. Durch Einstellen des Kontextes des Parameters von Element auf Space kann der Parameter auch einem ganzen Raum anstatt nur einem einzelnen Element zugewiesen werden. Außerdem muss natürlich im Facility Template eine Klasse, wie hier beispielsweise Rauminformationen, enthalten sein, die den erstellten Parameter Raumbelegung enthält (vgl. Abb. 4.4).

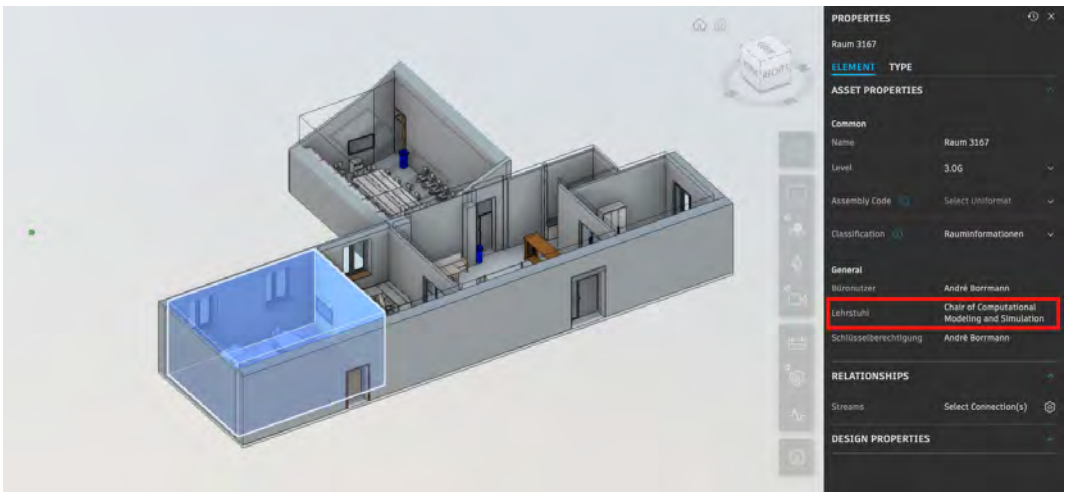

Abbildung 4.5: Asset Properties eines Raums in Autodesk Tandem

4.2.4 Integration von Sensordaten in Autodesk Tandem

Zur Integration von Echtzeitdaten wird in Autodesk Tandem ein sogenannter Stream erstellt. Ein Stream wird in der Tandem Datenbank wie ein normales Element erstellt, klassifiziert und gelöscht. Der Unterschied besteht darin, dass sie über eine Uniform Resource Locator (URL) aktualisiert werden. Diese URL ist für jeden Stream individuell und kann von außerhalb genutzt werden, um dorthin Sensordaten zu schicken. Die Streams agieren laut der Dokumentation von Tandem wie Leitungen zum Sammeln und Speichern von Zeitreihendaten. Derzeit gibt es zwei Möglichkeiten, um in Tandem einen Stream einzurichten: Entweder wird der Stream über das GUI erstellt und verwaltet und die REST API wird nur verwendet, um die Sensordaten in dem Stream auf dem neuesten Stand zu halten. Oder die Tandem API wird verwendet, um den Stream durch Programmierung zu erstellen, zu verwalten und neue Sensordaten abzuspeichern. Allerdings müssen auch hierbei einige Schritte in der GUI durchgeführt werden.

Um die Sensordaten mit dem Modell zu verknüpfen, müssen innerhalb des Facility Template passende Parameter erstellt werden (vgl. Abb. 4.6).

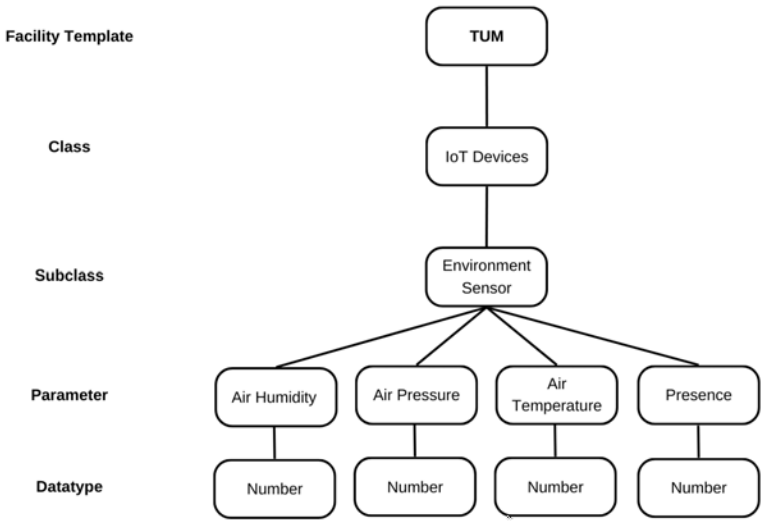

Abbildung 4.6: Struktur zur Integration von Sensordaten

In diesem Fall wurde unter der Klasse IoT Devices die Subklasse Environment Sensor erstellt. Diese enthält mehrere Parameter, die die Umgebung bzw. den derzeitigen Zustand innerhalb eines Raums repräsentieren.

Nun kann die Verknüfung der Daten innerhalb der GUI durchgeführt werden. Dazu wählt man den Stream aus und wählt "configure Stream". Die über den Stream eintreffenden Daten sollte nun in dem sich öffnenden Fenster zu sehen sein und die Zuweisung kann durchgeführt werden.

Abbildung 4.7: Konfigurierung eines Streams in Tandem

Auf der linken Seite werden die auf dem Stream eintreffenden Daten angezeigt, auf der rechten die Parameter innerhalb der Datenbank (vgl. Abb. 4.7). Die Daten treffen dabei im JavaScript Object Notation (JSON)-Format ein. Die Generierung der Daten und das Versenden an die Tandem Datenbank werden in 5 beschrieben. Per Mausklick können die Sensordaten mit dem jeweiligen Parameter verknüpft werden. Durch Zuweisung der Entität des konkreten Environment Sensor zu einem Raum, können die Daten innerhalb des Gebäudemodells verortet werden.

Nach dem Einrichten des Streams können die Daten in der GUI angezeigt und visualisiert werden. Beispielsweise kann mithilfe einer sogenannten Heatmap der aktuelle Zustand eines bestimmten Parameters, wie der Lufttemperatur, über das gesamte Gebäude hinweg angezeigt werden. Vorausgesetzt es sind mehrere Sensoren in dem Gebäude vorhanden. Da in diesem Fall testweise nur ein Raum mit Sensoren ausgestattet wurde, kann auch nur die Temperatur innerhalb eines Raums dargestellt werden. (vgl. Abb. 4.8)

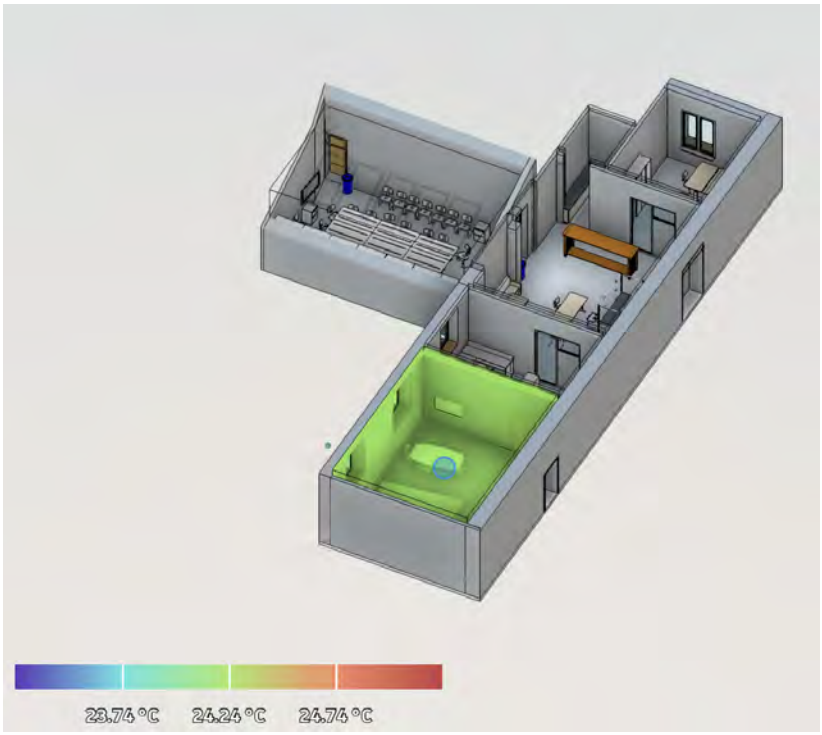

Abbildung 4.8: Visualisierung eines Parameters mittels einer Heatmap

Um den Verlauf eines Parameters über die Zeit zu visualisieren, kann dieser mittels eines Diagramms dargestellt werden (vgl. Abb. 4.9)

Abbildung 4.9: Grafische Darstellung des Temperaturverlaufs in Tandem

Da über den zeitlichen Verlauf Mittelwerte dargestellt werden, sind die Werte der Präsenzsensoren in dieser Darstellung verzerrt (vgl. Abb. 4.10). Nur der Wert 0 für keine Person erkannt oder der Wert 1 für Person erkannt ergeben einen Sinn.

