Teil I - Kurzfassung des Schlussberichts

Nutzerinterfaces sind die zentrale Stelle, die es Menschen ermöglicht mit Maschinen zu interagieren. Während Maschinen einer Baureihe wenig individualisiert werden und immer auf die gleiche Art und Weise behandelt werden können, trifft dies auf die menschlichen Kommunikationspartner eher in den seltensten Fällen zu. Dennoch wird seit dem Beginn des digitalen Zeitalters genau dies mehr oder weniger vorausgesetzt bzw. nicht berücksichtigt. Eine der wohl am häufigsten bedienten Maschinen sind elektrische Haushaltsgeräte, die seit einigen Jahren immer mehr Funktionen erhalten sowie digitaler werden. Dieser Fortschritt bringt jedoch auch den Nachteil einer immer stärker werdenden Komplexität in der Bedienung mit sich. Insbesondere für weniger technik-affine oder unerfahrene Nutzer, kann somit das einfache Zubereiten einer Tiefkühlpizza im schlimmsten Fall zu einer echten Alltagshürde werden.
Mit einer „One fits all“ Bedienung laufen Hersteller Gefahr, den individuellen Bedürfnissen, Kenntnissen und Fähigkeiten ihrer Kund*innen nicht gerecht zu werden. Die gleiche Bedienung für alle ist mit einer Vielzahl von Kompromissen verbunden, die sowohl den Fortschritt als auch das Bedienerlebnis einschränken.
Das Projekt Plug-In adressiert diese Thematik und stellt im Ergebnis eine technische Möglichkeit zur Individualisierung von Nutzerinterfaces zur Verfügung.

Der Projektverlauf war von der Coronapandemie gezeichnet, so dass das initial geplante, partizipative Vorgehen mit potenziellen Nutzer*innen nicht direkt umgesetzt werden konnte. Stattdessen setzten die Beteiligten in der ersten Projekthälfte daher auf aufsuchende Formate und eine quantitative Befragung. In der zweiten Hälfte wurden anschließend die initial Experten-basierten Artefakte mit potenziellen Nutzer*innen gemeinsam in Workshops weiterentwickelt und fortlaufend getestet.
Aus technischer Sicht wurde im Laufe des Projektes ein funktionsfähiger Prototyp entwickelt, um die Interaktion zwischen Menschen und Maschinen individuell auf die Bedürfnisse, Vorlieben sowie Fähigkeiten von Nutzer*innen anzupassen. Dabei wurde darauf geachtet, die Verfahren so minimal invasiv für den Alltag der Menschen wie möglich zu gestalten und Daten lokal zu verarbeiten.

Die Firma QuinScape hat hierfür zusammen mit Projektpartnern einen Baukasten für die Generierung von entsprechenden grafischen Nutzerinterfaces entwickelt. Hierfür wurden exemplarische Interfaces für spezifische Haushaltsgeräte in einem iterativen Designprozess entworfen, aus welchem sich anschließend wiederverwendbare Bedienelemente ableiten ließen.
Gerätehersteller wurden durch eine im Projekt entwickelte Beschreibungssprache dazu befähigt, eigene Interfaces für entsprechende Haushaltsgeräte aller beliebiger Art zu entwerfen. Ein Tooling um diese Interfaces individuell auf Benutzer anzupassen wurde ebenfalls zur Verfügung gestellt. Durch Projektpartner wurde eine umfangreiche Softwareinfrastruktur entwickelt, um Nutzer*innen zu identifizieren und dessen Bediengewohnheiten in Form von verschiedenen Metriken zu. Auf Basis der so ermittelten Informationen konnte QuinScape Algorithmen entwickeln, welche eine personen-spezifische Anpassung der durch die Gerätehersteller zur Verfügung gestellten Nutzerinterfaces auf den jeweiligen Benutzer ermöglicht. Ebenfalls wurden Algorithmen entwickelt, die die voraussichtliche Interaktion des Nutzers vorhersagt und auf der Bedienschnittstelle entsprechende Shortcuts anzeigt.

Für die Integration in das Gesamtsystem wurden in enger Kooperation mit der Fachhochschule Dortmund moderne und hoch anpassungsfähige Softwarearchitekturen entworfen und in einem iterativen Prozess realisiert. Eine Möglichkeit für das Hinzufügen weiterer Funktionalität nach Einführung eines Haushaltsgerätes wurde durch Mehrwertservices ebenfalls implementiert und im Projektverlauf erfolgreich getestet.

PLUGIN

Schlussbericht

Zuwendungsempfänger:
QuinScape GmbH

Förderkennzeichen:
16SV8452

Vorhabensbezeichnung:
Plattform selbstadaptiver Benutzungsschnittstellen zur Gerätebedienung als individuelles Assistenzsystem (Plug-In)

Teilvorhaben:
Teilvorhaben IV: „Schnittstellenbaukasten, Reasoning, Mehrwertservices und Geräteintegration“

Laufzeit des Vorhabens:
01.03.2020 – 31.05.2023

Autor:
Christopher Klewes

1. Kurzdarstellung

1.1. Aufgabenstellung

Ziel des Gesamtforschungsvorhabens Plug-In war es, ein Assistenzsystem in Form einer lokalen, technischen Plattform (Plug-In-Plattform oder kurz Plug-In) zu entwickeln, welches selbstadaptive, d. h. sich auf den Nutzenden einstellende, Mensch-Geräte-Schnittstellen für Geräte in der häuslichen Umgebung bereitstellt. Die Plug-In-Plattform soll durch personalisierbare und komplexitätsreduzierende Benutzungsschnittstellen ein sukzessives, transparentes und wechselseitiges Kennenlernen zwischen Mensch und Gerät ermöglichen.

Die QuinScape GmbH zeigte sich im Rahmen dieser Gesamtzielsetzung speziell für das Teilvorhaben IV verantwortlichen. In diesem Teilvorhaben waren vor allem die Umsetzung bzw. Implementierung eines flexiblen **Designs**, d.h. die Gestaltung zusammensetzbarer Bedienelemente, und die softwaretechnische Implementierung des konkreten **Reasonings** im Fokus. Das als Softwarebaustein konzipierte Reasoning sollte ermöglichen, auf Basis von vorliegenden Nutzer*innen-, Nutzungs- und Umgebungsdaten, für eine interagierende Person die *passende* Benutzungsschnittstelle aus den vorher gestalteten Bedienelementen zusammenzusetzen.

