Technische Universität München
Ingenieurfakultät Bau Geo Umwelt
Lehrstuhl für Bauphysik

Entwicklung eines integralen Ansatzes zur Implementierung von bauphysikalischen Funktionalitäten in der Automatisierung von Wohngebäuden

Alexander Peikos, M.Sc.

Vollständiger Abdruck der von der Ingenieurfakultät Bau Geo Umwelt der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender: Prof. Dr. Christian Große

Prüfende der Dissertation:

1. Prof. Dr.-Ing. Klaus Peter Sedlbauer
2. Prof. Dipl.-Ing. Thomas Auer

Die Dissertation wurde am 04.06.2019 bei der Technischen Universität München eingereicht und durch die Ingenieurfakultät Bau Geo Umwelt am 21.10.2019 angenommen.

Danksagung

„Die Zeit der Promotion hat mich den Unterschied zwischen allein und eigenständig gelehrt."

An erster Stelle gilt mein Dank meinem Doktorvater Herr Prof. Dr. Klaus Peter Sedlbauer für seine wissenschaftliche und methodische Unterstützung. Danke für den kleinen Schubs am Anfang, um sich auf das Abenteuer Dissertation überhaupt einzulassen und dafür, dass er in allen schwierigen Phasen ein offenes Ohr für mich hatte.

Ein besonderer Dank gilt Dr. Simon Schmidt, der mir in Sachen Programmieren den ersten Schubs gegeben hat und mir immer als Vorbild gedient hat. Ebenso möchte ich mich bei Dr. Manuel Lindauer bedanken, der seine Wochenenden mit mir verbracht hat, um MATLAB-Code zu debuggen und mir auch sonst immer mit Rat und Tat zur Seite stand. Außerdem gilt mein Dank Dr. Roland Göttig, der unzählige seiner Mittagspausen mit mir verbracht hat, um über die Aspekte meiner Arbeit zu diskutieren und mir damit eine große Hilfestellung war.

Danke auch an das Team vom Lehrstuhl, das mich in der anstrengenden Schlussphase gut entlastet hat und es mir so ermöglichte, mich voll auf diese Arbeit zu konzentrieren. Genauso wie die Mitarbeiter vom Ingenieurbüro-Peikos, die mir mit Formulierungstipps und ihrer fröhlichen Art die Schreibart erleichtert haben.

Besonders möchte ich an dieser Stelle auch bei meiner Familie Thanos, Ellen und Philipp Peikos bedanken, die mich unermüdlich immer wieder motiviert haben und mir zur Seite standen. Sei es durch Korrekturlesen, motivierende Worte oder dem spontanen Installieren einer Server-Anlage. Ebenso möchte ich mich bei Michael Rauscher bedanken, der immer, wenn's mal brenzlig wurde, für einen guten Rat zu haben war. Außerordentlicher Dank gilt auch meiner Großmutter Eleni Peikou in Griechenland, bei der ich eine Woche für intensive Schreibarbeit untertauchen konnte und gut umsorgt wurde.

Ein Dank gilt auch Kristina Keil, die mich einen Großteil der Zeit begleitet hat und alle Fehlsteuerungen meiner Wohnung immer mit einem Lächeln hingenommen hat. Außerdem für ihre tatkräftige Unterstützung beim Umbau und der Planung meiner Wohnung, die letztendlich das Grundgerüst für diese Arbeit bereitstellte.

Vielen Dank an meine Tanzpartnerin Sarah Jean Reiner ohne die eine erfolgreiche Turniersaison auch während der Promotion nicht möglich gewesen wäre. Danke, dass du mich auch in der Schlussphase trotz dutzender Trainings, die ausfallen mussten, immer unterstützt hast und mich immer wieder ermutigt hast nicht aufzugeben.

Danke an all meine Freunde, die ich hier namentlich nicht alle aufführen kann, die mir immer zur Seite standen und ihre Hilfe angeboten haben.

Ohne all diese Menschen hätte ich diese Arbeit alleine schreiben müssen, so hatte ich die Möglichkeit sie eigenständig zu verfassen.

Zusammenfassung

Gebäude erfüllen die unterschiedlichsten Funktionen, wie den Schutz vor Wetter und das Sicherstellen von Privatsphäre. Hauptsächlich sind sie jedoch der Mittelpunkt unseres Lebens, vor allem wenn man den Wohnraum betrachtet. Dabei haben sich die Funktionen, die ein Wohnraum erfüllen muss, im Laufe der Zeit mit der Gesellschaft verändert. Während früher Schutzbedürfnisse und die Vereinbarkeit von Handwerk und Privatleben Hauptfunktionen von Wohngebäuden waren, dienen diese nun eher den Aspekten der Selbstverwirklichung und bieten Platz für viele private Interessen der Bewohner. Der Wandel unserer Gesellschaft, welcher auch stark von der voranschreitenden Digitalisierung beeinflusst wird, hinterlässt auch hier seine Spuren. Doch wie lässt sich abschätzen in welche Richtung sich der Wohnraum durch die Digitalisierung entwickelt? Regelmäßig wurden aus den in Science-Fiction-Filmen dargestellten Konstrukten reale Gegenstände oder Techniken entwickelt. Während Bildtelefonie noch vor 47 Jahren als unmögliche Zukunftsmusik galt, ist sie nun zu einem normalen Bestandteil unseres alltäglichen Lebens geworden.

Weitere Beispiele für solche Entwicklungen lassen sich in hoher Zahl anführen. Dies lässt den Schluss zu, dass auch mögliche zukünftige Entwicklungen im Wohnraum von diesem Zukunftsdenken beeinflusst werden. Mit der Digitalisierung unserer Gesellschaft wird so auch eine Digitalisierung des Wohnraums einhergehen. Ein prominentes Beispiel hierfür ist die Entwicklung sogenannter Smarthomes. Dargestellt werden diese oft als sehr interaktive und intelligente Systeme. Sie nehmen häufig die Position eines virtuellen Assistenten ein und gestalten so einen stark individualisierten Wohnraum.

Doch auch wenn Smarthomes bereits in unserer Gesellschaft kommerziell erwerbbar sind, ist deren Grad an Intelligenz und Individualisierbarkeit noch weit von den Zukunftsvisionen aus Science-Fiction-Filmen entfernt. Einfache Regelungen auf Basis von sogenannten „Wenn-Dann“-Regeln bestimmen die Mehrheit der erfolgten Steueraktionen. Tatsächliche Individualisierungen sind bei weitem nicht der Standard. Die Rolle eines wirklichen Assistenten spielen diese Wohnräume auch nicht, eher die Rolle einer digitalen Unterstützung, um die Technik eines Gebäudes leichter steuern zu können. Viele Bereiche des alltäglichen Lebens sind jedoch schon genauer und individueller beschreibbar, als es mit einfachen „Wenn-Dann“-Regeln möglich ist. So lässt sich beispielsweise die thermische Behaglichkeit auf Basis unzähliger individueller Parameter beschreiben und darauf basierend die Raumtemperatur regulieren. Die Integration dieses Wissens in das Smarthome ist jedoch bis jetzt nicht möglich. Auch die Abbildung der hohen Dynamik, der ein Wohnumfeld beispielweise durch die sich verändernde Nutzung einzelner Räume unterliegt, lässt sich mit den derzeit aktuellen Systemen nicht abbilden. Ebenso mangelt es an Möglichkeiten stark kontextbezogene Daten in die Steuerung eines Smarthomes einfließen zu lassen. Die Zahl von notwendigen „Wenn-Dann“-Formulierungen würde allein bei der Berücksichtigung des Wohnkontextes für eine biologisch wirksame Beleuchtungssteuerung Ausmaße annehmen, die einer vollständigen Neu-Entwicklung eines Algorithmus für diesen Fall gleichkommt. Auch Funktionen, die die Bauphysik zur Verfügung stellen kann, lassen sich auf Grund ihres hohen Potentials an gegenseitiger Beeinflussung und Interaktion nicht abbilden. Das Fehlen modularer Ansätze im Smarthome, wie es in Bereichen der Gebäudesimulation bereits Stand des Wissens ist, behindert diese Entwicklung ebenso wie die fehlende Interoperabilität im Smarthome-Sektor.

Daher wurde eine Methodik entwickelt, die eben diese Lücke schließt. Der vorgestellte Ansatz erhebt den Anspruch, den derzeit üblichen Automationsansatz, welcher durch einfache Abhängigkeiten zwischen Events und Schaltaktionen funktioniert, durch einen weit komplexeren und realitätsnäheren Ansatz zu ersetzen. Es wird die Möglichkeit eröffnet Simulationsmodelle, sowie andere Lösungsverfahren aus der Bauphysik zu integrieren.

Um sowohl diese Komplexität als auch die besonderen Zusammenhänge im häuslichen Umfeld zu berücksichtigen, wurde ein Ansatz gewählt, der durch eine realitätsnahe Abbildung der Zusammenhänge die Interaktion zwischen Nutzer und Gebäude ermöglicht.

Dies ermöglicht vor allem die Sicherstellung der folgenden für ein Smarthome als notwendig erachteten Faktoren:

- Nutzerindividualität
- Interaktion
- Dynamik
- Kontextsensitivität
- Modularer Aufbau

Um diese Faktoren sicherstellen zu können, wurden im Rahmen dieser Arbeit sowohl für das Gebäude als auch für den Nutzer, jeweils ein digitaler Zwilling definiert. Digitale Zwillinge repräsentieren deren reale Pendants in Form digitaler Konstrukte. Sie können durch einfache Datenbanken oder komplexe Simulationsmodelle erstellt werden. Durch die Generierung dieser Zwillinge ist es möglich, das Verhalten von Gebäude und Nutzer virtuell abzubilden. Für die Verarbeitung der daraus resultierenden Erkenntnisse ist ein Framework notwendig, das die Verbindung zwischen Gebäude und Nutzer herstellt und die klassischen Automationsaufgaben in dieses Konstrukt übersetzt.

Hierfür erfolgte weiterhin eine Teilung des in der Automation klassisch verankerten Funktionsansatzes in verschiedene Module. Es konnte eine klare Aufteilung mathematischer und physikalischer Sachverhalte auf die jeweiligen digitalen Zwillinge umgesetzt werden. Für diese Aufteilung wurde ein einfaches Prinzip, das auf der Struktur des menschlichen Bedürfnisbewusstseins aufbaut, entwickelt. In der folgenden Grafik wird die Aufteilung der digitalen Zwillinge des Gebäudes sowie des Nutzers in ihre thematischen Untergliederungen dargestellt. Die Entstehung von Nutzerbedürfnissen und deren Wunsch nach Befriedigung führt zu der abgebildeten Interaktionsstruktur. Diese kann als Grundkomponente bezeichnet werden, welche nahezu alle Steuerungen im häuslichen Umfeld bedingt. Diese Interaktionsstruktur basiert auf den bereits genannten Bedürfnissen des Nutzers, welche sich in Form von Anforderungen beschreiben lassen. Sie werden meist vom Nutzer gestellt und sollen Gebäude erfüllt werden.

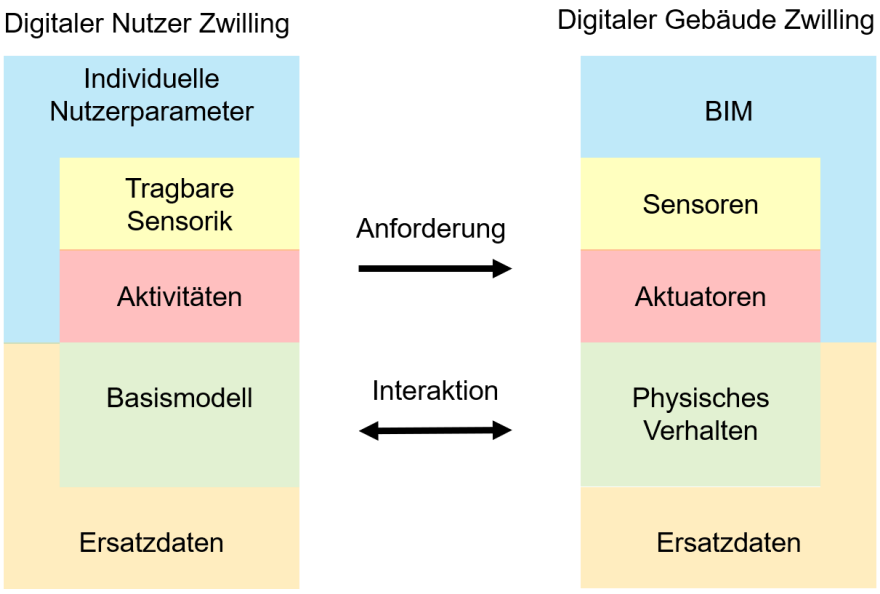


Abb. 1 Interaktionsmodell digitaler Gebäude- und Nutzerzwilling

Die Darstellung der einzelnen Aspekte des jeweiligen digitalen Zwillings erfolgt in Form von Modulen und trägt damit der für ein Smarthome notwendigen Modularisierung Rechnung.

Um die Diversität der verschiedenen Zusammenhänge, vor allem in Bezug auf die bauphysikalischen Aspekte der Automatisierung des Wohnraumes abbilden zu können, ist auf einer weiteren Skala eine Unterscheidung notwendig. Zu diesem Zweck wurde daher neben der Einteilung in Nutzer- und Gebäudemodule in vier weitere grundsätzliche Typen von Modulen unterschieden:

- Funktionsmodule
- Hilfsmodule
- Wirkmodule
- Module für maschinelles Lernen und Vorhersagen

Funktionsmodule beschreiben die grundsätzlichen physikalischen und mathematischen Funktionen des jeweiligen Zwillings und stellen den Kernalgorithmus der klassischen Funktionen im Smarthome dar.

Hilfsmodule hingegen bilden mathematische und physikalische Zusammenhänge ab, die keiner Funktionalität zugeordnet werden können, aber für Berechnungen wichtige Daten erzeugen. Durch dieses Konstrukt ist es möglich grundsätzliche physikalische Zusammenhänge abzubilden und deren systemübergreifende Verwendung zu ermöglichen.

Wirkmodule beschreiben, welche Aktionen notwendig sind, um gewünschte Wirkungen zu erzeugen oder berechnen die Auswirkungen von Steueraktionen auf das Umfeld. Sie stellen damit das digitale Bindeglied zu physischen Aktionen dar. Von ihnen werden auch die letztendlich notwendigen Steueraktionen für Geräte generiert.

Module für maschinelles Lernen und Vorhersagen erweitern das Framework um die Möglichkeit aus speziellen Algorithmen Wissen über den Nutzer und sein Umfeld zu generieren und dadurch stark kontextbezogene und individualisierte Steuerungen zu erzeugen.

Ein Modul hat im Sinne dieser Arbeit eine einfache Struktur, bestehend aus einer Importschicht, einem Funktionsblock und einer Exportschicht. Hierbei werden an der Importschicht die für eine Berechnung notwendigen Daten in das Modul eingegeben. Im Funktionsblock werden diese verarbeitet und an der Exportschicht werden die daraus resultierenden Daten wieder ausgegeben. Dabei erfolgt die Ein- und Ausgabe der Daten immer in einer standardisierten Art. Durch diese Standardisierung ist eine automatische Zuordnung von Eingangsgrößen möglich. Eingangsgrößen können im Sinne von Modulen verschiedene Ausprägungen annehmen. Mögliche Größen sind Sensorwerte, Parameter aus Datenbanken oder Ausgangsdaten aus anderen Modulen.

Um eine funktionsfähige Zuordnung der möglichen Eingangsdaten zu einem Modul zu gewährleisten, müssen mehrere Informationen über die vorhandenen Daten zur Verfügung stehen. Die Verarbeitung solcher wissensbasierten Zuordnungen wurde mit Hilfe eines ontologiebasierten Algorithmus realisiert.

Hierfür werden in Form einer Ontologie digitale Repräsentanten der jeweiligen Module, Sensoren und verschiedener Gebäudeteile erstellt. Das Wissen über die jeweiligen Repräsentanten wird in Form von Klassenzuordnung oder deren Zuweisung von Eigenschaften abgebildet. Die dafür notwendigen Daten können beispielsweise aus einem „Building Information Model“ bezogen werden. Besonders für die Bauphysik relevante Bereiche des Gebäudes können mit dieser Systematik abgebildet werden. Damit ist die Methodik in der Lage, aufgrund des gespeicherten Wissens über das Gebäude und den Nutzer selbstständig mittels Reasoning auf Basis generierter Regeln Verbindungen zwischen Modulen und Sensoren zu erzeugen. So besteht die Möglichkeit ein individuelles Berechnungssetup für das jeweilige Gebäude und den darin lebenden Nutzer zu schaffen.

Eine Berechnung kann damit ohne vorherige Konfiguration der Grundbedingungen durch den Nutzer oder einen Experten durchgeführt werden. Für die Berechnung können so standardisierte Module verwendet werden, welche den Stand des Wissens aus der jeweiligen Disziplin der Bauphysik abbilden. Durch die vom Modulentwickler mitzuliefernden Ersatzdaten kann sichergestellt werden, dass auch beim Nichtvorhandensein von verschiedenen Messwerten oder Parametern eine Berechnung weiterhin durchgeführt werden kann. Somit ist eine Speichermöglichkeit aller Datenarten, die in Form von beispielsweise Normen oder statistischer Auswertungen erfasst wurden, gegeben und eine umfassende Sammlung standardisierter gebäude- und nutzerrelevanter Daten möglich. Auch Module im Bereich der Ersatzdaten sind möglich, diese erstellen mit Hilfe von Berechnungsverfahren standardisierte Gebäudedaten.

Um diese nicht individuellen Daten im laufenden Betrieb durch individualisierte Daten ersetzen zu können, sind oft umfangreiche Analysen der erfassten Messdaten eines Gebäudes und der daraus ableitbaren Nutzerpräferenzen notwendig. Um diese Arbeiten leisten zu können wurden die bereits genannten „Module für maschinelles Lernen und Vorhersagen“ eingeführt. Ihr Zweck ist allein das Lernen verschiedener für die Individualisierung der Wohnraumautomation notwendigen Daten.

Nach der Ermittlung aller für das Gebäude relevanten Interaktionen erstellen die im Rechnungssetup erfassten Module ihre Berechnungen. Eine besondere Position nehmen dabei die Funktionsmodule ein, welche Ausgangsdaten auf Basis grundlegender Wirkgrößen generieren. Diese stellen Anforderungen in Form eines zeitbasierten Arrays mit Handlungsvorschlägen dar. Ein Modul ist dabei grundsätzlich in der Lage unterschiedliche Versionen an Handlungsvorschlägen auszugeben, welche zur Lösung des Problems beitragen.

Brauchbare Lösungsstrategien, die zum Beispiel von einem Modul zur Vermeidung von Schimmelpilzbildung ausgegeben werden können, wären die Erhöhung des Luftwechsels sowie das stärkere Beheizen des Raumes. Diese Strategien enthalten die vom Modul präferierten Änderungen an der Umgebung von jetzt bis in die Zukunft, sowie einen Zielwert, einer Priorität und einer möglichen Abweichung, bedingt durch Ungenauigkeiten in der Berechnung. Für die Ermittlung der Prioritäten wurde speziell hierfür ein vierstufiges Modell entwickelt, welches die unterschiedlichen Gewichtungen von verschiedenen Bedürfnissen berücksichtigt.

Hier werden Bedürfnisse in die Kategorien „Sicherheit“, „Gesundheit“, „Komfort“ und „Sonstiges“ eingeteilt. Die Priorität lautet dementsprechend „A“, „B“, „C“ oder „D“. Je nach Einstufung der im folgenden beschriebenen Wirkgrößen erfüllen die Handlungsanweisungen der Funktionsmodule dementsprechend Aspekte zur Gewährleistung von Sicherheit, Gesundheit, Komfort oder sonstigen Belangen des Nutzers. Die Priorität A ist dabei als essenzielle Priorität anzusehen, sie repräsentiert überlebenswichtige Grundaspekte im häuslichen Umfeld und soll sicherstellen, dass dem Nutzer kein Schaden entsteht. Die Priorität B ist von ähnlichem Stellenwert wie A ermöglicht jedoch eine feinere Abstufung, sodass vor allem in Entscheidungsfällen noch Lösungen gefunden werden können, die für den Nutzer zwar nicht Ideal aber dennoch ertragbar sind. Die Prioritäten C und D haben eher fakultativen Charakter und stellen daher keine schädigenden Sachverhalte dar, wenn sie nicht beachtet werden.

Dadurch ist es möglich, die Gesamtsumme der Handlungsvorschläge aller im System erfassten Funktionsmodule auf eine prioritätenbereinigte Form zu überführen. Die Ausgabe erfolgt als Prognose orientiertes zeitskaliertes Array. Somit ist es möglich, auch die Prioritäten, welche sich in der Zukunft ändern, zu berücksichtigen. Auch einer im zeitlichen Verlauf steigenden Relevanz von verschiedenen Aspekten wird hiermit Rechnung getragen.

Diese Prioritätenbereinigung erfolgt mittels eines eigens dafür entwickelten Systems, das durch konsolidieren der verschiedenen Anforderungsarrays die gemeinsamen Lösungsräume der Module sucht. Anschließend werden alle Einträge bereinigt, die in der Zukunft zu einem Dilemma führen würden. Die Überprüfung auf drohende Dilemmata erfolgt dabei auf Basis der von verschiedenen Modulen ausgegebenen Handlungsanforderungen.

Ergeben diese an einem Punkt in der Zukunft keine Möglichkeit einer gemeinsamen Lösung, so tritt durch die Interaktion dieser Module ein Dilemma auf. Dies kann zu zwei möglichen Ergebnissen führen:

1.) Der Zustand wird ignoriert, weil die in Konflikt stehenden Bedürfnisse Komfortbedürfnisse mit einer Priorität von C oder D sind. In diesem Falle entsteht kein echtes Dilemma im Sinne dieser Methode und die Handlungsanweisung mit der höheren Priorität überwiegt.
2.) Der zweite mögliche Zustand ist, dass sich zwei Handlungsanforderungen mit höherwertiger Priorität widersprechen. In diesem Fall muss von einer Steuerung dieser Handlungsanforderung abgesehen werden. Dies erfolgt durch das vollständige Streichen der Lösungsansätze aus der Gesamtbetrachtung und ist in folgender Grafik abgebildet.

	Output 1.1	Output 1.2	Output 3	Output 4.1	Output 4.2
Z	X	[…]	[…]	[…]	[…]
E	[…]	[…]	[…]	[…]	[…]
I	[…]	[…]	[…]	[…]	[…]
T	[…]	[…]	X	[…]	[…]

Abb. 2 Zu verwerfende Lösungsansätze wegen Gegenläufigkeit

Um sicherzustellen, dass bei diesem Prozess keine Informationen verloren gehen, wird nun noch einmal geprüft, ob von jedem Modul, das in der Lage ist Versionen seiner Handlungsanforderungen zu erstellen, noch zumindest eine Version der Anforderung vorhanden ist. Sollte dies dazu führen, dass keine Lösung mehr verbleibt mit welcher das vom Funktionsmodul bearbeitete Problem gelöst werden kann, so ist nur noch eine Benachrichtigung des Nutzers möglich. Das aufgedeckte Dilemma kann auf Basis der zur Verfügung stehenden Informationen dann nicht weiter vom System gelöst werden.

Bei einer Berücksichtigung aller Wirkgrößen, mit deren angegebenen Abweichungen, ist davon auszugehen, dass realitätsgetreu in weiter Zukunft keine Dilemmata auftreten. Die zur Berechnung zur Verfügung stehende Datengrundlage reicht nicht aus, um bei einem solch großen Zeithorizont präzise Aussagen zu generieren. Ein überrestriktives Verhalten der Algorithmik ist damit ausgeschlossen. Dabei wird zu Grunde gelegt, dass die Genauigkeit der erfassten Daten in der Zukunft abnimmt.

Für den Fall des Verbleibens mehrerer Versionen wird diejenige gewählt, die auf das Gesamtsystem den geringsten Einfluss erzeugt oder vom Modulentwickler als präferiert angegeben wurde. Die somit gefilterten und verarbeiteten Anforderungen, auch als Wirkgrößen bezeichnet, können nun über den bereits beschriebenen Weg mit Hilfe einer Ontologie den jeweiligen Eingängen der sogenannten Wirkmodule zugeordnet werden. Das so entstehende Gesamtanforderungsarray wird in zwei weiteren Schritten um alle Einträge bereinigt, deren Steuerung aus technischer Sicht nicht möglich ist, beispielsweise weil die dafür notwendige Hardware im Gebäude nicht vorhanden ist oder keine freien Kapazitäten mehr aufweist. Über das Wirkmodul kann nun eine Zuordnung der zu den

Wirkgrößen passenden Geräten und deren benötigter Steueraktionen erfolgen. Damit ist ein vollständiger Ablauf vom Erzeugen eines Bedürfnisses und Befriedigen desselben durch den Automationsalgorithmus durchgeführt.

Zusätzlich ermöglicht das Wirkmodul die Erzeugung zukunftsorientierter Daten, welche auf Basis der erfolgten Steueraktionen ermittelt werden können. Somit haben Wirkmodule nicht nur die Aufgabe die notwendigen Steueraktionen zu erzeugen, sondern zusätzlich zukunftsbasierte Daten an den Beginn des Algorithmus zurückzugeben und übernehmen damit die Funktion von prognostizierten Messwerten.

Diese können durch den ontologiebasierten Ansatz wieder den jeweiligen Modulen als Eingangsgrößen zugeordnet werden. Dadurch wird die Möglichkeit erzeugt, iterative Berechnungsverfahren sowie zukunftsbasierte Simulationen, wie sie beispielsweise in der Hygrothermik der Bauphysik üblich sind, durchzuführen. Um die Funktionsfähigkeit der Algorithmik zu evaluieren, wurden drei Fallstudien durchgeführt, welche jeweils einzelne Aspekte der Gesamtmethodik genauer untersuchen.

Die erste Fallstudie zeigt, dass für eine einfache Berechnung der Schimmelbildungsgefahr in einem Gebäude bei der hier beschriebenen Modulkonzipierung eine automatische Zuordnung aller notwendigen Datenströme zwischen Modulen und Sensoren erfolgen kann. Eine vollautomatische Berechnung erfolgt ohne externes Eingreifen von der Erfassung eines Sensorwertes bis zur Auslösung eines Aktuators.

Die zweite Fallstudie dient der Prüfung des Prioritäten-Algorithmus und der richtigen Bestimmung der jeweiligen, für das Gebäude notwendigen, Wirkgrößen. Durch die Implementierung nutzerabhängiger Variablen wurde auch die Möglichkeit einer kontextsensitiven Betrachtung gezeigt. Um dies zu demonstrieren wurde zusätzlich ein Modul zur Verhinderung einer zu hohen Radonkonzentration integriert, das widersprüchliche Anweisungen zum bereits vorhandenen Schimmelmodul erzeugt.

Die dritte Fallstudie zeigt die Möglichkeiten zur Individualisierung des Systems auf den Nutzer und verschiedener Gebäudespezifika. Durch die Möglichkeit, Algorithmen maschinellen Lernens mit in das System zu integrieren, kann eine signifikante Verbesserung der Berechnungsgenauigkeit in Bezug auf eine Individualisierung erlangt werden. Auch die Möglichkeit durch ein Wirkmodul Ausgangsdaten zu erzeugen, welche für iterative Berechnungen oder Simulationen genutzt werden können, wurde hier gezeigt.

Durch die Fallstudien konnte somit nachgewiesen werden, dass diese Methode dazu führt, dass in Bezug auf bauphysikalische Funktionen ein Smarthome:

- Nutzerindividuell, selbstlernend und vorhersagend
- Interaktiv und interdisziplinär
- Dynamisch
- Modular und skalierbar
- Offen und nicht geschlossen
- Kontextsensitiv und nutzerunabhängig

gestaltet werden kann.

So wird es den individuellen Bedürfnissen des Menschen in seinem privaten Umfeld gerechter, als die derzeit üblichen funktionsbasierten Automationsansätze mit „Wenn-Dann“-Regeln. Denn diese lassen nur wenig Spielraum für automatisierte Individualisierungen und Interaktionen. Die hier gezeigte Methode schließt damit bisher unbearbeitete Lücken im Bereich der Wohnraumautomation und spielt damit eine bedeutende Rolle in der voranschreitenden Digitalisierung des menschlichen Lebensraums.

Abstract

Residential buildings fulfil a wide variety of functions, such as protection from the weather and ensuring privacy. However, they are mainly the focus of our lives, especially when we look at the living space. In the course of time, the functions, that a living space has to fulfil, have changed with society. Whereas in the past the need for protection and the compatibility of craftsmanship and private life were the main functions of residential buildings, these now serve more aspects of self-realization and offer space for many private interests of the residents. The change of our society, which is also strongly influenced by the advancing digitalization, leaves its traces here as well. But how can we estimate in which direction living space will develop as a result of digitisation? The constructs depicted in science fiction films have regularly turned into real objects or techniques. While video telephony was still considered an impossible dream of the future 47 years ago, it has now become a normal part of our everyday lives.

Further examples of such developments can be found in large numbers. This leads to the conclusion that possible future developments in living space will also be influenced by this thinking about the future. The digitalization of our society will thus be accompanied by the digitalization of living space. A prominent example of the digitization of living space is the development of so-called smarthomes. These are often presented as very interactive and very intelligent systems. They often assume the position of a virtual assistant and thus create a highly individualized living space.

But even if smarthomes are already commercially available in our society, their degree of intelligence and individualization is still far from being a vision of the future. Simple regulations based on so-called "if-then" rules determine the majority of actions. Actual individualizations are by far not the standard. These living spaces also do not play the role of a real assistant, rather the role of a digital support in order to be able to control the technology of a building more easily. However, many areas of everyday life can already be described more precisely and individually than is possible with simple "if-then" rules. For example, thermal comfort can be described with innumerable individual parameters and the room temperature can be regulated on this basis. However, the integration of this knowledge into the smarthome is not yet possible. Even the representation of the high dynamics of a living environment, for example due to the changing use of individual rooms, cannot be depicted with the current systems. There is also a lack of possibilities to integrate strongly context-related data into the control of a smarthome. The number of necessary "if-then" formulations would take on proportions that would be tantamount to a completely new development of an algorithm for this case if only the residential context for a biologically effective lighting control were considered. Even functions that building physics can provide cannot be mapped due to their high potential for mutual influence and interaction. The lack of modular approaches in the smarthome, as it is already state of the art in building simulation, hinders this development as does the lack of interoperability in the smarthome-sector.

Therefore, a methodology has been developed that closes this gap. The presented approach claims to replace the current automation approach, which works by simple dependencies between events and switching actions by a much more complex and realistic approach. It opens the possibility to integrate simulation models as well as other solution methods from building physics.

In order to consider this complexity as well as the special relationships in the domestic environment, an approach was chosen which enables the interaction between user and building by a realistic illustration of the relationships.

This allows above all to ensure the factors necessary for a good smarthome, which are:

- User individuality
- interaction
- dynamism
- context sensitivity
- Modular design

In order to ensure these factors, a digital twin was defined for both the building and the user. Digital twins represent their real counterparts in the form of digital constructs. They can be created by simple databases or complex simulation models. By generating these twins, it is possible to virtually map the behaviour of buildings and users. For the processing of the resulting findings, a framework is necessary that establishes the connection between the building and the user and translates the classic automation tasks into this construct.

For this purpose, the functional approach, which is classically anchored in automation, was further divided into various modules. A clear division of mathematical and physical facts between the respective digital twins could be implemented. A simple principle based on the structure of the human need consciousness was developed for this division. The following graphic shows the division of the digital twins of the building and the user into their thematic subdivisions. The emergence of user needs and their desire for satisfaction leads to the following interaction structure. This is a basic component, which requires almost all controls in the domestic environment. This interaction structure is based on the already mentioned needs of the user, which can be described in the form of requirements. They are provided by one side, usually the user, and are to be fulfilled by the other digital twin.

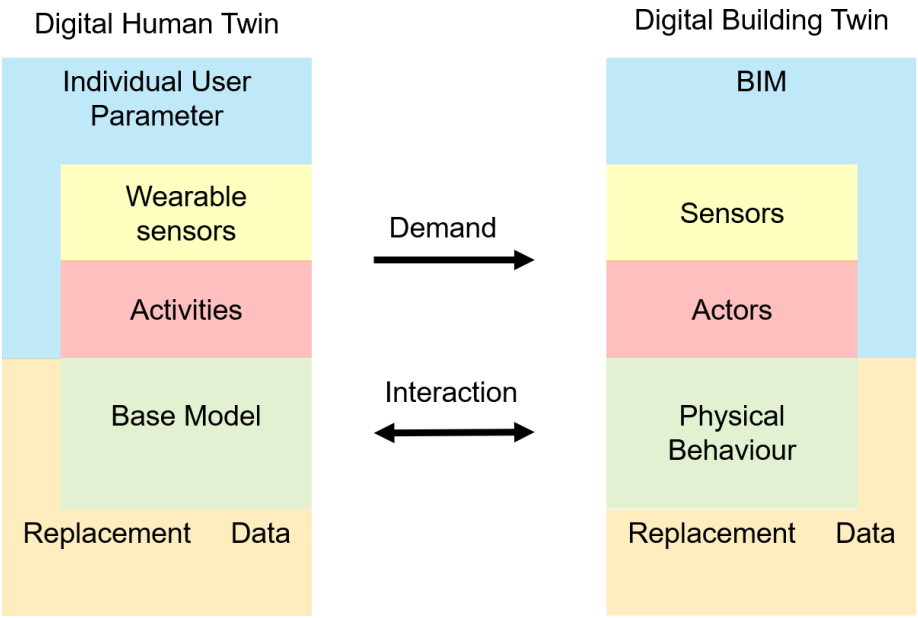


Abb. 3 Interaction model of digital building and user twinning

The individual aspects of the respective digital twin are presented in the form of modules, thus taking into account the modularization required for a smarthome.

In order to be able to represent the diversity of the different contexts, especially regarding the building physics aspects of the automation of living space, a distinction is necessary on a further scale. For this purpose, in addition to the division into user and building modules, four other basic types of modules were distinguished.

- Function-modules
- Helper-modules
- Driver-modules
- Modules for machine learning and prediction

Function modules describe the basic physical and mathematical functions of the respective twin and represent the core algorithm of the classical functions in smarthome. Helper-modules, on the other hand, represent mathematical and physical correlations that cannot be assigned to any functionality, but generate important data for calculations. With this construct it is possible to map basic physical correlations and to enable their cross-system use.

Driver-modules describe which actions are necessary to generate desired effects or calculate the effects of control actions on the environment. They thus represent the digital link to physical actions. They also generate the final control actions for devices.

Modules for machine learning and predictions extend the framework by the possibility to generate knowledge about the user and his environment from special algorithms and thus to generate strongly context-related and individualized controls.

A module in the sense of this work has a simple structure consisting of an import layer, a function block and an export layer. The data necessary for a calculation are entered into the module at the import layer. These are processed in the function block and the resulting data is output again at the export layer. The input and output of the data is always done in a standardized way. This standardization allows an automatic assignment of input variables. Input variables can have different characteristics in the sense of modules. Possible variables are sensor values, parameters from databases or output data from other modules.

In order to ensure a functional assignment of the possible input data to a module, several information about the existing data must be available. The processing of such knowledge-based assignments was realized with the help of an ontology-based algorithm.

For this purpose, digital representatives of the respective modules, sensors and various parts of the building are created in the form of an ontology. The knowledge about the respective representatives is shown in the form of their class assignment or their assignment of properties. The necessary data can be obtained, for example, from a Building Information Model. Areas of the building that are particularly relevant for building physics can be mapped with this system. This enables the methodology to independently generate connections between modules and sensors on the basis of stored knowledge about the building and the user by means of reasoning based on generated rules. This makes it possible to create an individual calculation setup for the respective building and the user living in it.

A calculation can therefore take place without prior configuration of the basic conditions by the user or an expert. Thus, standardized modules can be used for the calculation, which reflect the state of the art in the respective discipline of building physics. The substitute data to be supplied by the module developer can ensure that a calculation is still carried out even in the absence of different measured values or parameters. Thus, a storage possibility of all data types, which were seized in the form of for example standards or statistic evaluations, is given and a comprehensive collection of

standardized building and user-relevant data is possible. Modules in the area of substitute data are also possible, these create standardised building data with the aid of calculation methods.

In order to be able to replace these non-individual data with individualised data during operation, extensive analyses of the recorded measurement data of a building and the user preferences that can be derived from this are often necessary. In order to be able to perform this work, the already mentioned "modules for machine learning and prediction" were introduced. Their sole purpose is to learn various data necessary for the individualisation of home automation.

After the determination of all interactions relevant for the building, calculations are made by all modules included in the Calculation-setup. A special position is taken by the function modules, which generate initial data based on basic parameters. These represent requirements in the form of a time-based array with suggestions for action. In principle, a module is able to output different versions of suggested actions, which contribute to the solution of the problem.

Useful solution strategies, which can be issued by a module for the avoidance of mould-formation, would be for example the increase of the air-change as well as the stronger heating of the room. These strategies include the module's preferred changes to the environment from now to the future, as well as a target value, a priority and a possible deviation due to inaccuracies in the calculation. To determine the priorities, a four-step model was developed specifically for this purpose, which takes into account the different weightings of different needs.

Here, needs are divided into the categories "safety", "health", "comfort" and "other". Accordingly, the priority is "A", "B", "C", or "D". Depending on the classification of the parameters described in the following, the instructions for action of the functional modules accordingly fulfil aspects to ensure the safety, health, comfort or other interests of the user. Priorities A and B are to be regarded as essential priorities; they represent vital basic aspects in the home environment and are intended to ensure that no harm is done to the user. Priorities C and D are optional in nature and therefore do not represent harmful issues if ignored. Thus, it is possible to transfer the total sum of the proposals for action of all functional modules recorded in the system to a priority-adjusted form. The output is a forecast oriented time-scaled array.

Thus, it is also possible to consider the priorities which will change in the future. This also takes into account the increasing relevance of various aspects over time.

This priority adjustment is carried out by means of a specially developed system, which searches for the common solution spaces of the modules by consolidating the different requirement arrays. All entries that would lead to a dilemma in the future are then cleaned up. The check for impending dilemmas is carried out based on the action requirements issued by the various modules.

If, at some point in the future, these do not provide any possibility of a joint solution, a dilemma will arise through the interaction of these modules. This can lead to two possible results:

1.) The state is ignored, because the conflicting needs are comfort needs with a priority of C or D. In this case there is no real dilemma in the sense of this method and the instruction with the higher priority prevails.
2.) The second possible state is that two action requests with higher priority contradict each other. In this case, the action request must not be controlled. This is done by completely deleting the solution approaches from the overall view and is illustrated in the following diagram.

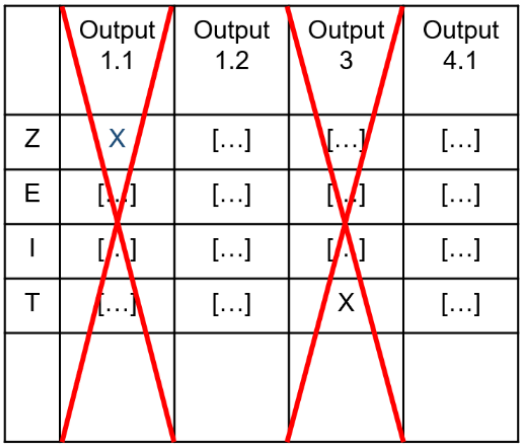


Abb. 4 Approaches to solutions to be rejected due to contradictory nature

In order to ensure that no information is lost during this process, it is now again checked whether at least one version of each module capable of creating versions of its action requests is still available. If this leads to the fact that no solution remains with which the problem processed by the function module can be solved, then only a notification of the user is possible. The dilemma that has been discovered cannot then be solved further by the system based on the available information.
Since all effective variables which are taken into account by the module with specified deviations, it can be assumed that no dilemmas will occur realistically in the distant future. The data basis available for the calculation is not sufficient to generate precise statements for such a large time horizon. An over restrictive behaviour of the algorithms is thus excluded. This is based on the assumption that the accuracy of the recorded data will decrease in the future.

If several versions remain, the one that has the least influence on the overall ecosystem or was specified as preferred by the module developer is selected. The thus filtered and processed requirements, also referred to as active variables, can now be assigned to the respective inputs of the so-called Driver-modules by means of an ontology. In two further steps, the resulting overall requirements array is cleaned of all entries whose control is not possible from a technical point of view, for example because the hardware required for this is not available in the building or no longer has any free capacities. The active module can now be used to assign the devices matching the active variables and their required control actions. This means that a complete sequence of generating a need and satisfying it is carried out by the automation algorithm.

In addition, the Driver-module enables the generation of future-oriented data, which can be determined based on the control actions performed. Thus, Driver-modules not only have the task to

generate the necessary control actions, but also to return future-based data to the beginning of the algorithm and thus take over the function of predicted measured values.

These can be assigned to the respective modules as input variables again by the ontology-based approach and thus generate the possibility of iterative calculation methods as well as future-based simulations as they are usual, for example, in the hygrothermics of building physics. In order to evaluate the functionality of the algorithms, three case studies were carried out, each of which examined individual aspects of the overall methodology in more detail.

The first case study shows that for a simple calculation of the risk of mould formation in a building, an automatic assignment of all necessary data streams between modules and sensors can take place in the module design described here. A fully automatic calculation takes place without external intervention from the acquisition of a sensor value to the triggering of an actuator.

The second case study serves to check the priority algorithm and the correct determination of the respective effective variables required for the building. By implementing user-dependent variables, the possibility of context-sensitive observation was also demonstrated. In order to demonstrate this, an additional module to prevent a too high radon concentration was integrated which generates contradictory instructions to the already existing mould module.

The third case study shows the possibilities to individualize the system to the user and different building specifics. By the possibility of integrating algorithms of machine learning also into the system, a significant improvement of the computation accuracy can be attained regarding an individualization. Also, the possibility to generate output data, which can be used for iterative calculations or simulations, was shown here.

The case studies showed that this method leads to a smarthome, which is

- User-specific, self-learning, predictive
- Interactive and interdisciplinary
- Dynamic
- Modular and scalable
- Open and not closed
- Context-sensitive and user-independent

This makes it more suitable for the individual needs of people in their private environment than the currently common function-based automation approaches with "if-then" rules. These leave little room for automated individualization and interaction. The method shown here thus closes hitherto unresolved gaps in the field of housing automation and thus plays an important role in the advancing digitalisation of human living space.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BIM	Building Information Modeling
BPS	Building Performance Simulation
CFD	Computational Fluid Dynamics
DGZ	Digitalen Gebäude Zwilling
DNZ	Digitalen Nutzer Zwilling
CO^2	Kohlenstoffdioxid
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
IDS	Input Datenströme
KI	Künstliche Intelligenz
MIT	Massachusetts Institute of Technology
ODS	Output Datenströme
OWL	Web Ontology Language
Radonmodul	Funktionsmodul zur Beschreibung der zulässigen Radonkonzentration für Menschen
RDF	Resource Description Framework
Schimmelmodul	Funktionsmodul zur Berechnung der Schimmelbildungsgefahr
SWL	Semantic Web Technologies
SWRL	Semantic-Web-Rule-Language
UCAmI	Ubiquitous Computing and Ambient Intelligence
W3C	World Wide Web Consortium
z.B.	zum Beispiel

1 Einleitung und Motivation

Menschen nutzen seit jeher Unterstände, um sich vor Wetter und anderen Gefahren zu schützen. Mit der Entwicklung und der Ausprägung von Gesellschaftsstrukturen entwickelten sich so auch diese Unterstände, die zunächst einen reinen Schutzcharakter hatten, zu geschlossenen Häusern weiter, die nun bei weitem mehr als nur diese Funktion erfüllen. Gebäude sind zum zentralen Mittelpunkt des Lebens in unserer Gesellschaft geworden.
Im Jahr 2016 wurde die Zahl der Obdachlosen, welche kein Gebäude bewohnen in Deutschland auf 860.000 Personen geschätzt (Bundesarbeitsgemeinschaft Wohnungslosenhilfe e. V., 2017). Bei der aktuellen Bevölkerungszahl von 83 Millionen Menschen (Statistisches Bundesamt, 2019) folgt daraus, dass 96,5 % der Bevölkerung in Gebäuden wohnen. Allein die Tatsache, dass auch auf europäischer Ebene eine ausgiebige Diskussion der Thematik der Obdachlosigkeit stattfindet, zeigt, dass das Bewohnen eines Hauses in unserer Gesellschaft einen nicht wegzudenkenden Wert besitzt (Busch-Geertsema, 2018). Der Schutzcharakter des Gebäudes ist weiterhin geblieben, auch wenn der Fokus in der Planung von Gebäuden längst nicht mehr nur auf den Schutz vor Natureinwirkungen gelegt wird. In der heutigen Zeit sind architektonische sowie technische Aspekte der Gebäudeplanung im Vordergrund. Die zu beantwortenden Fragen bei der Planung drehen sich um Themen wie Ästhetik, technische Funktionalität, Energieeffizienz und viele mehr. Nachdem der Mensch die Schutzfunktion des Wohnraums beherrscht, scheint es, als würde dem Gebäude nun eine weitere Funktion zugeschrieben. Selbstverwirklichung ist das neue Ziel bei der Gebäudeplanung (dial.de, 2015).
Der Nutzer soll nun eine Immobilie erhalten, die die üblichen Schutzfunktionen bietet. Zusätzlich soll der Bau aber auch noch viele andere individuelle Bedürfnisse erfüllen. Auch die Veränderung der Grundrisse von Gebäuden im Laufe der Zeit spiegeln diese Beobachtung wider. Während früher die Gebäude noch einen funktionalen Charakter hatten und Menschen versuchten das Leben darin mit dem verrichteten Handwerk zu verbinden, was zu neuen Raumbezeichnungen im Grundriss führte wie „Stall“, „Werkstatt“, „Lager“ etc. so sind nun Wörter wie „Hobbyraum“, „Heimkino“, „Sauna“, „Fitnessraum“ etc. zu finden. Das Wohngebäude ist nun also ein Teil der Selbstverwirklichung des Menschen geworden. Damit wird versucht, die Architektur nun so individuell wie möglich zu gestalten. Menschen, die ihre Selbstverwirklichung im Sport sehen, erhalten ein auf ihre Bedürfnisse zugeschnittenes Eigenheim mit einem „Fitnessraum“ oder sogar einem „Schwimmbad“. Wer handwerkliche Tätigkeiten zu seiner Selbstverwirklichung bestimmt hat, baut stattdessen eine „Werkstatt“. Durch die Raumdefinition sind also viele Tätigkeitsbereiche in das Gebäude eingezogen, die früher nicht denkbar gewesen waren.
Nun wandelt sich seit mehreren Jahren unsere Gesellschaft in eine „digitale“ Gesellschaft. Viele Teile unseres Lebens werden in digitale Medien ausgelagert. Soziale Kontakte knüpfen wir nun über Plattformen wie „Facebook“ oder „Instagram“. Kommunikation findet örtlich getrennt über Messenger-Plattformen und Bildtelefonie statt. Auch Teile unserer Selbstverwirklichung verschieben sich damit in eine „digitale Richtung“. Was im Kleinen begonnen hat, entwickelte sich im Laufe einiger Jahre zu einem Zustand, in dem man von einem „… allgegenwärtige[n] Wandel durch Digitalisierung …“ (Schawel & Billing, 2018) spricht. Alles wird digitalisiert, alles wird vernetzt. Dies führte letztendlich auch zur Prägung des Begriffes „Internet of Things“ und spiegelt den Prozess wider, den unsere Gesellschaft derzeit erlebt. Die physische Trennung der „digitalen Welt“, repräsentiert durch das Internet, ist nun aufgehoben. Auch materielle Objekte können nun Teil einer digitalen Welt sein. Dass dieser Prozess Auswirkungen auf unser Leben und unseren Lebensmittelpunkt hat, sollte zu erwarten sein.
Daher scheint es kaum verwunderlich, dass auch in der Baubranche das Thema Digitalisierung vorherrschend ist und so das Schlagwort „Smarthome“ als Repräsentant für die Digitalisierung des

Wohnraums geprägt wurde. In der Vision soll das Smarthome das Leben des Bewohnenden erleichtern, es soll ihm die Möglichkeit geben, ganz individuell und kontextbezogen Steuerungen vorzunehmen:
„These services make the lives of householders easier, as individuals gain finer control over their environment by accessing a variety of context- and situation-aware applications." (gsma.com, 2011)
Wenn man sich im Bereich der Science-Fiction-Filme einen Überblick über Smarthomes verschafft, wird jedoch klar, dass diese noch einen Schritt weiter gehen. Filme wie „Iron Man" zeigen uns Gebäude mit virtuellen Assistenten und eigener Persönlichkeit. Sie erlauben eine Kommunikation mit dem Menschen, auch eigenständiges Denken und Lösen von Problemen ist großer Bestandteil dieser Zukunftsvisionen. Dadurch ist ein viel höheres Maß an Individualisierbarkeit im Wohnraum und vor allem bei der Steuerung des Wohnraums möglich. (Favreau, 2018)
Science-Fiction-Schöpfungen haben zu Entwicklungen in eben diese Richtung geführt. Beispiele wie das iPad von Apple (Star Trek, 1966), der Sprachassistent Alexa von Amazon (Hal9000, 2001), Videotelefonie (Metropolis 1927), Hologramme (Star Wars, 1977), virtuelle Realität („Welt am Draht" 1973 oder Matrix 1999) zeigen, dass Zukunftsvisionen, auch wenn sie in der jeweiligen Zeit als absolut unmöglich in der Umsetzung galten, letztendlich dazu führen können, dass genau diese Umsetzung erreicht wird. (Jung, 2018)
Unter diesem Aspekt scheint ein Smarthome, das eine hochindividuelle Wohnumgebung gewährleistet und mit dem Nutzer in eine Interaktion tritt, nicht abwegig. In der Realität befinden wir uns jedoch noch weit von diesem Ziel entfernt. Aktuelle Steuerungen im Wohnbereich stützen sich noch hauptsächlich auf einfache Logiken mit so genannten „Wenn-Dann-Schaltungen". Diese erfreuen sich so großer Beliebtheit, dass sogar eigene Plattformen zur Definition dafür geschaffen wurden, wie am Beispiel der Plattform IFTTT zu sehen ist (IFTTT Inc., 2019). Von komplexen Steuerungen, wie sie in Science-Fiction-Filmen vorkommen, ist die digitale Version des Wohnraums derzeit noch weit entfernt. Auch kann man bis zum jetzigen Zeitpunkt nicht von Wohnungen sprechen, welche eine eigene Persönlichkeit repräsentieren oder die Fähigkeit eigenständiger Problemlösung besitzen. Zwar haben Sprachassistenten durch ausgeklügelte Technik die Möglichkeit uns zu suggerieren, dass wir individuelle Antworten erhalten, letztendlich sind die derzeitigen Systeme jedoch nur Antwortautomaten mit einer hochwertigen Spracherkennung.
Auch wenn die Technik für individuelle Steuerungen im Smarthome bereits vorhanden ist, ist die mangelnde Interoperabilität und die fehlende „Intelligenz" in der Steuerung derzeit das größte Hemmnis auf dem Weg zum intelligenten Smarthome. Das Wissen für individuelle Steuerungen im Gebäude liegt vor allem in Disziplinen wie der Bauphysik bereits seit mehreren Jahren vor, konnte jedoch noch keinen Weg in die Wohnraumsteuerung finden. So wurde beispielweise bereits 1973 durch Fanger ein System entwickelt, um der Beschreibung der individuellen Behaglichkeit einen Schritt näher zu kommen (Fanger, 1973). Den Wohnraum temperieren wir aber immer noch durch Messung der Lufttemperatur und Vorgabe eines Sollwertes.
Hieraus entstand die Idee, sich dieses bereits vorhandene Wissen auf dem Weg zum intelligenten Smarthome zu Nutze zu machen. Das Ziel ist ein System zu entwickeln, dass es ermöglicht dieses Wissen aus der Bauphysik in all seinen Bereichen in ein Smarthome integrierbar zu machen. Gesucht wurde also ein integraler Ansatz, der es ermöglicht, das Wissen im Bereich der Bauphysik, in der Form von Funktionalitäten, in ein Smarthome zu integrieren. Dabei wurde vor allem versucht Raum zu schaffen, um Individualisierungen auf den jeweiligen Nutzer zu ermöglichen.
So soll die vorliegende Dissertation einen Anteil auf dem Weg zur Digitalisierung des Wohnraumes beitragen und einen weiteren Schritt in Richtung intelligente Wohnraumsteuerung ermöglichen. Auch wenn die Bauphysik nicht alle Bereiche im häuslichen Wohnraum abdecken kann und viele Bereiche noch lange nicht ausreichend erforscht sind, so kann mit diesem Ansatz doch ein großer Bereich der derzeit üblichen Steuerungsaufgabe im Smarthome ein Stück näher an die Zukunftsvision gebracht werden.

2 Methodisches Vorgehen

Im Vorfeld zu dieser Arbeit wurde eine systematische Literaturrecherche zur Integration bauphysikalischer Funktionen im automatisierten Wohnraum durchgeführt. Das Ergebnis war, dass ein umfassender integraler Ansatz für eine Implementierung bauphysikalischer Funktionen in das Smarthome fehlt. Auch wenn einzelne Aspekte exemplarisch umgesetzt wurden, lässt sich die Interaktion unterschiedlicher bauphysikalischer Betrachtungen im Wohnraum bis zum jetzigen Zeitpunkt nicht abbilden.
Im Anschluss wurden für mehrere interagierende Bereiche der Bauphysik die Lösungsstrategien, die von Experten der Bauphysik verfolgt werden, analysiert. Hierdurch konnte ein einfacher Ansatz unter Berücksichtigung von Prioritäten, zur Beschreibung dieser Interaktionen, entwickelt werden.
Auch zeigte sich bei den Literaturrecherchen, dass bauphysikalisches Wissen derzeit hauptsächlich bei der Planung oder der Begutachtung von Schäden Einsatz findet. Bei Letzterem wird häufig das Nutzerverhalten in Korrelation mit Normwerten gesetzt (Pohl, 2002). Dieser Zusammenhang ließ auf eine starke Relevanz des individuellen Nutzerverhaltens schließen und muss daher berücksichtigt werden.
In weiteren Recherchen zum Bereich des Smarthome-Sektors zeigen sich starke Defizite in Bezug auf die Interoperabilität und Interaktion verschiedener Funktionsbereiche. Außerdem werden Mängel in der Konzipierung der Strukturen von Smarthomes erkennbar. Dies führt weitestgehend dazu, dass reale Smarthomes noch weit von den Zukunftsvorstellungen entfernt sind.

Im Rahmen dieser Arbeit wurde daher mit Hilfe einer MATLAB-basierten Eigenentwicklung ein Framework geschaffen, das die Funktionsweise der entwickelten Methodik sowie die verschiedenen Teilaspekte, die die Methodik erfüllen muss, nachweisen kann. Der Nachweis der einzelnen Kriterien erfolgt daraufhin in Form von Fallstudien.

In den folgenden Unterkapiteln wird die Zielsetzung der Arbeit sowie die Abgrenzung des Themas definiert. Außerdem wird der Aufbau dieser Arbeit beschrieben.

2.1 Zielsetzung der Arbeit

Ziel dieser Arbeit ist es, eine Systematik bereitzustellen mit deren Hilfe eine Integration von bauphysikalischen Funktionen im Smarthome ermöglicht wird. Damit beantwortet sie die Frage, wie sich bauphysikalische Funktionen im Smarthome integrieren lassen und welche Ansätze erforderlich sind, um die besonderen Eigenheiten dieser Funktionen richtig im Umfeld eines automatisierten Wohnraums abzubilden. Besonderer Fokus liegt hierbei auf dem Interaktionspotential verschiedener Funktionen untereinander.
Verschiedene bauphysikalische Zusammenhänge können bei einer Einzelbetrachtung zeitweise zu widersprüchlichen Handlungsanweisungen führen. So kann die notwendige Luftwechselrate zur Reduktion der Radonbelastung in einem Kellerraum, bei hohen Außentemperaturen und relativen Luftfeuchten, zu einer Erhöhung des Schimmelrisikos in demselben führen. Eine Lösung für diese Problematik kann nur gefunden werden, wenn sowohl eine Aussage über die Höhe des Schimmelrisikos getroffen werden kann, als auch die Anwesenheit des Nutzers mitberücksichtigt wird. Denn nur bei seiner Anwesenheit ist eine Reduktion der Radonbelastung notwendig.
So kann bereits bei diesem Beispiel geringer Komplexität gezeigt werden, dass für einen Steuerungsansatz in einem Smarthome eine Einzelbetrachtung der Sachverhalte nicht zielführend ist. Es wird also eine Methode benötigt, welche die Randbedingungen beider Themen berücksichtigt und eine Bewertung vornimmt um die Steuerung dem Kontext entsprechend umzusetzen. Mittels der

Methode muss man also in der Lage sein, einen integralen Ansatz zur Abbildung bauphysikalischer Phänomene bereitstellen zu können. Hinzu kommt, dass sehr viele bauphysikalische Phänomene stark von nutzerindividuellen Parametern abhängen.
Während in der Planung das Rechnen mit Norm- und Durchschnittswerten logisch und ist, so ist diese Vorgehensweise im personalisierten Wohnraum nicht mehr zielführend. Eine Unterschätzung des individuellen Duschverhalten eines Nutzers, zum Beispiel mit statistischen Werten, kann in Einzelfällen sogar zu Schäden führen. Daher sollen in der bereits genannten Methode genügend Raum und Mittel generiert werden, um die Möglichkeit für Individualisierungen in diesen Funktionen bereitzustellen.

Ziel ist es, mit der Methode die Möglichkeit zu eröffnen, dass bauphysikalische Funktionen so personalisiert und individuell berechnet werden können, wie es der Stand des Wissens zulässt. Dabei muss es möglich sein, die zu verwendenden Parameter dem System vorzugeben oder sie vom System selbst erlernen zu lassen. Auch diese Möglichkeiten muss die Methode bereitstellen und unterstützen. Aber wie am oben genannten Beispiel der Radonbelastung beschrieben, ist die reine Verwendung nutzerindividueller Parameter im Smarthome noch nicht ausreichend, um die nötige Genauigkeit bei der Abbildung bauphysikalischer Interaktionen zu erzeugen. Um diese sicherzustellen ist auch die Berücksichtigung des genauen Kontextes und der Randbedingungen erforderlich. Deshalb ist es ein weiteres Ziel dieser Methodik den Rahmen für die Erfassung dieser kontextrelevanten Parameter zu erstellen und in der notwendigen Auflösung zu verarbeiten ohne dabei die Interaktion verschiedener Funktionen zu stören. Um das gewährleisten zu können ist sicherzustellen, dass eine genaue und fehlerfreie Zuordnung von Daten im Smarthome erfolgt. Interoperabilitätsprobleme sind jedoch noch das größte Problemfeld im Smarthome. Eine automatische richtige Zuweisung von Datenströmen kann bis jetzt noch nicht erfolgen.

Ein weiteres Ziel dieser Arbeit ist es deshalb, die Zuweisung der Datenströme zu automatisieren. Damit kann ein essenzieller Anteil an der Lösung des Interoperabilitätsproblems im Smarthome-Sektor beigetragen werden. Da der Smarthome-Sektor ein sich sehr schnell entwickelnder Sektor ist (Deloitte Touche Tohmatsu Limited, 2018) und die aktuellen Lösungen von den bereits vorhandenen technischen Möglichkeiten noch weit entfernt sind (Deloitte & Technische Universität München, 2015), ist es außerdem ein weiteres Ziel dieser Methode das System so offen zu generieren, dass auch neue Lösungen oder Methoden integriert werden können und keine starken Einschränkungen entstehen.

2.2 Abgrenzung des Themas

Die hier vorgestellte Methodik soll einen grundlegenden Ansatz zeigen, um bauphysikalische Funktionen im Smarthome unter Einbezug aller relevanten Zusammenhänge zu erfassen und daraus Steuerungen abzuleiten. Im Rahmen der Entwicklung zeigte sich ein stark modularer Ansatz als vielversprechendes Mittel um dies zu realisieren. Durch die Bildung von Modulen und der damit einhergehenden Verkapselung von Quellcode ist der Inhalt der Module für die Prüfung der Funktionsfähigkeit des Ansatzes nicht mehr relevant. Aus diesem Grund wird die Gestaltung und die Richtigkeit des Inhaltes und damit der in Modulen implementierten Funktionen nicht weiter berücksichtigt.
Über den grundsätzlichen Aufbau der Module und deren Schnittstellen werden Rahmenbedingungen und Notwendigkeiten definiert. Eine genaue Anweisung, welche Funktion in welcher Form in ein Modul integriert werden kann, ist jedoch nicht Ziel und Aufgabe dieser Arbeit.
Die Integration von Simulationsframeworks zeigt keinen Einfluss auf die in dieser Arbeit vorgestellte Grundmethodik. Deshalb wird ebenso wenig auf die genauen Zusammenhänge zur Entwicklung oder

Implementierung des im Folgenden angesprochenen und notwendigen Master-Algorithmus eingegangen. Dieser ist zur Integration von Simulationsmodellen wie zum Beispiel „Functional Mock-up Units“ notwendig.
Auch wenn eine grundsätzliche Lösung des Interoperabilitätsproblems im Smarthome, welches Probleme bei der Zusammenarbeit verschiedener Systemteile beschreibt, eine Zielsetzung dieser Arbeit ist, so bezieht sich diese Lösung doch nur auf die Verarbeitung von Daten. Eine Lösung des Interoperabilitätsproblems auf Hardware-Ebene ist weder für die vorgestellte Methode relevant noch ihr Ziel.
Somit kann die Methode also grundsätzlich als Framework für die integrale Implementierung von bauphysikalischen und auch anderen Funktionen auf mathematischer oder physikalischer Basis angesehen werden.
Um das Framework in seiner grundsätzlichen Funktion zu testen, werden im weiteren Verlauf der Arbeit Module und Daten erstellt, um sowohl das Problem der Schimmelbildung als auch der Radonkonzentration im Wohnraum zu beschreiben. Die Funktionalität des Algorithmus ändert sich in seiner Funktionsweise nicht in Abhängigkeit der Anzahl der in einem Smarthome anwesenden Nutzer. Daher wurde im Sinne einer besseren Nachvollziehbarkeit die Anzahl der anwesenden Nutzer auf eine Person beschränkt. Für die Anwesenheit mehrere Nutzer unterscheidet sich die Vorgehensweise in den vorgestellten Fallstudien kaum, grundsätzlich gibt es aber Themenbereiche, wie beispielsweise die thermische Behaglichkeit, in denen sich die Anforderungen verschiedener Nutzer überlagern können. Die grundlegenden Mittel zur Lösung solcher Problematiken werden von der hier vorgestellten Methode in Form der Prioritätenverarbeitung jedoch erfasst und sollen daher im Folgenden nicht weiter betrachtet werden.

2.3 Aufbau der Arbeit

Die Arbeit ist in acht Kapitel untergliedert.

Das erste Kapitel bietet eine Einleitung, die auf die Relevanz und die Hintergründe der Thematik und ihren Kontext hinweisen.

Das zweite Kapitel definiert die Zielsetzung so wie die Limitationen dieser Arbeit. Weiterhin ist das methodische Vorgehen beschrieben.

Im dritten Kapitel wird der aktuelle Stand des Wissens genauer erläutert. Im Bereich Smarthome werden technische Grundlagen zusammengefasst. Darauffolgend wird auch der Stand des Wissens zu den Themen Aktivitätserkennung, Kontextsensitivität und künstlicher Intelligenz in den für diese Arbeit relevanten Bereichen erörtert. Abschließend werden Grundlagen zu den für die Fallstudien verwendeten Themenbereichen Radon im Wohnumfeld und Schimmelbildung im Wohnbereich zusammengefasst.

Das vierte Kapitel behandelt die neu entwickelte Methodik und zeigt ihren systematischen Aufbau. Dazu werden in einem ersten Schritt der digitale Nutzerzwilling (DNZ) sowie der digitale Gebäudezwilling (DGZ) definiert. Darauffolgend wird die Definition der notwendigen Module für die Zwillinge durchgeführt und deren Interaktion so wie das zur Interaktion notwendige Framework generiert.

Um im Rahmen von Fallstudien die Methodik evaluieren zu können, werden im fünften Kapitel Informationen zur Implementierung der einzelnen Bestandteile der Methodik umrissen. Es wird ein Überblick über notwendige technische Voraussetzungen zur Umsetzbarkeit der Methode erstellt.

Die Evaluierung der Methodik folgt daraufhin in Kapitel sechs in Form von drei Fallstudien. Deren Aufbau wird zuerst beschrieben und danach ihre Durchführung und deren Ergebnisse dokumentiert und analysiert. Hier hat jede Fallstudie ein anderes Evaluationsziel. Fallstudie 1 beweist, dass die automatische Zuordnung von Datenströmen funktioniert und zeigt die Funktionstüchtigkeit des Gesamtalgorithmus mit einer bauphysikalischen Funktion im Smarthome. In Fallstudie 2 wird diese um eine zweite Funktion erweitert. Somit kann die Funktionalität der Interaktion und auch kontextbasierter Steuerungen im Wohnraum gezeigt werden. Hierbei kann eine Betrachtung beider Funktionen und auch daraus entstehender eventuell gegenläufiger Steuerungsaktionen im Smarthome berücksichtigt werden. Fallstudie 3 zeigt neben der Möglichkeit von Individualisierung auch die Integrierbarkeit von Algorithmen des maschinellen Lernens.

Im siebten Kapitel wird daraufhin überprüft, ob mit der Methodik alle in Kapitel drei aufgestellten Kriterien erfüllt werden konnten und welchen Limitationen die Methodik unterliegt.

Daraufhin wird im letzten Kapitel ein Fazit erstellt und ein Ausblick auf weitere Potentiale und sich öffnende Forschungsfelder durch die Methodik gegeben.

Dem folgenden Anhang können technische Details sowie weitere für diese Arbeit notwendige Daten entnommen werden.

3 Stand des Wissens

Die Ergebnisse aus systematischen stichwortbasierten Literaturrecherchen zu den Bereichen Smarthome und zu Algorithmischen-Konzepten, zur Abbildung verschiedenster Aspekte eines zukunftsgerechten Smarthomes, werden im folgenden Kapitel näher erläutert. Ausgehend von einer kurzen Beleuchtung der Forschungslandschaft im Bereich Smarthome wird dieser Begriff definiert und die für ein erfolgreiches Smarthome im oben geschilderten Sinn notwendigen Parameter aufgezeigt. Anschließend werden Konzepte zu der Algorithmik und verschiedenen Konzepten, die im weiteren Verlauf der Arbeit verwendet werden, erklärt. Als Abschluss werden die notwendigen Inhalte, der für die Evaluation der Methodik erstellten Fallstudien, kurz aufgezeigt.

3.1 Smarthome

Die deutsche Übersetzung für das Wort „Smarthome“ ist „intelligentes Zuhause“ (Wisser, 2018). Der Begriff prägt die auf den Wohnraum spezialisierte Automation von Gebäuden. Aktuell finden immer mehr Geräte sowie darauf aufbauende Dienstleistungen in diesem Sektor Einzug. Auch die Hauselektronik spielt neben der Datenerhebung und der Regelung von Gebäudefunktionen inzwischen in diesem Bereich eine große Rolle. (Strese, Seidel, Knape, & Botthof, 2010) Folgende Tabelle zeigt die nach Strese et al. (2010) derzeit vorliegenden Teilbereiche des Smarthomes mit jeweils einem Beispiel.

Tabelle 1 Teilsysteme im Smarthome (Strese et al., 2010, S. 2-3)

Teilbereich	**Beispiel**
Heizung	Heizungsanlagen
Lüftung/Klima	Schadstoffableitung
Sanitär	Wasser, Armaturen
Elektrik	Installation und Verteilung
Energiemanagement	Lastverteilung und Prognose
Licht	Beleuchtung und Lichtmanagement
Zutritt	Zutrittskontrolle
Überwachung	Feuer, Gas, Einbruch
Notfall	Unabhängige Stromversorgung, Fluchtwege
Metering	Verbrauchszähler für Strom, Gas, Wasser
Umfeld	Gartenberegnung
Kommunikation	Lan, WAN, Inter-/Intranet
Konsumelektronik	TV, Audio, Internet
Hausgeräte	Kühlschrank, Waschmaschine
Gesundheit/ Pflege	Medizinische Diagnostik und Vorsorge
Heimlogistik	Einkaufs- und Speiseplanung
Hobby	(elektronische) Spiele, Aquarienmanagement
Mobilität	e-Mobility, Navigationssystem

Die Schlüsselfaktoren eines Smarthomes, nach dem aktuellen Stand der Technik, sind eine starke Vernetzung der einzelnen Komponenten des Gebäudes und eine Ausstattung mit einer Form von Intelligenz in der Art der Ansteuerung.

Nach Strese et al. (2010) ist der Nutzen, der zur Akzeptanz eines Smarthomes führt, in folgenden Bereichen anzufinden:

- Sicherheit
- Komfort
- Wirtschaftlichkeit im Betrieb

(Strese et al., 2010)

Die Verarbeitung und das Zusammenwirken aller Komponenten ist erst durch die Vernetzung möglich. Als in den 60er Jahren die ersten Computernetzwerke entwickelt wurden, waren Computer meist nur für eine spezielle Aufgabe konzipiert. Um ihre Leistungsfähigkeit zu erhöhen und Datenaustausch zu ermöglichen wurden sie zu Rechnernetzen zusammengeschlossen (Baun, 2018, S. 15). Durch diese Entwicklung entstanden die Netzwerke, welche nun als essenzielle Grundlage für Smarthomes angesehen werden können. Nachfolgende Grafik gibt einen Überblick über derzeit in Deutschland ansässige Initiativen und Forschungsprojekte zum Thema Smarthome.

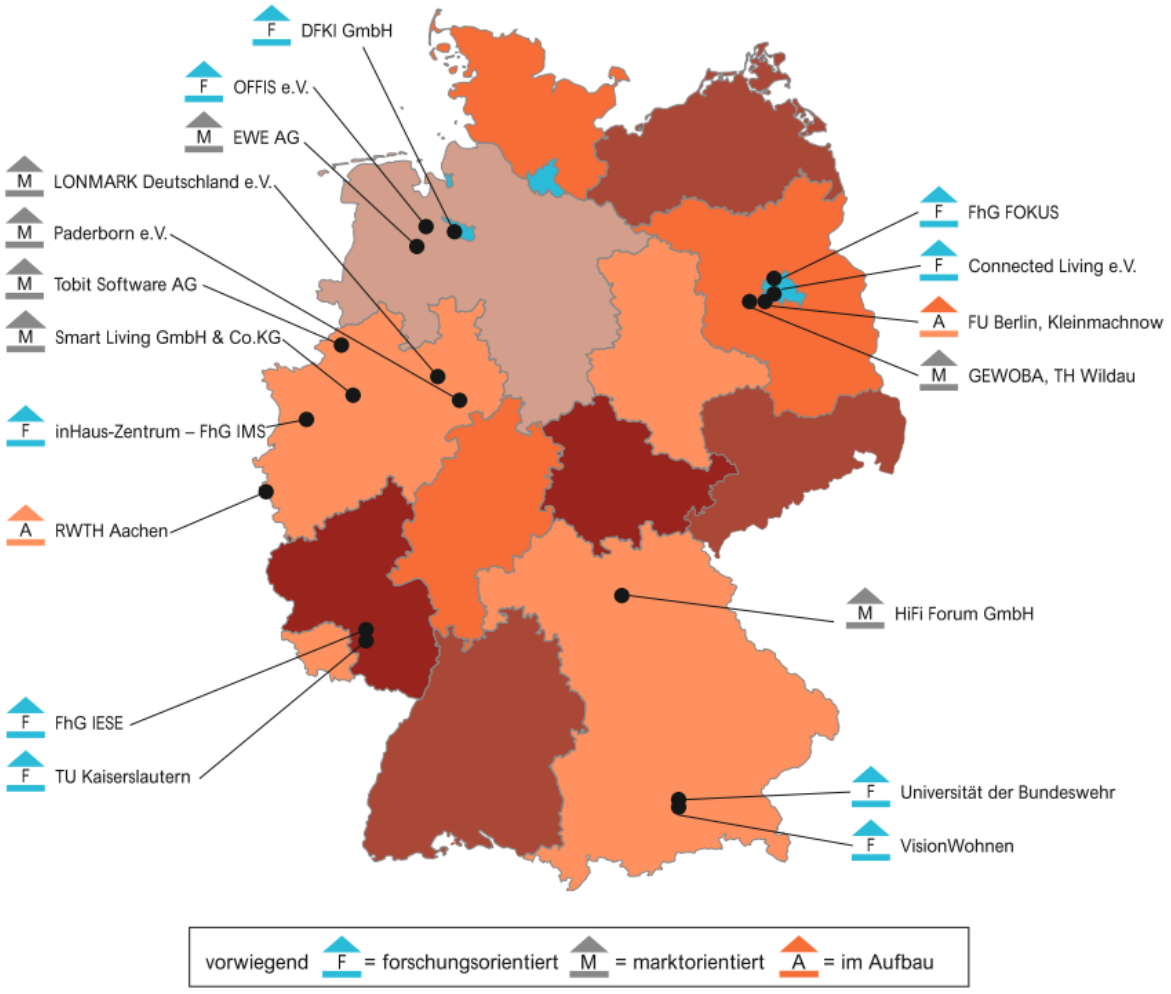


Abb. 5 Smarthome Forschung in Deutschland (Strese et al., 2010, S. 16)

Auffällig ist dabei, dass die größte Anzahl der Gebäude Präsentationsgebäude der jeweiligen Hersteller von Produkten im Smarthome-Sektor sind. Forschungseinrichtungen oder Initiativen beschäftigen sich hauptsächlich mit der Thematik des altersgerechten unterstützten Wohnens.

Für die weitere Entwicklung des Smarthomes in der Zukunft prophezeit der Bericht „Smart Home in Deutschland“ von Strese et al. (2010) und herausgegeben vom Bundesministeriums für Wirtschaft und Technologie bereits für das Jahr 2030:

„Es ist zu erwarten, dass in technische Artefakte des Smart Home kognitive Fähigkeiten integriert sein werden, um intuitive, adaptionsfähige und personalisierte Lösungen bereitstellen zu können.“ (Strese et al., 2010, S. 38)

Zudem existiert folgende Prognose:

„Smart Home-Funktionalitäten und -Dienste werden zur Basisausstattung von Wohnungen.“ (Strese et al., 2010, S. 38)

Die aktuelle Situation im Smarthome-Bereich ist jedoch geprägt von Hürden, wie mangelnder Interoperabilität, hohen Investitionskosten, mangelnder Nutzerakzeptanz, intransparentem Datenschutz, fehlender Modularität und Integrierbarkeit. An tatsächlichen „intuitiven, adaptionsfähigen und personalisierten Lösungen“ (Strese et al., 2010, S. 38) mangelt es derzeit noch gänzlich. (Strese et al., 2010, S. 38)

3.1.1 Definition Smarthome

Schon Lütolf (1992) definierte das Smarthome wie folgt:

„The Smart Home concept is the integration of different services within a home by using a common communication system. It assures an economic, secure and comfortable operation of the home and includes a high degree of intelligent functionality and flexibility“
(Lütolf, 1992, S. 1)

Die Definition einer intelligenten Maschine, wie man ein Smarthome bezeichnen könnte, gestaltet sich jedoch schwierig. Ein Indiz für die Fähigkeiten einer solchen Maschine erlaubt jedoch der sogenannte „Turing-Test“. Bei diesem müssen Mensch mit einer Maschine in Form von Fragen interagieren. Dabei weiß der Fragende nicht ob der Interaktionspartner eine Maschine oder ein Mensch ist. Eine Maschine ist als intelligent zu bezeichnen, wenn 30% der Probanden keinen Unterschied zwischen dem maschinellen und dem menschlichen Interaktionspartner erkennen. (Lenzen, 2018, S. 25)

Bereits zu Beginn der Entwicklung des Begriffes „Smarthome“ zeigten sich also folgende Aspekte als prägend für die Thematik:

- Verschiedene Akteure („Services“)
- Vernetzung
- Ökonomie
- Sicherheit
- Komfort
- Hohes Maß an Intelligenz
- Fokus auf Funktionalität
- Flexibilität

Nach aktuellem Stand des Wissens enthält ein Smarthome folgende Aspekte:

- Lernfähigkeit
- Personalisierung
- Automatisierung
- Interoperabilität
- Vernetzung
- Programmierbarkeit
- Fernbedienbarkeit

(Leitner, 2018, S. 55)

Die Auflistung wurde von Leitner (2018) in der Reihenfolge so erstellt, dass die Verfügbarkeit der Aspekte im Smarthome von unten nach oben abnimmt. Die Intelligenz und damit die „smartness“ des Smarthomes nimmt jedoch von unten nach oben zu.

Das Smarthome im Sinne dieser Arbeit kann demnach als Gebäude mit vernetzter Technik gesehen werden, welches durch intelligente Steuerung versucht flexibel den jeweiligen benutzerspezifischen Vorzügen gerecht zu werden. Dies tut es in den fünf Hauptbereichen Sicherheit, Gesundheit, Komfort, Energiemanagement und Multimedia.

3.1.2 Smarthome-Interaktion Mensch-Maschine

Der Mensch im Wohnraum interagiert mit dem Gebäude auf unterschiedlichste Arten. Wenn das Smarthome als intelligent angesehen wird, kann diese Interaktion mit dem Wohnraum auch zu einer Kommunikation werden. Kommunikation ist definiert als:

„Prozess der Übertragung von Nachrichten zwischen einem Sender und einem oder mehreren Empfängern.“ (G. W. Maier, 2018)

Während im klassischen Sinne bis heute der Empfänger immer das Smarthome war und der Sender der Nutzer, der Befehle gibt, muss bei der Verwendung intelligenter Systeme davon ausgegangen werden, dass mit steigendem Grad der Intelligenz auch die Kommunikation in die andere Richtung relevant wird. Die derzeit üblichen verschiedenen Kommunikationswege im Bereich der Smarthomes sind:

- Bildübertragung
- Textanzeigen
- Befehlseingaben
- Sprachein- und ausgaben
- Signale in Form von Tönen oder Lichtern
- Ein- und Ausgaben über Taster, Displays oder Smartphone-Apps

Auch wenn derzeit noch haptische Kommunikationswege mit dem Smarthome überwiegen, ist doch davon auszugehen, dass die Verwendung von Sprache auf längere Sicht dominieren wird (Gräfe, 2017). Dies hat sicherlich auch den Hintergrund, dass es für den Menschen die natürlichste Kommunikationsform ist. Marktführende Unternehmen im Bereich der Sprachassistenten sind derzeit Amazon und Google. Sie haben erkannt, dass die Sprache der Kommunikationsweg des Smarthome werden wird und vermutlich deshalb bereits sehr früh ihre Sprachassistenten mit Möglichkeiten der Steuerung im häuslichen Umfeld versehen. Auch der Sprachassistent „Siri“ von Apple und dessen Anbindung an das appleeigene Smarthome-Protokoll „Homekit“ zeigen diesen Trend. Vorteil vor allem in Kombination mit der Aktivitätserkennung ist, dass diese Geräte technisch in der Lage sind die Nutzer anhand ihrer Stimme zu unterscheiden (Hüber, 2018).

Auch wenn aktuell als Kommunikationsmittel die Verwendung von Apps gängig ist, so werden sich im Laufe der Zeit doch für den Menschen intuitivere Kommunikationswege durchsetzen. Der aktuelle Vorteil von Apps ist die hohe Verbreitung von Smartphones (Panella & Altilio, 2019) bei Smarthome-Besitzern. Doch auch dieser wird mit der flächendeckenden Verfügbarkeit von Sprachassistenten schwinden. Als Zwischenlösungen sind auch Kombinationen aus Apps und Sprachsteuerung denkbar. Letztendlich wird der Nutzer jedoch immer den für ihn leichtesten Kommunikationsweg suchen. Aus der Physiologie des Menschen heraus ist dies die Kommunikation über Sprache. Weitere Systeme zur Kommunikation, wie zum Beispiel Gestensteuerung, werden sich im Smarthome ähnlich wie bei Smart-TVs vermutlich nicht durchsetzen können (Staub, 2018).

Eine Studie, in der ein Smarthome mittels eines Roboters gesteuert wurde, zeigt, dass die Interaktion mit Geräten deren Haptik und Informationsübertragung der unter Menschen üblichen ähnelt, von Nutzern als am angenehmsten bewertet wird. Der Roboter kommunizierte dabei mit menschenähnlichen Bewegungsmustern mit dem Nutzer und übermittelte ihm so Informationen. Die klassischen Kommunikationsformen im Smarthome, wie Apps oder Wanddisplays, wurden in dieser Studie als äußerst unkomfortabel identifiziert. (Luria, Hoffman, & Zuckerman, 2017)

Auch technische Lösungen, die über die Messung von Gehirnströmen Steuerungen durchführen, befinden sich in der Entwicklung. Die Erfassung und Verarbeitung dieser Ströme ist jedoch noch derart technisch aufwändig, dass in einem häuslichen Umfeld keine brauchbare, komfortable Interaktion möglich ist. Für die Unterstützung von Menschen mit körperlichen Einschränkungen könnte diese Methode aber im Bereich des „Ambient Assisted Living“ eine Möglichkeit sein, um einen barrierefreien Zugang für diese Menschen zu gewährleisten. (Samsung, 2018)

Durch die hohe Sensordichte ist jedoch eine Kommunikation vom Menschen zur Maschine auf viele weitere Arten möglich. Durch Sensorik kann das System Informationen über den Menschen und seine Umgebung aufnehmen und verarbeiten. Die damit erfolgte indirekte Kommunikation kann hier über viele mögliche Wege erfolgen, wie zum Beispiel Datenerfassung durch Sensorik, Eye-Tracking, Smart-Watches, Radartechnologie, Kameras, BLE-beacons, etc. (Google.ATAP, 2019; Liu, Yang, & Deng, 2018; Panella & Altilio, 2019; Sumra, 2018; Tobiidynavox, 2019; Zhang et al., 2019). Ein ganzheitliches System, in dem sich diese vielfältigen Möglichkeiten verbinden lassen, existiert jedoch bis zum jetzigen Zeitpunkt nicht.

3.2 Technische Grundlagen zur Erstellung eines intelligenten Smarthome

Um die in dieser Arbeit vorgestellte Methode evaluieren zu können, bedarf es eines Frameworks. Vor allem um sicherzustellen, dass aufgrund mangelnder Interoperabilität der verschiedenen Funktionsbausteine und der Geräte untereinander keine Einschränkungen entstehen. Die hier aufgeführten Punkte, die im Rahmen der Literaturrecherche erarbeitet wurden, stellen wesentlichen Aspekte für eine erfolgreiches Smarthome Framework dar.

Interoperabilität

Grundsätzlich sind bereits nahezu alle Funktionen in einem Gebäude automatisierbar. Um diese aber alle gleichzeitig und auch miteinander verknüpft verwenden zu können, bedarf es der Verwendung verschiedener Smarthome-Systeme. Die Einschränkung auf ein System würde automatisch eine Einschränkung im Funktionsumfang bedeuten. Dies hat den Grund, dass die Interoperabilität zwischen verschiedenen Herstellern bis jetzt nur in seltenen Fällen vorhanden ist. Um ein intelligentes Smarthome mit vollem Funktionsumfang zu erstellen muss daher auf verschiedene Funktionen von unterschiedlichen Produkten und Herstellern für Automationslösungen zurückgegriffen werden. Dies hat zur Folge, dass eine Software benötigt wird, die den hohen Grad an Interoperabilität bietet, den die einzelnen Hersteller nicht zur Verfügung stellen. (Leitner, 2018)

Flexibilität

Viele Aufgaben, vor allem Simulationsaufgaben, können nur von speziellen Softwarelösungen angeboten werden. Daher ist ein Framework erforderlich, dass es zulässt flexibel Schnittstellen von verschiedenen Anbietern einzubinden und anzusprechen, oder auch eigene Schnittstellen zu erstellen. Dies unterstützt den Wunsch der Nutzer eines Smarthomes, dieses flexibel nach ihren Wünschen zusammenzustellen (Deloitte Touche Tohmatsu Limited, 2018, S. 30).