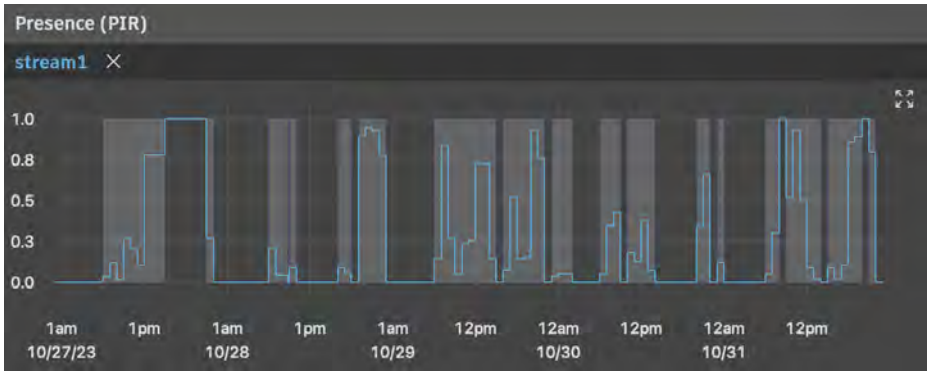

Abbildung 4.10: Grafische Darstellung des Verlaufs der Präsenzdaten in Tandem

Allerdings lässt sich auch in dieser Darstellung bereits grob erkennen, zu welchen Zeiten sich Personen in diesem Raum aufgehalten haben. Zur genaueren Analyse wäre es vermutlich notwendig die Daten zu extrahieren und außerhalb der Datenbank auszuwerten.

Kapitel 5

Umsetzung und Experiment der Sensorik

In diesem Kapitel wird das Vorgehen zur Einrichtung der Generierung und dem Versenden der Sensordaten beschrieben. Dazu wird zunächst die Systemarchitektur erläutert. Daraufhin wird die Hardware und Software einer Sensor Node beschreiben. Zuletzt wird das Einrichten des Gateways, in diesem Fall einer Version von openHAB auf einem Raspberry Pi, beschrieben.

5.1 Systemarchitektur

Im Allgemeinen macht es nur Sinn, ein Gebäude mit einer Vielzahl von Sensoren auszustatten. Denn um den aktuellen Zustand eines Gebäudes und dessen zeitlichen Verlauf wirklich abzubilden, zu interpretieren und gegebenenfalls Änderungsvorschläge für den Betrieb machen zu können, bedarf es einer differenzierten Betrachtung mehrerer Parameter an möglichst vielen Punkten im Gebäude. Da die Konfigurierung weiterer Sensor Knotenpunkte allerdings analog zum Konfigurieren einer einzelnen Sensor Node abläuft, wird im Folgenden nur die Einrichtung der Datengenerierung und -übertragung von nur einer Sensor Node betrachtet. In Abbildung 5.1 ist die schematische Darstellung der Sytemarchitektur dargestellt.

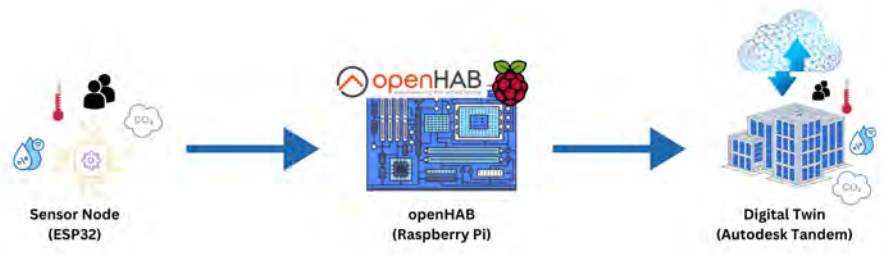

Abbildung 5.1: Schematische Darstellung der System Architektur

Es werden verschiedene Messwerte auf der Sensor Node generiert, gesammelt und per MQTT an das openHAB (Raspberry Pi), welches als Gateway bzw. zentraler Knotenpunkt fungiert, gesendet. Von diesem werden dann die Daten mittels HTTP an die URL des Streams aus 4.2.4 versandt.

5.2 Sensor Node

Im folgenden Unterkapitel wird das Vorgehen zur Programmierung einer Sensor Node beschrieben. Dazu muss zunächst eine Firmware und damit verbunden eine Programmiersprache gewählt werden. Es werden im folgenden einige Beispiele genannt und im Falle von MicroPython die Einrichtung einer Entwicklungsumgebung beschrieben.

5.2.1 Firmware

Zunächst muss eine Firmware auf den ESP32 übertragen werden, um darauf selbst erstellte Programme starten zu können. Es gibt mehrere Möglichkeiten bei der Wahl einer Firmware für den Mikrocontroller, beispielsweise:

- Arduino
- ESPHome
- Micropython

Arduino ermöglicht es in der Programmiersprache C in Kombination mit der Arduino IDE eigene Lösungen zu entwickeln. Bei ESPHome kann durch recht einfach aufgebaute Konfigurierungsdateien die individuelle Einrichtung des Mikrocontrollers erfolgen. Im folgenden Versuch wird die ESP32-Version von MicroPython als Firmware verwendet. Dabei handelt es sich um eine Implementierung der Python Programmiersprache, welche für die Verwendung auf Microcontrollern optimiert wurde. Dies bietet hohe Flexibilität in der Programmierung gleichzeitig aber auch die einfachere Syntax dieser Hochsprache. Dadurch können auch Programmierneulinge mit geringem Vorwissen eigene Softwarelösungen für IoT-Geräte umsetzen.

Zunächst sollte eine aktuelle Version von Python auf dem Computer installiert werden, welche auf https://www.python.org heruntergeladen werden kann. Um die Firmware auf den ESP32 zu übertragen, wird das esptool.py benötigt. Dabei handelt es sich um ein open-source, plattformunabhängiges Werkzeug, welches es ermöglicht mit dem ROM bootloader des Espressif Chips zu kommunizieren (AHLBERG, GRATTON et al., n. d.). Dieses kann mithilfe von pip installiert werden. Bei pip handelt es sich um ein Paketverwaltungsprogramm für Python, welches bei der Installation von Python bereits inkludiert ist.

Nun kann man ein Terminalfenster öffnen und mit folgendem Befehl das esptool durch pip herunterladen und installieren.

```
python3 -m pip install esptool
```

Bevor die Firmware überspielt wird, sollte allerdings zuvor der Speicher gelöscht werden. Dazu muss der Port bestimmt werden, über den der ESP32 mit dem Computer verbunden

ist. Je nach verwendetem Betriebssystem variiert die Bestimmung des Ports. Während des Löschens des Speichers muss auf dem Esp32 der BOOT-Knopf für einige Sekunden gedrückt werden. Bei Eingabe des folgenden Befehls wird der Speicher gelöscht (der Port muss individuell geändert werden):

```
esptool.py --chip esp32 --port <serial_port> erase_flash
```

Anschließend muss noch die passende MicroPython Firmware von https://micropython.org/download/ESP32_GENERIC/ heruntergeladen und lokal abgespeichert werden. Mithilfe des folgenden Befehls wird diese auf den Microcontroller übertragen. Wieder muss dabei der BOOT-Knopf auf dem ESP32 für einige Sekunden gedrückt werden.

```
esptool.py --chip esp32 --port <serial_port> write_flash -z 0x1000 <
    esp32-X.bin>
```

5.2.2 Integrierte Entwicklungsumgebung(IDE)

Es gibt verschiedene Wege eigene Programme zu entwickeln und auf den Microcontroller zu übertragen.

PyCharm

Eine Möglichkeit ist PyCharm als Entwicklungsumgebung zu verwenden. Diese Integrated Development Environment (IDE) bietet zahlreiche Vorteile, die das Programmieren mit Python vereinfachen und effektiver macht.

In PyCharm muss zunächst in den Einstellungen das Plugin MicroPython heruntergeladen und installiert werden.

Nach dem Erstellen eines neuen Projekts musss in den Einstellungen unter Languages and Frameworks > MicroPython die Verbindung zu dem Microcontroller eingerichtet werden. Als Gerätetyp steht nicht der ESP32 sondern nur der ESP8266 zur Auswahl, welcher das Vorgängermodell des ESP32 ist. Außerdem muss der Port, bzw. der Gerätepfad angegeben werden, dieser ist derselbe wie oben.