1.2. Voraussetzungen

Digitale Technologien durchdringen zunehmend die Arbeits- und Lebenswelt von Nutzer*innen und bieten ein umfangreiches, stetig wachsendes Angebot an Diensten und Funktionen. Dieses umfangreiche Angebot bietet das Potenzial, ein zeitgemäßes und selbstständiges Leben in der eigenen häuslichen Umgebung in jeder Lebenslage und bis ins hohe Alter zu ermöglichen. Die konkrete Umsetzung scheitert für viele Nutzer*innen jedoch häufig an zu vielen und zu komplexen Bedienkonzepten, die die hohe und weiter steigende Anzahl feingranularer Haushaltsgerätefunktionen in für alle Nutzer*innen gleiche Benutzungsschnittstellen unterbringen.

Auch wenn in vergangenen Jahren eine starke Innovationskraft im Bereich Connectivity und Smart Home gezeigt hat, war zu Projektbeginn die klassische Interaktionssituation zwischen Menschen und Haushaltsgerät noch unangetastet. Geräteschnittstellen wurden und werden nach dem Prinzip „One fits all“ konzipiert. Egal ob hochaltriger Hausbewohner im Mehrgenerationenhaus oder technikaffine Enkelin, alle Personengruppen nutzen die selben Bedienelemente zur Interaktion.

1.3. Planung und Ablauf des Vorhabens

Der Ablauf des Projektes war primär gezeichnet von den aus der Coronapandemie resultierenden Herausforderungen und den dazu gefundenen Lösungen.

Initial sollten innerhalb des Projektes sechs partizipative Formate unter Leitung der Technischen Universität Dortmund und der Hochschule Hamm Lippstadt durchgeführt werden, die die Datengrundlage für Anforderungen und Designentscheidungen liefern sollten. Auf dieser Basis sollte QuinScape die Bedienelemente für den Benutzungsschnittstellengenerator und, auf Grundlage parallel entwickelter Beschreibungssprachen der Fachhochschule Dortmund, das Reasoning entwickeln.

Dieses zu Beginn stark auf Nutzerpartizipation ausgelegte Vorgehen musste durch die Pandemie insofern abgeändert werden, da eine Durchführung von Workshops nicht möglich war. Dies hatte zur Folge, dass im ersten Projektjahr auf Seiten der QuinScape viel Experten-basiert vorgegangen werden musste. So wurde bspw. eine Recherche nach Funktionen von den gängigen Haushaltsgeräten Backofen und Waschmaschine durchgeführt.

Insgesamt war es nötig, das Vorgehen insofern abzuändern, als dass Arbeitspakete und Implementierungsentscheidungen, die ohne Informationen von echten potenziellen Nutzer*innen bearbeitet werden konnten, vorgezogen wurden. Da sich auf Seiten der Fachhochschule Dortmund die Fertigstellung des Nutzermodells aus selbigem Grund verzögerte, verzögerte sich auf Seiten der QuinScape bspw. auch die Fertigstellung des Reasonings. Von den anfänglichen sechs Partizipationsformaten wurden in der zweiten Projekthälfte drei Workshops durchgeführt, was auf Seiten QuinScapes für eine Überarbeitung des anfänglich Experten-basiert erstellte Designs der Bedienelemente gesorgt hat. Insgesamt resultierte aus diesen Rahmenbedingungen die Notwendigkeit, das Projekt um 3 Monate zu verlängern.

Nichtsdestotrotz konnte, nach Verlängerung und Umstellung der Methodik innerhalb des Gesamtprojektes auf Nutzer*innenpartizipation durch aufsuchende Formate und das Einfangen der Nutzer*innenperspektive über eine quantitative Studie, das Projektziel grundsätzlich erreicht werden. Anfänglich angefacht durch die Pandemie fanden innerhalb des Konsortiums wöchentliche digitale Arbeitstreffen zum Reasoning statt, in denen der Prozess kollaborativ erarbeitet wurde, so dass er von der QuinScape umgesetzt werden konnte. Durch diese enge, digitale Abstimmung entfiel die Notwendigkeit zu kostspieligen Dienstreisen und es konnte direkt zu Mitte des zweiten Projektjahres an einem gemeinsamen Prototyp der Plug-In Plattform gearbeitet werden, den die Fachhochschule Dortmund in ihren Räumlichkeiten betrieb. Dieser früh verfügbare Prototyp wirkte sich sehr positiv auf die nötige Abstimmung der Interfaces und die generelle Testbarkeit des Reasonings und der Designelemente aus. Insgesamt bleibt zu attestieren, dass aufgrund der Rahmenbedingungen weniger Iterationen und frühe Prototypen entwickelten werden konnten sowie gleichzeitig die enge Verzahnung der Projektpartner in den regelmäßigen, digitalen und kollaborativen Arbeitstreffen für vorher ungeplante Synergien gesorgt hat. Die Coronapandemie zeigte sich also für das Projekt als sprichwörtlicher Fluch und Segen zugleich. Vor diesem Hintergrund kam die QuinScape GmbH insgesamt mit weniger Personalaufwand aus.

1.4. Ausgangslage in Wissenschaft und Technik

Wissenschaftlich zeigt die Studienlage spätestens seit den 2010er Jahren, dass die wahrgenommene Nützlichkeit (Perceived Usefulness) und die wahrgenommene Einfachheit in der Benutzung (Perceived ease of use) zu den Hauptfaktoren bei der Akzeptanz von neuen Technologien und Geräten zählen (Hubert et al., 2019). Insbesondere Ältere und nicht-technik affine Personen fühlen sich jedoch häufig von wahrgenommener Komplexität in der Bedienung abgeschreckt und nehmen diese als Hürde wahr (Vaportzis et al., 2017).