Erweiterbarkeit

Besonders Funktionen und Sensorik aus dem Bereich der Bauphysik werden derzeit durch keine kommerzielle Automationshardware auf dem Markt unterstützt. Daher ist es erforderlich Hardwarekomponenten selbst anzufertigen und deren Firmware zu programmieren. Damit eine Kommunikation dieser Firmware mit dem Automationsserver erfolgen kann, muss dafür die Möglichkeit zur Integration verschiedener offener Protokolle im Automationsserver gegeben sein. So wird ein breites Feld an Kommunikationsmöglichkeiten mit selbst erstellter Sensorik und Aktuatorik eröffnet. Diese Anforderung deckt sich mit der von Strese et al. (2010, S. 10) definierten Nutzeranforderungen an ein zukünftiges Smarthome.

Ressourcenschonung

Viele der anzubindenden Interfaces für die Hardware werden derzeit über einen Universal Serial Bus (USB)-Anschluss direkt mit dem Server verbunden. Eine Platzierung einer steuernden Hardware direkt innerhalb des Wohnraums ist dadurch, vor allem durch die Verwendung von Funklösungen mit begrenzter Reichweite, unausweichlich. Daher muss die Grundsoftware des Automationsservers auf einer möglichst leisen und ressourcenschonenden Hardware lauffähig sein, um den Nutzer nicht in seinem gewohnten Wohnumfeld zu stören. Zusätzlich würde eine hohe Abwärme von leistungsstarken Computern auch dazu führen, dass Messergebnisse, vor allem im thermischen Bereich verfälscht werden. Weiterhin ist es eine Anforderung des Nutzers möglichst ressourcenschonend vor allem mit der benötigten Energie umzugehen (Strese et al., 2010, S. 10).

Support

Damit Probleme zeitnah behoben werden können, ist es essenziell, dass hinter der für das Framework verwendeten Software ein leistungsfähiger Konzern oder bei einem Open-Source-Projekt eine funktionsfähige Community steht und zeitnah Fehler beseitigt oder Änderungen an der Software durchgeführt werden können.

Verträglichkeit für ein Wohnumfeld
Wie bereits erwähnt muss ein Teil der steuernden Technik im Wohnumfeld platziert werden. Daher ist darauf zu achten, dass die Hardware möglichst wartungsarm und ohne störende Geräusche oder Beleuchtung betrieben werden kann. Andernfalls kann nicht von einem ungestörten Wohnumfeld ausgegangen werden.

Ökonomie
Da im Vorfeld nicht absehbar ist, welche Automationsaufgaben und Messaufgaben erbracht werden müssen, ist ein Framework zu wählen, bei dem möglichst kostengünstige Umbauten vorgenommen werden können. (Strese et al., 2010)

Bei Einhaltung der hier aufgeführten Randbedingungen kann demnach, nicht nur für das in dieser Arbeit benötigte Framework, sondern auch generell für Smarthomes davon ausgegangen werden, dass eine reibungsfreie Implementierung eines solchen Systems möglich ist.

3.2.1 Automationsserver

Bei einem Automationsserver muss in erster Linie zwischen Software und Hardware unterschieden werden. Die bereits genannten Felder Interoperabilität, Flexibilität und Erweiterbarkeit werden durch die Software eines Automationsservers beeinflusst. Die Bereiche Erweiterbarkeit, Ressourcenschonung und Verträglichkeit für das Wohnumfeld sind meistens von der Hardware abhängig. Für eine Bewertung der Bereiche Erweiterbarkeit, Ressourcenschonung und Kosteneffizienz ist jedoch eine ganzheitliche Betrachtung des Systems bestehend aus Hardware und Software unvermeidbar.

Software
Automationsserver können grundlegend in zwei Kategorien unterschieden werden:

- Kommerzielle Systeme
- Quelloffene Systeme

Kommerzielle Systeme sind meist darauf beschränkt Geräte zu steuern, die einem speziellen Hersteller zuzuordnen sind oder die eine spezielle Zertifizierung oder einen Standard aufweisen. Wie zum Beispiel die verfügbaren Softwaresysteme für die nach „KNX“ oder „EnOcean“-Standard zertifizierten Geräte. Grundsätzlich besteht bei den meisten kommerziellen Softwaresystemen die Möglichkeit neue Funktionen hinzuzufügen. Die Schnittstellen sind aber von den Softwareherstellern meist nicht freigegeben und können daher nicht von Nutzern bedient werden. Hinzu kommt, dass diese Systeme kaum Kompatibilität zu anderen Hardwarekomponenten anderer Hersteller aufweisen und so keine geeignete Interoperabilität gegeben ist. Selbst entwickelte Hardware kann hier beispielsweise kaum integriert werden.

Quelloffene Automationssysteme werden meist von einer Community programmiert und danach auch unterstützt. Sie haben durch die Quelloffenheit den Vorteil einer hohen Interoperabilität und zusätzlich die Möglichkeit sehr flexibel Daten über viele verschiedene Protokolle auszutauschen. Durch den Open Source Charakter ist es möglich im Falle eines nicht unterstützen Gerätes die Anbindung in Modulform selbst umzusetzen. Dies kann jedoch einen aufwändigen Prozess an Reverse Engineering der vom Hardwarehersteller verwendeten Protokolle erforderlich machen. Nachteil dieser Softwaretypen ist, dass Entwicklungen vor allem bei kleinen Communitys stagnieren können oder eingestellt werden, wenn die Zahl der Projektunterstützer nicht groß genug ist.

Eine eingehende Recherche ergab, dass derzeit ca. 16 große Communitys existieren, welche quelloffene Systeme zur Steuerung des Wohnraums unterstützen. Eine genaue Analyse des Angebotes erfolgt in Kapitel 5.4.1, die dazu erhobenen Daten können dem Anhang entnommen werden. Eine Ausnahme im Open-Source Bereich stellt die Software OGEMA 2.0 der Fraunhofer-Gesellschaft dar. Sie ist zwar quelloffen, wird aber nicht von einer Community entwickelt und unterstützt, sondern von einem Institut für angewandte Forschung (Fraunhofer-IIS, 2019). Dieses System hebt sich von anderen Lösungen grundsätzlich durch die starke Modularisierung ab. Die Struktur dieses Automationsservers ist in nachfolgender Grafik abgebildet.

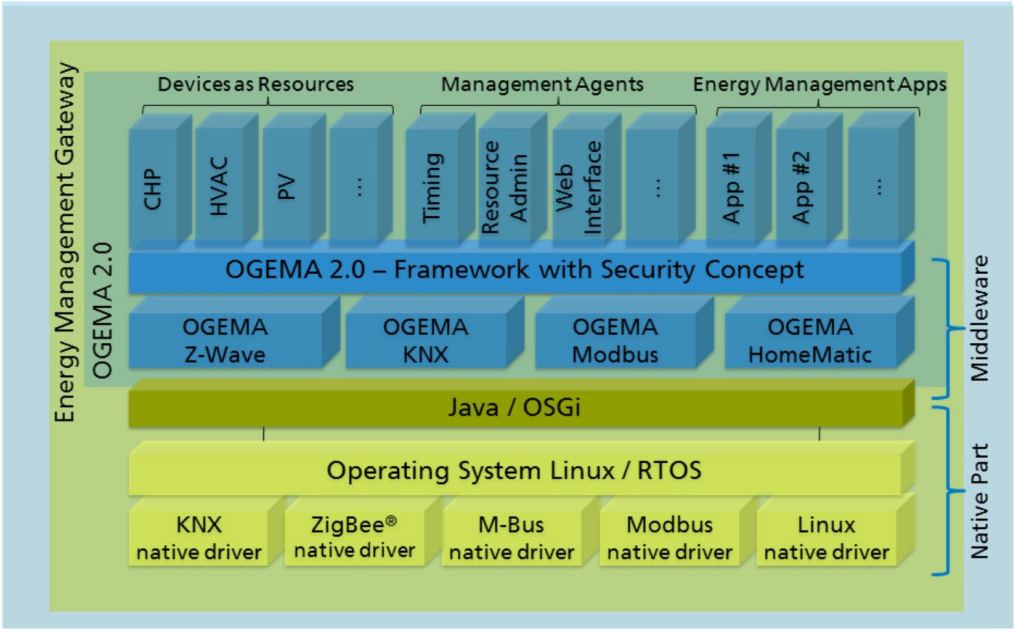


Abb. 6 Modularitätsstruktur von OGEMA 2.0 (Fraunhofer-IIS, 2019, S. 2)

Hierbei wird versucht, durch das OGEMA Framework eine modulare Struktur, für die jeweiligen entwickelten Applikationen hardware-universell verfügbar zu machen (Fraunhofer-IIS, 2019). Mittels der beiden quelloffenen Smarthome-Server FHEM und OpenHab wird versuchen eine ähnliche Funktionalität zu gewährleisten (FHEM e.V., 2019; openHAB Foundation e.V., 2019). Grundsätzlich gibt es aber bei jedem dieser Systeme Restriktionen, die eine vollständige Interoperabilität verhindern.

Hardware

Als Hardware ist für Automationsserver sogar ein Einplatinencomputer, wie der Raspberry Pi, ausreichend. Eigene Tests zeigten jedoch, dass bei steigender Anzahl an zu verarbeitenden Geräten die Reaktionsgeschwindigkeit des gesamten Systems stark sinkt. Daher sind bei größeren Smarthome-Installationen eigene Server oder NAS-Systeme (Network-Attached-Storages) empfehlenswert. Damit erhält man genügende Leistungsreserven, um auch komplexere Berechnungen durchzuführen. Für Simulationsaufgaben reichen die Hardwarespezifikationen von NAS-Systemen jedoch meist nicht aus.

Aus dem Gebäudeautomationsbereich können weiterhin verschiedene Ebenen für eine bessere Verständlichkeit übernommen werden.

Grundsätzlich wird unterschieden in:

- Feldebene
- Automationsebene
- Managementebene

(Wisser, 2018, S. 10)

Eine Integration mehrerer Ebenen in eine Hardware ist grundsätzlich technisch möglich. Dies erzeugt jedoch die Problematik, dass bei Hardwaredefekten große Teile des Systems ausgetauscht werden müssen oder ausfallen. Daher ist eine Trennung der oben genannten Ebenen mit jeweils eigener Hardware als Stand des Wissens anzusehen.

Als Verbindungskomponente zu steuerbaren Aktuatoren oder um Sensoren auszulesen, ist als Bindeglied ein Interface als abschließende Komponente notwendig. Diese Komponenten basieren auf unterschiedlichen Kommunikationsprotokollen. Folgend ist eine Auflistung der derzeit üblichen Protokolle aufgeführt.

Tabelle 2 Übersicht über gängige Übertragungsprotokolle im Smarthome

Protokoll	**Anbindung über**
Bid-Cos	Funk
KNX	Kabel oder Funk
EnOcean	Funk
Bluetooth und BLE	Funk
Loxxon	Kabel
Powerline	Kabel
Multimediageräte	Infrarot

Visualisierungen im Smarthome finden meist über die mit proprietären Systemen mitgelieferten Frontends oder Apps statt. Teilweise erfolgt die Anzeige auch über deren mitgelieferte Hardware (Gira, 2019). Im Bereich der quelloffenen Server finden Anzeigen meist über die mit den Automationsservern mitgelieferten Weboberflächen statt. Diese sind jedoch in Optik und Nutzerfreundlichkeit meist nicht besonders weit fortgeschritten oder auf bestimmten Anzeigegeräten wegen technischer Restriktionen nicht nutzbar. Daher werden einige spezielle Software Systeme entwickelt, die nur dafür konzipiert sind, ansprechende und flexible Visualisierungen im Smarthome-Bereich zu erstellen. Ein Beispiel für so ein System ist die Opensource-Lösung „SmartVisu“ (Martin, 2016).

Viele weitere Dienste im Smarthome sind ebenfalls vor allem bei quelloffenen Systemen über Zusatzdienste oder Zusatzserver implementierbar. In einigen Fällen werden diese sogar über eigene Communitys entwickelt. So ist eine Anbindung des von Apple entwickelten Smarthome-Standards „Homekit“ beispielsweise über eine Spezialsoftware mit dem Namen „Homebridge“ möglich und eine Implementierung dieser Struktur in einen quelloffenen Smarthome-Server kann damit erfolgen (Homebridge Community, 2019). Im weitesten Sinne sind aus diesen Gründen quelloffene Smarthome-Server der beste Weg das vorherrschende Interoperabilitätsproblem zwischen den verschiedenen Smarthome-Herstellern zu lösen.
Im Bereich der Aktuatorik und Sensorik können bereits viele einfache Mess- oder Steueraufgaben erfüllt werden. Besonders aus dem Bereich der Bauphysik fehlen jedoch Möglichkeiten wichtige Messgrößen zu erfassen. Darunter zählt zum Beispiel die Bestimmung von

Oberflächentemperaturen, Wärmeflüssen, Globaltemperaturen und Luftgeschwindigkeiten. Eine Übersicht der aktuell möglichen Messaufgaben, die mit herkömmlichen Smarthome-Systemen umsetzbar sind, ist folgend abgebildet. Es gilt jedoch zu beachten, dass sich der Markt für Smarthome-Lösungen schnell entwickelt. Daher liefert Tabelle 3 nur eine Übersicht zum jetzigen Zeitpunkt.

Tabelle 3 Auswahl an mögliche Messgrößen im Smarthome

Messgröße
Lufttemperatur
Relative Luftfeuchte
Wasseraustritt
CO_2
CO
Gas
Rauch
Hitze
Tankinhalt und Füllstand
Tür und Fensterstatus
Leckagen
Elektrische Impulse
Bewegung
Neigung
Niederschlagsmenge
Wetterdaten
Elektrische Größen (Spannung, Strom, Frequenz)
Beleuchtungsstärke
Durchgang
Vibration

3.2.2 Modularisierung in der Informationstechnik

Vorreiter in der Modularisierung war die Automobilindustrie. Die in diesem Bereich verzeichneten Erfolge dieser Technik führten zu einer Modularisierung in vielen Bereichen unseres täglichen privaten und beruflichen Lebens. (Fildebrandt, 2012)
In der Informationstechnik gestaltet sich Modularisierung durch die Verkapselung von Quellcode in Blöcke mit definierten Schnittstellen. Die so entstehenden Module sind folglich unabhängig voneinander nutzbar und es wird kein tiefgehendes Wissen über deren interne Funktionalität benötigt. Solange die Schnittstellenbedingungen eingehalten werden, können die Module genutzt werden.(Pomberger & Blaschek, 1996; Spraul, 2012).

Ziele der Modularisierung sind hauptsächlich:

- Reduzierung der Komplexität
- Effizienzsteigerung
- Flexibilitätssteigerung
- Höherer Individualisierungsgrad bei kürzeren Produktentwicklungszeiten
- Geringerer Produktentwicklungsaufwand.

(Wildemann, 2014)

Eine hierfür häufig verwendete Technik ist die objektorientierte Programmierung. Bei dieser werden Prozeduren in Objektdateien gespeichert und in Form von Bibliotheken zusammengeführt. Objekte können zudem Eigenschaften und Methoden an nachfolgende Objekte vererben, die von Vorgängerobjekten abgeleitet wurden. Ein Modul, welches in diesem Fall eine Sammlung von Funktionen und Datenstrukturen ist, wird folgend durch einen Kompilierungsvorgang in eine für Menschen nicht mehr lesbare Datei überführt. Dieser Prozess nennt sich Verkapselung. Ohne eine genaue Definition der vom Modul verwendeten Schnittstellen ist dieses hierauf jedoch nicht mehr verwendbar. (Nahrstedt, 2009)
Die folgende Abbildung visualisiert den grundsätzlichen Aufbau eines Moduls und seine Trennung in eine Import- und Exportschicht sowie den Funktionsblock:

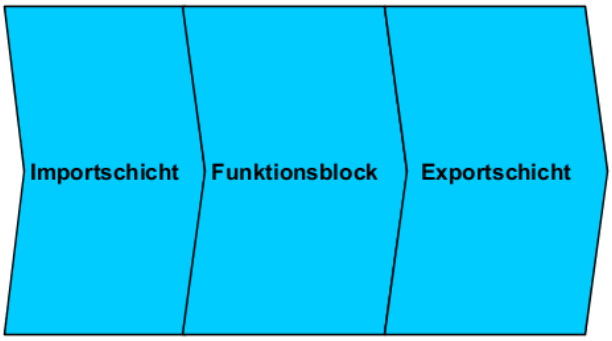


Abb. 7 Modulschichten

Eine vor allem für den Bereich der Bauphysik interessante Modularisierungstechnik stellt hierbei das „Functional Mock-up Interface“ (FMI) dar. Es bietet einen von der Programmierumgebung unabhängigen Standard. Der FMI Standard spezifiziert die notwendigen Schnittstellen und Umgebungen um zwei Simulationsmodelle voneinander unabhängig berechnen zu können. Zusätzlich bietet dieses Framework die Möglichkeit der sogenannten Co-Simulation. Dabei ist es möglich mehrere Simulationsmodelle miteinander zu koppeln und Daten zwischen den einzelnen Berechnungsschritten auszutauschen. Der Vorgang wird von einem „Masteralgorithmus“ gesteuert, der den Datenaustausch sicherstellt und die Steuerung der Simulationsschritte übernimmt. Die nach dem FMI Standard verkapselten Simulationsmodelle nennen sich Functional Mock-up Unit (FMU). Durch den Standard ist es möglich Probleme unabhängig der softwarebedingten Solver zu lösen und in Co-Simulation zu betrachten, da sofern notwendig Solver mit in das FMU integriert werden. (Nouidui, Wetter, & Zuo, 2013) Um diese Co-Simulationen bereitstellen und verschiedene Eingangsparameter individueller Natur erfassen zu können, müssen unterschiedliche Algorithmen und Systemkonzepte kombiniert werden.

3.3 Algorithmus und Systemkonzepte für die Verwendung in Smarthomes

Um Individualität bieten zu können, die im Wohnkontext erforderlich ist und die Interaktionen abzubilden, die im Bereich der Bauphysik berücksichtigt werden müssen, ist die Verwendung verschiedener Techniken erforderlich. Die Grundlagen, der im Rahmen dieser Arbeit verwendeten Konzepte und Systeme, sollen daher im Folgenden erläutert werden.

3.3.1 Aktivitätserkennung von Nutzern

Der Bereich der Aktivitätserkennung und der richtigen Ortung des Nutzers im Wohnraum ist ein vergleichsweise neuer Bereich in der Forschung. Größtenteils initiiert wurde die Thematik durch den demografischen Wandel unserer Gesellschaft. „Ambient Assisted Living“ ist hier ein Schlüsselbegriff. Die Thematik „altersgerechte Assistenzsystem für ein unabhängiges Leben“ (UKV - Union Krankenversicherung, 2018) ist einer der vielversprechendsten Faktoren zur Lösung des Problems der nicht mehr ausreichenden Pflegekräfte (Rafferty et al., 2018). Durch altersgerechte Assistenzsysteme ist es daher auch mit Hilfe der Aktivitätserkennung möglich kritische Situationen zu erkennen und so Hilfspersonal gezielt einzusetzen, wenn es benötigt wird. Es dient somit größtenteils zur Sicherheit der Bewohner. Ein weiterer Aspekt, in dem Sicherheit eine essenzielle Rolle spielt, ist die Sicherheit des Wohnraums und der Absicherung vor unbefugtem Zutritt. Durch das Erkennen nicht gewünschter oder ungewöhnlicher Aktivitäten kann so auch ohne Anwesenheit des Menschen die Sicherheit des Wohnumfelds gewährleistet werden.
Im Bereich der Bauphysik ergibt sich durch die Aktivitätserkennung unter anderem die Möglichkeit auch ineffizientes oder ungewöhnliches Nutzerverhalten zu identifizieren. Dadurch kann übermäßiger Energieverbrauch oder drohende Gefahren, wie zum Beispiel schimmelbegünstigendes Verhalten, identifiziert und Gegenmaßnahmen initiiert werden, bevor ein Schaden entsteht. (Fauzi, Sulistyo, & Widyawan, 2018)
Somit können durch gezielte Erkennung menschlicher Aktivitäten, Steuerungen mit dem Ziel der Steigerung der Sicherheit, Energieeffizienz, Gesundheit und des Wohnkomfort generiert werden.
Die Aktivitätserkennung im Sinne dieser Arbeit basiert auf der Sammlung verschiedenster Rohdaten und deren intelligenter Auswertung. Intelligent bezieht sich dabei in den meisten Fällen auf das Auswerten mit Hilfe von Methoden des maschinellen Lernens, welche im weiteren Verlauf noch näher beschrieben werden. Das Feld der Aktivitätserkennung gewinnt in den letzten Jahren nicht nur im Bereich des Smarthome immer mehr an Relevanz. Während Spracherkennung inzwischen auf einem sehr ausgereiften Niveau vorhanden ist und von den Menschen regelmäßig in Form von Sprachassistenten genutzt wird, ist die Aktivitätserkennung vor allem im häuslichen Bereich noch nicht über die Phase einzelner Prototypen hinausgekommen. Dies hat hauptsächlich den Hintergrund, dass sich die Beschaffung von Daten, welche zur Untersuchung dieser Phänomene erforderlich sind, schwierig gestaltet. Derzeit berufen sich Untersuchungen, wie Beispielsweise die des folgend erläuterten UCAmI-Cups, zur Aktivitätserkennung im Wohnumfeld ausschließlich auf Daten aus, dafür angelegten, Testwohnungen, die zur Untersuchung dieser Problemstellung eingerichtet wurden. Ein weiteres Problem, das bis heute kaum Beachtung fand, stellt die Kategorisierung von Aktivitäten dar. Diese ist vor allem für die Verwendung von Algorithmen des maschinellen Lernens von essenzieller Bedeutung.

Grundlegend kann für die Aktivitätserkennung von folgenden vier Grundaufgaben, die zu erfüllen sind, ausgegangen werden:

- Auswahl und Einsetzen geeigneter Sensoren
- Erfassen, Speichern und Weiterverarbeitung der gewonnenen Daten
- Softwaregestütztes Generieren von Aktivitätsmodellen
- Auswahl oder Entwicklung von Algorithmen zur Erkennung der Aktivitäten

(Chen, Hoey, Nugent, Cook, & Zhiwen, 2012)

Die Sensorik, die zur Erkennung von Nutzeraktionen verwendet werden kann, ist grundsätzlich in drei Bereiche untergliedert.

- Audiovisuelle - (Beispielsweise eine Kamera)
- Physische (tragbare) - (Beispielsweise eine Smartwatch)
- Physische (objektbezogene) - (Beispielsweise herkömmliche Sensorik)

(Fauzi et al., 2018, S. 3)

Da die Erkennung von Nutzeraktionen mit Hilfe von Sensordaten immer mehr an Relevanz gewinnt, wurde im Rahmen der Konferenz UCAmI 2018 zu diesem Zweck ein Wettbewerb ausgeschrieben, dessen Ergebnisse gut den aktuellen Stand der Forschung widerspiegeln. In diesem wurde die beste Vorhersage der Nutzeraktionen mit einem vorgegebenen Datensatz ausgezeichnet. Als Datenquellen standen hierfür Sensoren und Nutzerpositionen verschiedenster Art zur Verfügung. Die dafür erforderlichen Daten wurden im University of Jaen's Ambient intelligence (UJAmI) SmartLab generiert. Zur Verfügung standen Datensätze mit 24 verschiedenen häuslichen Aktivitäten von Nutzern sowie deren Ausführungszeit und verschiedenen Messdaten von ca. 130 „SmartDevices" und Sensoren. (Salomón & Tîrnăucă, 2018) Ergebnis dieses Wettbewerbs war, dass sich für die Ortung und Bestimmung der Aktivität des Nutzers verschiedene Ansätze als brauchbar, erwiesen. Diese sind:

- Die „Aktivitätserkennung mittels endlicher Automaten"
- „Domain knowledge-based Activity recognition"
- Verwendung von Algorithmen maschinellen Lernens

Die im Rahmen dieser Dissertation verwendete Testwohnung wurde bereits im Vorfeld mit den meisten Sensoren, die auch für den Wettbewerb relevant waren, ausgestattet. Bei der Auswertung der Daten zeigten sich die gleichen Probleme, die auch bei Veröffentlichungen im Rahmen des „UCAmI-Cup" in diesem Bereich mehrfach angebracht wurden. Diese können wie folgt zusammengefasst werden:

1. Fehler in Messdaten müssen bereinigt werden.
   Beispiel: Ein vollständiges Durchqueren der ganzen Wohnung in unter einer Sekunde ist nicht möglich.
2. Daten haben unterschiedliche Formate (Binary, Location, Audio, etc.)
3. Es gibt Aktionen, die unterschiedlich für den Nutzer sind, aber ähnliche Datenmuster durch die sensorische Erfassung erzeugen.
4. Die Genauigkeit der Vorhersage von Aktionen und der Erkennung hängt stark von der Menge an Daten ab.
5. Algorithmen benötigen meistens Trainingsdaten, die gegebenenfalls schwer zu erhalten sind.
6. Die Ansätze sind nicht notwendigerweise modular auf andere Wohnräume übertragbar, sondern spezialisiert auf ein Testszenario.

(Neldel, 2018)

Weiterhin existieren bei den bekannten Algorithmen zur Bestimmung der Aktivität noch systematische Fehler. So werden teilweise Aktivitäten erst vergleichsweise spät erkannt. Auf Grund der Menge an Fehlerquellen und dem noch sehr geringen Wissensstand auf diesem Gebiet, sind die Genauigkeiten der Vorhersagen bis jetzt noch in keinem für ein Produktivsystem verwertbaren Bereich vorhanden. Einzelne Algorithmen erzeugten hierbei zeitweise Genauigkeiten im Bereich von ca. 95 % und einer Gesamtgenauigkeit von ca. 91%. Diese wurden jedoch mit speziell, für den im Wettbewerb verwendete Datensatz, entwickelten Modellen, wie Beispielsweise die Aktivitätserkennung mittels endlicher Automaten erstellt („Automata“). (Salomón & Tîrnăucă, 2018)
Ansätze wie die „Domain knowledge-based Activity recognition“ setzen auf die Erkennung mittels einfacher „Wenn-Dann-Regeln“ und wurden speziell auf den zur Verfügung stehende Datensatz zugeschnitten. Es zeigte sich, dass eine Verwendung aller zur Verfügung stehenden Daten des „UCAmI-Cup“ bei dieser Methode nicht brauchbar ist und die Autoren beschrieben ihren Ansatz auf Grund der starken Modellvereinfachung als verbesserungsbedürftig. Trotzdem konnten damit auf diesen speziellen Testfall hin Genauigkeiten von ca. 81% erzielt werden. (Karvonen & Kleyko, 2018)
Zusammenfassend kann man über die Ergebnisse des „UCAmI-Cup“ sagen, dass mit herkömmlichen „Machine-Learning-Algorithmen“ durchgehend schlechtere Vorhersagen mit einer Genauigkeit von nur etwa 60% erzielt werden. Diese schlechten Prognoseergebnisse lassen sich auf einen zu kleinen Datensatz zurückführen. Markant ist hier, dass jedoch in Bezug auf den Trainingsdatensatz eine Genauigkeit von ca. 92% erzielt werden kann. Dies lässt auf ein bekanntes Problem bei Algorithmen des maschinellen Lernens hindeuten. Dem sogenannten „Overfitting“. Dies wird erzeugt, wenn dem Algorithmus nicht Merkmale, sondern die Daten selbst angelernt werden. Übertragen könnte man sagen der Algorithmus hat die Daten „auswendig gelernt“. Dieser Effekt tritt auch im Zusammenhang mit zu kleinen Datenmengen auf und ist ein bekanntes Problem im Bereich des maschinellen Lernens. Auch wurde von den Cup-Teilnehmern, die Inkonsistenz und die Art der Daten als Ursache für diese Probleme identifiziert. (Cerón, López, & Eskofier, 2018)
Den direkten Einfluss der Datenmenge und der Datenqualität auf die Genauigkeit der Aktivitätserfassung zeigten auch Mehr, Polat, und Cetin (2016). Deren Tests wurden in einem ähnlichen Testumfeld durchgeführt, welches vom Massachusetts Institute of Technology (MIT) Labors und dem TIAX (Entwicklungsunternehmen) zur Verfügung gestellt wurde. Sie konnten einen direkten Zusammenhang der Häufigkeit der durchgeführten Aktionen und der Genauigkeit der Erkennungsrate feststellen. Unterstützend konnte festgestellt werden, dass durch die Kombination der richtigen Algorithmen auch ein starker Einfluss auf die Genauigkeit der Aktivitätserkennung genommen werden kann und so auch mittels maschinellen Lernens in speziellen Fällen Genauigkeiten von bis zu 97% möglich sind. (Zou et al., 2018)

Grundsätzlich kann für den Bereich Smarthome festgehalten werden, dass zur weiteren Erforschung der Erkennung und Ortung von Nutzeraktionen folgende Probleme bestehen:

- Es existiert bisher kein Datensatz mit ausreichender Größe für Untersuchungen
- Für bestimmte Aktivitäten liegen grundsätzlich zu wenig Daten vor, da die Aktivität von Menschen zu selten ausgeführt wird, um sie mit den derzeitigen Algorithmen sicher zu erfassen

Im Rahmen dieser Arbeit wird die Thematik der Aktivitätserkennung in zwei Bereiche unterteilt, die im Folgenden getrennt behandelt werden. Dadurch wird den Ergebnissen der „UCAMI 2018“ Rechnung getragen, die belegen, dass durch einen Ansatz, der das Wissen über den Kontext des Nutzers verwendet - in diesem Fall durch die Bestimmung seiner Position im Raum - eine effizientere Aktivitätserkennung stattfinden kann. Zumal Daten zur Bewegung des Nutzers explizit als förderlich für eine höhere Genauigkeit durch die „UCAmI-Cup“ Teilnehmer beschrieben wurden (Karvonen & Kleyko, 2018).

Ortung der Nutzerposition

Die Bestimmung der genauen Position des Nutzers im Wohnumfeld, kann derzeit auf mehrere Arten und durch unterschiedliche Sensorik erfolgen. Eine Untergliederung dieser ist ebenso wie bei der Aktivitätserkennung in die von Fauzi et al. (2018, S. 3) erstellten Kategorien möglich.

Allen Systemen gemeinsam ist aber die Tatsache, dass es sich um einmalige Testaufbauten in meist stark vereinfachten Testumfeldern handelt. Im Bereich der physischen (objektbezogenen) Sensorik zur Nutzerortung existiert derzeit ein Hersteller, der seit kurzer Zeit ein System vertreibt mit dessen Fußbodenleisten eine Ortung der Person im Raum möglich ist. Dafür muss aber in einem Bestandsgebäude, das aufwändige Tauschen aller Fußbodenleisten in Kauf genommen werden. Auch muss eine direkte Sichtverbindung zwischen Nutzer und Fußbodenleiste bestehen. (nevisQ GmbH, 2018)

Ein weiterer vielversprechender Ansatz, der bereits in der Praxis Anwendung findet, ist die Ortung mittels BLE Beacons. Dieser stellt eine Mischung aus dem „Physisch(tragbaren)“ und dem „Physisch(objektbezogenen)“ Ansatz dar. Dabei wird die Position des Nutzer-Smartphones geortet in dem die Signalstärken der Verbindungen zu mehreren sogenannten BLE Beacons ausgewertet werden und eine Art Triangulation stattfindet. Wenngleich diese Systeme für den kommerziellen Einsatz in Kaufhäusern mit geringen Anforderungen an die Genauigkeit erstellt wurden, sind die Genauigkeiten auf einem weitaus kleineren Raum, wie dem Wohnraum, noch in einem verwendbaren Bereich für eine Ortung der Nutzerposition. (Liu et al., 2018)

Nachteilig ist, dass das Smartphone für die Ortung dauerhaft vom Nutzer getragen werden muss, was für den Wohnraum nicht immer eine praktikable Lösung darstellt.

Erkennung der Aktivität

Die Aktivität kann nach entsprechender Einschränkung durch die Nutzerposition somit entsprechend der Methode nach Salomón und Tîrnăucă (2018) bestimmt werden. Alternativ kann diese bei ausreichender Datenbasis mittels maschinellen Lernens voraussichtlich noch effizienter, bestimmt werden. Für dies Methoden ist jedoch eine sinnvolle Merkmal- und Dateneingrenzung notwendig.

Zusammenfassend kann gesagt werden, dass im Bereich der Erkennung von Aktivitäten noch enormer Forschungsbedarf besteht. Die Entwicklung der Aktivitätserkennung steht hier noch in den Anfängen.

3.3.2 Kontextsensitive Steuerung

Kontextsensitive Steuerung im Sinne des Smarthomes beschreibt nicht das rechtzeitige Reagieren auf Randbedingungen durch vorgefertigte Lösungsstrategien. Im Sinne eines intelligenten Smarthome beschreibt sie viel mehr die Wahrnehmung dieser Randbedingungen und des aktuellen Wohnkontext des Nutzers. Die sich daraus ergebenden notwendigen Steuerungen sollen den Nutzer unterstützen. Dabei ist auf dessen individuelle Präferenzen und Bedürfnisse zu achten, weshalb vorgefertigte Steuerstrategien selten zum gewünschten Ziel führen. Auf Grund der Individualität müssen eben diese Lösungen auch individuell gestaltet sein und Annahmen über die vom Nutzer präferierte Variante treffen. (Robles & Kim, 2010)

Kim und Lee (2014) beschreiben den Kontext als jede Information, die dazu dient eine Einheit (Person, Objekt, Ort) genauer zu beschreiben.

„Being context aware is the ability of an entity to be aware of the current circumstances and be aware of relevant information that may influence any decision and behavioural change based on the context.“ (Kim & Lee, 2014, S. 1)

Ein kontextsensitives System muss daher in der Lage sein, alle zur Problembeschreibung notwendigen Parameter, zu erfassen und auszuwerten und darauf basierend intelligente Steuerungen als Reaktion zu entwickeln. Dabei spielt auch die Berücksichtigung von Benutzerpräferenzen und der kontextbezogenen Situation eine große Rolle. (Ojagh, Malek, Saeedi, & Liang, 2018)

Ohne Kontextsensitivität können Smarthomes demnach keine Funktionen bieten, die dem individuellen Nutzerverhalten angepasst sind. (Cheng, Chen, & Tseng, 2006) Deshalb ist vor allem die bereits erläuterte Aktivitätserkennung ein essenzieller Bestandteil für Kontextsensitivität. Wenngleich einfache Ansätze bereits existieren und umgesetzt wurden, existieren bis jetzt keine effizienten Techniken, die dynamisch verändernde Situationen erkennen und dementsprechend darauf reagieren können. (Jun, Kim, & Kim, 2018)
Die bereits erfolgte Unterteilung der Aktivitätserkennung in einen Standort und eine Aktivität wird, durch die Dimensionen des Kontextes, welche von Ma, Kim, Ma, Tang, und Zhou (2005) definiert wurden, unterstützt. Diese sind folgend aufgeführt:

Tabelle 4 Kategorien des Kontextes (Ma et al., 2005, S. 2, Tabelle 1)

Kategorie	**Instanz**	**Beispiel**
Zeit	- Zeit - Zeitsequenz - Standort	Sekunde/Minute Ereignis durch auftretende Sequenz Schlafzimmer/Wohnzimmer
Umgebung	- Status - Messwert - ID	Ankommend/gehend Temperatur Personen ID
Person	- Nutzerprofil - Angewohnheiten - Individuelle Eigenschaften	Name/Alter/Geschlecht Sport/Temperierung Krankheiten

Aufgrund der oben genannten Probleme bei der Aktivitätserkennung und der bis heute nicht erforschten Zusammenhänge im Fall von mehreren Nutzern im Wohnumfeld, ist der Stand der Forschung bei kontextsensitiven Systemen ähnlich wenig fortgeschritten wie bei der Aktivitätserkennung. Die Systeme sind zurzeit noch stark technikzentriert und wenig nutzerzentriert gestaltet. (Ospan, Khan, Augusto, Quinde, & Nurgaliyev, 2018)
Der Ansatz, der einem kontextsensitiven Prinzip am nächsten kommt, ist das sogenannte „MavHome“ (Cook et al., 2003). Es verfolgt einen auf Agenten basierenden Ansatz. In diesem wird versucht die Bedürfnisse des Nutzers durch „Hineinversetzen“ zu verstehen und durch logisches agieren Lösungsansätze selbst zu erzeugen. Es ist gekennzeichnet durch eine starke Modularisierung und verfolgt einen Bottom-Up Ansatz. (Cook et al., 2003)
Dabei werden erst kleinteilig Problemlösungen gefunden und danach zu Handlungsstrategien aggregiert (Siepermann, 2018). Dieser Ansatz eines intelligenten Vertreters war durch seine hohe Modularisierung und seine nutzernahe Konzipierung prägend für die hier vorgestellte Arbeit. Das Prinzip der nutzernahen Konzipierung spiegelt sich in der hier vorgestellten Methode in Form der Interkation des DGZ mit dem DNZ wieder. Es legt jedoch einen starken Fokus auf die Nutzeraktionen und deren Vorhersage sowie Verarbeitung und weniger auf die Zusammenhänge der umzusetzenden Funktionen. Auf diesen grundlegenden Ansätzen aufbauend, konnte somit die essenzielle Lücke für die Beschreibung bauphysikalischer Probleme mit der hier vorgestellten Methode geschlossen werden. Auch die bereits angesprochenen Interoperabilitätsprobleme können somit gelöst werden.

Ein weiterer weit in der Entwicklung von kontextsensitiven Systemen vorgeschrittener Ansatz ist der von Mahroo, Spoladore, Caldarola, Modoni, und Sacco (2018). Hier wird eine Informationsanreicherung und Interaktion einzelner Geräte im Smarthome mittels „Semantic Web Technologies“ erzeugt. Dies bietet die Möglichkeit die vom Nutzer präferierten Einstellungen des Gebäudes regelmäßig durch das Erkennen von Zusammenhängen zu erweitern. Damit ist eine dynamische Anpassung auch an veränderte Wohnkontexte möglich. Auch dieser Ansatz war prägend für die Entwicklung der hier beschriebenen Methode. Die Möglichkeit mittels vorhandener Technologie eine starke Individualisierung auf den jeweils nutzerspezifischen Kontext zu erzeugen wurde hierdurch bewiesen. (Mahroo et al., 2018) Die Arbeit von Mahroo et al. (2018) belegt das Kontextsensitivität ein Schlüsselfaktor für ein intelligentes Smarthome ist.
Ein weiterer Aspekt, dem Rechnung getragen werden muss, ist der Detailgrad der Kontextsensitivität. So beschreibt Jun et al. (2018) eine Möglichkeit, je nach Informationsgehalt unterschiedliche oder erweiterte Steuerungen unter Berücksichtigung der Verarbeitungsgeschwindigkeit zu generieren. Unterschieden wird hier in zwei Kategorien, einer Methode mit niedrigem Informationsgehalt über den Kontext und der Notwendigkeit einer schnellen Entscheidung. Außerdem sind Situationen mit hohem Informationsgehalt und genauen Entscheidung möglich. Je nach Systemleistung bietet dieses System somit eine schnelle, aber eher generelle Reaktion auf einen Kontext. Nachfolgend kann diese durch mehr Informationen verfeinert werden. Jun et al. (2018) wandte diese Methodik am Beispiel der Aktionsgenerierung bei Stürzen älterer Menschen an. Dabei entspricht das Alarmieren von Hilfspersonal einer schnellen Entscheidung und das Nachalarmieren eines Arztes entsprechend der Krankenhistorie des Menschen einer genauen Entscheidung. Das Ergebnis der Arbeit war die Notwendigkeit, auf Situationen schnell reagieren zu können und das unabhängig von den zur Verfügung stehenden Ressourcen. (Jun et al., 2018)
Kontextsensitivität durch „Bestärkendes Lernen“ mittels „Deep Learning Algorithmen“ ist zusätzlich ein vielversprechender Ansatz, um einem System zu ermöglichen die individuellen Randbedingungen zu erlernen. Hierbei wird ein Algorithmus ähnlich einem Belohnungssystem darauf trainiert, sich selbst zu verbessern. Auch wenn erste Ergebnisse keine hohen Genauigkeiten aufzeigen so scheint dieser Ansatz, vor allem in Bezug auf die Nutzerindividualisierungen vielversprechend, die noch nicht ausreichend erforscht sind oder durch Sensorik schwer erfassbare Phänomen enthalten. (Brenon, Portet, & Vacher, 2018)

Grundsätzlich zeichnen sich für die kontextsensitive Steuerung daher folgende Probleme ab:

- Geringe Verfügbarkeit von Daten
- Daten sind fehlerbehaftet
- Ansätze zur Kontextsensitivität sind stark spezialisiert und nicht übergreifend
- Notwendige Rechenleistung für diese Konzepte ist sehr hoch
- Keine Standardisierung der Datensätze vorhanden
- Hoher Zeit und Ressourcenaufwand zur Erhebung der Daten

(Neldel, 2018)

3.3.3 Systeme künstlicher Intelligenz

Künstliche Intelligenz (KI) ist ein „Teilgebiet der Informatik, welches versucht, menschliche Vorgehensweisen der Problemlösung auf Computern nachzubilden, um auf diesem Wege neue oder effizientere Aufgabenlösungen zu erreichen.“ (Lämmel & Cleve, 2011, S. 13)

Ein KI gestütztes System ist daher gegeben, wenn eine der folgenden Komponenten enthalten ist:

- Sprachverarbeitung
- Darstellung von Wissen
- Automatisches Schlussfolgern
- Adaptionsfähigkeit
- Bildanalyse
- maschinelles Lernen

(Lenzen, 2018, S. 23)

Im Folgenden werden die zur Entwicklung in dieser Arbeit benötigten Grundlagen und Konzepte, welche dem Bereich der KI zugeordnet werden können, näher erläutert.

Ontologie gestützte Algorithmik

Ontologien stellen eine Möglichkeit dar, Wissen vor allem über Zusammenhänge einfacher verwertbar, in Form von Graphen zu beschreiben und so dieses Wissen für Maschinen lesbar zu machen. Zu diesem Zweck werden Zusammenhänge mit Hilfe von Klassen, Eigenschaften und Individuen beschrieben. Klassen können Unterklassen enthalten, um so auch komplexe Zusammenhänge und Abgrenzungen zu erzeugen. Weiterhin können Klassen verknüpft werden, dies geschieht zum Beispiel durch die Zuweisung von Eigenschaften. So können auch komplexe Zusammenhänge, die für den Menschen nicht mehr überschaubar sind, abgebildet und bestimmt werden.(Stuckenschmidt, 2011)
Grundsätzlich wird hierbei unterschieden in eine „T-Box“ und eine „A-Box“. Die T-Box enthält alle Klassen und deren Beziehungen, sowie deren Eigenschaften und Datentypen. Sie beschreibt ausschließlich das allgemeine Grundkonzept, welches mit der Ontologie abgebildet wird und macht keine Aussage über einen speziellen individuellen Zusammenhang.(Motik, Grau, & Sattler, 2008)
Eine beispielhafte Gliederung eines Familienstammbaumes in Klassen und Subklassen wäre:

- Familienmitglieder
- Eltern
    - Vater
    - Mutter
- Kinder
    - Sohn
    - Tochter

Die „A-Box“ dagegen enthält alle Individuen mit Information über deren Klassen und Eigenschaften aus der T-Box (Motik et al., 2008). Die A-Box repräsentiert somit die Anwendungsseite für ein spezielles Beispiel.
Am eben genannten Beispiel wären dies folgende Informationen:

Tom gehört zur Klasse Eltern mit der Subklasse Vater.
Lisa gehört zur Klasse Eltern mit der Subklasse Mutter.
Ben gehört zu Klasse Kinder mit der Subklasse Sohn.

Durch die Trennung in T-Box und A-Box ist es möglich, Ontologien unabhängig ihrer speziellen Anwendung immer wiederzuverwenden, da die Inhalte der T-Box so verallgemeinert wurden, dass sie vom jeweiligen Einsatzgebiet und den speziellen Eigenschaften verschiedener Individuen frei sind.
Die Definition von Ontologien geschieht mit Hilfe spezieller Sprachen, die dazu entwickelt wurden, wissensbasierte Systeme zu definieren. Ein Beispiel im Bereich der „Semantic Web Technologies“ (SWL) ist die „Web Ontology Language” (OWL). (Hitzler, Krötzsch, Parsia, Patel-Schneider, & Rudolph, 2009)

Durch die Möglichkeit, Zusammenhänge ohne Einschränkung durch den speziellen Anwendungsfall zu beschreiben und diesen dann auf spezielle Anwendungsfälle zu übertragen, spricht man von dem sogenannten „Open-World“ Konzept. Damit ist gemeint, dass durch die Erweiterbarkeit der Ontologien und durch Hinzufügen der einzelnen Individuen keine Begrenzungen gesetzt werden und somit die Beschreibung der jeweiligen Umgebung - „world“ - offen und damit beliebig erweiterbar ist. (Norvig & Russell, 2013)
Diese Eigenschaft der Ontologien ist eines der Hauptkriterien für die Verwendung in der hier vorgestellten Methode. Dadurch kann der aktuelle Stand des Wissens über digitale Zwillinge von Gebäude und Nutzer abgebildet werden. Auch in Zukunft können so, neue Zusammenhänge in ein Smarthome implementiert werden. Daher werden die semantischen Webtechnologien im Folgenden näher erläutert.

Semantische Webtechnologien

Wie Mitterhofer (2017) schon nachgewiesen hat, ist die Verwendung von „Semantic Web Technologies“ und der „Web Ontology Language“ (OWL) im Bereich der „Building Performance Simulation“ (BPS), ein zielführendes Mittel, um komplexe, stark interagierende Modelle miteinander zu verknüpfen und Rechnungen automatisiert durchzuführen. Diese waren bis dahin nur durch händisches Zuweisen von Variablen möglich. Die OWL wird vom World Wide Web Consortium (W3C) als Standard herausgegeben (Hitzler et al., 2009) und basiert auf der Grundidee des „Semantic Web“. Sie soll helfen, relevante Inhalte im Internet schneller zu finden, indem Web-Elementen durch Hinzufügen von Textblöcken, maschinenlesbare Bedeutungen gegeben werden. Dadurch ist es möglich, die bereits erläuterten Ontologien mit den Web-Elementen zu verbinden und mit Hilfe dieser Verbindungen weitere Beziehungen zwischen Elementen abzuleiten.
Die „Wissensbasierte Algorithmik“ ist laut Definition ein Teil des Bereiches der künstlichen Intelligenz (Norvig & Russell, 2013), da es somit möglich ist, aus bereits bekannten Zusammenhängen neues Wissen zu generieren. Folgende Teile spielen beim Umgang mit Wissensgraphen im Kontext dieser Arbeit eine besondere Rolle.

Die Sprache OWL wurde generiert, um im Bereich des „World Wide Web“ Ontologien zu beschreiben und für Maschinen lesbar zu machen. Sie wird verwendet, um die in dieser Arbeit erzeugte Ontologie für das Smarthome zu beschreiben. Grundsätzlich basiert sie auf „Resource Description Framework“ (RDF), das auch vom W3C für das „World Wide Web“ als Standard herausgegeben wurde. (W3C, 2014)
Dieses Framework dient dazu, logische Zusammenhänge mittels einfacher Systematik beschreiben zu können. Dazu wird das sogenannte „Tripel“, welches Zusammenhänge über eine Subjekt-Prädikat-Objekt-Beziehung beschreibt, verwendet. Diese Vorgehensweise hat sich als Standard in den „Semantic Web Technologies“ etabliert. (W3C, 2011)

Ein Beispiel für ein „Tripel“ wäre im Kontext dieser Arbeit:

Sensor Value X1 → hasUnit → Degree Celcius

Die Sprache „SPARQL“ ist dabei eine Sprache, die eigens dafür entwickelt wurde, Systeme auf Basis von Wissensgraphen abzufragen. Somit ist es möglich, ähnlich den Abfragen in Datenbanken, gezielte Abfragen zu Informationen im Wissensgraph zu erstellen. (Krötzsch & Rudolph, 2010)

Der Vorteil am „Open-World“ Konzept von Ontologien besteht in der flexiblen Erweiterbarkeit der Wissensgraphen. Dies ermöglicht die Übernahme von bereits existierenden Ontologien und erhöht damit die Wiederverwendbarkeit der neuen Ontologie und die Interoperabilität, da dasselbe Vokabular verwendet wird.

Um Ontologien mittels Regeln nutzen zu können, wurde die „Semantic-Web-Rule-Language“ (SWRL) entwickelt. Sie ermöglicht das Erweitern von Ontologien um Regeln und kann damit einen größeren Funktionsspielraum in der Wissensverarbeitung anbieten. Die SWRL wurde entwickelt, um maschinenlesbar zu sein und ist dadurch nicht leicht lesbar für Menschen. Daher wird im Folgenden grundsätzlich die für Menschen lesbare Syntax der SWRL verwendet. (Horrocks et al., 2004)

Das oben gezeigte Beispiel eines Tripels in dieser Syntax schreibt sich wie folgt:

hasUnit(X1,Degree Celcius)

Wobei hierbei auch Subjekt und Objekt als Variable definiert werden können.

Der folgende Ausdruck beschreibt alle Messgrößen, die die Einheit Grad Celsius haben, wobei das „?“ die variable repräsentiert:

hasUnit(X?, Degree Celcius)

Wie Mitterhofer (2017) bereits gezeigt hat, ist es gerade im Bereich des Gebäudes wichtig, eine richtige Objektzuordnung zu Räumen und Raumgruppen oder Bereichen eines Gebäudes zu erstellen. Mitterhofer zeigte, dass die Verwendung des Dijkstra-Algorithmus hierbei zielführend sein kann. Da hier bereits eine brauchbare Methode zur automatischen Zuordnung existiert, wird auf die technischen Notwendigkeiten und Details im Weiteren nicht mehr ausführlich eingegangen. Es bleibt jedoch festzuhalten, dass eine automatische Zuordnung von Informationen aus einem „Building Information Model“ in andere Systeme möglich ist. So konnten bereits Welle, Haymaker, und Rogers (2011) Informationen wie Räume, Raumgruppen und thermischen Zonen automatisch zu einer thermischen Simulation transferieren.

Reasoning-Prozesse

Reasoner bilden die Möglichkeit ab, aus komplexen, bereits bekannten Ontologien, neues Wissen zu generieren, das aufgrund der Systemkomplexität so nicht sofort sichtbar ist. Sie nutzen dabei die von der OWL bereits vorgegebenen Axiome und für den Reasoning-Prozess speziell gegebenen Regeln, welche beispielsweise in SWRL-Syntax formuliert werden können. Auf diese Weise kann ein Reasoner verschiedene Aufgaben erfüllen.
Durch die Klassenzugehörigkeit und deren Verbindung durch Eigenschaften ist es einem Reasoner möglich, Verbindungen zwischen Klassen herzustellen, die zu Beginn nicht definiert waren. So kann zum Beispiel vom Reasoner festgestellt werden, dass bei drei unterschiedlichen Individuen unterschiedlicher Klasse eine Verbindung zwischen Individuum eins und drei besteht, wenn es eine Verbindung zwischen eins und zwei und eine Verbindung zwischen zwei und drei aufgrund von Individuen oder Klasseneigenschaften gibt.

Alternativ ist es mit Hilfe von Regeln möglich, aus Ontologien weitere Informationen abzuleiten. Zur Definition dieser Regelsätze wird die Sprache SWRL verwendet. Besonderheit bei der Regeldefinition in dieser Sprache ist jedoch, das folgende Zusammenhänge nicht abbildbar sind:

- Disjunktionen
- Negationen
- freie Variablen

(Paulheim, 2013)

Grundlegend ist die Syntax für das Reasoning wie folgt aufgebaut:

Vorgeschichte → Folge

(Horrocks et al., 2004)

Sowohl der Bereich der Vorgeschichte als auch der Folge wird nun aus sogenannten „Atoms“, also kleinsten Einheiten, zusammengesetzt und diese über ein und, als Zeichen „∧“, miteinander verbunden. Variablen in diesem Zusammenhang werden, wie bereits gezeigt, durch ein Fragezeichen vor dem Variablennamen ausgedrückt.

Dadurch besteht die Möglichkeit das „Atom“, „Eine beliebige Person 1 ist ein Elternteil einer beliebigen Person 2“, wie folgt auszudrücken.

hasParent(?X1,?X2)

Ein Beispiel für eine in SWRL beschrieben Regel wäre:

hasParent(?x1,?x2) ∧ hasBrother(?x2,?x3) ⇒ hasUncle(?x1,?x3)

(Horrocks et al., 2004)

Diese Regeln lassen sich bis zur gewünschten Komplexität zusammenstellen und dem Reasoner übergeben. Damit besteht die Möglichkeit, neue Informationen aus einer Ontologie zu generieren und zu speichern.
Wie bereits beschrieben ist beispielsweise die Abbildung des logischen Operators „Oder“ mit dieser Sprache nicht möglich. Es existieren jedoch sogenannte Workarounds, um die beschriebenen Restriktionen zumindest teilweise zu umgehen. So kann beispielsweise eine Negation durch Verwendung einer Inversen-Klasse erzeugt werden. (Paulheim, 2013)
Auf weitere Möglichkeiten diese Beschränkungen zu umgehen soll hier aber nicht eingegangen werden. Durch die Eigenschaft, eigenständig Problemstellungen zu lösen und Informationen zur Lösung selbst zu generieren, gehört das Reasoning an sich zum Bereich der künstlichen Intelligenz (Norvig & Russell, 2013). Im Software-Bereich stehen für die Aufgabe des Reasonings verschiedene Algorithmen zur Verfügung. Die bekanntesten Reasoner sind:

- HermiT
- Pellet
- Fact++

(Mitterhofer, 2017; Motik et al., 2008)

Eine vollständige Liste aller aktuell gepflegten und unterstützen Reasoner findet sich unter „http://owl.cs.manchester.ac.uk/tools/list-of-reasoners/“ und wird von der University of Manchester aktualisiert. (University of Manchester, 2018)

Maschinelles Lernen

Die Definition des Begriffs „maschinelles Lernen“ erweist sich als diffizil. Chui und McCarthy (2018) definieren maschinelles Lernen wie folgt:

„Machine-learning algorithms detect patterns and learn how to make predictions and recommendations by processing data and experiences, rather than by receiving explicit programming instruction. The algorithms also adapt in response to new data and experiences to improve efficacy over time.“ (Chui & McCarthy, 2018, S. 1)

Einem Algorithmus des maschinellen Lernens unterliegt demnach immer eine Aufgabe eines zu erlernenden Zusammenhangs. Grundsätzlich ist eine Unterscheidung, Anhand der Art und Weise wie diese Aufgabe erfüllt wird, möglich. Die klassischen Algorithmen des maschinellen Lernens verwenden mathematische Methoden, wie beispielsweise Regressionsverfahren, um ihre Lernaufgabe zu erfüllen. Die sogenannten „Deep-Learning“ Algorithmen hingegen verwenden sogenannte künstliche neuronale Netze zur Verarbeitung der Lernaufgaben. Eine Unterscheidung beider ist demnach nur eine Unterscheidung der grundlegenden Arbeitsweise zur Erfüllung der Lernaufgabe.

Die dem aktuellen Stand des Wissens am nächsten kommende Definition der Lernaufgabe, die an die hier beschriebenen Maschinen übergeben wird, lautet:

„Eine Lernaufgabe wird definiert durch eine Beschreibung der dem lernenden System zur Verfügung stehenden Eingaben (ihrer Art, Verteilung, Eingabezeitpunkte, Darstellung und sonstigen Eigenschaften), der vom lernenden System erwarteten Ausgaben (ihrer Art, Funktion, Ausgabezeitpunkte, Darstellung und sonstige Eigenschaften) und den Randbedingungen des Lernsystems selbst (z.B. maximale Laufzeiten oder Speicherverbrauch).“(Görz & Rollinger, 2000)

Grundsätzlich wird ein derartiges System mit Eingangsdaten und Hintergrundinformationen, meist in Form von weiteren Daten, versorgt.

„Die genaue Definition der Lernaufgabe selbst muss dann gesondert erfolgen.“ (Görz & Rollinger, 2000)

Eine klassische Lernaufgabe als Beispiel wäre die Klassifizierung von Bildern. Hier wird dem System eine Menge an Bilddaten gegeben und deren Zuordnung zu bestimmten Klassen als Zielwerte.

Beispiel: Es werden Bilder mit Katzen und Hunden und der Zuordnung, welches Bild eine Katze abbildet und welches einen Hund vorgegeben. Das System muss nun die Unterscheidung der Bilder selbst lernen und dann an einem Testdatenset, von dem die Klassifizierung dem System nicht bekannt ist, testen. Die Quote der richtig zugeordneten Bilder bestimmt die Güte des Lernprozesses.

Eine Übersicht über relevante Lernaufgaben und dafür verwendete Methoden im maschinellen Lernen finden sich in der Veröffentlichung der Fraunhofer-Gesellschaft: „Maschinelles Lernen – Eine Analyse zu Kompetenz, Forschung und Anwendung“ (Döbel et al., 2018). Die darin enthaltene Übersicht über die Thematik ist im Folgenden abgebildet.

Lernstil	Lernaufgabe	Lernverfahren	Modell
Überwacht	Regression	Lineare Regression	Regressionsgerade
		Klassifikations- und Regressionsbaum-verfahren (CART)	Regressionsbaum
	Klassifikation	Logistische Regression	Trennlinie
		Iterative Dichotomizer (ID3)	Entscheidungsbaum
		Stützvektormaschine (SVM)	Hyperebene
		Bayessche Inferenz	Bayessche Modelle
Unüberwacht	Clustering	K-Means	Clustermittelpunkte
	Dimensionsreduktion	Kernel Principal Component Analysis (PCA)	Zusammengesetzte Merkmale
Bestärkend	Sequentielles Entscheiden	Q-Lernen	Strategien
Verschiedene	Verschiedene	Rückwärtspropagierung	Künstliche Neuronale Netze

Abb. 8 Übersicht über Modelle maschinellen Lernens (Döbel et al., 2018, S. 10)

Für die Wahl der richtigen Lernmethode ist daher zuerst die Lernaufgabe zu klären. Im Anschluss kann der richtige Algorithmus gewählt werden. So ist für die Bild-Klassifizierungen der wohl bekannteste und geeignetste Algorithmus das „Convolutional Neuronal Network“. Dieser eignet sich aufgrund seiner Komplexität jedoch nicht für einfachere Aufgaben. Die genaue Auswahl eines Algorithmus bleibt aufgrund der Vielseitigkeit der Thematik und der hohen Anzahl an Variablen, die es zu berücksichtigen gilt, daher einem Experten in diesem Bereich überlassen. Für einen groben Überblick kann aber nachfolgend abgebildete Tabelle dienen.

Tabelle 5 Übersicht Zuordnung von Datentypen zu Algorithmen des maschinellen Lernens (Hortonworks Inc., 2016, slide 5)

Data Sector	**Use Case**	**Input**	**Transform**	**Neural Net**
Text	Sentiment analysis	Word vector	Gaussian Rectified	RNTN or DBN (with moving window)
	Named-entity recognition	Word vector	Gaussian Rectified	RNTN or DBN (with moving window)
	Part-of-speech tagging	Word vector	Gaussian Rectified	RNTN or DBN (with moving window)
	Semantic-role labeling	Word vector	Gaussian Rectified	RNTN or DBN (with moving window)
Document	Topic modeling/ semantic hashing (unsupervised)	Word count probability	Can be Binary	Deep Autoencoder (wrapping a DBN or SDA)
	Document classification (supervised)	TF-IDF (or word count prob.)	Binary	Deep-belief network, Stacked Denoising Autoencoder
Image	Image recognition	Binary	Binary (visible and hidden)	Deep-belief network
		Continuous	Gaussian Rectified	Deep-belief network
	Multi-object recognition			Convolutional Net, RNTN (image vectorization forthcoming)
	Image search/ semantic hashing		Gaussian Rectified	Deep Autoencorder (wrapping a DBN)
Sound	Voice recognition		Gaussian Rectified	Recurrent Net
				Moving window for DBN or ConvNet
Time Series	Predictive analytics		Gaussian Rectified	Recurrent Net
				Moving window for DBN or ConvNet

Eine Unterscheidung ist hierbei nicht nur im Lernalgorithmus gegeben, sondern auch in der Art, wie der Lernprozess stattfindet.

Grundlegend kann hier in vier verschiedene Lernstile unterschieden werden.

- Überwachtes Lernen
- Unüberwachtes Lernen
- Teilüberwachtes Lernen
- Bestärkendes Lernen

Auch hier ist für die jeweilige Anwendung ein geeigneter Typ zu wählen. (Rebala, Ravi, & Churiwala, 2019, S. 19)

Als erläuterndes Beispiel wird im Folgenden, die wohl bekannteste Form eines Machine Learning Algorithmus verwendet, das „Neuronale Netz“.

Ein künstliches neuronales Netz besteht aus Knoten und Verbindungen zwischen den Knoten. Knoten, die übereinander angeordnet sind, werden als Schicht zusammengefasst. Es gibt eine Schicht für den Eingang und eine für den Ausgang von Daten. Schichten mit Knoten, die dazwischen liegen, nennt man Hidden-Layer. Die Verbindungen der Neuronen können mit Gewichten belegt werden. Diese können positiv, negativ oder neutral sein. Das gelernte Wissen eines neuronalen Netzes ist in den Gewichten gespeichert. Bei einem Lernprozess werden diese so lang verändert, bis das Netz die gestellte Lernaufgabe zufriedenstellend erfüllt. Die Anwendung eines solchen Algorithmus wird grundsätzlich in zwei Phasen unterschieden, die als Lernphase (Entwicklung) und Anwendungsphase (Implementierung) bezeichnet werden. (Gray, 2015; Rumelhart, Widrow, & Lehr, 1994)

Dabei ist die Entwicklungsphase der Zeitraum, in dem das System auf Basis der zur Verfügung stehenden Daten die gewünschten Zusammenhänge erlernt und, im Falle eines neuronalen Netzes, in Form der gewichteten Verbindungen speichert. Diese Phase kann je nach Komplexität des Algorithmus und der Menge der Daten sehr ressourcenaufwändig sein und bedarf großer Mengen an Rechenleistung. Hinzu kommt, dass viele Machine-Learning-Algorithmen aufgrund ihrer Architektur besser auf Grafikprozessoreinheiten (GPU’s) als auf normalen Prozessoren (CPU’s), berechnet werden können. Somit ist eine Ausstattung der rechnenden Computercluster mit dieser speziellen Hardware erforderlich. Eine wesentliche Ersparnis an Leistung und Zeitaufwand kann hier verzeichnet werden, wenn zur Lösung ähnlicher Probleme vortrainierte sowie generalisierte Algorithmen als Vorlage verwendet werden und diese mittels eines kleinen Datensatzes auf das jeweilige Problem angelernt werden. In diesem Fall spricht man von Transferlernen. (Gray, 2015)

Die Implementierungsphase ist die Phase, in der der bereits trainierte Algorithmus verwendet wird, um die Lernaufgabe, die ihm gestellt wurde, regelmäßig zu beantworten. Zum Beispiel in dem Hunde- und Katzenbilder voneinander unterschieden werden. Die hier benötigte Rechenleistung ist vergleichsweise geringer und bedarf keiner speziellen Hardware. Ein weiteres Lernen ist aber in dieser Phase nicht mehr möglich. (Nvidia, 2019)

Der oben beschriebene Ablauf der Entwicklung eines solchen Algorithmus ist in Abb. 9 noch einmal illustriert.

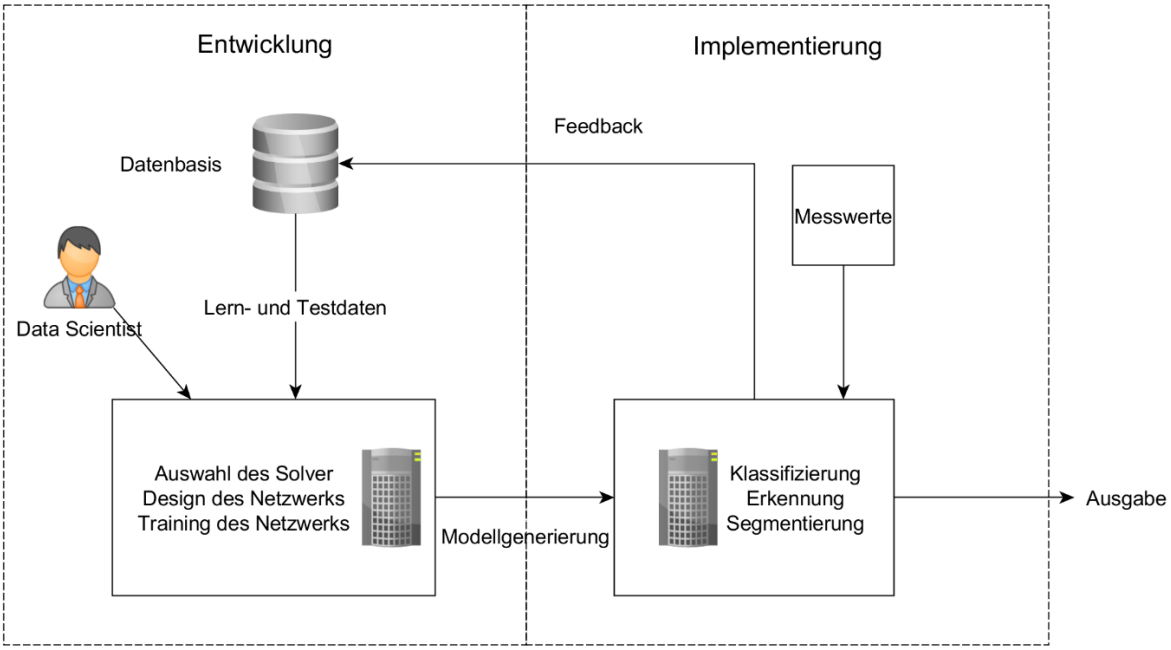


Abb. 9 Entwicklung von Machine-Learning Algorithmen (in Anlehnung an Gray, 2015)

Je nach gewähltem Lernstil, Lernverfahren und Modell benötigen diese Algorithmen unterschiedliche Mengen an Daten, um eine ausreichende Genauigkeit bei der Bearbeitung der Lernaufgabe zu erreichen. Dabei müssen die Trainingsdaten für die Beschreibung des Problems so repräsentativ wie möglich sein. Grundsätzlich gilt, je mehr Daten existieren, desto genauer arbeitet der Algorithmus nach der Trainingsphase. Größere Datenmengen bedeuten aber auch, dass bei gleicher Zeit mehr Rechenleistung zum Trainieren des Algorithmus benötigt wird. (Pedregosa et al., 2011)

Eine grobe Übersicht, welche Algorithmik welche Menge an Daten benötigt, gibt folgendes Bild. Hier ist der Entscheidungsweg für einen bestimmten Algorithmus des maschinellen Lernens auf Basis der vorhandenen Daten visualisiert.

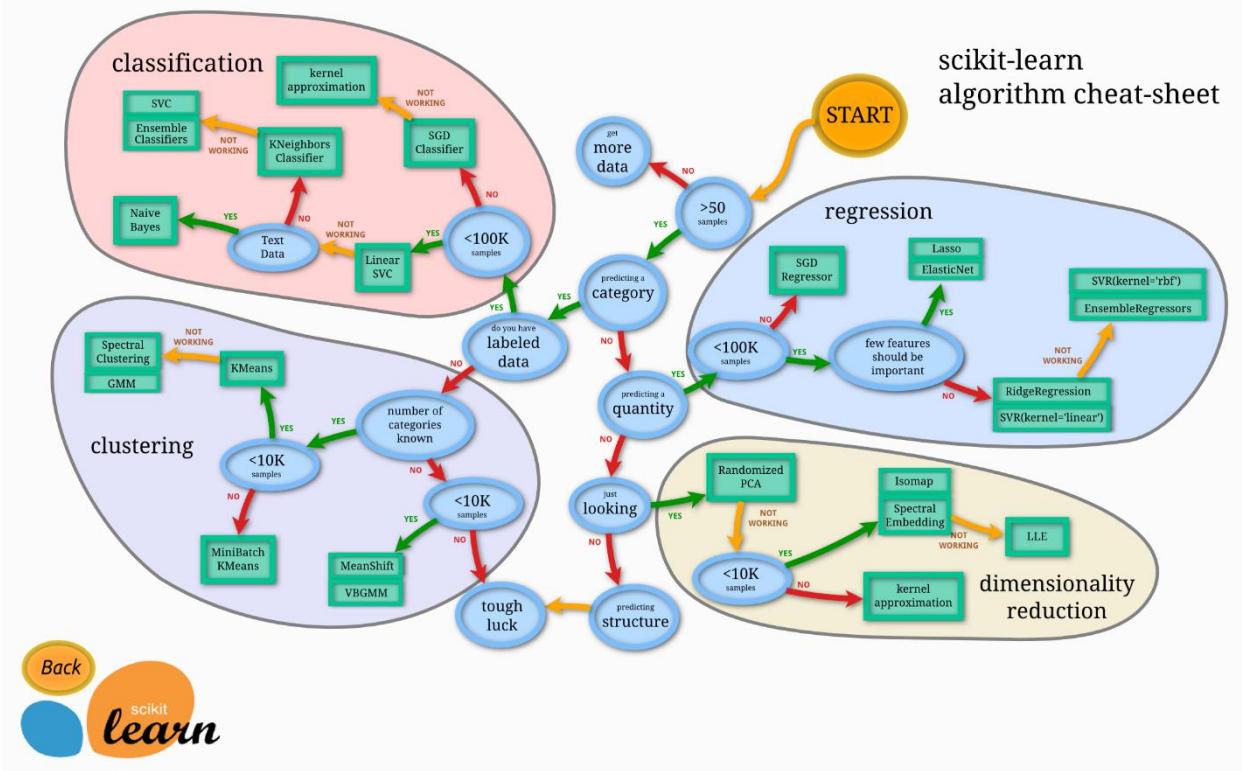


Abb. 10 Datenmengen verschiedener Machine-Learning Algorithmen (Pedregosa et al., 2011)

Ein zusätzliches Problem beim Anlernen eines Machine-Learning Algorithmus besteht darin, dass in der Lernphase „gelabelte“ Daten vorhanden sein müssen. Das bedeutet, die Daten müssen im Vorfeld bereits für das Anlernen mit den zu lernenden Eigenschaften (Label) markiert worden sein. Bei den üblichen Datenmengen für eine Lernaufgabe erzeugt dies einen enormen Aufwand.

Ausnahmen bezüglich des Aufwandes bei der Lerndatenaufbereitung stellen in diesem Fall nur unüberwacht lernende Algorithmen dar. Im Bereich der Hausautomation liegt die Ausnahme bei dem Ansatz des bestärkenden Lernens, auf dessen Einsatzmöglichkeit später nochmals gesondert eingegangen wird.

Eine weitere Möglichkeit, um das Problem der hohen Datenmengen zu umgehen, ist die Methode der „Data Augmentation“ (Daten Vermehrung). Dabei werden von einem Datensatz viele Duplikate mit jeweils kleinen Änderungen erstellt. In der Lernphase werden diese abgeänderten Datensätze wie normale Trainingsdatensätze verwendet und somit in den Lernprozess mit einbezogen. Diese Methode macht den Algorithmus grundsätzlich robuster. Ein Beispiel für diese Technik der Datenvermehrung ist in folgendem Bild dargestellt. Aus dem Bild einer Pflanze werden durch kleine Änderungen mehrere Bilder erzeugt. Für die Lernaufgabe der Klassifizierung von verschiedenen Pflanzentypen ist dies ein ausreichender Ansatz. (Pawara, Okafor, Schomaker, & Wiering, 2017)

Abb. 11 Data Augmentation am Beispiel von Pflanzenfotografien (Pawara et al., 2017, S. 5)

Ein zusätzliches Problem, dem bei der Entwicklung eines Algorithmus für maschinelles Lernen Rechnung getragen werden muss, ist das bereits angesprochene „Overfitting“. Dieser Effekt tritt auf, wenn der Algorithmus die Merkmale der Lerndaten so gut gelernt hat, dass er diese quasi „auswendig“ beherrscht, ähnlich einem Menschen beim Vokabeln lernen. Das hat zur Folge, dass die Lerndaten besonders akkurat wiedergegeben oder klassifiziert werden können, bei neuen Daten aber eine besonders hohe Fehlerquote auftritt (Rebala et al., 2019). Durch eine intelligente Konzipierung des Algorithmus kann dem „Overfitting“ bereits zu Beginn vorgebeugt werden, hierfür ist aber Erfahrung und Expertenwissen im Bereich dieser Techniken notwendig.

Anwendung

Zur Anwendung von Machine Learning Algorithmen gibt es verschiedene Frameworks, wie zum Beispiel „caffe2“, „MATLAB“, „MXNet“, „NvidiaCaffe“, etc. Sie bedienen meist verschiedene Programmiersprachen, oder sind für die Verwendung spezieller Hardware ausgelegt. Einen Überblick über verschiedene Frameworks bietet die für Entwickler bereitgestellte Internetseite der Firma Nvidia. (Nvidia, 2019)

Bekannte Anwendungen von „Deep-Learning“ Algorithmen im alltäglichen Leben sind zum Beispiel:

- Die von Google durchgeführte Image-Classification in Bildern ihrer Suchmaschine (Gray, 2015)
- ImageNet ein weltweit verfügbares Netz zur Klassifizierung von Bildern (Stanford Vision Lab, Stanford University, & Princeton University, 2016)

Andere Beispiele sind Objekt Detektion auf Videomaterial, Vorhersage von Ereignissen wie Nutzeraktionen und die Segmentierung von Daten.

3.3.4 Digitaler Zwilling

„Der digitale Zwilling ist eine Schlüsseltechnologie für die Digitalisierung unserer Welt.“ (Kuhn, 2017)

Grösser (2018) definiert den Begriff „Digitaler Zwilling“ wie folgt:

„Ein digitaler Zwilling (englisch „Digital Twin“) bezieht sich auf ein computergestütztes Modell eines materiellen oder immateriellen Objekts, welches für verschiedene Zwecke verwendet werden kann.“ (Grösser, 2018)

Ein digitaler Zwilling kann demnach ein Simulationsmodell eines Teilsystems sein. Durch die Kombination vieler Teilsysteme entsteht so der digitale Zwilling des Gesamtsystems. Die Herausforderung liegt hier also in der Schaffung eines Frameworks, das die Kombination vieler digitaler Zwillinge zu einem Gesamtsystem zulässt. Anwendung findet diese Thematik vor allem in der Produktionstechnologie sowie in der Produktentwicklung, zum Beispiel im Fahrzeugbau.
Dort werden Simulationsmodelle als digitale Zwillinge erstellt und im weiteren Entwicklungsprozess jeweils gegen akkurate Versionen getauscht. Auf diese Wiese ist es möglich, den digitalen Zwilling des Gesamtmodells zu verbessern, ohne jede einzelne Komponente bei Änderungen überarbeiten zu müssen.

Digitaler Zwilling im Bauwesen

Im Gegensatz zu anderen Branchen ist der digitale Zwilling im Bauwesen noch keines der regelmäßig verwendeten Planungstools. Wenngleich sein Einsatz auf verschiedenen Ebenen von der Erstellung über die Nutzung bis zum Abriss eines Gebäudes Vorteile verspricht. Zwar existieren grundsätzlich Ansätze zur Beschreibung digitaler Zwillinge von Gebäuden, wie beispielsweise die Datenhaltung der Gebäude über BI-Modellen, letztendlich existiert jedoch noch kein volleinsatzfähiges etabliertes Framework für diesen Anwendungsfall. Jedoch zeigen Einzelanwendungen in verschiedenen Forschungsvorhaben die Möglichkeiten für entsprechende Frameworks bereits auf. (Mitterhofer, 2017).

Im weiteren Verlauf wird auf einen digitalen Zwilling des Menschen und dessen essenzielles Verhalten in Bezug auf seine Bedürfnisse eingegangen. Daher wird im Folgenden eine kurze Zusammenfassung der Bedürfnisstruktur von Menschen gegeben. Eine grundsätzliche Einteilung der Nutzerbedürfnisse ist in Form der Maslow Pyramide zu finden. Diese wurde zur Illustration nachfolgend abgebildet.

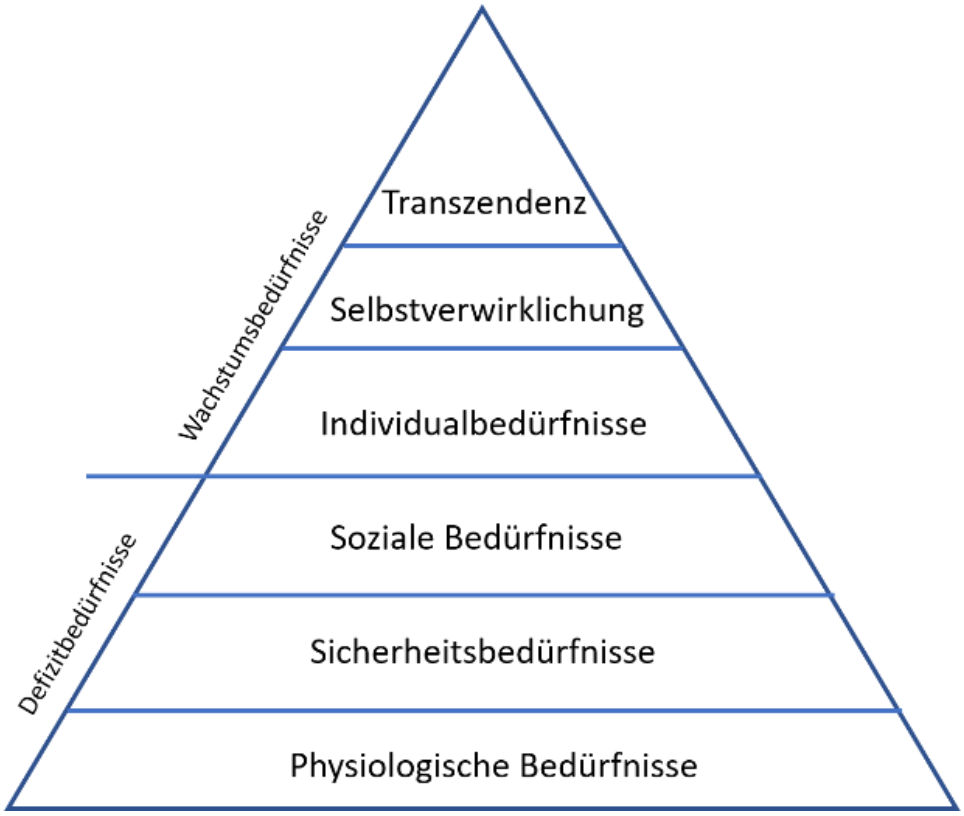


Abb. 12 Bedürfnis Pyramide (in Anlehnung an Maslow, 1943)

Auch wenn die Bedürfnispyramide von Maslow regelmäßig kritisiert wird, weil sie die komplexe Bedürfnisstruktur des Menschen bei schweren Entscheidungen angeblich regelmäßig unterschätzt, ist sie für den hier benötigten Anwendungsfall ausreichend. Im weiteren Verlauf wird ersichtlich, dass aus der Maslowschen Bedürfnispyramide nur grundlegende und keine detaillierten Informationen zu den Bedürfnissen entnommen werden. So zum Beispiel die Unterteilung der Bedürfnisse in einen Wachstums- und einen Defizitbereich.

3.3.5 Simulationsmodelle und deren Modularität

Nach Livio Mazzarella (2009) können bei der Modularität von Software verschiedene Arten unterschieden werden:

- **„Functional Layout Modularity“**
  Diese ermöglicht dem Anwender Module entsprechend ihrer Funktionalitäten an- oder abzuwählen oder in einer neuen Weise zu rekombinieren. Der Fokus liegt dabei auf der Funktion, die ein Modul erfüllt und damit eine logische Rekombination zulässt.

- **„Mathematical Models Modularity“**
  Mit mathematischer Modularität wird das Kapseln einzelner Algorithmen in Programmpakete bezeichnet. Diese Pakete können somit unabhängig verwendet werden. In Betriebssystemen kann dies zum Beispiel durch sogenannte Bibliotheken geschehen. Diese stellen verschiedene Funktionen zur Verfügung und können unabhängig voneinander verwendet werden. Eine Library selbst hat jedoch weder einen Programmablauf noch ein Interface. Daher müssen die Funktionen immer von einem Programm außerhalb der Library aufgerufen werden. Schnittstellen bei dieser Form von Modularisierung sind meist Stellen, an denen Informationen eingegeben oder ausgegeben werden.

- **„Standardized Mathematical Models Modularity“**
  Von standardisierter mathematischer Modularität kann gesprochen werden, wenn die zum Datenaustausch zwischen Modulen verwendete Sprache nicht nur gewährleistet, dass die Daten richtig übergeben werden, sondern auch, dass die Datenübergabe keine numerischen Probleme, sowohl in Teilmodellen als auch im Gesamtmodell, erzeugt.

- **„Code’s Modularity“**
  Die Modularität des Codes beschreibt die Möglichkeit im Rahmen objektorientierter Programmierung einzelne Codeteile, zum Beispiel in Form von Funktionen, wiederzuverwenden.

(Livio Mazzarella, 2009)

Verkoppeln von modularen Simulationen

Um mehrere Simulationsmodelle beziehungsweise Module miteinander zu verbinden, benötigt man eine Strategie. Da nicht alle Module dieselbe zeitliche Skala haben, auf der sie arbeiten oder dieselbe Menge an Daten benötigen, muss die Kopplungsstrategie bereits im Vorfeld definiert werden.

Eine Unterteilung in drei verschiedene Kopplungsarten geschieht entsprechend Struler, Hoeflinger, Kale, und Bhandarkar (2000) und Trcka (2008).