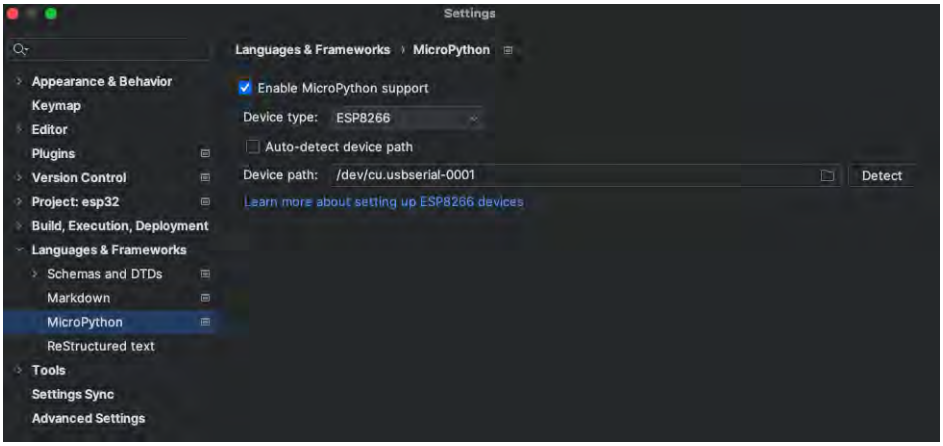

Abbildung 5.2: Beispielhafte Gerätekonfiguration in Pycharm

Leider ist das Plugin für PyCharm allerdings sehr fehleranfällig. An einem Computer war es dem Autor nicht möglich sich mit dem Microcontroller zu verbinden. Auch die Rezensionen im Marketplace von PyCharm fallen zum Teil sehr negativ aus. Einige Funktionen arbeiten teils nur mangelhaft bis gar nicht.

Für Visual Studio Code existiert das Plugin "Pymakr". Dieses wurde durch den Autor nicht getestet, aber auch hier fallen die Rezensionen schlecht aus und bemängeln die Funktionsweise des Plugins. Die aktuelle Version der Software (2.22.5) stammt vom 16.08.2022 und wurde somit lange nicht mehr aktualisiert (Stand: 13.10.2023). Somit ist die Verwendung von komplexeren IDEs in Kombination mit Micropython stark eingeschränkt. Dies ist sehr zu bedauern, da man in den einfacheren IDEs auf zahlreiche unterstützende Funktionen beim Programmieren verzichten muss.

Thonny IDE

Bei der Thonny IDE handelt es sich um eine simple und recht übersichtliche IDE. Die Programmierung von Microcontrollern mit MicroPython ist hier allerdings wesentlich besser integriert als beispielsweise in PyCharm.

In den Einstellungen musss zunächst als Interpreter MicroPython(ESP32) ausgewählt werden. In der Liste der auswählbaren Ports ist der zum Microcontroller verbundene Port zu wählen. Danach kann über dieses Fenster direkt die MicroPython-Firmware auf dem ESP32 installiert oder aktualisiert werden.

Die Skripte können in einem Editorfenster erstellt und dann direkt auf dem Esp32 abgespeichert werden. Auch können Skripte aus dem Editor direkt abgespielt werden, was das Debugging und Testen des Codes vereinfacht und somit das Einrichten des Sensor Node beschleunigt.

Was sich im Endeffekt als das beste Vorgehen herausgestellt hat, war zunächst in Pycharm den Code zu entwerfen und ihn dann mit Thonny auf den ESP32 zu übertragen und zu debuggen. In PyCharm hat man, wie bereits erwähnt, ein bessere Unterstützung

beim Programmieren. In Thonny hingegen ist die Schnittstelle zum Microcontoller besser umgesetzt.

5.2.3 Programmierung

Im folgenden Kapitel wird die Software der Sensor Node erläutert. Dabei wird zunächst auf die Struktur eingegangen, dann auf die Integration der Sensoren und des Displays, die Verbindung zum Netzwerk, die Hauptschleife und zuletzt auf das Senden der Daten per MQTT.

Struktur

Beim Verbinden des ESP32 mit einer Stromversorgung wird die boot.py Datei automatisch abgespielt. Um nun ein autark laufendes System zu generieren, welches eine bestimmte Funktionalität erfüllt, müssen in dieser Datei die gewollten Funktionen implementiert werden. Es können auf dem ESP32 auch mehrere Dateien gespeichert werden und diese dann innerhalb der boot-Datei importiert werden, um deren Funktionen zu nutzen.

In unserem Fall befinden sich drei Dateien auf dem Microcontroller:

- boot.py
- bme680.py
- sh1106.py

In Boot.py befindet sich der selbst entwickelte Code, welcher die gewünschte Funktionsweise der Sensor Node implementiert. Bei der Datei bme680.py handelt es sich um eine Library zum Ablesen der Sensordaten des BME680 (LIMOR 'LADYADA' FRIED OF ADAFRUIT AND JEFF RABER, 2017). Die Datei sh1106.py enthält eine Library zur Integration des Displays (DOPIERALSKI et al., 2021).

Integration der Geräte

An den ESP32 werden folgende Geräte angeschlossen:

- Sensor für das Raumklima (BME 680)
- Bewegungsensor (HC-SR501)
- Präsenzsensor (HLK-LD2410)
- Display (SH1106)

Zur Messung des Raumklimas wird ein BME680 von Bosch verwendet. Mit diesem Sensor können Lufttemperatur, Luftfeuchtigkeit, Luftdruck und Gaswiderstand gemessen werden. Um daraus die Luftqualität zu berechnen, muss der gemessene Gaswiderstand mit der Luftfeuchtigkeit ins Verhältnis gesetzt werden. Von Bosch gibt es dazu die BSEC Library, um auf Grundlage der rohen Messwerte einen Indoor Air Quality(IAQ)-Index zu kalkulieren (BOSCH SENSORTEC GMBH, 2023). Weiterhin ermöglicht diese Library die Genauigkeit der Messwerte zu verbessern. Diese Library ist allerdings nicht direkt in Micropython verfügbar und konnte daher hier nicht integriert werden.

Um Rückschlüsse darauf ziehen zu können, ob sich eine Person in einem Raum befindet, werden zwei verschiedene Sensoren kombiniert. Ein HC-SR501 und ein HLK-LD2410. Bei dem HC-SR501 handelt es sich um einen PIR Sensor. Der HLK-LD2410 ist ein Radar Sensor.

Beide Sensoren liefern einen Boolean-Wert: also Wahr oder Falsch, ob sich eine Person in dem Raum befindet. Allerdings lässt sich in Tandem kein Boolean-Wert übertragen. Daher wird ein Binärcode verwendet, als 0 für keine Person und 1 für mindestens eine Person erkannt. Um eine recht zuverlässige Vorhersage treffen zu können, sollten die Sensoren so plaziert werden, dass sie den Großteil des Raumes abdecken. Dazu sollte der Sensor idealerweise an der Wand etwa auf Augenhöhe oder mittig an der Decke angebracht werden. Da an die Sensor Node ein Display zum Ablesen der Raumklimadaten angebracht wird, ist der Anbringungsort an einer Wand ideal, da dann auch von Personen gut die Informationen vom Display abgelesen werden können.

Das Display und der BME680 sind beide über I^2C mit dem Microcontroller verbunden. Bei I^2C handelt es sich um einen Datenbus. Durch serielle Datenübertragung können über zwei Kabel bis zu 128 Geräte miteinander kommunizieren. Die eine Leitung wird als SDA bezeichnet. Dies steht für serial data. Hier werden die Daten seriell übertragen. Über die zweite Leitung (SCL = serial clock) werden Taktimpulse gesendet.

Die beiden Präsenzsensoren werden lediglich mit einem Pin verbunden. Falls sie eine Person erkennen, geben sie über dieses Kabel Strom aus.

```
# Festelegen der Pins für die Sensoren und das Display

bme_i2c = I2C(1, scl = Pin(25), sda = Pin(26))
bme = BME680_I2C(bme_i2c)

dis_i2c = I2C(scl=Pin(22), sda=Pin(19), freq=400000)
display = SH1106_I2C(128, 64, dis_i2c, Pin(16), 0x3c)
display.sleep(False)

pir_sensor = Pin(15, Pin.IN)
ld2410 = Pin(16, Pin.IN)
```

Abbildung 5.3: Integration der Geräte

Main loop

Zunächst werden die Messwerte von den Sensoren ausgelesen. Bei dem Klimasensor erfolgt dies mit Hilfe der in der Library bme680.py implementierten Funktionen für die

jeweiligen Messwerte. Temperatur, Luftfeuchtigkeit und Luftdruck werden dabei auf zwei Nachkommastellen gerundet. Die beiden Präsenzsensoren geben, falls sie eine Person detektiert haben, auf dem Signalkabel Strom aus. Durch die value-Funktion des zugewiesenen Pins kann nun abgefragt werden, ob hier Strom vorliegt. Falls Strom anliegt, wird die jeweilige Variable für die Präsenz auf den Wert 1 gesetzt.

```
# Messwerte von Sensoren ablesen
temp = round(bme.temperature, 1)
hum = round(bme.humidity, 1)
pres = round(bme.pressure, 1)
gas = bme.gas
if ld2410.value() == 1:
    radar = 1
if pir_sensor.value() == 1:
    pir = 1

# Messwerte in Listen abspeichern
temp_list.append(temp)
hum_list.append(hum)
pres_list.append(pres)
gas_list.append(gas)
```