Das Forschungsvorhaben Plug-In setzte zur Adressierung dieser Bedienhürden auf Techniken und Methoden aus dem Bereich Adaptive User Interfaces (AUI) (Browne, 2014). AUI findet und fand bereits häufig Anwendung bei der Gestaltung von Benutzungsschnittstellen für Smartphone Anwendungen oder Computerbetriebssysteme. Hier werden Buttons, Farben, Kontrastwerte usw. angepasst an vorher abgefragte Präferenzen und Wissen von Nutzenden. Ziel von Plug-In und der von QuinScape zu implementierenden Reasoningkomponente war es über dieses grundsätzliche Verständnis hinaus, die Adaption der Interfaces von Haushaltsgeräten zu automatisieren, so dass wenig zusätzlicher Konfigurationsaufwand in der Nutzung entsteht und die Adaption sich gleichzeitig nicht übergriffig durch Nutzende wahrgenommen wird (Lavie & Meyer, 2010).

Zum Start des Projektes waren Haushaltsgeräte zwar in der Lage in bestimmten Parametern konfiguriert zu werden, bspw. ist es üblich, dass Nutzende Favoritenprogramme bei neueren Haushaltsgeräten festlegen können, jedoch erstreckt sich die Adaption häufig lediglich auf der angezeigten Funktionalität und muss explizit vorgenommen werden. Handelsübliche Haushaltsgeräte unterscheiden nicht zwischen unterschiedlichen Nutzenden und liefern das gleiche Interface für jede*n Nutzende*n. Technische Systeme im Haushalt, die personalisiert auf einzelne Bewohner*innen reagieren, finden sich zum Projektstart vor allem im Bereich Robotik bei der Erforschung von Mensch-Roboter-Interaktion bspw. in (Marsá-Maestre et al., 2008) oder (Saunders et al., 2016).

1.5. Zusammenarbeit mit anderen Stellen

Die QuinScape hat entsprechend des Projektplans eng mit den am Projekt beteiligten Stellen zusammengearbeitet. Insbesondere durch die Coronapandemie ergab sich ein enger, wöchentlicher Austausch zwischen allen Projektpartnern. Darüber hinaus nahm QuinScape an den halbjährlichen Lenkungskreisen unter Beteiligung des Projektträgers und den monatlichen Projekttreffen regelmäßig teil.

Bzgl. der Zusammenarbeit mit Projekt-externen Stellen ist primär das Advanced Interaction Team des assoziierten Geräteherstellers Miele zu nennen, mit denen in der ersten Hälfte des Projektes Austausch stattfand. In der zweiten Projekthälfte gelang es außerdem in Kooperati-

on mit dem Projekt Prediction to Agile Interventions in the Social Science (FAIR) der Technischen Universität Dortmund den Prototypen inkl. des von QuinScape implementierten Designs und Reasonings in echter Umgebung zu testen, so dass wertvolle Nutzungsdaten in diesem Kontext gesammelt werden konnten. Der Testbetrieb hält zum aktuellen Zeitpunkt (Okt. 2023) auch über das Projektende von Plug-In an, so dass weiter Erfahrungen für die Verwertung der Plug-In Ergebnisse auch über den Projektzeitraum hinaus gesammelt werden.

2. Eingehende Darstellung

2.1. Erzielte Ergebnisse

Die erzielten Ergebnisse innerhalb des Forschungsvorhabens beziehen sich für die QuinScape GmbH im Wesentlichen auf den im Gesamtvorhaben vorgesehenen Bereich **Reasoning** inkl. Integration in die Softwareplattform und den Bereich **Design**. Die QuinScape zeigte sich hierbei insbesondere für die praktische Umsetzung verantwortlich und implementierte auf Basis der Konzepte der Hochschulpartner zahlreiche Funktionen und Konzepte, so dass das Plug-In System zu Projektende prototypisch vollständig funktionsfähig war bzw. ist.
Im Folgenden werden die erzielten Ergebnisse mit Bezug auf die Arbeitspakete (APs) des Vorhabens erläutert. Die AP-Nummerierung basiert auf der initialen Teilvorhabensbeschreibung.

AP1: Partizipationsworkshops

QuinScape nahm wie geplant an den partizipativen Workshops teil. Durch die aufgrund der Coronapandemie geänderten Methodik fanden jedoch weniger Workshops als anvisiert statt. In den durchgeführten drei Workshops wurden zunächst MockUps und später Designprototypen diskutiert, hierbei wirkte QuinScape in der Planung und Durchführung beratend mit nutzte die Ergebnisse in der Umsetzung.

AP2: Elemente identifizieren

Die Hochschule Hamm-Lippstadt als Hauptverantwortliche im AP 2 hat mit Unterstützung von QuinScape über den Projektverlauf vier Oberkategorien von Elementen identifiziert. Diese Oberkategorien können bezeichnet werden als: (1) *Elemente genereller Benutzung*, (2) *Informative Elemente*, (3) *Konfigurationselemente* und (4) *Graphische Elemente*. Insgesamt wurden alle Elemente in ihre Einzelteile zerlegt, um die Basis für den Schnittstellengenerator zu schaffen. Die im Folgenden beschriebenen Ergebnisse wurden iterativ unter Berücksichtigung der Nutzer*innenperspektive aus den Erhebungen und des Nutzer*innenfeedbacks aus den Workshops entwickelt.

In der Oberkategorie *Elemente genereller Benutzung* befinden sich normale Buttons, Text Elemente und auch Cards. Cards sind einfarbige Hintergrund Elemente und stellen die Basis für etwaige Menüs dar. Buttons können in zwei Typen vorkommen. Das Text Element stellt

einen gegebenen Text auf der Benutzeroberfläche dar. Ein Button im Hauptmenü eines Bildschirms wird dabei mit einer Card, ein Symbol auf der Card und einem Text unter der Card dargestellt. Wohingegen ein Button in einem Untermenü oder außerhalb des Hauptmenüs lediglich eine farbige Umrandung um seinen Inhalt enthält.