- **“Loose Coupling”**
  basiert auf nicht iterativen Modellen und ist daher am einfachsten in der Handhabung. Sie wird in dieser Arbeit zur Validierung der Methodik gewählt. Die im folgenden abgebildete Grafik zeigt zwei Beispiele für den Simulationsfortschritt und notwendige Schritte für den Datenaustausch im bezogen auf eine Zeitskala. (Struler et al., 2000)

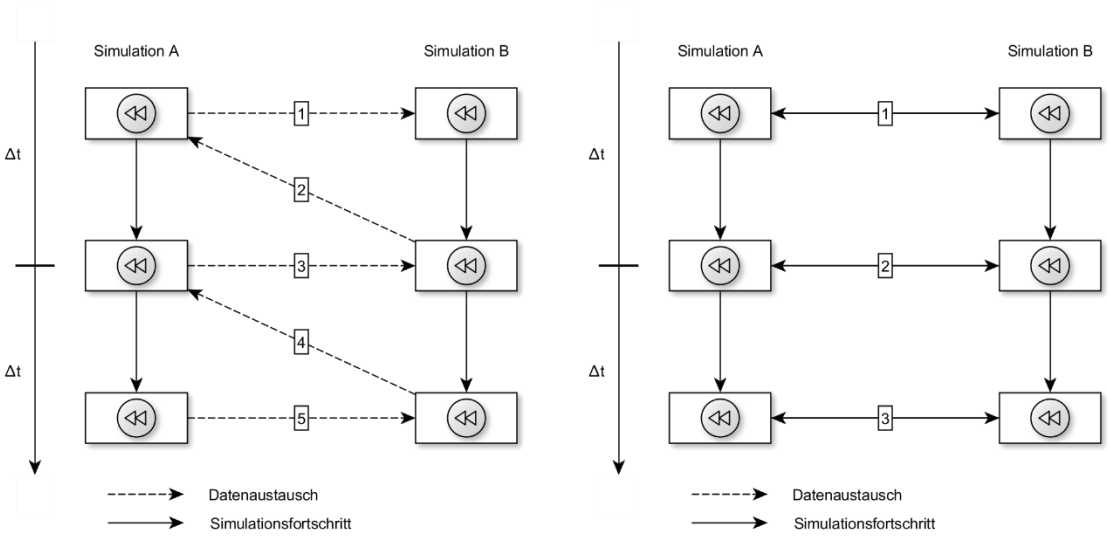


Abb. 13 Loose Coupling Method (in Anlehnung an Trcka, 2008, S. 35)

- **“Strong Coupling”**
  erfordert ein häufiges Iterieren bei Simulationen. Sollten Modelle iterativ arbeiten, ist darauf zu achten, dass die Daten zu den jeweiligen Zeitschritten an die richtigen Modelle übergeben werden, für den nächsten Zeitschritt Berechnungen angestellt werden und die Daten erneut ausgetauscht werden. Diesen Prozessablauf zeigt Abb. 14. (Struler et al., 2000)

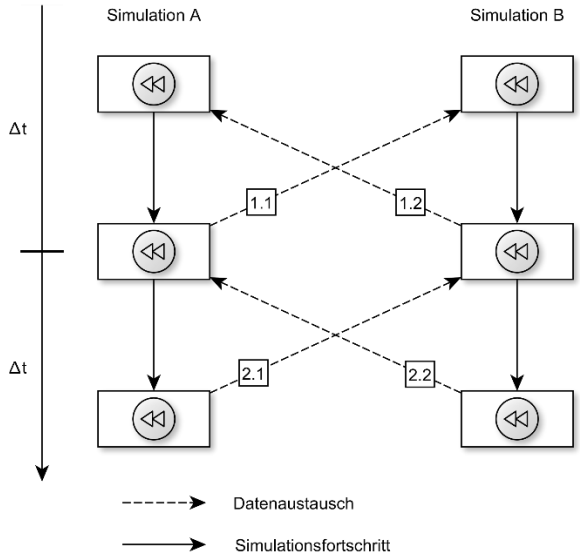


Abb. 14 Strong Coupling (in Anlehnung an Trcka, 2008, S. 35)

- **“Decomposition Strategy “**
  Grundsätzlich ist eine Zerlegung innerhalb der jeweiligen Domäne oder domänenübergreifend möglich. Berechnungen finden in den Domänen statt. Ein Beispiel dazu findet sich Abb. 15. Hier wird das Simulationsmodell in domänenübergreifende Komponenten gruppiert. (Trcka, 2008)

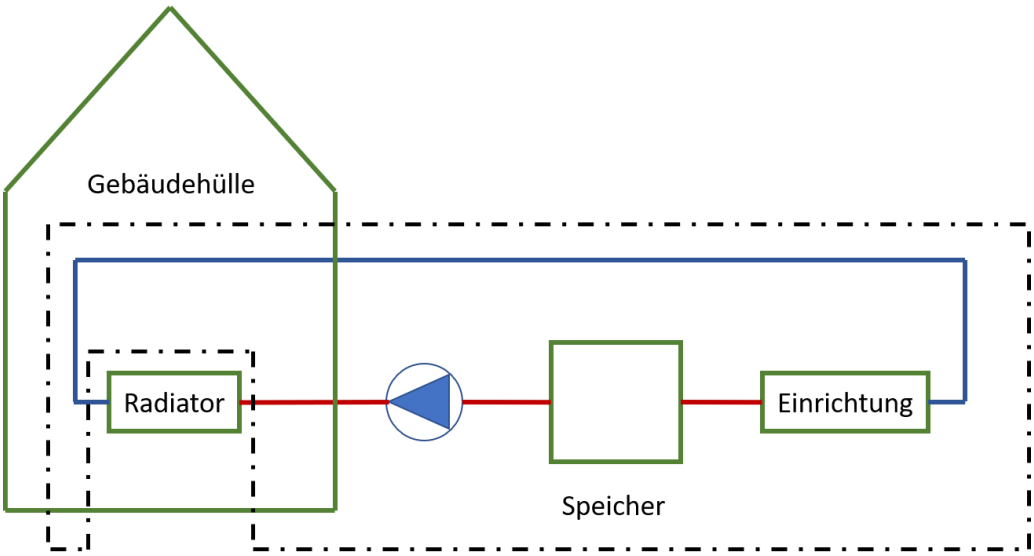


Abb. 15 Exemplarischer Intra-Domain Ansatz (in Anlehnung anTrcka, 2008, S. 30)

Nach einer Zusammenstellung von Mitterhofer (2017) ist gerade im Gebäude-Sektor eine „Decomposition Strategy“ vielversprechend, jedoch noch nicht realisiert.

Kopplungsstrategie

Folgende Grafik zeigt den Zusammenhang einer Simulation mit einem sogenannten „Master-tool“. Dieses ist in der BPS meist selbst ein Simulationstool und übernimmt die koordinierende Aufgabe zwischen den einzelnen Simulationen und seiner eigenen. Es stellt die richtige Übergabe der benötigten Parameter an den jeweiligen Berechnungsschritten sicher und erzeugt so eine aus mehreren Simulationsmodellen zusammengesetzte Gesamtsimulation. Problematisch ist bei dieser Vorgehensweise jedoch die Begrenzung einzelner Simulationsprogramme auf den Austausch ausgewählter Variablen. Auch ist der Zeit- und Entwicklungsaufwand für eine solche gekoppelte Simulation meist sehr hoch und erfordert das Eingreifen in das System auf Quellcodeebene. Die Funktionsfähigkeit einer solchen Simulation mit einer automatischen Variablenzuordnung hat bereits Mitterhofer (2017) bewiesen. (Mitterhofer, 2017)

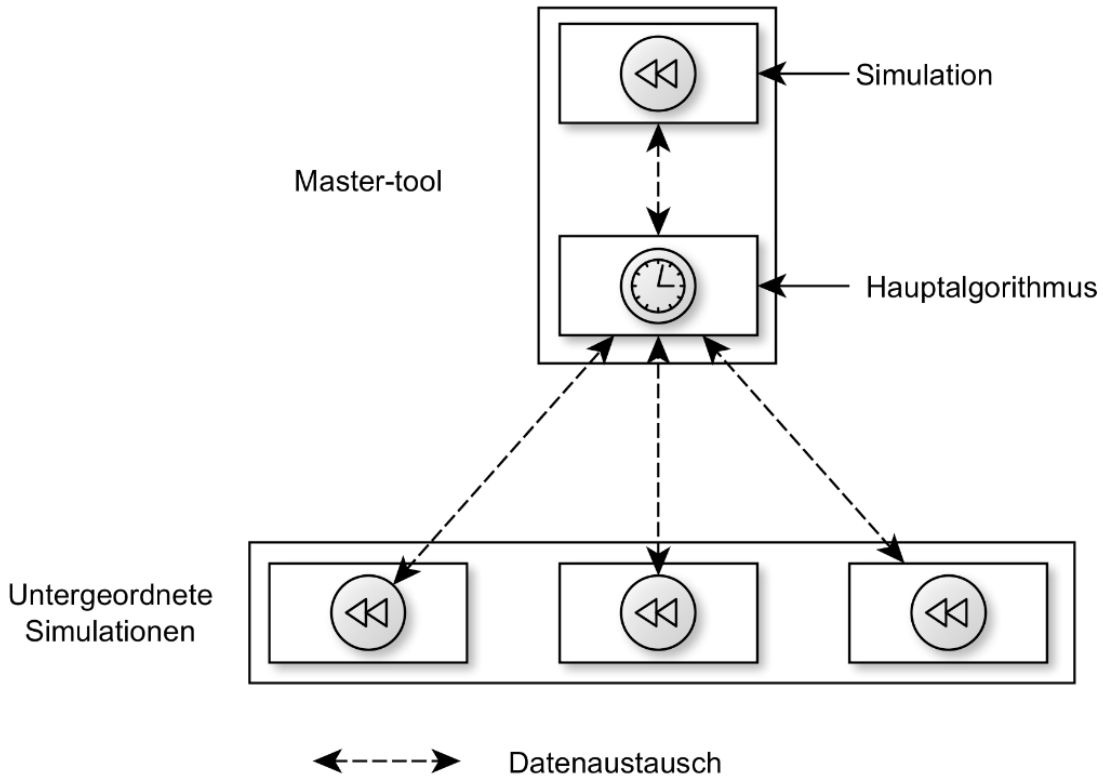


Abb. 16 Co-Simulation mit Masteralgorithmus (in Anlehnung an Mitterhofer, 2017, S. 23)

Bei der Implementierung der Simulationsmodelle ist auf deren Modellierungstyp zu achten. Entsprechend einer Zusammenfassung von Mitterhofer (2017), die auf den Ergebnissen von Citherlet (2001), (Clarke & Hensen, 2015) basiert, sind folgende Modellarten im Gebäudesimulationssektor bekannt. Ihre Unterteilung erfolgt entsprechend der Verwendungsart von Daten.

- **„Stand-Alone“**
  Hier werden unterschiedliche auf den jeweiligen Anwendungsfall spezialisierte Systeme mit unterschiedlichen Ansprüchen an die Eingabedaten verwendet. Die Auswertungen sind demnach nur auf einen Fall spezialisiert und können nicht zusammengefasst werden.

- **„Data-Model-Interpretation“**
  Auch hier werden spezialisierte Systeme zur Beschreibung eines jeweiligen Problems verwendet, jedoch beziehen hier alle Systeme ihre Daten aus derselben standardisierten Quelle, beispielsweise einem BIM Modell. Eine Auswertung der Ergebnisse muss wieder getrennt erfolgen.

- **„Process-Model-Interpretation“**
  Durch eine gesamtheitliche Betrachtung der physikalischen Gegebenheiten können durch die generierten Modelle verschiedene Rechenaufgaben gleichzeitig von einer Simulation erfüllt werden.

- **„Process-Model-Cooperation“**
  Hier werden verschiedene Modelle untereinander verknüpft und tauschen Daten gegenseitig aus. Auch wenn sie eigenständige, in sich geschlossene Systeme darstellen, sind sie in der Lage Informationen von anderen Modellen zu verarbeiten und so spezielle Fragestellungen zu beantworten.

Die bereits angesprochenen Modelle können entsprechend Mitterhofer (2017) anhand des Grades der Detailierung in die folgende Typen unterschieden werden.

- **„Single-Zone Model“**
  Physikalische Aspekte werden räumlich durch einen Simulationsknoten repräsentiert. Dieses Modell hat den geringsten Detailierungsgrad in dieser Zusammenstellung.

- **„Multi-Zone Model“**
  Einzelne Zonen, Räume oder Raumgruppen werden als in sich geschlossene Knoten betrachtet. Ein Austausch von Daten ist nur an den Zonengrenzen möglich.

- **„Zonal Flow Models“**
  Diese sind in der Lage Räume oder Zonen in jeweils kleinere repräsentative Raumvolumen zu teilen und Ströme zwischen diesen abzubilden. So können beispielsweise Massenströme in einem Raum berücksichtigt werden ohne eine vollständige fluiddynamische Simulation erstellen zu müssen. Ein Beispiel für eine solches Modell ist das „velocity propagating zonal model“ (VEPZO), das von Norrefeldt (2013) entwickelt wurde.

- **„Computational Fluid Dynamics“**
  Hier können unabhängig von Zonen fluiddynamische Zusammenhänge simuliert werden. Zu diesem Zweck wird der Untersuchungsraum in ein feinmaschiges Netz von Volumen geteilt und für diese jeweils Berechnungen sowie der Massenaustausch ermittelt. Im Vergleich zum vorhergehenden Modell ist die Netzgröße hier um ein Vielfaches kleiner und befindet sich im Größenbereich von Millimetern.

Zusammengefasst kann festgehalten werden, dass der Detaillierungsgrad der aufgelisteten Modelle von oben nach unten zunimmt. So besitzt das „Single-Zone-Model“ im Vergleich zu „Computational Fluid Dynamics“ (CFD) Modellen eine geringe Detailgenauigkeit und kann daher eher nur für Abschätzungen verwendet werden.

3.4 Inhaltliche Grundlagen für die Verwendung in Fallstudien

Im Folgenden werden die bauphysikalischen Grundlagen beschrieben, welche in den Fallstudien zur Evaluation der in dieser Arbeit vorgestellten Methodik verwendet wurden. Diese sind die Gefahr von Schimmelpilzbildung und die Belastung und der Umgang mit Radonkonzentrationen in Wohnräumen. Die Inhalte zu diesen Themen werden lediglich in den für die Fallstudien erforderlichen Umfang vorgestellt und stellen daher keinen vollständigen Überblick über die jeweiligen Thematiken und möglichen Methodiken dar. Ihre Auswahl erfolgte auf Grund der übersichtlichen Umsetzungsmöglichkeiten und damit im Sinne einer verständlichen Evaluation. Zudem ist in ihrer Kombination eine der im folgenden Verlauf öfter angesprochenen Dilemma-Situationen zu erwarten. Damit ist eine gute Eignung zur Prüfung der Funktionsfähigkeit dieses Teilaspektes der Methode gegeben.

3.4.1 Berechnung der Schimmelbildungsgefahr

Insbesondere in sanierten Altbauten ist die Bildung von Schimmelpilz ein häufiges Problem, mit dem Nutzer im laufenden Betrieb des Gebäudes konfrontiert werden. Teilsanierungen und mangelnde Lüftungskonzepte sind der Hauptgrund hierfür. Durch das Austauschen der Fenster in Altbauten und die damit verbundene höhere Luftdichtheit der Gebäudehülle können sowohl Transmissionswärme- als auch Lüftungswärmeverluste reduziert werden. Die zur Reduktion der Schimmelgefahr notwendigen Mindestluftwechsel können jedoch nicht immer sichergestellt werden. (Innenraumlufthygiene-Kommission des Umweltbundesamtes, 2017, S. 61)

Die Verantwortung einer regelmäßigen Wohnraumlüftung wird auf den Nutzer übertragen. Leider wird häufig mangels Sensibilisierung ein geeignetes Lüftungsverhalten hierbei zu spät entwickelt. Zusätzlich sind die erforderlichen Lüftungsintervalle (z.B.: drei Mal täglich, 5-10 min) bei solchen Bauten für regulär arbeitende Bewohner kaum realisierbar. Im Schadensfall obliegt es dann meist dem Nutzer nachzuweisen, dass er das für seine Wohnung erforderliche Lüftungsmaß eingehalten hat. Die Berechnung der drohenden Gefahr von Schimmelbildung ist jedoch bereits mit Hilfe des Isoplethen-Modells oder einem komplexeren Bio-Hygrothermischen Ansatz möglich (Sedlbauer, 2001). Bei diesen Methoden wird die Expositionszeit einer virtuellen Schimmelspore in für Auskeimung relevanter Atmosphäre betrachtet**.** Die Bildung von Schimmelpilz im Wohnraum hängt dabei im Wesentlichen von den Faktoren Oberflächentemperatur, relative Feuchte und Nährboden ab. Bei der Bildung von Schimmel wird grundsätzlich in zwei Phasen unterschieden.

Keimung: Hier hat sich noch kein Schimmel gebildet, die Sporen des Schimmels haben noch nicht ausgekeimt. In diesem Zustand ist der Pilz weder sichtbar noch schädlich und verbreitet sich auch nicht. Wenn die Umgebungsbedingungen über einen ausreichenden Zeitraum günstig sind, keimen die Schimmelsporen aus.

Wachstum: Nun bildet sich ein Pilzgeflecht, das bei ausreichender Stärke auch sichtbar wird. Ab dem Zeitpunkt des Wachstums bildet der Pilz neue Sporen in großen Mengen und damit auch die für den Menschen toxischen Produkte.

(Sedlbauer, 2001)

Am Beispiel des vereinfachten Isoplethen-Modells wird die Zeit berücksichtigt, in der eine Schimmelspore einer gewissen Kombination aus relativer Oberflächenfeuchte und -temperatur ausgesetzt ist. Mit Hilfe der graphischen Methode kann so in einer der folgend beispielhaft aufgeführten Grafiken eine Aussage über den Entwicklungsstand einer Schimmelspore erfolgen.

Die Grafik zeigt die Abhängigkeit der Sporenauskeimungszeit von der relativen Oberflächenfeuchte sowie der Oberflächentemperatur.

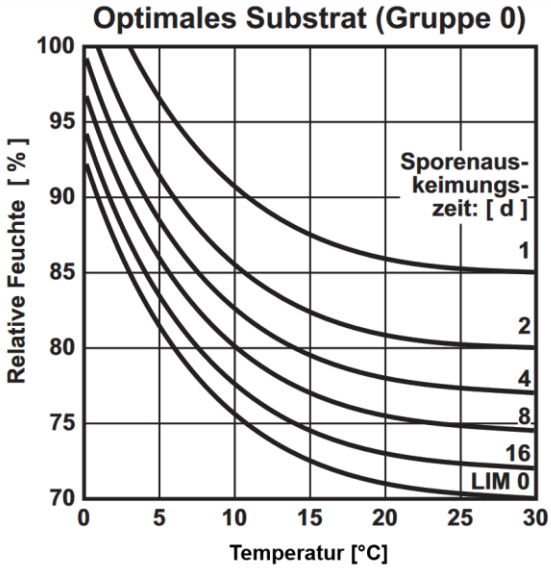


Abb. 17 Isoplethendiagramm zur Bestimmung der (Sedlbauer, Zillig, & Krus, 2001, S. 1)

Die verschiedenen Expositionszeiträume werden dabei zusammengefasst, um so eine Gesamtaussage über den Stand der Schimmelkeimung auf Grund der vergangenen Temperatur- und Feuchteverhältnisse zu machen. Aus diesem kann bestimmt werden, wie lange eine Schimmelspore bei stationären Bedingungen benötigt, um auszukeimen, wobei die Keimung der Zustand ist, der verhindert werden muss. Auch kann mit diesen Systemen das Wachstum eines Schimmelpilzes genauer beschrieben werden. Da jedoch von einem Schadensfall bereits bei Erfolgen der Keimung ausgegangen werden kann, soll dieser Aspekt nicht weiter betrachtet werden.

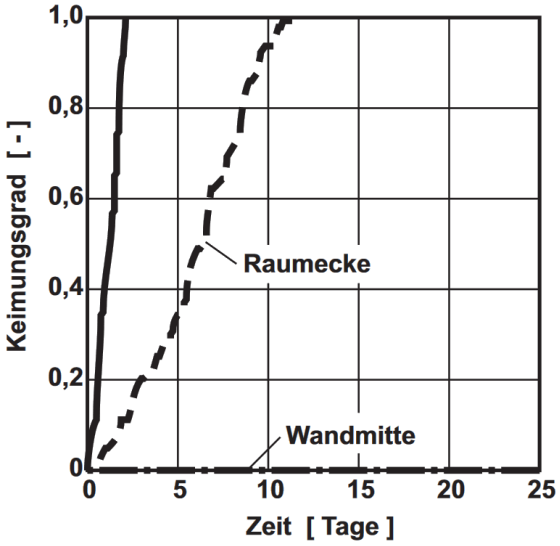


Abb. 18 Keimungsgrad einer Schimmelspore (Sedlbauer et al., 2001, S. 2)

Die oben abgebildete Grafik zeigt den Verlauf des Keimungsgrades einer Schimmelspore an verschiedenen Raumpositionen bei sonst gleichen Randbedingungen in Abhängigkeit von der Zeit. Ein Keimungsgrad von 1,0 entspricht der Auskeimung der Spore.

3.4.2 Berücksichtigung der Radon Belastung in Wohnräumen

Radon ist ein radioaktives Gas, das aus Spalten in der Erdoberfläche austritt. Es ist geruch- und farblos und kann dadurch vom Menschen nicht wahrgenommen werden. Daher ist seine Wirkung so verheerend. Durch die Inhalation wirken die Zerfallsprozesse direkt im Körper und sind damit um ein Vielfaches gefährlicher als bei einer äußeren Kontamination. Man geht davon aus, dass bei einer Erhöhung der Radonbelastung um 100 Bq/m³ die Gefahr einer Lungenkrebserkrankung um 16% steigt. (Darby et al., 2005)

Die Beschreibung der Radonverträglichkeit des Menschen ist somit auf der sicheren Seite durch eine Sprungfunktion zu beschreiben. Befindet sich der gemittelte Jahreswert über dem Referenzwert von 300 Bq/m³, so ist die Konzentration zu verringern. Befindet sich der Wert unter 300 Bq/m³, so ist die Belastung tolerierbar. (Bundesministerium für Umwelt Naturschutz Bau- und Reaktorsicherheit (BMUB), 2017)

Unter einer Konzentration von 100 Bq/m³ gilt die Belastung als unbedenklich (Darby et al., 2005). Die Radonbelastung eines Raumes hängt von vielen Faktoren ab. Neben der geografischen Position des Raumes sind vor allem die untersten Räume eines Gebäudes, insbesondere Kellerräume, von der Kontamination mit diesem Gas betroffen (Bundesamt für Strahlenschutz, 2018). Möglichkeiten zur Verringerung der Belastung sind durch bauliche Maßnahmen, wie einer radondichten Baustruktur, oder durch abwehrende Maßnahmen, wie einem aktiven Lüften und damit durch eine Erhöhung des Luftwechsels möglich (Bayerisches Landesamt für Umwelt, 2018). Aufgrund dieser Tatsache kann der Fall auftreten, dass ein aus Sicht der Radonbelastung sinnvolles Lüften in einem Kellerraum zu einer Erhöhung der Oberflächenfeuchtigkeit an kalten Kellerwänden führt und damit die Gefahr der Schimmelpilzbildung im Raum steigt. Eine genaue Vorhersage der Radonbelastung ist auf Grund der unterschiedlichen beeinflussenden Parameter kaum möglich, daher ist bis jetzt nur eine sensorische Erfassung und eine damit verbundene bedarfsgeführte Lüftung als Lösung des Problems vorhanden. (Bayerisches Landesamt für Umwelt, 2018)

3.5 Bewertung des Standes des Wissens

Im Folgenden der Stand des Wissens über Smarthomes zusammengefasst und diese auch in Bezug auf bauphysikalische Themen bewertet.

3.5.1 Technischer Stand des Smarthome

Der Smarthome-Sektor ist ein stark divergierender Sektor. Auf der einen Seite existieren proprietäre Systeme, mit denen versucht wird den Kunden über den Funktionsumfang der Software an die Hardware zu binden. Auf der anderen Seite stehen Community-Projekte, die genau diesen Effekt reduzieren wollen. Grundsätzlich ist das omnipräsente Problem des Smarthomes auch derzeit noch die mangelnde Interoperabilität, welche die anzubindenden Hardwarekomponenten betrifft. Communitygestützte Lösungen wie OpenHab oder FHEM ebenso, wie der Homekit von Apple, versuchen dem entgegen zu wirken. Auch diese können aber keine vollständige Lösung des Problems bieten. Die Hardware für Steuerung und Sensorik sowie deren Anbindbarkeit an ein Smarthome betreffend, ist also grundsätzlich derzeit noch mit Interoperabilitätsproblemen zu rechnen. Eine vollständige Lösung, mit welcher ein Smarthome erstellt und in Betrieb genommen werden kann und deren Komfort der Installation eines Computers oder einem anderen digitalen Verbraucherproduktes gleichkommt, ist derzeit noch nicht vorhanden.
Die Hardware, welche logische Operationen und Rechnungen durchführt, ist im Gegensatz zu den in Science-Fiction Filmen propagierten Großrechnern, derzeit noch auf Einplatinencomputern oder peripherer Hardware der Netzwerktechnik zu realisieren. Dies liegt aber grundsätzlich an der

geringen Auslastung der Systeme mit rechenintensiven Aufgaben. Solche Aufgabenstellungen wären zwar vorhanden, in der Praxis mangelt es jedoch an produktiven Systemen, die solche Aufgaben erfüllen. In der Forschung sind Systeme für solche Aufgaben häufiger vorzufinden. Hier liegt jedoch eine einzelfallbezogene, stark differenzierte Forschungslandschaft, ohne grundlegendem Konzept, vor.

3.5.2 Smarthome und Bauphysik - eine Gegenüberstellung

Durch eine eingehende Untersuchung im Rahmen dieser Arbeit zeigte sich, dass die Steuerabläufe im Smarthome eine parallele Vorgehensweise zu den Abläufen von Berechnungen oder Simulationen in der Bauphysik haben. Dieser Zusammenhang wird in nachfolgender Grafik in stark vereinfachter Weise abgebildet. Auf der linken Seite befindet sich eine Gliederung der grundsätzlichen Vorgehensweise bei bauphysikalischen Problemen. Auf der rechten Seite das Pendant aus dem Bereich Smarthome.

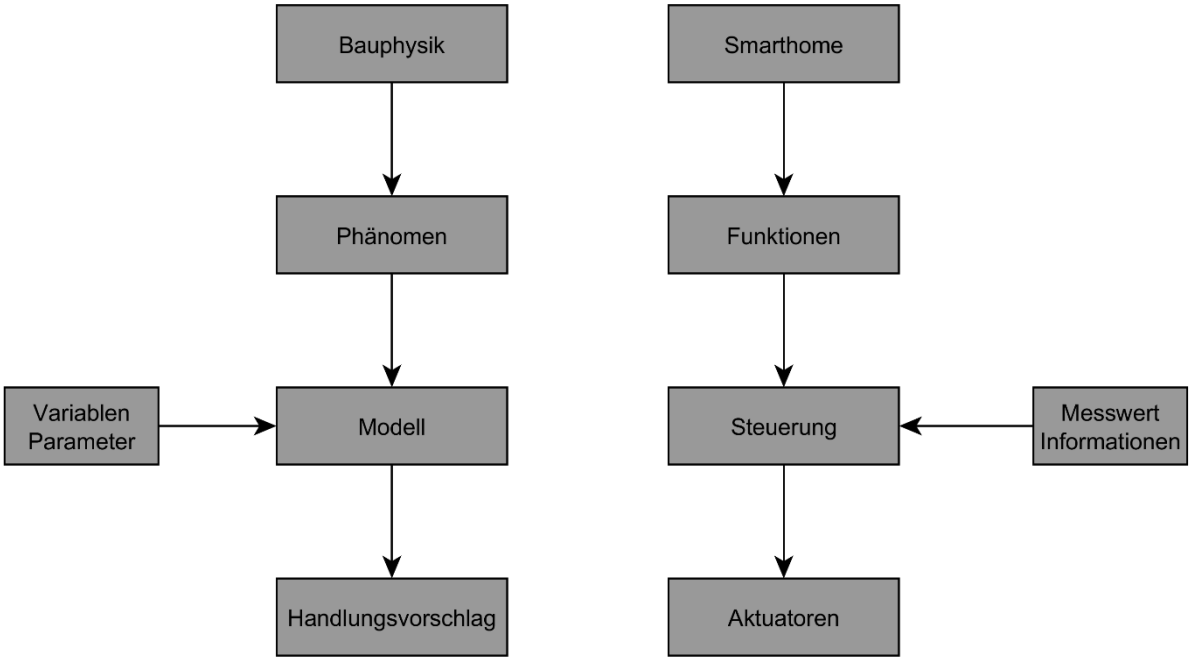


Abb. 19 Gegenüberstellung Modellierung in der Bauphysik und Steuerungen im Smarthome

Die Strukturelemente, Variablen und Messgrößen sowie Handlungsvorschlag und Regeln, lassen auf ähnliche Abläufe in beiden Systemen schließen. Eine Überprüfung der thematischen Schwerpunkte der Bauphysik und des Smarthomes zeigte, dass auf Basis dieser Struktur Überschneidungen der beiden Bereiche aufgefunden werden konnten. Die Analyse ergab, dass in nahezu allen Themenbereichen der Bauphysik eine Interaktion mit dem Smarthome stattfinden kann.

Folgende Grafik zeigt die Überschneidungen zwischen Themengebieten des Smarthomes mit denen der Bauphysik, wobei die Bereiche der Bauphysik blau und die des Smarthomes grün abgebildet wurden.

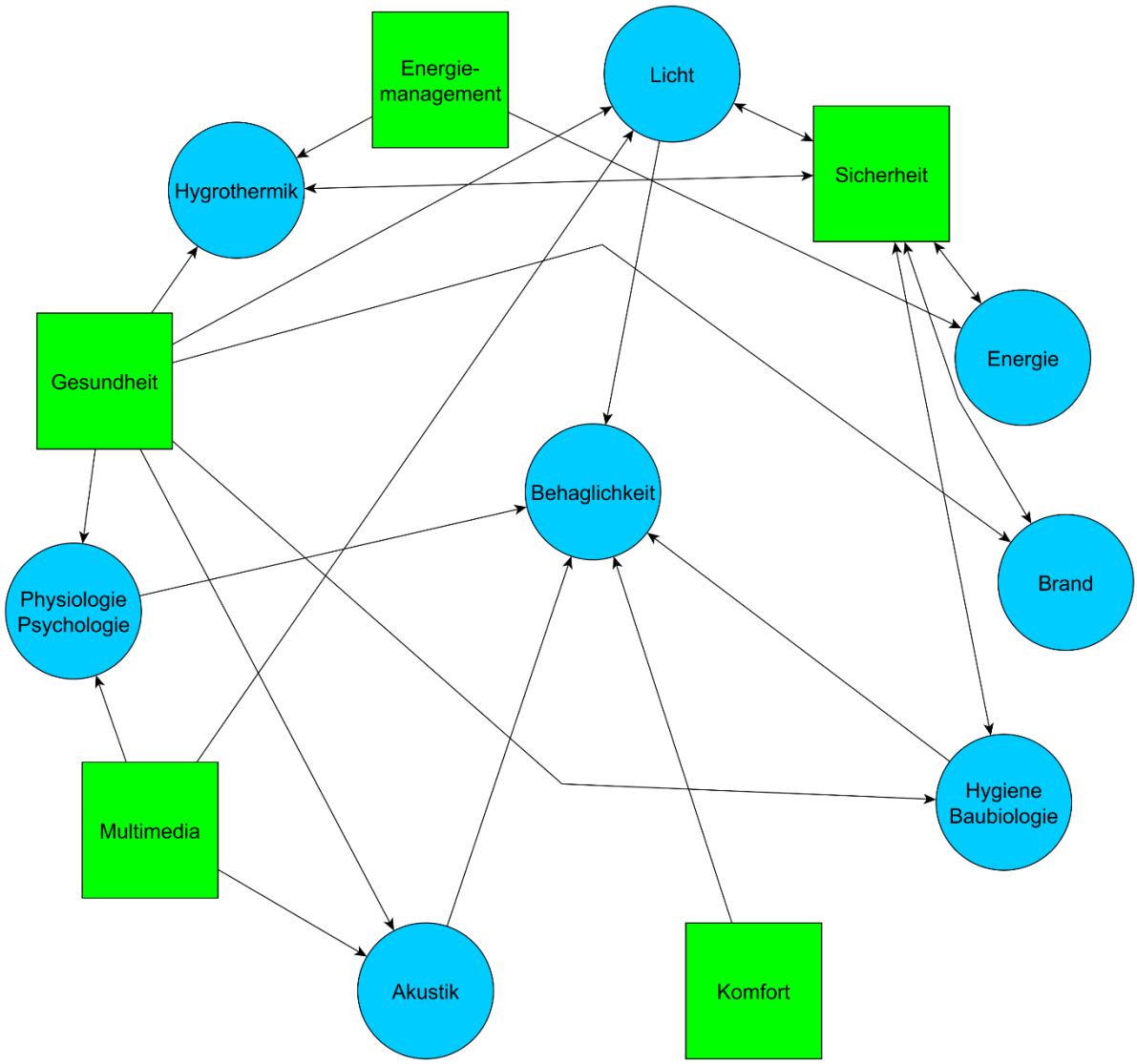


Abb. 20 Überschneidung der Themengebiete des Smarthome und der Bauphysik

Auffällig ist dabei, dass vor allem die Thematik um die bauphysikalische Behaglichkeit einen starken Anteil des Themengebietes „Komfort“ im Smarthome einnimmt. Die Steuerungen zur Bereitstellung des Komforts im Smarthome können jedoch in keiner Weise den Stand des Wissens der Bauphysik zur Thematik abbilden. Dies liegt hauptsächlich daran, dass viele Einflüsse auf Grund ihrer Komplexität oder schwierigen sensorischen Erfassung bis heute nicht berücksichtigt werden. Beispielsweise wird im Smarthome zur Steuerung des thermischen Komforts als alleiniger Parameter die Raumlufttemperatur betrachtet, während technisch bereits sogar die Bestimmung des menschlichen Metabolismus oder des Bekleidungsgrades möglich wären. (Peikos & Binsfeld, 2018) Während die olfaktorische, visuelle, akustische und psychologische Behaglichkeit im Bereich der Bauphysik eine große Rolle spielen und deren Beachtung für die Erstellung von Arbeitsplätzen üblich ist, so haben diese Themen im Bereich des Smarthomes bis jetzt keinen oder nur sehr geringen Einzug gefunden. Auch Sicherheitsfunktionen oder Gesundheitsfunktionen, die mit Hilfe der

Bauphysik einfach zu beschreiben wären, wie die Gefahr der Schimmelbildung oder die Überwachung des Radongehalts der Luft, haben bis zum jetzigen Zeitpunkt keinen flächendeckenden Einzug in Smarthomes gefunden. Zwar lassen sich vereinzelt in der Forschung Untersuchungen zu diesen Themen finden, einen für gängige Systeme anwendbaren Ansatz ist jedoch nicht zu finden. Im Bereich der Sicherheit finden sich hier mehr Funktionen zu Brand, Einbruch oder Wetterschutz.

3.5.3 Kontextsensitive und Aktivitätsbezogene Steuerungen

Auch wenn es bereits einige Forschungsergebnisse im Bereich der Aktivitätserkennung gibt, so reichen diese bis jetzt nicht aus, um tatsächlich kontextsensitive Steuerungen im Bereich des Smarthomes zu ermöglichen. In der Praxis sind kontextsensitive Steuerungen und Aktivitätserkennung bis jetzt nicht umsetzbar. Die lückenhafte Abdeckung des Forschungsfeldes lässt derzeit auch keine übergreifenden Themen oder Erkenntnisfelder erkennen. Grundsätzlich kann festgehalten werden, dass Aktivitätserkennung technisch, sowohl die Sensorik als auch die Software betreffend, möglich ist. Die Systeme sind jedoch weder standardisiert noch so weit fortgeschritten, dass Muster zuverlässig und übertragbar erkannt werden können. Die bisher durchgeführten Forschungsarbeiten beschränken sich auf spezielle Einzelthemen. Eine grundlegende Untersuchung zur Thematik Aktivitätserkennung im häuslichen Umfeld wurde bis jetzt nicht durchgeführt. Auch eine grundlegende Strategie zur Erschließung dieses Forschungsfeldes ist nach umfangreicher Recherche nicht zu erkennen. Einigkeit herrscht jedoch meist darin, dass vor allem individuelle Steuerungen stark kontextbasiert sein müssen. Kontextsensitivität ohne funktionsfähige Aktivitätserkennung ist zudem kaum möglich, was mehrfach festgestellt werden konnte. Daher kann man zusammenfassend festhalten, dass Aktivitätserkennung sowohl technisch als auch softwaretechnisch möglich ist. Für eine breite übergreifende Anwendung ist dieses Feld jedoch noch zu wenig erforscht. Außerdem sind kontextsensitive Steuerungen stark von der Erkennung dieser Aktivitäten abhängig. Aus dieser gegenseitigen Bedingung folgt, dass auch kontextsensitive Steuerungen technisch möglich wären, aber vor allem auf Grund des mangelnden Wissensstandes derzeit noch nicht weiterführend umgesetzt werden können. Das im folgenden vorgestellte Konzept muss demnach einen Spielraum für die Integration dieser Möglichkeiten lassen, damit sie nach ihrer Erforschung problemlos integriert werden können.

4 Entwicklung einer Methodik zur ganzheitlichen Steuerung eines Smarthomes

Um ein, im Vergleich zu den bestehenden Systemen, intelligentes und besser auf den Nutzer zugeschnittenes Smarthome entwickeln zu können, muss mit einer hohen Interdisziplinarität gearbeitet werden. Dem Entwickelten System muss ein Konzept zugrunde liegen, das es ermöglicht neue Technologien zu implementieren. Einem Entwickler muss die Möglichkeit gegeben werden, ohne aufwändige Recherche zu erkennen, wie ein gewisser Aspekt einzubinden ist und wie die internen Abläufe abzubilden sind. Dabei ist darauf zu achten, dass eine Kommunikation zwischen Entwicklern unterschiedlicher Bereiche kaum bis gar nicht möglich ist, sofern sie nicht im selben Unternehmen tätig sind. Mittel eines Kernkonzeptes muss also sichergestellt werde, dass die Schnittstellen so klar definiert sind, dass kein Spielraum für Interpretationen bleibt. Das im folgenden erläuterte System basiert dabei auf dem sogenannten „Open-World“ Konzept. Dieses impliziert eine Offenheit des Systems für Neuerungen und neue Aspekte. Aus diesem Grund kann bei so einem System im Vorfeld nicht abgeschätzt werden, für welche Aufgaben es letztendlich Verwendung finden wird. Diese Tatsache macht die vollständige Beschreibung verschiedener Aspekte und Interaktionen nahezu unmöglich. Daher sind nachfolgende Auflistungen zwar immer nach dem aktuellen Wissensstand erstellt, erheben aber grundsätzlich keinen Anspruch auf Vollständigkeit.

4.1 Konzipierung von Anforderungen anhand eines Kriterienkataloges

Ein intelligentes Smarthome erfüllt, wie bereits erläutert, mehr Funktionen als das bloße Regeln primärer Funktionen im Gebäude. Ziel muss es sein, den Nutzer in seinem Lebensalltag so gut wie möglich zu unterstützen. Nutzer agieren selbst in stark geregelten Umfeldern, wie einem Arbeitsplatz, bereits unterschiedlich und pflegen persönliche Abläufe, die auf ihre bestimmte Situation zugeschnitten sind. Deshalb ist im häuslichen Umfeld noch viel mehr davon auszugehen, dass Abläufe und Handlungen stark individualisiert geschehen. Zudem kommt, dass im häuslichen Umfeld als Mittelpunkt des Lebens eine große Anzahl verschiedener Tätigkeiten durchgeführt werden, die sich in ihren Anforderungen und Bedürfnissen jeweils stark unterscheiden. Weiter lassen sich diese Aktionen nicht, wie im betrieblichen Umfeld oft üblich, in einzelne Bereiche oder Räume trennen. Bereits am Schnitt vieler Wohnungen erkennt man, dass Räume wie zum Beispiel „offene Wohnküchen“ für viele verschiedene Tätigkeiten genutzt werden. Im häuslichen Umfeld ändert sich dementsprechend der Handlungskontext auf kleinem Raum in vergleichsweiser kurzer Zeit. Durch die verschiedenen Präferenzen der Nutzer bei verschiedenen Tätigkeiten, bleibt es selbst bei einem Experten nicht aus, in direkte Kommunikation und Interaktion mit dem Nutzer zu treten, um den bestmöglichen Steuerablauf zu finden. Ein intelligentes Smarthome muss daher individualisierbar, interaktiv, dynamisch und kontextsensitiv sein.

Um den schnellen technischen Veränderungen und dem immer steigenden Wissen auf Basis von Forschung in den verschiedensten Bereichen gerecht zu werden, ist zusätzlich eine hohe Modularisierung unumgänglich. Softwaresysteme ohne Modularisierung sind bereits jetzt in streng abgegrenzten Themenbereichen nicht mehr auf aktuellem Stand zu halten. Bei einem System, das keine wirkliche thematische Abgrenzung hat und alle Bereiche des Lebens unterstützen soll, ist eine Modularisierung dementsprechend umso wichtiger.

Bei der Entwicklung der Methodik musste daher darauf geachtet werden, dass sowohl Probleme auf unterschiedlichen zeitlichen Skalen miteinander interagieren können, als auch eine Struktur

geschaffen werden, die es zulässt die verschiedenen Bedürfnisebenen des Menschen zu berücksichtigen.

Es sind daher sowohl physikalische Phänomene, als auch biologische, psychische und physische Phänomene und deren Interaktion bei der Berechnung zu berücksichtigen. Diese können alle eine unterschiedliche Zeitauflösung der notwendigen Daten oder Ergebnisse erfordern. Zusätzlich gilt es bei der auf den Ergebnissen basierenden Steuerung zu berücksichtigen, dass sich in einem Raum nicht nur eine Person, sondern auch mehrere Personen mit unterschiedlichen Bedürfnissen und Präferenzen befinden können.

Dabei besteht die Gefahr, dass durch vielfältige Anforderungen und dem Wunsch, jedem Nutzer die bestmögliche Umgebung zu ermöglichen, die Steuerung in einen Zustand gerät, in der sie handlungsunfähig wird. Auch ein Zustand, aus dem es keinen schadensfreien Ausweg mehr gibt, wäre denkbar. Dementsprechend muss bereits im Vorfeld abgesichert sein, dass diese Zustände überprüft und vermieden werden.
Um die Modularisierung sicherzustellen musste das System so konzipiert werden, dass ein späterer Modulentwickler möglichst wenig Einschränkungen erfährt und innerhalb des Moduls frei arbeiten kann. Jedoch sind, um eine Interaktion von Modulen unterschiedlichster Entwickler zu gewährleisten, auch Rahmenbedingungen zu schaffen, die möglichst wenig Interpretationsspielraum zulassen.
Ein weiterer komplexer Faktor stellt die gewünschte Individualisierung dar, die es notwendig macht, verschiedene Aspekte des Nutzerverhaltens und Nutzerparameter mit der Zeit zu erlernen. Diese können sich aber über die Zeit auch ändern und müssen entsprechend angepasst werden. Eine Integration vielfältiger Algorithmen maschinellen Lernens muss also möglich sein, um für die jeweilige Lernaufgabe das Werkzeug wählen zu können.
Um schnell genug auf die individuellen Bedürfnisse des Nutzers eingehen zu können, ist ein detailliertes Wissen über seine aktuelle Situation im Wohnkontext erforderlich. Nur dadurch ist eine genaue Anpassung an diesen möglich. Sollten für die Erfassung dieses Kontextes Sensordaten erforderlich sein, die eine bestimmte Hardware benötigen, so ist bei einem Fehlen dieser Informationen zu berücksichtigen, dass das System mit ausreichend Ersatzdaten ausgestattet wird, um Grundfunktionen trotzdem bereitstellen zu können.

Auf Basis der im Rahmen dieser Arbeit erstellten Literaturrecherche konnten daher die, für ein intelligentes Smarthome wesentlichen Aspekte, herausgearbeitet werden. Der hierfür erstellte folgende Kriterienkatalog stellt demnach die grundlegende Basis für die Entwicklung der hier vorgestellten Methode bereit, wenngleich er auf Grund des verwendeten Open-World Ansatzes keine vollständige Auflistung für alle in Zukunft auftretenden Kriterien bereitstellen kann.

Nutzerindividuell

Ein Smarthome lernt von den Gewohnheiten und Eigenschaften des Nutzers, mittels zugehöriger Algorithmen und berücksichtigt diese im weiteren Steuerungsverlauf. Das Lernen ist also ein Schlüsselfaktor, um eine starke Individualisierung erzeugen zu können. Zudem ist das Lernen von dem Nutzer die Schlüsselkomponente, um in richtige Interaktion mit diesem treten zu können.

Interaktiv

Ein Smarthome muss von der Interaktion mit dem Nutzer lernen, um seine Steuerung daraufhin anzupassen. Im Umkehrschluss ist durch Interaktion auch ein Lernprozess beim Nutzer möglich. Für das Gebäude schädliches Verhalten wird aufgrund regelmäßigem Feedbacks des Systems aus Einsicht oder Resignation des Nutzers mit der Zeit eingestellt oder es zumindest eingeschränkt. Ein Beispiel aus dem Kontext der Bauphysik wäre hier ein schädliches oder nicht vorhandenes Lüftungsverhalten, das dem Nutzer durch regelmäßige Aufforderungen zum richtigen Lüften in der jeweiligen Situation abtrainiert werden kann (Lally, van Jaarsveld, Potts, & Wardle, 2010).

Dynamisch

Um Individualität und Interaktion sicherstellen zu können, muss der automatisierte Wohnraum seine Berechnungen auf aktuelle Daten stützen. Soweit möglich sind diese an die aktuelle Lebenssituation des Nutzers und das spezielle Gebäude angepasst. Dadurch kann das Smarthome dynamisch auf veränderte Randbedingungen oder veränderte Lebensgewohnheiten eingehen. Durch diese schnelle Reaktion auf Veränderung ist eine Verbesserung der Interaktion möglich und so auch ein besseres Lernverhalten der Algorithmik. Zudem erzeugt ein System, das dynamisch und nicht träge wirkt, ein höheres Vertrauen beim Nutzer. Dieses ist essenziell für die richtige und brauchbare Interaktion. Ein System, welches den Nutzer aufgrund seiner Trägheit behindert, wird keine Akzeptanz und damit auch keine Unterstützung bei einem Lernvorgang erhalten.

Kontextsensitiv

Durch Erkennung des aktuellen Handlungskontextes des Nutzers, kann die Steuerung besser an die nutzerspezifischen und temporären Wünsche angepasst werden. Dadurch ist es möglich das Steuerungskonzept nicht in einer generischen Weise umzusetzen, wie es bis heute oft der Fall ist, sondern die Steuerung individuell und zeitnah den veränderten Gegebenheiten im Wohnraum anzupassen. Kontextsensitivität ist also der Schlüssel, um das System aus Nutzerindividualisierung, Interaktion und Dynamik, die sich gegenseitig verstärken, zu gewährleisten.

Modular

Um die vielfältigen Bereiche im Leben eines Menschen umfassen zu können und sich nicht nur auf die lebensnotwendigen Kernfunktionen beschränken zu müssen, ist auch im systembedingten Funktionsumfang eine Individualisierung erforderlich. Daher muss das Einbinden neuer Funktionalitäten in das System leicht möglich sein. Zusätzlich muss durch eine gute modulare Struktur sichergestellt werden, dass alle relevanten Verbindungen zwischen den Modulen und Anbindungen an Daten und vor allem individualisierten Daten des Nutzers und des Gebäudes vom System automatisch erkannt werden. Nur so ist es möglich auch die Systemstruktur regelmäßig an die Individualität und Dynamik des menschlichen Lebens im häuslichen Umfeld anzupassen.

4.2 Kernkonzept – „Human centered“

Aus den vorhergehend erfolgten Anforderungen entstand für die hier vorgestellte Methode die Notwendigkeit, das System so zu konzipieren, dass bei den Kalkulationen der Mensch und seine Bedürfnisse im Vordergrund stehen. Auf diese Weise ist es möglich, sowohl den Kontext des Nutzers zu berücksichtigen als auch nutzerindividuelle Parameter und spezielle Nutzervorlieben mit in das Konzept zu integrieren.
Grundsätzlich wird im Folgenden davon ausgegangen, dass ein Nutzer, wenn er mit seiner Umgebung vollständig zufrieden ist, kein Bedürfnis verspürt an seiner Situation etwas zu ändern. Somit besteht auch kein Bedarf an den Rahmenbedingungen etwas durch ein automatisiertes Haus ändern zu lassen. Sollte der Nutzer mit seinem Umfeld in irgendeiner Weise unzufrieden sein, wird er dies zum Ausdruck bringen. Entweder direkt über die Kommunikation seines Bedürfnisses oder indirekt über eine Handlung, welche zur Befriedigung seines Bedürfnisses führt.

Ein Beispiel für die direkte Kommunikation ist:

„Mir ist zu kalt!“

Hier wird das Bedürfnis nach einer behaglichen Raumtemperatur ausgedrückt und dass die aktuelle Situation diese Anforderung nicht erfüllt.

Ein Beispiel für eine indirekte Interaktion wäre, wenn der Nutzer zum Thermostat geht und den Sollwert für die Raumtemperatur erhöht.

Die direkte Formulierung von Bedürfnissen hat den Vorteil, dass das Bedürfnis dadurch klar erkannt werden kann. Mit den üblichen Methoden der Wissenschaft kann so schnell eine Möglichkeit gefunden werden das Bedürfnis zu befriedigen.

Bei der indirekten Methode kann zwar auf einen aus der Handlung resultierenden primären Bedürfnischarakter geschlossen werden, jedoch besteht im Gegensatz zur ersten Methode eine gewisse Unsicherheit, ob die Aktion nicht aus einem anderen und nicht sofort abzuleitenden Grund ausgeübt wird. So kann das Öffnen des Fensters den Wunsch nach Frischluft stillen oder den Sinn haben wegen zu hoher Luftfeuchtigkeit zu lüften. Allein aus der Handlung kann diese Aktion also keinem eindeutigen Bedürfnis zugeordnet werden. Aus diesem Grund ist es sinnvoll bereits im Vorfeld zu vermeiden, dass solche Anforderungen entstehen, wodurch auch die Menge der fehlinterpretierten Aktionen des Nutzers reduziert wird. Um die Bedürfnisse des Nutzers zu stillen bevor sie entstehen, ist eine genaue Kenntnis über die Entstehung und die Zusammenhänge dieser erforderlich. Daher kann dieses Konzept als „Human-Centred“ beschrieben werden. Die Bedürfnisse und Eigenschaften des Menschen stehen im Mittelpunkt der Betrachtung.

4.2.1 Digitaler Zwilling des Gebäudes

Um die Interaktion zwischen Nutzer und Gebäude richtig abbilden zu können, wird daher eine digitale Version des Gebäudes erstellt. In der Informatik spricht man von dem bereits erläuterten digitalen Zwilling.
Diese Technologie eröffnet die Möglichkeit, die speziellen einzigartigen Besonderheiten einer jeden Wohnsituation zu berücksichtigen, nicht nur um die Interaktion zwischen Mensch und Gebäude abzubilden, sondern auch zwischen verschiedenen Gebäudeteilen und Bereichen. Durch die Teilung des DGZ in Unterkomponenten, die alle mit einer einheitlichen Schnittstelle verbunden sind, ist ein ungehinderter Datenaustausch zwischen den einzelnen Teilen möglich. Dies ermöglicht Komponenten des Gesamtsystems, die einen höheren Detaillierungsgrad erfordern, genauer zu betrachten, ohne das gesamte Simulationsmodell überarbeiten zu müssen. Auch ein Austausch von Systemteilen gestaltet sich damit besonders komfortabel.

Der derzeit übliche funktionsbasierte Automationsansatz wäre dadurch um ein breites Spektrum an Möglichkeiten erweitert. So könnten auch Funktionen in Form von Modulen anderer Entwickler untereinander interagieren und Daten austauschen. Durch diese bei komplexen Systemen kaum noch vorhersehbaren Datenströme wird eine bessere Abbildung der Realität erzeugt als bisher mit einzelnen in sich abgeschlossenen Modulen, die ihre Berechnungen ausführen und direkte Steuerungen vornehmen. Der Ansatz der digitalen Zwillinge ist besonders in der Fertigung und Automobilindustrie bereits Stand des Wissens und wird regelmäßig eingesetzt, da hier eine sehr hohe Wiederverwendbarkeit von validierten Simulationsmodellen besteht. Für ein sich entwickelndes Smarthome, das von verschiedenen Akteuren beeinflusst wird, ist dies zusätzlich ein Weg, um das derzeitige Interoperabilitätsproblem zu lösen und erzeugt damit die Möglichkeit auch auf Software-Ebene eine sinnvolle Vernetzung zu gewährleisten.

Grundsätzlich ist der hier vorgestellte DGZ in zwei Bereiche gegliedert. Eine schematische Abbildung des DGZ findet sich in folgender Abbildung.

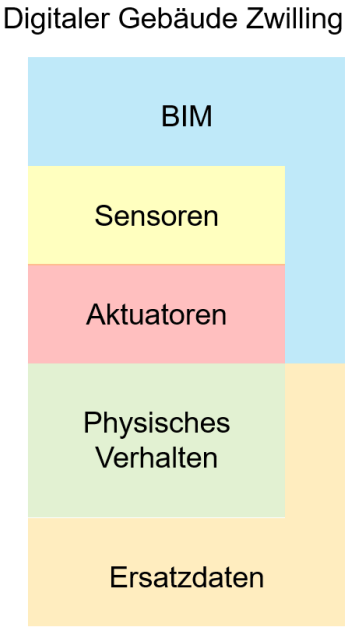


Abb. 21 Schematische Gliederung des digitalen Gebäudezwilling

Der obere Bereich repräsentiert den physikalischen Bereich des Gebäudes mit Wänden, Türen, Fenstern, Sensoren, Materialien, Aktuatoren etc. während der untere Bereich des DGZ den theoretischen Bereich repräsentiert. Hier sind Funktionen, Berechnungsgrundlagen, Simulationsmodelle und statistische sowie Normdaten hinterlegt.

Im Detail beinhalten die Bereiche folgendes:

BIM

Alle relevanten Gebäudedaten die gebäudespezifisch sind, wie zum Beispiel:

- Position von Bauteilen, Räumen, Einrichtungsgegenständen, etc.
- Abmessungen von Bauteilen, Räumen, Einrichtungsgegenständen, etc.
- Art der Bauteile, Räume, Bereiche etc.
- Parameter (wie zum Beispiel: verwendete Materialien, Reflexionsgrade etc..) der Bauteile, Räume, Einrichtungsgegenstände, etc.

Grundsätzlich lässt sich sagen, dass sich alle Eigenschaften von physikalischen Repräsentanten des Gebäudes hier datentechnisch abbilden lassen.

Sensoren

Im Bereich Sensoren werden alle Daten aufgelistet die aktiv im zeitlichen Verlauf des Gebäudes erfasst werden. Beispielhafte Sensordaten wären:

- Temperaturen
- Feuchtigkeiten
- Wärmeströme
- Stromstärken
- Spannungen
- Ein- und Auszustände
- Öffnungszustände

Der untere Teil des abgebildeten Modells eines digitalen Zwillings enthält alle theoretischen Aspekte, die ein Gebäude betreffen. Im Detail sind das für die einzelnen Bereiche:

Aktuatoren

Alle Komponenten die automatisiert eine Aktion im Gebäude ausführen können. Beispiele hierfür sind:

- Lüftungsgeräte
- Heizungen
- Jalousien
- Leuchtmittel
- Sprachausgabegeräte
- Anzeigegeräte

Physisches Verhalten

Einzelne Algorithmen, die das physikalische oder auch materielle Verhalten oder Daten des Gebäudes näher beschreiben. Beispielhaft wäre zu nennen:

- Berechnung von Oberflächentemperaturen mit Hilfe von Messdaten
- Berechnung von Luftfeuchtigkeiten beim Öffnen von Fenstern
- Alle denkbaren Simulationsaufgaben die das Gebäude betreffen

Ersatzdaten

Die Ersatzdaten stellen einen Datenspeicher dar, über welchen sichergestellt wird, dass verschiedene Module auch ohne das Vorhandensein spezieller Sensorik funktionieren. Hier ist es möglich, übliche Ersatzwerte aus Normen oder Studien zu hinterlegen. Auch kann klar definiert werden, dass für den gewünschten Datentyp keine Ersatzdaten verfügbar oder sinnvoll sind.
Prinzipiell kann eines der nachfolgend beschriebenen Module mit zwei Typen an Daten verbunden sein: Eingangsdaten und Ausgangsdaten. Für jeden dieser Datentypen muss entweder ein Satz an Ersatzdaten zur Verfügung stehen oder definiert sein, dass kein Ersatz vorhanden ist. Auf diese Weise ist sichergestellt, dass alle Module soweit möglich, auch ohne spezielle Hardware lauffähig sind. Inwieweit das möglich und sinnvoll ist, kann hier durch den Modulentwickler bestimmt werden. Eine Anbindung von real erfassten Daten bewirkt im Normalfall immer eine bessere Anpassung an die individuelle Situation. Sollte dies in einem speziellen Fall eines Moduls nicht der Fall sein, kann bei der Modulkonzipierung im Zuge der Definition der Eingangsdaten, statt einer Eingangsvariable

ein Eingangsparameter gewählt werden. Damit ist im Folgenden klar definiert, dass als Daten keine Sensorwerte verwendet werden.

4.2.2 Digitaler Zwilling des Nutzers

Eine grundsätzliche Einteilung der Nutzerbedürfnisse ist in Form der bereits beschriebenen Maslow Pyramide zu finden, welche trotz vorherrschender Kritik für die Bestimmung der folgenden erläuterten Prioritäten und deren Abstufung verwendet werden kann, da sie dir grundlegenden Bedürfnisse des Menschen darstellt. Die Entscheidungen, die ein Smarthome auf dieser Basis für den Nutzer trifft, dienen der Unterstützung und sind damit keine existentiellen schweren philosophischen Fragestellungen. Im Regelfall werden solche existenziellen Entscheidungen noch direkt vom Menschen getroffen. Andernfalls wären Situationen wie sie in Science-Fiction Filme wie „Smart House“ (Burton, 1999) über Smarthomes, die ihrem Nutzer durch zu über protektives Verhalten schaden, durchaus denkbar. Die ethische Frage nach der Bevormundung der Menschen durch eine Maschine soll an dieser Stelle jedoch nicht behandelt werden. Die Maslow Pyramide unterscheidet in zwei grundlegende Bedürfnisbereiche:

- Defizitbereich
- Wachstumsbereich

Im Rahmen dieser Arbeit spielt nahezu ausschließlich der Bereich der Defizitbedürfnisse eine Rolle. Diese wurden zum Zweck der programmtechnischen Verarbeitung in vier Kategorien eingeteilt und deren Wichtigkeit entsprechend der Pyramide von Maslow wie folgend absteigend angeordnet.

- Sicherheit
- Gesundheit
- Komfort
- Sonstige

Um vorrausschauend zu agieren, so dass ein bestimmtes Bedürfnis erst gar nicht entsteht, sind genaue Kenntnisse über das Entstehen von Bedürfnissen erforderlich. Dies ist im Bereich der Bauphysik in vielen Bereichen bereits vorhanden. Beispielsweise kann das thermische Behaglichkeitsempfinden im Vorfeld ermittelt werden und dadurch vorab bereits erkennbar sein, ob ein Mensch in einer bestimmten Situation das Bedürfnis entwickeln wird die Raumtemperatur zu verändern. Das bedeutet, zur Funktionstüchtigkeit dieses Konzepts, ist ein essenzieller Bestandteil, dass die Entstehung von Nutzerbedürfnissen beschrieben werden kann. Die Beschreibung dieser Zusammenhänge des Nutzers führt zur hier vorgestellten Entwicklung eines „Digitalen Zwilling des Nutzers“. Für die Funktionstüchtigkeit dieses Konzeptes ist ein essenzieller Bestandteil, dass die Entstehung der Nutzerbedürfnisse beschrieben werden kann. Die Individualisierung der nutzerspezifischen Parameter lassen sich am besten über ein möglichst detailliertes digitales Abbild des Nutzers generieren. Dadurch ist prinzipiell auch die Interaktion zwischen Nutzern abbildbar, da für eine Individualisierung jeder Nutzer seinen eigenen digitalen Zwilling benötigt. Die Entwicklung eines Frameworks zur Beschreibung von Nutzerinteraktionen ist jedoch nicht Aufgabe dieser Arbeit.

Was die grundsätzliche Möglichkeit der Modellierung durch digitale Zwillinge bei Nutzern betrifft, kann ein Vergleich zur Verwendung von Simulationsmodellen gezogen werden. Diese können je nach Anwendungsfall einen unterschiedlichen Grad der Detailierung aufweisen. Dementsprechend ist die Einteilung der verwendeten Modelle in die bereits gezeigte Unterteilung klassischer Simulationsansätze möglich. Je höher der Detailierungsgrad der Simulationsmodelle, desto höher ist auch der Rechenaufwand zur Lösung der Modelle. Das hier entwickelte Model lässt es dem Entwickler der jeweiligen Komponenten frei, sich für eine sehr detaillierte Lösung oder einen eher approximativen Ansatz zu entscheiden.

Auch der DNZ hat eine ähnliche Untergliederung wie der DGZ in einen oberen Bereich und einen unteren Bereich. Ähnlich wie beim DGZ beschreibt der obere Bereich eher Parameter und physikalische Eigenschaften des Nutzers, während der untere Bereich das theoretische Wissen und Ersatzdaten liefert. Die genaue Aufteilung des DNZ ist der folgenden Abbildung zu entnehmen. Auch hier ist entsprechend der Abbildung eine Unterteilung in fünf Bereiche möglich.

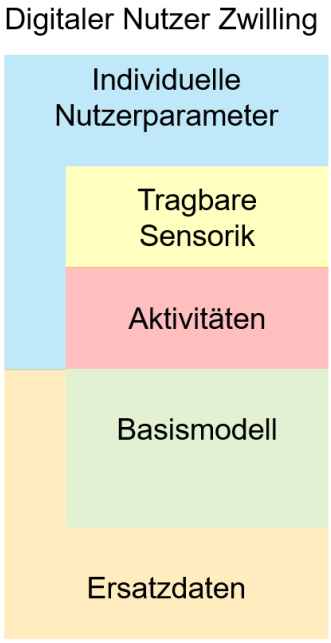


Abb. 22 Schematische Gliederung des digitalen Nutzerzwilling

Im Detail beinhalten diese einzelnen Bereiche des DNZ folgendes:

Individuelle Nutzerparameter

Alle relevanten Nutzerparameter die auf das jeweilige Individuum zutreffen:

- Gewicht
- Alter
- Geschlecht
- Chronotyp
- Thermische Empfindlichkeit
- Linsentrübung
- Körperoberfläche

Tragbare Sensorik

Dieser Bereich deckt alle Methoden ab, welche Daten am und im Nutzer während des laufenden Betriebes erfassen. Es ist das Pendant zu den Sensoren im Gebäude, jedoch beziehen sich gemessene Werte hier direkt auf den Nutzer:

- Hauttemperatur
- Puls
- Blutzuckerspiegel
- Bekleidungsgrad
- Stimmung
- Hormonspiegel
- Schlafzustand

Aktivitäten

Dieser Bereich bildet alle Aktionen ab, die der Nutzer tätigen kann. Dadurch ist es möglich, Aktionen, die das Gebäude nicht selbst veranlassen kann, durch den Nutzer ausführen zu lassen. Implementierte Module müssen dafür jedoch die notwendigen Aktionen integrieren und verarbeiten können, nur dann ist das Gebäude auch in der Lage diese Möglichkeiten zu nutzen. Im Fall einer möglichen Strategie gegen eine drohende Schimmelpilzbildung wäre als Aktion das manuelle Lüften durch den Nutzer empfehlenswert. Wie diese Aktion anzusteuern ist, kann im Gebäude jedoch nur umgesetzt werde, wenn das entsprechende Modul dies vorher kommuniziert hat. Beispiele für mögliche Aktivitäten sind:

- Fenster öffnen
- Türen öffnen
- Geräte ein- und ausschalten
- Anwesenheit im Raum
- Ausüben von Tätigkeiten wie Kochen, Schlafen, Duschen

Basismodelle

Ähnlich wie im DGZ werden im Bereich der „Basismodelle“ alle theoretischen Modelle hinterlegt, die den Nutzer oder sein direktes Interaktionsfeld beschreiben. Auch Grundanforderungen, des Nutzers sind hier zu definieren. Diese könnten sein:

- Behaglichkeitsberechnungen bei Nutzern
- Anforderung wie zum Beispiel: keine Schimmelbildung im Raum
- Radon Expositionsklassen beim Menschen
- Berechnung der erforderlichen biologisch wirksamsten Beleuchtung zur entsprechenden Tageszeit
- Metabolismus-Bestimmung aus Pulsdaten

Ersatzdaten

Ersatzdaten sind ähnlich wie beim DGZ einen Pool an Daten oder Modulen, die solche erstellen, deren Grundlagen auf Normen und Statistiken beruhen. Diese könnten genutzt werden wenn keine passenden Daten von Sensorik oder anderen akkuraten Quellen zur Verfügung stehen, was bei technischen Defekten oder unzureichender technischer Ausstattung eines Gebäudes der Fall ist. Auch hier gilt wieder die Prämisse, dass jedes Modul für seine Ein- und Ausgangsdaten eine Aussage in Bezug auf die Verwendung von Ersatzdaten machen muss.

4.2.3 Interaktion von Nutzer- und Gebäudezwilling

Im hier vorgestellten Ansatz basiert die Interaktion des DGZ und dem DNZ basiert auf dem Konzept von Bedürfnis und Bedürfnisbefriedigung. Der Nutzer und damit im Idealfall sein DNZ, erzeugen ein „Bedürfnis“ woraufhin das Gebäude und damit der DGZ reagieren muss, um dieses zu befriedigen. Die grundlegende Systematik ist im unten abgebildeten Schema zu erkennen. Auf der linken Seite ist hier der bereits bekannte DNZ abgebildet, auf der rechten Seite der DGZ. Das Bedürfnis des Nutzers wird hier als Anforderung generiert und führt wie dargestellt zu einer Interaktion.

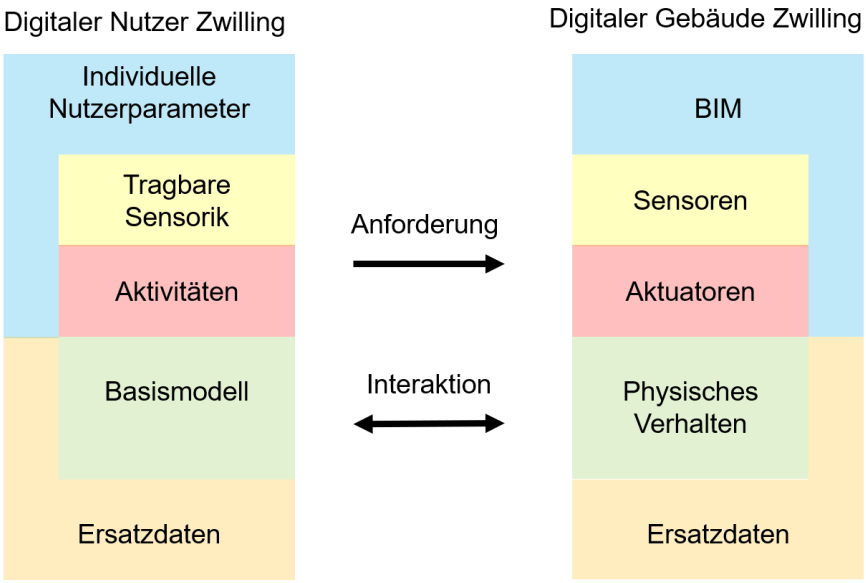


Abb. 23 Interaktionsmodell digitaler Gebäude- und Nutzerzwilling

Damit wird klar, dass der früher in der Automation übliche Funktionsansatz in Form von Modulen auf beide Zwillinge und ihre Interaktion aufgeteilt wird. Während bisher Inhalte und Daten in einem Programm und einer Berechnung integriert waren, werden diese nun auf die Bestandteile beider Zwillinge aufgeteilt, sodass ein Konzept entsteht um die Interaktion realitätsnah zu beschreiben. Damit dieses Prinzip auch interdisziplinär funktioniert, wurden in bestimmten Bereichen bereits Einschränkungen und minimale Anforderungen definiert welche eingehalten werden müssen. Hauptsächlich betrifft dies eine konsequente Aufteilung der Funktionsbereiche auf die jeweiligen Bereiche der digitalen Zwillinge, die Definition der Ersatzdaten sowie die Definition der Anforderungen im Bereich der digitalen Zwillinge.
Letztendlich wird bei der Entwicklung von Modulen für den DGZ und den DNZ auch die im Smarthome verbaute Rechenleistung der Hardware eine begrenzende Rolle spielen. Diese begrenzende Komponente wurde aber mithilfe von Cloudsystemen auch schon bei mehreren Anwendungsfällen gelöst. So bietet AutoCAD im kommerziellen Bereich und das LRZ im Bereich der Wissenschaft für viele sehr rechenaufwändige Kalkulationen Cloud Rechensysteme an, die diese Arbeit übernehmen und die Ergebnisse zurückgeben. Diese Systeme spielen daher im Bereich der Skalierbarkeit eines solchen Systems und dem modularen Aufbau eine wichtige Rolle.
Das bedeutet, dass durch Anwendung dieser Methodik der Funktionsumfang und vor allem der Interaktionsumfang der eingebundenen Module dramatisch zunimmt. Im Gegenzug erfordert es aber vom Entwickler von seiner üblichen, auf Funktionen basierenden Denkweise abzusehen und stattdessen die Funktionen in Teilbereiche des DGZ oder des DNZ zu zerlegen und dementsprechend zu implementieren. Damit werden zur Bestimmung der früheren Funktionalität im Normalfall mehrere kleine Module in den verschiedenen Bereichen notwendig, die den jeweiligen Bereich des betreffenden digitalen Zwillings beschreiben.

4.3 Modularer Aufbau

Um die durch das soeben vorgestellte Konzept entstehenden Module unterstützen zu können, ist ein modularer Aufbau der gesamten Systemumgebung erforderlich. Die dafür notwendigen Randbedingungen und Anforderungen werden nachfolgend vorgestellt.

Skalierbarkeit

Um nicht bereits im Vorfeld Beschränkungen zu erzeugen, muss das System möglichst skalierbar konzipiert sein. Daraus resultierend sollte die Zahl der zu implementierenden Module und Softwarekomponenten möglichst nicht oder nur in begrenztem Maße eingeschränkt sein. Ebenso muss Rücksicht auf die derzeit in Haushalten übliche verbaubare Hardware genommen werden. Überschreiten Module die Leistungsspezifikationen der Module deren Grenzen, dürfen sie keine Verwendung auf der hauseigenen Hardware finden. Daher sind alternative Strategien zur Integration dieser Module erforderlich.

Bei der hier vorgestellten Methodik, unter Berücksichtigung des derzeitigen Wissensstandes, bieten sich hierzu folgende Möglichkeiten mit ihren Vor- und Nachteilen an:

- Anbindung an eine Cloud
    - Vorteile: Keine eigene Hardware und Systempflege notwendig
    - Nachteile: Daten liegen auf externen Systemen. Datenschutz muss geklärt sein. Meist kommerzieller Hintergrund, wodurch weitere kosten resultieren.
- Dedizierte spezielle Hardware, die zusätzlich zum Modul beschafft werden muss.
    - Vorteile: Daten bleiben im eigenen Besitz somit kein Datenschutzproblem, Aufstellort kann frei gewählt werden (z. B.: Keller, etc.)
    - Nachteile: Hardware ist meist teuer, Wartung und Stromverbrauch erzeugt weitere Kosten, Abwärme kann in Neubauten mit sehr gutem Wärmeschutz problematisch für das Wohnumfeld werden
- Rechenintensive Aufgaben auf Zeiten niedriger Systemlast zu verteilen und die Rechenlast der jeweiligen Aufgaben begrenzen.
    - Vorteile: Keine weitere Hardware nötig, keine zusätzlichen Wartungskosten, Daten verbleiben im eigenen Gebäude, kein Datenschutzproblem.
    - Nachteile: Hardware wird dauerhaft höher ausgelastet, daraus resultiert ein schnellerer Verschleiß. Stromverbrauch steigt durch dauerhaft höhere Last am Gerät, Reserven für Erweiterung schwinden, keine Echtzeit relevanten Tasks mit dieser Methodik umsetzbar

Somit ist also erkennbar, dass die Skalierbarkeit durch die Modularisierung keine Einschränkungen erfahren darf. Als begrenzende Größe bei der Skalierbarkeit gilt aber derzeit noch die Leistungsfähigkeit der Hardware.

Interdisziplinär

Im Bereich der Bauphysik ist eine klare Abgrenzung von Handlungs- und Systemgrenzen schwierig, da viele Aktionsbereichen im Smarthome starke Überschneidungen aufweisen. Daher ist eine Möglichkeit, auch interdisziplinäre Ansätze umsetzen zu können, notwendig. Dem ersten Anschein nach ist zum Beispiel eine Interaktion zwischen der Beleuchtung und der Beheizung eines Gebäudes nicht erkennbar. Berücksichtigt man aber Studien, denen zufolge die Lichtfarbe das Temperaturempfinden beeinflusst, so ist ein direkter Bezug dieser beiden Bereiche zueinander klar definiert, obwohl diese üblicherweise keine gemeinsamen Interaktionen vorweisen. (Marggraf-Micheel, Maier, & Albers, 2011). Um diese Zusammenhänge abzubilden ist ein System nötig, welches in seiner Konzipierung sehr offen angelegt ist. Die darin enthaltenen Module müssen eigenständig die Interaktionsbereiche festlegen können statt über das beherbergende Rahmengerüst eingeschränkt zu werden. Nur dieser Ansatz ermöglicht in vollem Umfang interdisziplinär zu agieren.

Offen/Nicht abgeschlossen

Die Abbildung aller üblichen Parameter, Einfluss- und Steuergrößen, welche sich im Smarthome und seinen interagierenden Disziplinen finden, muss durch das Grundkonzept gegeben sein. Nur so kann verhindert werden, dass bereits bei der Entwicklung Einschränkungen generiert werden. Wenn daher zum Beispiel eine Begrenzung generiert wird, die besagt, dass Parameter immer in einer speziellen Form übergeben werden müssen, beispielsweise wie folgt:

Tabelle 6 Beispiel für eine Einschränkung von Übergabeparametern

Nummerischer Wert	Einheit	Varianz

So wären damit alle Systeme ausgeschlossen, die als Eingangsgröße Parameter ohne numerischen Wert verwenden oder deren Wert keine Einheit besitzt. Dies wäre zum Beispiel bei der Skala von Fanger der Fall. Diese besitzt zwar einen numerischen Wert hat aber keine Einheit. Eine Übergabe aller Daten, welcher Form auch immer, muss daher gewährleistet sein. Umgekehrt muss aber auch eine eindeutigere Zuordnung aller Daten gewährleistet sein. Da im Vorfeld nicht für jedes zukünftige Modul eine Zuordnung erfolgen kann, muss aufseiten des Moduls eindeutig definiert sein welche Daten benötigt werden. Dies geschieht im vorgestellten System durch eine eindeutige Definition der sogenannten „Inputvariablen“ oder „Inputparameter“.

Integriert und Dynamisch

Die Integration unterschiedlichster Disziplinen bringt auch die Integration vielfältiger Tools und Methoden zur Bestimmung von Daten mit sich. Eine 100-prozentige Modularisierung auf verschiedenste Disziplinen würde zu einer mehrfachen Integration von Systemkomponenten führen, da für jedes Modul eine separate Implementierung erforderlich wäre. Ein leistungsfähiges Konzept muss daher ermöglichen, dass mehrfach zur Verwendung kommende Algorithmen so integriert werden können, dass sie von verschiedenen Modulen und Subsystemen genutzt werden können. Dies beinhaltet auch ein Konzept, das gewährleistet, dass die Dynamik einer Simulation gewahrt wird und nicht von der Interaktion mit anderen Systemen, die evtl. auf einer anderen Zeitskala interagieren unterbunden wird. Grundsätzlich ist bezüglich der Modularität im Kontext eines Smarthomes in verschiedene Kategorien zu unterscheiden. Diese werden im Folgenden betrachtet.

Modularität der Hardware

Die Hardware eines Smarthome-Systems untergliedert sich in die bereits erläuterten verschiedenen Bereiche Feldebene, Automationsebene und Managementebene. Prinzipiell wäre es machbar, so viele Ebenen wie möglich in einem Gerät zu vereinen. Das hätte jedoch den gravierenden Nachteil, dass bei Ausfall eines Gerätes auch andere Komponenten mit ausfallen und dementsprechend bei Hardwaredefekten große Teile des Systems ihre Funktionsfähigkeit verlieren. Daher ist es aktuell bereits Stand des Wissens, dass die Systeme in verschiedene modulare Untersysteme getrennt werden. Üblicherweise gibt es eine zentrale Einheit für Berechnungen und logische Aufgaben, den Smarthome-Server. An diesen werden, sogenannte Interfaces, angeschlossenen, die die Kommunikation mit den jeweiligen Geräten oder Sensoren und Steuermodulen herstellen. Bei Ausfall eines der Geräte können die restlichen im Regelfall weiter ihre Funktion gewährleisten. Diese Praxis unterstützt die Zielsetzungen der hier vorgestellten Methode bereits und sollte soweit wie möglich beibehalten werden. Auch wenn zeitgleich anzubringen ist, dass die Interoperabilität immer noch eines der größten Probleme des Smarthomes ist. Derzeit werden noch viele unterschiedliche Standards vertrieben, was zu einer Vielzahl an Interfacemöglichkeiten führt. Die hier vorgestellte Methode bringt jedoch den Vorteil mit sich, dass bei fachgerechter Implementierung von Geräten und Sensoren unabhängig vom Hersteller eine richtige Ansteuerung und Verwendung der Hardware gewährleistet ist.

Auch proprietäre Betriebssysteme von Smarthome-Herstellern unterstützen nicht alle Hardwareanbieter und schränken damit die Verwendbarkeit und den Umfang des Systems ein. Diese Lücke wird durch die in dieser Arbeit vorgestellte Methodik zur Konzeption eines Smarthome-Servers geschlossen. Dadurch wird das Problem der Interoperabilität von aktueller als auch von zukünftiger Smarthome-Hardware gelöst sowie die Bereitstellung einer Interoperabilität der im Smarthomeserver verwendeten Software ermöglicht. Beim derzeitigen Stand bezüglich Lösungsangeboten um Smarthomes stellt dies ein Alleinstellungsmerkmal dar. Eine solche automatisierte Interaktion zwischen den jeweiligen Softwaremodulen kann derzeit keiner der aktuellen Automationsserver zur Verfügung stellen.

Als Vergleich für die hier vorgestellte Methode kann die Funktionsweise eines Betriebssystems bei einem Personal Computer (PC) dienen. Die fachgerechte Implementierung der Hardware beim Betriebssystem wird in Form von Treibern realisiert. Ist beim PC ein Gerät, wie beispielsweise ein Drucker, fachgerecht eingerichtet kann es von jedem Softwaremodul über eine standardisierte Schnittstelle verwendet werden. Dies ist unabhängig vom jeweiligen Hersteller des Druckers oder eines verwendeten Protokolls, möglich.

In folgender Tabelle sind die üblichen Aufgaben eines Betriebssystems am Computer in der linken Spalte aufgeführt die dazu passenden Pendants im Smarthome und welche Komponente diese übernehmen in den rechten zwei Spalten aufgeführt:

Tabelle 7 Vergleich Aufgaben Betriebssystem Smarthome (ersten Spalte nach Plate, 2019)

Aufgaben eines Betriebssystems am Computer nach Plate (2019)	**Aufgabenpendent im Smarthome**	**Aufgabenerfüllender Part im Smarthome**
Verbergen der Komplexität der Maschine vor dem Anwender	Nutzerinterface, Visualisierung	Frontend, eigenständige Software
Bereitstellen einer Benutzerschnittstelle	Kommandozeile zur Steuerung einzelner Geräte	Automationsserver
Bereitstellen einer normierten Programmierschnittstelle (API)	Koordination der Datenflüsse und der Interaktion der einzelnen Komponenten	Die hier vorgestellte Methode
Verwaltung der Ressourcen der Maschine	Verwalten und Steuern der im Smarthome integrierten Geräte und Sensoren	Die hier vorgestellte Methode
Verfolgung von Schutzstrategien bei dieser Ressourcenbereitstellung	Verwalten der Prioritäten bei der Entscheidungsfindung zur Steuerung der verschiedenen Strategien	Die hier vorgestellte Methode
Koordination von Prozessen	Daten auslesen, Schnittstellen bedienen	Automationsserver

Das Smarthome kann mithilfe der Abstraktion des Maschinenbegriffs nach Coy (1992), wie folgender Tabelle zu entnehmen ist, beschrieben werden.

Tabelle 8 Vergleich des Maschinenbegriff nach Coy (1992) mit Computer und Smarthome

Maschinenbegriff	**Computer**	**Am Beispiel Smarthome**
Reale Maschine	Zentraleinheit + Geräte	Server + Hardware im Gebäude
Abstrakte Maschine	Reale Maschine + Betriebssystem	Reale Maschine + Automationsserver + diese Methode + Frontend
Benutzermaschine	Abstrakte Maschine + Anwendungsprogramm	Abstrakte Maschine + Module

Das Betriebssystem bildet also die Schnittstelle zwischen Hard- und Software. Es ist erkennbar, dass die hier vorgestellte Methode mit einem geeigneten Frontend alle Aufgaben eines Betriebssystems am Beispiel eines Computers übernehmen kann. Auf die Notwendigkeit einer Bedienoberfläche wird in Kapitel 8.2 noch einmal eingegangen. Im weiteren Verlauf dieser Arbeit wird dieses Thema jedoch nicht näher bearbeitet.

Der zweite Bereich, in der die Modularität eine Rolle spielt, ist die Software. Diese wird folgend näher erläutert.

Modularität der Software

Es wird auf die Ausführung verschiedener Modularitäten bei Automationsserver und der Anbindung von Hardware und deren Funktionen und Protokollen in dieser Arbeit verzichtet, da diese nicht Teil der hier vorgestellten Methodik sind. Vergleichbar wäre dieser Teil mit der Entwicklung und Einbindung von Gerätetreibern. Es sei aber darauf aufmerksam gemacht, dass natürlich auch auf diesem Level der Software eine Modularisierung große Vorteile bringt.

Die Modularität, die im Bereich der hier vorgestellten Algorithmik angestrebt wird, betrifft vor allem die Bereitstellung von Smarthome-Funktionen und deren Untergliederung in Teile des DNZ und DGZ sowie der Sicherstellung ihrer reibungslosen Interaktion. Da die hier als Module bezeichneten Teile der klassischen Funktionen im weitesten Sinne starke Ähnlichkeiten mit Simulationen oder Simulationsmodellen aus der Bauphysik aufweisen, ist ein Vergleich der Modularisierungsstrategien aus der Simulationsdomäne naheliegend. Die bereits beschriebenen Ebenen der Modularität in die bei Softwaresystemen unterschieden werden kann, werden im Folgenden in Bezug auf die vorgestellte Methode kurz bewertet.

- **„Functional Layout Modularity“**
  Hierbei wird auf die Modularität bei der Interaktion mit dem Nutzer und dessen Auswahl einzelner Programmbausteine Wert gelegt. Diese Art der Modularität ist am besten mit dem aktuell gebräuchlichen Ansatz der Funktionen in Automationsservern kompatibel. Diese können einzeln hinzugefügt oder entfernt werden. Um jedoch „Functional Layout Modularity“ gewährleisten zu können, muss eine Übersicht existieren, die den Zusammenhang der einzelnen Module auf DNZ und DGZ Ebene mit den Funktionen des Smarthomes visualisieren und verwalten kann. Damit ist sichergestellt, dass Abhängigkeiten bei Entfernen oder Hinzufügen einer Funktion im klassischen Sinne der Automatisierung berücksichtigt werden. Beim Vergleich mit dem Computerbetriebssystem handelt es sich hierbei um das Installieren und Deinstallieren von Software. Bei Entwicklung des bereits angesprochenen geeigneten Frontends, welches in Kapitel 8 noch einmal behandelt wird, kann die hier vorgestellte Methode diese Form der Modularisierung bieten.

- **„Mathematical Models Modularity“**
  Diese Form der Modularisierung ist das Grundkonzept der hier beschriebenen Methode. Zwar werden bauphysikalische Zusammenhänge nicht in Form von so genannten Bibliotheken erstellt, eine Kapselung der physikalischen Zusammenhänge erfolgt dennoch in Form der verschiedenen und im Folgenden noch genauer beschriebenen Module. Um diese zu berechnen ist kein Wissen über deren Inhalt erforderlich, was das Hauptkriterium für eine erfolgreiche Kapselung ist.

- **„Standardized Mathematical Models Modularity“**
  Diese Methode erfüllt grundsätzlich die Anforderungen an die standardisierte mathematische Modularität. Die Eignung der in der hier vorgestellten Methode zum Einsatz kommenden ontologiebasierten Algorithmen, zur Kopplung von Simulationen, wurde bereits von Mitterhofer (2017) nachgewiesen. Daher wird im weiteren Verlauf auf diesen Aspekt nur oberflächlich eingegangen. Grundsätzlich ist jedoch sowohl bei der Modulentwicklung als auch bei der gesamtheitlichen Umsetzung darauf zu achten, dass numerische Simulationen auch über die Modulgrenzen hinaus korrekt erfolgen.