Abbildung 5.4: Auslesen der Messwerte

Zur späteren Kalkulation des Mittelwerts werden die Messdaten des Raumklimas pro Messwert in einer Liste abgespeichert. Im Anschluss daran werden die Raumklimadaten auf dem Display ausgegeben (5.5).

```
# Messwerte auf dem Diplay ausgeben
display.fill(0)
display.text('Temperature', 0, 0, 1)
display.text(str(temp)+ ' C', 0, 10, 1)
display.text('Humidity', 0, 25, 1)
display.text(str(hum)+' %', 0, 35, 1)
display.show()
time.sleep(4)

display.fill(0)
display.text('Pressure', 0, 0, 1)
display.text(str(pres) + ' kPa', 0, 10, 1)
display.text('Gas resistance', 0, 25, 1)
display.text(str(gas), 0, 35, 1)
display.show()
time.sleep(4)
```

Abbildung 5.5: Ausgeben der Daten auf dem Display

Aus Platzgründen auf dem Display werden zuerst Temperatur und Luftfeuchtigkeit für 4 Sekunden angezeigt. Danach Luftdruck und Gas Widerstand ebenfalls für 4 Sekunden. Somit werden in etwa alle 8 Sekunden Messwerte generiert und abgespeichert.

```
# Bei überschreiten des Timers von 60 Sekunden: Daten senden
delta = time.ticks_diff(time.ticks_ms(), start)
if delta > 60000:
    send_data()
    start = time.ticks_ms()
```

Abbildung 5.6: Timer für das Versenden der Daten

Falls sich der Timer bei unter 60 Skekunden befindet, beginnt das Programm von vorne. Falls sich allerdings der Timer bei über 60 Sekunden befinden sollte, werden zunächst die

Daten gesendet, danach der Timer neu gestartet und erst dann beginnt das Programm von vorne.

Versenden der Daten

Wie in Abbildung 5.6 bereits zu erkennen ist, wird beim Überschreiten des Timers von 60 Sekunden, das Senden der Daten durch die Funktion send_data() ausgeführt.

Innerhalb dieser Funktion wird zunächst für Temperatur, Luftfeuchtigkeit, Luftdruck und Gaswiderstand die Mittelwerte der letzten 60 Sekunden auf Grundlage der in den Listen abgespeicherten Messwerte gebildet. Daraufhin werden die Boolean-Werte des Radar und des PIR-Sensors integriert. Diese liefern jeweils den Wert True, falls einmal innerhalb der letzten Minute ihr zugehöriger Sensor eine Person wahrgenommen hat.

Die Mittelwerte mit den beiden Boolean-Werten werden nun zuerst in einem Dictionary abgespeichert. Ein Dictionary ist ein Datentyp, bestehend aus Schlüssel-Objekt Paaren (vgl. Abb. 5.7). Dieses Dictionary wird daraufhin in eine JSON-Datei umgewandelt und mittels MQTT verschickt.

Zuletzt werden noch alle Messwerte aus den Listen gelöscht und die Variablen der Präsenzsensoren auf False gesetzt.

```
def send_data():
    # Mittelwert der letzten 60 Sekunden bilden
    temp_m = round(sum(temp_list) / len(temp_list), 1)
    hum_m = round(sum(hum_list) / len(hum_list), 1)
    pres_m = round(sum(pres_list) / len(pres_list), 1)
    gas_m = round(sum(gas_list) / len(gas_list))

    global radar
    global pir

    data = {"status": "ON", "temperature": temp_m, "humidity": hum_m, "pressure": pres_m,
            "air_quality": gas_m,"radar": radar, "pir": pir}
    data = json.dumps(data)
    try:
        publish_msg('raum_3165', data)
    except Exception as e:
        display.text('MQTT failed', 0, 50, 1)
        print(e)

    temp_list.clear()
    hum_list.clear()
    pres_list.clear()
    gas_list.clear()
    radar = False
    pir = False
```

Abbildung 5.7: Senden der Daten

5.2.4 Prototypenbau

Um die Funktionsweise des Microcontrollers, der Sensoren und des Displays, sowie deren Zusammenspiel zu analysieren, wurden die Geräte zunächst mittels eines Breadboards und DuPont Kabel miteinander verbunden.

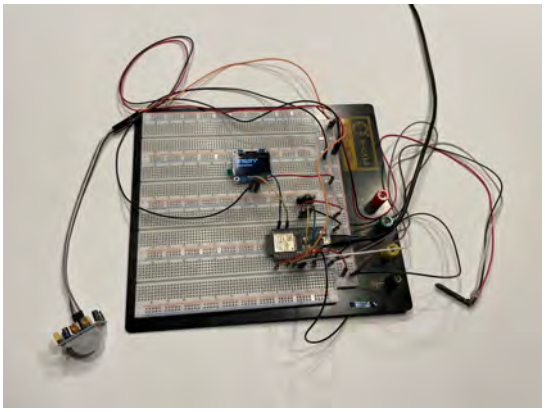

Abbildung 5.8: Prototypenbau auf einem Breadboard

Die Vekabelung der Geräte erfolgte wie in Abb. 5.9 dargestellt. Dies entspricht ebenfalls der Verkabelung des fertigen Prototypen.

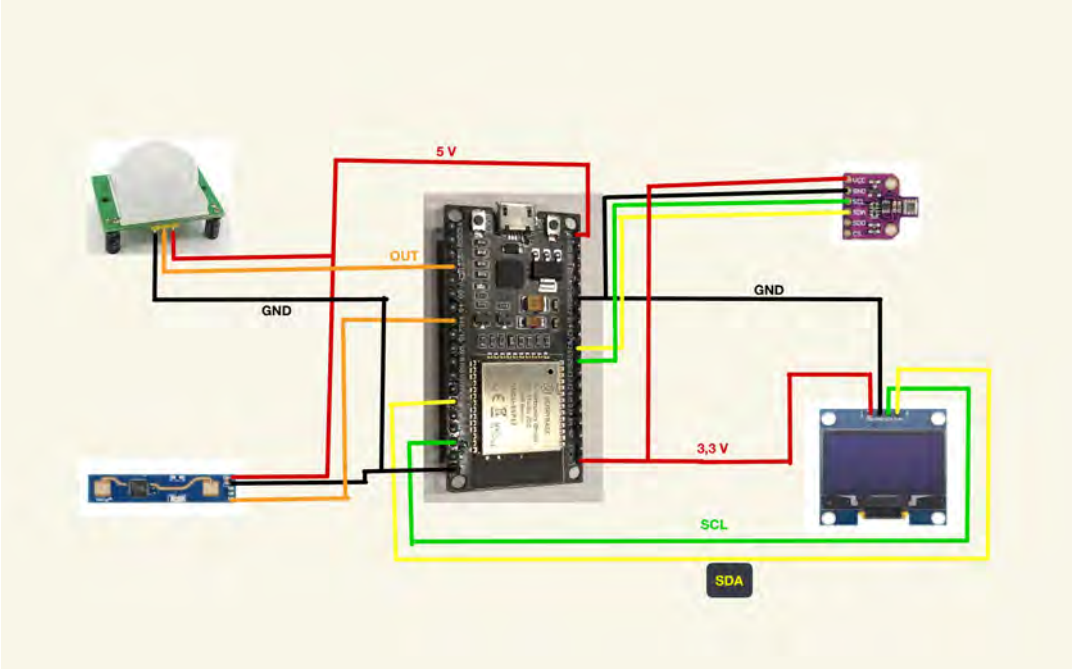

Abbildung 5.9: Verkabelung der Geräte

Die beiden Präsenzsensoren (links im Bild) benötigen eine Spannung von 5V und werden daher mit dem zugehörigen Pluspol des ESP32 verbunden. Zusätlich müssen sie mit Ground(Masse) und ihr Output-Pol mit dem ihm zugehörigen Pin verbunden werden (vgl. Abb. 5.3). Der BME680 und das Display hingegen müssen mit 3,3V, Masse, SCL und SDA auf den jeweils zugewiesenen Pins verbunden werden (vgl. Abb. 5.3).

Als nächstes wurden auf einer Lochrasterplatine ein Sockel für den Microcontroller sowie Anschlüsse für die Sensoren und das Display verlötet (vgl. Abb. 5.10). Diese wurden, wie in Abb. 5.9 dargestellt, miteinander verbunden.