Die *Informativen Elemente* beinhalten einen Kalender, eine Liste, eine Einstellungsübersicht und ein Ladebalken. Die Einstellungsübersicht zeigt die aktuelle Konfiguration der Aktion des Haushaltsgeräts an und besteht aus einer länglichen Card, Text Elemente, Buttons, Symbole und einem Ladebalken. Durch ein Tippen auf die Buttons, welche einen Parameter darstellen, wird ein Untermenü aufgerufen, in welchem eine Änderung des Parameters möglich ist. Auf der rechten Seite befindet sich abschließend noch ein Button zum Starten oder Abbrechen der jeweiligen Aktion des Haushaltsgeräts. Der Kalender besteht hauptsächlich aus einer großen Card und mehreren Text Elementen mit Symbolen. An Tagen, welche einen Datensatz hinterlegt haben, beispielsweise die Historie an diesem Tag, befindet sich ein Button mit einer weißen Umrandung. Durch das Tippen auf diesen würde sich anschließend das Untermenü zur Anzeige des Datensatzes öffnen. Oben befinden sich noch Buttons zur Auswahl des Monats.

In der Oberkategorie *Konfigurationselemente* befinden sich alle Elemente, welche eine Änderung von Parametern oder aber auch Einstellungen ermöglicht. Zur Änderung numerischer Werte kann eins von zwei Einheitsauswahlelementen verwendet. Das erste Element stellt die angrenzenden numerischen Werte in einer Art Skala dar und ermöglicht die Änderung des Werts über seitliches Wischen oder aber zwei durch Knöpfen jeweils an den Seiten der Skala. Diese Auswahlart soll es älteren Menschen ermöglichen, genauer zu erkennen, welcher Wert aktuell ausgewählt ist, da in Workshops der übliche Schieberegler oft für ältere Personen schwieriger zu verwenden war. Dieses Element besteht jeweils aus zwei Buttons und veränderbarer Text in der Mitte. Bei dem zweiten Element handelt es sich um einen üblichen Schieberegler, welcher es ermöglicht den Wert mittels einer Zieh-Bewegung zu verändern. Der Schieberegler besteht dabei aus einem Schieberegler-Griff, einer Schieberegler-Leiste, sowie Text Einheiten mit Symbolen als Skala. Abschließend ermöglicht ein umschaltbarer Button die Änderung binärer Parameter oder Einstellungen. Dieser Button verändert korrespondierend zum Wert die Hintergrund Farbe und ändert den jeweiligen Wert durch einen Druck auf den Button. Insgesamt besteht dieser umschaltbare Button, genau wie der Button, aus einer Card mit der jeweilig korrespondierenden Farbe, einem Symbol und Text.

Schließlich befinden sich in der Oberkategorie *Graphische Elemente* sowohl ein Element zum Anzeigen von Bildern als auch ein anpassbares Tabellen Element. Das Tabellen Element wird hauptsächlich in den Einstellungen zur Darstellung der vom Reasoning gesammelten Daten verwendet und besteht aus Button und Text Elementen.

Zusammenfassend konnten als Ergebnis des AP 2 die grundlegenden Elemente für den Schnittstellenbaukasten identifiziert und die Grundlage für das dynamische Assemblieren gelegt werden.

AP3: Designstudie

Im Gesamtvorhaben zeigte sich die Hochschule Hamm-Lippstadt hauptverantwortlich für die Designstudie. QuinScape lieferte in diesem Kontext die tatsächliche gestalterische Umsetzung. In einem iterativen Prozess mit der Fachhochschule Dortmund, die die technische Perspektive aus Sicht der Softwareplattform einbrachte, wurde ein Mockup kreiert, um die Elemente und Symbole des Schnittstellenbaukastens in ein Benutzerinterface umzusetzen. Dabei ergaben sich vier Oberkategorien für die Verwendung des Systems.

Abbildung 1: Hauptmenü des Mockups der Designstudie

Die erste Kategorie mit dem Namen *Identifizierung* beinhaltet alle Bildschirme, welche im Standby Modus des Geräts und beim Anmelden des/der Nutzenden erscheinen.

Anschließend wird die Person in die zweite Kategorie mit dem Namen *Menüs* geführt. Auf diesen Bildschirmen hat die interagierende Person die Möglichkeit, ein Programm für das jeweilige Haushaltsgerät auszuwählen oder je nach Gerät die Konfiguration eines eigenen Programms zu starten. Ansonsten hat der/die Nutzer*in auch die Möglichkeit, die Einstellungen des Systems oder des Haushaltsgeräts zu öffnen. Das Hauptmenü des dafür angelegten Mockups ist in Abbildung 1 dargestellt.

Hieraus ergeben sich abschließend die letzten zwei Kategorien: *Konfiguration* und *Einstellungen*. In der Kategorie *Konfiguration* befinden sich alle Bildschirme, welche direkt mit der Verwendung des Haushaltsgeräts in Verbindung stehen. Hingegen befinden sich in der Kategorie *Einstellungen* alle Bildschirmmasken, welche Parameter des Systems oder des Haushaltsgeräts permanent verändern können.

Da im weiteren Verlauf des Projektes diese Bildschirme dynamisch über das System zusammengebaut werden, hat QuinScape ein Mockup für eine Beispiel Verwendung des Haushaltsgeräts erstellt und diese iterativ auf Basis des Nutzer*innenfeedbacks evolviert.

AP4: Benutzungsschnittstellenbeschreibungssprache

Für die Realisierung einer maschinenlesbaren Interpretation eines Nutzerinterfaces war es zunächst notwendig, entsprechende Elemente zur Ansicht und Bedienung zu definieren (siehe Designstudie und Elementdesign). Gemäß der Projektskizze wurde in Zusammenarbeit mit der Fachhochschule Dortmund eine Syntax zur Definition eines Interfaces entworfen, welche einer Recherche zu bestehenden Lösungen vorausging.

In gängigen Smarthome Systemen, wie z.B. das initial verwendete OpenHAB, werden eigene hierarchische DSLs (**D**omain **S**pecific **L**anguages) verwendet, um hieraus statische Userinterfaces zu generieren. Wie in der Vorhabensbeschreibung geplant, wurden zunächst Maßnahmen unternommen, um mit Modellierungsframeworks ebenfalls eine eigene Plug-In spezifische Beschreibungssprache zu entwickeln. Aufgrund neuer Trends und Erkenntnisse im Bereich der Anwendungsentwicklung hat sich das Konsortium im frühem Entwicklungsstadium jedoch dazu entschieden, stattdessen der aufkommenden Trends von deklarativer UI-Programmierung zu folgen. Hierbei wird als Metametamodell, also die Elemente aus denen sich eigene Modellierungssprachen konstruieren lassen, jeweils direkt die Programmiersprache genutzt, sodass es zukünftigen Entwicklern stark vereinfacht wird, hierarchische Konstrukte als DSL in der gleichen Programmiersprache zu entwickeln. Dieses Konzept wird z.B. im Apple Ökosystem mit SwiftUI oder im Android Umfeld mit JetPack-Compose als Standard eingesetzt.