- **„Code’s Modularity“**
  Wenn die hier vorgestellten Module als Codeteile betrachtet werden, ist also eine Modularität des Gesamtcodes gegeben. Module können nach der Einbindung im Weiteren von allen anderen Modulen verwendet werden. Ist beispielsweise ein Modul zur Berechnung der Oberflächentemperaturen von Außenwänden implementiert, so können andere Module dieses zur Informationsermittlung verwenden. Eine Wiederverwertbarkeit einzelner Codezeilen ist jedoch nicht gegeben. Teile innerhalb der Module können daher nicht wiederverwendet werden. Dies hat aber auch den Hintergrund, dass der Code innerhalb von Modulen bei verschiedenen Entwicklern Firmengeheimnisse enthalten kann und damit gar keine übergreifende Wiederverwendbarkeit gewünscht ist.

Verkoppeln von modularen Simulationen

Grundsätzlich wird bei richtiger Konfiguration des Systems eine übergreifende Kalkulation aller an einem Problem beteiligten Komponenten durchgeführt. Dabei ist irrelevant, aus welcher Domäne sie stammen. Dies geschieht jedoch automatisch und ist grundlegend davon abhängig, wie granular Funktionen als Module, im hier vorgestellten Kontext, integriert werden. Je mehr Kalkulationen in einzelne Module implementiert werden, desto höher wird auch die Möglichkeit Ressourcen domänenübergreifend zu nutzen.

Die gewählte Kopplungsstrategie spielt für die vorgestellte Methodik nur für die Auswahl der zur Berechnung erforderlichen Zeit- und Iterationsschritte eine Rolle. Daher ist keine Auswirkung auf die grundlegende Funktion der hier gezeigten Methode zu erwarten. Unabhängig davon ist jedoch für einen produktiven Einsatz ein Framework notwendig, welches diese Funktion erfüllen kann. Das so ein Framework grundsätzlich möglich ist, hat Mitterhofer, wie bereits erläutert, bewiesen. (Mitterhofer, 2017)

Modellierungskonzept der digitalen Zwillinge

Im Umgang mit Input- und Outputdaten bei Simulationen existieren verschiedene Modelle. Im Folgenden wird für den Bereich der Bauphysik deren Übertragbarkeit zur Modellierung der digitalen Zwillinge im Smarthome untersucht.

Stand-Alone Ansätze und Data-Model-Interpretation Ansätze wirken den Zielen der Einführung der digitalen Zwillinge entgegen und können daher ausgeschlossen werden, weil sie keine Skalierbarkeit und Flexibilität bieten.

Ein Process-Model-Interpretation Ansatz würde bedeuten, ein Grundsystem zu entwickeln und darauf basierend den Funktionsumfang zu erweitern. Diese Herangehensweise steht in Konflikt mit dem Ziel der interdisziplinären Verwendbarkeit.

Somit ist der einzige Ansatz, der aus der Gebäudesimulation übernommen werden kann, die sogenannte „Process-Model-Cooperation“. In dieser kommunizieren unterschiedliche Simulations- und Recheneinheiten miteinander und tauschen während der unterschiedlichen Berechnungszeitpunkte Daten untereinander aus. Daher wurde für die Entwicklung der Methode ein Process-Model-Cooperation Ansatz gewählt.

Um sicherzustellen, dass die bereits erwähnte Menge an kleinen Modulen an den richtigen Schnittstellen miteinander verknüpft werden und miteinander interagieren, bedarf es eines entsprechenden Konzeptes. Dieses muss die Verkoppelung und Interaktion der Module sowie eine bedingte funktionale und eine standardisierte Mathematische Modularität gewährleisten.

Da nun die grundlegenden Anforderungen an die Modularisierung im System gestellt sind, kann im Folgenden dazu übergegangen werden, die jeweiligen Modultypen und deren Anforderungen zu definieren und in das Gesamtkonzept einzugliedern.

4.3.1 Konzipierung der Module

Für die Modulkonzipierung wurden folgende Randbedingungen aufgestellt:

- Standardisierte Mathematische Modularisierung
- Process Model Cooperation
- Der Aufbau eines DGZ und eines DNZ
- Hohe Modularität
- Integration von Sensorik und Datenstrukturen als Eingangsdaten

Auf Basis dieser Parameter bietet es sich an Module so zu konzipieren, dass ihre Interaktion auf Basis ihrer Ein- und Ausgangsdaten erfolgt. Dazu ist ein Abgleich der vorhandenen Eingangsgrößen mit den je Modul verfügbaren Inputvariablen notwendig. Die Ausgangsvariablen, die ein Modul generiert, können somit anderen Modulen wieder als Eingangsvariablen dienen. Die folgende Abbildung zeigt diesen Aspekt.

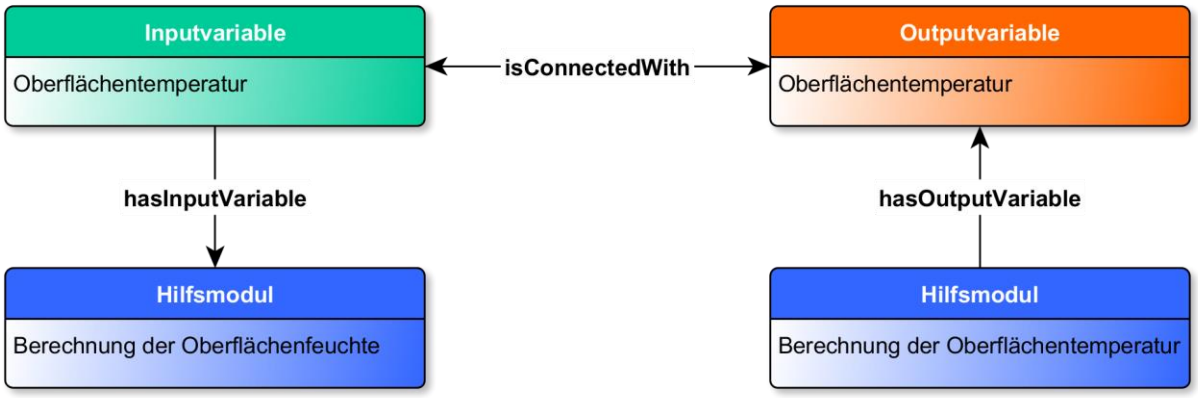


Abb. 24 Verknüpfung von Inputvariablen und Outputvariablen verschiedener Module

Schnell wird dabei klar, dass auf diese Weise Zirkelbezüge möglich sind, die so im Hausautomationskontext nicht gewünscht wären. Dieses Phänomen visualisiert Abb. 25.

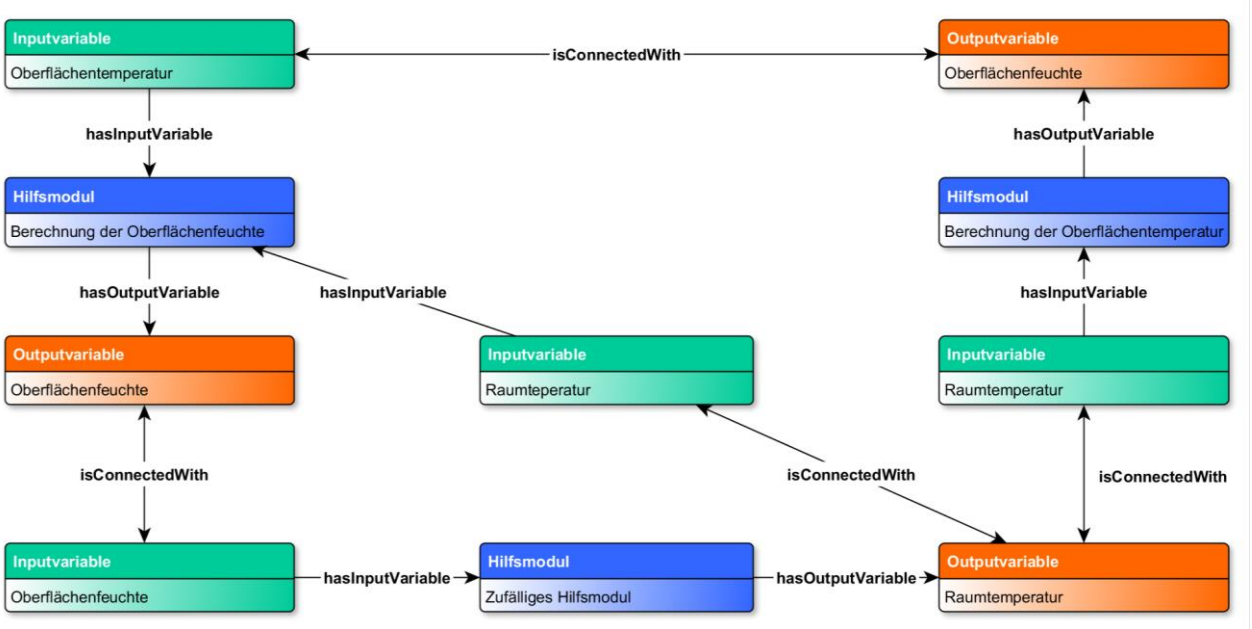


Abb. 25 Beispiel für unerwünschte Zirkelbezüge

Hier wird eine Inputvariable mit einer Outputvariable verbunden die letztendlich in ihrer Entstehung Daten aus dem gleichen Vorgängermodul bezieht und damit einen Zirkelschluss erzeugt. Um dieses Phänomen zu vermeiden, werden die Module des hier vorgestellten Ansatzes noch bezüglich ihrer Verwendung getrennt. Das kommt auch dem Konzept der digitalen Zwillinge entgegen. Dadurch sind Zirkelbezüge nur noch zwischen DGZ und DNZ möglich, da die Berechnungsrichtung durch die Interaktion beider Zwillinge genau vorgegeben ist.

Im realen Leben ist dieser Zusammenhang dadurch zu beobachten, dass ein Nutzer ein Bedürfnis hat und versucht das Gebäude so einzustellen, dass dieses Bedürfnis befriedigt wird. Ein Zirkelbezug zwischen Nutzer und Gebäude ist damit genauso wenig möglich wie zwischen den digitalen Zwillingen. Um Zirkelbezüge zwischen DGZ und DNZ zu vermeiden ist also eine weitere Kategorisierung der Module notwendig. Diese erfolgt in Funktionsmodule und Helpermodule.
Funktionsmodule bilden eine essenzielle Funktion entsprechend des derzeit üblichen Funktionsansatzes ab und sind in der Lage auf ein Nutzerbedürfnis zu reagieren. Helpermodule dienen ausschließlich dazu die hierbei notwendigen Daten zu beschaffen. Bei einer grafischen Darstellung der Datenströme wird daher als Regel definiert: Der Datenfluss darf pro Zeitschritt nur einmal ein Funktionsmodul passieren. Somit können auch Zirkelbezüge innerhalb der Zwillinge verhindert werden. Prinzipiell heißt das aber nicht, dass Ausgangsdaten eines Moduls nicht auch als Eingangsdaten eines anderen Moduls agieren dürfen.
Um das zu gewährleisten ist ein Datenspeicher notwendig. Dieser wird üblicherweise in Form einer Datenbank bereitgestellt, in der die Daten abgelegt werden. Um die richtigen Informationen entsprechend der Modellspezifikationen aus der Datenbank auszulesen und aufzubereiten, wird ein Datensammlermodul benötigt. Dieses stellt auch sicher, dass die Daten in der richtigen zeitlichen Auflösung und im richtigen Format an die Module übergeben werden.
Um nach der Bearbeitung der Bedürfnis-Interaktion der Zwillinge ein richtiges Gerät für die Ausübung der Aktion auswählen zu können, bedarf es auch hier eines speziellen Moduls, das eine Verbindung zwischen Gerät und Wirkungspotential eines Gerätes erstellt. So ein Wirkmodul ist beispielsweise in

der Lage die Verbindung zwischen dem Öffnungsstand von Fenstern und dem daraus resultierenden Luftwechsel herzustellen. Auch können diese Module einen Zusammenhang zwischen Aktionen eines Zwillings und resultierenden zukünftigen Veränderungen physikalischer Größen im Gebäude herstellen. Damit bilden sie auch Prognosefunktionen ab, die in iterativen Berechnungen als Eingangsgrößen genutzt werden können. Man kann diese Module auch als Beschreibung der Wirkseite des Systems bezeichnen.

Eine grundsätzliche Unterteilung in Modultypen muss daher im Sinne der hier vorgestellten Methode wie folgt geschehen. Ein Modul ist einer der beiden Kategorien Nutzer- oder Gebäudemodule zuzuordnen. Danach kann weiter unterschieden werden in:

- Funktionsmodule
- Hilfsmodule
- Datensammlermodule
- Wirkmodule
- Module für maschinelles Lernen und Vorhersagen

Modulaufbau

Um später mit Hilfe von Reasonern Interaktionen zwischen Modulen finden zu können, müssen Rahmenbedingungen und Schnittstellen definiert sein, auf deren Basis eine Ontologie erstellt werden kann. Eine solide Basis schafft dabei die standardisierte mathematische Modularisierung. Durch sie ist es möglich Module auf eine einheitliche Basis der Beschreibung zurückzuführen. Als Schnittstelle für Module sind sogenannte In- und Output-Datenströme definiert.

Input Datenströme (IDS) bezeichnen dabei Daten, die den Modulen als Start- oder Eingangsvariablen dienen. Es wird vom IDS gesprochen, weil dies dem Weg der Daten in ein Modell bei einer grafischen Darstellung entspricht. Im Bereich der Physik und Mathematik werden IDS meist durch Variablen oder Parameter repräsentiert. Grundsätzlich ist es aber möglich, Datenströme durch jede erdenkliche Form von Information zu repräsentieren. Wichtig ist lediglich, dass es sich um ein Datenformat handelt, das vereinheitlicht werden kann.

Output Datenströme (ODS) bezeichnen daher im Umkehrschluss die Informationen, die ein Modul als Ergebnis wieder ausgibt. Bei den ODS von Funktionsmodulen ist besonders auf das spezielle Format zu achten, in dem die Daten ausgegeben werden müssen. Welche Daten dies genau sind, hängt stark von der speziellen Funktion des jeweiligen Moduls ab. Die folgende Abbildung verdeutlicht den Zusammenhang zwischen Modulen, IDS und ODS noch einmal.

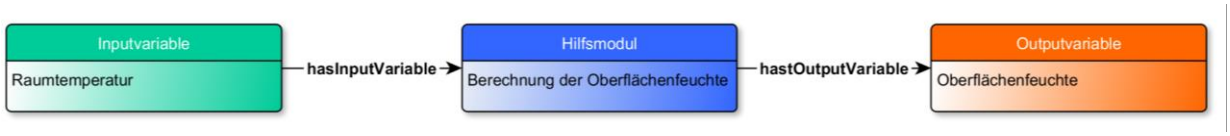


Abb. 26 Abbildung Zusammenhang IDS und ODS bei Modulen

Das Wissen über die jeweiligen IDS und ODS eines Moduls muss innerhalb einer Ontologie abgelegt werden. Dadurch wird diese Information für einen Reasoner lesbar. Um diese Initialisierung durchzuführen ist jedoch bei der ersten Verwendung des Moduls ein Skript erforderlich, das diese Informationen sowie das Modul selbst in die Ontologie implementiert. Konkret muss dieses Skript in der Ontologie die Individuen der einzelnen Module und Datenströme mit deren Eigenschaften anlegen. Auch die Eingliederung der Module zu den jeweiligen Klassen der Modultypen muss in diesem Schritt erfolgen sowie das Initialisieren aller für die Funktion der Ontologie mit dem Modul in

Verbindung stehenden Daten. Im Fall von mehreren Nutzern, Gebäudeteilen, Räumen oder Objekten, auf die sich die Funktion des Moduls bezieht, ist eine mehrfach Instanziierung der Module notwendig. Das bedeutet, dass für jedes Modul bei der neuen Instanziierung anderer Bestandteile innerhalb der Ontologie geprüft werden muss, ob zusätzliche Instanzen erforderlich werden. So müsste durch das Hinzufügen eines Nutzers im Gebäude geprüft werden, welche Module für diesen eine zusätzliche Instanziierung benötigen. Damit ist die grundsätzliche Funktionalität dieser Methodik für mehrzonale Berechnungen in jeder Form, in Verbindung mit einer Ontologie, gegeben. Dass diese Rechnungen durchgeführt werden können ist bereits von Mitterhofer (2017) nachgewiesen worden.

Bei den physikalischen Problemen, die im Rahmen dieser Arbeit untersucht werden, existiert nur selten ein Lösungsweg. Meist gibt es mehrere Wege, um mit solch einem Problem erfolgreich umzugehen. Damit diese Varianten möglichst detailliert wiedergegeben werden können, müssen Module verschiedene Versionen der beinhaltenden Algorithmik unterstützen, um die Ergebnisse in verschiedenen Varianten auszugeben. Um die Problematik von Schimmelpilzbildung zu vermeiden sind beispielsweise mehrere Lösungsstrategien möglich. Die erste ist den Luftwechsel zu erhöhen. Eine weitere die Raumtemperatur zu erhöhen. Um beide Lösungswege zu berücksichtigen muss das System also in verschiedenen Varianten rechnen und eine Auswahl treffen.
Daher erfolgt im Rahmen des hier vorgestellten Ansatzes eine Unterteilung eines Moduls in Versionen. Eine vollständig getrennte Instanziierung der verschiedenen Varianten als eigene Module wäre für das spätere Zuordnungen einer Handlung hinderlich, da nicht mehr identifiziert werden könnte, ob zwei Ergebnisse zu unterschiedlichen Modulen und dadurch zu unterschiedlichen Bedürfnisanforderungen gehören oder dasselbe Problem zu lösen versuchen.

Im derzeitigen Stand der Methode schließt dies jedoch auch eine Mischung von Lösungswegen, welche in einigen Szenarien sinnvoll wäre, aus. Die hier generierte Methodik lässt es zu, dass über Optimierungsalgorithmen die beste Lösung in Form einer Mischung von Moduloutputs gefunden und als zusätzliche Modulvariante im Zuge der Lösungsfindung angeboten wird. Dieser Vorgang könnte zum Beispiel in Form eines iterativen Verfahrens geschehen und benötigt daher dieselben Anforderungen wie beim zeitschrittweisen Datenaustausch von Informationen bei Simulationsmodellen.

Im Folgenden werden nun die speziellen Eigenschaften der verschiedenen Modultypen näher definiert.

4.3.2 Gebäudemodule

Die Gebäudemodule dienen dem Zweck Informationen und Zusammenhänge im Gebäude zu repräsentieren und durch ihre Interaktion den DGZ abzubilden. Das bedeutet, dass alle Module, die in irgendeiner Form Informationen über das Gebäude liefern oder aufbereiten in diese Kategorie an Modul eingegliedert werden müssen. Das können Module aus der Kategorie des „Building Information Modeling“ (BIM) sein, die Informationen zu Gebäudegeometrie oder Baustruktur liefern oder auch aus dem Bereich des physischen Verhaltens, die das Verhalten des Gebäudes abbilden. Kennzeichnend für Module des DGZ ist, dass sie Daten aktiv verarbeiten. Beispielsweise wären dementsprechend Module im Bereich der Ersatzdaten denkbar die anhand von wenigen Sensordaten mit Hilfe von Normen und Statistiken weitere Daten über das Gebäude zur Verfügung stellen. So wäre ein Modul denkbar, das mithilfe des Baualters und des Bauortes, auf Basis statistischer Auswertungen, eine Aussage über die Qualität der Baustruktur und der anzunehmenden Außenwandparameter trifft.

Informationen über Sensoren sowie Module, die aus Sensorinformationen weitere Gebäudeinformationen erzeugen, zählen hier ebenso dazu, wie Informationen über die Aktuatorik und deren Position und Funktion. Eine exemplarische Auflistung von Informationen, die in Gebäudemodulen verarbeitet werden können, findet sich in Tabelle 9.

Die Schnittstelle, an der die Grenze des DGZ festgemacht werden kann, wird durch den Nutzer erzeugt. So ist die physikalische Position des Nutzers noch dem DGZ zuzurechnen, da sich der Nutzer ähnlich der Einrichtung im Gebäude befindet. Beschreibungen über den Nutzer selbst sind jedoch durch den DNZ abzubilden. Sollte ein Nutzerverhalten spezielle Auswirkungen auf das Gebäude selbst haben, so ist deren Gebäudeauswirkung jedoch wieder beim Gebäude abzubilden. Eine klare Trennung der Zuordnung kann folglich nur über die Betrachtung der Wirkung erfolgen. Der Nutzer ist im Konzept also eher mit der Funktionalität von Einrichtungsgegenständen zu vergleichen. Diese können einen eigenen digitalen Zwilling haben. Auswirkungen auf das Gebäude werden jedoch im DGZ verarbeitet. So ist zum Beispiel die Menge an ausgestoßenem Kohlendioxid des Nutzers, so lang er sich im Gebäude befindet, relevant für das Gebäude und wird somit bei der CO_2-Bilanz des DGZ beachtet. Der Mensch übernimmt hier lediglich die Funktion einer Quelle und liefert Informationen über das Quellverhalten. Eine Beachtung erfolgt in der im Folgenden noch beschriebenen „One-to-Many“ Verbindung und eine geeignete Zuordnung über die Ontologie. Bei richtiger Modulkonzipierung ist die Information über den CO_2-Output des Nutzers als ODS definiert. Ein Gebäudemodul zur Berechnung des CO_2-Gehalts, auf Basis der anwesenden Personen, würde demnach durch die Zuordnung über das Reasoning der Ontologie die CO_2-Ausstöße der anwesenden Personen addieren.

Im Folgenden werden Beispiele für Gebäudedaten und -module aus verschiedenen Bereichen des DGZ aufgelistet.

Tabelle 9 Beispielhafte Gebäudedaten und Module des DGZ

DGZ-Bereich	**Daten/Modulfunktion**	**Beispiel**
BIM		
	Raumgeometrie	Koordinaten
	Baustruktur	Materialien
	Raumverbund	Lageplan
	Raumzweck	Wohnzimmer
	Wohn- und Nutzbereiche	Essbereich
	Sensorik und Aktuatorik Verortung	Position und Montage Punkt
Sensorik		
	Raumsensoren	Raumtemperatur
	Außensensoren	Außenluftfeuchte
	Wetterdaten	Windgeschwindigkeit
	Externe und Prognosedaten	z. B.: Wettervorhersage
	Weiterentwicklung von Messdaten	Berechnung der Oberflächentemperatur
	Gas und Umwelt Sensorik	CO-Messung
	Standorterfassung und Personenerkennung	Personenstandorte
Aktuatorik		
	Klassische Aktuatorik	Lichtaktuatoren (On/Off)
	Feedback System	Akustische oder visuelle Nachrichten, Text, etc.
	Brand spezifische und nicht Brand spezifische Anlagentechnik	Brandmelder, Heizungsanlage
Physisches Verhalten		
	Schimmelbildung im Gebäude	Bestimmung der Zeit bis zur Keimung
	Reaktion des Raums auf thermische Quellen	Änderung der Raumtemperatur durch Aktivieren eines elektrischen Gerätes
	Reaktion des Raums auf Senken	Änderung der Raumtemperatur durch Öffnen eines Fensters
	Berechnung von Oberflächentemperaturen	Wärmebrückentemperatur
	Neuronale Netze für gelernte Raumreaktionen, die noch nicht beschreibbar sind	Reaktion der Luftfeuchte auf das Öffnen von ungewöhnlichen Fenstertypen
	Beleuchtung	Reduzierung der künstlichen Beleuchtung durch Kalkulation der Tageslichtversorgung
Ersatzdaten		
	Alle Daten in Form von Statistiken, Normwerten oder ähnlichem	U-Werte alter Bauteile

4.3.3 Nutzermodule

Nutzermodule stellen, ähnlich wie die Gebäudemodule, erweiterte Informationen zum Nutzer bereit. In ihrer Gesamtheit erzeugen sie den DNZ. Dieser ist, wie bereits erwähnt, in mehrere Bereiche gegliedert. Zu jedem dieser Bereiche sind Module möglich. Diese Module dienen im hier vorgestellten Ansatz dem Zweck, den Nutzer detaillierter und vor allem individueller beschreiben zu können und dessen Daten so aufzubereiten, dass sie im Rahmen der Ontologie verarbeitbar sind. Im Bereich „Individueller Nutzerparameter“ wären hier hauptsächlich Algorithmen des maschinengestützten Lernens zu nennen. Diese können auf Grundlage vorhandener Daten individuelle Daten des Nutzers und individualisierte Parameter für Berechnungen erzeugen. Aber auch Module, die Daten aus anderen Plattformen wie „Social Media“ oder „Healthcare Produkten“ integrieren, wären in diesem Bereich zu nennen.
Im Bereich der tragbaren Sensorik werden alle Daten erfasst, die sich mit dem speziellen Individuum befassen, dass der DNZ repräsentiert. Hier kommen auch Module zum Einsatz, die Informationen zu Körperfunktionen wie Herzrhythmus und Blutdruck erfassen, aber auch Daten wie der Blutzuckerspiegel oder der aktuelle Gemütszustand wären denkbar. So wäre ein Modul zur Erfassung des Gemütszustandes aufgrund von Videoauswertungen des Gesichtsausdrucks des Nutzers in diesem Bereich möglich.
Besonderheiten sind hier, ähnlich wie beim Gebäude, durch die mehrfache Instanziierung zu beachten. So müssen bei Anwesenheit mehrerer Nutzer sowohl Daten als auch Module für jeden Nutzer einzeln instanziiert und auch berechnet werden. Nur so kann eine Individualisierung für jeden bewohnenden Nutzer sichergestellt werden. Dies betrifft nahezu alle Bereiche des DNZ. Vor allem für Aufgaben des maschinengestützten Lernens sind große Mengen an Daten erforderlich, um hohe Genauigkeiten bei den Lernvorgängen zu erzielen. Für einige Daten können diese Mengen an Informationen vermutlich erst nach längerer Zeit vorliegen.
Beispielhaft kann hier für die Menge der Ereignisse die Häufigkeit eines Duschvorganges berechnet werden. Bei einer durchschnittlichen Duschhäufigkeit von einmal am Tag, die 2006 bei 65,5 % der Deutschen vorlag (Statista - Das Statistik-Portal, 2019), würde man für ein Lerndatenset von minimal 10.000 Ereignissen ca. 27 Jahre benötigen. Was bei einer durchschnittlichen Lebenserwartung von 75 Jahren bei Männern und 82 Jahren bei Frauen, bei einer Geburt im Jahr 2018, abzüglich der Zeit als Kind, nahezu die Hälfte eines Menschenlebens beträgt (Deutsche Stiftung Weltbevölkerung, 2018).
Ein üblicher Deep-Learning Algorithmus scheidet also zur Erkennung dieser Tätigkeit auf Basis angelernter Messdaten aus. Da Lernaufgaben mit bedeutend geringeren Datenmengen als 10.000 Sample Daten jedoch auch in anderen Bereichen häufiger vorkommen, wurden Methoden entwickelt, um auch für solche Lernaufgaben Möglichkeiten zur Verfügung zu stellen. Zum Einsatz kommen dann sogenannte transferlernenden Algorithmen.(Mathworks, 2018) Bei diesen Verfahren werden bereits auf den Durchschnittsfall trainierte Modelle durch eine deutlich kleinere Zahl an Trainingsdatensätzen auf den individuellen Fall angepasst. Ein alltägliches Beispiel wäre das Anlernen der eigenen Stimme an einen Sprachassistenten, wie Siri von Apple, durch das Einsprechen vorgefertigter Satzbausteine. Viele individualisierte Daten können jedoch vor allem im Bereich der Bauphysik durch Nutzen des spezifischen Wissens generiert werden und stellen daher eine brauchbare Alternative zur Verwendung von Deep-Learning Algorithmen dar. So kann, wie im Fall des in Fallstudie 2 gezeigten Beispiels, die Klassifizierung von Duschereignissen auch durch einfache Rechenoperationen, wie der Untersuchung des Verlaufs der relativen Luftfeuchte, erfolgen. Zusätzlich kann eine Kombination von Algorithmen generiert werden und universell generierte Algorithmen des maschinellen Lernens mit Hilfe dieser Daten individualisiert werden. Eine Auflistung denkbarer Bestandteile der einzelnen Bereiche, in denen Nutzermodule vorzufinden sind, findet sich in folgender Tabelle.

Tabelle 10 Beispielhafte Daten und Module des DNZ

DNZ-Bereich	**Daten / Modulfunktion**
Individuelle Nutzerparameter	
	Nutzerpräferenzen - Persönliche Bekleidungsparameter - Prognostizierte Handlungen - Krankheiten - Duschereignisse - Duschprognose
	Alter
	Geschlecht
	Herzrate
	Blutdruck
	Schlafanalyse
	Augen-Linsentrübung
	Hauttemperatur
	Persönliche Bekleidungsparameter
	Prognostizierte Handlungen
	Krankheiten
	Duschereignisse (Vergangenheit)
	Duschprognose
Tragbare Sensorik	
	Kleidung mit Sensorik
	Smartwatch
	Fitness Tracker und Apps
	Blutdruckmessgeräte
	Waagen
	Schlafsensoren
Aktivitäten	
	Positionsbestimmung
	Aktivitätsbestimmung
	Handlungsparameterbestimmung
Basismodelle	
	Beleuchtung
	Behaglichkeit
	Schimmel
Ersatzdaten	
	Durchschnittsgewicht eines Nutzers
	Durchschnittliche Hautoberfläche eines Nutzers
	Durchschnittliches Duschverhalten
	Durchschnittliche Duschzeit und Temperatur
	Weitere Daten aus Normen und Statistiken
	Standard Daten für Metabolismus

4.3.4 Unterscheidung in Modultypen entsprechend ihrer Funktion

Nachdem nun alle Module in Gebäude- und Nutzermodule unterteilt sind benötigt die Methode die bereits erwähnten weiteren Modulunterscheidungen. Dadurch sollen unerwünschte Zirkelbezüge ausgeschlossen werden und sichergestellt sein, dass die hier beschriebene Interaktion des DGZ und des DNZ ablaufen kann. Zu diesem Zweck erfolgt eine weitere unabhängige Unterscheidung in Modultypen. Diese wären:

- Funktionsmodule
- Hilfsmodule
- Datensammlermodule
- Module für maschinelles Lernen und Vorhersagen
- Wirkungsgrößenmodule

Diese Modultypen werden im Folgenden näher definiert.

4.3.5 Funktionsmodule

Der Name des hier entwickelten Funktionsmoduls (FM) ist abgeleitet von dem derzeit in der Automation üblichen funktionsbasierten Ansatz. Daher werden in Funktionsmodulen Kernfunktionen abgebildet, welche derzeit als Funktionen in der Automation realisiert werden. Um das näher zu erläutern, wird als Beispiel die Bestimmung der Gefahr von Schimmelbildung im Innenraum angeführt. Bei einem derzeit in der Automation üblichen Funktionsansatz würde die Berechnung der Gefahr einer Schimmelbildung mit dem Isoplethenmodell wie folgt ablaufen:

1.) Festlegung, welche Aktionen in der Automation die Berechnung neu anstoßen sollen (individuell)
2.) Bestimmen, welche Sensordaten vorhanden sind (individuell)
3.) Umrechnen der Sensordaten auf die für den Algorithmus benötigten Werte (individuell)
4.) Berücksichtigen der vom Sensor vorgegebenen Aktualisierungsintervalle (individuell)
5.) Berechnung der Schimmelgefahr mit Hilfe des Isoplethenmodells
6.) Implementierung einer für das individuelle Gebäude möglichen Abwehrstrategie (individuell)
7.) Programmieren der für diese Abwehrstrategie nötigen Aktionen in Abhängigkeit der Ergebnisse aus 4.) (individuell)
8.) Steuern der Gegenmaßnahmen (sofern erforderlich)

In den hier gezeigten Methoden enthält das Funktionsmodul lediglich zwei Punkte:

- Berechnung der Schimmelgefahr mit Hilfe des Isoplethenmodells
- Berechnung einer Zeitreihe an möglichen standardisierten Handlungsanweisungen oder Wirkvorschlägen.

Die im klassischen Automationsansatz mit „(individuell)“ markierten Schritte benötigen in derzeit üblichen Systemen noch das Eingreifen eines Experten. Zumindest einmal bei der Inbetriebnahme des Systems. In der folgenden Auflistung der Schritte bei der Verwendung des hier vorgestellten Konzeptes fällt auf, dass ein Eingreifen eines Spezialisten kaum mehr erforderlich wird.

Dafür ist es nötig, die tatsächliche Kernfunktionalität einer klassischen Funktion in ein Funktionsmodul zu verpacken. Die Eingangs- und Ausgangsdatenströme dieses Moduls und den Datenströmen zugeordnete Eigenschaften der Ontologie, müssen dadurch jedoch immer standardisiert sein. Um einen IDS eindeutig zuordnen zu können, bedarf es einiger Eigenschaften, die im Folgenden am Beispiel des IDS für die relative Luftfeuchtigkeit des Raumes tabellarisch aufgelistet werden:

Tabelle 11 Notwendige Eigenschaften zur Zuordnung von Ein- und Ausgangsdatenströme bei Funktionsmodulen

Eigenschaftstyp	**Eigenschaft**
hasMedium	Air
hasQuantity	RelativeHumidity
hasLocation	Bath
hasObjectAffiliationTo	Air_Space_Bath
hasParttype	No_Parttype
isLocatedAt	Inside

Die Eigenschaften wurden hier so gewählt, damit sichergestellt ist, dass alle Informationen, die den IDS beschreiben und einzigartig, machen erfasst sind. Auf eine Abfrage der Einheit wurde für diese Bestimmung beispielsweise absichtlich verzichtet. So ist es möglich Datenströme zu verknüpfen deren Einheit nicht identisch ist. Da diese jedoch im weiteren Verlauf vom Datensammlermodul umgewandelt werden können, ist trotzdem sichergestellt, dass die Daten immer in der notwendigen Form an das Modul übergeben werden.

Ein beispielhafter ODS für das Schimmelmodul ist folgend abgebildet:

Tabelle 12 Beispiel für einen Ausgangsdatenstrom eines Funktionsmoduls

Wirkgröße	**Zielgröße**	**Logischer Operator**	**Priorität**
Luftwechsel	2 - ∞	>=	A

Prinzipiell sollen Funktionsmodule immer eine funktionale Berechnung abbilden und abschließend eine Zeitreihe an Vorschlägen zur Regulierung der Umgebung ausgeben. Sofern die Regulierung durch verschiedene Strategien erreichbar ist, so sind auch immer verschiedene Versionen des Moduls zu implementieren und die verschiedenen Regelungen in Form von ODS als Zeitreihen auszugeben. Die in dieser Arbeit getestete Variante geht davon aus, dass immer nur eine Regelungsstrategie zum Erfolg führt. In der Realität werden jedoch auch oft Mischformen zu vielversprechenden Ergebnissen führen. Über die Möglichkeiten der Implementierung von Mischformen befasst sich diese Arbeit jedoch nicht näher, da ihre Implementierung technisch möglich ist, dazu jedoch noch ein geeignetes Framework erstellt werden muss. Mehr zu dieser Thematik wird in Kapitel 8.2 erörtert. Wie zugehörige Zeitreihen aussehen können wird im Folgenden näher beschrieben. Abb. 27 zeigt eine Zusammenfassung der nötigen Daten und Informationen zur vollständigen Definition eines Funktionsmodul in diesem Kontext.

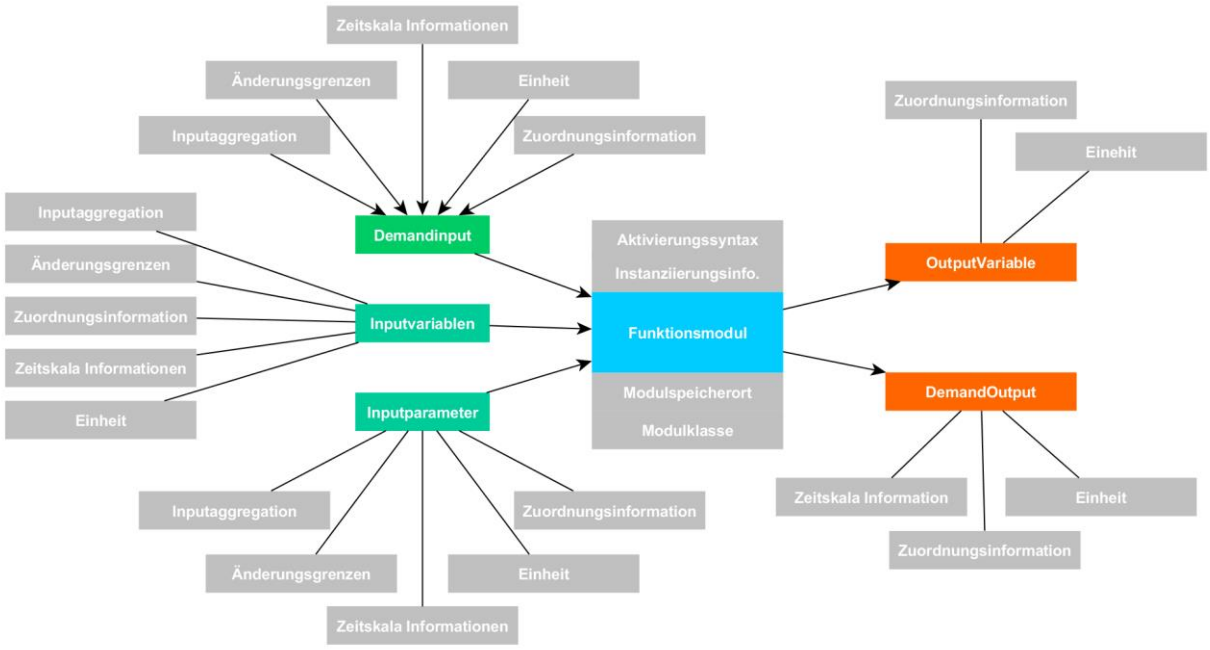


Abb. 27 Notwendige Definitionen und Informationen zur Beschreibung eines Funktionsmoduls

Ein Funktionsmodul stellt schlussendlich immer bestimmte Anforderungen an Gebäude oder Nutzer. Funktionsmodule werden in den meisten Fällen dem Bereich „Basismodell“ des DNZ oder „Physikalisches Verhalten“ des DGZ zugeordnet. Sofern eine Funktion aus dem Bereich der klassischen Automation nicht eindeutig einem Bereich, zuzuordnen ist, muss sie unterteilt werden, bis dies der Fall ist. In Bezug auf das Beispiel der Schimmelbildung ist dies intuitiv entstanden, da man davon ausgehen kann, dass die Entstehung von Schimmel vermieden werden soll. Diese Information wurde jedoch dem System am oben genannten Beispiel nicht übergeben, da diese Information zur Schimmelvermeidung erst im Basismodell des DNZ hinterlegt werden muss. Auch dies ist wie oben beschrieben in einem Funktionsmodul abzubilden, wenn auch in einem sehr einfachen, das lediglich beschreibt das kein Schimmel auftreten darf. Bei Verwendung logischer Operatoren und einer binären Beschreibung des Vorhandenseins von Schimmel wäre eine maschinenlesbare Definition dementsprechend wie folgt

$$Schimmel \neq 1$$

Als Beispiel für solch ein Funktionsmodul auf Nutzerseite ist in Kapitel 6.3 eine Fallstudie zu finden. Da das Funktionsmodul den Kern der Algorithmik abbildet, folgt eine Erläuterung zu den anderen Modultypen, welche für die Berechnungen der Funktionsmodule notwendig sind. Ebenso werden die mit „individuell“ markierten Bereiche in der vorherigen Auflistung der klassischen Berechnungsschritte erörtert.

4.3.6 Hilfsmodule

Hilfsmodule dienen, wie der Name bereits erwähnt, zur Unterstützung von anderen Modulen. In den meisten Fällen Funktionsmodulen. Sie können aus allen Bereichen des DGZ und DNZ stammen und bilden kleine Aufgaben ab, die zur Berechnung eines Funktionsmoduls dienen und als solches keine Anforderungen an den Nutzer oder das Gebäude stellen.
Ein vereinfachtes Beispiel für ein Hilfsmodul mit der Funktion einer Simulation oder einer komplexen Berechnung ist die Bestimmung von Oberflächentemperaturen im Raum mit Hilfe der

Außentemperatur, der Innentemperatur und der Wärmeleitfähigkeit des Bauteils wie sie in Fallstudie 1 erfolgt. Die schematische Gliederung der für diese Arbeit entworfenen Hilfsmodule, wird in nachfolgender Grafik dargestellt. Ihr sind die verschiedenen für die Berechnung relevanten IDS und deren verbundene Datenquellen zu entnehmen.

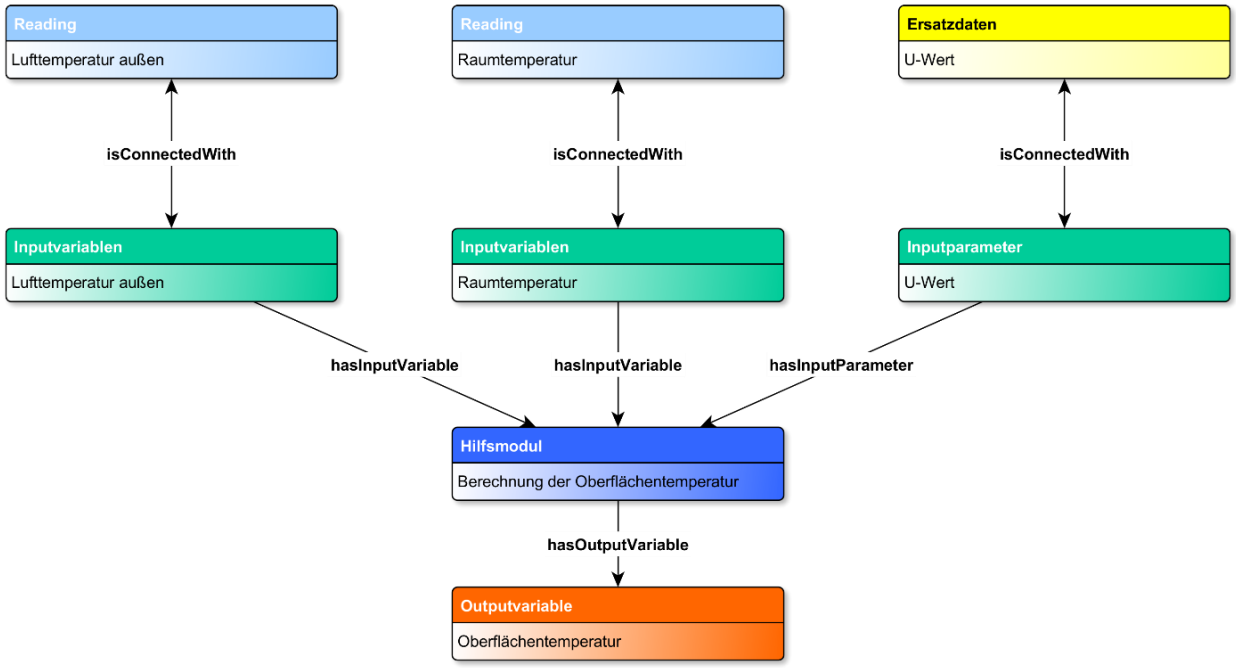


Abb. 28 Module einer vereinfachten Oberflächentemperaturberechnung

Aus dieser Berechnung allein entsteht jedoch keine Anforderung an das Gebäude oder den Nutzer. Hilfsmodulen fehlt demnach die Komponente einer Bewertung; nach der Dateneingabe werden diese strukturiert verarbeitet und wieder ausgegeben. Um bei der Berechnung eine möglichst genaue Version zu erhalten, ist es nötig bei der Zuordnung der Datenströme eine Unterscheidung zu erstellen. Das bedeutet, wenn es mehr als eine Verbindung gibt, ist prinzipiell entsprechend der Genauigkeit der Datenströme eine Auswahl zu treffen, welcher Datenstrom verwendet werden soll. Sollten diese Daten dieselbe Genauigkeit haben, ist dem Modul, das den IDS benötigt, von Seiten des Entwicklers eine Information mitzugeben, mit welcher Methode diese Information verarbeitet werden soll. Ein Beispiel wäre die Auswahl eines Maximalwertes oder Minimalwertes. Am Beispiel der Schimmelberechnung wäre der minimale Wert der Oberflächentemperatur zu wählen, da hier die größte Gefahr einer Schimmelbildung besteht. Die Problematik wird in nachfolgender Grafik gezeigt.

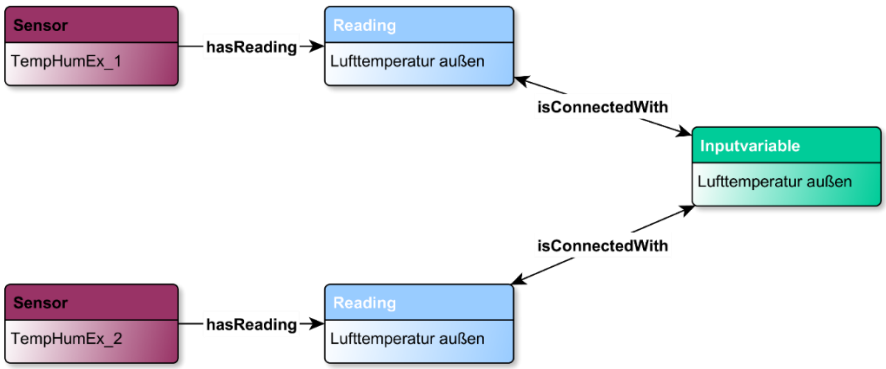


Abb. 29 Hilfsmodul mit einer “Many-to-One“ Verbindung

Im abgebildeten Beispiel stehen für die Bestimmung der Außenlufttemperatur zwei mögliche Datenströme zur Auswahl. Durch eine Definition der „Many-to-One“ Problematik ist die Methodik am gewählten Beispiel des Datensammlermoduls in der Lage, den Sensorwert mit dem niedrigsten Wert für die Berechnung zu wählen.
Durch eine Vererbung von Eigenschaften in der Ontologie ist es möglich die „Many-to-One“ Bedingungen eines Funktionsmoduls somit auch an Helpermodule weiterzugeben. Ein Beispiel für eine solche Vererbung von Eigenschaften wird mit nachfolgender Regel gezeigt. Diese vererbt die Zugehörigkeit eines Funktionsmoduls zu einem speziellen digitalen Zwilling an seinen ODS und stellt damit sicher, dass die verschiedenen Outputvariablen von Funktionsmodulen auch entsprechend ihrer Zuordnung unterschieden und verarbeitet werden können.

#Reasoning der Eigenschaft „isPartOf“ von verknüpftem Zwilling zum Anforderungsoutput:

User_Twin(?n)
Function_Modul(?l),
hasDemandOutput(?l, ?m),
isPartOf(?l, ?n),
-> isPartOf(?m, ?n)

Da die Einstufung eines Moduls in die Modulkategorie stark vom wissenschaftlichen Hintergrund und der Algorithmik abhängt, ist diese dem Entwickler solcher Module zu überlassen. Grundsätzlich gilt jedoch, dass die Verwendung gebäude- oder nutzerindividueller Parameter bei Hilfsmodulen ein Indiz für die Zuordnung zu einem digitalen Zwilling ist. Sollte ein Modul gleich viele Parameter jedes Zwillings haben, ist zu prüfen, ob dieses nicht in zwei Module getrennt werden kann.
Durch die Konzipierung der Hilfsmodule auf die hier vorgestellte Art wird dem bereits angesprochenen „Integriert und Dynamisch“ Kriterium Rechnung getragen. Hierdurch ist es möglich, die in Hilfsmodule verpackten Algorithmen auch für andere Berechnungen wiederzuverwenden. Durch die standardisierte Schnittstelle in Form der im Folgenden beschriebenen Datensammlermodule ist auch eine Übergabe der Daten im richtigen Format gewährleistet. Abb. 30 können alle zur Definition eines Hilfsmoduls notwendigen Informationen entnommen werden.

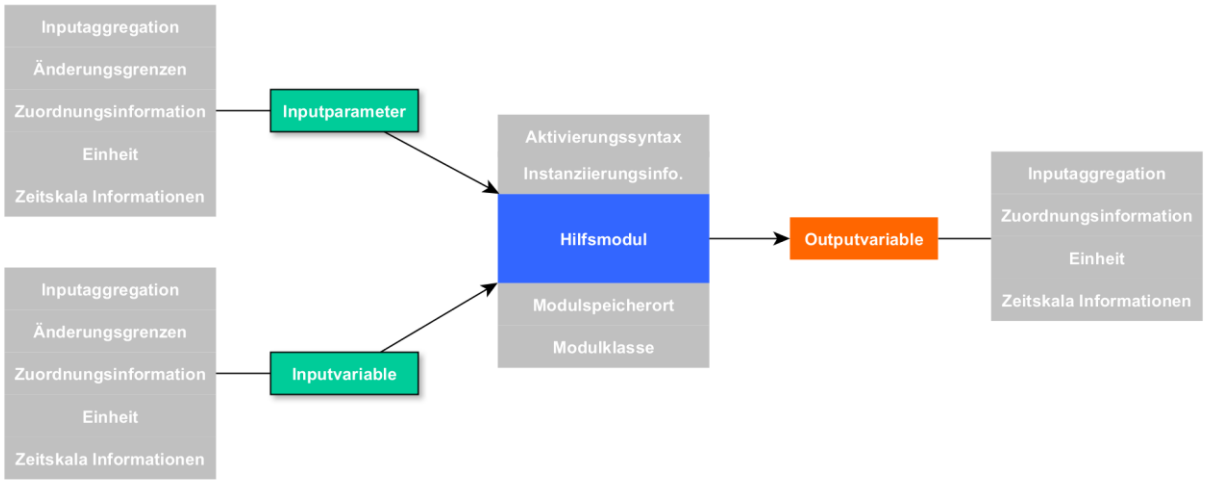


Abb. 30 Übersicht über Definitionen und Informationen zur Beschreibung eines Hilfsmoduls

4.3.7 Datensammlermodule

Für grundlegende Aufgaben der Datenaufbereitung wurde für die hier beschriebene Methode ein Datensammlermodul entwickelt. Dessen Aufgabe ist das Suchen und Aufbereiten der für Module benötigten Daten. Dazu können im Einzelfall verschiedene Aufgaben nötig sein. Im einfachsten Fall geschieht dies, wenn die Daten bereits im richtigen Format und zeitlichen Intervall vorliegen. Dann werden Daten kopiert und an das Modul, welches sie benötigt, übergeben. Im Rahmen der hier vorgestellten Methode hinterlegt ein Modul für die benötigten Inputwerte folgende Information:

[Datentyp],[Dateneinheit],[zeitliche Auflösung],[optional: Genauigkeit],[Reihenfolge der Übergabe]

Das Datensammlermodul sucht nun nach den benötigten Daten in den zur Verfügung stehenden Ressourcen. Dazu macht es sich die Zusammenhänge, die mithilfe der Ontologie erfasst wurden zu Nutze. Im Anschluss werden die Daten konvertiert und in eine geeignete Zeitreihe gebracht, um sie dann im richtigen Format an das Modul übergeben zu können.

Das Datensammlermodul ist prinzipiell ein generisches Standardmodul und unterstützt alle üblichen Datentransformationen. Somit ist nur dann ein zusätzliches Datensammlermodul vom Entwickler bereitzustellen, wenn der generische Ansatz nicht die notwendigen Bedürfnisse erfüllt. Da aufgrund des „Open-World“-Ansatzes bei Ontologien im Vorfeld nicht bestimmt werden kann, welche Daten konvertiert werden müssen, ist es durchaus möglich, dass für bestimmte Aufgaben Datensammlermodule vom Modulentwickler mitgeliefert werden müssen. Auch hier ist folgender Abbildung eine Entnahme aller nötigen Definitionen und Informationen zur Entwicklung eines Datensammlermoduls möglich.

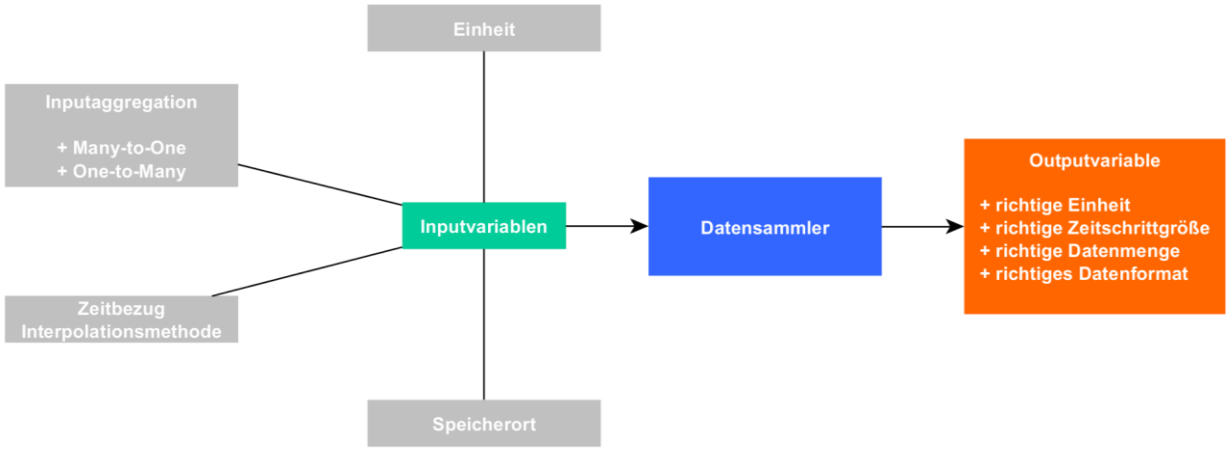


Abb. 31 Notwendige Definitionen zur Beschreibung eines Datensammlermoduls

Einheitenkonvertierung

Die Konvertierung von Einheiten in andere Formate sind standardisierte Abläufe, die das Datensammlermodul erledigt. Da die Umrechnungen meist mit einfachen Multiplikationen oder Division erledigt werden, sind die hier zu hinterlegenden Algorithmen nicht komplex und sollten hardwareunabhängig eine schnelle Berechnung ermöglichen.

„Many-to-One“ und „One-to-Many“ Konvertierung

Wie bei Mitterhofer (2017) bereits beschrieben, gibt es im Bereich der Simulation die Möglichkeit „One-to-One“ oder „One-to-Many“ Verbindungen zu erzeugen. Hierbei wird ein Wert an ein Modul oder an mehrere Module übergeben. In der hier konzipierten Methode können aber prinzipiell auch „Many-to-One“ oder „Many-to-Many“ Verbindungen auftreten. In diesem Fall muss definiert sein, wie die Daten übernommen und vom Datensammlermodul verarbeitet werden.

Sollten „Many-to-One“ Verbindungen auftreten, so ist bereits im Vorfeld vom Entwickler anzugeben, mit welcher Methode das Problem eventuell konkurrierender Datenquellen zu beheben ist. Da es sich um gleiche Information nur aus verschiedenen Quellen handelt, ist im Regelfall anhand der Genauigkeit des Wertes zu entscheiden, welcher Wert übernommen werden soll. Es sind jedoch auch Szenarien denkbar, bei denen eine Mittelwertbildung zielführend ist. Im speziellen Fall des Vorliegens vieler gleichartiger Datenquellen entscheidet der Entwickler wie verfahren werden soll und hinterlegt die entsprechenden Informationen im Modul. In wenigen Fällen kann es zielführend sein, einen stark individuellen aber mit größerer Abweichung versehenen Wert zu verwenden. Diese Fälle sind aber aufgrund des fachspezifischen Hintergrundes des Modulentwicklers nahezu immer im Vorfeld bekannt. Der Entwickler kann somit auch hier eine Information in seinem Modul implementieren, wodurch gewährleistet wird, dass ein individueller Nutzerparameter unter allen Umständen vorgezogen wird. Für sehr spezielle Fälle, in denen keine der hier aufgeführten Strategien zielführend ist, kann auch ein eigenes Datensammlermodul vom Entwickler mitgeliefert werden, welches diese Zuordnung besser verarbeiten kann. Zu diesem Zweck werden Module, in der hier vorgestellten Systematik, mit einer Dateneigenschaft versehen, die eine Zuordnung zum entsprechenden Datensammlermodul herstellt.

Die sogenannte „One-to-Many“ Konvertierung übernimmt der ontologiebasierte Algorithmus durch das getrennte Sammeln aller Daten für das jeweilige Modul bereits automatisch. Daher wird hier nicht weiter darauf eingegangen.

4.3.8 Wirkmodule

Wirkmodule stellen die Verknüpfung zwischen Funktionsmodulen und deren ODS mit den Geräten des Smarthomes her. Sie kalkulieren die nötigen Maßnahmen, um die im ODS formulierte Wunschwirkung eines Funktionsmoduls zu erzeugen. Dabei können diese Maßnahmen sowohl auf DGZ als auch auf DNZ Seite liegen. Durch diese offene Gestaltung sind beispielsweise auch Sprachausgaben an den Nutzer mit Handlungsanweisungen möglich. Auch wenn die primäre Funktion von Wirkmodulen die Verbindung von Maßnahmen mit Geräten darstellt, bilden sie doch im übertragenen Sinn die Wirkungen auf das Gesamtsystem ab. Damit stellen sie für gewünschte Simulationen mit Iterationsschritten auch die Stelle dar, an der Werte und Informationen wieder in die Funktionsmodule für zukünftige Berechnungsschritte eingespeist werden können. Grundsätzlich nimmt ein Wirkmodul Eingangsdatenströme entgegen, die der Form der ODS von Funktionsmodulen nach der nachfolgend erläuterten Prioritätenbereinigung entsprechen.

Wirkmodule stellen zusätzlich die Schnittstelle zwischen Aktuatoren und dem von den digitalen Zwillingen formulierten Bedürfnissen dar. Sie sind in der Lage Konvertierungen von einfach formulierten Wirkungsanforderungen in Aktionsvorschläge zu generieren.
Wirkmodule haben, wie die meisten anderen Module, Ein- und Ausgangsdatenströme. Im Regelfall erhalten sie als IDS die ODS von Funktionsmodulen. Der im Folgenden beschriebene Reduzierungsprozess aller Ausgangsdaten auf Basis der Prioritäten, ändert am Format der Daten grundsätzlich nichts. Die Eingangsdaten bei Wirkmodulen müssen daher im Format, welches in der folgenden Tabelle beispielhaft aufgeführten ist, vorliegen.

Tabelle 13 Beispiel Eingangsdatenstrom für ein Wirkmodul

Wirkgröße	**Zielgröße**	**Logischer Operator**	**Priorität**
Luftwechsel	2 - ∞	>=	A

Ihre Ausgangsdatenströme können in zwei Kategorien unterschieden werden:

- Steuerbare Ausgangsdaten
- Feedback Ausgangsdaten

Steuerbare ODS dienen dazu aktiv Steuerungen im Haus vorzunehmen. Sie werden an die in der Ontologie eingepflegten und zuordenbaren Geräte übergeben und bewirken damit die direkte Ansteuerung von Geräten. Um diese Verknüpfungen herzustellen, laden sie entsprechend der Ontologie alle verfügbaren und für den Modulzweck relevanten Geräte und verarbeiten intern die richtige Auswahl und Steuerung dieser. Dazu benötigen sie jedoch auf Seiten der Geräte relevante Daten. Diese werden entsprechend der Zuordnung von Sensordaten durch spezifizierende Merkmale mithilfe der Ontologie verbunden. Folgende tabellarische Auflistung zeigt beispielhaft Zuordnungsmerkmale, um Geräte einem Wirkmodul zuzuordnen:

Tabelle 14 Eigenschaften für die Zuordnung von Aktuatoren zu Wirkmodulen

Eigenschaftstyp	**Beispiele für Eigenschaft**
hasAction	Ventilation
hasLocation	Bath
hasObjectAffiliationTo	Air_Space_Bath
hasParttype	No_Parttype
isLocatedAt	Inside

Zusätzlich zu dem zur Zuordnung relevanten Datenstrom, sollten sie jedoch noch Informationen zur Art der Ansteuerung und der Wirkung des Gerätes enthalten. Dies kann durch die im Folgenden aufgeführten Eigenschaften geschehen.

Tabelle 15 Zusätzliche erforderliche Eigenschaften von Aktuatoren

Eigenschaftstyp	**Beispiele für Eigenschaft**
hasGranulation	10 % steps
hasAmmount	0-150
hasUnit	m^3

Feedback-Datenströme hingegen erzeugen, ähnlich wie Hilfsmodule, Daten, die das Feedback der Wohnsituation auf die Wirkgröße darstellen. Im Beispiel der Schimmelpilzvermeidung unter anderem die resultierende Luftfeuchte aus einer Erhöhung des Luftwechsels als Zeitreihe. Damit stellen sie die, für eine iterative oder situationsbedingte Berechnung notwendige Datenbasis zur Verfügung. Dementsprechend können Wirkmodule ähnlich komplexe Algorithmen wie Funktionsmodule oder Hilfsmodule enthalten. Folgende Abbildung zeigt die nötigen Informationen und Definitionen zur Implementierung eines Wirkmoduls.

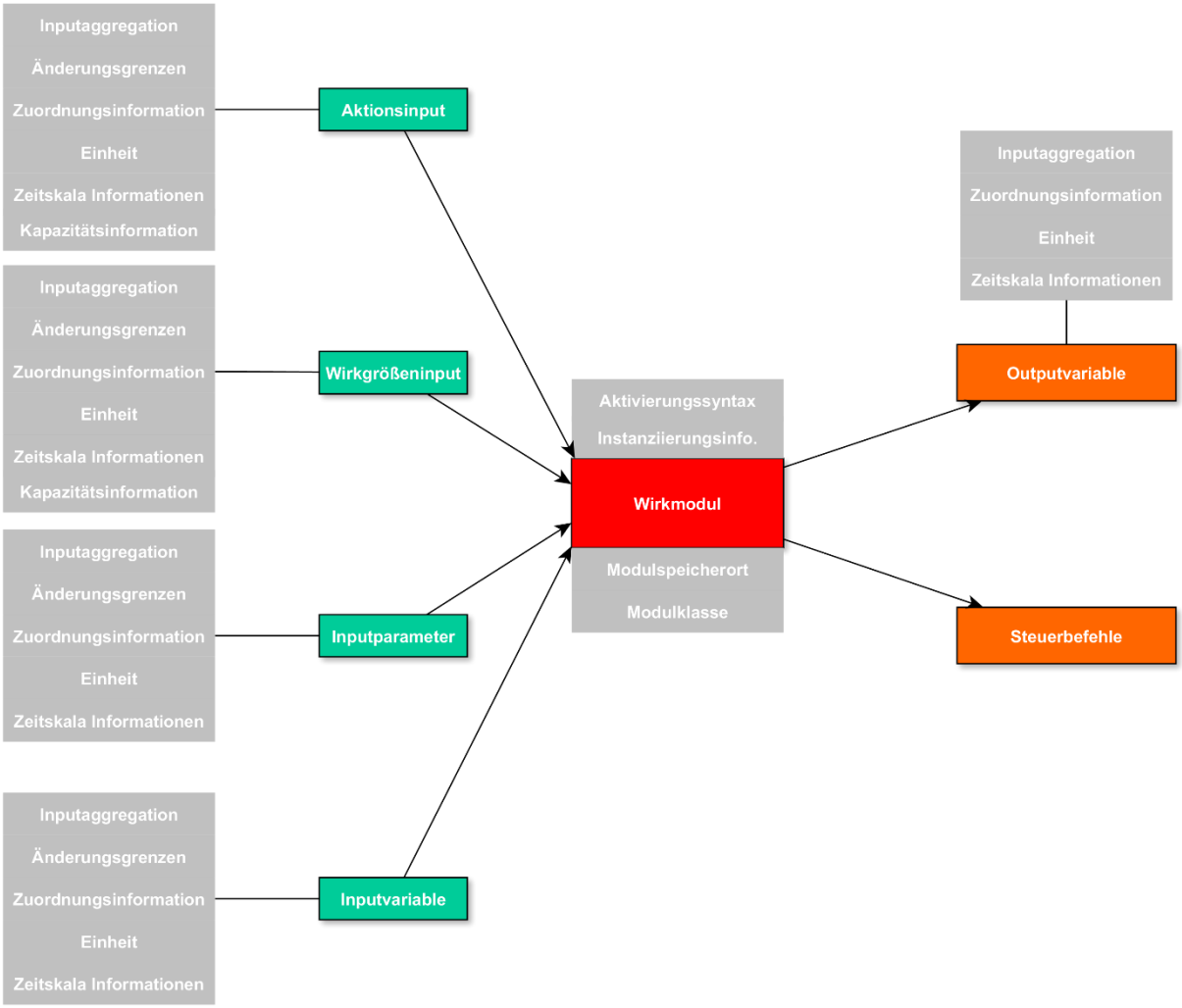


Abb. 32 Übersicht über Definitionen und Informationen zur Beschreibung eines Wirkmoduls

Die Ausgabe der Feedback-Datenströme erfolgt wie bereits erwähnt im selben Format wie bei Hilfsmodulen oder Sensoren. Ihre Zuordnung über die Ontologie erfolgt daher über die gleichen Merkmale, die verwendet werden, um Sensoren zuzuordnen. Für die in dieser Arbeit nicht näher angesprochene Berechnung von iterativen Simulationen für die Berechnung zukünftiger Ereignisse ist dementsprechend ein Reasoning dieser Ausgangsdatenströme an die Eingangsdatenströme der Funktions- oder Hilfsmodule notwendig. Für die Berechnung mit einem Zeithorizont in der Zukunft, sind demnach die Plätze der Eingangsdaten vollständig aus ODS von Wirkmodulen zu besetzen. Eine Regel zum Reasoning dieser Verbindung wird beispielhaft im Folgenden dargestellt, im Weiteren aber nicht mehr verfolgt.

#Reasoning der Zuordnung Output Wirkmodul → Input Funktionsmodul

Driver_Modul(?a),
hasOutputVariable(?a, ?b),
hasQuantity(?b, ?c),hasMedium(?b, ?d),hasParttype(?b, ?f),hasObjectAffiliationTo(?b, ?g),
hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Function_Modul(?l),
hasInputVariable (?l, ?m),
hasQuantity(?m, ?n),hasMedium(?m, ?o),hasParttype(?m, ?q),hasObjectAffiliationTo(?m, ?r),
hasLocation(?m, ?s),isLocatedAt(?m, ?t),

SameAs (?c, ?n),SameAs (?d, ?o),SameAs (?f, ?q),SameAs (?g, ?r),SameAs (?h, ?s),
SameAs (?i, ?t)

-> isConnectedWith(?b, ?m), isConnectedWith(?m, ?b)

4.3.9 Module für maschinelles Lernen und Vorhersagen

Um Individualisierungen vorzunehmen werden bereits jetzt in verschiedensten Disziplinen selbstlernende Algorithmen verwendet. Mit diesen Algorithmen ist es möglich, Computern Lernmethoden anzueignen, die dem menschlichen Lernen ähneln und damit in der Lage sind, komplexe Lernaufgaben zu lösen. Mithilfe dieser Techniken ist es auch möglich, Verhaltensmuster von Personen zu analysieren und somit deren Vorlieben situationsabhängig vorherzusagen. Die Algorithmik für diese Vorhersagen ist grundsätzlich bereits in verschiedenen Frameworks verfügbar. Ein standardisiertes Ansprechen dieser Algorithmen, unabhängig von der Lernaufgabe, ist damit bereits ebenfalls verfügbar. Daher ist es im Sinne der Modularität sinnvoll, diese Algorithmen in einem universalen Modul zur Verfügung zu stellen. Somit muss nicht jeder Entwickler, der die Ressourcen des maschinellen oder „Deep Learning“ nutzen möchte, diese Strukturen selbst implementieren. Das ist ressourcensparend und verringert den Aufwand der Systempflege. Die Datenübergabe an solche Module kann auf dieselbe Weise erfolgen wie bei üblichen Hilfsmodulen. Auf eine genaue Ausarbeitung dieser Modulstruktur soll an dieser Stelle verzichtet werden, da diese stark von der Art der verwendeten Algorithmen abhängig ist. Welche dieser Systeme für die jeweiligen bauphysikalischen Fragestellungen Verwendung finden sollen muss jedoch erst noch in weiteren Forschungen überprüft werden.

Grundsätzlich kann bei diesen Modulen von zwei unterschiedlichen Betriebsarten ausgegangen werden, der Entwicklungs- und der Implementierungsphase. Die Entwicklungsphase dient, wie bereits erwähnt, der Durchführung des Lernprozesses. Für den produktiven Betrieb ist demnach die Implementierungsphase relevant. In dieser werden aktiv Vorhersagen oder Klassifikationen durchgeführt.

In bestimmten Fällen kann die Ausführung der Entwicklungsphase auch zum Erlernen eines einzelnen Parameters genutzt werden. Dies ist beispielsweise bei Regressionsproblemen möglich. Im Fall des Erlernens eines Parameters, ist die Funktion der Implementierungsphase durch das Schreiben des neuen Parameters in eine Datenbank beendet. Die genaue Auslegung und Integration dieser Algorithmen unterliegen hierbei jedoch dem Entwickler. Durch die hier vorgestellte Methodik sind in der Handhabung nahezu keine Grenzen gesetzt, es wird eher ein Rahmen geschaffen, um diese Methoden möglichst entwicklungsarm einsetzen zu können. Somit ist eine Möglichkeit geschaffen sowohl Lernalgorithmen als auch Algorithmen für Vorhersagen zu implementieren.

4.4 „Knowledge-Based“ Algorithmik zur Interaktion

Um die Zusammenhänge zwischen all diesen Modulen abbilden zu können und automatisch den richtigen Bezug zu finden, ist es notwendig, Informationen und Wissen über diese Struktur in die Algorithmik zu integrieren. Für diese Aufgabe zeigten sich Ontologien als geeignetes Mittel. Diese sind in der Lage Verknüpfungen von Datenströmen mit Hilfe der bereits definierten Eigenschaften zu erstellen. Für die Integration der Simulationsmodelle wäre dadurch auch die Nutzung von FMUs möglich. Diese kapseln das Simulationsmodell in einer „Blackbox“ und geben über eine beschreibende Datei nur Informationen zu den Ein- und Ausgangsdatenströmen der Simulation bekannt. Bei der richtigen Definition der Datenströme in der Beschreibungsdatei ist so eine Implementierung dieser FMU’s in einer Ontologie entsprechend des hier vorgestellten Ansatzes möglich. Demnach können alle Interaktionen zwischen Modulen gefunden werden.

Nach der im Folgenden beschriebenen Definition der Klassen, werden die einzelnen Module als Instanzen in der Ontologie initialisiert und mit den für sie gültigen Eigenschaften versehen. Sind alle Informationen in der Ontologie enthalten, kann der Reasoner die Verbindungen zwischen den einzelnen Modulen und Teilen der Ontologie ermitteln und speichern.
Diese gespeicherte Information der Zusammengehörigkeit verschiedener Module ist essenziell für die Ermittlung der späteren Rechnungssetups und der Reihenfolge, in der die Module berechnet werden müssen. Das Finden der Verbindungen erfolgt durch das Prüfen der Übereinstimmung der Eigenschaften verschiedener Datenströme. Den Inputvariablen und -parametern sowie Sensoren und Outputvariablen und -parametern wurden dazu verschiedene Klassen und Eigenschaften zugeordnet. Diese definieren deren physikalische Beschaffenheit sowie deren Raumkontext in der Wohnung. Über diese kann der Reasoner erkennen, ob ein Sensorwert als Input für ein Modul verwendbar ist.

Durch die gleichzeitige Verwendung verschiedener Hilfsmodule ist die Möglichkeit gegeben, verschiedenste Transformationen an Daten automatisch zuordnen zu lassen. So können auch komplexere physikalische Berechnungen zum Erhalten von Inputvariablen durchgeführt werden.

Durch das Speichern der Informationen in der Ontologie, muss der Reasoner nicht jedes Mal von neuem die Verbindungen ermitteln. Solange keine neuen Module oder anderen Komponenten hinzukommen, wird sich an der Zuordnung der Datenströme in der Regel keine Änderung ergeben. Eine dauernd wiederkehrende Neuberechnung ist daher nicht notwendig und auch nicht zielführend.

Beim Hinzufügen oder Ändern von Modulen ist jedoch immer zwingend, ein vollständiges Neu-Reasoning der kompletten Ontologie durchzuführen. Ein partielles Reasoning könnte aufgrund der unüberschaubaren Struktur dazu führen, dass neue Verbindungen außer Acht gelassen werden.

Das Ermitteln der geeigneten Daten, als Inputvariablen zu den benötigten Modulen und der richtigen Aktuatoren zu den Outputvariablen von Wirkmodulen, ermöglicht somit die Abbildung einer vollständigen Steuerkette. Deren Start liegt in der Erfassung von Sensordaten und endet mit dem Auslösen einer Aktion. Durch die ontologische Abbildung wird dieser Zusammenhang maschinenlesbar gemacht. Der genaue Ablauf dieser Prozedur wird im Folgenden näher beschrieben. Vor allem um Probleme bei der Findung des richtigen Outputs und Fehlsteuerungen zu vermeiden, ist hier ein spezielles System notwendig. Dieses priorisiert verschiedene Aktionen und bestimmt den optimalen Ablauf, damit das System nicht durch vermeintlich positive Steueraktionen in eine Dilemma-Situation gerät, aus welcher es keinen schadensfreien Ausweg mehr gibt.

4.4.1 Klassen in der „Smarthome Ontologie“

Im Folgenden werden die wichtigsten Domänen und Klassen definiert, die in der Ontologie enthalten sein müssen, damit ein Reasoner später Zusammenhänge finden kann. Diese werden benötigt, um Interaktionen zwischen den Modulen abbilden zu können. Domänen und Klassen wurden anhand der beispielhaften Verwendung von Mitterhofer auf den hier gegebenen Anwendungsfall angepasst. Im Folgenden werden die notwendigen Domänen und deren Klassen erläutert. Da das Gesamtsystem den „Open-World“ -Ansatz verfolgt, ist eine nachträgliche Erweiterung der Klassen oder Domänen möglich. Für die üblichen bauphysikalischen Anwendungsfälle ist diese Ontologie jedoch ausreichend.

Domäne: Gebäude

Die Gebäudedomäne enthält alle Klassen, die das Gebäude beschreiben oder umschreiben.
Wie zum Beispiel Räume, Stockwerke, Wände und Türen. Eine Unterscheidung in verschiedene Zonenarten, Anhand des LOD der Berechnung, existieren in diesem Kontext nicht. Die Zonen werden immer alle mit derselben Auflösung berechnet, da für alle Zonen dasselbe Modul verantwortlich ist. Bei Datenübergaben zu anderen Modulen können Konflikte, entsprechend der bereits beschriebenen „Many-to-One“ Problematik, auftreten. Deren Behandlung ist vom Modulentwickler im Vorfeld zu bestimmen. Im Folgenden sind die, nach derzeitigem Stand in der Gebäudedomäne, implementierten Klassen tabellarisch aufgeführt.

Tabelle 16 Klassenübersicht der Gebäudedomäne

Klasse	**Beispiel-Instanzen**
Stockwerk	Keller Erdgeschoss Erster Stock Dachboden
Raum	Wohnzimmer Küche Badezimmer Kinderzimmer Toilette Flur
Bauteiletypen - Wand - Boden - Decke	Innenwand Außenwand Boden EG-OG Decke EG-OG
Komponenten - Luftvolumen - Oberflächen	Luftvolumen Bad Innenoberfläche der Wand Möbeloberfläche
Standort	Innen Außen

Domäne: Gebäudeequipment

Die Domäne Gebäudeequipment enthält alle Einträge, die die Ausstattungen des Gebäudes betrifft. Das umfasst sowohl Sensoren als auch Aktuatoren, sowie jegliche Geräte, die für Funktionalitäten im Smarthome hinzugefügt werden. Die Zuordnung eines Gerätes zur Position im Gebäude erfolgt über die Eigenschaften “hasLocation“, „isLocatedAt“, „hasObjectAffiliationTo“, hasParttype“ und gibt damit auf unterschiedlichen Skalen an, wo sich das Equipment befindet.

Bei einem Verschieben des Equipments von, zum Beispiel einem Raum in den Nächsten, muss damit nur der Eintrag „hasLocation“ geändert werden, womit die Zuordnung in einen neuen Raum erfolgt und ein neues Reasoning der dadurch entstandenen Abhängigkeiten stattfinden kann. Im Folgenden werden tabellarisch die Klassen der Gebäudeequipment Domäne abgebildet, welche im Rahmen dieser Arbeit entwickelt wurden.

Tabelle 17 Klassenübersicht der Gebäudeequipmentdomäne

Klassen	**Beispiel-Instanzen**
Sensor - Innen - Außen	Temperatursensor Feuchtigkeitssensor Luftgeschwindigkeitssensor Wärmestromsensor
Aktuatoren - Licht - Heizung - Kühlung - Verschattung - Ventilation - Entfeuchtung - Reinigungsgeräte - Smarte-Geräte	Wohnzimmerlicht Heizkörper Klimaanlage Jalousien Abluftanlage Staubsaugerroboter Smarte-Waage
Aktion	Kühlung Heizung Lüftung
Granulierung	An/Aus Linear

Domäne: Module

In der Domäne „Module“ sind alle Klassen aufgeführt, mit denen das Reasoning der Modulinteraktionen durchgeführt werden kann. Sie enthält die im Folgenden aufgeführten Klassen.

Tabelle 18 Klassenübersicht der Moduldomäne

Klassen	**Beispiele**
Datensammlermodul	Generischer Datensammler
Wirkmodul	Lüftungssteuerung
Funktionsmodul	Berechnung der Schimmelbildungsgefahr
Hilfsmodul	Berechnung der Oberflächenfeuchtigkeit
Ersatzdatenmodule	Extrapolation des Bauteil-U-Wertes aus dem Baualter des Gebäudes
Module für maschinelles Lernen und Vorhersagen	Vorhersage von Nutzerereignissen
Inputvariablen	Raumtemperatur
Inputparameter	Durchschnittlicher U-Wert der außen Bauteile
Outputvariablen	Oberflächenfeuchte
Outputparameter	Extrapolierter U-Wert aus Baualter
Aktionsinputs	Heizgerät
Anforderungsinputs	Bedingung über die zulässige Radonkonzentration
Anforderungsoutputs	Zulässige Radonkonzentration
Wirkgrößeninputs	Lüften
Ersatzdaten	Durchschnittliche Duschdauer

Domäne: Personen

Die Domäne „Personen“ enthält die Klassen, welche den Nutzer betreffen und für das Reasoning der Interaktion von DGZ und DNZ relevant sind, diese sind in folgender Tabelle aufgeführt.

Tabelle 19 Klassenübersicht der Personendomäne

Klassen	**Beispiele**
Person	Name Alter Geschlecht
Position	Anwesenheit
Stimmung	Fröhlich
Bedürfnisse	Spezielle Anforderungen an Beleuchtung
Aktivitäten	Aktiver Einfluss Passiver Einfluss Besondere Bedürfnisse

Domäne: Physik

Die Domäne „Physik“ enthält physikalische Basiswerte, die weltweit genormt sind und im hier gezeigten Ansatz Verwendung finden. Ihre Verwendung geht auf den Umgang mit Sensordaten zurück. Eine Übersicht der Klassen ist nachfolgender Tabelle zu entnehmen.

Tabelle 20 Klassenübersicht der Physikdomäne

Klassen	**Beispiele**
Medium	Wasser Luft Stein
Quantität	Relative Feuchtigkeit Temperatur Elektrischer Strom
Einheit	% Grad Celsius Ampère
Manifestation	Schimmel Radon

Domäne: Ersatzdaten

Die hier entwickelte Domäne „Ersatzdaten“ beinhaltet alle notwendigen Klassen, um Ersatzdaten zu beschreiben. Im hier gezeigten Fall ist das eine Klasse mit dem Namen Ersatzdaten. Für eine Verwendung dieser Methode im größeren Stil ist jedoch eine genauere Ausarbeitung dieser Klasse sinnvoll, um die Art der Daten und ihrer Genauigkeit zu untergliedern. Im Folgenden sind die Klassen tabellarische aufgelistet, die nach derzeitigem Stand sinnvoll erscheinen.

Tabelle 21 Klassenübersicht der Ersatzdatendomäne

Klassen	**Beispiele**
Normen	DIN 4102
Statistische Daten	Bundesamt für Statistik – durchschnittliche Duschlänge in Deutschland
Normdaten	mindest U-Werte bei Neubauten

4.4.2 Reasoning und Verbindungen in der Ontologie

Wie bereits angesprochen dient das Reasoning der Findung von Interaktionen von verschiedenen Modulen und dem Auffinden der richtigen IDS sowie der richtigen Aktionen und Geräte. Das Reasoning ist ein Prozess, der für eine bestehende Ontologie einmalig ausgeführt werden muss. Solange an der Struktur des Smarthome keine Veränderungen durchgeführt werden, ist kein Neu-Reasoning notwendig und die Berechnungen können durch Auslesen der Ontologie durchgeführt werden. Beim Reasoning wird nach Verbindungen zwischen Individuen der Ontologie gesucht, die zueinanderpassen. Dies passiert ähnlich einem Schlüssel-Schloss-Prinzip. Das Reasoning erfolgt in dieser Methode mehrstufig. Jede Stufe erzeugt dabei Verbindungen zwischen unterschiedlichen Bereichen der Ontologie

Stufe Ⅰ: Reasoning von Sensorwerte

Stufe Ⅱ: Reasoning von Modulinteraktionen

Stufe Ⅲ: Reasoning von Ersatzdaten

Stufe Ⅳ: Reasoning der Interaktionen von Funktionsmodulen

Stufe Ⅴ: Reasoning von Inputs der Wirkmodule zu Funktionsoutputs

Stufe Ⅵ: Reasoning von Geräten zu den Wirkmodule

Stufe Ⅶ: Reasoning der Vollständigkeit

Bei iterativen Rechenansätzen, die in dieser Arbeit nicht weiter beschrieben werden, erfolgt zwischen der Stufe Ⅴ und Ⅵ eine zusätzliche Stufe, die für die Iterationen zwei und weitere, ODS von Wirkmodulen als Ersatz für Sensorwerte, im Sinne einer Prognose, zu anderen Modulen verbinden.

Stufe Ⅰ

In der ersten Stufe werden die Inputvariablen eines Moduls mit den zu ihnen passenden Sensordaten verbunden. Dazu müssen sowohl Sensordaten als auch Inputvariablen von Modulen über ihre Eigenschaften mit Instanzen der in folgender Tabelle aufgeführten Klassen verbunden werden. Die aufgeführten Klassen ermöglichen damit die Sicherstellung, dass sowohl die örtliche als auch die inhaltliche Zuordnung von Daten sichergestellt ist.

Tabelle 22 Eigenschaftenübersicht zur Zuordnung von Sensoren

Klassen	**Eigenschaften**	**Beispiele**
Medium	hasMedium	SensorReading1 hasMedium Air
Quantity	hasQuantity	SensorReading1 hasQuantity Temperature
Room	hasLocation	Bath
Versch. Klassen mögl. z.B. aus Gebäudedomäne	hasObjectAffiliationTo	Air_Space_Bath
Parttype	hasParttype	Wall
Location	isLocatedAt	Inside

Die folgende Abbildung veranschaulicht die notwendigen Definitionen zur Beschreibung eines Sensors in der Ontologie grafisch. Ein Sensor kann hierbei mehrere Readings bereitstellen. Die richtige Zuordnung kann daher nur erfolgen, wenn jedes Reading eigene Informationen zur Zuordnung erhält.