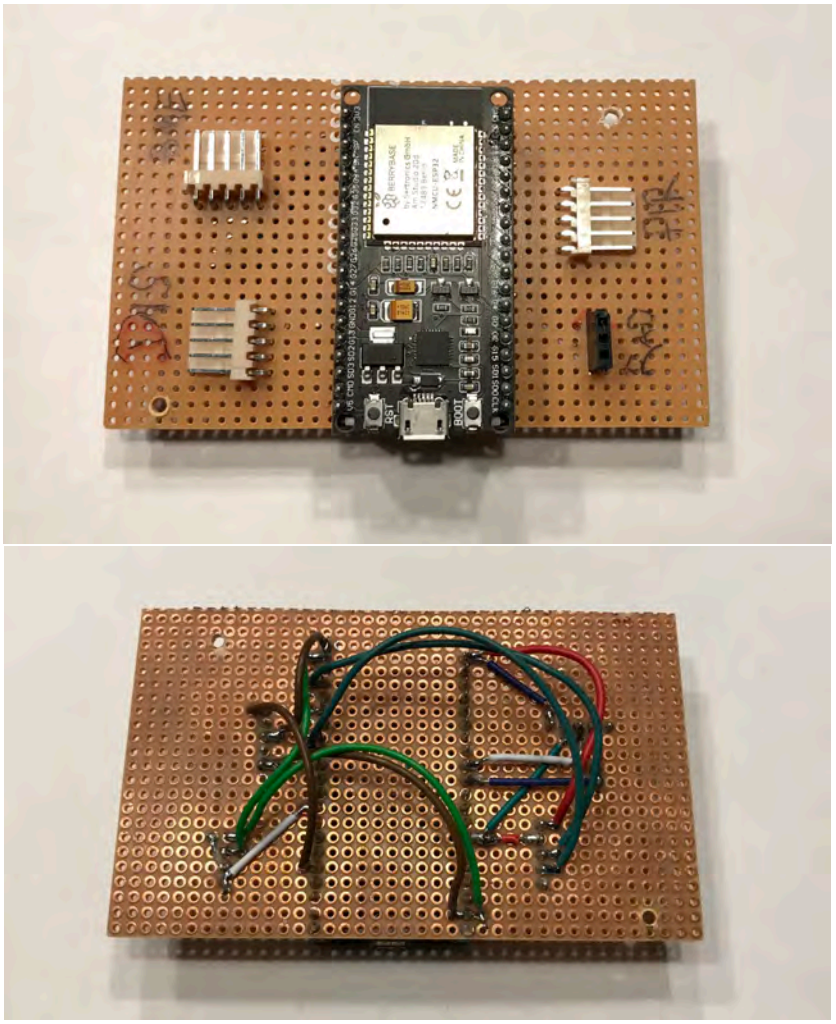

Abbildung 5.10: Platine mit Microcontroller und Buchsen

Zuletzt wurde die Platine innerhalb eines Gehäuses plaziert und passende Ausschnitte für Sensoren und Display angebracht. Durch mit passenden Steckern versehene Kabel kann nun die Elektronik einfach miteinander verbunden werden. Bei Verbindung des fertigen Prototyps mit einer Stromversorgung wird das in 5.2.3 beschriebene Programm automatisch gestartet (vgl. Abb. 5.11).

Abbildung 5.11: Fertiger Prototyp

5.3 Gateway - Raspberry Pi mit openHAB

5.3.1 Setup

Als zentraler Knotenpunkt wird auf einem Raspberry Pi 4 eine Version von OpenHAB betrieben. OpenHAB steht für Open Home Automation Bus. Es ist eine Open-Source-Plattform, welche die Einrichtung und Einbindung verschiedener Systeme im Bereich Home Automation unterstützt.

Zuerst muss auf der SD-Karte des Raspberry Pi ein startfähiges Image von OpenHABian (dies ist die Version für den Raspberry Pi) installiert werden. Eine detaillierte Anleitung dazu stellt OpenHAB zur Verfügung.

Nachdem die SD-Karte beschrieben wurde, kann sie in den Raspberry Pi eingelegt werden und dieser gestartet werden. Es wird lediglich eine Netzwerkverbindung (am besten per Ethernet-Kabel) benötigt. Beim starten wird automatisch die Initalisierung des OpenHABian durchgeführt. Dies dauert ca. 15-45 Minuten. Nach dieser Initialisierungsphase ist das OpenHAB unter http://openhabian:8080 erreichbar. Falls nicht, sollte es unter der Ip-Addresse des Servers erreichbar sein. Also beispielsweise: http://192.168.3.253:8080. Die IP-Adresse wird beim Anschließen eines Bildschirms an den HDMI-Port des Raspberry Pi angezeigt.

Beim Aufruf der Seite des Servers muss zunächst ein erster Benutzer angelegt werden und ein kleineres Setup von verschiedenen Parametern durchgeführt werden.

5.3.2 Grundlegender Aufbau von OpenHAB

Um die in der Umsetzung verwendeten Begriffe zu verstehen, ist ein grundlegendes Verständnis der Konzepte in OpenHAB notwendig. Dazu folgt nun eine kurze Übersicht der wichtigsten Begriffe:

- **Bindings**: Ein Binding stellt das Interface zur Verfügung, um mit Geräten zu interagieren.
- **Things** stellen die erste digitale Repräsentation eines Geräts innerhalb von OpenHAB dar. Things müssen aber nicht zwangsläufig physische Gegenstände sein. Sie können auch einfach nur die Repräsentation einer Funktionalität oder eines Services, wie beispielsweise des MQTT-Brokers, sein.
- **Channels** stellen eine Verbindung zwischen Thing und Item her.
- **Items** sind die digitale Repräsentation der Informationen eines Geräts.
- **Rules** sind automatische Reaktionen/ Skripte die unter gewissen Bedingungen (z.B. alle 30 Sekunden oder bei Veränderung eines Wertes) durchgeführt werden.
- **Pages** sind ein User Interface(UI), über das Informationen dargestellt werden und Interaktionen mit dem User stattfinden.

5.3.3 Einrichten der Datenübertragung

Als MQTT-Broker wird Mosqitto verwendet. Dieser kann direkt über OpenHABian geladen und installiert werden. Dazu muss man allerdings auf die Konsole zugreifen. Unter openhabian-config können zusätzliche Komponenten heruntergeladen werden. Hier ist auch Mosquitto zu finden. Nach der Installation kann ein MQTT Binding als Thing erstellt und konfiguriert werden. Bei der Konfigurierung können viele Einstellungen vorgenommen werden, aber im Grunde werden nur die IP-Adresse des Brokers, sowie Benutzername und Passwort für die MQTT-Verbindung benötigt. Exemplarisch wird nun die Einrichtung eines Raumes beschrieben. Die Umsetzung für weitere Räume läuft analog dazu ab.

In diesem Fall wurde zunächst ein Thing erstellt, welches die Sensor Node in dem Raum 3165 repräsentiert. Diese pubilshed unter dem Topic 3165 eine JSON-Datei, welche die gesamten Sensorinformationen enthält, wie etwa:

$$\{"humidity" : 49.68, "temperature" : 24.52, "pressure" : 966.14\}$$

Wenn man diese Daten nun in openHAB auswerten möchte, muss pro Messwert ein Channel eingerichtet und konfiguriert werden. Dazu wird das Addon "JSON Path Transformation"benötigt, welches die Aufteilung des kompletten JSON in die einzelnen Messwerte ermöglicht. Zur späteren Verwendung in Autodesk Tandem ist es allerdings einfacher die JSON-Datei im Ganzen weiterzuleiten. Denn, wie in 4.2.4 beschrieben, wird dort als Input ebenfalls eine JSON-Datei benötigt. Die Aufteilung erfolgt dort ebenfalls auf Grundlage

einer JSON Path Transformation. Somit kann man nur einen Channel einrichten, der den kompletten JSON als einen String, also als Text betrachtet (vgl. Abb. 5.12).

```
UID: mqtt:topic:76d6b3cee7:room_3165
label: room_3165
thingTypeUID: mqtt:topic
configuration:
  availabilityTopic: "3165"
bridgeUID: mqtt:broker:76d6b3cee7
channels:
  - id: json
    channelTypeUID: mqtt:string
    label: json
    description: ""
    configuration:
      stateTopic: "3165"
```

Abbildung 5.12: Thing (Raum 3165) in OpenHAB

Dieser Channel muss nun noch mit einem Item verlinkt werden. Das Item stellt dann die Repräsenation der Sensordaten innerhalb von OpenHab dar. Beim Eintreffen neuer Daten von der SensorNode wird somit das Item geupdatet. Dieses Ereignis kann nun genutzt werden, um auf dieser Grundlage eine Rule zu entwerfen. Bei einer Statusveränderung des Items soll ein Skript ausgeführt werden. Dieses Skript nimmt die JSON-Datei, welche in dem Item gespeichert ist und leitet diese an die URL des bereits in Autodesk Tandem eingerichteten Stream weiter. Die Datenübertragung geschieht durch ein HTTP Post Request (vgl. Abb. 5.13).

```
configuration: {}
triggers:
  - id: "3"
    configuration:
      itemName: room_3165_json
    type: core.ItemStateUpdateTrigger
conditions: []
actions:
  - inputs: {}
    id: "2"
    configuration:
      type: application/vnd.openhab.dsl.rule
      script: >-
        val data = room_3165_json.state.toString

        val url = "https://:BwNGdtjER9SXUHjnNDpcTQ@tandem.autodesk.c

        val contentType = "application/json"

        sendHttpPostRequest(url, contentType, data)
    type: script.ScriptAction
```

Abbildung 5.13: Rule zur Weiterleitung der Sensordaten in OpenHAB

Danach müssen dann nur noch die einzelnen Werte den passenden Parametern in Tandem zugewiesen werden (vgl.4.2.4).