Da die Fachhochschule Dortmund das lokale Plug-In Backend in Kotlin realisiert hat und Kotlin oben genannte Mechanismen für eine In-Language-DSL bereitstellt, wurde das ursprünglich geplante Vorgehen das Eclipse Modeling Framework (EMF) zu verwenden abgeändert und stattdessen eine Kotlin basierte DSL entwickelt. Entwicklern von Userinterfaces für das Plug-In Ökosystemen wird so die Möglichkeit gegeben, die Logik für die Generierung der Userinterfaces inline mit der Verwendung der Elemente zu kombinieren.

Das mithilfe der Kotlin DSL generierte Metamodell kann anschließend in eine JSON-Repräsentation serialisiert werden und dem lokalen Backend mittels einer Softwareschnittstelle zugeführt werden. Durch die Serialisierung ist es zukünftig denkbar, Generatoren von Schnittstellen auch in DSLs, die in anderen Programmiersprachen eingebettet sind, anzubieten oder gänzlich auf diese zu verzichten.

Die Generatoren der Benutzungsschnittstellen wurden in Docker Images verpackt und mit einem Smarthome Gerät verknüpft. Die Generatoren werden bei Applikationsstart vom lokalen Backend heruntergeladen und just-in-time angefragt, sobald ein*e Benutzer*in identifi-

ziert wurde. In der Anfrage sind die Ergebnisse der Reasoning Prozesse enthalten, sodass die generierende Logik diese passend zum speziellen Haushaltsgerät berücksichtigen kann.

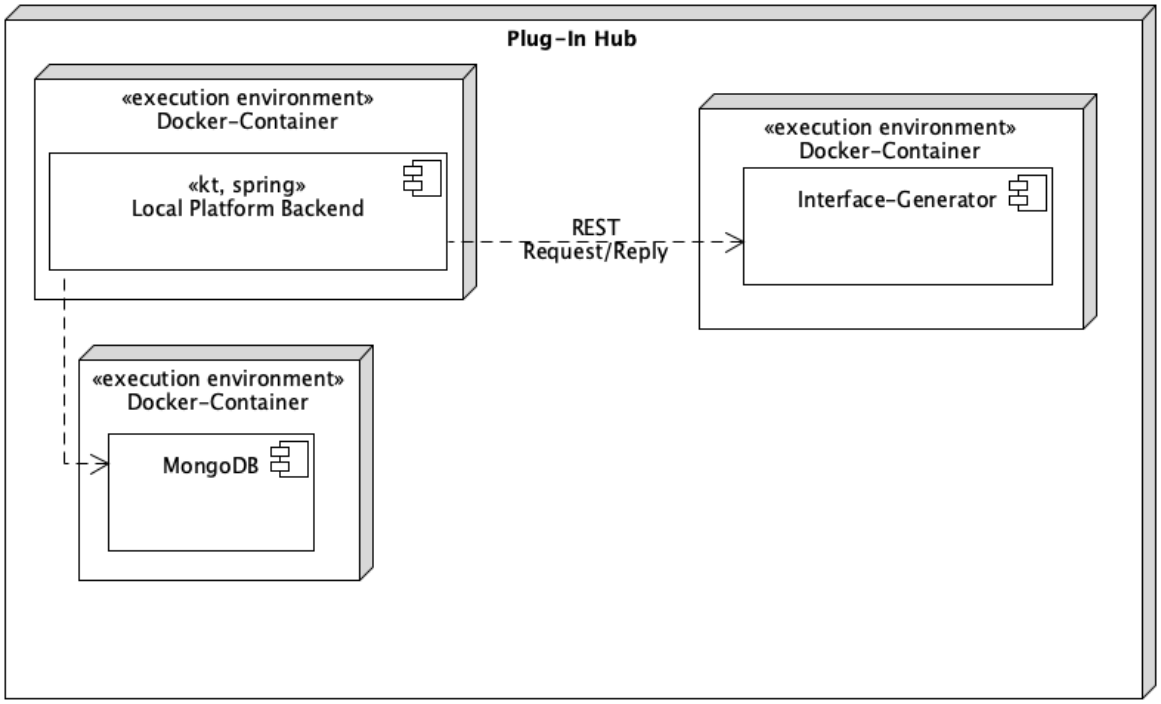

Abbildung 2 Verteilungsdiagramm Interface-Generator

AP5 & 6: Elementdesign I + II

QuinScape unterstützt die Hochschule Hamm-Lippstadt dabei, auf Basis der im AP 2 identifizierten Elemente in mehreren Iterationen drei Styleguides zu gestaltet. Diese drei Styleguides wurden für jeweils drei Elementgrößen angepasst, d.h. es existieren insgesamt neun Ausprägungen. Jeder Styleguide enthält das grundlegende Design von allen Elementen aus dem Arbeitspaket *Elemente identifizieren*. Die Styleguides stellen das Ergebnis des AP 5 und 6 dar. In Abbildung 2 ist ein Ausschnitt aus dem Styleguide mit mittlerer Elementgröße abgebildet.