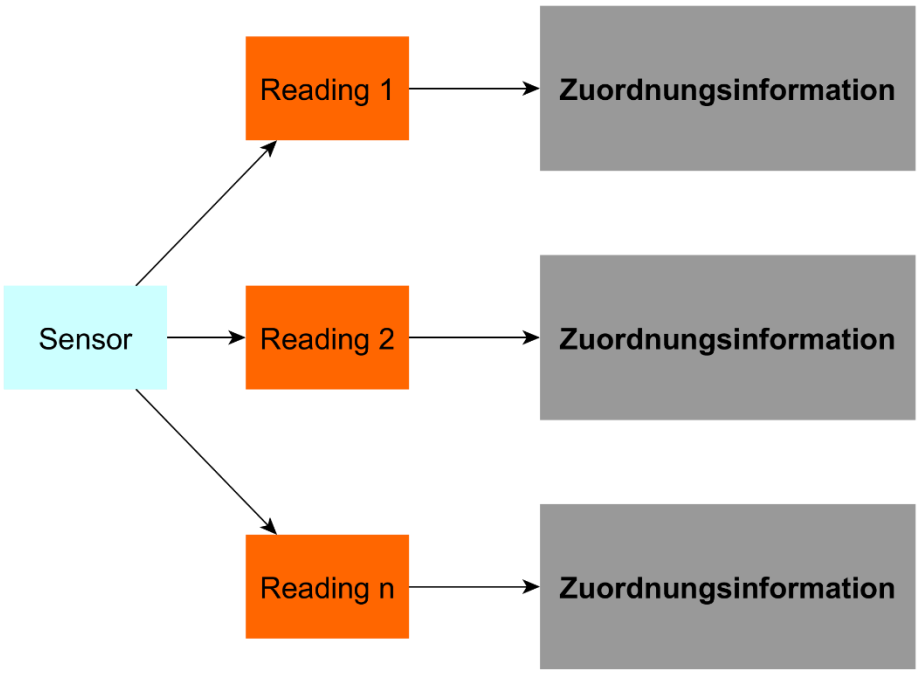


Abb. 33 Definitionsanforderungen an einen Sensor

Die Zuordnung der Sensoren zu Inputvariablen erfolgt im Rahmen der hier vorgestellten Methodik, durch Definition von Regeln in der Sprache SWRL. Die „Human readable“ Syntax ist im Folgenden abgebildet:

#Reasoning der Zuordnung Input Funktionsmodule → Reading Sensoren

Function_Modul(?a),
hasInputVariable(?a, ?b),
hasQuantity(?b, ?c),hasMedium(?b, ?d),hasParttype(?b, ?f),hasObjectAffiliationTo(?b, ?g),hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Sensor(?l),
hasReading(?l, ?m),
hasQuantity(?m, ?n),hasMedium(?m, ?o),hasParttype(?m, ?q),hasObjectAffiliationTo(?m, ?r),hasLocation(?m, ?s),isLocatedAt(?m, ?t),

SameAs (?c, ?n),SameAs (?d, ?o),SameAs (?f, ?q),SameAs (?g, ?r),SameAs (?h, ?s),SameAs (?i, ?t)

-> isConnectedWith(?b, ?m), isConnectedWith(?m, ?b)

#Reasoning der Zuordnung Input Hilfsmodule → Reading Sensoren

Helper(?a),
hasInputVariable(?a, ?b),
hasQuantity(?b, ?c),hasMedium(?b, ?d),hasParttype(?b, ?f),
hasObjectAffiliationTo(?b, ?g),hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Sensor(?l),
hasReading(?l, ?m),
hasQuantity(?m, ?n),hasMedium(?m, ?o),hasParttype(?m, ?q),
hasObjectAffiliationTo(?m, ?r),hasLocation(?m, ?s),isLocatedAt(?m, ?t),

SameAs (?c, ?n),SameAs (?d, ?o), SameAs (?f, ?q),SameAs (?g, ?r),
SameAs (?h, ?s),SameAs (?i, ?t)

-> isConnectedWith(?b, ?m), isConnectedWith(?m, ?b)

Ein Reasoning von Sensorwerten zu Wirkmodulen oder Modulen für maschinelles Lernen und Vorhersagen erfolgt analog. Die so gefundenen Verbindungen werden nun durch eine Eigenschaft über “isConnectedWith“ direkt in die Ontologie geschrieben.

Grundsätzlich muss die Spezifikation der Variable mindestens so genau erfolgen, dass beim Reasoning keine Daten verknüpft werden, die zur Berechnung unbrauchbar sind. Die Verantwortung der genauen Bestimmung der Eingangsvariablen unterliegt hier also dem Modulentwickler und somit einem Spezialisten.

Stufe Ⅱ

Im zweiten Schritt des Reasoning wird nun versucht alle an Modulen noch offenen Inputvariablen, zu denen es keinen direkten Sensorwert gibt, mithilfe von anderen Modulen zu verknüpfen. Der Prozess des Reasonings selbst erfolgt auf dieselbe Art wie im ersten Schritt, so dass statt Sensorwerten Outputvariablen von Hilfsmodule verbunden werden können. Auch die hier gefundenen Verbindungen werden wieder in die Ontologie geschrieben. Die dafür erforderliche Regel ist im Folgenden abgebildet:

#Reasoning der Zuordnung Output Hilfsmodule → Input Hilfsmodule

Helper(?a),
hasInputVariable(?a, ?b),
hasQuantity(?b, ?c),hasMedium(?b, ?d),hasParttype(?b, ?f),
hasObjectAffiliationTo(?b, ?g),hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Helper(?l),
hasOutputVariable(?l, ?m),
hasQuantity(?m, ?n),hasMedium(?m, ?o),hasParttype(?m, ?q),
hasObjectAffiliationTo(?m, ?r),hasLocation(?m, ?s),isLocatedAt(?m, ?t),

SameAs (?c, ?n),SameAs (?d, ?o),SameAs (?f, ?q),SameAs (?g, ?r),
SameAs (?h, ?s),SameAs (?i, ?t)

-> isConnectedWith(?b, ?m), isConnectedWith(?m, ?b)

#Reasoning der Zuordnung Output Hilfsmodule → Input Funktionsmodule

Function_Modul(?a),

hasInputVariable(?a, ?b),

hasQuantity(?b, ?c),hasMedium(?b, ?d),hasParttype(?b, ?f),

hasObjectAffiliationTo(?b, ?g),hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Helper(?l),

hasOutputVariable(?l, ?m),

hasQuantity(?m, ?n),hasMedium(?m, ?o),hasParttype(?m, ?q),

hasObjectAffiliationTo(?m, ?r),hasLocation(?m, ?s),isLocatedAt(?m, ?t),

SameAs (?c, ?n),SameAs (?d, ?o),SameAs (?f, ?q),SameAs (?g, ?r),

SameAs (?h, ?s),SameAs (?i, ?t)

-> isConnectedWith(?b, ?m), isConnectedWith(?m, ?b)

Stufe Ⅲ

Im dritten Schritt erfolgt ein Reasoning, dass sich bei der Wahl der Eingangsgrößen ausschließlich auf Ersatzdaten beschränkt. So können im Normalfall alle noch fehlenden Inputvariablen besetzt werden. Auch diese werden durch den Eintrag einer Eigenschaft „isConnectedWith“ markiert. Das dafür erforderliche Regelwerk stellt sich folgendermaßen dar:

Reasoning der Zuordnung Ersatzdaten → Inputparameter

Modul(?a),

hasInputParameter(?a, ?b),

hasQuantity(?b, ?c),hasMedium(?b, ?d),hasParttype(?b, ?f),

hasObjectAffiliationTo(?b, ?g),hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Replacementdata(?m),

hasQuantity(?m, ?n),hasMedium(?m, ?o),hasParttype(?m, ?q),

hasObjectAffiliationTo(?m, ?r),hasLocation(?m, ?s),isLocatedAt(?m, ?t),

SameAs (?c, ?n),SameAs (?d, ?o), SameAs (?f, ?q),SameAs (?g, ?r),

SameAs (?h, ?s),SameAs (?i, ?t)

-> isConnectedWith(?b, ?m), isConnectedWith(?m, ?b)

Im Regelfall liegt es im Aufgabengebiet des Modulentwicklers für den Fall von fehlender Sensorik in einem Gebäude ausreichend Hilfsmodule bereit zu stellen oder ausreichend Ersatzdaten zur Verfügung zu stellen. Dies ist natürlich nur bis zu dem Punkt möglich, an dem die Genauigkeit von Berechnungen durch die Verwendung derart leidet, dass keine brauchbaren Outputs mehr vom Funktionsmodul generiert werden können. Die Entscheidung, ab wann dieser Punkt erreicht ist, liegt aufgrund des erforderlichen Fachwissens über modulinterne Berechnungen immer im Entscheidungs- und Verantwortungsbereich des einzelnen Entwicklers. Durch die Modulinteraktionen ist jedoch nicht immer im Vorfeld abzusehen mit welchen Genauigkeiten die Eingangsdaten von Modulen versehen sind. Daher wird für jedes Modul eine integrierte Fehlerberechnung notwendig die auf Basis der Eingangsdaten die Genauigkeiten in Bezug auf den Moduloutput kalkuliert. Damit ist unabhängig von der Genauigkeit der Eingangsdatenströme eine Aussage über die Grenzen des Verwendungsbereiches eines Moduls im individuellen Fall möglich.

Stufe Ⅳ

In der vierten Stufe werden nun Verbindungen zwischen Funktionsmodulen gesucht. Dieser Teil des Reasonings erstellt die notwendigen Verbindungen zwischen Funktionsmodulen des DGZ und des DNZ. Das Reasoning der Interaktionen zwischen Funktionsmodulen läuft nahezu gleich ab, wie bei den vorherigen Schritten. Die zum Reasoning verwendeten Eigenschaften sind in folgender Tabelle aufgeführt.

Tabelle 23 Notwendige Eigenschaften zur Zuordnung von Anforderungsdatenströmen

Klassen	**Eigenschaften**	**Beispiele**
Manifestation	hasManifestation	Mould
Medium	hasMedium	Surface
Parttype	hasParttype	No_Parttype
Room	hasLocation	Bath
Versch. Klassen mögl. z.B. aus Gebäudedomäne	hasObjectAffiliationTo	No_Object
Location	isLocatedAt	Inside

Für den Prozess der Zuordnung erforderliche Regel:

#Reasoning der Zuordnung Anforderungsoutput Funktionsmodule -> Anforderungsinput Funktionsmodule

Function_Modul(?a),

isPartOf(?a, ?SomeTwin1),

(User_Twin) (?SomeTwin),

hasDemandOutput(?a, ?b),

hasManifestation(?b, ?c),hasMedium(?b, ?d),hasParttype(?b, ?f),

hasObjectAffiliationTo(?b, ?g),hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Function_Modul(?l),

isPartOf(?l, ?SomeTwin2),

(Building_Twin)(?SomeTwin2),

hasDemandInput(?l , ?m),

hasManifestation(?m, ?n),hasMedium(?m, ?o),hasParttype(?m, ?q),

hasObjectAffiliationTo(?m, ?r),hasLocation(?m, ?s),isLocatedAt(?m, ?t),

SameAs (?c, ?n),SameAs (?d, ?o),SameAs (?f, ?q),SameAs (?g, ?r),

SameAs (?h, ?s),SameAs (?i, ?t)

-> isConnectedWith(?b, ?m), isConnectedWith(?m, ?b)

Stufe Ⅴ

In diesem Schritt werden daraufhin Verbindungen zwischen Outputdaten von Funktionsmodulen und den sogenannten „ImpactInputs“, den Inputdaten von Wirkmodulen verbunden. Im Anschluss erfolgt noch eine Bearbeitung dieses Outputs nach dem Algorithmus zur Prioritätenbereinigung. Dieser ändert jedoch nichts an der Form der Daten und wird daher erst im Folgenden betrachtet. Einem direkten Reasoning der Outputvariablen steht demnach nichts entgegen. Die vom vorher genannten Algorithmus für eine Steuerung bestimmten ODS können mithilfe dieser Reasoningschritte einem Wirkmodul zugeordnet werden. Das Reasoning der Outputvariablen zu den Inputvariablen der Wirkmodule erfolgt mit nachfolgend aufgeführten Eigenschaften.

Tabelle 24 Notwendige Eigenschaften zur Zuordnung von Wirkgrößen

Klassen	**Eigenschaften**	**Beispiele**
Medium	hasMedium	Air
Parttype	hasParttype	No_Parttype
Room	hasLocation	Bath
Quantity	hasQuantity	AirChangeRate
Versch. Klassen mögl. z.B. aus Gebäudedomäne	hasObjectAffiliationTo	Air_Space_Bath
Location	isLocatedAt	Inside

Das für die Zuordnung erforderliche Regelwerk ist nachfolgend abgebildet:

#Reasoning der Zuordnung Output Funktionsmodule → Wirkgrößeninput Wirkmodul

Function_Modul(?a),

hasOutputVariable(?a, ?b),

hasMedium(?b, ?c),hasQuantity(?b, ?d),hasParttype(?b, ?f),hasObjectAffiliationTo(?b, ?g),

hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Driver_Modul(?l),

hasInputVariable(?l, ?m),

hasMedium(?m, ?n),hasQuantity(?m, ?o),hasParttype(?m, ?q),

hasObjectAffiliationTo(?m, ?r),hasLocation(?m, ?s),isLocatedAt(?m, ?t),

SameAs (?c, ?n),SameAs (?d, ?o),SameAs (?f, ?q),SameAs (?g, ?r),

SameAs (?h, ?s),SameAs (?i, ?t)

-> isConnectedWith(?b, ?l), isConnectedWith(?m, ?b)

Stufe VI

Der sechste Schritt des Reasonings stellt nun die Verbindung der Wirkmodule zu den jeweiligen Geräten dar. Das Reasoning erfolgt dabei nach gleichem Muster, wie bei den vorhergehenden Schritten. Die für den Prozess erforderlichen Eigenschaften sind in nachfolgender Tabelle aufgeführt.

Tabelle 25 Notwendige Eigenschaften zur Zuordnung von Aktuatoren

Klassen	**Eigenschaften**	**Beispiele**
Parttype	hasParttype	No_Parttype
Room	hasLocation	Bath
Versch. Klassen mögl. z.B. aus Gebäudedomäne	hasObjectAffiliationTo	Air_Space_Bath
Location	isLocatedAt	Inside
Quantity	hasQuantity	VolumetricFlowRate
Actiontyp	hasAction	Ventilation

Um den vollen beschriebenen Funktionsumfang zu gewährleisten, sind dafür auch Datenströme von einem Gerät zum Wirkmodul in Form von „Readings“ notwendig. Der Prozess der Zuordnung ist ähnlich wie bei der Verwendung von Sensordaten. Die dafür zu definierenden Informationen zu dem jeweiligen Gerät sind im Folgenden abgebildet.

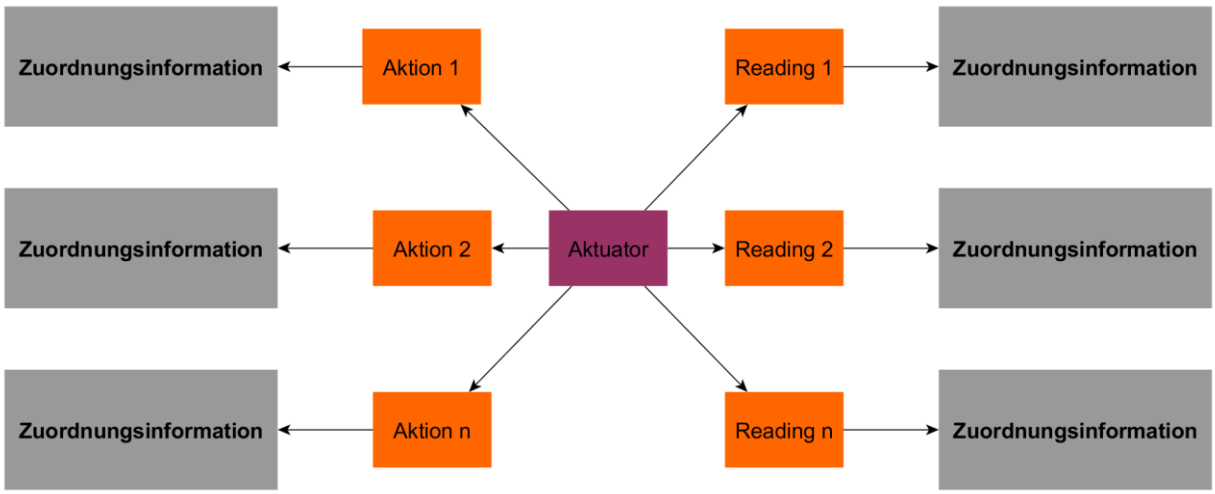


Abb. 34 Definitionsanforderungen an einen Aktuator

Die, für die richtige Zuordnung erforderliche Regel, ist folgend abgebildet:

#Reasoning der Zuordnung Aktuator → Aktionsinput Wirkmodul

Driver_Modul(?a),

hasInputVariable(?a, ?b),

hasAction(?b, ?c),hasQuantity(?b, ?d),hasParttype(?b, ?f),hasObjectAffiliationTo(?b, ?g),

hasLocation(?b, ?h),isLocatedAt(?b, ?i),

Actuator(?l),

hasAction(?l, ?n),hasQuantity(?l, ?o),hasParttype(?l, ?q),hasObjectAffiliationTo(?l, ?r),

hasLocation(?l, ?s),isLocatedAt(?l, ?t),

SameAs (?c, ?n),SameAs (?d, ?o),SameAs (?f, ?q),SameAs (?g, ?r),

SameAs (?h, ?s),SameAs (?i, ?t)

-> isConnectedWith(?b, ?l), isConnectedWith(?l, ?b)

Stufe Ⅶ

Im siebten Schritt muss nun noch untersucht werden ob Funktionsmodule existieren, deren Inputdaten nicht bedient werden können. Dies wird überprüft durch eine rückwärtige Betrachtung der Datenpfade. Endet ein Datenstrom rückwärts in einer „Sackgasse“ in Form einer offenen Inputvariable so kann sicher festgestellt werden, dass für das Modul nicht genügend Daten zur Verfügung stehen. Endet jede Inputvariable dagegen in einem übertragenen Wert aus einem Sensor oder im Bereich der Ersatzdaten, so kann sicher festgestellt werden, dass genug Daten für die Berechnung verfügbar sind. Dies gilt auch, wenn sich ein Datenpfad über mehrere Hilfsmodule erstreckt oder in einem Modul ohne Inputvariablen endet. Auf eine beispielhafte Umsetzung dieser Funktion wurde verzichtet, da die Funktionalität trivial ist. Über eine einfache Abfrage können Variablen abgerufen werden die keine Eigenschaft mit „isConnectedWith“ besitzen. Sollten diese Funktionsmodule sein, sind sie als nicht rechenbar zu deaktivieren. Sollten diese Hilfsmodule sein und deren Outputvariable mit einem anderen Modul verbunden sein, so ist zu prüfen, ob dieser Pfad zu einem Funktionsmodul führt und in diesem Fall dieses Modul zu deaktivieren ist. Die Algorithmik, um diese einfache Funktion abzubilden, lässt sich mit einfachen „if-Schleifen“ abbilden.

Nach Abschluss des letzten Reasoningschrittes stehen genug Informationen zur Verfügung, um die notwendigen Berechnungen durchzuführen. Sollte bei einem Schritt des Reasonings mehrere Möglichkeiten zur Verbindung existieren, so werden diese automatisch alle verknüpft. Die dadurch entstehende „One-to-Many“ oder „Many-to-One“ Problematik wird, wie bereits beschrieben, durch das Datensammlermodul entschärft.

In der vorgestellten Methode spielt die Unterscheidung von des LOD verschiedener Zonen keine übergeordnete Rolle. Das hat den Hintergrund, dass ein Modul bei Berechnungen für verschiedene Räume mehrfach instanziiert wird und damit mit demselben „Level of Detail“ (LOD) rechnet. Sollten jedoch für denselben Anwendungsfall zwei gleiche Module vorhanden sein, die jedoch einen unterschiedlichen LOD haben, so ist allein aus Gründen der Genauigkeit immer das Modul mit dem größten LOD zu bevorzugen. Zu diesem Zweck benötigen alle Module eine Angabe, mit der das LOD

vergleichbar wird. Bei zwei dann immer noch identischen Modulen ist vom Nutzer oder dem Entwickler eine Vorgabe für die Berechnung zu machen.

Zu guter Letzt stellt sich noch die Frage, welcher Prozess das Reasoning aktiviert. Ein neues Reasoning ist erforderlich, wenn an den Randbedingungen oder der Struktur von Modulen, Sensoren oder Aktuatoren Änderungen auftreten. Dies kann zum Beispiel sein.

Hinzufügen, Verändern oder Entfernen von:

- Modulen
- Geräten
- Sensoren
- Nutzern
- Gebäuden oder Gebäudeteilen wie Räumen
- Daten oder Eigenschaften bestehender Individuen
- Klassen oder Änderungen an der Ontologie
- Ersatzdaten

Prinzipiell sollte jede Veränderung in der Ontologie einen neuen Reasoning Prozess anstoßen. Die genannten Trigger sind reine Trigger für den Prozess des Reasonings und haben damit keinen Einfluss auf das Triggern von Berechnungen, welche nun im Folgenden beschrieben werden.

4.4.3 Bestimmung des Rechensetups und Triggerung

Nachdem alle Verbindungen zwischen eingehenden Daten und Modulen gefunden wurden, können nun Berechnungen stattfinden.

Triggern der Berechnungen

Den Anstoß für eine neue Berechnung liefert ein sogenannter Trigger.

Grundsätzlich kann jedes Event in einem Smarthome einen Trigger auslösen. Events sind dabei jede Information, die im Automationsserver eingeht und verarbeitet wird. Das können beispielhaft neue Sensorwerte oder Informationen über geänderte Zustände von Lichtaktuatoren sein. Zusätzlich zu diesen Events gibt es noch zeitbasierte Events, die einen Trigger auslösen könne. Diese können entweder absolut in Form einer Datums- und Uhrzeitangabe hinterlegt sein oder relativ als Zeitspanne seit dem letzten Event.

In einem üblichen, mit Sensorik und Aktuatorik nach dem neuesten Stand der Technik, ausgestatteten Smarthome entstehen sekündlich bis zu mehrere hundert Events. Daher ist eine Auswahl zu treffen, welche der Events als Trigger fungieren sollen und welche nicht.
Selbst wenn die Auswahl der triggernden Events, die Häufigkeit der Triggerzeitpunkte stark reduzieren kann, so ist die Menge an Events die verbleibt, in der Regel noch in einem Frequenzbereich der sekündlich bis minütlich Berechnungen erzeugt. Daher ist noch eine weitere Hürde einzubauen, bevor aus einem Event ein Trigger für eine Neuberechnung des gesamten, durch Reasoning gefundenen und interagierenden, Berechnungssetups wird.

Zu diesem Zweck ist bei der Berechnung eines Funktionsmoduls anzugeben in welchem Rahmen sich eine Inputvariable verändern darf, bevor eine erneute Berechnung erforderlich ist. Überschreitet diese Änderung die vorgegebenen Grenzen nicht, wird das Event zwar registriert aber nicht zum Trigger einer Neuberechnung berücksichtigt. Dies vermeidet unnötig viele Neuberechnungen. Zusätzlich ist hier die Möglichkeit gegeben durch Methoden der Sensitivitätsanalyse diese Grenzen möglichst sinnhaft zu bestimmen. Dies ist jedoch nicht Betrachtungsbestandteil dieser Arbeit.

Die Information, welches Event eines Automationsservers zu einem Trigger werden könnte, muss hierbei nicht neu erfasst werden. Diese wird bereits durch das Reasoning der Stufe I erfasst. Es

bedarf nun lediglich einer Komponente, die diese Information auswertet und die Liste der triggernden Events aktualisiert. Diese Liste enthält alle Sensoren die mit einer „isConnectedWith“ Eigenschaft im Reasoning der Stufe I versehen wurden. Zusätzlich enthält diese Liste die Information in welchen Grenzen sich der Sensorwert bewegen darf, ohne dass eine Neuberechnung angestoßen wird. Sollten durch mehrere Funktionsmodule Grenzen für denselben Sensor vorgegeben werden, so wird hier das Minimum der Wertänderung eingetragen.

Der Algorithmus zur Triggerdefinition wird nach jedem Reasoningprozess als letzter Schritt ausgeführt. Die in der Liste aufgeführten Sensoren werden dann im Automationsserver als eventerzeugende Sensoren aktiviert. Dadurch werden alle, von diesen Events erzeugten, Nachrichten übergeben, womit sich die Möglichkeit ergibt zu bestimmen, ob die Wertänderung einen Trigger erzeugt. Der Eintrag im Automationsserver erfolgt dabei am Beispiel von FHEM durch Setzen eines Attributes mit der Information, bei welchem Reading des Sensors ein Event erzeugt werden soll. Durch Absetzen eines Befehls in der folgenden Form kann genau spezifiziert werden, bei welchem Reading eines Sensors dies passieren soll.

```
attr <Devicename> event-on-change-reading <Readingname>
```

In bestimmten Fällen kann auch die Information des Reading-Updates von Interesse sein. Wie zum Beispiel bei Sensoren, bei welchen sich der Value des Readings nur selten ändert. Für diesen Fall kann eine Definition als eventgenerierendes Gerät wie folgt erfolgen.

```
attr <Devicename> event-on-update-reading <Readingname>
```

Um beispielsweise bei einer Wetterstation die Werte für Windgeschwindigkeit, Temperatur und relativer Luftfeuchte als Eventgenerator zu aktivieren wäre folgende Zeile auf der Kommandozeile des Automationsservers nach dem Reasoning abzusetzen.

```
attr Wetterstation event-on-update-reading wind,temperature,humidity
```

Um nun noch die angesprochene Einschränkung auf einen Wertebereich zu erzeugen kann mit Hilfe einer sogenannten „Notify-Funktion“ der Wert des Events untersucht werden und im Bedarfsfall das Rechensetup aktiviert werden. Auch diese Verarbeitung kann direkt im Automationsserver erfolgen. Ein solches „Notify“ wird wie folgt definiert:

```
define Trigger_Temperatur notify TEMPHUMBad.*{ if

(ReadingsVal("TEMPHUMBad","temperature","") > 26) { system("Pfad zum Matlab startenden Skript ")} else {  }}
```

Im gezeigten Beispiel wird durch die Untersuchung der Events des Sensors mit dem Namen „TEMPHUMBad“ das Reading „temperature“ überwacht. Wenn dieses den Wert von 26 Grad Celsius überschreitet wird das Skript, welches das Rechensetup aktiviert mittels „system“ Befehl aktiviert.

Besagtes Skript stellt sicher, dass das Rechensetup in gewünschter Form startet und nimmt Einstellungen an dem für die Berechnung gewählten, Framework vor. In diesem Fall startet es eine Matlabinstanz ohne Benutzeroberfläche und übergibt ihr den Befehl zur Ausführung des Rechensetup.

```
#!/bin/bash

tmux send-keys -t matlab "run('/home/alexanderpeikos/LRZ Sync+Share/Rechensetup.m')"$'\n'
```

Bestimmung der Reihenfolge der Modulberechnung

Wie oben bereits beschrieben wurden nach dem mehrstufigen Reasoningprozess nun alle Modul- und Sensorinteraktionen gefunden und geben die Möglichkeit das Rechensetup zu erstellen. Das bedeutet, dass hier eine Programmdatei erstellt wird, die durch das Triggern der Berechnung vollständig durchlaufen wird und für alle inkludierten Module Ergebnisse erzeugt. Damit diese Berechnung reibungsfrei vonstattengehen kann, ist jedoch die Reihenfolge der Berechnung essenziell. Gerade im Fall der Hilfsmodule würde eine zu späte Einreihung der Berechnung bedeuten, dass wichtige Inputdaten für Funktionsmodule noch gar nicht vorliegen, die für die Berechnung benötigt werden. Um die Reihenfolge der Berechnung zu definieren wird mittels einer Schleifenfunktion nach Modulen gesucht, bei denen alle Inputdaten vorliegen und diese berechnet. Nach der Berechnung wird dieser Algorithmus so lange wiederholt, bis alle Module berechnet wurden. Da im letzten Schritt des Reasoningprozesses geprüft wurde, ob Module mit nicht bedienbaren Eingangsdaten vorhanden sind, kann sichergestellt werden, dass alle Module berechnet werden. Diese Herangehensweise ist für die in dieser Arbeit zu zeigenden Sachverhalte ausreichend.

Für eine weitere Verwendung wird jedoch darauf verwiesen, dass gerade im Bereich der Routenberechnung bessere und performantere Algorithmen existieren, um das Problem der Berechnungsreihenfolge zu lösen. Vor allem bei sehr großen Modulkonstellationen kann die Berechnung der Module mit diesem System stark zu Lasten der Performanz gehen.

4.4.4 Ergebnisausgabe und Verarbeitung von Prioritätenarrays

Nachdem die Reihenfolge der Berechnungen geklärt ist, kann die Berechnung selbst durchgeführt werden. Bei einer großen Anzahl an Modulen erhält man demnach auch eine große Anzahl an Ergebnissen. In diesem Fall sind hier die Ergebnisse von besonderem Interesse, welche von Funktionsmodulen ausgeben werden, da diese Einfluss auf die späteren Steueraktionen haben. Zwar müssen Hilfsmodule auch Ergebnisse generieren, diese werden jedoch in Funktionsmodulen wiederverwendet, um zu Steuerungen zu führen. Die Übergabe der Ergebnisse erfordert für die weitere Verarbeitung einige Randbedingungen. Dies hat folgende Gründe:

Es ist davon auszugehen, dass verschiedene Module auch widersprüchliche Wirkanweisungen ausgeben. Für diesen Fall ist ein Verfahren vorzusehen, welches diese Widersprüche ohne das Einwirken eines Experten automatisch lösen kann.
Weiterhin kann eine Kette an Steuerungen auf kurze Sicht logisch erscheinen, in der Zukunft jedoch in eine Situation führen, die umgangssprachlich als „Dilemma“ bezeichnet wird. Es ist eine Methode vorzusehen, die diese Dilemmata vermeidet.

Bei der Konzipierung dieses Übergabealgorithmus wurde sich an der Sichtweise eines Experten orientiert. Dieser ist üblicherweise in der Lage eine Anforderung, die aus einer Berechnung resultiert, so zu formulieren, dass nicht das Schalten eines bestimmten Gerätes gefordert ist, sondern eine Wirkung erzeugt werden soll. Am Beispiel der Schimmelberechnung wäre das möglicherweise:

„Der Luftwechsel muss erhöht werden.“
oder
„Es muss durch heizen die Raumtemperatur erhöht werden.“

Um diese Wirkungsanweisungen zu formulieren werden logische Operatoren verwendet. Eine Auflistung der möglichen Operatoren ist in der folgenden Tabelle zu finden.

Tabelle 26 Logische Operatoren für Wirkungsanweisungen

Bedeutung	**Operator**	**Beispiel**
Größer	>	Temperatur erhöhen
Kleiner	<	Temperatur verringern
Kleiner gleich	≤	Temperatur kleiner größer oder gleich (Ist- oder Sollwert)
Größer gleich	≥	Temperatur größer oder gleich (Ist- oder Sollwert)
Ist gleich	=	Temperatur gleich (Jetzt- oder Sollwert)
Ist ungleich	≠	Temperatur ungleich (Ist- oder Sollwert)
Und	∧	Fensterstatus geschlossen und verriegelt
oder	∨	Fenster gekippt oder offen
Nicht (Negation)	⌐	Feuer nicht vorhanden
Keine Anforderung		Keine Anweisung

Im speziellen Fall ist es auch möglich, diese Anforderungen durch eine Sollwertvorgabe zu versehen. Eine beispielhafte Ausgabe einer Wirkung durch ein Funktionsmodul ist wie folgend angegeben:

Der Luftwechsel muss auf $n = 2$ oder mehr gehoben werden:

$$[n \geq][2 - \infty]$$

Das macht zusätzlich zur Auswahl und Schaltung eines Gerätes jedoch noch ein Modul erforderlich, das diese Anweisung verarbeiten kann. Funktionsmodule sind dadurch allerdings so allgemein gestaltet, dass sie unabhängig von der vorhandenen Automationstechnik verwendet werden können.

Funktionsmodule müssen Steueranweisungen in Form eines Arrays mit einer Zeitskala ausgegeben. Damit ist es möglich, auch für die Zukunft Steueranweisungen zu erzeugen und beispielsweise zu überprüfen, ob das System mit den Steueranweisungen ein Dilemma erzeugt. Dabei stellt sich die Frage, auf welcher Datenbasis die Prognosen erstellt werden. Grundsätzlich ist dies abhängig von der Fragestellung, die das Modul beantwortet. Es gibt Fragestellungen, die problemlos für die Zukunft beantwortet werden können. Am Beispiel der Bewertung der Schimmelgefahr ist das aber nicht der Fall. Für eine adäquate Aussage sind sowohl Informationen über das Gebäudeverhalten notwendig, als auch über das Nutzerverhalten. Dieses spielt eine große Rolle, weil es die Hauptquellen der Feuchtigkeitsproduktion im Raum darstellt. Die Berechnung erfordert also viele Eingangsdaten, die für die Zukunft vorhergesagt werden müssen. Wie diese Daten ermittelt und verarbeitet werden, wird näher im Kapitel 4.4.5 behandelt. Nach Hinzufügen der Zeitskala hat die oben beschriebene Anweisung eines Funktionsmoduls folgende beispielhafte Form.

$$\left\{ \begin{array}{c|c} Jetzt & [n \geq][2 - \infty] \\ 12Uhr & [n \geq][2 - \infty] \\ 13Uhr & [n \geq][2 - \infty] \\ \ldots & \ldots \end{array} \right\}$$

Da Berechnungen als auch Vorhersagen fehlerbehaftet sind, ist zusätzlich zur Ausgabe eine Information über den möglichen Fehlerraum dieser Werte erforderlich, diese ist vor allem bei der Überprüfung auf „Dilemmata“ wichtig und wird in Kapitel 4.4.6 noch näher beschrieben. Ein um die Fehlerbehaftung der Berechnung erweitertes Array würde demnach wie folgt aussehen.

$$\begin{Bmatrix} Jetzt & [n \geq][2 - \infty][\pm 0{,}2\%] \\ 12Uhr & [n \geq][2 - \infty][\pm 0{,}3\%] \\ 13Uhr & [n \geq][2 - \infty][\pm 0{,}4\%] \\ \cdots & \cdots \end{Bmatrix}$$

Entstehen bei der Berechnung mehrerer Zusammenhänge Situationen, bei denen sich widersprüchliche Handlungsanweisungen ergeben, gibt es für einen Experten zwei Möglichkeiten dieses Problem zu lösen.

1. Entsprechend seines Wissens Prioritäten zu setzen und einen Schaden wissentlich in Kauf zu nehmen.
2. Die Entscheidung und damit die Verantwortung auf jemanden zu übertragen der sie besser treffen kann oder der sich aufgrund seiner eignen Entscheidung mit dem Schaden arrangieren kann

In der vorgestellten Methode werden beide Lösungsansätze für unterschiedliche Situationen verwendet. Grundsätzlich sollte versucht werden, dem Algorithmus zu ermöglichen, eine Entscheidung selbst zu treffen. Dafür muss jedoch Wissen über die prioritäre Einstufung der von den Funktionsmodulen erzeugten Handlungsanweisungen vorhanden sein. Um das sicherzustellen muss ein Funktionsmodul für jede, im Array ausgegebenen Handlungsanweisungen, eine zusätzliche Information über die Priorität generieren. Diese wird auf der folgend abgebildeten Skala eingeteilt.

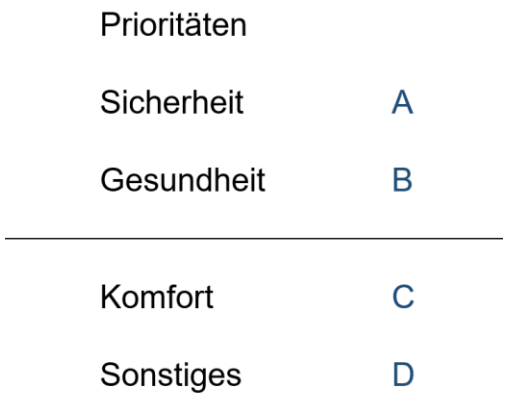


Abb. 35 Prioritäteneinteilung

Die Reihenfolge der Prioritäten ist hier absteigend ihrer Relevanz geordnet. Wobei A- und B-Prioritäten die essenziellen Prioritäten darstellen, die sich aufgrund der Maslowschen Pyramide als Basis definieren lassen. Weitere Prioritäten wie Komfort und Sonstige sind in der Maslowschen Pyramide weiter oben angeordnet.

Dem hier entwickelten Zeit-Wirkungsarray wird also eine weitere Größe in Form der Priorität hinzugefügt. Das gesamte Array hat danach folgende Form:

Tabelle 27 Format des Inhalts des Zeit-Wirkungsarrays mit Priorität

[Wirkgröße und Operator]	[Zielgröße: optional]	[Varianz: +- X]	[Priorität: A…D]

Am oben bereits verwendeten Beispiel gestaltet sich das ganze wie folgt:

$$\begin{Bmatrix} Jetzt \\ 12Uhr \\ 13Uhr \\ \dots \end{Bmatrix} \begin{matrix} [n \geq][2 - \infty][\pm 0{,}2\%][B] \\ [n \geq][2 - \infty][\pm 0{,}3\%][B] \\ [n \geq][2 - \infty][\pm 0{,}4\%][A] \\ \dots \end{matrix}$$

Durch die so generierten Wirkungsanweisungen kann in einem nächsten Schritt eine Priorisierung der Anweisungen und somit ein Filtern der essenziellen, für die Steuerung letztendlich relevanten, Daten erfolgen. Für diesen Schritt werden alle erstellten Wirkungsarrays der verschiedenen Module in eine dreidimensionale Struktur überführt und anschließend verarbeitet. Das können, da es von jedem Modul verschiedene Versionen gibt, mehrere Arrays pro Modul sein. Dazu erhält jedes Modul eine eindeutige Nummer. Jede Version des Moduls bekommt dieselbe Nummer versehen mit einem Index, der die Modulversionen nummeriert.

Das Modul zur Vermeidung von Schimmel habe beispielsweise die Nummer drei. Die Version, die die Erhöhung des Luftwechsels als Wirkgröße ausgibt, erhält den Index eins. Das Wirkgrößenarray erhält damit die Nummer 3.1. Die Version, die die Forderung nach einer stärkeren Beheizung ausgibt, bekommt den Index zwei. Dadurch bekommt das Wirkgrößenarray die Nummer 3.2.

Die Nummerierung der Arrays spielt in Bezug auf den „Dilemma-Check“ später noch eine essenzielle Rolle und wird hier nicht weiter erörtert. Die nun so nummerierten Arrays werden, wie bereits erwähnt, zu einer dreidimensionalen Struktur zusammengesetzt. Dazu werden die verschiedenen „Wirkgrößen“ der Steueraktionen nebeneinander geschrieben. Die gleichen Wirkgrößen von verschiedenen Modulen, werden hintereinander gesetzt. Dabei ist darauf zu achten, dass die Zeitskalen übereinstimmen müssen, näheres dazu kann Kapitel 4.4.5. entnommen werden. Die folgende Tabelle verdeutlicht die Zusammensetzung der eben beschriebenen Struktur und zeigt die Nebeneinanderreihung der verschiedenen Outputs. Auf eine Darstellung expliziter Werte der Outputs wurde aus Gründen der Übersichtlichkeit verzichtet.

Tabelle 28 Generieren des zweidimensionalen Wirkgrößenarrays

	Output 1.1	**Output 1.2**	**Output 3**	**Output 4.1**
Z	X	[…]	[…]	[…]
E	[…]	[…]	[…]	[…]
I	[…]	[…]	[…]	[…]
T	[…]	[…]	X	[…]
	[…]	[…]	X	[…]

Durch das hintereinander Reihen der Outputs mit derselben Wirkgröße wird daraus eine dreidimensionale Struktur, wie sie folgender Abbildung zu entnehmen ist, generiert.

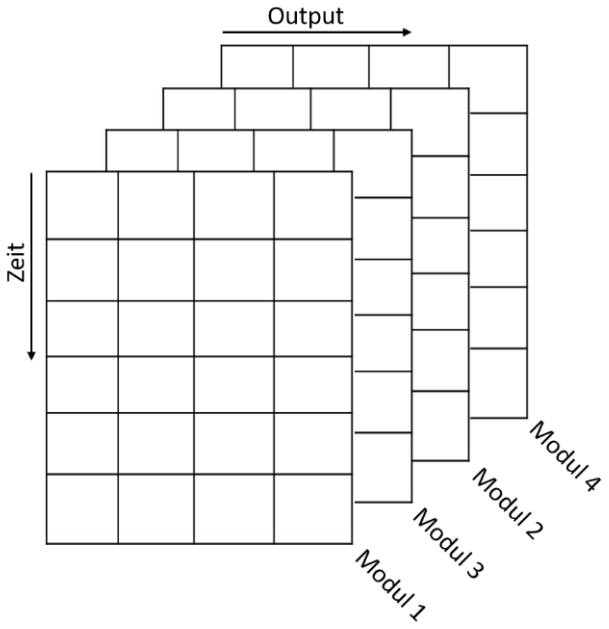


Abb. 36 Zusammensetzen des dreidimensionalen Wirkgrößenarray

Die so hintereinander gereihten Arrays und damit die ganze Struktur kann nun durch das Verarbeiten der Prioritäten wieder in eine zweidimensionale Form gebracht werden. Die dann verbleibende Struktur stellt die, entsprechend der Prioritäten ermittelte, bestmögliche Lösung dar, um den Anforderungen aller Module gerecht zu werden. Für die Verarbeitung der Prioritäten kann grundsätzlich folgendes festgehalten werden:

Wenn die höchste Priorität A oder B ist, ist eine Lösung oder keine Lösung vorhanden.

Wenn die höchste Priorität C oder D ist, ist immer eine Lösung vorhanden.

Das Verarbeiten der Prioritäten erfolgt mittels einfacher Logik. Die Anforderungen werden absteigend gemäß ihrer Priorität aufgelistet und dann für die jeweils oberen beiden Einträge eine gemeinsame Lösung gesucht. Wobei die Anforderung mit der höchsten Priorität immer die Grenzen des Möglichen definiert. Das somit gefundenen Intervall an Lösungen wird nun auf dieselbe Weise mit den folgenden Anforderungen verarbeitet. Das Ergebnis hat dabei immer die Priorität, die das am höchsten eingestuften Element hat, welches als letztes verarbeitet wurde.

Ein grafisches Beispiel soll im Folgenden den Prozess veranschaulichen. Abgebildet sind die Zielgrößen von Wirkgrößenarrays verschiedener Module, benannt mit „Array 1“ bis „Array 4“, sowie deren Priorität in absteigender Reihenfolge. Nach der Bereinigung der Intervalle entsprechend der Prioritäten verbleibt das rechts dargestellte Intervall.

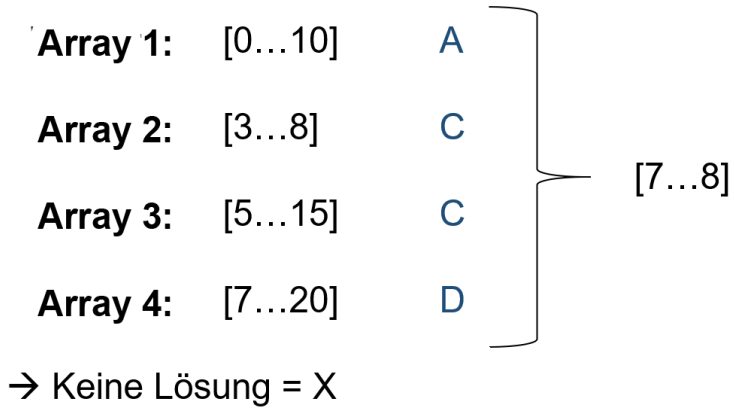


Abb. 37 Prioritätenbereinigung der Wirkgrößen

Die so gefundenen Lösungen werden nun im resultierenden zweidimensionalen Anforderungsarray aufgelistet. Dieses so entstandene Array wird nun darauf überprüft, ob ihre Anweisungen steuerbar sind. Dieses Vorgehen wird in Kapitel 4.4.7 beschrieben und soll hier nicht weiter erläutert werden. Anforderungen, die aufgrund mangelnder Aktionen nicht gesteuert werden können, werden durch Löschen der jeweiligen Spalte aus dem Array entfernt. Damit bleibt das auf Prioritäten bereinigte und steuerbare Anforderungsarray, im Folgenden als Wirkgrößenmatrix bezeichnet, übrig. Grundsätzlich wäre diese nun bereits in einem Format, welches an Wirkmodule zur Steuerung weitergegeben werden kann. Wie bereits erörtert, ist jedoch noch eine Überprüfung nötig, um herauszufinden, ob die Steuerungen nicht zu einer Dilemmasituation führen könnten. Diese Überprüfung wird in Kapitel 4.4.6 behandelt.

Das vorgestellte System der Prioritätenbereinigung kann Prioritäten verschiedener Module untereinander abbilden. Die Abbildung von Prioritäten, die unterschiedliche Nutzer jedoch haben können, sind so nur abbildbar, wenn die Funktionsmodule des DNZ diese auch unterschiedlich behandeln. Das bedeutet beispielsweise, dass die Raumtemperierung für einen alten, gesundheitlich geschwächten Menschen eine höhere Priorität haben muss als für einen erwachsenen gesunden Menschen mittleren Alters. Wenn die DNZ diese Phänomene abbilden, kann das Prioritätensystem diesen Zusammenhang auch verarbeiten. Das schließt auch unterschiedliche Temperierungswünsche von unterschiedlichen Geschlechtern mit ein. Sofern die Vorlieben und die Priorisierung dem System in Form des Moduls vorgegeben werden, kann es diese auch durchführen. In diesem Zusammenhang würden sich auch die im Ausblick beschrieben Methoden des Transfer-Learning anbieten, um solche unterschiedlichen Prioritäten zu lernen und so, die von den unterschiedlichen Nutzern gelebten Prioritäten, zu adaptieren und selbst verarbeiten zu können.

4.4.5 Zeitliche Auflösung der Interaktionsarrays

Üblicherweise werden verschiedene Problemarten auf verschiedenen Zeitskalen betrachtet und ausgewertet. So betrachten CFD Simulationen Probleme auf einer Zeitskala im Sekundenbereich. Andererseits gibt es auch Algorithmen die Probleme mithilfe von Monatsbilanzen betrachten. Eine Vereinheitlichung auf eine festgelegte Skala würde weder den betrachteten Problemen, noch dem

hier vertretenen „Open-World“-Ansatz gerecht werden. Daher wurde eine Lösung angestrebt, die es zulässt, dass Module die Auflösung ihrer zeitlichen Skala entsprechend ihres LODs wählen.
Beim Erstellen der oben beschriebenen Prioritätenarrays werden also Module mit einer hohen zeitlichen Auflösung mit denen vermischt, die eine geringe zeitliche Auflösung haben. Bei reinem Spreizen der gering aufgelösten Arrays ergeben sich dadurch Lücken, in denen für diese Zeitpunkte keine Werte vorhanden sind. Da dies auch einen Fehler durch die Nichtbeachtung dieser Outputs mit sich bringen würde, müssen diese Lücken gefüllt werden.
Die einfachste Methodik, um diese Lücken zu füllen, sind Sprungfunktionen. Die besagten Lücken werden solange mit den vorhergehenden Werten aufgefüllt, bis ein neuer Wert vorhanden ist. Dieses Vorgehen kann für verschiedene Probleme durchaus realitätsnah sein. Als Alternative könnten die fehlenden Werte auch zwischen dem letzten und dem nächsten Datenpunkt linear interpoliert werden. Grundsätzlich existieren jedoch noch viele weitere Möglichkeiten um eine Interpolation durchzuführen. Die Entscheidung, mit welcher Methodik diese Lücken gefüllt werden müssen, ist stark von der Art des Problems abhängig, dass das jeweilige Funktionsmodul verarbeitet. Eine einheitliche Lösung kann hier also nicht gefunden werden ohne für verschiedene Funktionsmodule ein beträchtliches Abweichen von der Realität zu generieren. Aus diesem Grund wird die Wahl der Interpolationsmethode zum Auffüllen der Lücken dem jeweiligen Modulentwickler überlassen. Die Methode der Interpolation muss daher, für den Gesamtalgorithmus zugänglich, in der Ontologie hinterlegt werden.
Sobald diese Information vorhanden ist, kann an der jeweiligen Stelle, an welcher die Arrays zusammengesetzt werden, das Array mit der höchsten zeitlichen Auflösung gesucht werden und alle anderen Arrays entsprechend dieser Vorgabe und der Information der Interpolationsmethode erweitert werden. Die Wahl der Interpolationsmethode betrifft hier nicht nur die Zielwerte, sondern natürlich auch die Auswahl der Prioritäten und der Varianzen. Die Einschätzung der Genauigkeit einer Berechnung kann im Zweifel nur ein Experte durchführen, daher ist ihm diese Aufgabe direkt bei der Modulentwicklung zu überlassen.

Ein weiteres Problem, das die Zeitskala mit sich bringt, ist die Interaktion von Modulen. Wie bereits oben beschrieben, müssen Module im hier gezeigten Ansatz ihre Anforderungen an Steuerungen oder Änderungen als Zeitskala für die Zukunft ausgeben. Dies kann zur Folge haben, dass für die Beschreibung der zukünftigen Werte Algorithmen genutzt werden, die Simulationscharakter haben. Um diese mit Informationen zu versorgen, gibt es mehrere Möglichkeiten.

Ein einfacher Ansatz ist ein statischer, der davon ausgeht, dass sich die Randbedingungen in der Zukunft nicht verändern. Dieser Ansatz ist in der Umsetzung der Fallstudien bereits berücksichtigt, da hier für eine in die Zukunft gerichtete Kalkulation keine weiteren Daten erforderlich sind. Das Funktionsmodul kann seine Ausgaben, wie auch in den Fallstudien dieser Arbeit, ohne neue Daten erstellen und eine Berechnung der Ausgaben kann entsprechend stattfinden.
Ein teildynamischer Ansatz wäre auch denkbar. Dieser beschreibt aufgrund des gelernten Verhaltens der Umgebung die aus den Steueranweisungen resultierenden Reaktionen und verwendet diese wieder als Eingangsgrößen um den folgenden Zeitschritt zu berechnen. Diese Methode erfordert letztendlich nur das iterieren der Daten innerhalb eines Funktionsmoduls oder eines kleinen Bereichs von Modulen, die aber aufgrund nicht definierter Schnittstellen vom selben Entwickler stammen müssen. Die Iteration findet also abseits des hier gezeigten Algorithmus statt.

Der dritte Ansatz wäre ein voll dynamischer Ansatz. Dieser bedeutet, dass die aus den Wirkgrößen resultierenden Änderungen im Gebäude als Datenstrom wieder zurück in die Inputvariablen des Systems laufen. Damit könnte eine zeitschrittbasierte Simulation abhängig vom Umfang des LOD der Funktionsmodule durchgeführt werden. Der Ablauf der Simulation würde im weitesten Sinne dem Vorgehen in der Dissertation von Mitterhofer (2017) entsprechen.

Um diese Funktionalität gewährleisten zu können ist es jedoch notwendig, die im ersten Teil des Reasoningprozesses gefundenen Verbindungen zwischen Modulen und Sensoren durch zukunftsbezogene errechnete Daten zu ersetzen. Zu diesem Zweck müssen für Simulationsaufgaben erstellte Wirkmodule einen ODS erzeugen, der keine Steuerung bezweckt, sondern eine Aussage über die durch Steueraktionen resultierende Veränderung im Gebäude macht.
Um diese Aussage treffen zu können sind jedoch mehr Informationen nötig, als sie von den Funktionsmodulen in Form der Wirkgrößen ausgegeben werden. Für eine hygrischen Simulation der Raumluft sind für die Lösung der Bilanzgleichung beispielsweise Informationen über Quellen und Senken notwendig. Diese können von Hilfsmodulen im DGZ oder DNZ zur Verfügung gestellt werden. Das Wirkmodul benötigt zur Lösung dieser Gleichung daher Eingangsdatenströme in Form von Quellen und Senken, um die Bilanz für jeden Zeitschritt zu erstellen und die simulierten Raumparameter als Ausgangsdatenstrom zurückgeben zu können. Damit wäre ein voll dynamischer Ansatz mit Simulationsumgebungen und der Einbindung von Methoden des maschinellen Lernens möglich.

Auf eine Umsetzung eines volldynamischen Beispiels wurde in dieser Arbeit aus folgenden Gründen verzichtet:

- An der Funktionalität der ontologiebasierten Algorithmik ändert sich für diesen Ansatz nichts
- Ein Framework für diese Berechnungsmethode besteht noch nicht und muss erst entwickelt werden. Eine Entwicklung wäre aber auf Basis diverser Vorarbeiten, wie der Arbeit von Mitterhofer (2017), möglich
- Die Funktionalität von Simulationen mit schrittweisem Datenaustausch zwischen den Modulen wurde durch Mitterhofer (2017) bereits nachgewiesen

Da also von einer Funktionsfähigkeit ausgegangen werden kann, wird auf eine exemplarische Umsetzung verzichtet. Um die angesprochene Verarbeitung von IDS bei Wirkmodulen zu zeigen, wurde die im Folgenden noch beschriebene Fallstudie 3 eingeführt. In ihr wird die Integrierbarkeit von nutzerindividualisierenden Algorithmen demonstriert.

Die beschriebene resultierende Wirkgrößenmatrix kann nun nach lösen der „Zeitskalenproblematik“ unabhängig davon ob durch statische Berechnung oder volldynamische Simulation entstanden, bereits zur Steuerung des Gebäudes verwendet werden. In einem letzten Schritt muss jedoch noch für mehrfache Lösungswege, in Form von Versionen des Funktionsmoduls, ein geeigneter Lösungsweg gefunden werden. Hierfür bietet sich an, die Tiefe der zu Beginn gebildeten dreidimensionalen Struktur an den Stellen der verbleibenden unterschiedlichen Lösungswege zu bestimmen und den Weg mit der geringsten Tiefe zu wählen. Dies hat den Hintergrund, dass üblicherweise nur Modulversionen fremder Funktionsmodule bei der Verarbeitung zu dieser dreidimensionalen Struktur hintereinandergelegt werden. Mit der Tiefe dieser Struktur kann dementsprechend eine Aussage über die Menge der betroffenen Module gemacht werden.
Eine Veränderung im Einflussbereich vieler Module kann den Effekt einer höheren Ungenauigkeit bei Berechnungen in der Zukunft haben und damit zu einem öfteren Anpassen der Regelstrategie führen. Um das zu verhindern, ist der Lösungsweg mit der geringsten dreidimensionalen Tiefe zu wählen. Dieser hat augenscheinlich den geringsten Einfluss auf das Gesamtsystem und damit auch den niedrigsten Einfluss auf andere Funktionsmodule. Zusätzlich zu dieser Methode können jedoch auch einzelne Versionen als präferiert markiert werden. In diesem Fall kann der Modulentwickler unabhängig von den Auswirkungen auf das Gesamtsystem angeben, dass eine bestimmte Wirkung bevorzugt werden sollte. Am Beispiel der Schimmelvermeidung wäre das eine Erhöhung des Luftwechsels, da sie üblicherweise den größten Effekt erzielt.

4.4.6 Durchführen eines „Dilemmachecks“

Wie bereits beschrieben besteht die Gefahr, dass durch eine Steuerung zwar augenscheinlich richtiger Aktionen in der Zukunft ein Zustand entsteht, der eine adäquate Steuerung nicht mehr zulässt oder zu einem Schaden führt. Um das zu verhindern ist bereits im Vorfeld zu prüfen, ob die aktuelle Steuerstrategie zu einem Schaden führen kann. Das hier entwickelte System an Arrays stellt hierfür die Werkzeuge bereit. Da die nach Prioritäten bereinigte Struktur bereits eine Zeitskala in entsprechend der Problemstellung detaillierter Auflösung besitzt, kann diese Skala auf ein anstehendes Dilemma untersucht werden.
Zu diesem Zweck wird nach der Prioritätenbereinigung der Arrays geprüft, ob in der resultierenden Wirkgrößenmatrix in einer Spalte ein X steht. Wobei das X dafür steht, dass die Prioritäten es nicht zugelassen haben hier eine adäquate Lösung zu finden. Dies kann der Fall sein, wenn sich zwei Anforderungen mit der Priorität A oder B widersprochen haben. Sollte in einer Spalte ein X vorhanden sein, so ist dieser Weg aufgrund gegenläufiger wichtiger Interessen von Funktionsmodulen zu verwerfen. Beispiele für so einen Fall wurden in folgender Grafik rot ausgekreuzt.

	Output 1.1	Output 1.2	Output 3	Output 4.1	Output 4.2
Z	X	[…]	[…]	[…]	[…]
E	[…]	[…]	[…]	[…]	[…]
I	[…]	[…]	[…]	[…]	[…]
T	[…]	[…]	X	[…]	[…]

Abb. 38 Zu verwerfende Lösungsansätze wegen Gegenläufigkeit

Das Auskreuzen eines Lösungsansatzes ist aber noch nicht direkt ein Zeichen für eine Dilemmasituation. Da jedes Modul mehrere Versionen von Lösungsansätzen haben kann, ist nicht ausgeschlossen, dass ein anderer Lösungsweg „dilemmafrei“ verläuft. Daher ist im letzten Schritt zu prüfen ob für jedes Funktionsmodul, nach dem Streichen der „dilemmabehafteten“ Lösungswege, noch ein Lösungsweg vorhanden ist. Falls ja, ist dieser zu präferieren und die problematischen Lösungen zu streichen. Ist kein Lösungsweg mehr für ein Modul übrig, befindet man sich in einer Situation, die in der Zukunft ein Dilemma erzeugen könnte und die Lösung des Problems ist dem Nutzer oder einem Experten zu überlassen, der die Risiken besser einschätzen kann als das System. In allen anderen Fällen kann davon ausgegangen werden, dass die bestmögliche Lösung gefunden wurde.

Grundsätzlich bestünde die Gefahr, dass bei einer zukunftsorientierten Berechnung, bei diskreten Werten, rein statistisch immer ein Dilemma für jeden der Lösungswege auftritt. Da aber für die Ausgabe der Anforderungen von Funktionsmodulen eine Angabe bezüglich der Genauigkeit in Form der Varianz gefordert ist, ist es möglich dieses Szenario auszuschließen. Bei Einbezug der Varianz werden die diskreten Graphen zu Wertebereichen. Die daraus resultierenden Überschneidungsbereiche der Lösungen werden mit genug Abstand zum aktuellen Zeitpunkt so groß, dass immer eine Lösung vorhanden sein wird. Die folgende Grafik verdeutlicht diesen

Zusammenhang am Beispiel zweier beliebiger Outputvariablen und deren Varianz. Je weiter die Prognose in der Zukunft liegt, desto größer werden für jede Variable die Lösungsbereiche bis letztendliche kein Bereich ohne gemeinsame Lösung mehr existiert. Dilemmata würden auftreten, wenn es in einem Bereich keine gemeinsame Lösung gibt. Dieser Zusammenhang wurde am Anfang der Grafik dargestellt.

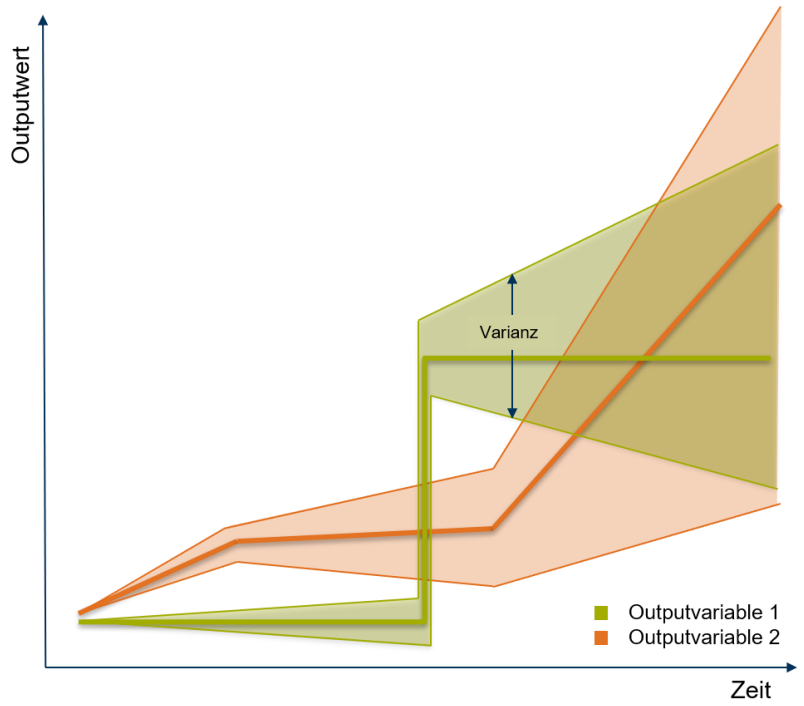


Abb. 39 Einfluss der Ungenauigkeiten auf die Überprüfung auf Dilemmata

4.4.7 Geräte und Aktionsauswahl

Nachdem die Anforderungen der Funktionsmodule prioritätenbereinigt wurden und alle unbrauchbaren Wirkgrößen entfernt wurden, müssen die zu der resultierenden Struktur passenden Steueraktionen gefunden werden. Dies passiert durch die ontologiebasierte, reasoninggestützte Zuordnung der Wirkgrößen zu Wirkmodulen. Auch die Anbindung der im Smarthome vorhandenen Geräten an die Wirkmodule erfolgt durch das Reasoning.
Damit dieses Reasoning stattfinden kann müssen die von einem Wirkmodul steuerbaren Wirkungen in der Ontologie hinterlegt sein. Zusätzlich benötigen auch die Geräte eine Beschreibung ihrer zuordenbaren Wirkung. Dann ist es möglich, die Wirkgrößen direkt einer geeigneten Steuerung zuzuweisen.
Bei der Definition der Module sind daher Eigenschaftsverbindungen zu den in Kapitel 4.4.2 genannten Klassen zu erstellen. Bei der Definition der Geräte ist zusätzlich abzulegen, welchen Einfluss Steueraktionen auf den Wohnraum erzeugen. Das kann am Beispiel einer dimmbaren Lampe die Möglichkeit eines Dimmens in Schritten von 0 bis 100 % sein. Passend zu dieser Steuermöglichkeit müssen auch Informationen über die Wirkung, in Form des resultierenden Lichtstroms der Lampe, bei einer gewissen Prozentzahl hinterlegt werden. Es muss also eine Angabe über die Kapazität des Gerätes in Bezug auf eine Wirkung erfolgen.
Mit dieser Information kann die Funktion des Wirkmoduls mit der Funktion eines Treibers in einem Betriebssystem verglichen werden. Der Algorithmus verlangt beispielsweise eine Beleuchtungsstärke von 500 Lux. Der Treiber des Leuchtmittels ermittelt, dass diese bei 45 % Dimmstärke gewährleistet werden kann und schaltet das Leuchtmittel auf 45%. Somit kann sichergestellt werden, dass sowohl

das richtige Gerät durch die Auswahl der zugeordneten Quantität als auch die richtige Aktion durch den „Treiber“ ausgeführt wird. Für den Fall, dass mehrere Geräte im selben Einflussbereich das Ziel erreichen können, sind folgende Strategien denkbar:

- Ein Gerät vollständig auslasten und erst daraufhin andere Geräte aktivieren
- Alle Geräte möglichst gleichmäßig auslasten
- Nur eine spezielle Kombination aus Geräten aktivieren.

Um dieses Problem zu lösen, ist eine weitere Logikschicht notwendig, die mehrere Geräte zu einer Gruppe verknüpft. Durch die generische Beschreibung der Geräte stellt diese Gruppierung aber keine Schwierigkeit dar und kann vom Wirkmodul übernommen werde. Durch diese koordinative Zwischenschicht ist es möglich, Gruppen zu definieren und die Umsetzung der Steueraktionen den persönlichen Bedürfnissen von Nutzern anzupassen. Sollten für diese Tätigkeit und zur Berücksichtigung der Individualisierung Eingangsdaten aus dem DNZ erforderlich sein, so können diese mithilfe der Ontologie auf die bereits bekannte Weise mittels Reasonings gefunden werden.

Durch das Reasoning von Gerätezugehörigkeiten zum Wirkmodul ist es auch möglich Abfragen zu generieren, ob für Moduloutputs überhaupt Geräte vorhanden sind, die in der Lage sind diese Wirkgrößen zu gewährleisten. So kann entweder keine Beeinflussung wegen fehlender Hardware erfolgen oder weil die vorhandenen Geräte keine freien Kapazitäten mehr zur Verfügung haben. Sollten beispielsweise bereits alle Lüftungsgeräte auf höchster Stufe laufen, so kann auch keine weitere Steigerung des Luftwechsels durch sie generiert werden. Sind keine Möglichkeiten gegeben eine Wirkgröße zu manipulieren, ist es auch nicht sinnvoll die jeweiligen Funktionsmoduloutputs in der Auswahl der möglichen Steuerungen zu belassen. Wie bereits beschrieben müssen diese Anforderungen dann aus der Steuerstruktur entfernt werden. Die Umsetzung kann durch das automatische Generieren einer Eigenschaft in der Ontologie geschehen, welche ein Wirkmodul als handlungsunfähig ausweist. Diese Eigenschaft muss daher von dem jeweiligen Modul selbst generiert werden. Die Randbedingungen für die Entscheidung einer solchen Maßnahme sind für den jeweiligen Fall vom Modulentwickler zu bestimmen. Grundsätzlich ist jedoch eine Schnittstelle vorzusehen, durch welche Wirkmodule die Möglichkeit haben, dem Gesamtalgorithmus das Setzen dieser Eigenschaft anzuweisen. Durch diese Maßnahme ist eine einfache Abfrage der Kapazität einer Wirkung möglich, bevor die aus der Prioritätenbereinigung resultierende Struktur verarbeitet wird. Hierdurch können nicht steuerbare Outputs entfernt werden.
Sollte das Streichen von Lösungen dazu führen, dass für ein Modul keine Lösungswege mehr vorhanden sind, so ist dies der Zeitpunkt, an dem die Verantwortung für die Handlung an den Nutzer abgegeben werden muss. Dies kann in Form einer akustischen Benachrichtigung stattfinden.

Nach der nun vollständigen Beschreibung aller Komponenten und Abläufe in der entwickelten Methode, wird deren Funktionsweise evaluiert. Zu diesem Zweck erfolgte eine Implementierung des Konzeptes in eine bereits bestehende automatisierte Wohnung.

5 Implementierung

Im Folgenden werden Details zur praktischen Umsetzung des bereits theoretisch erläuterten Ansatzes dargestellt. Dabei werden der jeweilige technische Teilbereich, der zur Erzeugung eines Testframeworks notwendig war, sowie die jeweils generierten Module kurz aufgezeigt und erläutert. Am Ende dieses Kapitels entsteht somit ein Überblick über die technische Komplexität und die einzelnen Komponenten des Testframeworks.

5.1 Datenbankstrukturen für die zentralisierte Datenverwaltung

Die zentrale Verwaltung von Daten wird in einer Datenbank gelöst. Dies betrifft alle Formen von Daten, die nicht ontologierelevant sind. Das beinhaltet Sensordaten, Nutzerparameter, Aktuatordaten und alle sonstigen Formen von Daten, die nicht direkt aus der Ontologie ausgelesen werden, sondern im Verlauf der Berechnung von einem Datensammlermodul abgerufen werden. Grundsätzlich empfiehlt es sich für die Datenhaltung ein einheitliches System zu verwenden. Übliche Datenbanksysteme weisen inzwischen eine weite Verbreitung auf und sind dementsprechend leicht zu implementieren und gut dokumentiert.

Innerhalb dieser Datenstruktur ist auch die bereits angesprochene Liste anzulegen, mit deren Hilfe im Sinne der „Functional Layout Modularity“ ein Zusammenhang zwischen den hier gebildeten Modulen und den Funktionen des klassischen Automationsansatzes hergestellt werden kann. Damit kann es dem Nutzer ermöglicht werden, einzelne Funktionen zu aktivieren und zu deaktivieren, ohne dabei Rücksicht auf modul-bedingte Abhängigkeiten nehmen zu müssen.

5.1.1 Gebäudedaten und Einbindung von BIM

Grundsätzlich existieren Möglichkeiten Daten aus BIM-Modellen zu transferieren. Für die hier aufgeführten Anwendungsfälle wurden Daten händisch ermittelt und implementiert. Dies hatte den Hintergrund, dass von der in den Fallstudien verwendeten Testwohnung kein BIM-Modell vorhanden ist. In einem BIM-Modell sind zwar alle wesentlichen Daten vorhanden, die für eine genaue Beschreibung eines Gebäudes erforderlich sind, diese sind jedoch nur maschinenlesbar im Sinne dieser Arbeit, wenn bestimmte Informationen davon in die Ontologie und auch in der Datenbank hinterlegt werden. Aus diesem Grund ist für einen Produktivbetrieb eine Software notwendig, die die jeweiligen Daten extrahiert und in den jeweiligen Bereichen abspeichert. Da ein regelmäßiges Umbauen des Gebäudes eher unüblich ist, betrifft dieses Problem die Ersteinrichtung eines Systems und nicht den Produktivbetrieb. Ein Beispiel für eine funktionsfähige Konvertierung eines IFC Modells für eine ontologiegestützte Simulation wurde bereits von Mitterhofer (2017) erstellt. Daher wird hier auf diese Umsetzung verzichtet.

5.1.2 Nutzerdaten

Da jeder Bewohner sein Gebäude unterschiedlich nutzen will, gilt es auch Spielraum für verschiedene Nutzerpräferenzen bei der Steuerung zu erzeugen. Dieser Spielraum ist bei den Funktionsmodulen vonseiten des Modulentwicklers einzuplanen und die zugehörigen Parameter, die die Nutzerpräferenzen beschreiben können, sind in der Datenbank abzulegen. Durch eine klare Trennung der Datenbanken für jeden Nutzer und jedes Gebäude, ist es somit möglich, dass auch im Falle eines Umzuges eines Nutzers die persönlichen Daten und angelernten Algorithmen übertragen werden. Dadurch müssen mit einer neuen Wohnung nicht alle maschinelle Lernprozesse von Neuem begonnen werden. Um das gewährleisten zu können, ist jedoch datentechnisch eine strikte Trennung zwischen nutzerrelevanten und gebäuderelevanten Daten erforderlich. Die Trennung in DGZ und DNZ ist also auch in Form der Datenhaltung weiter einzuhalten.

Der DNZ muss daher aus Datensicht folgendes enthalten:

- Gelernte Parameter
- Position und Bewegungsprofile des Nutzers mit Metainformationen zum Gebäudekontext
- Aktivitäten des Nutzers
- Vorhersagen von Aktivitäten und dazugehörige Randdaten
- Weitere singuläre Nutzerparameter

5.2 Generierung der Module

Für den Test der Funktionsfähigkeit wurden die Module erstellt, welche in den nachfolgenden Fallstudien benötigt werden. Auf deren Erstellung wird im Folgenden weiter eingegangen.

5.2.1 Konzept zur Modulerstellung

Die verschiedenen Modultypen wurden mit Hilfe der Programmiersprache Matlab in der Version 2017b mittels Funktionen umgesetzt. Die Module wie auch der übergreifende Algorithmus wurden dafür eigens erstellt und alle Module in ihrem Aufbau und ihrer Funktion an das hier vorgestellte Konzept angepasst. Das gewählte Framework lässt auch eine Implementierung von Modulen im FMI-Standard zu. Dies wurde im Rahmen einer Masterarbeit von Anderlik (2017) bereits nachgewiesen. Module wurden derart implementiert, dass trotz Sichtbarkeit des Quellcodes eine Verwendung im Sinne eines verkapselten Quellcodes möglich ist. Entwickelte Module enthalten daher keine auf die Fallstudien zugeschnittenen Programmabläufe oder Einschränkungen. Nähere Informationen zu implementierten Funktionen oder dem Programmablauf können der Dokumentation im Anhang entnommen werden. Um den automatischen Aufruf und das Ansprechen der Module gewährleisten zu können, wurden daher auf Seiten der Ontologie die in folgender Tabelle aufgelisteten zusätzlichen Informationen in Form einer „Data Property“ implementiert.

Tabelle 29 Verarbeitungsinformationen in Dateneigenschaften für Module

Information	**Dateneigenschaften**
Speicherort der Funktion	ModulLocation
Syntax zum Aufrufen der Funktion	ActivationSyntax
Reihenfolge der zu übergebenden Inputvariablen	InputOrder
Zu bevorzugender Moduloutput	PreferredOutput

Im Folgenden werden die in der jeweiligen Klasse implementierten Module mit ihrer Funktion aufgeführt.

5.2.2 Gebäudemodule

In der Kategorie der Gebäudemodule wurde folgende Algorithmen abgebildet

- Modul zur Verhinderung der Schimmelbildung in Wohnräumen
- Modul zur Kalkulation der Oberflächentemperatur mit Hilfe der Raum- und Außenlufttemperatur sowie dem U-Wert des Bauteils.
- Modul zur Kalkulation der Oberflächenfeuchte mittels der Oberflächentemperatur und der Raumluftfeuchte
- Modul zur Beschreibung des hygrothermischen Raumfeedbacks auf Nutzeraktionen
- Erweiterung um Ausgabe des hygrischen Raumfeedbacks auf Nutzeraktionen in Form von relativer Luftfeuchte

5.2.3 Nutzermodule

In der Kategorie der Nutzermodule wurde folgendes implementiert:

- Anforderung des Nutzers an das Nicht-Vorhandensein von Schimmel
- Anforderung des Nutzers an die Radonkonzentration
- Algorithmus zur Bestimmung der Anwesenheit des Nutzers
- Algorithmus zur Vorhersage des Duschverhaltens eines Nutzers

5.2.4 Funktionsmodule

Entsprechend der beschriebenen Systematik sind folgende bereits genannten Module als Funktionsmodule einzustufen:

- Modul zur Verhinderung der Schimmelbildung in Wohnräumen
- Anforderung des Nutzers an das Nicht-Vorhandensein von Schimmel
- Anforderung des Nutzers an die Radonkonzentration

5.2.5 Hilfsmodule

Nachfolgend aufgelistete Hilfsmodule wurden umgesetzt:

- Modul zur Kalkulation der Oberflächentemperatur mit Hilfe der Raumluft- und Außenlufttemperatur sowie dem U-Wert des Bauteils.
- Modul zur Kalkulation der Oberflächenfeuchte mittels der Oberflächentemperatur und der Raumluftfeuchte
- Algorithmus zum Bestimmen der Anwesenheit des Nutzers
- Datensammlermodul
- Algorithmus zur Vorhersage des Duschverhaltens eines Nutzers

5.2.6 Wirkmodule

Die folgend aufgezählten Zusammenhänge wurden als Wirkmodule umgesetzt:

- Luftwechselsteuerung zur Geräte- und Aktionsauswahl
- Erweiterung um Ausgabe des hygrischen Raumfeedbacks auf Nutzeraktionen in Form von relativer Luftfeuchte
- Heizungssteuerung zur Geräte- und Aktionsauswahl

5.2.7 Module für maschinelles Lernen und Vorhersagen

Die Verwendung von Algorithmen des maschinellen Lernens wurden anhand der Erkennung und der Vorhersage von Duschereignissen simuliert.
Eine konkrete Umsetzung mittels eines „Deeplearning“ Algorithmus war jedoch im Rahmen dieser Arbeit nicht möglich, da wie bereits beschrieben, für ein Grundmodell eine hohe Datenmenge bestehen muss. Im Bereich der Bauphysik und der demonstrierten Beispiele existieren aber kaum frei zugängliche Datensammlungen, um Ansätze für transferlernende „Deep-Learning“ Algorithmen zu testen.
Um trotzdem die Funktionsfähigkeit und die Möglichkeit einer Einbindung solcher Algorithmen nachweisen zu können, wurde ein Modul entwickelt, das mittels herkömmlicher Musterkennung arbeitet. In diesem wurde ein Algorithmus zum maschinellen Lernen implementiert und ist im Sinne der ontologischen Verwendung wie ein „Deep-Learning“ Algorithmus verwendbar. Der genaue Programmablauf, der im Bereich der Modultechnik als „verkapselt“ bezeichnet werden kann, wurde so generiert, dass brauchbare Vorhersagen zur Duschhäufigkeit eines Nutzers generiert werden konnten.

5.3 Algorithmen zur Erkennung von notwendigen Interaktionen

Um die bereits erläuterte automatische Zuweisung von Datenströmen zu gewährleisten und damit auch die Interaktion zwischen verschiedenen Modulen abzubilden, sind verschieden Algorithmen und Systeme notwendig. Deren Umsetzung im für diese Arbeit entwickelten Testframework wird in den folgenden Kapiteln näher erläutert.

5.3.1 Ontologiebasierte Algorithmen

Die Erstellung von Ontologien und deren Reasonings ist mit kostenfreien Softwarelösungen möglich. Bei der Erstellung der Ontologie für die Durchführung der Fallstudien wurde darauf geachtet, dass nur Daten angelegt wurden, die automatisch mit Hilfe der richtigen Algorithmik angelegt werden können. Somit kann alternativ ein Beschreiben der Ontologie auch mit einer externen Software erfolgen. Dieser Weg wurde von Mitterhofer (2017) mit dem „FMUontologyXtender“ gezeigt. Aufgrund des vergleichsweise geringen Umfangs der Ontologie für die Testzwecke dieser Arbeit, wurde auf eine aufwändige Programmentwicklung verzichtet und die Ontologie händisch erstellt.

Das Auffinden der richtigen Sensoren, Aktuatoren und Modulverbindungen kann mithilfe von SWRL-Regeln erfolgen, da die dafür notwendigen Informationen in Form von Eigenschaften hinterlegt sind. Lediglich für das Auslesen zusätzlicher Informationen, wie dem Speicherort für Sensorwerte in der Datenbank oder der Ansteuerung der verschiedenen Aktuatoren, müssen zusätzlich Daten in Form von beispielsweise Pfadangaben in der Ontologie hinterlegt werden. Diese Aufgaben lassen sich jedoch leicht automatisch von Softwaresystemen in Form eines Installationsskriptes umsetzen.

5.3.2 Auswahl und Anbindung des Reasoners

Bei der Wahl eines Reasoners ist auf dessen Kompatibilität zur erstellten Ontologie zu achten, zudem muss die Möglichkeit zur Verarbeitung von SWRL-Regeln enthalten sein. Eine Übersicht über alle gängigen, derzeit verfügbaren Reasoner mit ihrem Funktionsumfang stellt die „University of Manchester“ zur Verfügung (University of Manchester, 2018).

Wenn ein ausgewählter Reasoner die hier beschrieben Aufgaben, vor allem die Verarbeitung von SWRL-Regeln, unterstützt, ist der Funktionsumfang im Sinne der vorgestellten Methodik

ausreichend. Für das Prüfen der Methodik an sich spielen weder Performance noch andere Aspekte eine wesentliche Rolle. In einem produktiven System müssen solche Aspekte jedoch durchaus betrachtet werden. Daher sollte bei einer Weiterentwicklung auf die Auswahl eines geeigneten Reasoners besonderer Wert gelegt werden.

5.3.3 Interaktion mittels Prioritätenarrays

Die Abbildung der Interaktion und Prioritätenarrays wurde vollständig, wie bereits beschrieben, im Rahmen dieser Arbeit umgesetzt. Die Algorithmik zur Anpassung der Zeitskala wurde nur soweit bearbeitet, wie dies für die hier erforderlichen Daten im Rahmen der folgenden Fallstudien nötig war. Ein automatisches Anpassen ist für einen vollständigen Algorithmus unerlässlich. Grundsätzlich unterstützt MATLAB aber, alle benötigten Operationen und ist somit für diese Aufgaben geeignet. Der implementierte Ablauf zur Bereinigung der Arrays entsprechend ihrer Prioritäten ist in nachfolgender Grafik in Form eines Ablaufdiagrammes dargestellt.

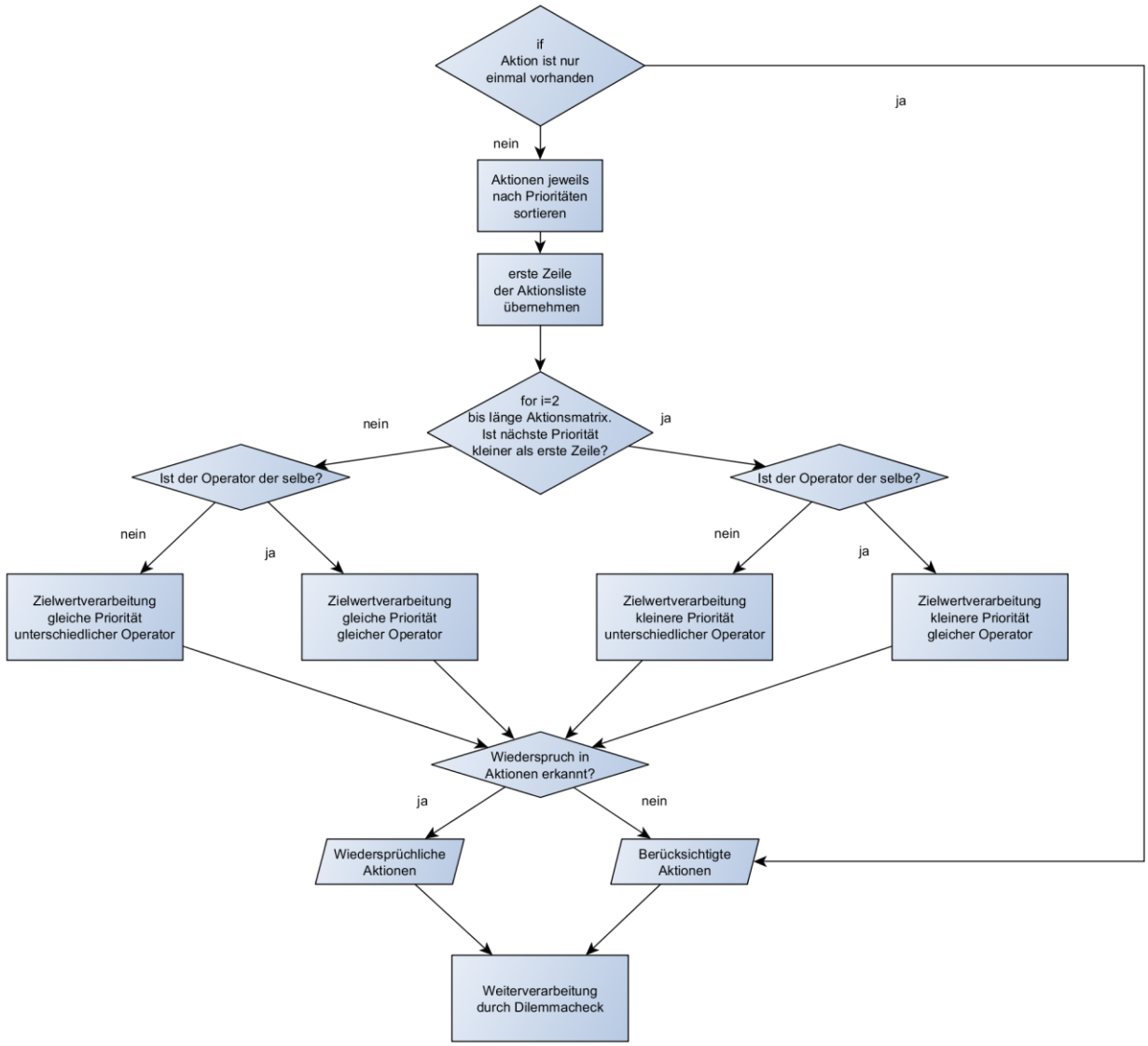


Abb. 40 Programmablauf Prioritätenbereinigung

5.3.4 Berücksichtigung zeitlichen Auflösung in den interagierenden Arrays

Die Berücksichtigung von verschiedene Zeitskalen bei Ein- und Ausgangsdatenströmen von Modulen, sowie deren automatisiertes Angleichen durch Interpolationsinformationen wurde in Form des Datensammlermoduls umgesetzt. Die dafür notwendigen mathematischen Operationen stellen Grundfunktionen der Programmierplattform MATLAB dar. Die benötigten Informationen zu Randbedingungen der Implementierung konnten aus der Ontologie mit Hilfe der Reasoner ausgelesen werden. Die folgende Grafik zeigt den grundsätzlichen schematischen Ablauf des Datensammlermoduls.