5.4 Evaluation

Die vorgestellte prototypische Generierung und das Versenden der Sensordaten konnte im Falle einer einfachen Netzwerkarchitektur einwandfrei umgesetzt werden. Allerdings kam es bei der Umsetzung in einer komplexeren Netzwerkarchitektur mit höheren Sicherheitsstandards, wie es in einer Universität vorliegt, zu Komplikationen. Vor allem die

Einrichtung einer Internetverbindung für eine Sensor Node, welche über WLAN verbunden ist, stellte sich als problematisch heraus.

Kapitel 6

Zusammenfassung der Ergebnisse

Die zentrale Zielsetzung der prototypischen Umsetzung eines Digitalen Zwillings von Räumen des Lehrstuhls für Computergestützte Modellierung und Simulation konnte erfolgreich durchgeführt werden.

Die allgemeinen Anforderungen an eine DT-Plattform wurden herausgearbeitet und auf ausgewählte Plattformen angewandt. Autodesk Tandem konnte dabei von den 14 Anforderungen sieben vollständig, vier nur teilweise und drei nicht erfüllen. Die für den Gebäudebetrieb spezifischen Anforderungen wurden herausgearbeitet und auf eine DT-Plattform (Autodesk Tandem) angewandt. Der FM-Handover, sowie die BIM-CAFM-Integration können mithilfe von Autodesk Tandem, bei dem es sich um ein Integriertes Asset-Informations-System handelt, einwandfrei durchgeführt werden. Beim FM-Handover gehen dabei kaum Informationen verloren. Auch die BIM-CAFM-Integration zum Aufbau einer CDE für die Betriebsphase eines Gebäudes wird durch die Verwendung von Tandem unterstützt. Ein zentraler Aspekt von Autodesk Tandem ist die einheitliche, strukturierte Datenhaltung, welche durch ein Klassifikaionssystem unterstützt wird. Es deckt viele technische Komponenten einer CDE ab, die in 2.1.3 aufgeführt wurden.

Die notwendigen Schritte zur Integration von sowohl sich zeitlich verändernden Daten, als auch Sensordaten konnten erfolgreich aufgezeigt werden. Die Integration von sich zeitlich verändernden Daten konnte anhand von zwei Beispielen erfolgreich durchgeführt werden. Innerhalb der Tandem-Datenbank müssen dazu der gewünschten Klasse Parameter hinzugefügt werden. Diese Parameter ermöglichen die Zuweisung von Werten, Text oder auch Links und Dokumenten. Auch die Integration von Sensordaten konnte erfolgreich durchgeführt werden. Dazu ist in Tandem zunächst die Erstellung eines Streams als Datenpipeline notwendig. Diese ist über eine URL erreichbar und kann dadurch aktualisiert werden. Bei dem Stream handelt es sich in Autodesk Tandem um ein Objekt. Dieses Objekt enthält Daten im JSON-Format. Durch die Verknüpfung von Parametern mit einzelnen Werten aus dieser JSON-Datei können die Sensordaten einem Element zugewiesen werden. In dem behandelten Beispiel wurde dazu eine neue Klasse Environment Sensor mit den Parametern Luftfeuchtigkeit, Lufttemperatur, etc. erstellt. Durch Zuweisung der Entität des konkreten Environment Sensor zu einem Raum, können die Daten innerhalb des Gebäudemodells verortet werden.

Das Vorgehen für die Generierung und Verteilung von Sensordaten innerhalb eines Gebäudes wurden erläutert. Das Messen von aktuellen Zustandsdaten innerhalb eines Raums und dessen vollständiges Versenden an eine DT-Plattform wurden erfolgreich umgesetzt. Die Daten werden auf einer Sensor Node generiert und über openHAB an die DT-Plattform weitergeleitet. In diesem Fall wurde mithilfe eines ESP32-Microcontrollers Lufttemperatur,

Luftfeuchtigkeit und Gas Widerstand der Luft gemessen, sowie die Werte zweier verschiedener Sensoren zur Präsenz von Personen abgefragt. Die Messwerte werden von den Sensoren über einen Zeitraum von 60 Sekunden mehrmals abgefragt und die daraus entstehenden Mittelwerte versendet. Bei den Präsenzsensoren wird im Fall, dass auch nur einmal innerhalb der letzten 60 Sekunden eine Person erkannt wurde, ein positiver Wert übergeben. Bei dem Sensorknotenpunkt wurde sowohl die Entwicklung der Hardware, als auch der Software beschrieben. Es könnten auch weitere Sensoren für andere Messwerte mit dem Microcontroller verbunden werden. Es wurden hiermit Grundlagen für weitere Experimente mit Sensorik innerhalb von Gebäuden im Hochschulkontext vermittelt. Durch die Wahl von Micropython und dessen einfache Programmierung könnten auch Personen mit niedrigen Vorkenntnissen diese Experimente durchführen. Allerdings funktionieren die Plug-ins für Micropython in IDEs wie PyCharm oder Visual Studio nur mangelhaft. Daher muss momentan leider bei der Programmierung mit Micropython eine einfachere IDE wie Thonny verwendet werden. Wie bereits in 5.4 erwähnt, stellt die Netzwerkarchitektur einer Universität und deren Sicherheitsstandards eine Hürde zur Implementierung von drahtlosen Sensornetzwerken dar.

Kapitel 7

Fazit und Ausblick

Laut der Einstufung von DENG et al., 2021 entspricht der derzeitige Entwicklungsstand von Autdesk Tandem der dritten Stufe auf der Evolutionsleiter. Dennoch ist die Plattform insgesamt gut für die Umsetzung eins DT für den Betrieb eines Gebäudes geeignet. Der FM-Handover lässt sich mithilfe dieser Plattform gut umsetzen. Auch die BIM-CAFM-Integration zum Aufbau einer CDE für die Betriebsphase wird durch diese Plattform maßgebend unterstützt. Die Schnittstellen zu anderen CAFM-Systemen könnten in diesem Kontext noch zu Problemen führen und müssen noch vollständig implementiert werden.

Der Hauptnutzen von Autodesk Tandem ist das Fungieren als zentrale Datenbank für den Betrieb. Die DT-Plattform enthält alle Informationen, die zum Betrieb eines Gebäudes benötigt werden. Allerdings deckt Autodesk Tandem nicht alle Funktionen innerhalb des Spektrums des CAFM ab. Daher sind die Schnittstellen bzw. der Datenaustausch mit anderen CAFM-Systemen ein kritischer und sehr wichtiger Punkt. Nur wenn im Sinne einer CDE Daten ohne Verluste innerhalb des gesamten Netzes an CAFM-Sytemen ausgetauscht werden können, kann ein wirklich reibungsloser und effizienter Betrieb gewährleistet werden. Derzeit befinden sich die Schnittstellen von Autodesk Tandem noch in der Entwicklungsphase. Sie sind somit auch Änderungen innerhalb ihrer Funktionsweise unterworfen. Dies birgt für andere Softwarehersteller Unsicherheiten, da sie bei der Entwicklung eigener Softwarelösungen bzw. der Verbindung ihrer Lösungen mit der Tandem-Datenbank bei einer Änderung der Funktionsweise der Schnittstelle mit Einschränkungen und Fehlern zu rechnen haben. Daher sollten Regelungen zu Dateiaustauschformaten und Schnittstellen klar definiert werden, um einen nahtlosen Betrieb sicherzustellen. Generell könnte eine Normung für DT im Bau- und Gebäudesektor erarbeitet werden. Dabei könnte man sich an der bereits für den Produktionssektor bestehenden Norm (»ISO 23247«, 2021) orientieren.

Laut der Einstufung von DENG et al., 2021 befindet sich insgesamt die, innerhalb dieser Arbeit dargestellten, Implementierung eines DT nur auf der Stufe drei von fünf. Denn bislang werden nur Sensordaten mit einem BIM verknüpft und somit Real-Time-Visualization und Real-Time-Monitoring betrieben. Die nächsten Schritte im Zuge einer Erstellung eines Digital Twins des TUM Innenstadt Campus könnten folgende sein:

1. Erweiterung des BIM
2. Installation von weiteren Sensoren
3. Generierung einer ausreichenden Datengrundlage
4. Implementierung von Analysetools

5. Automatische Reaktionen auf Gegebenheiten in den Gebäuden

Die Generierung, Weiterleitung und Integration von Sensordaten in eine DT-Plattform konnten erfolgreich umgesetzt werden. Dabei wurden die Daten auf einer selbstentwickelten Sensor Node generiert und über einen Gateway zu der Plattform weitergeleitet. Durch die Verwendung von openHAB als Gateway können sowohl selbstentwickelte Sensor Nodes, wie in diesem Fall, als auch auf dem Markt erhältliche Produkte einfach integriert werden. Lediglich die Netzwerkarchitektur in der Universität stellt eine Hürde dar. Als vielversprechende Alternative könnte eine Datenversendung per LoRaWAN untersucht werden. Dies wurde beispielsweise bereits von OPOKU et al., 2023 in einer Universität in Australien erfolgreich umgesetzt.