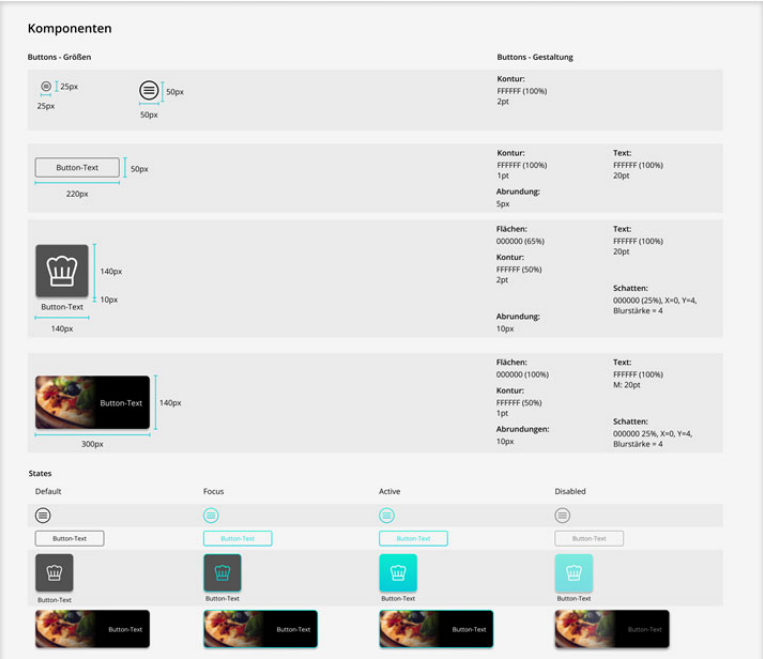

Abbildung 3: Ausschnitt aus dem Styleguide für mittlere Elementgröße

AP 7 & 8: Schnittstellengenerator I + II

Für den Schnittstellengenerator wurden zunächst die identifizierten Elemente aus AP 2 mit dem Elementdesign aus AP 5 und 6 in HTML-Komponenten umgesetzt. Diese können somit als eigene HTML Elemente zur Umsetzung der Bildschirmmasken aus AP 3 genutzt werden. Aufgrund der Vielzahl an Elementen und unterschiedlichen Ausprägungen in Bezug auf die Elementgröße, floss vergleichsweise viel Aufwand in diese Übersetzungsleistung. Insbesondere das CSS-Styling und die Sicherstellung der Kompatibilität zur React als Frontendframework, welches von der Fachhochschule Dortmund für den Prototyp verwendet wurde, stellte sich als zeitintensive Tätigkeit dar.

AP9 & 10: Statisches und Dynamisches Reasoning

Neben dem Design der Benutzungsschnittstellenelemente und deren Zusammensetzung, zeigt sich die QuinScape ebenfalls für die Umsetzung des Reasonings verantwortlich. Grundlage der Umsetzung bildet das im Gesamtkonsortium unter Leitung von acs plus GmbH entwickelte Reasoningkonzept. Dieses sieht vor, die Anpassung auf zwei Ebenen vorzunehmen, die im Folgenden als WIE-Engine und WAS-Engine bezeichnet werden.
Die primär im Kontext des AP9 entwickelte WAS-Engine fokussiert die Fragestellung, welche Funktionen eines Haushaltsgerätes innerhalb einer konkreten Interaktionssituation dem/der Nutzer*in angezeigt werden sollen (*Was* wird dem/der Nutzer*in angezeigt?). Hierzu erfolgt auf Basis von historischen Nutzungsdaten in Kombination mit dem zur jeweiligen Interaktion gegebenen Zeitpunkt eine Wahrscheinlichkeitsberechnung, ob ein*e Nutzer*in in einem bestimmten Zeitfenster eine bestimmte Funktion an einem bestimmten Gerät nutzen will. Die WAS-Engine schlussfolgert also, welche Funktion(en) ein*e Benutzer*in wahrscheinlich gerade nutzen möchte.

Die WIE-Engine wurde anschließend im AP10 entwickelt und befasst sich damit, wie die Darstellung für den/die jeweilige*n Nutzend*n dargestellt werden soll (*Wie* werden Funktionen dem/der Nutzer*in dargestellt?). Das Ziel der WIE-Engine ist eine personenbezogene Optimierung der Darstellung, d.h. sowohl die personenspezifische Erkennung als auch die Maßnahmenergreifung zur Beseitigung von potenziellen Bedienhürden.
Die WIE-Engine durchläuft dabei grundsätzlich vier Schritte zu Beginn jeder Interaktion:

1. **Hypothesenauswahl**
2. **Maßnahmenauswahl**
3. **Bewertung getroffener Maßnahmen**
4. **Update nach Bewertung**

Bei der **Hypothesenauswahl** geht es darum, eine oder mehrere Hypothesen für eine aktuell stattfindende Interaktion zu bestimmen. Eine Hypothese ist in diesem Kontext eine Annahme darüber, ob Nutzende Hürden bei der Gerätebedienung erleben. Bspw. Probleme mit der Lesbarkeit auf dem Gerätebildschirm. Eine Hypothese zur aktuellen Interaktion kann jedoch auch lauten, dass der Nutzende keine Probleme bei der Bedienung hat.

Die zu Projektende fertig implementierten Bedienhürden umfassen mangelnde Lesbarkeit, d.h. der/die Nutzende kann Beschriftungen schwerlich lesen, und Treffsicherheit in der Bedienung, d.h. der/die Nutzende trifft die Bedienelemente in der Interaktion nicht genau. Zu der Bestimmung werden unterschiedliche Metriken aus den vergangenen Interaktionen ausgewertet, die der WIE-Engine von der Plug-In Plattform zur Verfügung gestellt werden. Welche Metriken auf welche Hypothesen einzahlen, ist in der folgenden Tabelle 1 dargestellt.

Tabelle 1: Einfluss von Metriken auf Hypothesen

Hypothese	Metrik	Beschreibung der Metrik
Lesbarkeit	distance_display	Die gemessene Distanz von dem/der Nutzer*in zum Gerät.
	interaction_time	Die Zeit der gesamten Interaktion des/der Nutzenden am Gerät.
	interaction_time_step	Der Median der Zeit, welche der/die Nutzende auf den einzelnen Bildschirmen gebraucht hat.
	interaction_step_count	Die Anzahl der Klicks des/der Nutzenden am Gerät während der gesamten Interaktion.
	back_count	Die Anzahl der Benutzung des Zurück-Knopfs während der gesamten Interaktion.

Treffsicherheit	touch_force	Die Kraft mit der ein*e Nutzende*r durchschnittlich auf den Bildschirm drückt.
	touch_accuracy	Ein Maß, welches angibt, wie nah in die Mitte von Schaltflächen der/die Nutzende durchschnittlich drückt.
	back_count	Die Anzahl der Benutzung des Zurück-Knopfs während der gesamten Interaktion.