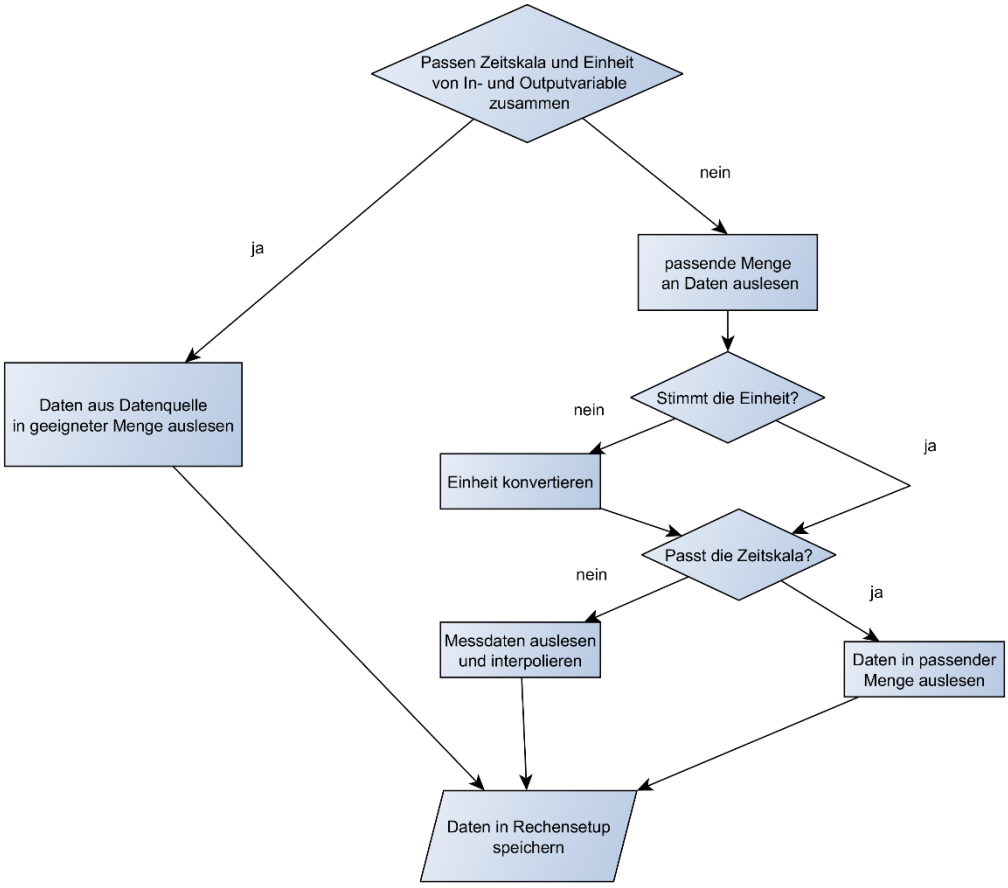


Abb. 41 Programmablauf Datensammlermodul

5.3.5 „Dilemmacheck“ bei der resultierenden prioritätenbereinigten Wirkgrößenmatrix

Die Umsetzung der beschriebenen Vorgehensweise zur Überprüfung auf eventuell auftretende Dilemmata wurde in MATLAB umgesetzt. Die beispielhafte Umsetzung wird in der Fallstudie 2 demonstriert. Da MATLAB grundsätzlich für mathematische Operationen und damit auch für den Umgang mit den hier beschriebenen Strukturen entwickelt wurde, war hier eine Implementierung als Teil des Gesamtalgorithmus möglich. Dieser stellt eine Erweiterung des vorhergehend vorgestellten Systems zur Prioritätenbereinigung dar und entfernt Moduloutputs, bei welchen keine Lösung gefunden wurde. Weiterhin wird überprüft, ob für die Lösung der Gesamtproblematik noch ausreichend Lösungsvorschläge vorhanden sind.

5.3.6 Algorithmus zur Geräte- und Aktionsauswahl

Vorhandene Geräte wurden händisch direkt in der Ontologie als, sogenannte „Individuals“ implementiert. Der Speicherort für deren Daten in der zentralen Datenbank erfolgte über eine gleiche Namensgebung der Geräte und Readings. Somit ist der Algorithmus mit Hilfe des Datensammlermoduls in der Lage, diese Daten auszulesen und aufzubereiten. Als Geräte wurden nur die für die Fallstudien, in Bezug auf die Ontologie relevanten Sensoren und Geräte eingefügt, um eine Übersichtlichkeit und Nachvollziehbarkeit zu gewährleisten. Um die Aktivierung der Geräte automatisch zu realisieren wurden deren Befehlsstruktur durch eine Dateneigenschaft, dem "ActivationSyntax“ in der Ontologie repräsentiert. Diese enthält alle notwendigen Informationen, um eine automatische Steuerung des Gerätes mit Hilfe des Automationsservers zu initiieren.

5.4 Technische Umsetzung – Hardwareanbindung

Im Folgenden werden Informationen zum technischen Hintergrund des Frameworks, welches für die folgenden Fallstudien entwickelt wurde, gegeben. Dabei wird sich eine Beschränkung auf allgemeine Informationen und Zusammenhänge angewandt. Detaillierte technische Informationen und Versionshinweise zu eingesetzter Hard- und Software können dem Anhang entnommen werden.

5.4.1 Auswahl und Einbindung des Automationsservers

Zur Auswahl des Automationsserver wurden die 16 gängigsten und bekanntesten quelloffenen Automationsserver untersucht. Diese sind:

- FHEM
- OpenHab2
- Domoticz
- HomeAssistant

Eine genaue Übersicht der für diese Auswahl erhobenen Daten kann dem Anhang entnommen werden.
OpenHab wurde nicht verwendet, da eine Weiterentwicklung in Form von OpenHab2 angekündigt wurde. Die Automationssoftware OGEMA des Fraunhofer IWES wurde nicht mit aufgeführt, da sie zwar Open Source Charakter hat, jedoch nicht durch eine Community gestützt ist. Die große Programmierer-Community, welche für FHEM Weiterentwicklungen und Updates zur Verfügung stellt, war letztendlich mitentscheidend für dessen Auswahl, im Rahmen dieser Arbeit, war.

5.4.2 Hardware und Geräteanbindung

Das Gesamtsystem wurde zweigeteilt umgesetzt. Die Software des Automationsservers läuft auf einem, für den Heimgebrauch spezialisierten, NAS-System. Diese Hardware hat genügend Leistungsreserven, um auch übliche Berechnungen in geeigneter Zeit durchzuführen. Die Möglichkeit Virtualisierungen auf dem System zu erstellen ermöglicht zusätzlich eine größere Flexibilität, die vor allem für die Integration von Diensten, wie dem Datenbankserver für das nachfolgend beschriebene Industriemesssystem, oder die Erweiterung der Homekitunterstützung von Apple hilfreich war. Für komplexere Rechenoperationen und auch Simulationsaufgaben sind diese Systeme jedoch nicht geeignet. Daher wurde hierfür ein spezieller Industrieserver außerhalb der Wohnung platziert. Auf diesem läuft eine Kopie des FHEM-Servers, welche es zulässt, über das Netzwerk Informationen

zwischen beiden FHEM-Instanzen zu versenden. Somit ist eine Anbindung an die Wohnung möglich, ohne eine Positionierung des Servers in der Wohnung gewährleisten zu müssen.
Die Anbindung der Automationshardware erfolgt weitgehend an der Serverinstanz, die in der Wohnung platziert wurde. Für das hier vorgestellte Framework wurde Wert darauf gelegt, möglichst viele Protokolle und Gerätetypen zu implementieren, um damit auch die Interoperabilität der Methode zu demonstrieren. Eine Anbindung der folgend aufgelisteten Protokolle wurde zu diesem Zweck realisiert:

Tabelle 30 Unterstützung von Protokollen

Protokoll	**Angebunden über**
Z-Wave	USB
Bluetooth	USB
BLE	USB
Panstamp (866MHZ Funk Eigenbau)	USB
BidCoS	Netzwerk
Beleuchtungssteuerung (Mi-light)	Netzwerk (WLAN)
Multimediageräte per Harmony-Hub	Netzwerk-Infrarot
diverse Eigenbauten	Netzwerk

Um Messungen der bauphysikalischen Größen durchzuführen, welche von Automationshardware nicht geliefert werden, wurde zusätzlich ein Industriecomputer der Firma Beckhoff eingesetzt, welcher eine Messung dieser Größen ermöglicht. Die Speicherung der Messergebnisse erfolgt in der zentralen Datenbank.

Die Einbindung von Geräten in den FHEM Server erfolgt über dessen Frontend. Das Triggern des Hauptalgorithmus erfolgt über das bereits beschriebene „Notify“ direkt aus FHEM heraus, auch das Abspeichern der Sensordaten in einer Datenbank erfolgt direkt aus dem Automationsserver.
Das Absetzen der erzeugten Steueraktionen geschieht über eine Telnet-Schnittstelle. Diese Schnittstelle gestattet es, Nachrichten in Textform zwischen zwei Programmen auszutauschen. Auf der Telnet-Konsole werden dazu einfache einzeilige Befehle übergeben. Die Zusammensetzung dieser Befehle hängt von den jeweiligen in FHEM definierten Geräten ab und muss in der Ontologie vorher in Form des „ActivationSyntax“ definiert sein.

Ein Beispiel für einen abzusetzenden Befehl, um ein Gerät mit dem Namen“ WohnzimmerLicht“ zu aktivieren, wäre:

```
Set WohnzimerLicht on
```

Für einfache Visualisierungszwecke ist es auf diese Art möglich, einen visuellen Repräsentanten des DGZ oder des DNZ im Automationsserver anzulegen. Dieses Vorgehen wäre auch eine Möglichkeit Daten aus dem zu integrierenden BI-Modelle zu implementieren.

Eine beispielhafte Umsetzung der Repräsentation des DNZ ist in der folgenden Abbildung dargestellt. Der Nutzer ist hier als Geräteeinheit im Smarthome-Server repräsentiert, welche verschiedene „Readings“, ähnlich der Beschreibung von Sensorwerten zur näheren Bestimmung, erhalten kann.

set rr_Alexander mood happy

Readings

Radon_Limit	<= 300	2019-04-01 10:19:55
durTimerAbsence	03:49:35	2019-04-06 16:14:30
durTimerAbsence_cr	230	2019-04-06 16:14:30
lastAwake	2019-04-06 12:03:30	2019-04-06 12:03:30
lastDeparture	2019-04-06 12:24:55	2019-04-06 12:24:55
lastDurAbsence	05:01:54	2019-04-05 22:58:29
lastDurAbsence_cr	302	2019-04-05 22:58:29
lastDurPresence	13:26:26	2019-04-06 12:24:55
lastDurPresence_cr	806	2019-04-06 12:24:55
lastDurSleep	10:19:19	2019-04-06 12:03:30
lastDurSleep_cr	619	2019-04-06 12:03:30
lastLocation	home	2019-04-06 12:24:55
lastMood	calm	2019-04-06 12:24:55
lastSleep	2019-04-06 01:44:11	2019-04-06 01:44:11
lastState	home	2019-04-06 12:24:55
lastWakeup	10:00	2019-04-05 09:30:00
lastWakeupDev	rr_Alexander_wakeuptimer1	2019-04-05 09:30:00
location	underway	2019-04-06 12:24:55
mood	happy	2019-04-06 16:16:29
nextWakeup	OFF	2019-04-06 08:16:23
nextWakeupDev		2019-04-06 08:16:23
presence	absent	2019-04-06 12:24:55

Abb. 42 FHEM-Seite DNZ Alexander Peikos

5.4.3 Multiple Recheninstanzen

Die mehrfache Instanziierung einzelner Module stellt in der Informationstechnik eine Grundfunktion der objektorientierten Programmierung dar und wurde daher im Rahmen dieser Arbeit nicht zusätzlich demonstriert und umgesetzt. Da die Module aber entsprechend den Anforderungen an ein modulares Konzept in sich abgeschlossen sind, ist solch eine Vorgehensweise prinzipiell möglich. Die Ausgaben verschiedener Modulinstanzen können mithilfe der Interaktionsarrays und der Prioritätenuntersuchung auf eine einfache Basis zurückgeführt werden und haben daher keinen Einfluss auf die Funktionalität des Gesamtalgorithmus. Zu diesem Zweck werden alle Modulinstanzen mit einer eindeutigen ID versehen.
Nichtsdestotrotz sollte im Rahmen weiterführender Forschung eine Untersuchung notwendiger Grenzen zur Beschreibung des Wohnraumes für eine mehrfache Instanziierung erfolgen. Eine grundsätzliche Funktionsfähigkeit ontologiebasierter Algorithmen im Einsatz mit Modulen deren LOD eindeutig unterschiedlich sind, sowie deren Problematik bei der Interaktion, zeigte bereits Mitterhofer (2017). Daher sind die Möglichkeiten für eine Beschreibung und Zusammenführung mehrfacher Modulinstanzen und deren Ein- und Ausgangsdatenströme als gegeben anzunehmen.

6 Evaluierung mittels Fallstudien

Um die Funktionalität der entwickelten Methode nachzuweisen wurden Fallstudien durchgeführt, mittels derer die Funktionsfähigkeit der Algorithmik für die jeweiligen Teilbereiche nachgewiesen werden kann. Die so entstandenen drei Fallstudien weisen die grundsätzliche Funktionsfähigkeit der automatischen Zuweisung von Datenströmen sowie den reibungsfreien Ablauf von einem Trigger-Event bis zum Auslösen einer Aktion nach. Außerdem werden durch das Hinzufügen einer weiteren bauphysikalischen Funktion weitere Teilaspekte geprüft. Diese sind sowohl die Prioritätenberücksichtigung bei der Generierung von Wirkgrößen als auch die Überprüfung auf Dilemmata. Zuletzt werden noch die Möglichkeiten der Individualisierung durch Module für maschinelles Lernen und Vorhersagen demonstriert. Zu diesem Zweck wird das vergangene Duschverhalten eines Nutzers ausgewertet und damit eine Prognose für das zukünftige Duschverhalten erstellt.

6.1 Übersicht der Fallstudien zur Evaluierung einzelner Algorithmusbestandteile

Um Tests verschiedener Funktionen durchzuführen und einen Überblick über die Komplexität von Hausautomationen zu erlangen, wurde eine Zweizimmerwohnung automatisiert. Die Testwohnung steht in Olching und ist Teil eines Mehrparteienhauses mit 53 Wohneinheiten.
Im Vorfeld wurden im Rahmen erster Tests bauphysikalische Funktionen in die automatisierte Wohnung implementiert. Ziel war es, sicherzustellen, dass die in dieser Arbeit vorgestellte Methode für möglichst viele Bereiche der Bauphysik anwendbar ist. Zur Validierung der Funktionalität der hier vorgestellten Methodik wurden daraufhin beispielhafte Anwendungen ausgewählt, mit welchen die einzelnen Teilaspekte der Methodik und deren Funktionalität demonstriert werden konnten. Die in weiteren Fallstudien genannten beispielhaften Anwendungen wurden so gewählt, dass jeweils ein besonderes Merkmal der Methodik demonstriert werden kann. Eine Übersicht der zusätzlich überprüften Ansätze auf deren Verwendbarkeit mit der hier vorgestellten Methode kann dem Anhang entnommen werden.

Nach erfolgreichem Abschluss aller Fallstudien konnte damit sichergestellt werden, dass jeder Teil der Methodik als funktionsfähig evaluiert wurde. Die einzelnen Fallstudien weisen konkret folgende Aspekte nach:

Fallstudie 1 weißt die grundsätzliche Funktionalität des ontologiebasierten Algorithmus nach und dass eine automatische Zuweisung von zugehörigen Datenströmen möglich ist.

Fallstudie 2 weißt Möglichkeiten nach, bei welchen auch gegensätzliche Handlungsanweisungen aus verschiedenen Bereichen mittels Prioritätensystem selbstständig zu lösen sind. Durch die Einführung einer stark kontextsensitiven Größe in die Berechnung wird zusätzlich gezeigt, dass die Methodik kontextsensitiv und dynamisch funktioniert. Im direkten Vergleich zwischen Fallstudie 1 und 2 kann zusätzlich nachgewiesen werden, dass Modulinteraktionen automatisch erkannt werden und daher ein sich selbst regulierendes Berechnungssystem geschaffen wurde, wodurch der Nachweis für die Modularität des Systems erbracht ist.

Fallstudie 3 weißt die Möglichkeit nach, nutzerindividuelle Parameter zu ermitteln und somit eine Individualisierung in der Betrachtung und Steuerung des Wohnkontextes zu erstellen. Zusätzlich wird die Implementierung und Verarbeitung eines Moduls mit einem Machine-Learning-Algorithmus demonstriert. Durch die im Falle eines Dilemmas demonstrierte Ausgabe einer Warnung an den

Nutzer wird zusätzlich die Möglichkeit eines interaktiven Systems nachgewiesen. Zusammengefasst kann so mithilfe der drei Fallstudien eine Umsetzung der geforderten Kriterien

- Dynamik
- Kontextsensitivität
- Modularität
- Interaktivität
- Nutzerindividualität

erbracht werden.

6.2 Randbedingungen für die in den Fallstudien verwendete Testwohnung

Die Testwohnung wurde mit handelsüblicher sowie selbst entwickelter Automationshardware ausgestattet. Ziel war es, eine Testumgebung zu schaffen, die einer realen Anwendung eines Smarthomes so nahe wie möglich kommt. Damit sollten nicht nur theoretische Aspekte der Methode, sondern auch eine reale Umsetzbarkeit gezeigt werden. Probleme, wie die mangelnde Interoperabilität von Smarthome-Systemen, konnten durch die Verwendung von unterschiedlichsten Systemen mit in die Fallstudien einbezogen werden und die Eignung der Methode auch für diesen Anwendungsfall nachgewiesen werden. Details zur verwendeten Wohnung sowie deren Ausstattung mit Automationshardware werden in den folgenden Kapiteln erläutert.

6.2.1 Räume und Grundriss

Es handelt sich um eine Zweizimmerwohnung mit dem Baujahr 2013. Sie hat einen für die Erstellungszeit typischen Grundriss mit einer Wohnküche. Jeder Raum, auch das Bad, hat ein außenliegendes Fenster. Zusätzlich ist sowohl im Bad als auch in der Küche eine Abluftanlage vorhanden. Folgender Grundriss zeigt die räumliche Anordnung.

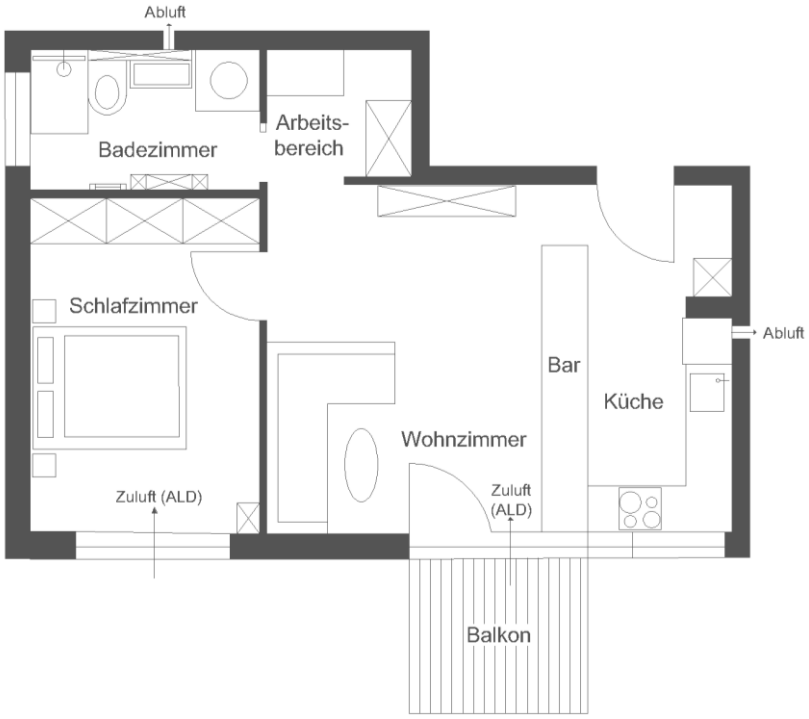


Abb. 43 Grundriss und Raumaufteilung der Testwohnung

6.2.2 Sensorik

In der oben beschriebenen Testwohnung wurden für die Durchführung der Fallstudien mehrere Systeme zur Erfassung von Messdaten verbaut. Es wurden unter anderem kommerzielle Sensoren, die im Smarthome-Sektor angeboten werden, eingebunden. Die hierbei über Funk stattfindende Übertragung ermöglichte eine flächendeckende Installation der Sensorik. Die üblichen im Smarthome-Bereich angebotenen Sensoren erfassen aber meist nur Temperaturen, relative Feuchten und in begrenztem Umfang lichttechnische Werte, wie zum Beispiel die Beleuchtungsstärke. Zusätzlich wurden Öffnungssensoren an Fenstern, Türen und Schubladen angebracht und flächendeckend Bewegungsmelder in der Wohnung verbaut. Bei der Wahl der Aktuatoren wurde darauf geachtet, ein System mit bidirektionaler Übertragung zu verwenden. Somit ist die Übertragung von Statusmeldungen der Geräte möglich. In vereinzelten Fällen wurden an elektrischen Verbrauchern Messsteckdosen installiert, die die elektrischen Größen messen können. Die Sensoren in Smarthome-Geräten sind bezüglich der Genauigkeit jedoch generell nicht vergleichbar mit üblicher Messsensorik aus der Bauphysik.
Daher wurde ein professionelles Messsystem der Firma Beckhoff installiert. Zum einen, um im Bedarfsfall für Fallstudien Messwerte mit höherer Genauigkeit verfügbar zu machen, zum anderen aber auch um fehlende Messgrößen, für die derzeit im Smarthome-Bereich kein Angebot an Sensoren existiert, erfassen zu können. Hierunter zählen zum Beispiel Wärmeflussscheiben, Globaltemperatur, Oberflächentemperatursensoren sowie Wetterdaten.
Der Anwesenheitsstatus des Nutzers für die zweite Fallstudie wurde über eine Anwesenheitskontrolle mit Hilfe des Smartphones des Nutzers erstellt. Es hat sich gezeigt, dass bei richtiger Konfiguration einer solchen Schaltung die Anzahl der Fehlauslösungen für „An- und Abwesenheit“ nahezu auf null verringert werden konnte. Eine Grundvoraussetzung dafür war, dass der Nutzer ein Smartphone immer beim Verlassen der Wohnung bei sich tragen muss. Dieses Vorgehen ist allerdings bei Personen, die ihr Smartphone nicht dauerhaft mit sich führen, nicht zielführend.

Da sich die verwendete Testwohnung im ersten Stock eines Mehrfamilienhauses befindet, ist die Radonkonzentration in der Wohnung, im Vergleich zu einem Kellerraum, sehr gering. Sie war für die Tests in den Fallstudien mit durchschnittlich 56 Bq/m³ nicht ausreichend, somit wurde die Radonbelastung durch ein einfaches Bilanzmodell ermittelt. Entsprechend des in der Wohnung vorliegenden Luftwechsels wurde ein realitätsnahes Steigen und Sinken der Konzentration simuliert. Um natürliche Schwankungen abzubilden wurde zusätzlich auf die erzeugten Radonwerte jeweils ein zufälliger Wert aufgeschlagen oder abgezogen. So konnte eine für die Fallstudien ausreichend realitätsnahe Abbildung des Raumverhaltens in Bezug auf die Radonkonzentration umgesetzt werden.
Das beschriebene Bilanzmodul nutzt hierfür eine eigene Matlab-Instanz und aktualisiert in einem Ein-Minuten-Intervall den Messwert der Radonkonzentration. Die Verarbeitung des Bilanzmodells ist daher im Vergleich zu einem realen Sensor ausreichend abgebildet.

6.2.3 Aktuatorik

Im Rahmen der Automation der Testwohnung wurden möglichst viele verschiedene Systeme sowie Protokolltypen verwendet, um einen guten Überblick über die jeweiligen Funktionsweisen und deren Übertragungseigenheiten zu erlangen. So wurden für die Steuerung der Abluftanlagen Aktuatoren von HomeMatic genutzt, die das BidCoS Protokoll einsetzen. Zur Automation der Lüftung kommen Stufenschalter zum Einsatz, mit denen sich die beiden vorhandenen Abluftventilatoren in drei Stufen funkgesteuert ansteuern lassen.
Zur Automation der Steuerung der Fußbodenheizung gibt es derzeit kein geeignetes kommerzielles und frei an einen Automationsserver anbindbares System. Um trotzdem die Heizfunktion abbilden zu

können, wurde ein Aktuator an den vorhandenen Heizstab im Handtuchheizkörper des Bades angebracht.

Bei den Modulen, welche in den Fallstudien verwendet werden, kommt es selten zu Anforderungen, die das Ausschalten eines Gerätes notwendig machen. Die generische Beschreibung eines radonsicheren Wohnraums führt nicht dazu, dass eine Lüftungsanlage ausgeschaltet werden muss. Ein dauerhaft hoher Luftwechsel stellt im Gegenteil eine durchgehend niedrige Radonkonzentration sicher. Belange des Energiesparens werden bei isolierter Betrachtung der Radonsicherheit nicht berücksichtigt. Ebenso gibt es für die Sicherstellung eines schimmelfreien Wohnumfeldes nur selten die Situation, dass die Lüftung deaktiviert werden muss. Dies wäre nur dann der Fall, wenn die Außenluft zu feucht ist, um einen Trocknungsvorgang in der Wohnung zu bewirken. Das Abschalten dieser Geräte erfolgt daher normalerweise aus energetischen Gründen oder um den Verschleiß zu minimieren.
Für die Fallstudien hat dies jedoch den Nachteil, dass bisher nur Module implementiert wurden, die die Geräte an aber nicht mehr zwingend ausschalten. Daher wurde eine zusätzliche Funktion implementiert, welche die Geräte, ähnlich einem Energiesparmodus, wieder deaktiviert.
Dies geschieht immer dann, wenn diese von keiner Anforderung benötigt werden. Grundsätzlich kommt diese Vorgehensweise einer Art Energiesparmodus gleich. Dieses Vorgehen bringt jedoch den Nachteil mit sich, dass Geräte nach einem einmaligen Aktivieren erst wieder deaktiviert werden, wenn für Ihre Wirkgröße keine Anforderung mehr vorhanden ist. Das kann bei Modulen, wie dem zur Berücksichtigung der Schimmelbildungsgefahr dazu führen, dass die Heizung vergleichsweise spät ausgeschaltet wird und es so zu ungewollten Mischungen von Moduloutputs kommt. Ein Beispiel für dieses Verhalten ist in der Fallstudie 2 im Szenario 3 in Abb. 59 zu sehen. Aufgrund des hohen Keimungsgrades wird dauerhaft eine Anforderung an das Heizen gestellt. Dadurch wird die Heizung nicht mehr deaktiviert bis der Keimungsgrad wieder in einem unkritischen Bereich ist. Da dieser Fall im gewählten Szenario erst mehrere Tage in der Zukunft liegt, konnte er nicht abgebildet werden. Für die gezeigten Fallstudien und deren Visualisierung ist dieser Ansatz brauchbar, da dadurch keine negativen Effekte in Bezug auf die Modulziele - hier Schimmelvermeidung - auftreten. Es entsteht jedoch eine unbeabsichtigte Mischung von Moduloutputs, deren Auswirkungen aber durch die Dynamik des Systems im weiteren Verlauf berücksichtigt werden.

Um einem Nutzer direkt informieren zu können, wurde ein System verbaut, das über Sprachausgaben mit diesem in Kontakt treten kann und ermöglicht, in jedem Raum unterschiedliche Sprachansagen zu tätigen.

6.2.4 Automationsserver

Um alle Steuerungsaufgaben und vor allem die Anbindung der Interfaces für die vielfältige Hardware gewährleisten zu können, musste eine mit einem Automationsserver versehene Recheneinheit in der Wohnung platziert werden.
Als Hardware eignet sich ein Network Attached Storage (NAS) besonders gut, da dieses den benötigten Speicherplatz für die Messdaten bereitstellen kann. Zusätzlich wurde in der Wohnung ein dLAN Netzwerk installiert, mit welchem es möglich war, eine Netzwerkverbindung zwischen Wohnung und Kellerabteil herzustellen. Hier wurde ein Industrieserver installiert, welcher rechenintensive Arbeiten übernimmt.
Mittels einer VPN-Verbindung kann man von außerhalb der Wohnung auf das Smarthome-System zugreifen.
Zur Steuerung der einzelnen Geräte sind, sofern die Geräte nicht netzwerkfähig sind, Interfaces notwendig, die zum Beispiel Funkstrecken auf ein USB-Signal oder ein Netzwerksignal umsetzen. Im letzteren Fall ist man bei der Platzierung des Interfaces nur auf eine Netzwerkverbindung angewiesen und benötigt keine direkte Verbindung zum Server. Sofern es sich aber um Interfaces mit USB-

Verbindung handelt, ist eine direkte physische Verbindung mit der Hardware, in diesem Fall der NAS, notwendig. Dies ist auch der Hauptgrund, wieso eine physische Instanz des Smarthome-Servers in der Wohnung erforderlich war.

Mit dem eben beschriebenen Framework wurden nun im Folgenden die drei Fallstudien durchgeführt, welche zur Evaluation der vorgestellten Methodik erforderlich sind.

6.3 Fallstudie 1: Beispielhafte Berechnung der Schimmelbildungsgefahr

Um die grundlegende Funktionalität der Methodik zu zeigen, wird in dieser Fallstudie die Thematik der Schimmelbildungsgefahr im Wohnraum untersucht. Zu diesem Zweck wurde das Badezimmer als Betrachtungsfeld ausgewählt. Die hier auftretenden hohen Luftfeuchtigkeiten stellen das Maximum der Feuchteproduktion im Wohnkontext dar und sind daher für diese Untersuchung geeignet. Im Rahmen dieser Fallstudie wurden mehrere Duschereignisse auf deren Einfluss auf die Schimmelbildungsgefahr untersucht.

6.3.1 Einleitung zur Fallstudie 1

Diese Fallstudie zeigt die Modulimplementierung am Beispiel der Schimmelberechnung zum Nachweis der Funktionalität der automatischen Zuordnung von Sensorwerten zu Modulen. Weiterhin wird nachgewiesen, dass eine automatische Generierung der interaktionsbedingten Verbindungen zwischen Modulen des DGZ und des DNZ erfolgt und eine automatische Verknüpfung von Wirkmodulen zu Geräten des Smarthomes erfolgt. Zu diesem Zweck wurde im ersten Schritt auf ein einfaches Modell der Schimmelbildungsberechnung zurückgegriffen: Das Isoplethenmodell.

6.3.2 Systemaufbau der Fallstudie 1

Bei der konkreten Umsetzung wurden auf Seiten des DNZ folgendes Modul implementiert:

- Anforderung an das Wachstum von Schimmel

Aufseiten des DGZ wurden folgende Module und Daten integriert:

- Funktionsmodul Schimmelberechnung (im weiteren Verlauf Schimmelmodul genannt)
- Hilfsmodul Oberflächentemperatur aus Raumtemperatur U-Wert und Außentemperatur
- Hilfsmodul Oberflächenfeuchte aus Raumtemperatur, Raumluftfeuchte und Oberflächentemperatur
- U-Wert der Wandbauteile aus Ersatzdaten

Als Geräte wurden folgende instanziiert:

- Sensor zur Bestimmung der Raumlufttemperatur
- Sensor zur Bestimmung der Raumluftfeuchte
- Sensor zur Bestimmung der Außenlufttemperatur
- Sensor zur Bestimmung der Außenluftfeuchte
- Aktuator zur Erhöhung des Luftwechsels

Die Instanziierung dieser Module ist in der nachfolgenden Grafik abgebildet. Sie zeigt Hilfs- und Funktionsmodule, in blau dargestellt, sowie deren Inputvariablen, in grün dargestellt, und deren Outputvariablen, in Orange dargestellt. Die Datenströme wurden thematisch so geordnet, dass ersichtlich wird, welche eine Interaktion zu anderen Modulen erzeugen und welche Verbindungen zu Sensorik oder anderen Datenquellen erfordern.

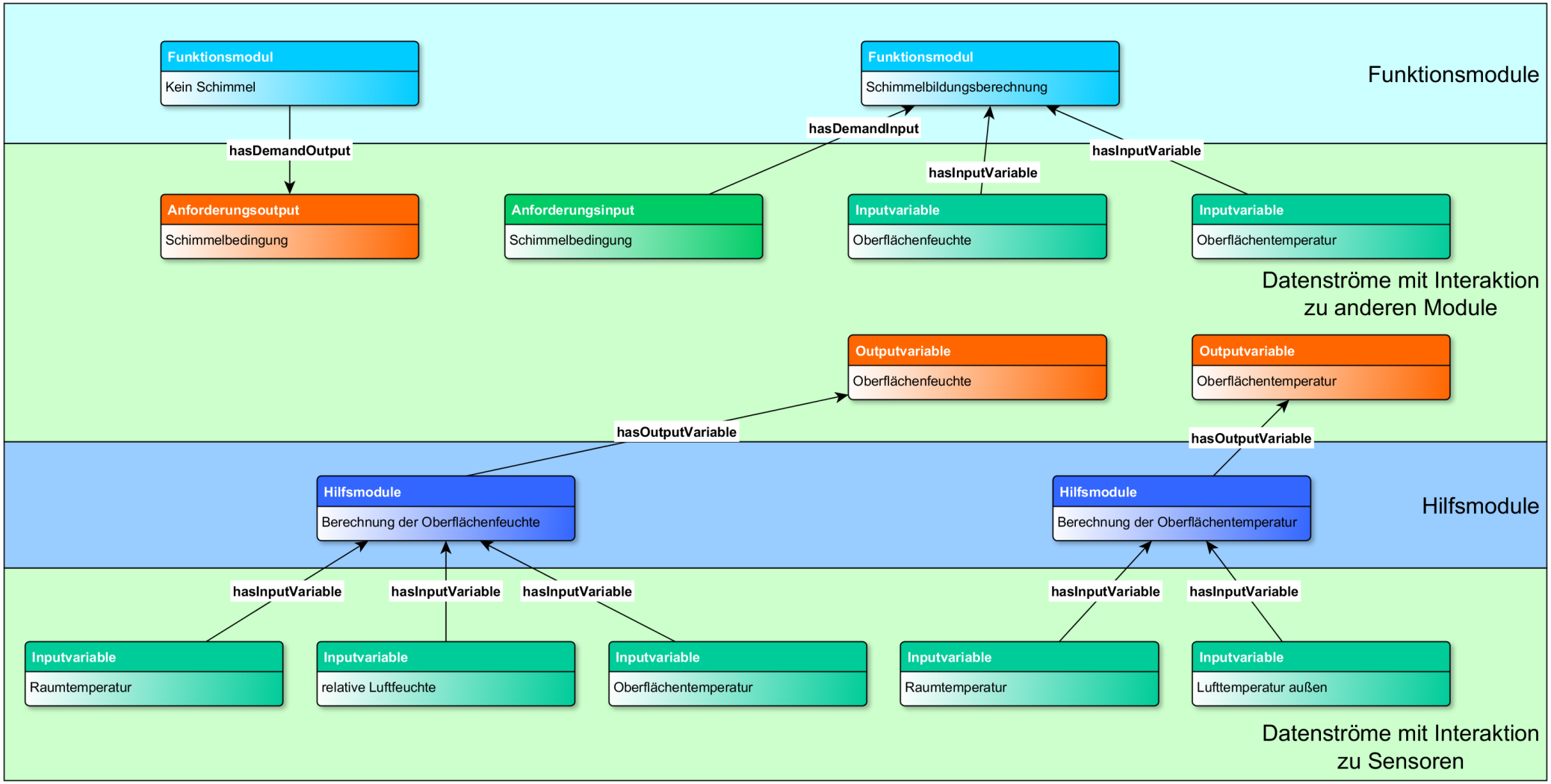


Abb. 44 Ontologie ohne Reasoning

6.3.3 Umsetzung der Fallstudie 1

Die Definition des digitalen Zwillings des Nutzers ist möglichst einfach gehalten: Ein Auskeimen des Schimmels muss verhindert werden. Daher enthält das Modul aufseiten des DNZ nur die Information:

Anforderung

$$Schimmelkeimung \neq 1$$

Auf der Seite des DGZ wird hier das Isoplethenmodell implementiert, mit dem es möglich ist auf Basis empirischer Daten die Keimung der Spore mithilfe von Tabellen und Grafiken vorherzusagen. Auch die komplexere Variante zur Schimmelvorhersage, der biohygrothermische Ansatz von Sedlbauer, wurde getestet (Anderlik, 2017). Damit wäre aber eine Implementierung aufgrund des notwendigen Simulationsframeworks deutlich aufwändiger. Somit wurde auf eine Demonstration mit diesem Algorithmus verzichtet. Wesentliche Aspekte der entwickelten Methodik lassen sich auch mit dem vereinfachten Berechnungsmodell zeigen, daher wurde dieses ausgewählt. Die erforderlichen Eingangsgrößen werden auf Basis von Sensorwerten erfasst oder durch Hilfsmodule des DGZ auf Basis von Berechnungen zur Verfügung gestellt.

So wird mit Hilfe der Raumluftfeuchte, der Raumtemperatur und der Außenlufttemperatur durch ein Hilfsmodul die Oberflächentemperatur der Außenwand ermittelt. Anschließend wird diese auf die Oberflächenfeuchte umgerechnet und somit der erforderliche Eingangswert für die Schimmelbewertung generiert. Das Funktionsmodul (Schimmelmodul) berechnet, ob bei aktuellen Randbedingungen im stationären Fall im weiteren Verlauf eine Keimung erfolgt und wenn ja, wie weit diese in der Zukunft liegt. Basierend auf dieser Abschätzung erzeugt das Modul eine Anforderung in Form einer Luftwechselrate. Um deren Priorisierung zu ermitteln, wurde die Zeit bis zur voraussichtlichen Auskeimung als Bewertungsmaßstab herangezogen. Nachstehende Tabelle zeigt die für den Testzweck definierten Randbedingungen bezüglich der Prioritätenbestimmung des Moduls in Abhängigkeit von der Zeit bis zur Auskeimung, der aktuellen hygrothermischen Randbedingungen und dem grundsätzlichen Keimungsgrad. Die dargestellte resultierende Handlungsempfehlung ist hier nur exemplarisch zu verstehen. Eine allgemeingültige Abbildung kann auf Grund der dynamischen Ermittlung der Abhängigkeiten der Randbedingungen nicht stattfinden. Die in folgender Tabelle abgebildeten Grenzen bilden dabei jeweils mit den in der Tabelle darüberliegenden höheren Anforderungen ein Intervall. Auf eine Abbildung des gesamten Intervalls wurde aus Gründen der Übersichtlichkeit verzichtet.

Tabelle 31 Randbedingungen zur Bestimmung der Prioritäten der Wirkgröße Lüftung am Beispiel der Berechnung der Schimmelbildungsgefahr

Kann Lüften die Rel. Luftfeuchte reduzieren	Aktuelle Bedingung keimungsfördernd	Zeit bis zur Keimung in Tagen	Status der Keimung in %	Priorität	Resultierende Handlungs-empfehlung Lüftung
Ja	Ja	-	>= 70%	A	n > 6
Ja	Nein	-	>= 70%	B	n > 5
Ja	Ja	-	>= 50%	B	n > 5
Ja	Nein	-	>= 50%	B	n > 2
Ja	Ja	>= 2	>=20%	C	n > 2
Ja	Ja	> 0	>=20%	B	n > 5
Ja	Nein	-	>=20%	C	n > 2
Ja	Ja	>= 2	>= 0%	C	n > 2
Ja	Ja	> 0	> = 0%	B	n > 5
Ja	Nein	> 0	>= 0%	D	Keine Anforderung
Nein	Ja	-	>=70 %	A	n = 0
Nein	Nein	-	>= 70%	B	n = 0
Nein	Ja	-	>= 50%	B	n = 0
Nein	Nein	-	>= 50%	B	n = 0
Nein	Ja	>= 2	>= 20%	C	n = 0
Nein	Ja	> 0	>= 20%	B	n = 0
Nein	Nein	-	>= 20%	C	n = 0
Nein	Ja	>=2	>= 0%	C	n = 0
Nein	Ja	> 0	>= 0%	B	n = 0
Nein	Nein	-	>= 0%	D	n = 0

Nach dem Reasoning der Stufe I wurden die Verknüpfungen zwischen Sensoren und Moduleingängen gefunden. Dabei konnten alle vorhandenen Sensorwerte den richtigen Inputvariablen zugeordnet werden. Das Ergebnis ist in der folgenden Abbildung dargestellt. Diese Zuordnung erfolgte, da sowohl Inputvariablen als auch Outputvariablen dieselben Eigenschaften haben, welche zur Verdeutlichung mit abgebildet wurden. Durch die Definition des Sensorwertes als Innensensor (Located: Inside), der Zuordnung zum Luftraum des Bades (ObjectAffiliation: Air_Space_Bath), der Bestimmung des Mediums (Medium: Air), der beschreibenden Größe als Temperatur (Quantity: Temperature) und der Lokalisierung im Raum Bad (Location: Bath) kann

sichergestellt werden, dass nur Sensoren mit der richtigen Konfiguration und den benötigten Messwerten verwendet werden.

Als Beispiel für die automatische Weitergabe von Objektparametern wurde die „Location“ des Sensors direkt auf sein zugeordnetes „Reading“ weitergegeben. Die folgende Grafik zeigt die Bedingungen, die geprüft werden, in grau, um eine Verbindung zwischen einer Inputvariable und einem Sensor-Reading zu finden. Die Beschriftungen der Pfeile stellen dabei die Definitionen von Eigenschaften dar. Am Beispiel des Sensors ist ersichtlich, dass die Eigenschaften, wie hier die „Location“, auch an ein Reading vererbt werden können.

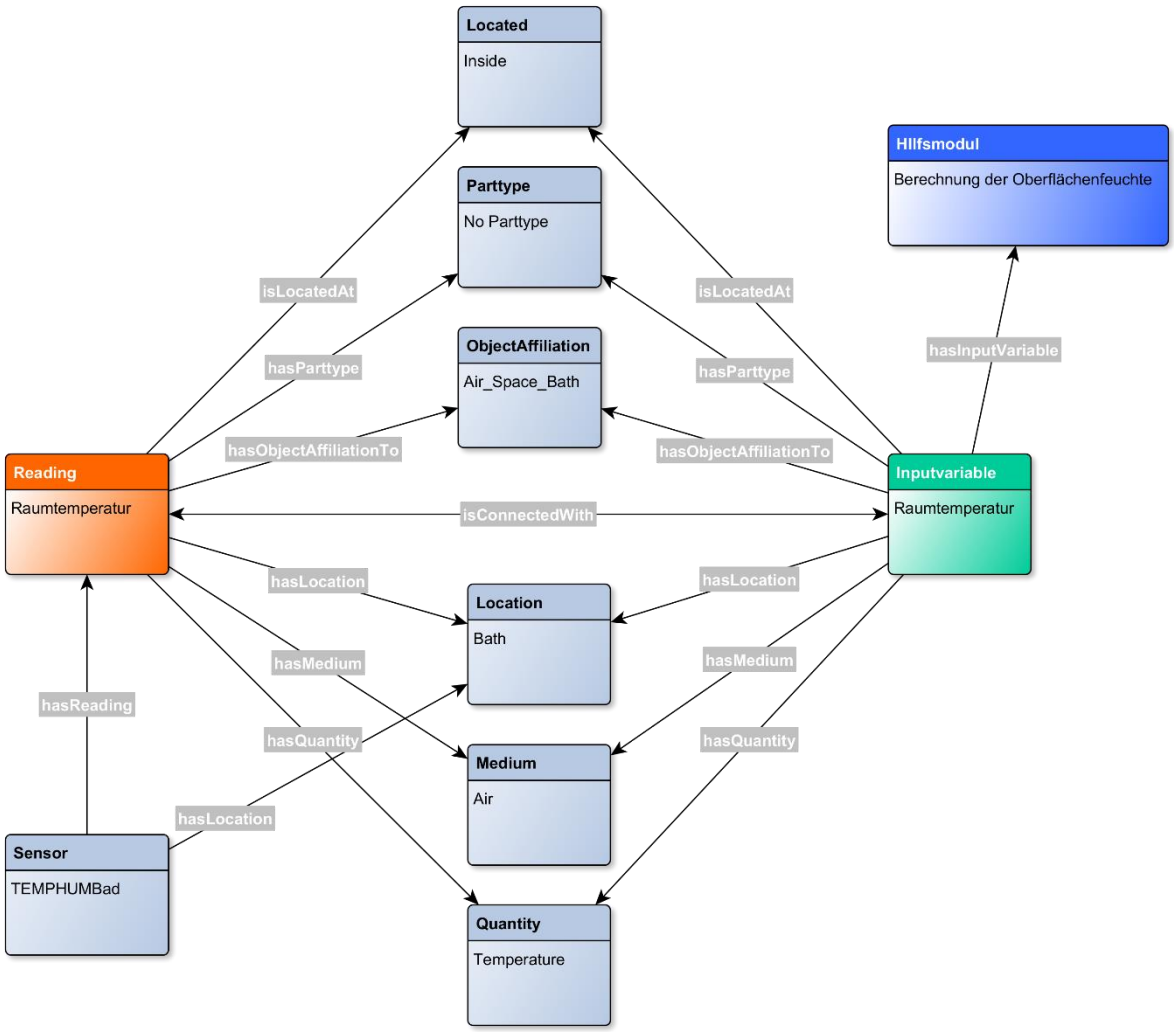


Abb. 45 Verbindung von Sensorwerten mit Inputvariablen

Diese Vorgehensweise zeigt, dass man mit dieser Methode in der Lage ist, Sensorwerte automatisch den richtigen Inputvariablen zuzuordnen. Ebenso kann dadurch eine automatische Weitergabe von Eigenschaften erfolgen, welche bei einer mehrfachen Instanziierung zur nötigen Unterscheidbarkeit von verschiedenen Inputvariablen, beispielsweise bei verschiedenen Räumen, notwendig ist.

Nach dem Reasoning der Stufe Ⅱ wurden für die noch offenen Inputvariablen die fehlenden Verknüpfungen mithilfe der Hilfsmodule hergestellt. Die Ergebnisse des Stufe Ⅰ und Ⅱ Reasonings sind in folgender Grafik abgebildet. Hier sind neben den bereits bekannten Modulen und deren In- und Outputvariablen auch die jeweiligen Verbindungen der Datenströme mit der Eigenschaft

„isConnectedWith“ gekennzeichnet, welche automatisch durch den Prozess des Reasonings gefunden wurden.

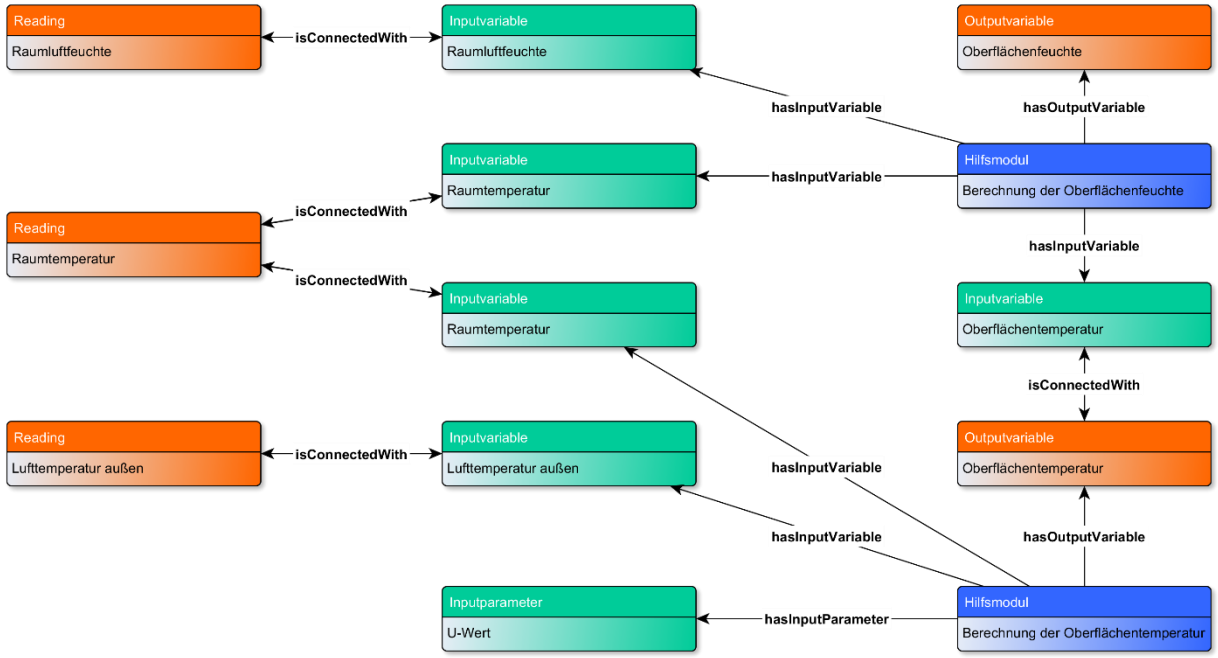


Abb. 46 Verbindungen zwischen Hilfsmodulen

Durch diese Verbindungen können so auch fremde Module zu einer Lösung des Problems beitragen. Mit steigender Modulzahl ist es dadurch möglich, immer die beste Lösung für eine Inputvariable entsprechend der „Many-To-One“ Systematik zu finden. Eine Interaktion thematisch verwandter Funktionsansätze ist dadurch sichergestellt und konnte nachgewiesen werden.

Das Reasoning der Stufe Ⅲ vervollständigte die fehlenden Werte durch Angaben aus Ersatzdaten, im konkreten Fall die Angabe des U-Wertes, wie folgend abgebildet.

Abb. 47 Verbindungen zu Ersatzdaten

Damit konnte nachgewiesen werden, dass bei korrekter Modulkonzipierung auch fehlende Daten nicht zu einer Beeinträchtigung der Funktionalität führen. Eine Individualisierung auf einen speziellen Untersuchungsraum, ohne dafür programmtechnische Anpassungen vornehmen zu müssen, konnte so nachgewiesen werden.

Das Reasoning der Stufe Ⅳ führte zu der noch fehlenden Verbindung des DNZ mit dem DGZ über die Anforderung, dass kein Schimmel auftreten darf. Diese Verbindung wird in der bereits bekannten Weise in nachfolgender Grafik abgebildet.

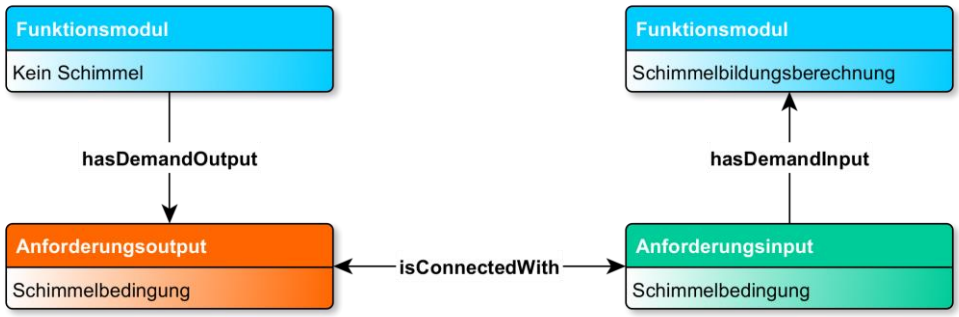


Abb. 48 Verbindung Funktionsmodule DNZ und DGZ

Damit konnten die für eine Berechnung notwendigen Informationen, welche für eine Untersuchung der Problematik notwendig sind, vervollständigt werden. Eine automatische Abbildung der Interaktion der digitalen Zwillinge konnte damit erfolgreich nachgewiesen werden. Wenn auch der Interaktionsumfang im gewählten Beispiel äußerst einfach ist, so zeigt die Fallstudie, dass durch die modulare Konzipierung nicht die expliziten Werte der jeweiligen Daten bei den Schnittstellen der Module ausschlaggebend sind. Vielmehr sind es die Eigenschaften der Daten, die in der Ontologie modelliert wurden und die die essenziellen Informationen für eine Zuordnung liefern.

Das Reasoning der Stufe Ⅴ konnte nun erfolgreich den Output des Funktionsmoduls zum Wirkmodul für die Steuerung des Luftwechsels zuordnen. Dies kann der folgenden Grafik, durch die erfolgreich gesetzte Eigenschaft „isConnectedWith“, entnommen werden.

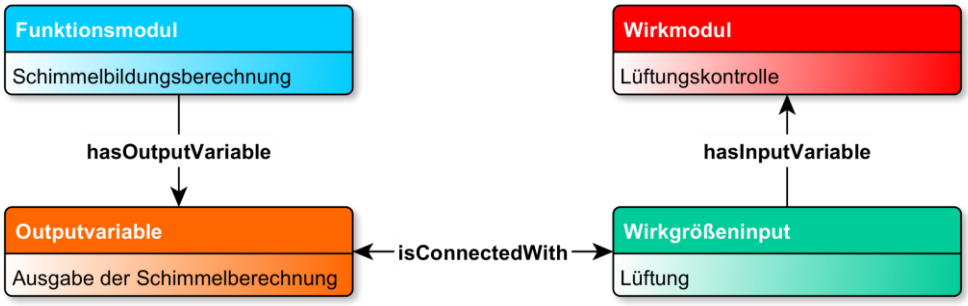


Abb. 49 Verbindung Funktionsmodul zu Wirkmodul

Durch diesen Schritt konnte von der Generierung eines Bedürfnisses und der Erstellung von Handlungsvorschlägen eine Verbindung zu möglichen Umsetzungsstrategien erfolgen. Da auch hier nur die Eigenschaften der übergebenen Daten und nicht die Daten selbst betrachtet wurden, kann eine Berücksichtigung des jeweiligen individuellen Gebäudekontext erfolgen, indem Handlungsvorschläge von Lösungsstrategien getrennt werden und eine Datenübergabe in einer generischen Form erfolgt.

Zum Abschluss konnte das Reasoning der Stufe VI die verfügbaren Geräte mit dem Wirkmodul für die Steuerung des Luftwechsels verbinden. Auch diese Verbindungen werden nachfolgend grafisch abgebildet.

Abb. 50 Verbindung von Wirkmodul zu Geräten

Somit werden dem Modul nun die gebäudeindividuellen Daten zum Entwickeln einer Lösungsstrategie übergeben. Da diese unabhängig von der Handlungsanweisung ermittelt und letztendlich im Wirkmodul verarbeitet werden, ist auch die benötigte Modularität vom Beginn bis zum Ende des Algorithmus nachgewiesen.

Da eine Vollständigkeit in diesem einfachen Fall leichter händisch nachgewiesen werden konnte, wurde auf das explizite Entwickeln der Vollständigkeitsregel verzichtet und der Schritt des Reasonings der Stufe VII übersprungen.

Wie im Folgenden abgebildet, konnte die Funktionsfähigkeit der Ontologie und des Reasoningprozesses damit nachgewiesen werden. Die Verbindungen der Sensoren mit den Modulen der einzelnen Zwillinge und die Modulverbindungen untereinander konnten dementsprechend erfolgen und die Basis für eine weitere Berechnung und einer nachfolgenden Generierung von Steueraktionen gebildet werden. Folgende Abbildung zeigt die vollständige für den individuellen Raum erzeugte Ontologie mit ihren einzelnen Individuen, welche die für den Hauptalgorithmus notwendigen Informationen für eine automatische Problemlösung enthält.

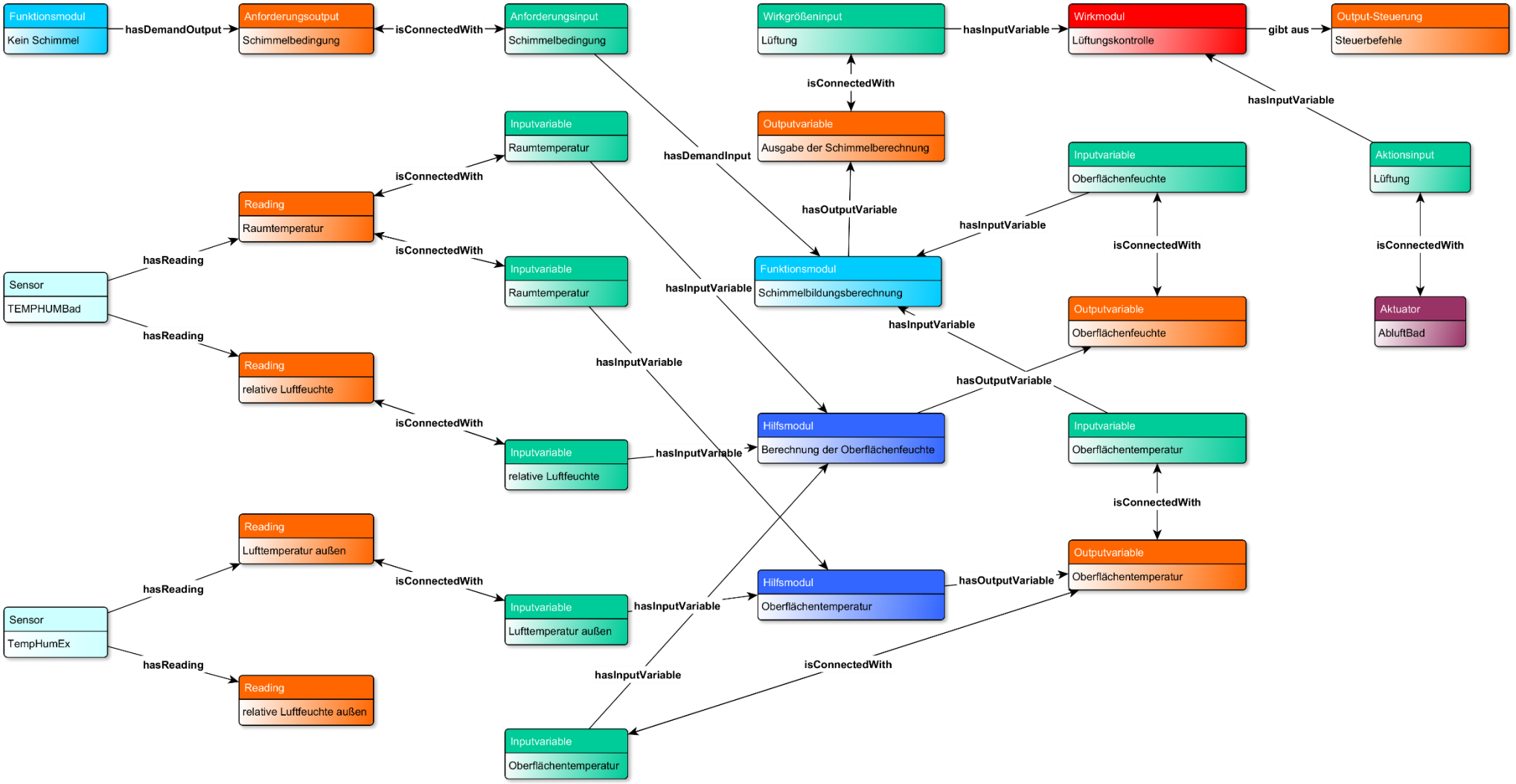


Abb. 51 Vollständige Ontologie mit Verbindungen durch Reasoning

Der Hauptalgorithmus ermittelt nun automatisch mit Hilfe der Sensor- und Ersatzdaten die nötigen Modulausgaben, um im Anschluss die Berechnung durch das Funktionsmodul zu initiieren. Für das hier gewählte Beispiel wurde als Trigger der Berechnung die relative Innenluftfeuchte und die Innenraumtemperatur gewählt. Da diese systembedingt vom Automationssystem ohnehin innerhalb eines ausreichend kleinen Zeitfensters aktualisiert werden, konnte so eine brauchbare Wiederholungsrate an Events für das Testszenario geschaffen werden. Im genannten Beispiel handelt es sich nur um ein Modul mit einem Output, daher wird an dieser Stelle der Bereich der Prioritätenverarbeitung übersprungen. Der Moduloutput entspricht dem letztendlich gültigen Gesamt-Output, der zu Steueraktionen verarbeitet werden kann. Hierzu wurden beispielhafte Tests durchgeführt, wofür die Wohnung über einen längeren Zeitraum leer stand. Nachdem sich der Raum in einem hygrisch nahezu ausgeglichenen Zustand befand, wurden für den Wohnkontext übliche Situationen nachgestellt. Diese sind im Folgenden aufgelistet:

- Duschen mit anschließendem Lüften
- Mehrfaches Duschen hintereinander

Das Resultat dieser Testreihen wird im Folgenden dargestellt. Abb. 52 zeigt den Verlauf der relativen Luftfeuchte und der Lufttemperatur bei einem Duschvorgang ohne nachfolgendes manuelles oder automatisches Lüften. Dies soll als Vergleich für die weiteren Untersuchungen dienen.

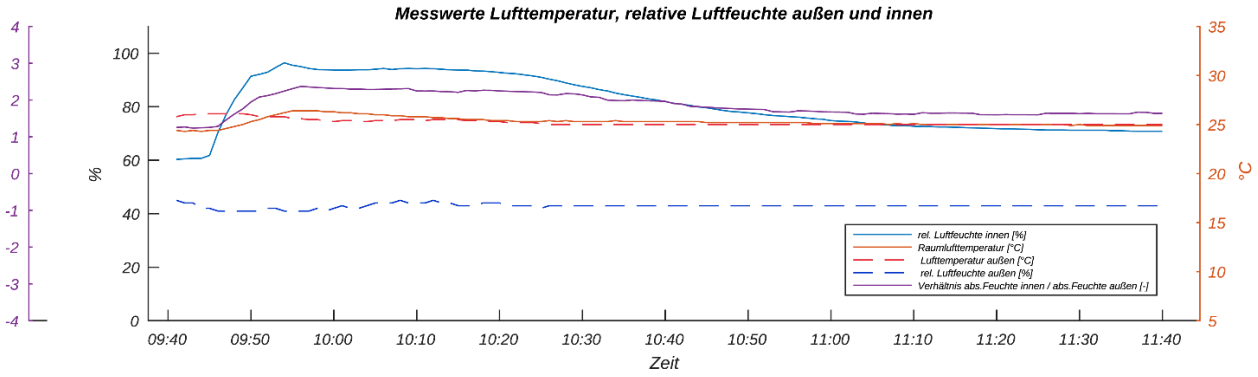


Abb. 52 Duschereignis ohne Lüftung

Die direkte Ausgabe des Schimmelmoduls, für den Fall eines gerade stattfindenden Duschvorganges, ist der folgenden Tabelle exemplarisch zu entnehmen. Die erste Spalte zeigt die empfohlenen, zu manipulierenden Wirkgrößen. Die zweite Spalte liefert den gewünschten Zielwert. In der dritten Spalte ist für den Zielwert die berechnungsbedingte Varianz hinterlegt. Die vierte Spalte listet die für den jeweiligen Zeitpunkt relevante Priorität auf. Die erste Zeile der Tabelle beschreibt den aktuellen Zeitpunkt. Alle folgenden Zeilen beschreiben darauffolgende Zeitpunkte, wobei die Größe des Zeitschrittes abhängig von der zeitlichen Auflösung des gesamten zeitlichen Berechnungssystems ist. Im hier gezeigten Beispiel handelt es sich dabei um Minutenwerte. Die weiteren Spalten wurden zwecks einer besseren Nachvollziehbarkeit mit aufgeführt und zeigen die Bewertung der Umgebung auf keimungsfördernde Bedingungen sowie die aktuell berechnete Keimzeit und den derzeitigen Keimungsgrad. Die Werte liegen alle ungerundet vor, wie sie direkt von der Software für die Weiterverarbeitung verwendet werden.

Tabelle 32 Programmausgabe MATLAB-Schimmelmodul Wirkgrößenarray Fallstudie 1

Aktion	Zielwert	Abweichung	Priorität	Keim.-fördernd % je Zeitschritt	Keimzeit in Stunden	Keimungsgrad in %
n>	5 - ∞	0,05	B	0,12988	12,7806	69,0912
n>	5 - ∞	0,10	B	0,12988	12,7639	69,2211
n>	5 - ∞	0,15	B	0,12988	12,7472	69,3510
n>	5 - ∞	0,20	B	0,12988	12,7306	69,4808
n>	5 - ∞	0,25	B	0,12988	12,7139	69,6107
n>	5 - ∞	0,30	B	0,12988	12,6972	69,7406
n>	5 - ∞	0,35	B	0,12988	12,6806	69,8705
n>	6 - ∞	0,40	A	0,12988	12,6639	70,0004
n>	6 - ∞	0,45	A	0,12988	12,6472	70,1302
n>	6 - ∞	0,50	A	0,12988	12,6306	70,2601

Die Berechnung der erforderlichen Steueraktion konnte mithilfe des Wirkmoduls zur Steuerung des Luftwechsels erfolgen und die für die Steuerung brauchbaren Geräte dadurch ausgewählt und ausgelöst werden. Im hier aufgeführten Beispiel wurde als einziges Gerät die Abluftsteuerung im Bad erkannt, welches das Bedürfnis nach einer Erhöhung des Luftwechsels befriedigen kann und dementsprechend die Aktivierung des Gerätes veranlasst. Der an den Automationsserver übergebene Befehl lautet wie folgt.

set AbluftBad scene 2

Die Stufe 2 (scene 2) der Abluft im Bad erzeugt einen Luftwechsel von n = 5 [1/h]. Dieser wurde mit Hilfe der Ontologie automatisch über das Raumvolumen und dem Volumenstrom, welcher dem Abluftgerät zugeordnet ist, ermittelt.

Die nachfolgenden Grafiken zeigen den Zusammenhang der Wirkgrößenmatrix im weiteren Zeitverlauf für ein Duschereignis. Die Aktualisierung eines Sensorwertes führt hierbei zur neuen Berechnung des Gesamtsystems und damit zur Ermittlung einer neuen Matrix. Die jeweils für den Berechnungszeitpunkt ermittelten Werte (erste Zeile der Matrix) wurden gespeichert und in folgender Grafik visualisiert. Auch die aus der Berechnung erfolgten Steueraktionen wurden nachfolgen dargestellt.

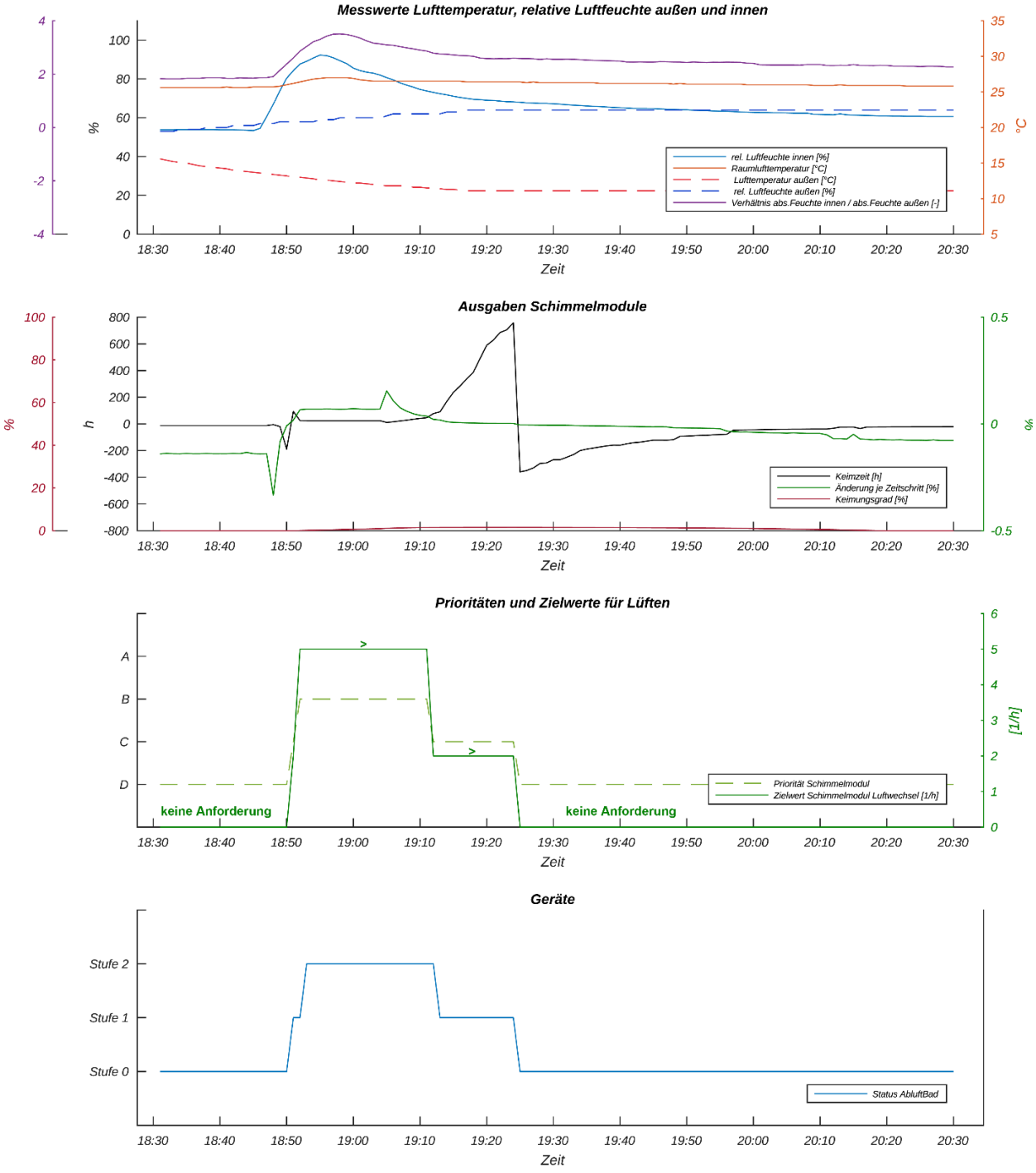


Abb. 53 Duschereignis und Darstellung der relevanten Berechnungsergebnisse

Die abgebildete Grafik zeigt, dass bei steigender Luftfeuchte auch die Gefahr einer Schimmelbildung steigt. Die Algorithmik erkennt dies um 18:50 Uhr und reagiert zeitnah mit dem Aktivieren der Lüftung in der höchsten Stufe. Im weiteren Verlauf wird eine Reduktion der Schimmelbildungsgefahr sichtbar. Daraus resultiert um 19:12 das Zurückschalten der Lüftung auf die Stufe 1 und ein Deaktivieren um 19:24 Uhr. Es ist zu erkennen, dass sich die Priorität für das Lüften, welche vom Modul ausgegeben wird, bei einer Änderung der Luftfeuchte entsprechend anpasst.

Im folgenden Beispiel wurde ein ähnlicher Ablauf mit der Simulation einer hygrothermischen Vorgeschichte in Form eines erhöhten anfänglichen Keimungsgrades von 69 % simuliert. Zusätzlich wurden zwei Duschereignisse in einem Abstand von 10 Minuten durchgeführt. Ziel war es, ein

Szenario zu schaffen, bei dem der Keimungsgrad die Grenze von 70 % überschreitet. Die Ergebnisse sind der folgenden Grafik zu entnehmen.

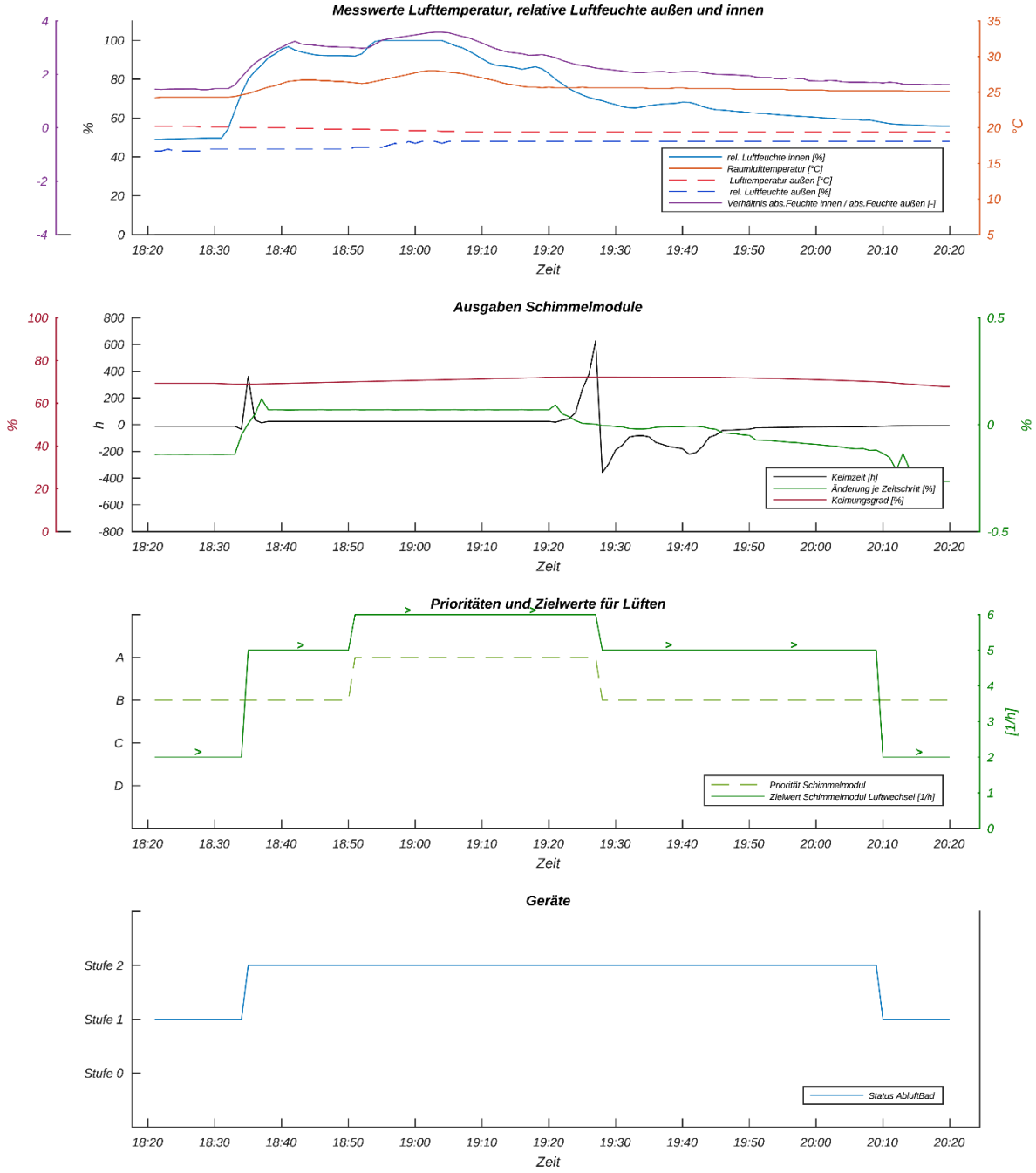


Abb. 54 Mehrfache Duschereignisse, Darstellung der relevanten Berechnungsergebnisse bei einem Anfangskeimungsgrad von 69%

Der aktuelle Keimungsgrad wurde zu Beginn der Messung auf 69 % gesetzt. Die Lüftung ist zu diesem Zeitpunkt, entsprechend der vorgegebenen Definitionen, bereits eingeschaltet. Durch die nun stattfindenden Duschereignisse um 18:32 Uhr und um 18:53 Uhr, steigt der Keimungsgrad weiter. Dies resultiert in einer Erhöhung der Priorität auf die Kategorie „A“ um 18:50 Uhr. Das System versucht dadurch, die weiterhin ansteigende Schimmelbildungsgefahr durch einen noch höheren Luftwechsel zu kompensieren. Nach dem Ende des zweiten Duschvorgangs ist zu erkennen, dass die Schimmelbildungsgefahr reduziert werden kann und damit auch um 20:09 Uhr die Lüftung wieder

auf Stufe 1 verringert wird. Aufgrund des hohen Keimungsgrades ändert sich an der Priorität jedoch nichts.

6.3.4 Fazit der Fallstudie 1

Durch die hier gezeigte Fallstudie konnte die grundlegende Funktionalität des Gesamtsystems nachgewiesen werden. Auch wurde gezeigt, dass die Modularisierung zu einer Unabhängigkeit des Systems vom Inhalt der einzelnen Module führt und eine Individualisierung in Bezug auf das Gebäude möglich ist. Da fehlende Daten aus Ersatzdaten verwendet werden und nur im jeweiligen speziellen Gebäudekontext vorhandene Geräte überprüft und geschaltet werden, kann von einer automatischen Zuordnung der Datenströme gesprochen werden. Die Fallstudie zeigt, dass bauphysikalische Funktionen in das hier vorgestellte System überführbar sind und wie die grundlegenden Interaktionen der digitalen Zwillinge vonstattengehen. Auch die richtige Zuordnung von Geräten zu einem geeigneten Wirkmodul und die richtige Steuerung durch dasselbe konnte gezeigt werden.

6.4 Fallstudie 2: Prioritäten und kontextsensitive Bewertung der Radonkonzentration

Nach der erfolgreichen Berücksichtigung der Schimmelbildungsgefahr wird nun in der zweiten Fallstudie die Berücksichtigung der Radonbelastung in einem Raum als weitere bauphysikalische Funktion integriert. Vor allem bei Räumen in Kellern und im Erdgeschoss spielt dies eine Rolle für die Gesundheit der Nutzer. Im Falle von Kelleraußenwänden kann die gleichzeitige Berücksichtigung der Schimmelbildungsgefahr und der Radonbelastung zu konträren Handlungsanweisungen führen. Vor allem im Sommer, bei hohen Außenlufttemperaturen und hohen absoluten Feuchten der Außenluft, kann das notwendige Lüften auf Grund einer erhöhter Radonkonzentration zu Problemen bezüglich der Schimmelbildung führen. Lösungen für dieses Problem ergeben sich in den meisten Fällen durch eine kontextsensitive Betrachtung. Damit wird das langsame Keimungsverhalten von Schimmel berücksichtigt und somit ein Lüften der Räumlichkeiten bei vorherrschender Anwesenheit des Nutzers erreicht. Auch erfolgt eine nachträgliche Regulierung in Bezug auf die Keimungsgefahr bei Schimmelpilzen für den Fall einer darauffolgenden Abwesenheit des Nutzers. Lange Anwesenheitszeiten des Nutzers können hier jedoch zu dem bereits erläuterten Dilemma führen. Die Ergebnisse können den folgenden Kapiteln entnommen werden.

6.4.1 Einleitung zur Fallstudie 2

Die Fallstudie baut auf der vorherigen auf. Sie zeigt die erfolgreiche Abbildung von Modulinteraktionen und der Priorisierung am Beispiel der Interaktion zwischen Schimmelbildung und Radonvorkommen im Raum. Um die Interaktion und Verknüpfung zweier Module mit eventuell gegensätzlichen Outputs zu demonstrieren, wurde als zusätzliches Modul die Bewertung der Radonkonzentration gewählt. Der Standort des Raumes, an dem die Schimmelbildungsberechnung durchgeführt wird, wurde virtuell in einen Kellerraum verlegt. Die Gewährleistung einer radonsicheren Umgebung erzeugt augenscheinlich, vor allem im Sommer, gegensätzliche Wirkgrößen zur Schimmelprävention.

Damit kann die Priorisierung der Wirkgrößen in Abhängigkeit von der Nutzeranwesenheit demonstriert werden. Durch die Nutzeranwesenheit wurde zusätzlich eine kontextsensitive Größe implementiert. Durch diese kann nachgewiesen werden, dass die vorgestellte Methode Kontextsensitivität unterstützt. Der Radongehalt spielt in der Regel hauptsächlich eine Relevanz bei Anwesenheit von Personen, hingegen muss die Schimmelbildung während der gesamten Zeit im

Raum überwacht werden. Durch die Verwendung der Nutzeranwesenheit als Eingangsgröße kann damit die automatische Priorisierung in Abhängigkeit der jeweiligen situationsbedingten Parameter nachgewiesen werden. Dieser Aspekt wird in der Fallstudie 3 am Beispiel der Vorhersage von Duschaktionen und der damit einhergehenden Berücksichtigung von auftretenden nutzerinduzierten Feuchtequellen noch weiter präzisiert.

6.4.2 Systemaufbau der Fallstudie 2

Um die Funktionsfähigkeit der Prioritätenbereinigung zu testen, wurde zusätzlich zu den Modulen der Fallstudie 1 noch eine Funktion zur Beschreibung der Radonbelastung in Räumen implementiert. Dazu wurden folgende Module zusätzlich instanziiert:

Auf der Seite des DNZ:

- Funktionsmodul zur Beschreibung der zulässigen Radonkonzentration für Menschen (im weiteren Verlauf Radonmodul genannt)

Auf der Seite des DGZ:

- Erstellen einer zusätzlichen Modulversion des Schimmelmoduls, welche eine Erhöhung der Beheizung des Raumes vorschlägt.
- Wirkmodul zur Steuerung der Beheizung des Raumes

Weiterhin wurden folgende Veränderungen durchgeführt:

- Radon Sensor im betroffenen Raum
- Der Raum befindet sich nun im Keller.
- Heizungsaktuator des Handtuchheizkörpers hinzugefügt.

Um diesen Sachverhalt und die Reaktion des Algorithmus zu testen, werden die beiden Funktionen parallel implementiert und in der Ontologie instanziiert.

6.4.3 Umsetzung der Fallstudie 2

Die Berechnung mit dem beschriebenen Prioritätensystem soll bei adäquater Modulentwicklung eine Lösung finden, die bei der Anwesenheit des Nutzers zu Lüften beginnt und bei Abwesenheit des Nutzers für die Reduktion der Schimmelbildungsgefahr optimale Bedingungen einstellt. Dies kann auch eine Reduktion des Luftwechsels oder das Heizen des Raumes bedeuten. Die Auslegung der Prioritäten und Handlungsanforderungen des Radonmoduls sind folgend abgebildet.

Tabelle 33 Randbedingungen zur Bestimmung der Prioritäten und der Wirkgröße Lüftung am Beispiel der Radonkonzentration im Wohnraum

Anwesenheit des Nutzers	**Radonkonzentration > 300 Bq/m³**	**Radonkonzentration > 100 Bq/m³**	**Priorität**	**Resultierende Handlungs-empfehlung Lüftung**
Ja	Ja	Ja	A	n > 6
Ja	Nein	Ja	B	n > 0,5
Ja	Nein	Nein	B	Keine Anforderung
Nein	Nicht relevant	Nicht relevant	B	Keine Anforderung

Zur Evaluation der Methodik werden im folgenden verschiedene Szenarien untersucht.