Das Programm der Sensor Node hatte in den bisherigen Tests von einigen Tagen ohne Probleme funktioniert. Dies müsste allerdings über einen längeren Zeitraum nochmals genauer beobachtet werden. Falls es zu Fehlern kommen sollte, müsste der Code gegebenenfalls nochmals optimiert werden. Die generierten Raumklimadaten müssten nochmals auf Korrektheit überprüft werden. Es gäbe hierbei die Möglichkeit den Sensor zur Messung dieser Daten zu kalibrieren. Leider konnte im Rahmen dieser Arbeit nicht die Kalkulation der Luftqualität aus den Messwerten integriert werden. Um genauere Messwerte zu erhalten gibt es die closed-source library Bosch Sensortec Environmental Cluster (BSEC). Diese ermöglicht eine Kalibrierung der Messwerte sowie die Berechnung des Indoor air quality(iaq)-Index. Der Index könnte zum einen auf dem Display ausgegeben werden, um den aktuellen Luftzustand den Personen im Raum zu vermitteln. Zum anderen kann der Index natürlich auch in den DT übertragen und gespeichert werden. Es wäre technisch machbar die Berechnung direkt auf den Sensor Nodes durchzuzführen. Die Library von Bosch war allerdings zum derzeitigen Stand nicht in Python verfügbar. Es gäbe allerdings die Möglichkeit mittels Technologien wie beispielsweise Docker die BSEC library innerhalb eines Containers zu platzieren und so in Python zu integrieren (RICCARDI, 2021; PI3G, 2022). Eine Lösung zur Integration der BSEC library auf der Sensor Node könnte Gegenstand weiterer Forschung sein.

Zusätzlich gibt es die Möglichkeit den Stromverbrauch des ESP32 zu optimieren. Dieser besitzt die Fähigkeit sich in einen Tiefschlaf-Modus zu versetzen und so den Stromverbrauch zu senken. Somit könnte die Abfrage der Sensordaten beispielsweise alle 5 Minuten geschehen und in der Zwischenzeit befindet sich der ESP im Tiefschlaf. Vermutlich müsste dann allerdings auf ein Display zum Anzeigen der Messwerte verzichtet werden. Für Sensor Nodes, die nicht im Sichtfeld von Personen installiert werden, wäre dies sowieso nicht relevant. Nach der Optimierung des Stromverbrauchs wäre auch eine autarke Lösung denkbar, bei der die Sensor Nodes von einer Batterie mit Strom versorgt werden.

Außerdem müsste auch die Zuverlässigkeit des Bewegungssensors und des Präsenzsensors nochmals genauer untersucht werden. Bei den ersten durchgeführten Tests war zu sehen, dass ab ca. fünf Metern Entfernung die Erkennung von Personen durch den Radar-Sensor stark abnimmt. Es gäbe auch hier die Möglichkeit diesen Sensor zu kalibrieren

und damit genauere Werte zu generieren. Wie in 2.3.1 bereits vorgestellt wurde, kann man durch die Überlagerung der verschiedenen Messwerte eine recht hohe Genauigkeit beim Erkennen von Personen erreichen.

Anhang A

Verwendete Python-Skripte

Im folgenden finden Sie die verwendeten Python-Skripte:

Algorithmus A.1: boot.py

```
from machine import Pin, I2C
from sh1106 import *
import time
from bme680 import *
import network
import json
import machine
from umqtt.simple import MQTTClient
from machine import Timer
import math

# Festelegen der Pins für die Sensoren und das Display

bme_i2c = I2C(1, scl = Pin(25), sda = Pin(26))
bme = BME680_I2C(bme_i2c)

dis_i2c = I2C(scl=Pin(22), sda=Pin(19), freq=400000)
display = SH1106_I2C(128, 64, dis_i2c, Pin(16), 0x3c)
display.sleep(False)
display.flip()

pir_sensor = Pin(15, Pin.IN)
ld2410 = Pin(16, Pin.IN)

# Erstellen der Variablen für die Messwerte
temp_list = []
hum_list = []
pres_list = []
gas_list = []
pir = 0
radar = 0

# Startbildschirm
display.fill(0)
display.text('Starting...', 0, 10, 1)
display.text('Sensor Node', 0, 30, 1)
display.text('Raum 3165', 0, 50, 1)
display.show()
```

```
time.sleep(4)

# Netzwerkverbindung
ssid = 'ssid'
key = 'key'
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
if not wlan.isconnected():
    display.text('connecting␣to␣network...', 0, 0, 1)
    print('connecting␣to␣network...')
    wlan.connect(ssid, key)
    while not wlan.isconnected():
        pass
print('network␣config:', wlan.ifconfig())

# Funktion, um per MQTT auf einem bestimmten topic eine Nachricht zu
    publishen
def publish_msg(topic, msg):
    mqtt_server = '192.168.3.253'
    user = 'openhabian'
    password = 'openhabian'
    c = MQTTClient("raum_3165", server=mqtt_server, user=user, password
        =password)
    c.connect()
    c.publish(topic, msg)
    c.disconnect()
    print("Mqtt_Client␣is␣publishing␣message...")
    print(topic)
    print(msg)

def send_data():
    # Mittelwert der letzten 60 Sekunden bilden
    temp_m = round(sum(temp_list) / len(temp_list), 1)
    hum_m = round(sum(hum_list) / len(hum_list), 1)
    pres_m = round(sum(pres_list) / len(pres_list), 1)
    gas_m = round(sum(gas_list) / len(gas_list))
    global radar
    global pir

    # Aus den Daten wird ein JSON erstellt
    data = {"status": "ON", "temperature": temp_m, "humidity": hum_m, "
        pressure": pres_m,
            "air_quality": gas_m,"radar": radar, "pir": pir}
    data = json.dumps(data)

    # Senden der Daten
    try:
        publish_msg('raum_3165', data)
    except Exception as e:
```

```
            display.text('MQTT failed', 0, 60, 1)
            print(e)

    # Zurücksetzen der Variablen für die nächste Zeitspanne
    temp_list.clear()
    hum_list.clear()
    pres_list.clear()
    gas_list.clear()
    radar = 0
    pir = 0


start = time.ticks_ms() # Timer starten


while True:
    # Messwerte von Sensoren ablesen
    temp = round(bme.temperature, 1)
    hum = round(bme.humidity, 1)
    pres = round(bme.pressure, 1)
    gas = bme.gas
    if ld2410.value() == 1:
        radar = 1
    if pir_sensor.value() == 1:
        pir = 1

    # Messwerte in Listen abspeichern
    temp_list.append(temp)
    hum_list.append(hum)
    pres_list.append(pres)
    gas_list.append(gas)

    # Messwerte auf dem Diplay ausgeben
    display.fill(0)
    display.text('Temperature', 0, 5, 1)
    display.text(str(temp)+ ' C', 5, 20, 1)
    display.text('Humidity', 0, 35, 1)
    display.text(str(hum)+' %', 5, 50, 1)
    display.show()
    time.sleep(4)

    display.fill(0)
    display.text('Pressure', 0, 5, 1)
    display.text(str(pres) + ' kPa', 5, 20, 1)
    display.text('Gas resistance', 0, 35, 1)
    display.text(str(gas), 5, 50, 1)
    display.show()
    time.sleep(4)
```

```
    # Bei überschreiten des Timers von 60 Sekunden: Daten senden
    delta = time.ticks_diff(time.ticks_ms(), start)
    if delta > 60000:
        send_data()
        start = time.ticks_ms()
```

Literatur

ACCA SOFTWARE S.P.A. (2023). Verfügbar 25. Oktober 2023 unter https://www.accasoftware.com/de/uber-uns

AENGENVOORT, K., & KRÄMER, M. (2021). BIM im Betrieb von Bauwerken. In *Building Information Modeling - Technologische Grundlagen und industrielle Praxis, 2.Auflage* (S. 611–644). Springer Fachmedien Wiesbaden GmbH. http://www.springer.com/series/3482

AHLBERG, F., GRATTON, A., et al. (n. d.). Verfügbar 14. September 2023 unter https://github.com/espressif/esptool

AUTODESK, I. (2023). Verfügbar 25. Oktober 2023 unter https://autodesk-tandem.github.io

AUTODESK, INC. (2023). Verfügbar 25. Oktober 2023 unter https://www.autodesk.com/company

BABEL, W. (2023). *Internet of Things und Industrie 4.0*. Springer Fachmedien Wiesbaden GmbH. https://doi.org/https://doi.org/10.1007/978-3-658-39901-6

BÄCKLUND, K., MOLINARI, M., LUNDQVIST, P., & KARLSSON, P. (2022). Showcasing the First Steps Towards a Digital Twin for Campus Environments. *E3S Web of Conferences 362, 10003*.