Treffen eine oder mehrere Hypothesen zu, werden die ggf. zu ergreifenden Maßnahmen zur Optimierung des Interfaces bestimmt (**Maßnahmenauswahl**). Für die verschiedenen Hypothesen sind Maßnahmen festgelegt, welche für diese getroffen werden können. Die Engine nimmt jedoch immer nur eine Maßnahme auf einmal vor. Dabei bevorzugt das System die Maßnahme, welche den kleinsten Einfluss auf das Interface hat. Innerhalb der Engine wird dieser Einfluss daher als Kosten aufgefasst, die sich aus zwei Werten addieren: Den Userinterfacekosten (*uiKosten*) und den Einflusskosten (*eiKosten*). Die *uiKosten* sind dabei festgelegte Kosten für die individuelle Maßnahme. Die *eiKosten* hingegen sind Folgekosten, welche aus der Umsetzung einer Maßnahme resultieren. Bspw. kann die Maßnahme der Schriftgrößenerhöhung bedingen, dass die Bedienelemente, in denen Schriftzüge vorkommen, ebenfalls vergrößert werden müssen, so dass eine größere Änderung als nur die Schriftgrößenveränderung aus der Maßnahme resultieren würde.

Das entwickelte Verfahren wählt die Maßnahme mit den geringsten Gesamtkosten. Hierdurch ist es möglich, die Anpassung graduell zu gestalten und Nutzende nicht mit großen Änderungen zu überfordern. Sollte die initiale Hypothese sein, dass der/die Nutzer*in ein gutes Nutzungserlebnis hat, erfolgt keine Maßnahmenauswahl. Abbildung 4 zeigt den grundsätzlichen Ablauf zur Auswahl einer Maßnahme.

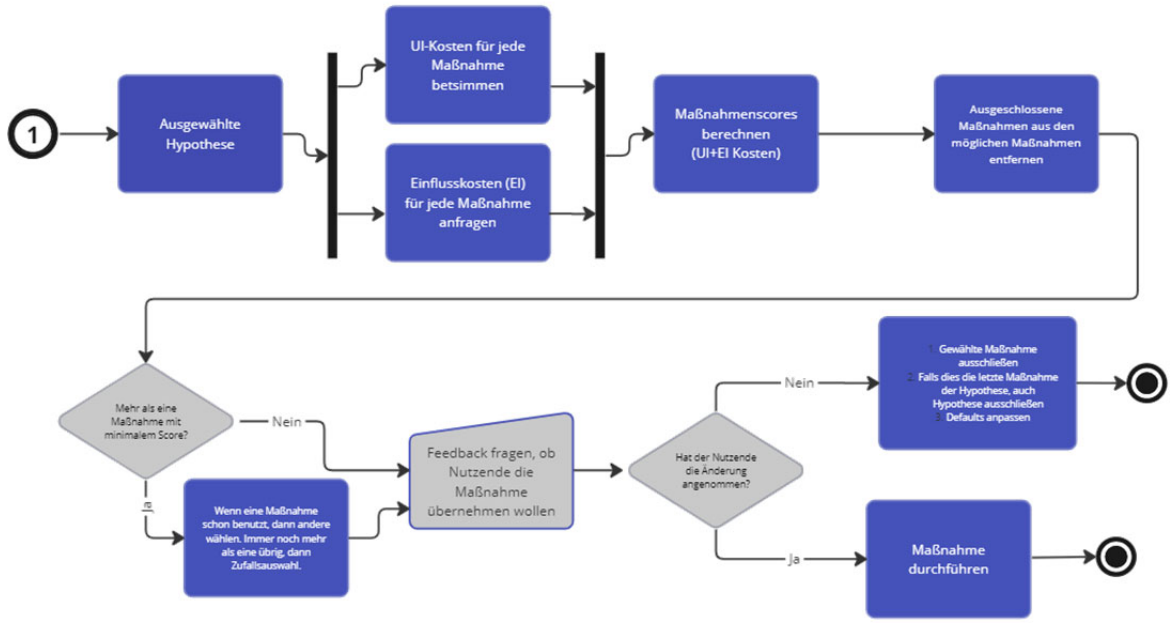

Abbildung 4: Ablauf der Maßnahmenauswahl

Tabelle 2 beschreibt den Zusammenhang zwischen Hypothese und den ggf. daraus folgenden Maßnahmen.

Tabelle 2: Zusammenhang Maßnahmen und Hypothesen

Hypothese	Maßnahme	Beschreibung der Maßnahme
Lesbarkeit	schriftgrösse_ändern	Der Multiplikator für die Schriftgröße wir erhöht oder verringert.
	kontrast_ändern	Den Multiplikator für den Kontrast verstärken oder verringern.
	schriftschnitt	Anpassen, wie dick die Schriftart ist. Die Möglichkeiten sind light, regular, semibold, bold und extrabold.
	icongröße_anpassen	Den Multiplikaor für die Größe von Icons anpassen.
Treffsicherheit	elementgroesse_aendern	Den Multiplikator für die Größe der Elemente (Buttons, Slider,...) anpassen.
	layout	Das Layout für den Nutzenden ändern. Die möglichen Layouts sind klein s, normal m oder groß l.

Innerhalb des **Bewertung getroffener Maßnahmen** Schritts prüft die entwickelte WIE-Engine den Erfolg von ausgewählten Maßnahmen in den Interaktionen nach der Anpassung, d.h.

dieser Schritt erfolgt über einen längeren Zeitraum. Hierbei wird der alte Dringlichkeitswert der Hypothese mit dem Wert verglichen, der nach der Änderung vorliegt.

Die letzte Phase (**Update nach Bewertung**) dient in Kombination mit der vorherigen Bewertungsphase als Korrektiv. Hat die Bedienung des Systems durch den/die Nutzer*in sich verbessert, wird die gewählte Maßnahme als richtige Entscheidung vermerkt. Ist jedoch keine Verbesserung zu erkennen, oder gar eine Verschlechterung, nimmt das System eine weitere Maßnahme vor und wählt ggf. auch eine alternative Hypothese. Bei einer Verschlechterung wird einem/einer Nutzenden außerdem vorgeschlagen, die falsch getroffene Maßnahme rückgängig zu machen.