Diese sind:

Szenario 1: Zu hohe Radonkonzentration, Schimmelbildungsgefahr unkritisch

Szenario 2: Zu hohe Radonkonzentration und Duschereignis bei Außenbedingungen, die ein Lüften zulassen

Szenario 3: Zu hohe Radonkonzentration und Duschereignis mit hohem Keimungsgrad bei ungünstigen Außenbedingungen, welche kein Lüften zulassen

Szenario 4: Anwesenheit des Nutzers und Ausgabe des Schimmelmoduls führen zu einem Dilemma

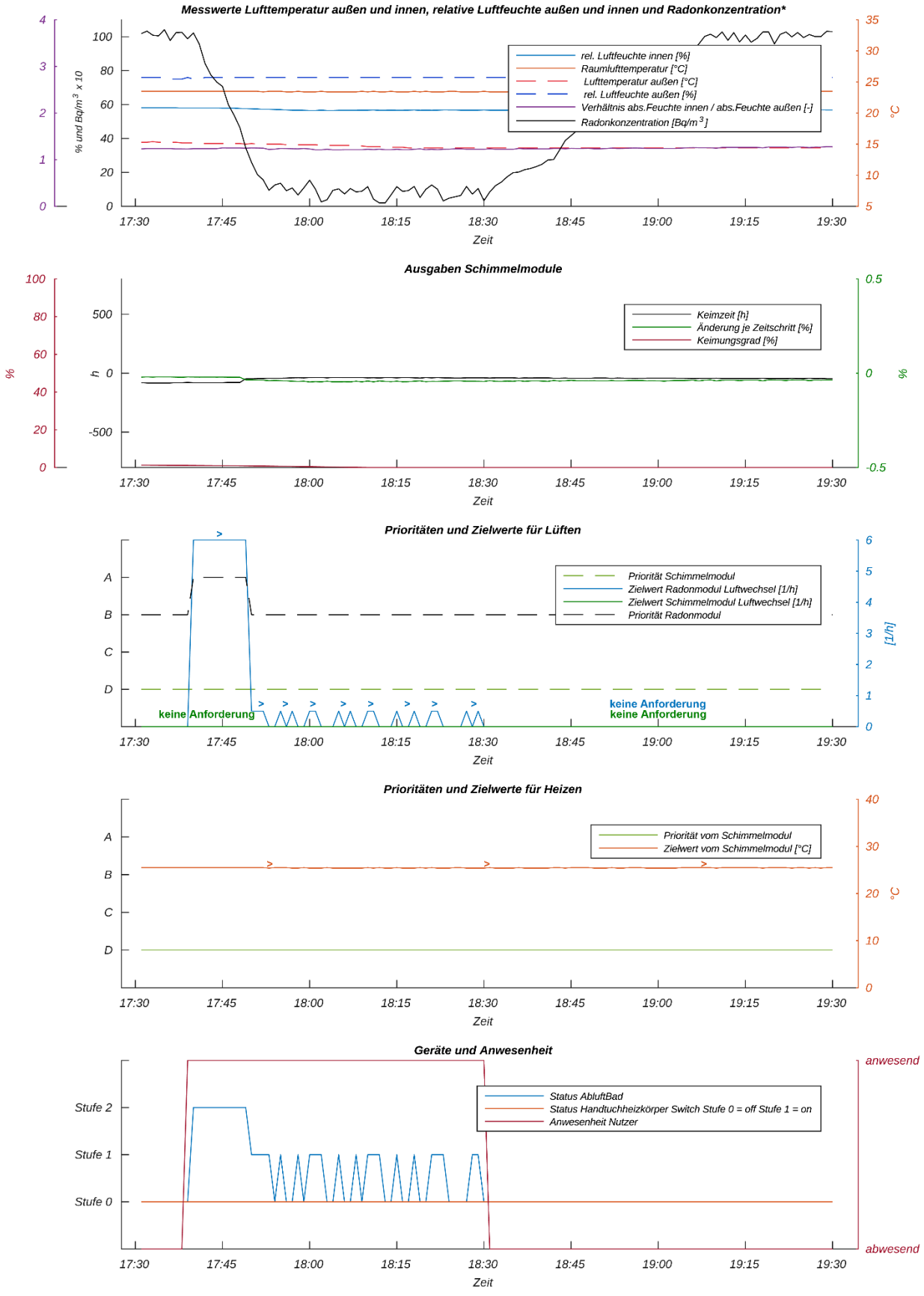
Für alle vier Szenarien muss zuerst die Ontologie als Grundlage für das Berechnungssetup erstellt werden. Die folgende Grafik zeigt die Ontologie nach dem sechsstufigen Reasoningprozess. Es ist erkennbar, dass sowohl die Zuordnung der Sensorwerte als auch die Modulinteraktionen sowie die Interaktionen des DGZ und des DNZ entsprechend der nun veränderten Modulkonfiguration gefunden wurden. Auch die Zuordnung der Wirkmodule und die Zuordnung der dazu passenden Geräte konnte wie geplant automatisch erfolgen.

Abb. 55 Vollständige Ontologie mit Verbindungen durch Reasoning der Fallstudie 2

Auf ein Reasoning der siebten Stufe wird aus den im Beispiel zuvor bereits genannten Gründen verzichtet. Nach Erstellung des Rechensetups und nachfolgender Berechnung der Hilfs- und Funktionsmodule konnte das folgend abgebildete Wirkgrößenarray für den Fall eines anwesenden Nutzers ermittelt werden. Folgend wird die Untersuchung der einzelnen bereits genannten Szenarien aufgeführt.

Szenario1

In diesem Beispiel wurde als Randbedingung davon ausgegangen, dass eine in Bezug auf die Schimmelbildung unkritische Situation vorliegt. Die Radonkonzentration ist jedoch mit ca. 1000 Bq/m³ vergleichsweise hoch. Zu Beginn der Messung war der Nutzer abwesend. Der Status der Anwesenheit änderte sich jedoch im Laufe der Messung. Mit diesem Szenario kann die grundsätzliche Funktionalität des Radon Moduls sowie die Reaktion auf die kontextsensitive Größe der Nutzerabhängigkeit demonstriert werden. Der Verlauf der gesamten Messung ist in untenstehender Grafik visualisiert.



* Siehe Kapitel 6.2.2

Abb. 56 Änderung der Nutzeranwesenheit und Darstellung der resultierenden und relevanten Berechnungsergebnisse des Radonmoduls

Es ist deutlich zu erkennen, dass die Priorität und die Handlungsanforderung ab dem Zeitpunkt der Anwesenheit des Nutzers um 17:38 Uhr verändert wird. Die daraus resultierende Steuerung des Luftwechsels führt zu einer schnellen Reduktion der Radonkonzentration in der Luft. Mit der eintretenden Abwesenheit des Nutzers ändern sich erwartungsgemäß die Priorität und die Handlungsanforderung, weshalb die Lüftung abgestellt wird. Auch dieser Zusammenhang ist ab 18:30 Uhr zu erkennen.

Zur Verdeutlichung der Modulausgaben wird nachfolgend noch der Output des Radonmoduls tabellarisch angegeben. Gewählt wurde der Zeitpunkt, als die Anwesenheit des Nutzers detektiert wurde.

Tabelle 34 MATLAB-Radonmodul Ausgabe des Wirkgrößenarrays für den Luftwechsel der Fallstudie 2 bei anwesendem Nutzer und zu hoher Radonkonzentration

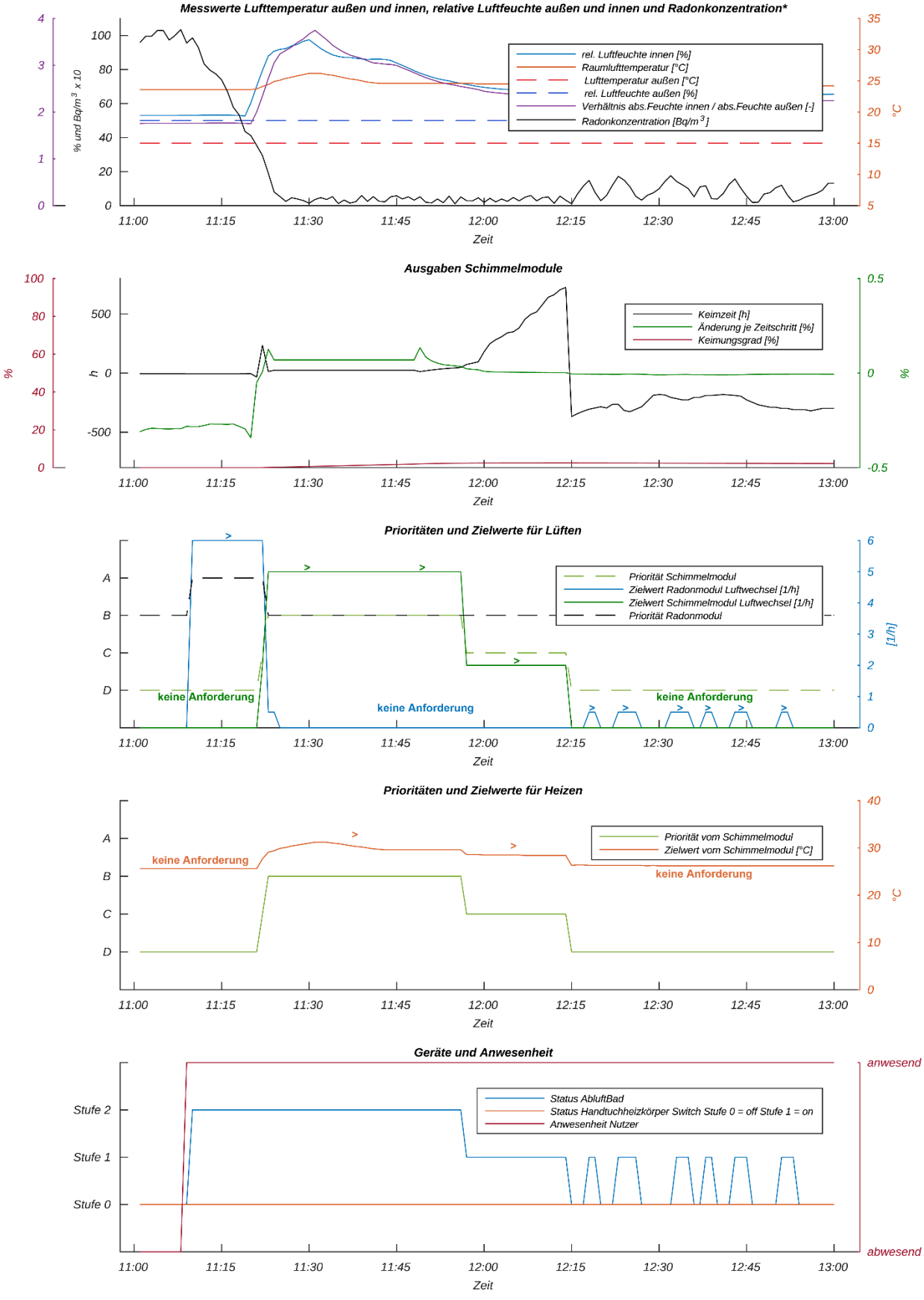
Aktion	Zielwert	Abweichung	Priorität	Radonkonzentration	Anwesenheit Nutzer
'n >'	6 - ∞	0,05	‘A‘	988	Ja
'n >'	6 - ∞	0,10	‘A‘	988	Ja
'n >'	6 - ∞	0,15	‘A‘	988	Ja
'n >'	6 - ∞	0,20	‘A‘	988	Ja
'n >'	6 - ∞	0,25	‘A‘	988	Ja
'n >'	6 - ∞	0,30	‘A‘	988	Ja
'n >'	6 - ∞	0,35	‘A‘	988	Ja
'n >'	6 - ∞	0,40	‘A‘	988	Ja
'n >'	6 - ∞	0,45	‘A‘	988	Ja
'n >'	6 - ∞	0,50	‘A‘	988	Ja

Während der weiteren Anwesenheit des Nutzers schwankt die Radonkonzentration um einen Wert von 100 Bq/m³. Ursächlich dafür ist die regelmäßige automatische Erhöhung und Reduktion des Luftwechsels bei Über- oder Unterschreiten des voreingestellten Schwellwertes von 100 Bq/m³. In Abb. 56 ist dieser Zusammenhang in der Zeit zwischen 17:52 Uhr und 18:30 Uhr deutlich zu erkennen. Die Lüftung wird regelmäßig aktiviert, um die immer wieder steigende Radonkonzentration zu regulieren.

Auf Grund der Ergebnisse des ersten Szenarios kann demnach von einer Funktionsfähigkeit des Radonmoduls sowie der kontextbasierten Steuerung durch den Gesamtalgorithmus gesprochen werden.

Szenario 2

Das zweite Szenario soll das Verhalten der Methodik bei gleichzeitigem Eintreten einer zu hohen Radonbelastung und eines Duschereignisses darstellen. Zu diesem Zweck wurde das Szenario eines nach Hause kommenden Nutzers und einem anschließend auftretenden Duschereignis abgebildet. Die unten abgebildete Grafik zeigt die im Rahmen dieses Szenarios erfassten Messdaten und Programmausgaben.



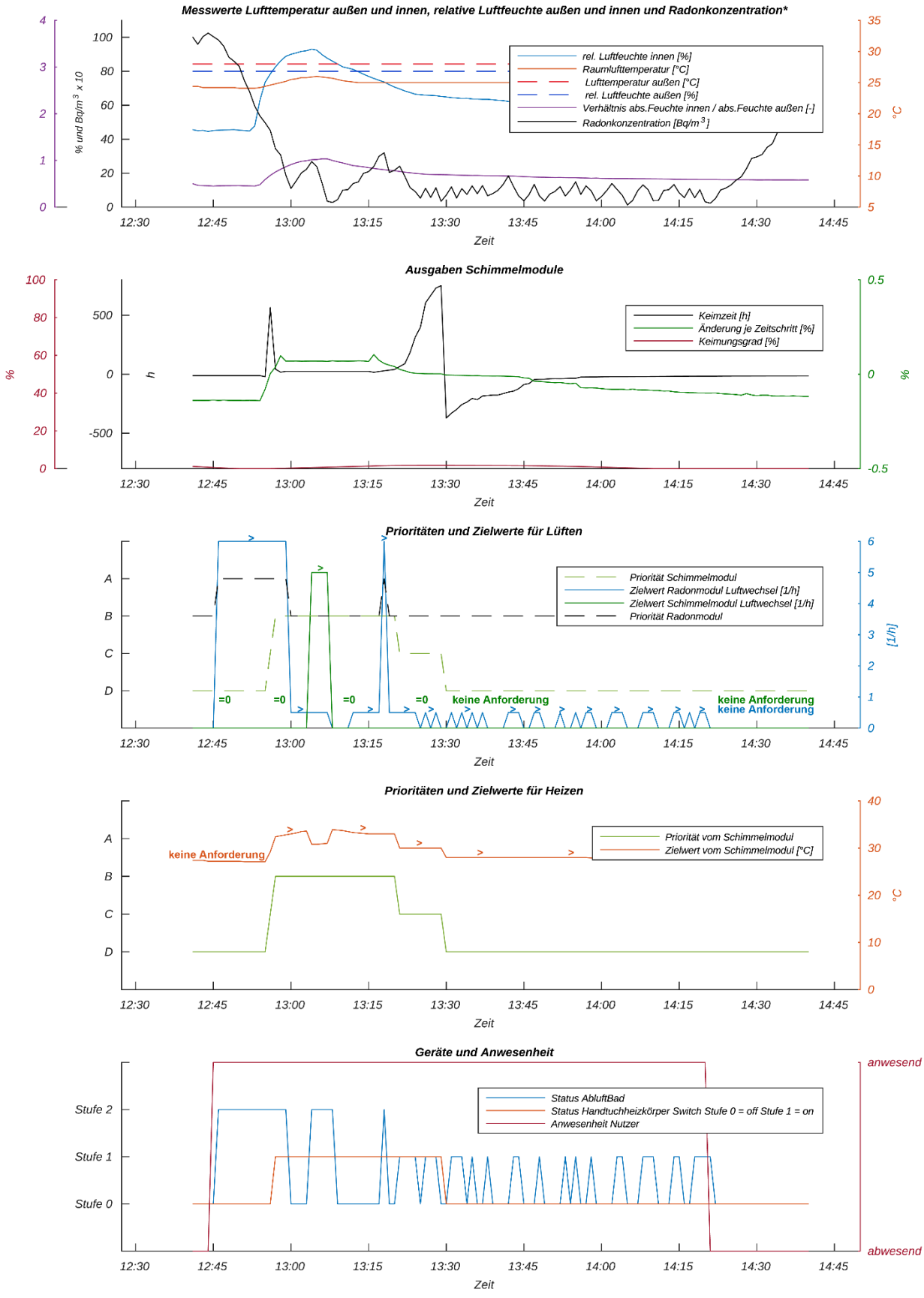
* Siehe Kapitel 6.2.2

Abb. 57 Änderung der Nutzeranwesenheit und darauffolgender Duschvorgang, Darstellung der relevanten Berechnungsergebnisse des Gesamtsystems

Der Grafik kann entnommen werden, dass wegen zu hoher Radonkonzentration sofort bei Detektieren der Anwesenheit des Nutzers um 11:08 Uhr die Lüftung in der höchsten Stufe aktiviert wurde. Zu diesem Zeitpunkt stellt das Schimmelmodul keine Anforderung an die Lüftung und das Heizen. Mit Einsetzen des Duschereignisses um 11:20 Uhr ergibt sich jedoch eine für die Sporenkeimung ideale Umgebung und die Keimzeit sinkt deutlich. Dies führt erwartungsgemäß zu der Anforderung eines hohen Luftwechsels. Die Anforderung an das Heizen wird hier ignoriert, da die präferierte Wirkgröße (Lüften) umgesetzt werden kann. Das Schimmelmodul hält diese Anforderung noch solange aufrecht, bis die hohe Luftfeuchtigkeit abtransportiert werden konnte, auch wenn die Radonbelastung bereits keine Rolle mehr spielt. Daraus resultiert ein Schwanken der Radonkonzentration in diesem Zeitraum zwischen 0 und 60 Bq/m³. Der Luftwechsel wird erst reduziert als die Keimzeit wieder höher und im weiteren Verlauf negativ wird. Zuerst um 11:56 Uhr und ein weiteres Mal um 12:14 Uhr. Ab dem ersten Zeitpunkt hat die Radonkonzentration wieder eine größere Relevanz als die Schimmelbildungsgefahr, was an der unterschiedlichen Priorität zu erkennen ist. Im weiteren Verlauf der Messung wird, wie bereits im vorherigen Beispiel, regelmäßig bei Überschreiten der definierten Maximalkonzentration die Lüftung aktiviert, um die Radonbelastung zu reduzieren. Es kann also festgehalten werden, dass die beiden Module wie vermutet kooperieren und sich im hier gezeigten Szenario nicht gegenseitig behindern. Die Funktionsfähigkeit der Prioritätenverarbeitung ist damit für das hier aufgeführte Szenario nachgewiesen.

Szenario 3

im letzten Szenario dieser Fallstudie wird ein kritischer Fall untersucht. Bei zu hoher Außenluftfeuchte würde, aufgrund der niedrigen Oberflächentemperaturen, ein Lüften den Anstieg des Keimungsgrades verursachen. Das Schimmelmodul wird daher diese Wirkgröße verhindern wollen und stattdessen ein Heizen fordern. Dies steht jedoch bei einer zu hohen Radonkonzentration im Widerspruch zu den Anforderungen des Radonmoduls, welches einen Lüftungsvorgang vorschlägt. Folgende Abbildung zeigt den Sachverhalt bei einem Szenario, in dem der Nutzer bei zu hoher Radonkonzentration nach Hause kommt und einen Duschvorgang beginnt.



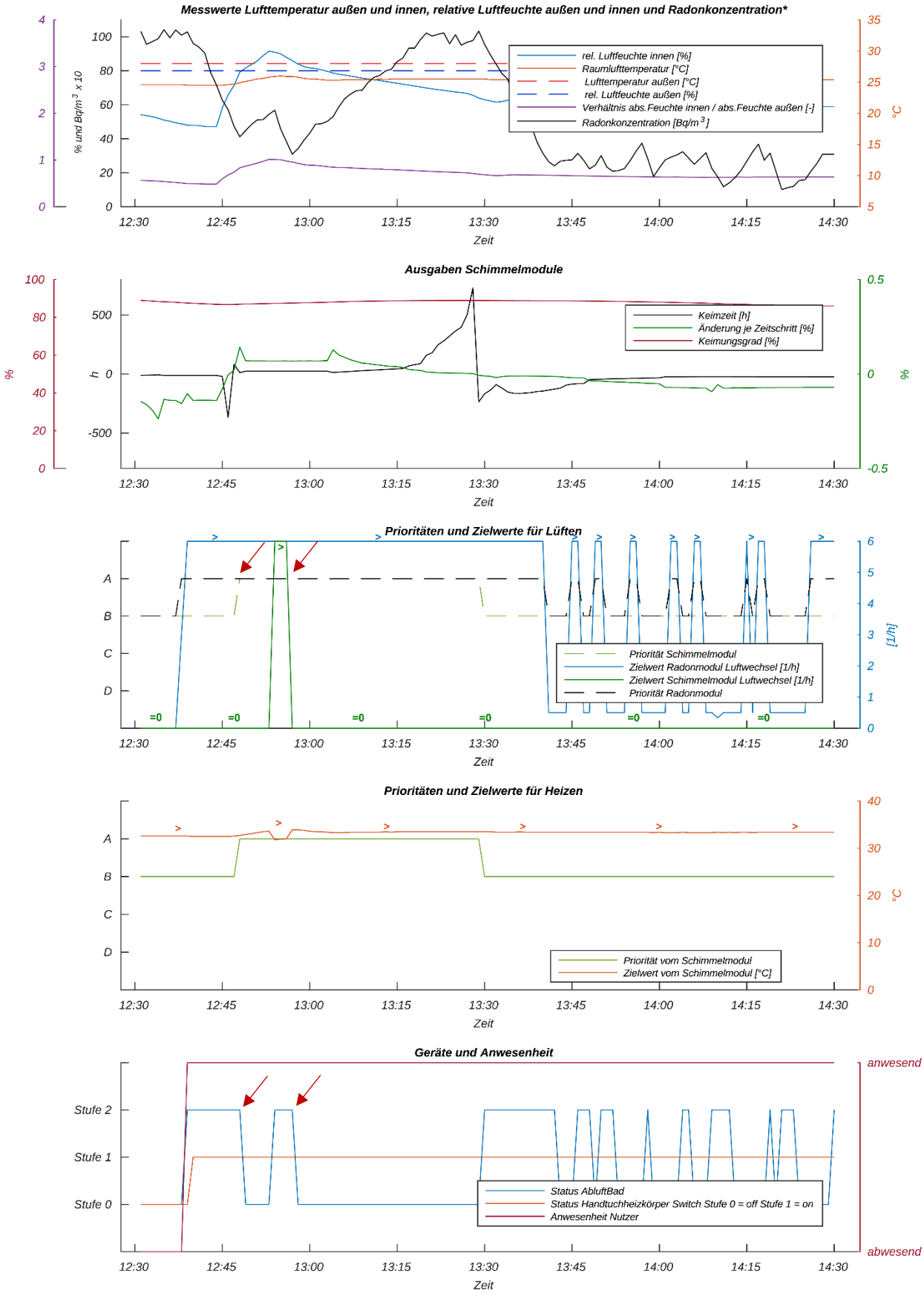
* Siehe Kapitel 6.2.2

Abb. 58 Änderung der Nutzeranwesenheit und Duschvorgang, Darstellung der relevanten Berechnungsergebnisse des Gesamtsystems bei hoher Außenluftfeuchte

Zu Beginn der Messung wird wegen einer zu hohen Radonkonzentration gelüftet. Das Schimmelmodul hat zu diesem Zeitpunkt die Anforderung eines Luftwechsels von null und keine Anforderung für das Heizen, da die Randbedingungen keine Schimmelbildung begünstigen. Mit Einsetzen des Duschereignisses um 12:53 Uhr fordert das Schimmelmodul weiterhin wegen der hohen Außenluftfeuchte eine Reduktion des Luftwechsels auf null, allerdings mit einer höheren Priorität von „B“. Da die Priorität beim Radonmodul größer ist, wird jedoch trotzdem gelüftet. Zur Kompensation der nicht erfüllten Forderung des Schimmelmoduls wird die alternative Wirkgröße zur Steuerung verwendet und deshalb die Heizung aktiviert. Durch die steigende Luftfeuchte und Raumtemperatur existiert zwischen 13:03 Uhr und 13:06 Uhr ein kurzer Zeitpunkt, in dem das Schimmelmodul eine Lüftung fordert. Dies geschieht, da die absolute Feuchte innen nun höher ist als außen. Durch dieses Lüften wird genug Radon abtransportiert, sodass zwischen 13:07 Uhr und 13:10 Uhr die Luftwechselforderung des Radonmoduls sogar auf null herabgesetzt wird. Im weiteren Verlauf wird das Lüften wieder vom Schimmelmodul unterbunden. Daraus resultiert um 13:17 Uhr ein kritischer Wert der Radonkonzentration weshalb die Priorität des Radonmoduls kurzfristig höher steigt als die des Schimmelmoduls. Durch den dadurch entstehenden Luftwechsel kann die Konzentration jedoch sofort verringert werden. Am Ende des Duschereignisses wird die Luftfeuchte so weit reduziert, dass auch die Priorität des Schimmelmoduls wieder auf „D“ sinkt. Da keine Gefahr der Schimmelbildung mehr besteht, wird die Heizung deaktiviert und die Lüftung nur noch durch das Radonmodul gesteuert. Damit ist nachgewiesen, dass das System Prioritäten erwartungsgemäß verarbeiten kann und auch kontextbezogen die Regelungsstrategien verändern und anwenden kann.

Problematisch ist jedoch hierbei, falls ein Nutzer dauerhaft anwesend ist. Dann würde regelmäßig gelüftet werden, wenn die Radonkonzentration zu hoch ist. Zur Reduktion der Schimmelgefahr müsste aber zusätzlich dauerhaft geheizt werden. Sollte die Heizleistung nicht ausreichen, um das Schimmelproblem zu kompensieren, würde kurz vor dem Zeitpunkt einer Sporenkeimung ein Abbruch der Steueraktionen erfolgen. In diesem Fall muss eine Ausgabe an den Nutzer oder einen Experten generiert werden. Um diesen Zeitpunkt bereits frühzeitig abfangen zu können, wäre eine iterative Berechnung des Gesamtsystems notwendig. Damit könnte auf Basis verschiedenster Vorhersagen eine Aussage über die Zukunft getroffen werden und somit ein Abbruch der automatischen Regelung noch vor einem Schaden erfolgen. Derzeit würde ein Abbruch erst erfolgen, wenn das System keine Kapazität mehr hat, um die Temperatur weiter zu erhöhen. Ob dieser Punkt rechtzeitig erreicht wird, hängt vom jeweiligen Gebäude ab und kann nicht generell beantwortet werden. Eine iterative Berechnung wäre daher für die Realisierung eines frühzeitigen Abbruchs erforderlich.

Im letzten Fall wird betrachtet was geschieht, wenn der Keimungsgrad aufgrund einer ungünstigen Vorgeschichte stark erhöht ist und ein Lüften des Schimmelmoduls in Widerspruch mit dem Radonmodul mit selber Priorität tritt. Zu diesem Zweck wurde der vorhergehende Versuch ein weiteres Mal durchgeführt, jedoch wurde der Keimungsgrad zuvor auf 90 % eingestellt.



* Siehe Kapitel 6.2.2

Abb. 59 Hohe Außenluftfeuchte, daher keine Lüftung wegen Schimmelmodul, Keimungsgrad 90 % Dilemma für Radonmodul

Anhand der abgebildeten Grafik ist zu erkennen, dass die hohen Prioritäten beider Module zu einem Ausschluss der Lüftung aus der Regelungsstrategie führen. Eine Lüftung wird daher zwischen 12:47 Uhr und 12:53 Uhr sowie zwischen 12:57 Uhr und 13:28 Uhr unterlassen. Um die Belange des Schimmelmoduls zu berücksichtigen, wird stattdessen geheizt. Zwischen 12:54 Uhr und 12:56 Uhr ist auch hier wieder ein Zeitraum vorhanden, in welchem die Luftfeuchte so hoch ist, dass sie durch Lüften abtransportiert werden kann. Aus diesem Grund wurde kurzzeitig die Lüftung aktiviert. Da das Radonmodul keine weiteren Regelungsstrategien außer Lüften besitzt, wird in den Zeiträumen, in denen keine Luftwechsel erzeugt werden kann, eine Warnmeldung an den Nutzer abgegeben. Die Punkte, an denen diese Meldungen ausgegeben werden, sind jeweils in der Grafik mit einem roten Pfeil markiert.

Die Warnmeldung wurde über eine Sprachausgabe realisiert. Der dafür verwendete Befehl ist unten angegeben. Das Wirkmodul benötigt dazu eine Inputvariable, der beispielsweise Sprachausgabegeräte zugeordnet werden können. So kann über den aufgeführten Befehl eine Benachrichtigung des Nutzers erfolgen.

set SqueezeboxBad talk „Achtung Radonkonzentration kritisch. Bitte Raum verlassen“

Die folgende Abbildung zeigt einen Ausschnitt der Kommandozeile des Servers, auf dem der Gesamtalgorithmus implementiert wurde. Sie verdeutlicht die Ausgabe des oben gezeigten Befehls.

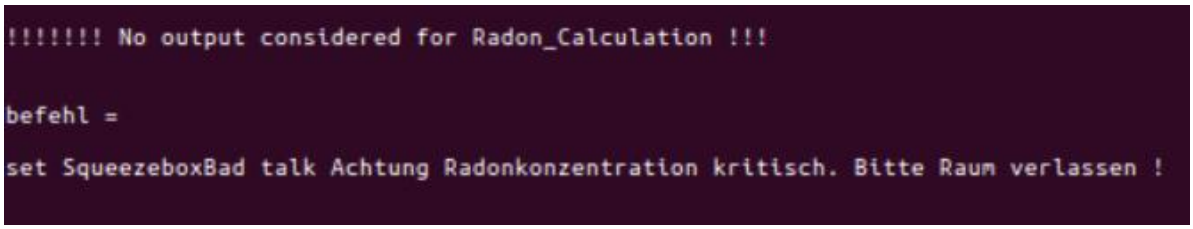


Abb. 60 Screenshot der Kommandozeilenausgabe bei erkennen einer Konfliktsituation

Damit konnte demonstriert werden, dass auch der Ausschluss einer Regelstrategie bei gleich hohen Prioritäten berücksichtigt wird. Sofern eine weitere Möglichkeit zur Lösung der Problematik vorhanden ist, wird diese vom System automatisch gewählt. Dies ist am Beispiel des Schimmelmoduls ersichtlich. Sollte keine weitere Lösung zur Verfügung stehen, bleibt dem System nur noch das Informieren des Nutzers.

Szenario 4

Das Beispiel für den Abbruch durch einen Dilemmacheck würde erfolgen, wenn das Schimmelmodul in der Zukunft eine Lüftung mit Priorität A untersagt und der Nutzer bei zu hoher Radonkonzentration als anwesend prognostiziert wird. Da für die Beseitigung der Radonproblematik nur ein Moduloutput vorhanden ist und die Lüftung wegen drohender Schimmelbildung unterbunden ist, würde dies zu einer Dilemmasituation führen. Diese muss dem Nutzer mitgeteilt werden.

Die folgenden Tabellen zeigen die Moduloutputs inklusive der Prognosen für den beschriebenen Fall.

Tabelle 35 MATLAB Wirkgrößenarray Schimmelmodul

Aktion	Zielwert	Abweichung	Priorität	Keim.-fördernd % je Zeitschritt	Keimzeit in Stunden	Keimungsgrad in %
n=	0	0,05	B	0,12069	13,5466	69,8705
n=	0	0,10	B	0,12069	13,5299	69,9911
n=	0	0,15	A	0,12069	13,5132	70,1118
n=	0	0,20	A	0,12069	13,4966	70,2325
n=	0	0,25	A	0,12069	13,4799	70,3532
n=	0	0,30	A	0,12069	13,4632	70,4739
n=	0	0,35	A	0,12069	13,4466	70,5946
n=	0	0,40	A	0,12069	13,4299	70,7153
n=	0	0,45	A	0,12069	13,4132	70,8360
n=	0	0,50	A	0,12069	13,3966	70,9567

Tabelle 36 MATLAB Wirkgrößenarray Radonmodul

Aktion	Zielwert	Abweichung	Priorität	Radonkonzentration	Anwesenheit Nutzer
'n >'	6 - ∞	0,05	‘A‘	500	Ja
'n >'	6 - ∞	0,10	‘A‘	500	Ja
'n >'	6 - ∞	0,15	‘A‘	500	Ja
'n >'	6 - ∞	0,20	‘A‘	500	Ja
'n >'	6 - ∞	0,25	‘A‘	500	Ja
'n >'	6 - ∞	0,30	‘A‘	500	Ja
'n >'	6 - ∞	0,35	‘A‘	500	Ja
'n >'	6 - ∞	0,40	‘A‘	500	Ja
'n >'	6 - ∞	0,45	‘A‘	500	Ja
'n >'	6 - ∞	0,50	‘A‘	500	Ja

Die prioritätenbereinigte Wirkgröße für den Luftwechsel ist in der folgenden Tabelle abgebildet.

Tabelle 37 MATLAB Wirkgrößenmatrix Prioritätenbereinigt mit Dilemma

Aktion	Zielwert	Abweichung	Priorität
'n >'	6 - ∞	0,05	‘A‘
'n >'	6 - ∞	0,10	‘A‘
X	X	X	X
X	X	X	X
X	X	X	X
X	X	X	X
X	X	X	X
X	X	X	X
X	X	X	X
X	X	X	X

Die mit „X“ gegenzeichneten Zellen zeigen ein prognostiziertes Dilemma für die Module an. Folglich wird vom System per Sprachnachricht auf diesen Zustand hingewiesen. Der dafür ausgegebene Befehl ist nachfolgend abgebildet.

set SqueezeboxBad talk „Achtung Dilemmasituation prognostiziert““

Damit wurde gezeigt, dass die Dilemmaerkennung grundsätzlich funktionsfähig ist. Für den frühzeitigen Abbruch ist jedoch eine interaktive Berechnung erforderlich, die auch die Reaktion des Raumes auf Steueraktionen berücksichtigt. Zudem sind Prognosen und ein iteratives Berechnungsverfahren erforderlich. Die Mittel für solche Berechnungen sind in der hier vorgestellten Methodik bereits vollständig enthalten.

6.4.4 Fazit der Fallstudie 2

Ergänzend zur ersten Fallstudie zeigt die zweite, dass die automatische Zuordnung der Datenströme auch bei einer höheren Anzahl unterschiedlicher Module erfolgt. Durch die Implementierung eines zweiten Wirkmoduls konnte gezeigt werden, dass weitere Aktuatoren automatisch zugeordnet werden können. Mit der zweiten Fallstudie konnte weiterhin erfolgreich nachgewiesen werden, dass die entwickelte Methode mit Hilfe der Verarbeitung grundlegender Prioritäten in der Lage ist, die unterschiedlichen Anforderungen in Form von Wirkgrößen entsprechend ihrer Wichtigkeit zu verarbeiten. Durch die Überprüfung vorhandener Geräte und deren Wirkpotential auf den Ist-Zustand ist damit auch eine Kontextsensitivität, nicht nur in Bezug auf den Nutzer, sondern auch in Bezug auf die Aktionsmöglichkeiten des Gebäudes nachgewiesen. Das System ist in der Lage, auf dynamische Änderungen der Randbedingungen zu reagieren und Entscheidungen entsprechend der individuellen Raumsituation und in Abhängigkeit kontextsensitiver Informationen zu verarbeiten. Durch den modularen Aufbau können Interaktionen verschiedener Module abgebildet werden. Auch Situationen, welche als „Dilemma“ bezeichnet werden, können verarbeitet werden.

6.5 Fallstudie 3: Individuelle Berücksichtigung des Duschverhaltens

Um die Betrachtung der vorhergehenden Fallstudien im Sinne eines individuellen Wohnraums bestmöglich an die tatsächlichen Gegebenheiten und Gewohnheiten des Nutzers anzupassen, sind Untersuchungen der individualisierbaren Parameter erforderlich. Die Berücksichtigung des individuellen Nutzerverhaltens lässt sich durch die, bis jetzt nicht implementierten, Module für maschinelles Lernen und Vorhersagen erreichen. Damit ist es möglich, das jeweilige Nutzerverhalten zu erlernen und in Form von Algorithmen Vorhersagen auf das künftige Nutzerverhalten zu erstellen. Vor allem in Bezug auf die Dilemma-Prüfung ergeben sich so Möglichkeiten, bereits frühzeitig auf ein, im Sinne der Bauphysik, ungünstiges Verhalten hinzuweisen.

6.5.1 Einleitung zur Fallstudie 3

Die Implementierung eines beispielhaften Modells, das das individuelle Nutzerverhalten widerspiegelt, wird in dieser Fallstudie am Beispiel der durchschnittlichen Duschzeit und der vorhergesagten Duschereignisse eines Nutzers demonstriert. Damit wird die Möglichkeit erzeugt, eine individualisierte Berechnung hygrothermischer Zusammenhänge im Wohnraum durchzuführen. Zu diesem Zweck wurde die Grundlage für die Auswertung der vergangenen Luftfeuchteverläufe im Badezimmer der Testwohnung und der Vorhersage von Duschereignissen ermittelt (Altfelix, 2017). Die Einspeisung der vorhergesagten Duschereignisse und die Berechnung der darauf resultierenden, prognostizierten Verläufe der relativen Luftfeuchtigkeit zeigen zusätzlich eine erweiterte Möglichkeit auf die individuelle Ausprägung von nutzerinduzierten Handlungen einzugehen. Damit wird die Möglichkeit der kontextsensitiven Betrachtung des Wohnraumes um die Möglichkeit einer Individualisierung auf einen speziellen Anwendungsfall erweitert.

6.5.2 Systemaufbau der Fallstudie 3

Um die Individualisierbarkeit von Nutzerparametern zu demonstrieren, wurde die durchschnittliche Duschzeit eines Nutzers aus den in einem Smarthome gewonnenen Daten ermittelt. Die Ermittlung erfolgte dabei automatisch durch einen Algorithmus. Zusätzlich wurde ermittelt, wie häufig und wann Duschereignisse auftraten (Altfelix, 2017). Daraufhin wurde eine Vorhersage erstellt, mit welcher die Beschreibung des nutzerindividuellen Duschverhaltens möglich ist.
Um die Auswirkungen auf das Gebäude abzubilden, wurde von Turan (2017) zusätzlich ein Algorithmus erstellt, welcher aus den Daten über das Auftreten einer Feuchtequelle den prognostizierten Verlauf der relativen Luftfeuchtigkeit, in Form einer Outputvariable, ausgeben kann. Dafür werden neben einigen Daten aus dem BIM-Kontext, wie die Größe der Öffnungen im Duschkopf und die Anzahl der Wasserstrahlen, auch die durchschnittliche Duschdauer als Eingangsvariable benötigt. Damit ist eine Vorhersage unter Berücksichtigung des konkreten individuellen Nutzerverhaltens und der damit eintretenden Wirkung auf den Raum möglich.
Dieser Algorithmus konnte nun als Erweiterung für das Wirkmodul, welches den Luftwechsel kalkuliert, implementiert werden. Durch die Rückgabe der Outputvariable ist die Möglichkeit für die bereits angesprochene iterative Berechnung oder einer eventuellen Kopplung mit Simulationsmodellen für die Prognostizierung geschaffen. Folgend wird dieser Prozess genauer erläutert.

6.5.3 Umsetzung

Durch die Analyse, der im Folgenden beispielhaft abgebildeten Temperatur- und Luftfeuchteverläufe war es möglich, die Duschereignisse in einem vergangenen Messzeitraum zu ermitteln.

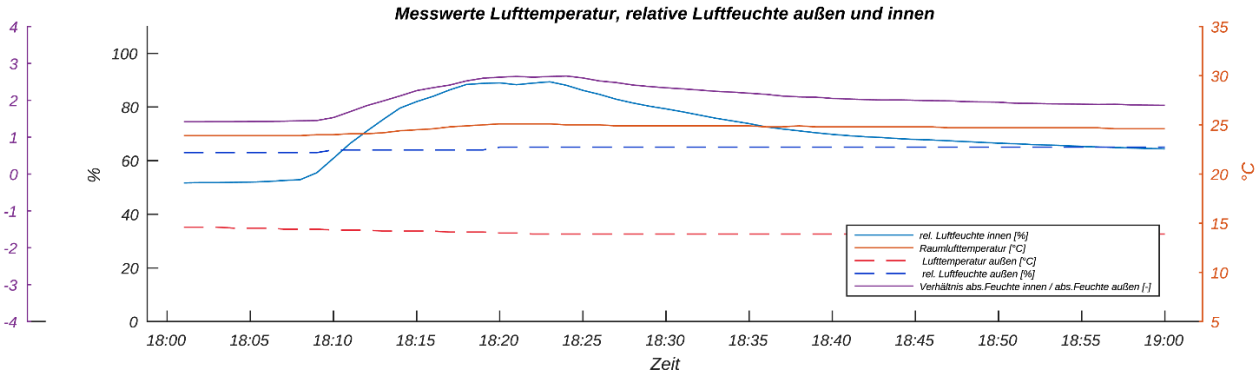


Abb. 61 Temperatur und Luftfeuchteverlauf eines Duschereignisses

Die Untersuchung ergab eine Übersicht der Duschereignisse in ihrer Häufigkeit sowie ihrer Länge. Damit war eine Datengrundlage für weitere Untersuchungen dieses nutzerindividuellen Verhaltens möglich. Die folgenden Abbildungen zeigen die gewonnenen Daten für den Untersuchungszeitraum von Anfang bis Mitte 2016. Zu erkennen ist sowohl die Verteilung der Duschereignisse mit ihrer jeweiligen Länge in Abb. 62 als auch die durchschnittliche Verteilung der Duschlänge über alle Ereignisse in Abb. 63.

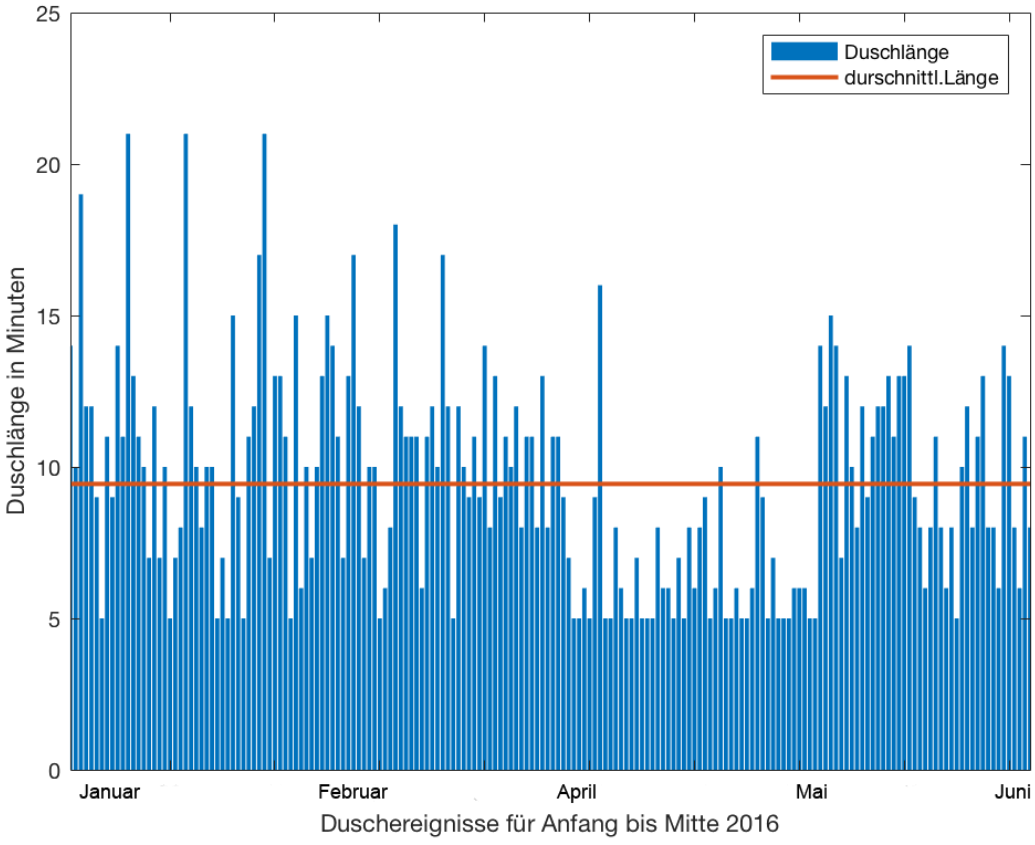


Abb. 62 Duschereignisse und Länge im Zeitraum Anfang bis Mitte 2016 (Turan, 2017, S. 59)

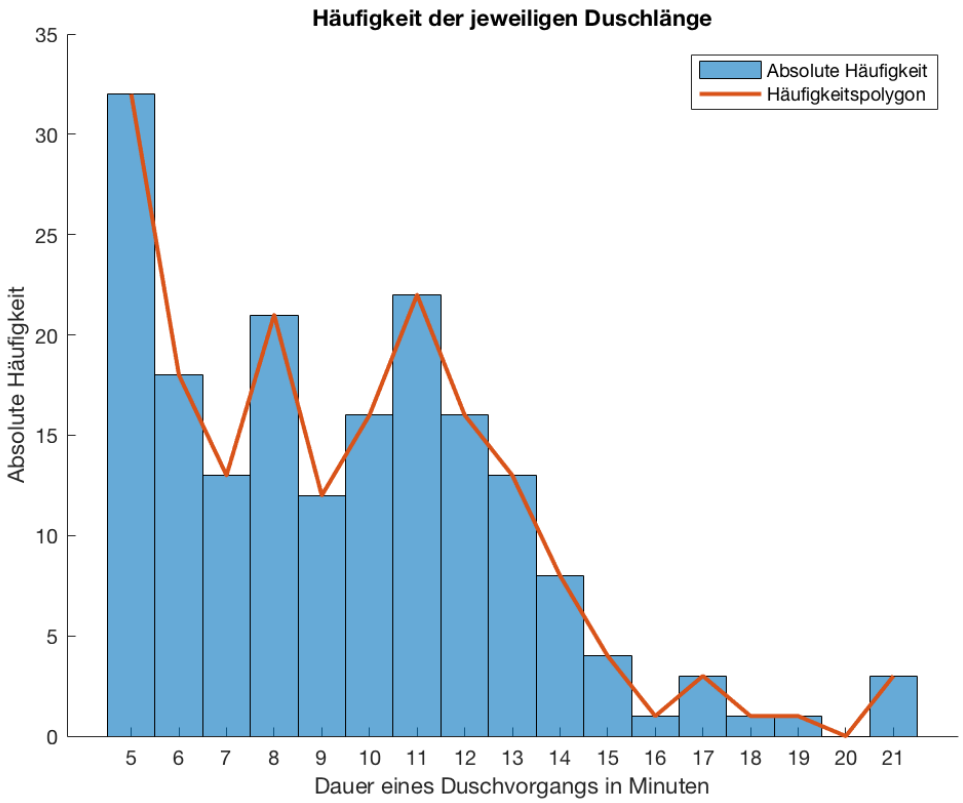


Abb. 63 Absolute Häufigkeit der Dauer eines Duschvorgangs (Turan, 2017, S. 59)

Zusätzlich konnte aus den Daten die durchschnittliche Duschzeit mit 9 Minuten ermittelt werden (Turan, 2017, S. 59). Damit weicht diese um 2 min vom deutschen Durchschnittswert nach unten ab und stellt somit eine nutzerindividuelle Größe dar. Mithilfe dieser Daten ist es nun möglich eine Prognose der in Zukunft auftretenden Ereignisse, in Form eines Vektors mit Zeitstempeln, zu erstellen.

Diese Ausgabe ist in der Ontologie als Outputvariable zu definieren. Die dazu passende Gegenschnittstelle am Wirkmodul ist der Input für die Verarbeitung der Quellenereignisse zu einer Prognose der relativen Feuchte. Nachfolgende Abbildung verdeutlicht den Zusammenhang am Beispiel der Ontologie. Auf eine Abbildung der vollständigen Ontologie wurde an dieser Stelle aus Gründen der Übersichtlichkeit verzichtet.

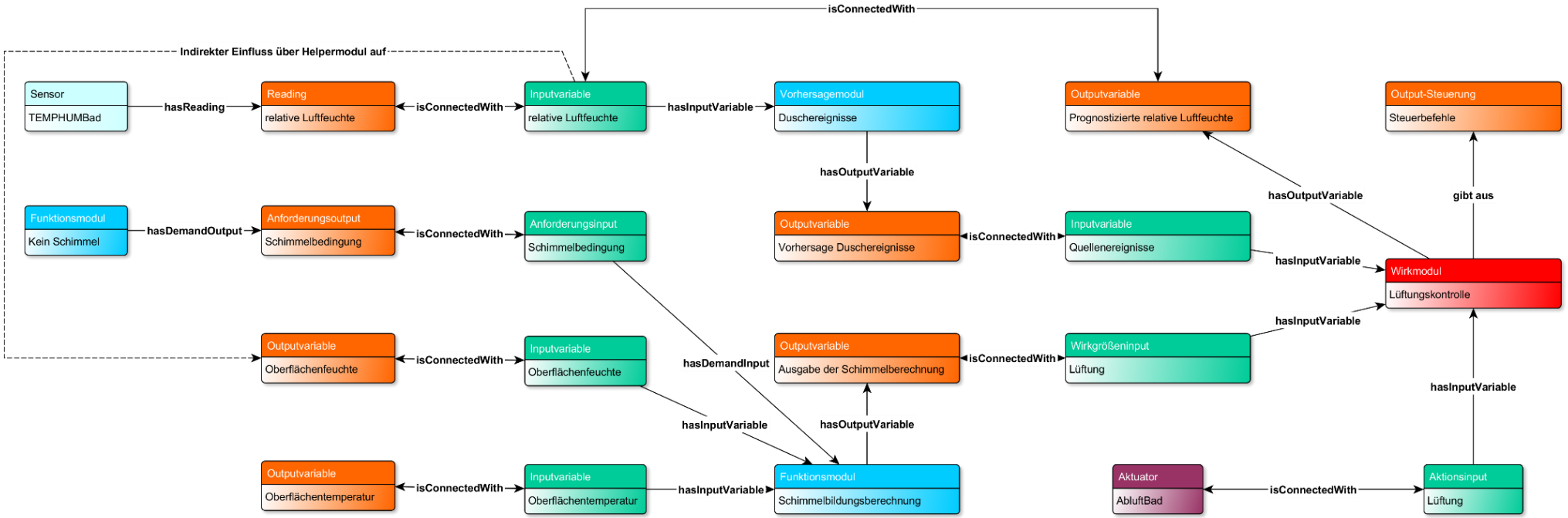


Abb. 64 Ontologie mit Reasoning der Fallstudie 3

Am Beispiel der Ontologie wird sichtbar, dass eine erfolgreiche Zuordnung der Outputvariable des Moduls zur Bestimmung der Duschereignisse in der Zukunft zur entsprechenden Inputvariable erstellt werden kann. Mithilfe der hier übergebenen Daten kann nun, wie bereits beschrieben, der Einfluss dieser zukünftigen Duschereignisse auf das Raumklima bei sonst stationären Bedingungen ermittelt werden. Diese Verarbeitung erfolgt direkt im Wirkmodul. Die Vorhersage gestaltet sich in Form der prognostizierten relativen Luftfeuchte-Peaks, aufgrund von zukünftigen Duschereignissen. Diese sind in Abb. 65 dargestellt. Darauf aufbauend wurde die absolute Feuchte im gleichen Zeitraum automatisch prognostiziert. (Turan, 2017)

Für die Ermittlung wurde auf eine Abbildung von Feuchtesenken verzichtet und eine generische Senke erstellt, die den Luftfeuchteverlauf nach Abschluss des Duschvorganges wieder auf den Jahresmittelwert der Luftfeuchte zurückführt. Dieses Vorgehen erzeugt die in Abb. 65 abgebildeten Luftfeuchte-Peaks. Die Vorgehensweise dient hier zur Überprüfung der Funktionsfähigkeit der Quellenermittlung. Für eine realistische Ermittlung des prognostizierten Luftfeuchteverlaufs ist die Berücksichtigung von Luftfeuchtesenken und deren Vorhersage in gleicher Weise erforderlich wie die Vorhersage der Quellen. Die Vorgehensweise ist dabei jedoch identisch. Die folgende Grafik kann daher die erfolgreiche Ermittlung, der für einen Duschvorgang markanten Luftfeuchte-Peaks nachweisen.

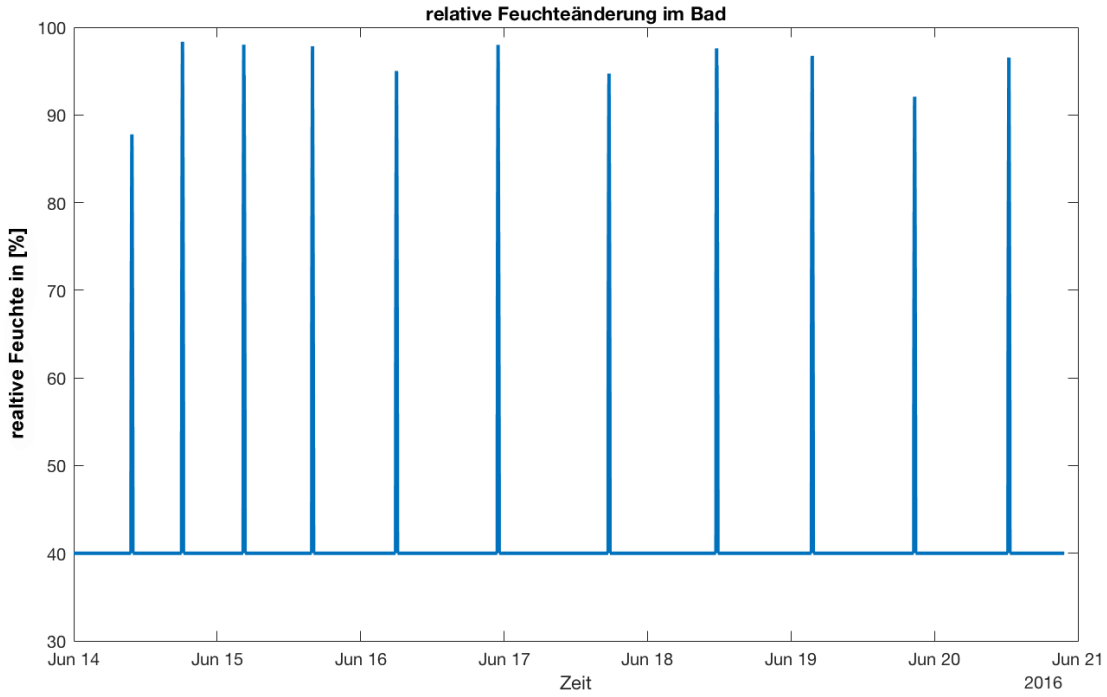


Abb. 65 Prognostizierter Verlauf der relativen Luftfeuchte (Turan, 2017, S. 65)

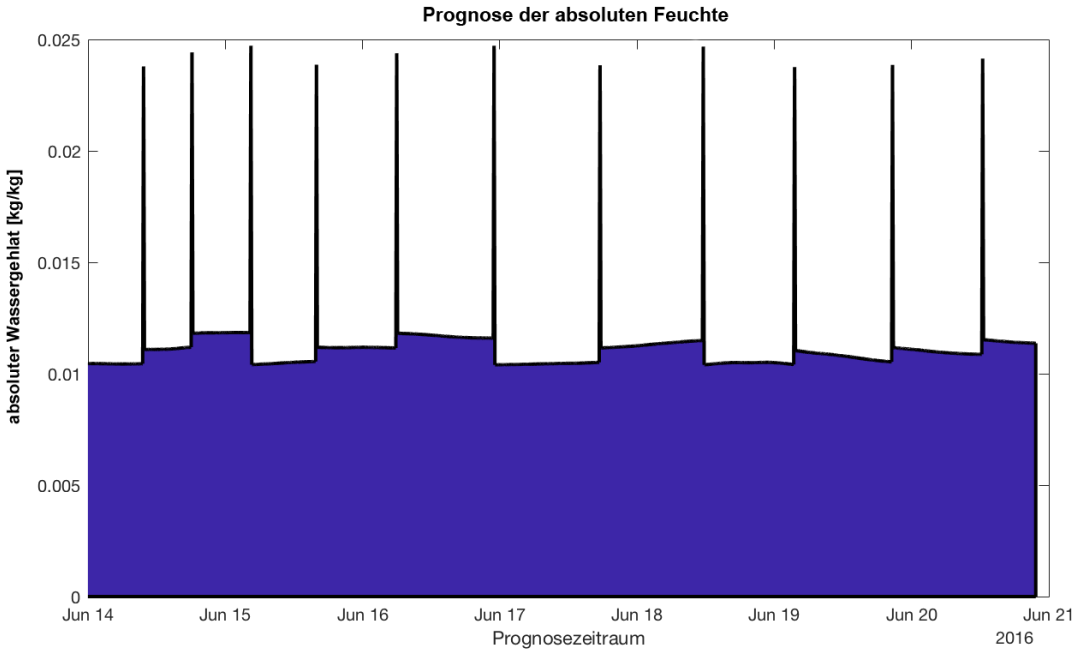


Abb. 66 Prognostizierter Verlauf der absoluten Feuchte (Turan, 2017, S. 66)

Im Nachhinein erfolgte eine Untersuchung der Prognosen mit den daraufhin eingetretenen Realdaten. Diese zeigte, dass die getroffenen Annahmen zum nicht erfassten Verhalten von Senken zu einer Überschätzung des durchschnittlichen relativen Feuchtegehaltes der Luft führten. Daher wurde, dieses Senkenverhalten, wie bereits erläutert, durch Zurücksetzen der relativen Luftfeuchte auf den Durchschnittswert simuliert.
Die Häufigkeit der auftretenden Duschereignisse ergibt aber eine realitätsnahe Prognose für den gesamten Betrachtungszeitraum und scheint damit ausreichend genau für eine Vorhersage der Schimmelbildungsgefahr. Das punktgenaue Aufeinandertreffen von prognostizierten und realen Duschereignissen ist für diesen Anwendungszweck nicht von Interesse. Die folgend abgebildete Grafik zeigt den Vergleich der Vorhersage (in blau) mit der später eingetretenen Realität (in lila).

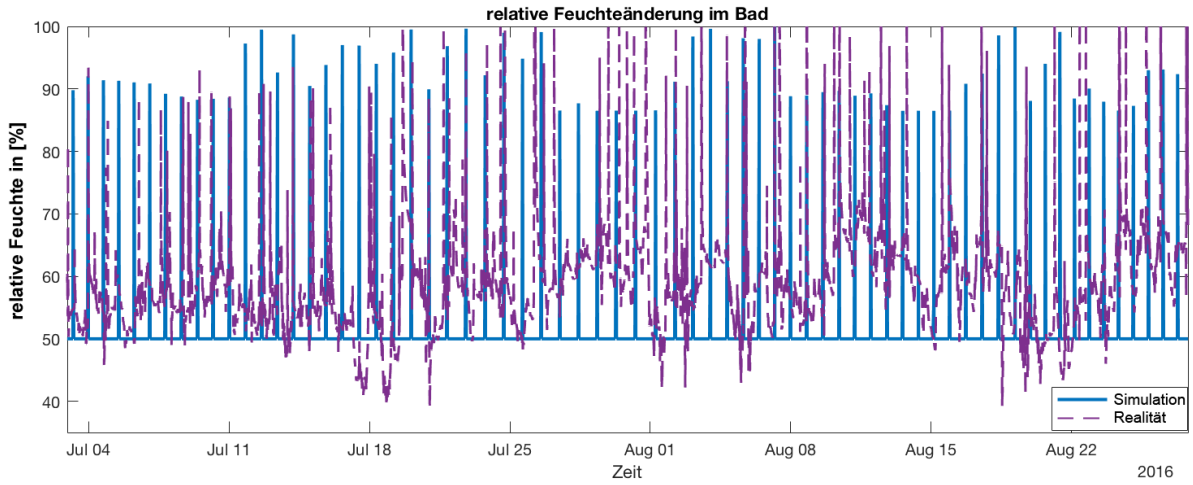


Abb. 67 Vergleich zwischen prognostiziertem und realem relativen Luftfeuchteverlauf (Turan, 2017, S. 71)

Bei einem Vergleich mit der durchschnittlichen Duschhäufigkeit in Deutschland (SBZ, 2008), laut der nur ca. 64% der Personen täglich duschen (Grohe-AG, 2008), wird ersichtlich, dass durch die Verwendung von Ersatzdaten die Duschhäufigkeit unterschätzt wird. Außerdem ist im direkten Vergleich ersichtlich, dass durch das Verwenden der Ersatzdaten die Duschlänge im Vergleich mit dem statistischen Durchschnitt von 11 Minuten überschätzt wird. Da die relative Luftfeuchtigkeit einer der Hauptparameter bei der Berechnung der Schimmelbildungsgefahr ist, kann davon ausgegangen werden, dass dies zu signifikanten Veränderungen in der Bewertung dieser Gefahr führt. Durch die Ermittlung der Duschzeit und -häufigkeit konnte so nachgewiesen werden, dass eine Individualisierung von Nutzerparametern auch zu einer Individualisierung der für Steueraktionen relevanten Werte führt. Hierdurch werden die Steuerungen im Smarthome, im Vergleich zur Verwendung von Ersatzdaten, besser an das Verhalten des Nutzers angepasst. Die Ausgabe der Ergebnisse muss hier mit dem bereits beschriebenen erweiterten Wirkmodul erfolgen und erzeugt so die Möglichkeit die Untersuchung auf eine Schimmelgefahr ähnlich der Herangehensweise kommerzieller Softwarelösungen wie WUFI-Bio (Fraunhofer-Institut für Bauphysik IBP, 2019) durchzuführen. Dies geschieht durch die Zuweisung der Outputvariable des Wirkmoduls zu einer Inputvariable anderer Module. So kann eine Einspeisung dieser Prognosedaten in den Gesamtablauf der Algorithmik erfolgen. Die erfolgreiche Zuordnung kann der bereits gezeigten Abbildung, welche die gesamte Ontologie der Fallstudie 3 zeigt, entnommen werden.

6.5.4 Fazit der Fallstudie 3

In dieser Fallstudie konnte erfolgreich die Möglichkeit der automatischen Ermittlung von individuellen Nutzerparametern aus Sensordaten gezeigt werden. Weiterhin wurde die für einen iterativen oder simulationsgestützten Berechnungsprozess erforderliche automatische Zuordnung der Outputvariablen eines Wirkmoduls, anstatt eines Sensorwertes, zu den Inputvariablen der Hilfs- und Funktionsmodule erfolgreich umgesetzt. Damit konnte die Bereitstellung einer Datenbasis für Prognosebetrachtungen demonstriert werden. Zusätzlich wurde gezeigt, dass die Verwendung von Messdaten aus der Vergangenheit für neue Fragestellungen automatisiert möglich ist. Ergebnisse können in einem Format extrahiert werden, das keine interaktionsbedingten Zusammenhänge abbildet und im DNZ abgespeichert werden. Somit ist auch bei einem Umzug bestimmter Personen die Überführung persönlicher Nutzerpräferenzen möglich. Die hier gezeigte Vorgehensweise entspricht der Verarbeitungsweise wie sie Algorithmen des maschinellen Lernens verwenden. Deren Einbindung in das Gesamtkonzept ist somit möglich. Durch die Verwendung individueller, auf den Wohn- und Nutzerkontext bezogener Prognosen kann also eine starke Individualisierung der resultierenden Steuerungen und Untersuchungen in Bezug auf die Verarbeitung von Prioritäten und der „Dilemmacheck“ erfolgen.

6.6 Bewertung der Fallstudien

Mit den hier aufgeführten Fallstudien konnte mittels einfacher Zusammenhänge die Grundfunktionalität der vorgestellten Methodik evaluiert werden. Es zeigte sich, dass eine automatische Zuordnung der relevanten Datenströme möglich ist und keine Fehlzuordnungen aufgetreten sind. Während der Bearbeitung der Fallstudien zeigte sich auch, dass vor allem bei der Definition der Eigenschaften sehr sorgfältig vorgegangen werden muss. Grundsätzlich zeigen die Ergebnisse der Fallstudie 1 jedoch die Grundfunktionalität des Gesamtalgorithmus. Durch die automatische Zuordnung und Umformung von Datenströmen ist eine Interoperabilität eines solchen Systems gewährleistet. Mit Hilfe der zweiten Fallstudie konnte weiterhin demonstriert werden, dass auch bei mehreren Modulen eine automatische prioritätengestützte Entscheidungsfindung durch das System möglich ist. Fraglich bleibt jedoch, ob die bisherige Abstufung der Prioritäten bei einer hohen

Zahl an implementierten Modulen ausreichend ist. Weitere Test müssen zeigen, ob hier eine Anpassung nötig ist.
Durch die Reaktion auf die Anwesenheit des Nutzers konnte eine kontextsensitive Steuerung und deren Implementierung in das Gesamtsystem demonstriert werden. Da die Anwesenheit einen Datenstrom ähnlich einem Sensorwert mit einem definierten Trigger darstellt, unterscheidet sie sich unwesentlich von anderen Datenströmen. Bei einer sinnvollen Repräsentation aller relevanten und kontextsensitiven Größen, in Form einer Instanz im Automationsserver und darauf abgerichteter Triggermomente, ist ein Auslösen der nötigen Rechenaktionen bei Veränderungen im Kontext sichergestellt. Die Erstellung auslösender Trigger wurde in den Fallstudien jedoch zu Testzwecken händisch implementiert.
Der in den Fallstudien verwendete Radonsensorwert entstand, wie bereits erläutert aus einem eigenständigen Bilanzmodell. Für eine produktive Anwendung der hier vorgestellten Methode in Bezug auf die Radonkonzentration müsste daher erst ein geeigneter Sensor gefunden und implementiert werden. Auch wurden Aspekte wie die Positionierung und Dimensionierung solcher Sensoren und deren Genauigkeit hier nicht im Detail betrachtet, die jedoch für reale Anwendungen berücksichtigt werden müssen.
Durch die dritte Fallstudie konnte darauf aufbauend gezeigt werden, wie mittels maschinellem Lernen Daten nachträglich verarbeitet werden können. Durch die Aufbereitung vergangener Messdaten konnte so erfolgreich das Duschverhalten eines Nutzers auf einfache Weise beschrieben werden. Der hier gezeigte Algorithmus wurde speziell für diese Arbeit entwickelt und es wurde darauf geachtet, dass die Datenströme für das Modul so konzipiert sind, dass sie denen für maschinelles Lernen und auch für „Deeplearning“-Algorithmen gleichen. Eine Übersicht der Brauchbarkeit verschiedener Algorithmen oder Konzepte konnte trotz einer vorhandenen Datenhistorie von mehreren Jahren, aufgrund von zu geringen Datenmengen, nicht erstellt werden. Damit wird deutlich, dass die Entwicklung und das Anlernen solcher Algorithmen nicht für jeden Haushalt einzeln erfolgen können. Aus diesem Grund scheinen Transferlearning-Algorithmen für die Aufgabe am geeignetsten. Trotz allem konnte eine erfolgreiche Einbindung der dafür benötigten Module für maschinelles Lernen und Vorhersagen geprüft und somit sichergestellt werden, dass eine Verwendung nach deren Entwicklung möglich ist.
Mit diesen drei Fallstudien wurde daher der Gesamtalgorithmus mit seinen einzelnen Komponenten ausreichend evaluiert.

7 Kritische Bewertung der Methodik

Im Folgenden soll nun die Methodik kritisch bewertet werden. Wesentlich ist hierbei ob die zu Beginn gestellten Anforderungen erfüllt wurden. Noch vorhandene Mängel sollen identifiziert und bewertet werden.

7.1 Umsetzung des Kriterienkataloges

Zu Beginn wurden in Form eines Kriterienkataloges folgende Maßstäbe für die Methode aufgestellt:

- Modular und skalierbar
- Interaktiv und interdisziplinär
- Offen/Nicht abgeschlossen
- Dynamische Modellierung
- Nutzerindividuell, selbstlernend, vorhersagend
- Nutzerunabhängige Kontextsensitivität

Modularer und skalierbarer Aufbau

Durch die Gliederung aller Komponenten in verschiedene Module und die Definition einer einheitlichen Schnittstelle sowie deren Ausprägung, wurde ermöglicht, dass Module aus allen Sparten miteinander interagieren können. Durch das automatisierte Reasoning wurde zudem die Möglichkeit geschaffen, das System beliebig um Module zu erweitern. So können nachträglich einzelne Module oder auch ganze Funktionen als Modulpakete eingebunden werden, ohne dass eine händische Anpassung erfolgen muss. Module können dabei verschiedene LODs abbilden. Zum einen können sie einfache Zusammenhänge in Form von Formeln enthalten, zum anderen aber auch komplexe Zusammenhänge in Form von Simulationsmodellen. Für das Fehlen von Informationen kann durch Ersatzdaten eine solide Basis geschaffen werden, um die Modulfunktionalität bei Wegfall verschiedener Teilbereiche des Systems möglichst lange aufrechtzuerhalten. Dies alles kann geschehen, ohne den grundsätzlichen Algorithmus ändern zu müssen. Die Arten der Modularität die von Livio Mazzarella (2009) bereits definiert wurden, werden nun kurz noch einmal aufgegriffen.

- **„Functional Layout Modularity“**
  Grundsätzlich ist es möglich, Module zu tauschen, hinzuzufügen oder zu entfernen. Die Struktur wurde so erstellt, dass keine Module auf das jeweilige Gebäude angepasst werden müssen. Zusätzlich ist es möglich gebäudespezifische Module zu integrieren. Da der Algorithmus nicht zwischen den Inhalten eines Moduls unterscheidet, spielt dieser auch für die Modularität keine Rolle. Durch das Einfügen von Modulen zum maschinellen Lernen kann so trotzdem eine Adaption an Individualitäten erfolgen.

  Nachteilig ist jedoch, dass durch diese Methode für den Nutzer nicht ersichtlich ist, welche Module für welche Funktion verantwortlich sind. Das liegt daran, dass Module nicht mehr mit den klassischen Funktionen der Automation vergleichbar sind. Um eine „Functional Layout Modularity“ zu gewährleisten, wurde daher eine für ein Frontend notwendige Zuordnung der Module zu klassischen Funktionen eingeführt, mit der es ähnlich der Paketverwaltung bei Betriebssystemen wieder möglich ist Funktionen und damit alle zugehörigen Module zu aktivieren oder zu deaktivieren.

- **„Mathematical Models Modularity“**
  Diese Art der Modularität kann mit der genannten Methode nachgewiesen werden. Da die Verarbeitung des Modulinhalts beim Reasoningprozess keine Rolle spielt und als Schnittstelle physikalische Grundgrößen definiert wurden kann davon ausgegangen werden, dass jeder mathematisch abbildbare Sachverhalt integriert werden kann. Auch in Bezug auf den LOD einer Untersuchung ist hier, wie bereits erwähnt, keine Einschränkung gegeben.

- **„Standardized Mathematical Models Modularity“**
  Da die Modulkommunikation standardisiert wurde und durch das Datensammlermodul eine Informationsversorgung in der richtigen Art und Weise sichergestellt ist, kann auch diese Art der Modularität bestätigt werden. Da der Modulinhalt frei wählbar ist, ist auch eine „Standardized Mathematical Models Modularity“ gegeben.

- **„Modularity on Codelevel“**
  Eine Modularität auf Codelevel ist mit diesem System nicht zwingend erfüllt. Natürlich bringt diese Modularität viele Vorteile mit sich. In Bezug auf den „Open-World“ Charakter würde das aber eventuell zu einer Ausgrenzung verschiedener Domänen führen, die mit den entstandenen codetechnischen Einschränkungen nicht umgangen werden kann. Die erfolgreiche Einführung der FMI Technologie in der Automobil Branche zeigt jedoch, dass fehlende „Modularity on Codelevel“ hier kein Hindernis darstellt.

Interaktiv und interdisziplinär

Durch das Grundgerüst dieser Methode ist eine Interaktion zwischen Modulen verschiedenster Disziplinen möglich. Berechnungen können mit Funktionalitäten auf verschiedensten Ebenen gekoppelt werden. Eine Abbildung der Interaktion zwischen Gebäude und Nutzer wurde modelliert. Dieses Interaktionskonzept ist ein essenzieller Bestandteil dieser Methode und eröffnet die Möglichkeit, das Gebäude in direkte Interaktion mit dem Nutzer treten zu lassen. Durch die Priorisierung kann eine adäquate Berücksichtigung verschiedenster Stakeholder-Interessen stattfinden. Die generische Beschreibung der Modulkonzipierung ermöglicht es in diese auch Modelle zu implementieren, die die Verkapselung des Quellcodes erlauben. Eine Kompatibilität dieser Modelle wurde bereits am Beispiel des FMI Standards von Mitterhofer (2017) gezeigt. Interdisziplinarität ist somit gegeben.

Offen/Nicht abgeschlossen

Durch die Verwendung ontologiebasierter Algorithmik, die auf dem „Open-World“ Ansatz aufbaut, ist das System offen für jegliche neuen Applikationen. Die Offenheit eines solchen Systems ist ein integraler Ansatz um neue oder fachfremde Forschungsbereiche einzubinden. Da die Ontologiebasis auf Grundlage der instanziierten Klassen arbeitet, ist ein weit übergreifender Charakter des Grundsystems denkbar. Eine Übertragbarkeit auf andere Bereiche ist, soweit eine Kategorisierung in ähnlicher Form erfolgen kann, gegeben. Durch die Einbindungsmöglichkeiten von standardisierten Konstrukten, wie dem FMI-Standard, ist eine barrierefreie Integration anderer Forschungsbereiche gewährleistet.

Dynamische Modellierung

Durch die Möglichkeit der objektorientierten Gestaltung der hier gezeigten Methode ist es möglich, Untersuchungen mit einer gewissen Dynamik auszuführen. Grenzen sind dem Ganzen dort gesetzt, wo die Dynamik durch das regelmäßige Neu-Reasoning zu einer Verlangsamung des gesamten Systems führt. Ansätze, die dies in Zukunft entschärfen könnten, wären ein intelligentes Teil-Reasoning oder ein Live-Reasoning, das auf das Speichern der Verbindungen verzichtet und die Verknüpfungen „on-the-Fly“ dauerhaft ermittelt. Grundsätzlich ist mit der Methode jedoch eine

Dynamik des Wohnumfeldes abbildbar. Durch die wenig reglementierten Inhalte von Modulen und die einzige Beschränkung auf die Schnittstellendefinition steht es den Entwicklern frei, welche Bestandteile sie implementieren. Dadurch ist es auch möglich, dynamische Inhalte in Modulform einfließen zu lassen. Dies wurde am Beispiel der vorhergesagten Duschaktivität bereits gezeigt. Im Simulationskontext hat sich des Öfteren gezeigt, dass stationäre Betrachtungen im Wohnraum nicht zielführend sind, da Einflüsse des Nutzers und der damit unterliegenden Dynamik ein zu geringer Stellenwert beigemessen wird. (Mahdavi, 2011)
Durch die Möglichkeit Teilalgorithmen mittels Reasoning miteinander zu verbinden, können auch Module verschiedener Herkunft miteinander verknüpft werden. Nach dem Reasoningprozess wird ein Modul integraler Bestandteil des Gesamtkonzeptes. Dadurch ist ein integrierter und dynamischer Ansatz bei dieser Methode gegeben.

Nutzerindividuell, selbstlernend, vorhersagend

Wie Mahdavi (2011) bereits festgestellt hat, spielt der Nutzer aufgrund seiner Unberechenbarkeit bei der Simulation von Gebäuden eine bedeutende Rolle. Demzufolge muss ein intelligentes Smarthome Eigenschaften zur Individualisierung ebenso wie die Möglichkeit der automatischen Adaption mit sich bringen. Um bei trägen Vorgängen eine hohe Reaktionsfähigkeit aufzuweisen, ist außerdem ein gewisses Maß an „Modul-Intelligenz“ für Vorhersagen notwendig. Durch die Einbindung von Algorithmen des maschinellen Lernens wurde der Weg für die Individualisierung auf die Nutzerpräferenzen eröffnet. Auch durch die Generierung des DNZ ist eine Individualisierung umsetzbar, dies liegt im Verantwortungsbereich des jeweiligen Modulentwicklers.
Vorhersagen sind durch die vielfältigen Verwendungsmöglichkeiten von Algorithmen des maschinellen Lernens möglich. Durch das Lernen und Speichern relevanter Nutzerparameter kann somit der gesamten Berechnungsablauf auf die Bedürfnisse des jeweiligen Nutzers angepasst werden. Eine weitere Form des Lernens, das sogenannte Feedbacklernen, wird nachfolgend beschrieben. Damit ist es möglich, noch genauer auf die Bedürfnisse des Nutzers einzugehen.

Nutzerabhängige Kontextsensitivität

Wie im vorherigen Kapitel bereits erwähnt, ist die Individualisierung ein essenzieller Teil dieser Methode. Daher ist eine Umsetzung ohne kontextsensitive Komponenten nicht denkbar. Um individuell auf die Benutzerbedürfnisse, die vom DNZ repräsentiert werden, eingehen zu können ist eine Reaktion auf dessen Handlungen und dessen Handlungskontext unerlässlich. Durch die Möglichkeit berechnungstriggernde Momente modulunabhängig zu wählen und durch die digitale Abbildung des Nutzers, kann Kontextsensitivität entsprechend dem jeweils gewünschten Level ermöglicht werden. Die Möglichkeiten der Ausprägung werden hier hauptsächlich durch die Art der verbauten Sensorik und der zugehörigen Module limitiert. Eine beispielhafte Umsetzung kontextsensitiver und nutzerindividueller Steuerungsmöglichkeiten wurde gezeigt. Damit ist bewiesen, dass die Methode Kontextsensitivität unterstützt und Smarthomes kontextsensitiv agieren können.

7.2 Aktuelle Limitationen

Die hier angewendete Methodik unterliegt hauptsächlich der Einschränkung, dass die Ein- und Ausgangsströme der Module korrekt definiert werden müssen.

7.2.1 Thematische Limitationen

Weiterhin kann keine Aussage dazu getroffen werden, inwieweit die Methode einen brauchbaren Ansatz darstellt, um Themengebiete abzubilden, deren Problemcharakter sich nicht auf einheitliche physikalische Basisgrößen zurückführen lässt. Gerade in Bereichen, die nicht mathematisch eindeutig abbildbar sind, wie beispielsweise der Psychologie, kann die Methode an ihre Grenzen stoßen. Vor allem bei dem Versuch zwischenmenschliche Beziehungen mithilfe zweier DNZ zu beschreiben ist fraglich, ob sich die auszutauschenden Parameter so standardisiert beschreiben lassen, wie bei anderen Problemen. Auch ob die Problembeschreibung in zwei getrennten DNZ erfolgen kann, bleibt offen. Das System der digitalen Zwillinge wurde in Bezug auf Simulationsansätze aus der Bauphysik erstellt und ist daher hauptsächlich für diese geeignet. Da die Auflösung in Form der Zeitschritte bei der Berechnung von Simulationsmodellen stark vom LOD der jeweiligen integrierten Module abhängig ist, ist dies eines der Hauptprobleme bei Simulationen mit verschiedenen Modulen aus verschiedenen Domänen. Einerseits resultiert hieraus, dass verschiedene Ausgangsdaten und Eingangsdaten fusioniert werden müssen und dafür die mathematisch und physikalisch richtigen Methoden gewählt werden müssen. Andererseits muss, für eine brauchbare Berechnung, die Auswirkung auf das Ergebnis dieser Module klar sein. Einen Ansatz, wie mit dieser Problematik umzugehen ist, zeigte Mitterhofer (2017) bereits für die BPS, deren Übertragbarkeit auf diese Methode gegeben ist. Für eine solche Verarbeitung ist jedoch vonseiten des Modulentwicklers viel Entwicklungsaufwand in das Testen und Beschreiben seines Moduls unter verschiedenen Bedingungen aufzuwenden. Eine Methodik, wie diese Entwicklungen stattzufinden haben, ist jedoch noch nicht verfügbar.

Für die Problematik der „Many-to-One“ Verbindungen wurde jedoch durch diese Arbeit eine funktionsfähige Lösung gefunden. Daher ist als nächster Schritt auf das Zeit- und Auflösungsproblem von Modulen mit unterschiedlichem LOD ein Fokus zu setzen, wenngleich für eine erste Anwendung der Methode eine hinreichende Lösung erstellt wurde.

7.2.2 Technische Limitationen

Weiterhin sind auf technischer Ebene die Limitationen der jeweiligen Module ausschlaggebend. Derzeit sind durch Standardisierungsprobleme und Begrenzungen hinsichtlich der verwendeten Softwaresysteme, vor allem bei BPS, nicht alle Variablen austauschbar oder auf Grund von nicht unterstützten Standards keine vollständige Co-Simulation möglich (Mitterhofer, 2017, S. 112). Dies beschränkt die Möglichkeiten der Methodik. Da ein Modul in die Gesamtmethodik eingebettet und damit Bestandteil des Gesamtalgorithmus wird, hat die Limitation eines Moduls voraussichtlich auch Auswirkungen auf Module anderer Ersteller. Durch das Schaffen eines digitalen Gesamtsystems, wie in dieser Methode dargestellt, ist davon auszugehen, dass Defekte an einzelnen Komponenten im schlimmsten Fall Auswirkungen auf das gesamte System haben. In diesem Fall kann jedoch Abhilfe geschaffen werden, indem die defekten Teile deaktiviert werden und stattdessen die mitgelieferten Ersatzdaten Verwendung finden.

Die Einbindung jeglicher Software in Form von Modulen ist möglich, somit sind aus technischer Sicht kaum Limitationen gesetzt. Ein Überführen bestehender Algorithmen in die gezeigte Methodik ist zwar nicht automatisiert abbildbar, da hierzu zu viel Expertenwissen an verschiedenen Stellen nötig ist. Grundsätzlich stellt die Überführung aber einen deutlich geringeren Zeitaufwand als eine Neuentwicklung dar.

Die Berücksichtigung der Rechenzeiten spielt in der hier vorgestellten Methode eine essenzielle Rolle. Da die Verschachtelung der Module, wie bereits erläutert, zu der Bildung eines Gesamtsystems führt, ist damit eine Abhängigkeit aller Module untereinander gegeben. Dadurch wirken sich lange Berechnungszeiten bei einzelnen Modulen merklich auf die Effektivität des Gesamtsystems aus. Bei der Modulentwicklung und der Implementierung muss daher sichergestellt werden, dass die verbaute Hardware die Anforderungen an die Berechnungen erfüllen kann. Andernfalls existieren aber auch noch weitere Möglichkeiten um Rechenlast auszulagern. Cloudbasierte Lösungen könnten diese Probleme lösen. Hier muss jedoch der Datenschutzproblematik Rechnung getragen werden, da sensible private Daten an Dritte gegeben werden. Diese Aspekte wurden in dieser Arbeit jedoch weitestgehend ausgespart, da sie hier von der erforderlichen Rechenlast oder einer möglichen Auslagerung derselben abhängen.

Ein weiterer Weg um Berechnungen zu beschleunigen wäre das Anlegen einer Berechnungshistorie. Da vor allem kleine Hilfsmodule, aber auch komplexere Module, verschiedene Berechnungen mit gleichen oder ähnlichen Randbedingungen mehrfach durchführen, wäre das Speichern aller durchgeführten Berechnungen mit deren Ein- und Ausgangsvariablen eine Möglichkeit der Beschleunigung. Sollten gleiche Eingangsvariablen auftreten, ist bei einfachen Problemen auch zu erwarten, dass die gleichen Ausgaben erfolgen. In diesem Fall können, vor allem bei Berechnungen langer Zeitketten, Ressourcen gespart werden, da die Daten nur aus gespeicherten Werten einer Datenbank ausgelesen werden müssen. Dieser Ansatz könnte durch die intelligente Nutzung von Hardwareressourcen zusätzlich unterstützt werden. Vor allem in Zeiten ohne Anwesenheit des Nutzers oder während der Nacht ist davon auszugehen, dass die Menge, der vom Nutzer verursachten Trigger-Events stark zurückgeht. Ein auf die Anwesenheit des Nutzers ausgelegtes System hat daher in dieser Zeit keine vollständige Auslastung zu verzeichnen. In dieser Zeit wäre eine Berechnung verschiedener Kombinationen aus virtuellen Eingangsdaten bei Modulen oder Modulkombinationen denkbar. Die so berechneten Daten könnten, wie bereits geschildert, gespeichert und später wieder aufgerufen werden. Vor allem das langwierige bearbeiten von Lernprozeduren wäre in dieser Zeit sinnvoll. Auch diese Aspekte wurden, da sie kaum Einfluss auf die grundlegende Funktionalität der Methodik haben, in dieser Arbeit nicht weiter betrachtet.

8 Fazit und Ausblick

Abschließend werden die wesentlichen Ergebnisse dieser Arbeit zusammengefasst, um einen Überblick über die Funktionalitäten und die Möglichkeiten des hier entwickelten Systems aufzuzeigen. Im Anschluss daran werden auch die vielfältigen Möglichkeiten für Weiterentwicklungen näher erläutert.