BENTLEY SYSTEMS INC. (2023). Verfügbar 28. September 2023 unter https://de.bentley.com/software/itwin-platform/

BERNERS-LEE, T., HENDLER, J., & LASSILA, O. (2001). The semantic web. *Scientific American*, *284(5)*, 35–43.

BORRMANN, A., KÖNIG, M., KOCH, C., & BEETZ, J. (2021). *Building Information Modeling - Technologische Grundlagen und industrielle Praxis, 2.Auflage*. Springer Fachmedien Wiesbaden GmbH. http://www.springer.com/series/3482

BORRMANN, A., SCHLENGER, J., BUS, N., & SACKS, R. (2022). AEC Digital Twin Data - Why structure matters.

BOSCH, M., & DECKERT, R. (2023). *Digitalisierung und Smart Building - Ein kritischer Erfolgsfaktor für nachhaltige Entwicklung*. Springer Fachmedien Wiesbaden GmbH. https://doi.org/https://doi.org/10.1007/978-3-658-41629-4

BOSCH SENSORTEC GMBH. (2023). Verfügbar 18. Oktober 2023 unter https://www.bosch-sensortec.com/software-tools/software/bme680-software-bsec/

BRILAKIS, I., PAN, Y., BORRMANN, A., MAYER, H.-G., RHEIN, F., VOS, C., PETTINATO, E., & WAGNER, S. (2020). *Built Environment Digital Twinning - Report of the International Workshop on Built Environment Digital Twinning presented by TUM Institute for Advanced Study and Siemens AG*. https://publications.cms.bgu.tum.de/reports/2020_Brilakis_BuiltEnvDT.pdf

BUILDINGSMART. (2022). Verfügbar 25. Oktober 2023 unter https://www.buildingsmart.org/from-standard-to-multinational-membership-for-acca-software/

BUNDESMINISTERIUM FÜR WIRTSCHAFT UND KLIMASCHUTZ. (2023). Verfügbar 15. November 2023 unter https://www.bmwk-energiewende.de/EWD/Redaktion/Newsletter/2014/22/Meldung/hoher-energieverbrauch-des-gebaeudesektor.html

CHAMARI, L., PETROVA, E., & PAUWELS, P. (2023). Extensible Real-Time Data Aquisition and Management for IoT Enabled Smart Buildings. *2023 European Conference on Computing in Construction, 40th International CIB W78 Conference*.

DENG, M., MENASSA, C., & KAMAT, V. (2021). From BIM to digital twins: a systematic review of the evolution of intelligent building representations in the AEC-FM industry. *Journal of Information Technology in Construction (ITcon), Special issue: 'Next Generation ICT - How distant is ubiquitous computing?'*, *26*, 58–83. https://doi.org/10.36680/j.itcon.2021.005

DIN EN ISO 12006-2: Hochbau – Organisation des Austausches von Informationen über die Durchführung von Hoch- und Tiefbauten – Teil 2: Struktur für die Klassifizierung (ISO 12006-2:2015);Deutsche Fassung EN ISO 12006-2:2020. (2020).

DONKERS, A., YANG, D., & de VRIES, B. (2023). Semantic Web-Enabled Outlier and missing value Detection and Replacement in Smart Buildings. *2023 European Conference on Computing in Construction, 40th International CIB W78 Conference*.

DOPIERALSKI, R., HAMMELRATH, R., & WEBER, T. (2021). Verfügbar 23. Oktober 2023 unter https://github.com/robert-hh/SH1106

DTWIN. (2023). Verfügbar 25. Oktober 2023 unter https://www.nemetschek-dtwin.com

GEFMA 400: Computer Aided Facility Management CAFM – Begriffsbestimmungen, Leistungsmerkmale. (2021).

GLAESSGEN, E. H., & STARGEL, D. (2012). The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles. https://ntrs.nasa.gov/citations/20120008178

GRIEVES, M. (2015). Digital Twin: Manufacturing Excellence through Virtual Factory Replication.

ISO 23247: Automation systems and integration - Digital twin framework for manufacturing. (2021).

ISO 41011: Facility Management - Vocabulary. (2017).

ISO DIN 19650-1: Organisation und Digitalisierung von Informationen zu Bauwerken und Ingenieurleistungen, einschließlich Bauwerksinformationsmodellierung (BIM) - Informationsmanagement mit BIM - Teil 1: Begriffe und Grundsätze. (2019).

KLEPEIS, N., NELSON, W., OTT, W., et al. (2001). The National Human Activity Pattern Survey (NHAPS): a resource for assessing exposure to environmental pollutants. *J Expo Sci Environ Epidemiol*, *11*, 231–252.

KRÄMER, M., BESENYÖI, Z., SAUER, P., & HERRMANN, F. (2018). Common Data Environment für BIM in der Betriebsphase – Ansatzpunkte zur Nutzung virtuell verteilter Datenhaltung. In *Handbuch Facility Management* (S. 1–32). ecomed SICHERHEIT Verlag, Heidelberg, München, Landsberg, Frechen, Hamburg.

KRÄMER, M., BENDER, T., BOCK, N., HÄRTIG, M., JASPERS, E., KOCH, S., OPIĆ, M., & SCHLUNDT, M. (2022). IT-Umgebungen für BIM im FM. In *BIM im Immobilienbetrieb - Anwendung, Implementierung, Digitalisierungstrends und Fallstudien* (S. 103–132). Springer Fachmedien Wiesbaden GmbH.

KROKER, M. (2018). *Die Geschichte des Internet of Things vom Programmable Logic Controller 1968 bis heute*. Verfügbar 7. August 2023 unter https://blog.wiwo.de/look-at-it/2018/02/16/die-geschichte-des-internet-of-things-vom-programmable-logic-controller-1968-bis-heute/

LEHNER, D., PFEIFFER, J., TINSEL, E., STRLJIC, M., SINT, S., VIERHAUSER, M., WORTMANN, A., & WIMMER, M. (2022). Digital Twin Platforms: Requirements, Capabilities, and Future Prospects. *IEEE Software*, *39*(02), 53–61. https://doi.org/10.1109/MS.2021.3133795

LIMOR 'LADYADA' FRIED OF ADAFRUIT AND JEFF RABER. (2017). Verfügbar 23. Oktober 2023 unter https://github.com/robert-hh/BME680-Micropython/blob/master/bme680.py

LU, V. Q., PARLIKAD, A. K., WOODALL, P., RANASINGHE, G. D., & HEATON, J. (2019). Developing a Dynamic Digital Twin at a Building Level: Using Cambrige Campus as a Case Study. *International Conference on Smart Infrastructure and Construction 2019 (ICSIC): Driving data-informed decision-making*.

MICROSOFT, INC. (2023a). Verfügbar 25. November 2023 unter https://learn.microsoft.com/de-DE/azure/digital-twins/overview

MICROSOFT, INC. (2023b). Verfügbar 25. Oktober 2023 unter https://learn.microsoft.com/de-de/azure/digital-twins/concepts-ontologies-convert

OPOKU, D.-G. J., PERERA, S., OSEI-KYEI, R., RASHIDI, M., BAMDAD, K., & FAMAKINWA, T. (2023). Digital twin for indoor condition monitoring in living labs: University library case study. *Automation in Construction*, *157*.

PAUWELS, P., COSTIN, A., & RASMUSSEN, M. H. (2022). Knowledge Graphs and Linked Data for the Built Environment. In *Industry 4.0 for the Built Environment - Methodologies, Technologies and Skills* (S. 157–184). Springer Nature Switzerland AG 2022.

PEDERSEN, T. H., NIELSEN, K. U., & PETERSEN, S. (2017). Method for room occupancy detection based on trajectory of indoor climate sensor data. *Building and Environment*, *115*, 147–156.

PI3G. (2022). Verfügbar 18. Oktober 2023 unter https://github.com/pi3g/bme68x-python-library

PREIDEL, C., BORRMANN, A., EXNER, H., & KÖNIG, M. (2021). Common Data Environment. In *Building Information Modeling - Technologische Grundlagen und industrielle Praxis, 2.Auflage* (S. 335–351). Springer Fachmedien Wiesbaden GmbH. http://www.springer.com/series/3482

REJA, V. K., & VARGHESE, K. (2022). Digital Twin Applications for Construction Project Management.

RICCARDI, T. (2021). Verfügbar 18. Oktober 2023 unter https://github.com/thomas-riccardi/micropython-esp8266-bsec-bme680

SHILPA, V., VIDYA, A., & PATTAR, S. (2022). MQTT based Secure Transport Layer Communication for Mutual Authentication in IoT Network. *Global Transitions Proceedings*, *3*, 60–66.

THULLIER, F., BEAULIEU, A., MAITRE, J., GABOURY, S., & BOUCHARD, K. (2022). A Systematic Evaluation of the XeThru X4 Ultra-Wideband Radar Behavior. *Procedia Computer Science*, *198*, 148–155.

U.S. ENVIRONMENTAL PROTECTION AGENCY. (2023). Verfügbar 15. November 2023 unter https://www.epa.gov/indoor-air-quality-iaq/introduction-indoor-air-quality