8.1 Fazit

Die vorgestellte Methode erhebt den Anspruch, dem derzeit durch Plattformen wie IFTTT (IFTTT Inc., 2019) geprägten Automationsansatz von einfachen Abhängigkeiten zwischen Events und Schaltaktionen, durch einen weit komplexeren und realitätsnäheren Ansatz zu ersetzen. Gewählt wurde zu diesem Zweck eine Methode, die Interaktion zwischen Nutzer und Gebäude abbildet. Um dies zu erreichen wurden sowohl für das Gebäude als auch für den Nutzer jeweils digitale Zwillinge implementiert, welche als digitale Repräsentationen des jeweiligen realen Pendants agieren. Durch die darauf basierende Teilung der verankerten Automationsansätze in verschiedene Module, konnte eine klare Aufteilung bauphysikalischer Prinzipien auf die jeweiligen digitalen Zwillinge umgesetzt werden. Unterschieden wird zu diesem Zweck neben der Einteilung in Nutzer- und Gebäudemodule in fünf grundsätzliche Typen von Modulen: Funktions-, Hilfs-, Wirk- und Datensammlermodule sowie solche des maschinellen Lernens und der Vorhersage. Während Funktionsmodule die grundsätzlichen physikalischen und mathematischen Funktionen des jeweiligen Zwillings beschreiben und den Kernalgorithmus der klassischen Funktionen im Smarthome darstellen, bilden Hilfsmodule mathematische und physikalische Zusammenhänge ab, die keiner Funktionalität zugeordnet werden, aber für Berechnungen wichtige Daten erzeugen. Das Konzept der digitalen Zwillinge und ihrer Module basiert darauf, dass von einer Seite, meist dem Nutzer, eine Anforderung in Form eines Bedürfnisses gestellt wird und diese vom anderen digitalen Zwilling zu erfüllen ist. Um eine eingriffsfreie und automatische Zuordnung von Eingangsgrößen und vorhandenen Mitteln in Form von Aktuatorik zu gewährleisten, wurde eine ontologiebasierte Algorithmik verwendet.
Damit können aufgrund des vorhandenen Wissens über die verschiedenen Einzelteile des Systems, selbstständig Verbindungen erzeugt werden. Zudem können individuelle Berechnungssetups für das jeweilige Gebäude erstellt werden sowie Berechnungen ohne vorheriges Eingreifen des Nutzers oder eines Experten durchgeführt werden. Dabei wird durch die vom Modulentwickler mitgelieferten Ersatzdaten sichergestellt, dass auch beim Nichtvorhandensein von verschiedenen Sensordaten eine Berechnung durchgeführt werden kann. Nach der Berechnung aller für das Gebäude relevanten Interaktionen werden von den Funktionsmodulen, auf Basis grundlegender Wirkgrößen Anforderungen in Form eines zeitbasierten Arrays ausgegeben.

Um den unterschiedlichen Gewichtungen von Abläufen im Gebäude Rechnung zu tragen werden diese, zusätzlich zu einer Information über ihre Ungenauigkeit, mit einer Angabe über ihre Priorität versehen. Dadurch ist es möglich, in einem nächsten Schritt die Handlungsvorschläge in eine prioritätenbereinigte Form zu überführen. Diese wird in zwei weiteren Schritten um alle Einträge bereinigt, deren Steuerung aus technischer Sicht nicht umsetzbar ist oder die in der Zukunft zu einem Dilemma führen würden. Um einen Informationsverlust in diesem Prozess auszuschließen wird nun geprüft, ob von jedem Modul noch zumindest eine Version der Handlungsanforderungen vorhanden ist. Sollte dies nicht der Fall sein, ist der Nutzer oder ein Experte mit einzubeziehen.

Für den Fall des Verbleibens mehrerer Versionen wird diejenige gewählt, die auf das Gesamtsystem den geringsten Einfluss hat oder vom Modulentwickler als präferiert angegeben wurde. Über das Wirkmodul kann nun eine Zuordnung der zu den Wirkgrößen passenden Geräten und deren benötigten Steueraktionen erfolgen. Damit ist ein vollständiger Ablauf, vom Erzeugen eines Bedürfnisses und Befriedigen desselben im Automationsalgorithmus abgebildet.

Die Berechnung und vor allem die Ermittlung verschiedener Parameter schließt auch die Möglichkeit von Algorithmen des maschinellen Lernens in den dafür vorgesehenen Modulen ein. Diese stellen, wie in den Fallstudien gezeigt wurde, eine signifikante Verbesserung der Berechnungsgenauigkeit in Bezug auf eine Individualisierung dar. Diese Methode führt somit dazu, dass ein Smarthome die folgend aufgelisteten Eigenschaften erfüllen kann.

- Nutzerindividuell, selbstlernend, vorhersagend
- Interaktiv und interdisziplinär
- Dynamisch
- Modular und skalierbar
- Offen und nicht geschlossen
- Kontextsensitiv und nutzerunabhängig

So wird es den individuellen Bedürfnissen des Menschen in seinem privaten Umfeld eher gerecht, als die derzeit üblichen funktionsbasierten Automationsansätze mit „Wenn-Dann Regeln“, wie das Beispiel von IFTTT (IFTTT Inc., 2019) zeigt. Diese lassen im Rahmen vorgefertigter Module nur wenig Spielraum für Individualisierung und Interaktion. Die hier gezeigte Methode schließt damit bisher unbearbeitete Lücken im Bereich der Wohnraumautomation. Durch die Möglichkeit, bauphysikalische Funktionen in das Smarthome zu integrieren und deren vielfältige Interaktionen abzubilden, wurde eine Plattform geschaffen, um die Steuerung des Wohnraums besser auf die Bedürfnisse des Menschen anzupassen. Damit rückt die Zukunftsvision des automatisierten Wohnraums in greifbare Nähe.

8.2 Ausblick

Die gezeigte Methode ist durch ihren offenen Charakter so konzipiert, dass noch viele andere Umsetzungsmöglichkeiten gegeben sind. Vor allem der ontologiebasierte Charakter ermöglicht das maschinengestützte Finden von Zusammenhängen. Diese sind in der hier beschriebenen Methodik auf mathematische oder physikalische Zusammenhänge, vor allem aus der Bauphysik, beschränkt. In einem erweiterten Ansatz wäre aber auch das Finden von anderen Zusammenhängen aus anderen Bereichen des menschlichen Lebens denkbar. Eine Erweiterung des Algorithmus müsste hier dann lediglich in Teilbereichen erfolgen. Vollständige digitale Abbildungen der Nutzer und damit auch der Psychologie wären beispielsweise denkbar. Durch die Integration maschinellen Lernens und einer großen Basis personenbezogener Daten können so Zusammenhänge erschlossen werden, die mit herkömmlichen wissenschaftlichen Methoden nicht auffindbar wären. In diesem Zusammenhang wäre auch eine Einbindung von Daten aus sozialen Medien denkbar.

Während sich die Planung eines Gebäudes derzeit noch hauptsächlich auf die Anordnung physikalischer Instanzen im Gebäude beschränkt, wird im Laufe der Zeit die Digitalisierung auch hier Einzug halten. Daher ist es denkbar, dass nicht nur die physikalische, sondern auch die digitale Konzipierung eines Gebäudes in naher Zukunft eine große Rolle spielt. Als die Computertechnologie an einem vergleichbaren Punkt der Entwicklung stand, brachten Betriebssysteme den Durchbruch, für eine schnelle und solide Funktionsentwicklung. Daher liegt die Vermutung nahe, dass im Gebäude der Zukunft solch ein System denselben Nutzen bringen könnte.

8.2.1 Integration von Simulationsmodellen

Eine entscheidende Rolle könnte dabei die Integration von Simulationsmodellen spielen. Hier sollten zukünftig vor allem die bereits angesprochenen Interdomän-Ansätze verfolgt werden, um effizientere und bessere Möglichkeiten zu finden und die verschiedenen Modelle zu beschreiben. Durch die

Technologie und Infrastruktur die das „Internet of Things“ vernetzt, wäre so eine Vernetzung zwischen verschiedenen Domänen möglich. Eine Basis dafür könnte ebenso diese Methode liefern.

Um die Möglichkeit einer mehrzonalen Berechnung auszuschöpfen, müssen zugehörige Berechnungen auch in der Ontologie Definitionen erhalten, in welchen Zonen oder Gebäudeteilen Berechnungen stattfinden sollen. Bei Berechnungen mit einem hohen LOD würde die Kalkulation aller Module in allen Räumen nur zu unnötiger Rechenlast führen und das System somit verlangsamen. Da die Ontologie alle Möglichkeiten abbildet, um mehrzonale Zuordnungen der Variablen zu erstellen, muss in einem nächsten Schritt ein Framework gefunden werden, in dem diese Zuordnung vollzogen werden kann. Grundsätzlich bieten sich hierfür mehrere Möglichkeiten an:

- Der Nutzer nimmt die Zuordnung vor. Dafür benötigt er spezielles Wissen über die Sinnhaftigkeit der Funktionen.
- Alternativ kann die Zuordnung auch von einem Experten vorgenommen werden. Entweder bei der Einrichtung des Systems oder durch eine Standardisierung der Raumbezeichnungen und Zuordnungen im Vorhinein.

Auch mehrere Nutzer in einem Gebäude führen zwangsläufig zu der Notwendigkeit Module mehrfach zu instanziieren. Aus diesem Grund sollte dieser Thematik Rechnung getragen werden. Um bei mehreren Nutzern im Gebäude eine Individualisierung sicherstellen zu können, sind Methoden des maschinengestützten Lernens näher zu betrachten.

8.2.2 Maschinengestütztes Lernen

Durch „Deeplearning“-Algorithmen können viele Zusammenhänge erkannt werden, die auf den ersten Blick nicht offensichtlich sind. Eine Möglichkeit Daten in Beziehung zu setzen ist es, Aktionen mit anderen Daten zu clustern. Damit ist die Handlung des Menschen und sein Einfluss auf die Daten abbildbar. Da Menschen vor allem aufgrund ihrer kulturellen Diversität oft vollkommen unterschiedliche Aktionen im Wohnraum ausführen und aufgrund der steigenden technischen Möglichkeiten auch immer neue Aktionen entwickeln, empfiehlt sich die Entwicklung eines Algorithmus, der aufgrund der gesammelten Daten auch neue Aktionen erkennen kann. Damit kann die Individualisierung der Wohnraumautomation vorangetrieben werden.
Auch die Problematik bei einem Umzug eines Nutzers in ein anderes Gebäude wurde bereits erwähnt. Hier muss die Möglichkeit bestehen, die über den Nutzer erfassten Daten im Nachhinein auf den neuen Wohnkontext untersuchen zu können. Nicht immer können dabei alle Informationen übertragen werden. Ein vollständiger Übertrag der Informationen in ein neues Gebäude wäre aber auch nicht sinnvoll. Vor allem Daten, die die Interaktion des Gebäudes mit dem Nutzer beschreiben, basieren auf einer Analyse in einem anderen Umfeld. Der Übertrag der nutzerrelevanten Rohdaten kann jedoch, bei einer nachträglichen Analyse auf potenzielle Einflüsse auf das neue Gebäude, durchaus einen Mehrwert erbringen. Dies konnte bereits am Beispiel der Ermittlung und Vorhersage der Duschhäufigkeit gezeigt werden. Durch die gebäudeunabhängige Form der Erkennung kann die Häufigkeit dieser Ereignisse in einem neuen Gebäude, einer neuen Bewertung des zukünftigen Luftfeuchteverlaufes zugeführt werden. Letztendlich geht mit dem Umzug in ein neues Wohnumfeld aber grundsätzlich auch oft ein neues Nutzerverhalten einher.
Darauf bezogene zusätzliche Funktionalitäten, die vor allem der Individualisierung dienen können, sind die Möglichkeiten des Feedbacklernens von Algorithmen des maschinellen Lernens. Damit könnten bestimmte Nutzerparameter durch das Feedback der Nutzer genauer bestimmt werden. Spezielle Eigenschaften, wie die individuellen Präferenzen der Nutzer, in Bezug auf die Lichtstärke oder Lichtfarbe, können durch Feedbacklernen ermittelt werden. Die Algorithmik muss dazu beim aktiven Eingreifen der Nutzer in die vom System als ideal berechneten Umgebungen schließen, dass die Randbedingungen nicht ideal eingestellt waren. Die Datengrundlagen müssen daraufhin in Korrelation mit den neuen, von den Nutzern eingestellten, Situationen gebracht werden. Vor allem die Bereiche der Behaglichkeitsbeschreibungen könnten dadurch optimiert werden. Durch dieses „In-

Bezug-Setzen“ sind vor allem die durch Ontologien zur Verfügung gestellten Wissensdaten besonders wertvoll.

8.2.3 Möglichkeiten von ontologiegestützten Algorithmen

Die Verwendung der OWL ermöglicht auch eine enge Verknüpfung des Gebäudes mit dem Internet. Auf dieser Grundlage könnten beispielsweise Produktdaten eigenständig vom Gebäude aus dem Internet bezogen und verarbeitet werden. Durch die Anbindung des Internets als Informationsquelle potenzieren sich so die Anwendungsszenarien der Algorithmik um ein Vielfaches. Die für diesen Prozess verwendeten Reasoner und deren Regelwerke sind jedoch erst noch auszuwählen oder zu erstellen.
Wie bereits erläutert, wurden die hier gewählten Reasoner mit Blick auf geeignete Integrierbarkeit in das Testframework ausgewählt. Für eine Weiterentwicklung der Methode sollte gezielt Wert auf die Wahl und Anbindung eines Reasoners gelegt werden, welcher nicht nur alle hier gezeigten Funktionen unterstützt, sondern zusätzlich eine ausreichend hohe Performance oder vielleicht sogar ein Live-Reasoning der Verbindungen aufweist. Damit wäre im Rahmen der fortschreitenden Digitalisierung auch eine Repräsentation von anderen Gegenständen im Wohnraum, in Form von digitalen Zwillingen, denkbar. Durch ein Live-Reasoning könnte so in Echtzeit auf die Veränderungen im Raum reagiert werden.

8.2.4 Optimierung der In- und Outputdatenströme

Auch die Integrierbarkeit weiterer Elemente durch die Einbindung von BIM-Informationen kann so zu einer erweiterten Funktionalität führen. Durch die stets voranschreitende Vernetzung von Geräten werden die Möglichkeiten auf der Aktionsseite der Automationsserver größer und vielfältiger. Damit steigen automatisch auch die digitalen Interaktionsmöglichkeiten des Gebäudezwillings mit dem Nutzerzwilling. Diese beiden Parameter erzeugen eine stetig steigende Vielfalt aufseiten des Smarthome. Da Informationen durch das richtige Integrieren in die hier gezeigte Methode automatisch erfasst und verarbeitet werden können, ist die Methode auch in der Zukunft offen für Veränderungen.

Die Entwicklung und Integration von Modulen ist bereits beim Anlegen eines BIM-Models, also in der Planung eines Gebäudes, denkbar. So besteht die Möglichkeit, die für ein Gebäude notwendigen Module bereits im Planungsprozess zu bestimmen. Auf diese Weise könnte bereits im Voraus ermittelt werden, welche Sensorik und Aktuatorik bei geplanten Gebäuden benötigt und verbaut werden sollte, um so einen Grundumfang an Funktionalität gewährleisten zu können. Auch eine Bestimmung vorhandener Modulin- und outputs wäre damit bereits im Planungsprozess möglich und könnte zu einer Erleichterung der Systemimplementierung führen. Es bleibt jedoch die Frage zu klären, wie dem System Änderungen am und im Gebäude mitgeteilt werden. Ein intelligentes System würde diese selbst erkennen und die Änderungen durchführen. Der aktuelle Stand der Methode lässt bisher jedoch nur ein händisches Ändern dieser Daten zu. Dahingehend stellt sich die Frage, wie dieser Prozess zum Beispiel mit bildgebenden Verfahren oder Sensorik optimiert werden kann. Weiterhin ist eine Abgrenzung der Objekte zu treffen. Sollten sehr kleine bewegliche Objekte, wie Smartphones, mit referenziert werden, so ergibt sich sehr schnell eine hohe Zahl an nötigen Neu-Reasonings, da diese regelmäßig in der Wohnung bewegt werden. Dies könnte das System träge und fehleranfällig machen. Auch die zu verarbeitenden Moduloutputs und deren Varianten können dieses System verlangsamen oder bei einer restriktiven Definition der Moduloutputs zu häufigen Dilemmata führen. Diesen Outputdatenströmen sollte daher besondere Aufmerksamkeit zuteilwerden.

In der derzeit vorliegenden Version der Methodik werden Moduloutputs nur getrennt voneinander betrachtet. Die Systematik ist jedoch so offen gestaltet, dass durch iterative Prozesse und Optimierungsprozesse in einem zusätzlichen Schritt, bei der Erstellung der Anforderungsmatrix, auch

neue Outputs mit einer kombinierten Anforderung denkbar wären. Um diese im sinnvollen Maße anbieten zu können, ist die Implementierung von Optimierungsalgorithmen unumgänglich. Eine beispielhafte Anwendung eines domänenübergreifenden Optimierungsframeworks im Bereich der Simulation kann bei Schmidt (2016) gefunden werden. Die Implementierung dieser Methodik würde die Auswahl einer jeweils optimalen Regelungsstrategie sicherstellen und somit einen großen Mehrwert bei der Effizienz des Gebäudes und seiner Schaltvorgänge erbringen.

Ein weiterer Aspekt, der am Beispiel des Szenario 2 der zweiten Fallstudie sichtbar wird, ist die Möglichkeit Wirkgrößen variierend zu beschreiben. Durch die Unterscheidung in Prioritäten, Zielwerte und generelle Anforderungen über Operatoren ist es so möglich, Wirkungen in verschiedener Art zu berücksichtigen. Bei der Beheizung eines Raumes wurde beispielsweise eine Zielgröße ermittelt, ohne dass diese bei der Prioritätenverarbeitung Verwendung fand, da über die Operatoren keine Forderung erstellt wurde. Das bietet für spätere Entwicklungen die Möglichkeit, Aussagen über eine positive Raumbedingung zu machen, ohne eine Steuerung zu initiieren. So können beispielsweise durch Optimierungsalgorithmen noch deutlich besser abgestimmte Anforderungen generiert werden. Damit wäre man dann nicht nur in der Lage die aktuell notwendigen Aktionen zu berücksichtigen, sondern gleichzeitig auch positive Nebeneffekte in Bezug auf einzelne Sachverhalte mit einzubeziehen.

Auch die automatische Auswahl der für Module verwendeten Variablen ermöglicht es, die Genauigkeit der Algorithmen, von dem Anbieten von Grundfunktionen bis hin zu einer komplexen detaillierten Simulation, zu variieren. Obwohl dafür noch ein System notwendig ist, dass die Unterteilung der Module und der Daten entsprechend ihres Genauigkeitsgrades ermöglicht, ist so eine automatische Zuordnung und ein dauerhaftes Anstreben der Rechenvarianten mit der höchsten Genauigkeit gegeben. Damit ist in Zukunft ein automatischer Ablauf dieser genauigkeitsgesteuerten Zuordnung denkbar. Aus Sicht des Verfassers sollte ein System zu diesem Zweck die Anzahl der aus den jeweiligen Bereichen der digitalen Zwillinge eingehende Datenströmen erfassen und mit Hilfe einer Sensitivitätsanalyse untersuchen. Je nach Sensitivität ist deren Einfluss auf das Ergebnis dann zu gewichten. Daher würde eine numerische Skala zur Einordnung der Position oder ein System mit Punkten für die Einteilung der Genauigkeit von Modulen sowie Ein- und Ausgangsdaten Sinn ergeben.

Um eine der jeweiligen Problemkomplexität entsprechende Auswahl für berechnungstriggernde Intervalle bei Sensorwerten zu bestimmen, ist eine genauere Untersuchung des gesamten durch die Ontologie erstellten Berechnungssystems notwendig. Eine Methode hierfür wäre die Implementierung eines Frameworks, das selbstständig für verschiedene Funktionsmodule Sensitivitätsanalysen erstellt. Basierend auf den so erhobenen Daten können sinnvolle Eventintervalle für eine Triggerung erstellt werden.

Eine weitere Verbesserung der Performance des Algorithmus ergibt sich durch die Integration von Algorithmen zur Bestimmung von Routen, mit denen die beste Reihenfolge zur Berechnung des Rechensetups effizienter bestimmt werden kann. Auch die spätere Möglichkeit, Moduloutputs mit Hilfe von Optimierungsalgorithmen zu mischen und durch das maschinelle Lernen zu individualisieren, sollte einen signifikanten Beitrag dazu leisten, Steuerungen im Smarthome intelligenter, interaktiver, nutzerindividueller und dynamischer zu gestalten und so auf den Nutzerkontext einzugehen.

8.2.5 Hardwareentwicklungen

Verbesserungen können auch in Bezug auf die eingesetzte Hardware erzielt werden. Beispielsweise wurde für die Radonkonzentration eine Messfrequenz von einer Minute verwendet. Übliche, in Smarthomes verwendbare Radonsensoren können jedoch nicht mit so hohen Taktraten angesprochen werden. Einige Geräte erlauben eine Messfrequenz von fünf Minuten, befinden sich jedoch in einem, für den Smarthome-Bereich, hohen Preissegment. Mit der Hilfe industrieller Sensorik ist man hingegen in der Lage, Messwerte im Minutenintervall zu erzeugen. Diese Geräte befinden sich dann jedoch in einem relativ hohen Preisbereich und scheiden, auch wegen ihrer Größe, für den Einsatz im Wohnbereich aus. Ebenso ist, mangels geeigneter Schnittstellen, nicht ohne Weiteres eine Anbindbarkeit an ein Smarthome gegeben. Im Bereich der Radonsensorik muss demnach bis zu einer produktiven Realisierung noch eine entsprechende Entwicklung stattfinden. Des Weiteren scheint eher eine Kombination mit Algorithmen des maschinellen Lernens hierbei vielversprechend. So könnte der Mangel im Bereich der Ansprechgeschwindigkeit durch Vorhersagen aus einem solchen Algorithmus kompensiert werden.

Ein weiteres vielversprechendes Potenzial stellt die Integration der möglichen Nutzeraktionen in eine Steuerung dar. So könnte der Nutzer situationsabhängig durch gezielte Sprachausgaben zu einzelnen Handlungen aufgefordert werden. Am Beispiel des Duschereignisses lässt sich vermuten, dass vor allem im Winter ein Abführen der hohen Luftfeuchte ausschließlich über maschinelle Lüftung deutlich länger dauert, als im Vergleich zur manuellen Fensterlüftung. Um dies zu berücksichtigen müssen dem Wirkmodul für die Steuerung des Luftwechsels die Fensterpositionen über die Ontologie übergeben werden. Außerdem ist eine Algorithmik zur Berechnung der möglichen Luftwechsel zu implementieren. So könnte situationsbezogen die beste Strategie zur Abfuhr der hohen auftretenden Luftfeuchten ermittelt werden. Der Nutzer kann daraufhin über Sprachausgaben zur Umsetzung angeleitet werden. Das rein maschinelle Lüften kann in diesem Szenario als Rückfallebene jedoch weiterhin genutzt werden.

Zusätzlich zeigt sich, dass viele Messgrößen zur Beschreibung bauphysikalischer Probleme derzeit im Smarthome noch nicht verfügbar sind. So wären für eine sichere Überprüfung, in Bezug auf die Gefahr einer Schimmelbildung, Oberflächentemperaturen zu messen und diese auf spezifische Werte für die Berücksichtigung von Wärmebrücken zu erweitern. Für viele bestehende Gebäude fehlen jedoch die nötigen detaillierten Informationen zur Bestimmung der dafür notwendigen Parameter. Hier werden durch den Einsatz von Modulen aus dem Bereich der Ersatzdaten Möglichkeiten geschaffen, genauere und sicherere Berechnungen zu erzeugen. So könnte entsprechend der Baualtersklasse des Gebäudes eine Abschätzung zu dessen Bausubstanz durchgeführt werden, um die nötigen Wärmebrücken in der Kalkulation der relevanten Oberflächentemperatur berücksichtigen zu können.

Außerdem kann über den generellen Rechenzeitenbedarf der Methodik derzeit aufgrund der nicht vorhandenen Kenntnis der Rechendauer der einzelnen Module, keine exakte Aussage getroffen werden. Um hier für das Gesamtsystem eine Bewertung erzeugen und damit feststellen zu können, ob die Hardware ausreichend dimensioniert ist, ist eine Methodik zu schaffen. Diese muss anhand der einzelnen Module und deren Bewertung des Entwicklers feststellen können, welche Laufzeiten das Ökosystem in verschiedenen Situationen erzeugt. Auch sind Algorithmen zu erforschen, die für triggernde Events einzelne Cluster an miteinander verbundenen Modulen finden. Somit muss nicht, wie derzeit bei einem triggernden Event, die Summe aller Module neu berechnet werden, sondern nur der als zusammenhängend erkannte Cluster an Modulen, der sich automatisch beim Prozess des Reasoning bilden würde. Dieser Prozess kann vor allem bei großen komplexen Systemen von Modulen eine enorme Ersparnis bringen. Auf die Bestimmung der durch Modulkonstellationen entstehenden Rechenzeiten wurde aber, wie bereits erläutert, in dieser Arbeit verzichtet. Ein System für solch eine Bewertung muss daher erst erstellt werden.

8.2.6 Ausblick – Zusammenfassung und Bewertung

Somit ist, nach erfolgreicher Entwicklung einer Decomposition-Strategie, eine vollständige Integration von multidisziplinären Sachverhalten möglich und aufgrund der ontologiebasierten Beschreibung die Modellierung des DGZ und des DNZ im Sinne eines Smarthomes realitätsnah umsetzbar.

Die Gestaltung eines Frontends, worüber die Kommunikation zwischen Nutzer und Gebäude hergestellt wird, würde von den Informationen aus den anderen Disziplinen einen großen Nutzen ziehen. So wäre eine vollständige, automatische Individualisierung des Frontends an den jeweiligen Nutzer denkbar, um diesen bei den wenigen Aufgaben, die er am Gebäude noch selbst steuern muss, zu unterstützen.

Bei cloudbasierten Ansätzen und einem gebäude- und nutzerübergreifenden Ansatz könnten aufgrund der Datenbasis weitere Dienste und leichter zu individualisierende Systeme geschaffen werden. Das übergeordnete Ziel ist hierbei stets, den automatisierten Wohnraum so individuell und persönlich wie möglich zu gestalten. Somit können dem Nutzer viele der alltäglichen Arbeiten abgenommen werden, um ihm ein erholsames Lebensgefühl in seinem eigenen Zuhause zu ermöglichen.

9 Literaturverzeichnis

Altfelix, S. (2017). *Entwicklung und Implementierung einer Methode zur Identifikation von raumluftfeuchterelevanten Nutzeraktionen unter Anwendung von Machine Learning*. Unpublished Master's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Anderlik, L. (2015). *Entwicklung eines Bewertungsschemas zur Bestimmung der Nutzerfreundlichkeit von Hausautomatisierungslösungen im Wohngebäudebereich am Beispiel ausgewählter Systeme*. Unpublished Bachelor's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Anderlik, L. (2017). *Entwicklung und Implementierung eines Algorithmus zur Vorhersage der Schimmelpilzbildung im Rahmen der Hausautomation.* (Master's thesis), Technische Universität München, MediaTUM. Retrieved from https://mediatum.ub.tum.de/node?id=1487887.

Baun, C. (2018). *Computernetze kompakt*. Berlin: Springer Verlag GmbH Deutschland.

Bayerisches Landesamt für Umwelt. (2018). *UmweltWissen – Strahlung: Radon in Gebäuden*. Augsburg: Bayerisches Landesamt für Umwelt (LfU) Retrieved from https://www.lfu.bayern.de/buerger/doc/uw_57_radon.pdf on 26.05.2019

Binsfeld, C. (2017). *Thermische Behaglichkeit im automatisierten Wohnraum – Bestimmung einer behaglichen Raumlufttemperatur in Abhängigkeit der individuellen Bekleidung und Aktivität*. Unpublished Master's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Black Duck Software Inc. (2018). Discover, Track and Compare Open Source Licensed under Creative Commons Attribution 3.0 Unported License (CC BY 3.0). Retrieved from https://www.openhub.net/ on 23.11.2018

Brenon, A., Portet, F., & Vacher, M. (2018). *Context Feature Learning through Deep Learning for Adaptive Context-Aware Decision Making in the Home*. Paper presented at the 2018 14th International Conference on Intelligent Environments (IE), Rom, Italien. doi:10.1109/ie.2018.00013

Bundesamt für Strahlenschutz. (2018, November 21). Radon in Gebäuden. Webpage. Retrieved from http://www.bfs.de/DE/themen/ion/umwelt/radon/vorkommen/gebauede.html on 08.05.2018

Bundesarbeitsgemeinschaft Wohnungslosenhilfe e. V. (2017). Pressemitteilung: BAG Wohnungslosenhilfe: 860.000 Menschen in 2016 ohne Wohnung Prognose: 1,2 Millionen Wohnungslose bis 2018 [Press release]. Retrieved from https://bagw.de/de/themen/zahl_der_wohnungslosen/261.html on 26.05.2019

Bundesministerium für Umwelt Naturschutz Bau- und Reaktorsicherheit (BMUB). (2017). *Entwurf eines Gesetzes zur Neuordnung des Rechts zum Schutz vor der schädlichen Wirkung ionisierender Strahlung.* Retrieved from https://www.bmu.de/fileadmin/Daten_BMU/Download_PDF/Strahlenschutz/strahlenschutzgesetz_entwurf_bf.pdf on 26.05.2019

Burton, L. (Writer). (1999). Smart House. In Alan Sacks Productions (Producer): Disney Channel.

Busch-Geertsema, V. (2018, June 15). Wohnungslosigkeit in Deutschland aus europäischer Perspektive. Webpage. Retrieved from https://www.bpb.de/apuz/270882/wohnungslosigkeit-in-deutschland-aus-europaeischer-perspektive?p=all on 26.05.2019

Cerón, J. D., López, D. M., & Eskofier, B. M. (2018). *Human Activity Recognition Using Binary Sensors, BLE Beacons, an Intelligent Floor and Acceleration Data: A Machine Learning Approach*. Paper presented at the 12th International Conference on Ubiquitous Computing and Ambient Intelligence UCAmI 2018. doi:10.3390/proceedings2191265

Chen, L., Hoey, J., Nugent, C. D., Cook, D. J., & Zhiwen, Y. (2012). Sensor-Based Activity Recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42*(6), 790-808. doi:10.1109/tsmcc.2012.2198883

Cheng, B.-C., Chen, H., & Tseng, R.-Y. (2006). *Context-Aware Gateway for Ubiquitous SIP-Based Services in Smart Homes*. Paper presented at the 2006 International Conference on Hybrid Information Technology, Cheju Island, Südkorea. doi:10.1109/ICHIT.2006.253636

Chui, M., & McCarthy, B. (2018). An executive's guide to AI. Webpage. Retrieved from https://www.mckinsey.com/~/media/McKinsey/Business%20Functions/McKinsey%20Analytics/Our%20Insights/An%20executives%20guide%20to%20AI/Executives-guide-to-AI on 26.05.2019

Citherlet, S. (2001). *Towards the Holistic Assessment of Building Performance Based on an Integrated Simulation Approach* (Dissertation), Swiss Federal Institute of Technology (EPFL), Retrieved from http://www.esru.strath.ac.uk/Documents/PhD/citherlet_thesis.pdf.

Clarke, J. A., & Hensen, J. L. M. (2015). Integrated building performance simulation: Progress, prospects and requirements. *Building and Environment, 91*, 294-306. doi:10.1016/j.buildenv.2015.04.002

Cook, D. J., Youngblood, M., Heierman, E. O., Gopalratnam, K., Rao, S., Litvin, A., & Khawaja, F. (2003). *MavHome: an agent-based smart home*. Paper presented at the Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003. (PerCom 2003). Fort Worth, TX, USA. doi:10.1109/PERCOM.2003.1192783

Coy, W. (1992). *Aufbau und Arbeitsweise von Rechenanlagen: eine Einführung in Rechnerarchitektur und Rechnerorganisation für das Grundstudium der Informatik* (2nd ed.). Wiesbaden: Vieweg+Teubner Verlag.

Darby, S., Hill, D., Auvinen, A., Barros-Dios, J. M., Baysson, H., Bochicchio, F., . . . Doll, R. (2005). Radon in homes and risk of lung cancer: collaborative analysis of individual data from 13 European case-control studies. *Bmj, 330*(7485), 223. doi:10.1136/bmj.38308.477650.63

Deloitte, & Technische Universität München. (2015, July). Ready for Takeoff? Smart Home aus Konsumentensicht. Presentation. Retrieved from https://www2.deloitte.com/content/dam/Deloitte/de/Documents/technology-media-telecommunications/Smart%20Home%20Consumer%20Survey%20Text%2020150701.pdf on 26.05.2019

Deloitte Touche Tohmatsu Limited. (2018, May 1). Smart Home Consumer Survey 2018 Ausgewählte Ergebnisse für den deutschen Markt. Survey. Retrieved from https://www2.deloitte.com/de/de/pages/about-deloitte/articles/about-deloitte.html on 26.05.2019

Deutsche Stiftung Weltbevölkerung. (2018, August 22). Soziale und demografische Daten weltweit DSW-Datenreport 2018. Report. Retrieved from https://www.dsw.org/wp-content/uploads/2018/08/DSW-Datenreport-2018_final.pdf on 26.05.2019

dial.de. (2015, April 14). Warum Gebäudeautomation? Eine Suche nach dem Sinn. Webpage. Retrieved from https://www.dial.de/de/article/warum-gebaeudeautomationeine-suche-nach-dem-sinn/ on 26.05.2019

Döbel, I., Leis, M., Molina, M., Neustroev, D., Petzka, D. H., Riemer, A., . . . Welz, D. J. (2018). Maschinelles Lernen - Eine Analyse zu Kompetenz, Forschung und Anwendung. Publication. Retrieved from https://www.bigdata.fraunhofer.de/content/dam/bigdata/de/documents/Publikationen/Fraunhofer_Studie_ML_201809.pdf on 26.05.2019

Fanger, P. O. (1973). Assessment of mans thermal comfort in practice. *British Journal of Industrial Medicine, 30*(4), 313-324. doi:10.1136/oem.30.4.313

Fauzi, C., Sulistyo, S., & Widyawan. (2018). *A Survey of Group Activity Recognition in Smart Building*. Paper presented at the 2018 International Conference on Signals and Systems (ICSigSys), Bali, Indonesien. doi:10.1109/ICSIGSYS.2018.8372651

Favreau, J. (Writer). (2018). IRON MAN. In Paramount Pictures, Marvel Enterprise, Marvel Studios, Fairview Entertainment, Dark Blades Films, & L. Entertainment (Producer).

FHEM e.V. (2019). FHEM. Webpage. Retrieved from https://www.fhem.de/ on 10.05.2019

Fildebrandt, U. (2012, November 9). Modularität und Liskovsches Prinzip in komplexen Systemen. Webpage. Retrieved from https://www.heise.de/developer/artikel/Modularitaet-und-Liskovsches-Prinzip-in-komplexen-Systemen-1746936.html?seite=all on 28.05.2019

Fraunhofer-IIS. (2019). Ogema 2.0 - Open gateway energy management. Brochure. Retrieved from https://www.iis.fraunhofer.de/content/dam/iis/de/doc/lv/nsa/Flyer/OGEMA2_Open_Gateway_for_Energy_Managemen_d.pdf on 10.04.2019

Fraunhofer-Institut für Bauphysik IBP. (2019). WUFI® Bio – Beurteilung des Schimmelpilzwachstumsrisikos. Webpage. Retrieved from https://wufi.de/de/2017/03/31/wufi-bio/ on 27.05.2019

Friebe, S. (2015). *Ableitung von Kennwerten für die künstliche Beleuchtung von Wohnräumen, auf Grundlage von Beleuchtungsrichtlinien für Arbeitsstätten, zur Verwendung in Hausautomationen*. Unpublished Bachelor's thesis. Lehrstuhl für Bauphysik. Technische Universiät München. München, Deutschland.

Friebe, S. (2017). *Entwicklung und Implementierung eines Modells zur Vorhersage von raumluftfeuchterelevanten Nutzeraktionen*. Unpublished Master's thesis. Lehrstuhl für Bauphysik. Technischen Universität München. München, Deutschland.

Gira. (2019). Gira X1. Webpage. Retrieved from https://partner.gira.de/gebaeudetechnik/systeme/knx-eib_system/knx-produkte/systemgeraete/x1/features.html on 18.05.2019

Google.ATAP. (2019). Soli. Webpage. Retrieved from https://atap.google.com/soli/ on 07.05.2019

Görz, G., & Rollinger, C. (2000, Juni 15). Einführung in die Künstliche Intelligenz. lecture script. Retrieved from https://www-ai.cs.uni-dortmund.de/LEHRE/VORLESUNGEN/MLRN/SKRIPT/handbuch_ki-ml.pdf on 26.05.2019

Gräfe, D. (2017, Januar 5). Sprachsteuerung wird zum Schlüssel. *IdeenwerkBW.de.* Webpage. Retrieved from https://www.ideenwerkbw.de/ces2017-sprachsteuerung/ on 07.05.2019

Gray, A. (2015, Juli 07). Easy Multi-GPU Deep Learning with DIGITS 2. Webpage. Retrieved from https://devblogs.nvidia.com/easy-multi-gpu-deep-learning-digits-2/ on 06.05.2019

Grohe-AG. (2008, Juli 18). Europäer mögen's heiß GROHE: Europäische Studie der Duschgewohnheiten. Webpage. Retrieved from https://www.presseportal.de/pm/15261/1230850 on 10.04.2019

Grösser, S. (2018, Februar 19). Digitaler Zwilling. Webpage. Retrieved from https://wirtschaftslexikon.gabler.de/definition/digitaler-zwilling-54371/version-277410 on 10.04.2019

gsma.com. (2011, September). Vision of Smart Home The Role of Mobile in the Home of the Future. Report. Retrieved from https://www.gsma.com/iot/wp-content/uploads/2012/03/vision20of20smart20home20report.pdf on 26.05.2019

Herzog, D. (2017). *Einsatzmöglichkeit der nichtbrandspezifischen Haustechnik zur Abwehr von Brandereignissen*. Unpublished Master's Thesis. Lehrstuhl für Bauphysik. Technischen Universität München. München, Deutschland.

Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P. F., & Rudolph, S. (2009, September 22). OWL 2 Web Ontology Language Primer. Technical Report. Retrieved from https://www.w3.org/TR/2009/PR-owl2-primer-20090922/all.pdf on 26.05.2019

Homebridge Community. (2019). Homebridge.io. Webpage. Retrieved from https://homebridge.io/ on 10.05.2019

Hoppe, N. (2018). *Kontextsensitive Beleuchtung und Umsetzung einer Fallstudie über ein Lichtsteuerungssystem*. Unpublished Bachelor's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., & Dean, M. (2004, Mai 21). SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Webpage. Retrieved from https://www.w3.org/Submission/SWRL/#2 on 08.05.2019

Hortonworks Inc. (2016, Juli 11). Deep Learning using Spark and DL4J for fun and profit. presentation slides. Retrieved from https://www.slideshare.net/HadoopSummit/deep-learning-using-spark-and-dl4j-for-fun-and-profit on 07.05.2019

Hüber, F. (2018, Dezember 7). „Alexa, wer bin ich?“: Alexa erkennt verschiedene Benutzer an der Sprache. Webpage. Retrieved from https://www.computerbase.de/2018-12/alexa-unterschiedliche-benutzer-sprach-profil/ on 07.05.2019

IFTTT Inc. (2019). IFTTT. Webpage. Retrieved from https://ifttt.com/ on 02.05.2019

Innenraumlufthygiene-Kommission des Umweltbundesamtes. (2017). *Leitfaden zur Vorbeugung, Erfassung und Sanierung von Schimmelbefall in Gebäuden*. Dessau-Roßlau: Innenraumlufthygiene-Kommission des Umweltbundesamtes, Retrieved from https://www.umweltbundesamt.de/sites/default/files/medien/421/publikationen/uba_schimmelleitfaden_final_bf.pdf on 26.05.2019

Jun, J. A., Kim, N.-S., & Kim, E. J. (2018). *The method of providing IoE-based hierarchical context awareness*. Paper presented at the 2018 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Südkorea. doi:10.1109/ICTC.2018.8539590

Jung, L. (2018, Juli 20). Diese 6 Technologien sind inspiriert von Sci-Fi-Filmen. *Techbook.de.* Webpage. Retrieved from https://www.techbook.de/special/digitales-leben/geraete-aus-science-fiction-filmen on 26.05.2018

Karvonen, N., & Kleyko, D. (2018). *A Domain Knowledge-Based Solution for Human Activity Recognition: The UJA Dataset Analysis*. Paper presented at the 12th International Conference on Ubiquitous Computing and Ambient Intelligence (UCAmI 2018). doi:10.3390/proceedings2191261

Kim, J. Y., & Lee, G. M. (2014). *Context Awareness for Smart Ubiquitous Networks*. Paper presented at the 2014 International Conference on Electronics, Information and Communications (ICEIC), Kota Kinabalu, Malaysia. doi:10.1109/ELINFOCOM.2014.6914371

Krötzsch, M., & Rudolph, S. (2010, Dezember 16). Semantic Web Technologies 1 -SPARQL Syntax und Intuition. lecture slides. Retrieved from http://semantic-web-grundlagen.de/w/images/3/35/9-SPARQL-Einfuehrung-Intuition-2010.pdf on 26.05.2019

Kuhn, D. T. (2017, November 2). Digitaler Zwilling. Webpage. Retrieved from https://gi.de/informatiklexikon/digitaler-zwilling/ on 08.05.2019

Lally, P., van Jaarsveld, C. H. M., Potts, H. W. W., & Wardle, J. (2010). How are habits formed: Modelling habit formation in the real world. *European Journal of Social Psychology, 40*(6), 998-1009. doi:10.1002/ejsp.674

Lämmel, U., & Cleve, J. (2011). *Künstliche Intelligenz*. München: Carl Hanser Verlag GmbH & Co. KG.

Leitner, G. (2018). Technologiewildwuchs im Smart-Home-Sektor und wie man den Endbenutzer wieder zurück ins Boot holt. *Wirtschaftsinformatik & Management, 10*(4), 52-65. doi:10.1007/s35764-018-0078-x

Lenzen, M. (2018). Teil 1 Was Künstliche Intelligenz kann. In *Künstliche Intelligenz: Was sie kann & was uns erwartet* (1st ed., pp. 20-145). München: Verlag C.H.Beck.

Liu, Q., Yang, X., & Deng, L. (2018). An IBeacon-Based Location System for Smart Home Control. *Sensors (Basel), 18*(6). doi:10.3390/s18061897

Livio Mazzarella, M. P. (2009). *Building energy simulation and object-oriented modelling review and reflections upon achieved results and further developments*. Paper presented at the Eleventh International IBPSA Conference, Glasgow, Scotland. retrieved from http://ibpsa.org/proceedings/BS2009/BS09_0638_645.pdf

Luria, M., Hoffman, G., & Zuckerman, O. (2017). *Comparing Social Robot, Screen and Voice Interfaces for Smart-Home Control*. Paper presented at the Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17, Denver, CO, USA. doi:10.1145/3025453.3025786

Lütolf, R. (1992). *Smart Home Concept and the Integration of Energy Meters into a Home Based System*. Paper presented at the Seventh International Conference on Metering Apparatus and Tariffs for Electricity Supply 1992, Glasgow, UK. retrieved from https://ieeexplore.ieee.org/document/187310

Ma, T., Kim, Y.-D., Ma, Q., Tang, M., & Zhou, W. (2005). *Context-Aware Implementation based on CBR for Smart Home*. Paper presented at the WiMob'2005), IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, 2005., Montreal, Que., Canada. doi:10.1109/WIMOB.2005.1512957

Mahdavi, A. (2011). *The human dimension of building performance simulation*. Paper presented at the 12th Conference of International Building Performance Simulation Association, Sydney, Australia. retrieved from http://www.ibpsa.org/proceedings/BS2011/K3.pdf

Mahroo, A., Spoladore, D., Caldarola, E., Modoni, G., & Sacco, M. (2018). *Enabling the Smart Home Through a Semantic-Based Context-Aware System*. Paper presented at the 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Athen, Griechenland. doi:10.1109/PERCOMW.2018.8480414

Maier, A.-L. (2016). *Modellierung der Lüftungsströmung in einer 2-Zimmer-Wohnung mittels CFD und Erarbeitung einer optimierten Lüftungsstrategie zur Vermeidung von Feuchteschäden*. Unpublished Master's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Maier, G. W. (2018, February 19). Kommunikation - Ausführliche Definition. Webpage. Retrieved from https://wirtschaftslexikon.gabler.de/definition/kommunikation-37167/version-260610 on 02.05.2018

Marggraf-Micheel, C., Maier, J., & Albers, F. (2011). *Klimaempfinden und Komfort in der Flugzeugkabine moderiert durch farbiges Licht?! – Ergebnisse einer Studie aus dem Projekt LiKab*. Paper presented at the Deutscher Luft- und Raumfahrtkongress, Bremen. retrieved from https://www.researchgate.net/publication/280876276_Klimaempfinden_und_Komfort_in_der_Flugzeugkabine_moderiert_durch_farbiges_Licht_-_Ergebnisse_einer_Studie_aus_dem_Projekt_LiKab

Martin, G. (2016). Smart Visu. Webpage. Retrieved from http://www.smartvisu.de/ on 10.05.2019

Maslow, A. H. (1943). A Theory of Human Motivation. *Psychological Review, 50*, 370-396. Retrieved from http://psychclassics.yorku.ca/Maslow/motivation.htm on 26.05.2019.

Mathworks. (2018). Introducing Deep Learning with MATLAB. Presentation. Retrieved from https://de.mathworks.com/content/dam/mathworks/tag-team/Objects/d/80879v00_Deep_Learning_ebook.pdf on 26.05.2019

Mehr, H. D., Polat, H., & Cetin, A. (2016). *Resident Activity Recognition in Smart Homes by Using Artificial Neural Networks*. Paper presented at the 2016 4th International Istanbul Smart Grid Congress and Fair (ICSG), Istanbul, Türkei. doi:10.1109/SGCF.2016.7492428

Mitterhofer, M. J. (2017). *A Methodology for a Scalable Building Performance Simulation based on Modular Components.* (Dissertation), Technische Universität München, Retrieved from http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20171220-1391983-1-3

Motik, B., Grau, B. C., & Sattler, U. (2008). *Structured Objects in OWL: Representation and Reasoning*. Paper presented at the Proceeding of the 17th International Conference on World Wide Web 2008, WWW'08|Int. World Wide Web Conf. - Proc. Int. Conf. World Wide Web, WWW, Beijing, China. doi:10.1145/1367497.1367573

Nahrstedt, H. (2009). *C++ für Ingenieure*. Wiesbaden: Vieweg + Teubner, GWV Fachverlage GmbH,.

Neldel, L.-F. V. (2018). *Darstellung der aktuellen Möglichkeiten zur Aktivitätserkennung und kontextsensitiven Steuerung in Wohnräumen eines Smart Homes.* (Bachelor's thesis), Technische Universität München, Retrieved from https://mediatum.ub.tum.de/1485150.

nevisQ GmbH. (2018). NevisQ. Webpage. Retrieved from https://www.nevisq.com/home on 10.05.2019

Norrefeldt, V. (2013). *VEPZO – Velocity Propagating Zonal Model – A locally refined airflow model for confined spaces to use in optimization applications.* (Dissertation), Universität Stuttgart, Retrieved from http://publica.fraunhofer.de/dokumente/N-236289.html.

Norvig, P., & Russell, S. J. (2013). *Artificial Intelligence: A modern approach* (3rd ed.). New Jersey: Prentice Hall.

Nouidui, T. S., Wetter, M., & Zuo, W. (2013). Functional Mock-up Unit for Co-Simulation Import in EnergyPlus. *Journal of Building Performance Simulation, 7*. doi:10.1080/19401493.2013.808265

Nvidia. (2019). Deep Learning Frameworks Webpage. Retrieved from https://developer.nvidia.com/deep-learning-frameworks on 08.05.2019

Ojagh, S., Malek, M. R., Saeedi, S., & Liang, S. (2018). *An Internet of Things (IoT) Approach for Automatic Context Detection*. Paper presented at the 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, Canada. doi:10.1109/iemcon.2018.8614780

openHAB Foundation e.V. (2019). openHAB. Webpage. Retrieved from https://www.openhab.org/ on 10.05.2019

Ospan, B., Khan, N., Augusto, A., Quinde, M., & Nurgaliyev, K. (2018). *Context Aware Virtual Assistant with Case-Based Conflict Resolution in Multi-User Smart Home Environment*. Paper presented at the 2018 International Conference on Computing and Network Communications (CoCoNet), Astana, Kazakhstan. doi:10.1109/CoCoNet.2018.8476898

Panella, M., & Altilio, R. (2019). A smartphone-based application using machine learning for gesture recognition: Using feature extraction and template matching via Hu image moments to recognize gestures. *IEEE Consumer Electronics Magazine, 8*(1), 25-29. doi:10.1109/MCE.2018.2868109

Paulheim, H. (2013, January 13). Vorlesung Semantic Web. lecture slides. Retrieved from https://www.ke.tu-darmstadt.de/lehre/archiv/ws-12-13/semantic-web/slides/13_Regeln.pdf on 26.05.2019

Pawara, P., Okafor, E., Schomaker, L., & Wiering, M. (2017). *Data Augmentation for Plant Classification*. Paper presented at the Advanced Concepts for Intelligent Vision Systems (ACIVS2017), Antwerp, Belgium. doi:10.1007/978-3-319-70353-4_52

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python: Choosing the right estimator. Webpage. Retrieved from https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html on 02.05.2019

Peikos, A., & Binsfeld, C. (2018). *Determination of the Thermally Comfortable Air Temperature with Consideration of Individual Clothing and Activity as Preparation for a New Smart Home Heating System*. Paper presented at the 12th International Conference on Ubiquitous Computing and Ambient Intelligence (UCAmI 2018), Punta Cana. doi:10.3390/proceedings2191224

Plate, J. (2019). Einführung in Betriebssysteme - Was ist ein Betriebssystem. Webpage. Retrieved from https://www.tu-chemnitz.de/informatik/friz/Grundl-Inf/Betriebssysteme/Script/bs1.html on 10.04.2019

Pohl, I. D. (2002). Richtiges Lüften und Möblieren – im Spiegel der Rechtsprechung. Webpage. Retrieved from https://www.berliner-mieterverein.de/recht/mieturteile/02477.htm on 23.05.2019

Pomberger, G., & Blaschek, G. (1996). *Software-Engineering Prototyping und objektorientierte Software-Entwicklung* (2nd ed.). München: Hanser Verlag.

Popgavrilova, G. (2017). *Entwicklung eines allgemeingültigen Models für ein sich am Schlafprozess orientierenden Kunstlicht-Beleuchtungssystem*. Unpublished Bachelor's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Rafferty, J., Synnott, J., Nugent, C., Cleland, I., Ennis, A., Catherwood, P., . . . Morrison, G. (2018). *Safe Beacon: A Bluetooth Based Solution to Monitor Egress of Dementia Sufferers within a Residential Setting*. Paper presented at the 12th International Conference on Ubiquitous Computing and Ambient Intelligence (UCAmI 2018), Punta Cana, Dominican Republic. doi:10.3390/proceedings2191218

Rebala, G., Ravi, A., & Churiwala, S. (2019). *An Introduction to Machine Learning*. Cham: Springer.

Robles, R. J., & Kim, T.-h. (2010). Review: Context Aware Tools for Smart Home Development. *International Journal of Smart Home, 4*, 1-12. Retrieved from https://pdfs.semanticscholar.org/d401/cc56463fbfabe011be42cb87432458e6e4fd.pdf on 26.05.2019.

Rumelhart, D. E., Widrow, B., & Lehr, M. A. (1994). The basic ideas in neural networks. *Communications of the ACM, 37*(3), 87-92. doi:10.1145/175247.175256

Salomón, S., & Tîrnăucă, C. (2018). *Human Activity Recognition through Weighted Finite Automata*. Paper presented at the 12th International Conference on Ubiquitous Computing and Ambient Intelligence (UCAmI 2018), Punta Cana, Dominican Republic. doi:10.3390/proceedings2191263

Samsung. (2018). Project Pontis: Mit dem Hirn einen TV steuern. Webpage. Retrieved from https://www.samsung.com/ch/discover/social-innovation/project-pontis/ on 30.01.2019

SBZ. (2008). Fast alles Warmduscher... *SBZ, 15*. Retrieved from https://www.sbz-online.de/gentner.dll/-24-25-1525-Badtrends_MjA5Mjgw.PDF on 27.05.2019.

Schawel, C., & Billing, F. (2018). Digitalisierung. In *Top 100 Management Tools: Das wichtigste Buch eines Managers Von ABC-Analyse bis Zielvereinbarung* (pp. 105-107). Wiesbaden: Springer Fachmedien Wiesbaden.

Schmid, J. (2018). *Entwicklung und Umsetzung eines machine-learning gestützten Frameworks zur Bestimmung von Position und Aktivität von Nutzern mittels Geräuscherkennung im Smart Home*. Unpublished Master's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Schmidt, S. (2016). *Entwicklung einer neuen Methode zur thermisch – energetischen und ökonomischen Optimierung von Wohngebäuden.* (Dissertation), Technische Universität München, Retrieved from http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20161122-1320146-1-8

Sedlbauer, K. (2001). *Vorhersage von Schimmelpilzbildung auf und in Bauteilen.* (Dissertation), Universität Stuttgart.

Sedlbauer, K., Zillig, W., & Krus, M. (2001). Isoplethensysteme ermöglichen eine Abschätzung von Schimmelpilzbildung. *IBP-Mitteilung, 388*. Retrieved from https://www.irbnet.de/daten/rswb/02079009443.pdf.

Siepermann, M. (2018, Februar 19). Bottom-up-Prinzip. Webpage. Retrieved from https://wirtschaftslexikon.gabler.de/definition/bottom-prinzip-27383/version-251039 on 26.05.2019

Socha, M. (2017). *Vollständig automatisierte Wohnraumbeleuchtung*. Unpublished Bachelor's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Spraul, V. A. (2012). *THINK LIKE A PROGRAMMER - An Introduction to Creative Problem Solving*. San Francisco, CA, USA: No Starch Press Inc.,.

Stanford Vision Lab, Stanford University, & Princeton University. (2016). IMAGNET. Webpage. Retrieved from http://image-net.org/about-overview on 12.05.2019

Statista - Das Statistik-Portal. (2019, Juli). Anteil der Deutschen, der im Jahr 2001 und 2006 mindestens einmal täglich geduscht hat. *in wellnesswelt24.de (2006).* Retrieved from https://de.statista.com/statistik/daten/studie/36517/umfrage/anteil-der-befragten-die-mindestens-taeglich-einmal-duschen-in-2001-und-2006/ on 27.05.2019

Statistisches Bundesamt. (2019). Pressemitteilung Nr. 029 [Press release]. Retrieved from https://www.destatis.de/DE/Presse/Pressemitteilungen/2019/01/PD19_029_12411.html;jsessionid=9AE5955BB0BD9D07A18BB83CFFF2B6A7.internet722 on 26.05.2019

Staub, M. (2018). Schlau und schön. *Bilanz Homes, 10*. Retrieved from https://www.wiso-net.de/document/BILA__159671720 on 26.05.2019.

Strese, H., Seidel, U., Knape, T., & Botthof, A. (2010). *Smart home in Deutschland*. Berlin: VDI.

Struler, E. d., Hoeflinger, J., Kale, L., & Bhandarkar, M. (2000). *A new approach to software integration frameworks for multi-physics simulation codes.* Paper presented at the The Architecture of Scientific Software, IFIP TC2/WG2.5, Ottawa, Canada. doi:10.1007/978-0-387-35407-1_6

Stuckenschmidt, H. (2011). *Ontologien Konzepte,Technologien und Anwendungen* (2nd ed.). Heidelberg: Springer.

Sumra, H. (2018, January 21). Smartwatch apps to help jazz up your smart home with ease. *The ambient.* Webpage. Retrieved from https://www.the-ambient.com/features/smartwatch-apps-to-make-your-smart-home-smarter-179 on 26.05.2019

Tobiidynavox. (2019). Wie funktioniert Eyetracking? Webpage. Retrieved from https://www.tobiidynavox.de/wie-funktioniert-eyetracking/ on 07.05.2019

Trcka, M. (2008). *Co-simulation for Performance Prediction of InnovativeIntegrated Mechanical Energy Systems in Buildings.* (Phd Thesis), Technische Universiteit Eindhoven, Eindhoven, Netherlands. doi:10.6100/IR637246

Turan, H. (2017). *Analyse zu einer einfachen Möglichkeit der Prognose von Feuchteproduktion im Wohnraum auf Basis von Messdaten.* (Master's thesis), Technische Universität München, Retrieved from https://mediatum.ub.tum.de/node?id=1485555.

UKV - Union Krankenversicherung. (2018, Juni 27). Pflege 4.0: Ambient Assisted Living. Webpage. Retrieved from http://seniorenratgeber.handelsblatt.com/2018/06/27/pflege-4-0-ambient-assisted-living/ on 26.05.2019

University of Manchester. (2018, June 19). List of Reasoners. Webpage. Retrieved from http://owl.cs.manchester.ac.uk/tools/list-of-reasoners/ on 11.01.2019

W3C. (2011, November 10). Semantic Web terminology. Webpage. Retrieved from https://www.w3.org/2001/sw/wiki/Semantic_Web_terminology#triple on 26.05.2019

W3C. (2014, March 15). Resource Description Framework (RDF). Webpage. Retrieved from https://www.w3.org/RDF/ on 26.05.2019

Weise, Y. (2017). *Entwicklung einer Methode zur vereinfachten Berücksichti-gung der Temperaturamplitudendämpfung bei der Berechnung der Innenoberflächentemperatur herkömmlicher massiver Außenwände in Wohngebäuden*. Unpublished Master's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Welle, B., Haymaker, J., & Rogers, Z. (2011). ThermalOpt: A methodology for automated BIM-based multidisciplinary thermal simulation for use in optimization environments. *Building Simulation, 4*, 293-313. doi:10.1007/s12273-011-0052-5

Werani, M. (2016). *Erstellung eines Modells zur Vermeidung von tageslichtbedingten Blenderscheinungen am häuslichen Arbeitsplatz durch automatisch gesteuerte Blend- und Sonnenschutzsysteme am Beispiel eines konventionellen Rollladens*. Unpublished Bachelor's thesis. Lehrstuhl für Bauphysik. Technische Universität München. München, Deutschland.

Wildemann, H. (2014). *Modularisierung in Organisation, Produkten, Produktion und Services*. München: TCW Transfer-Centrum GmbH & Co.KG.

Wisser, K. (2018). *Gebäudeautomation in Wohngebäuden (Smart Home)*. Wiesbaden: Springer Vieweg.

Zhang, S., Shapiro, N., Gehrke, G., Castner, J., Liu, Z., Guo, B., . . . Dannemiller, K. C. (2019). Smartphone App for Residential Testing of Formaldehyde (SmART-Form). *Building and Environment, 148*, 567-578. doi:10.1016/j.buildenv.2018.11.029

Zou, H., Zhou, Y., Yang, J., Jiang, H., Xie, L., & Spanos, C. J. (2018). *DeepSense: Device-free Human Activity Recognition via Autoencoder Long-term Recurrent Convolutional Network*. Paper presented at the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA. doi:10.1109/ICC.2018.8422895

Anhangsverzeichnis

Anhang A - Weitere untersuchte bauphysikalische Aspekte

Im Rahmen der Untersuchung verschiedener bauphysikalischer Aspekte wurden zusätzlich folgende Funktionen getestet, welche für diese Arbeit im weiteren Verlauf keine oder nur untergeordnete Relevanz hatten. Die Untersuchungen oder Umsetzungen erfolgten teilweise im Rahmen von Abschlussarbeiten am Lehrstuhl für Bauphysik unter der Anleitung des Verfassers.

Funktionalität	**Untersucht durch eine**	**Name des Verfassers**
Vermeidung von tageslichtbedingten Blenderscheinungen	Bachelorarbeit	Werani (2016)
Kennwerte für die künstliche Beleuchtung im Wohnraum	Bachelorarbeit	Friebe (2015)
Nutzerfreundlichkeit von Hausautomationslösungen	Bachelorarbeit	Anderlik (2015)
Biologische Wirksamkeit der Beleuchtung im automatisierten Wohnraum	Bachelorarbeit	Socha (2017)
Kontextsensitive Beleuchtung im Wohnraum	Bachelorarbeit	Hoppe (2018)
Konzipierung eines biologisch wirksamen Lichtweckers	Bachelorarbeit	Popgavrilova (2017)
Thermische Behaglichkeit im Wohnraum mit nutzerindividuellen Parametern	Masterarbeit	Binsfeld (2017)
Identifikation von raumluftfeuchterelevanten Nutzeraktionen unter Anwendung von Machine Learning	Masterarbeit	Altfelix (2017)
Vorhersage raumluftfeuchterelevanter Nutzeraktionen	Masterarbeit	Friebe (2017)
Entwicklung eines Machine Learning gestützten Frameworks zur Bestimmung von Position und Aktivität von Nutzern mittels Geräuscherkennung	Masterarbeit	Schmid (2018)
Einsatzmöglichkeit der nicht brandspezifischen Haustechnik zur Abwehr von Brandereignissen	Masterarbeit	Herzog (2017)
CFD-Modellierung von Luftströmungen in einer zwei Zimmerwohnung und Entwicklung optimierter Lüftungsstrategien	Masterarbeit	A.-L. Maier (2016)
Berücksichtigung der Temperaturamplitudendämpfung bei der Berechnung der Innenoberflächentemperatur massiver Außenwände.	Masterarbeit	Weise (2017)
Funktionsmodul für den DNZ zur außenlichtadaptiven künstlichen Beleuchtung	----------	Alexander Peikos
Wirkmodul zur Positionsabhängigen Ausgabe von Sprachausgaben im Wohnraum	----------	Alexander Peikos
Präsenzschaltung zur sicheren Anwesenheitserkennung mittels Smartphones	----------	Alexander Peikos

Anhang B - Übersicht quelloffener Smarthome-Server

Tabelle 38 Übersicht Smarthome-Server Open Source-Systeme (Daten aus Black Duck Software Inc, 2018)

Name	Lines of Code in 1000	contributors/ last year	contributors/ all time	commits /last year	commits /all time	Program Language	License
FHEM	558	80	220	2,265	17,437	Perl	GPL2
openHAB2	470	157	310	1,174	3,958	Java	EPL-1.0
openHAB	1,120	284	899	2,622	16,498	Java	EPL-1.0
Domoticz	512	91	279	1,174	7,594	C++	GPL2
Home Assistant	301	1,234	2,364	9,396	29,789	Python	MIT
MisterHouse	342	11	80	135	4,519	Perl	
Calaos	435	4	17	102	2,829	C++	GPL2.0+
MajorDoMo home automation	373	24	41	546	1,375	JavaScript	MIT
Eclipse SmartHome	275	90	207	1,118	4,929	Java	EPL-1.0
EventGhost							
ioBroker	1,756	152	234	8,109	27,006	JavaScript	Apache-ish License
Jeedom	1,056	68	101	6,288	13,337	PHP	
LinuxMCE	20,937	6	70	45	30,802	C	GNU GPL v2
OpenNetHome	60	1	7	7	646	Java	GPL-3.0+
Alljoyn	37	11	25	62	425	Java	Apache 2.0

Entwicklung der „Lines of Code“

Auf der Abszisse sind die Jahre angetragen, auf der Ordinate die Menge der Codezeilen
Grün steht für Leerzeilen, schwarz für Kommentare und blau für Codezeilen.

FHEM

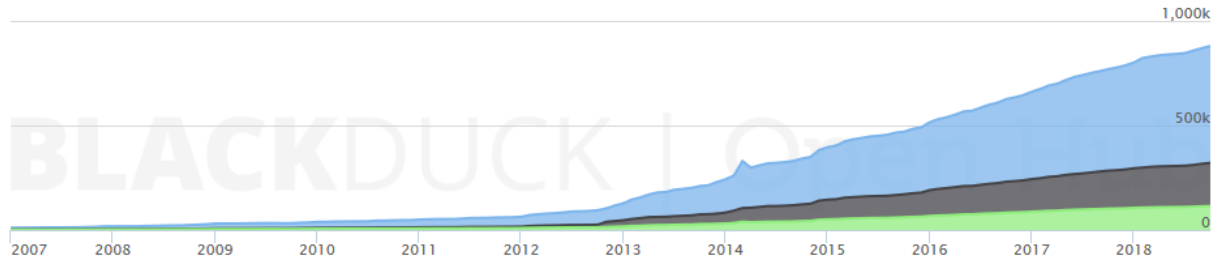


Abb. 68 FHEM Lines of Code (Black Duck Software Inc, 2018)

OpenHab2

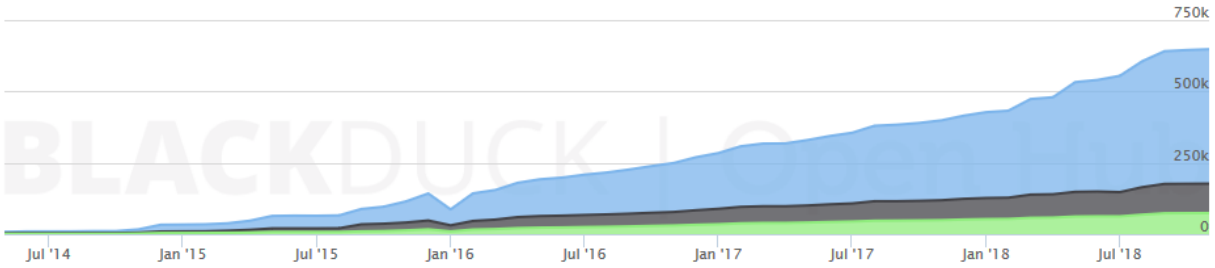


Abb. 69 OpenHab2 Lines of Code (Black Duck Software Inc, 2018)

Domoticz

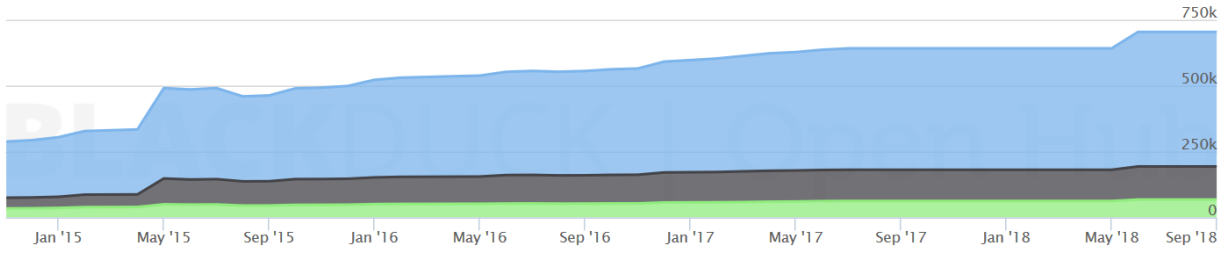


Abb. 70 Domoticz Lines of Code (Black Duck Software Inc, 2018)

Home Assistant

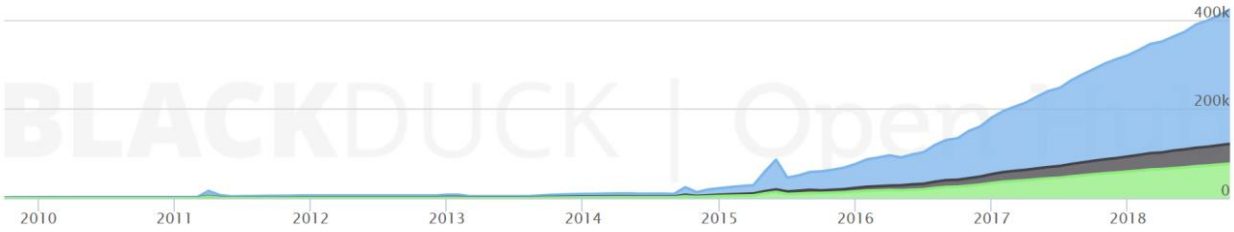


Abb. 71 Home Assistant Lines of Code (Black Duck Software Inc, 2018)

Entwicklung der „Comments“

Auf der Abszisse sind die Jahre angetragen, auf der Ordinate die Menge der unterstützenden Entwickler je Monat.

FHEM

Abb. 72 FHEM Activity Commits per Month (Black Duck Software Inc, 2018)

OpenHab2

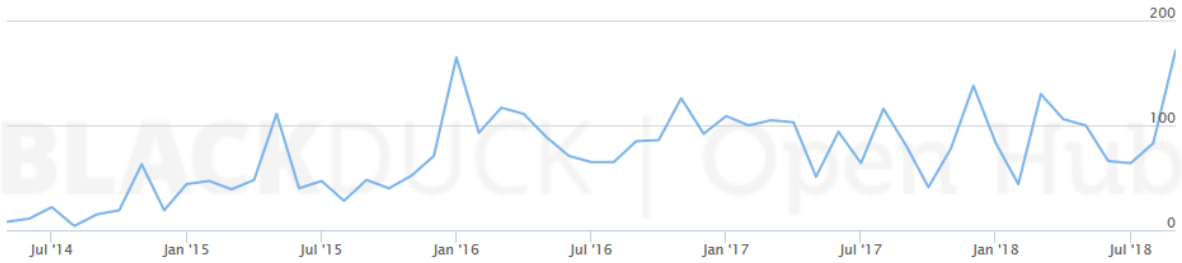


Abb. 73 OpenHab2 Activity Commits per Month (Black Duck Software Inc, 2018)

Domoticz

Abb. 74 Domoticz Activity Commits per Month (Black Duck Software Inc, 2018)

Home Assistant

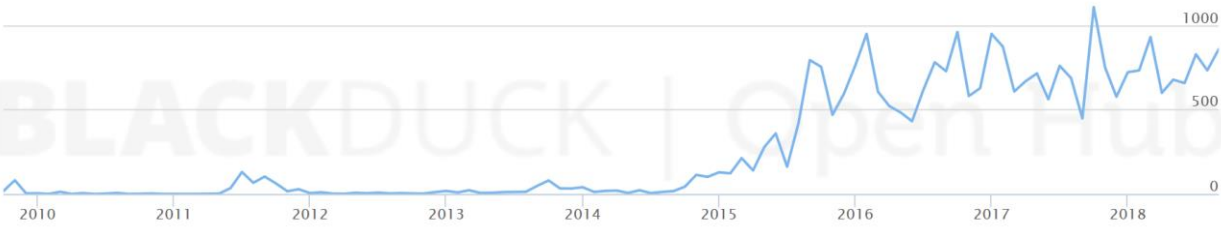


Abb. 75 Home Assistant Activity Commits per Month (Black Duck Software Inc, 2018)

Anhang C - Technische Informationen zum Framework

Für die Fallstudien wurde ein Framework erstellt, welches bestehend aus Hardware und Software die Grundlage zur Durchführung der Fallstudien bildete. Dieses wird im Folgenden näher erläutert.

Hardware

Direkt in der Wohnung wurde eine Hardwareinstanz platziert, deren Rechnerleistung für die getesteten Anwendungsfälle gerade ausreichend ist, um die Basis des Automationsservers zu beherbergen und die Interfaces, welche sich innerhalb der Wohnung befinden müssen, anzubinden. Zu diesem Zweck wurde ein NAS der Firma Synology verwendet. Das Modell 918+ ist bezüglich der Systemleistung für Virtualisierungen geeignet und bietet eine freie Auswahl des Betriebssystems für den Automationsserver. Zusätzlich bietet es durch die Möglichkeit vier Festplatten einzubinden genügend Speicherplatz, um die Messdaten zu verarbeiten.

Weiterhin wurde im Kellerabteil der Wohnung ein Industrieserver installiert, welcher die Virtualisierung verschiedenster Betriebssysteme ermöglicht. Es handelt sich hierbei um ein Gerät der Firma Fujitsu mit der Modellnummer RX300 S5.

Die Verbindung beider Server wurde über ein lokales Netzwerk hergestellt. Zu diesem Zweck musste das Netzwerk mittels D-Lan Technologie über die Stromleitungen des Gebäudes in den Keller erweitert werden. Das Management des Netzwerks wurde mit Hilfe eines Routers der Firma AVM in Form einer FritzBox 6490 Cable unter der Firmware Fritz!OS 07.01 bereitgestellt.

Die Anbindung verschiedener Smarthome-Geräte wurde, sofern möglich, über das Netzwerk erstellt oder mit Hilfe von USB-verbundener Gateways, welche direkt mit dem NAS verbunden wurden.

In folgender Tabelle sind die für die Fallstudien notwendigen Geräte aufgeführt.

Tabelle 39 Übersicht der, in den Fallstudien, verwendeten Geräte

Gerätename	Gerätetyp	Messgrößen/Aktionen	Hersteller - Interface
TempHumBad2	Sensor	Temperatur und Rel. Feuchte der Luft	Eigenbau – Eigenbau Panstamp (Arduino mit Funk)
TempHumEx	Sensor	Temperatur und Rel. Feuchte der Luft	Eigenbau – Eigenbau Panstamp (Arduino mit Funk)
AbluftBad	2 - fach Schaltaktuator	On/Off Schalten von zwei unabhängigen Geräten	HomeMatic – BidCoS Netzwerkinterface
SqueezeBoxBad	Audiowiedergabegerät	Audioausgabe verschiedener Formate und Farbdisplay	Logitech Squeezebox - Netzwerk
Handtuchheizkoerper_Switch	Steckdosen-Schaltaktuator	An- Ausschalten des Handtuchheizkörpers	HomeMatic – BidCoS Netzwerkinterface

Software

Eine Übersicht aller im Gesamtsystem verwendeten Softwarekomponenten und deren Aufgabe kann den folgenden Tabellen entnommen werden.

NAS-System Wohnung

Auf dem NAS-System kam das Synology eigene Betriebssystem DSM in der Version „6.2.1-23824 Update 4“ zum Einsatz. Dieses bietet bereits eine Applikation zur Erstellung einer MySQL ähnlichen Datenbanklösung mit dem Namen MariaDB in der Version 10.3.7-0051, wodurch eine einfache Integration möglich war. Durch die Virtualisierungsmöglichkeit auf dem NAS konnte für den Automationsserver ein geeignetes Betriebssystem ausgewählt werden. Eine Übersicht der auf dem NAS genutzten Systeme und Softwarekomponenten kann der nachfolgenden Tabelle entnommen werden.

Tabelle 40 Übersicht der auf der NAS eingesetzten Software-Systeme

Systemteil	Software und Version	Version	Integration
Betriebssystem NAS	DSM	6.2.1-23824	Integriert
Datenbanksystem	MariaDB	10.0.34-0013	Software
Administration der Datenbank	phpMyAdmin	4.7.8-0175	Software
Virtualisierungssoftware für Betriebssystem Automationsserver	Virtual Machine Manager	2.3.0-8722	Software
Virtualisierungssoftware für Betriebssystem Aufgabe der Datenhaltung	Docker	17.05.0-0395	Software
Audio und Mediaserver	Logitech Media Server	7.9.1-166	Software
Betriebssystem zur Verarbeitung regelmäßiger Aufgaben zur Datenhaltung	Debian GNU/Linux	9.3	Virtuell
Software für regelmäßige Aufgaben der Datenhaltung	SQL-Backup Perl-Script	0.1	Software
Betriebssystem Automationsserver	Ubuntu	16.04.5 LTS	Virtuell
Automationsserver	FHEM	5.9 Rev. 19381	Software

Server Keller

Auf dem Industrieserver wurde zur Virtualisierung die Software VMware verwendet. Mit dieser konnte das für die Programmierumgebung erforderliche Linux bereitgestellt werden. Eine Übersicht der auf dem Industrieserver instanziierten Systeme ist folgender Tabelle zu entnehmen.

Tabelle 41 Übersicht der auf dem Server eingesetzten Software-Systeme

	Software	Version	Integration
Betriebssystem Server	VMware ESXi	5.5 Update 1 VMKernel Release Build 1746018	integriert
Betriebssystem für Matlab und als Gegenstelle Automationsserver	Ubuntu	16.04.4 LTS	virtuell
Automationsserver	FHEM	5.9 Rev. 19381	Software
Datenbankserver	MySQL	5.7.26-Ubuntu0.16.04.1	Software
Programm für Hauptalgorithmus	Matlab	9.0.0.341360 (R2016a)	Software
Software um Matlabskripte „headless“ im Hintergrund laufen zu lassen	tmux	2.1	Software
OWL	Protege	3.5.6	Software
JFact	JFact	1.2.3	Software

Die beiden Automationsserver lassen sich über das sogenannte fhem2fhem Plug-in miteinander verbinden, wodurch ein direkter Daten- und Befehlsaustausch möglich ist.
Für komplexere Berechnungsaufgaben wurde auf dem Industrieserver MATLAB installiert, welches auch als Programmierumgebung für die hier vorgestellte Methode genutzt wurde. Zusätzlich ermöglicht MATLAB einen einfachen Zugriff auf Datenbanken und per Telnet-Protokoll auch eine Kommunikationsschnittstelle mit dem Automationsserver. Weiterhin beheimatet der Linux Server auch die OWL sowie den Reasoner JFact mit dessen Hilfe Matlab die Ontologie auslesen konnte.

Die Erstellung sowie das Reasoning der Ontologie wurde mit der Software Protege in der Version 5.5.0 Build beta-5-SNAPSHOT auf einem Laptop durchgeführt. Für das Reasoning wurde das Pellet Reasoner Plug-in in der Version 2.20 verwendet. Dieses eignete sich besser für das Reasoning von Regeln und ist bereits in Protege integriert.

Anhang D - Instanziierte Server im Gesamtsystem

Die folgende Abbildung zeigt die Softwarekomponenten des Gesamtsystems und deren Verknüpfung über Datenströme.

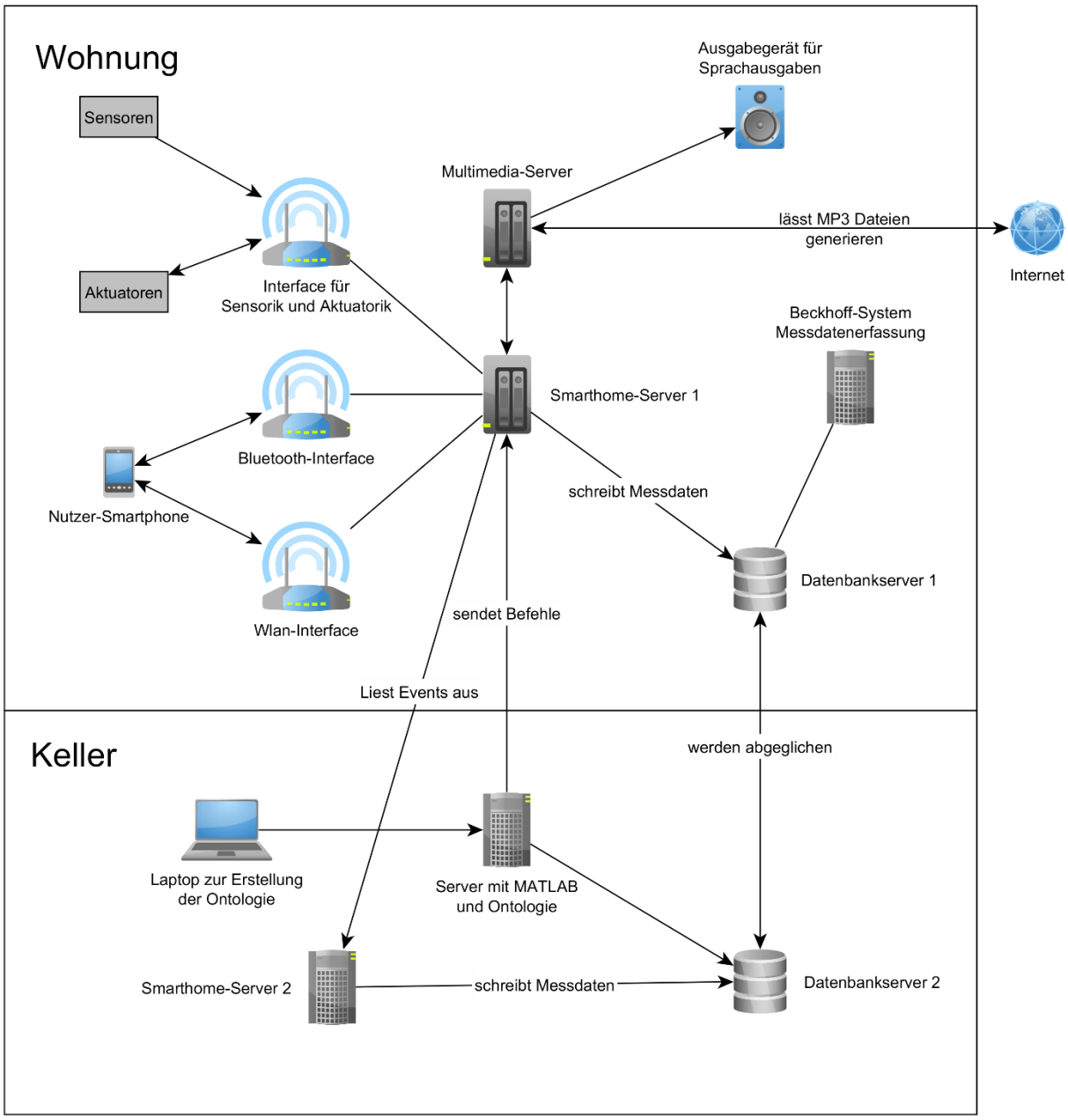


Abb. 76 Zusammensetzung des Smarthome-Framework

Anhang E – Beschreibung des Hauptalgorithmus

Der in dieser Arbeit entwickelte Hauptalgorithmus gliedert sich in sechs Abschnitte, deren grundsätzliche Funktion im Folgenden beschrieben wird.

1.) Laden der Ontologie-Daten und der Module
Im ersten Abschnitt wird die Ontologie geladen. Hierbei werden alle definierten Klassen und Eigenschaften ausgelesen sowie alle in der Ontologie enthaltenen Module geladen und instanziiert werden.

2.) Initialisieren des Rechensetups sowie Einlesen der Individuen
Daraufhin wird das Rechensetup initialisiert, damit es für die weiteren Schritte zur Verfügung steht. Alle vorhandenen Aktuatoren, Sensor-Readings und Module werden ausgelesen und mit ihren IDS und ODS in das Rechensetup integriert. Dabei wird auch überprüft, welche IDS oder ODS der Module genutzt werden. Weiterhin werden alle Verbindungen über die Eigenschaft „isConnectedWith“ aus der Ontologie ausgelesen und im Rechensetup hinterlegt. Gefundene Modulverbindungen werden im Rechensetup verknüpft.

3.) Sensordaten auslesen
Aus den Datenquellen werden die entsprechenden Sensordaten ausgelesen und ins Rechensetup geschrieben. Die Daten werden hierbei vom Datensammlermodul in die benötigte Form (Zeitintervall, Einheiten) überführt.

4.) Berechnen der Module
Im ersten Schritt werden alle Module berechnet, deren Inputdatenströme nur aus Sensordaten stammen. Darauffolgend werden Module berechnet, welche Inputdatenströme aus Sensoren und anderen Modulen benötigen. Hierbei erfolgt eine Überprüfung, ob bereits alle Daten für eine Berechnung zur Verfügung stehen, andernfalls werden Module, bei denen Daten fehlen, in der Reihenfolge nach hinten gestellt. Die dabei generierten Moduloutputs von Funktionsmodulen werden in einer „Structure“ gesammelt. Diese ist ein Konstrukt in MATLAB, das die uneingeschränkte Zuordnung von Daten jeglicher Form in gemischter Weise zulässt.

5.) Prioritätenverarbeitung
Zuerst wird eine Zusammenstellung aller ausgegebene Wirkgrößen der einzelnen Module erstellt. Alle Wirkgrößenoutputs werden nun, entsprechend ihrer Prioritäten und Zielwerte, verarbeitet. Es werden alle berücksichtigten und sich widersprechenden Wirkgrößen bestimmt und diese Informationen im Rechensetup hinterlegt.

6.) Auswahl der steuerbaren Wirkgrößen
Die steuerbaren Wirkgrößen werden ausgewählt. Zusätzlich wird überprüft, ob für ein Modul keine steuerbaren Wirkgrößen vorhanden sind und der „Dilemma-Check“ wird durchgeführt. Anschließend werden alle relevanten Daten in der Datenbank gespeichert. Die Berechnung der Wirkmodule führt zum Absetzen der notwendigen Befehle.