Insgesamt ist das beschriebene Reasoning, bestehend aus WIE- und WAS-Engine, als zwei separat ausführbare Python Anwendungen entwickelt und der Fachhochschule Dortmund übergeben worden. Die Implementierung ist auf dem durch die Fachhochschule Dortmund bereitgestellten GitHub Projekt des Gesamtforschungsvorhabens verfügbar.

AP11 & AP12: Cloudadaption I + II

Die ursprünglich geplante Cloudadaption bezog sich vor allem auf die Verbesserung des Reasonings durch eine breitere Datenbasis. Aufgrund der Erkenntnisse aus der quantitativen Befragung der Technischen Universität Dortmund wurden die APs 11 und 12 jedoch von der Cloudifizierung ausschließlich bezogen auf Nutzungsdaten bzw. auf das Reasoning, hin zu einer breiteren Auffassung von Cloudifizierung auf insgesamt grundsätzliche Systemfunktionen weiterentwickelt.

Ursächlich ist hier anzuführen, dass die Erkenntnisse der quantitativen Befragung daraufhin deuten, dass bereits die Option, Nutzungsdaten für die Cloud freizugeben, die Nutzungsbereitschafts für das System insgesamt massiv negativ beeinflusst. Dies bestätigte sich auch im Workshop 2 zum Thema Datentransparenz bzw. ELSI Aspekten. Die im Reasoning angewendeten, auf kleinen Datenmengen operierenden Algorithmen weisen zudem eine zufriedenstellende Qualität in der Vorhersage und Darstellungsoptimierung auf, so dass auch technisch eine Cloudifizierung der reinen Nutzungsdaten, um bspw. neuronale Netze einsetzen zu können, nicht nötig erschien und den ermittelten Ansprüchen von potenziellen Nutzer*innen im zweiten Partizipationsworkshop (WS2) des Projektes mit dem Überthema Datentransparenz entgegen liefe. Gleichzeitig bemängelten Expertinnen der assoziierten Geräteherstellers Miele des Projektes und Teilnehmer*innen des WS2, die schwierige Erweiterbarkeit einzelner Systemfunktionen wie bspw. dem Feedbackmechanismus für das Reasoning (bspw. Hinzufügen/Anpassungen von Fragen zur Bewertung der Güte von durchgeführten Anpassungen) und das Hinzufügen von neuen Haushaltsgeräten zum Plug-In System mittels OpenHAB.

Nach Rücksprache mit den Projektpartnern wurde daher im Rahmen der Cloudadaption insbesondere die Cloudifizierung des Plug-In Systems ganzheitlicher in Form eines Baustein-Systems neu konzipiert. Mit diesem neuen Ansatz können theoretisch auch einzelne Reason-

ingschritte, also bspw. die Nutzung von fremden Nutzungsdaten, in der Cloud ausgeführt werden (dies wird nur technisch ermöglicht, aufgrund des oben aufgeführten Nutzerfeedbacks aber nicht für den zu Veröffentlichung gedachten Demonstrator exemplarisch implementiert), es können jedoch vor allem auch neue Geräte und bspw. Einzelfunktionen wie Feedbackmechanismen aus der Cloud in das lokale System integriert werden. Dies geschieht softwaretechnisch in Form von *containerized templating*. Das Konzept wurde gemeinsam mit der FH Dortmund erarbeitet, welche hierzu auch eben entsprechende Veröffentlichung vorgenommen haben.

AP13: Algorithmikadapter

Die Algorithmik des Reasonings wurde im Rahmen des Entwicklungsprozesses iterativ und unter anderem mit dem Einsatz von Jupyter Notebooks realisiert. Die Technologieauswahl erfolgte aus dem Grund, dass entsprechende Frameworks und andere Referenzimplementierungen mit der Programmiersprache Python zu dem Zeitpunkt der Umsetzung sehr weit verbreitet waren. Aufgrund der bereits im Projektverlauf existierenden Python Implementierungen wurden sowohl die Reasoning Engines für die Bestimmung der zu erwartenden Interaktionen des Benutzers (Projektierern als WAS-Fall bezeichnet), als auch die für den Nutzer optimale Beschaffenheit der Bedienschnittstelle (Projektintern als WIE-Fall bezeichnet) entsprechende Softwareartefakte in dieser Programmiersprache entwickelt.

Die unterschiedlichen Implementierungen der Fälle basieren auf einer ähnlichen technologischen Grundlage um Erkenntnisse aus dem jeweiligen anderen Fall besser adaptieren und eine wechselseitige Synergie nutzen zu können. Dennoch sind die im Projekt von QuinScape entwickelten Reasoning Engines vollständig unabhängig voneinander ausführbar. Dies führt zum positiven Nebenergebnis, dass das Reasoning für den weiteren Gebrauch sowie zukünftige Projekte unabhängig wiederverwertbar ist.

Die Fachhochschule Dortmund hat als Hautverantwortliche für das Arbeitspaket 3 das technologische Grundgerüst des lokalen Backends umgesetzt. Der von der Fachhochschule Dortmund initial diskutierte, abgesprochene sowie umgesetzte Technolgiestack ist für jeweils andere Anwendungsfälle optimiert und lässt eine direkte Eingliederung des Reasoning Quellcodes nicht ohne größere Hürden zu. Aus diesem Grund wurde für die Integration der Reasoning Systeme eine servicebasierte Architektur entworfen und realisiert. Das Ziel dieser Architektur ist eine möglichst geringe Kopplung bei einer starken Kohäsion der Reasoning Bestandteile. Für den Zeitraum der Entwicklung wurde QuinScape von der Fachhochschule Dortmund der Quellcode des lokalen Backends zur Verfügung gestellt und laufend mittels einer Versionsverwaltung aktualisiert. So wurde ein ständiger Abgleich der Datenmodelle sichergestellt, sodass ein möglichst reibungsloser Austausch der benötigten Daten im späteren Projektverlauf vorbereitet werden konnte.

Da die Daten für den WAS-Fall (d.h. die Vorhersage des voraussichtlich intendierten Anwendungsfalles) aufgrund der einfacheren Erfassung früher zur Verfügung standen, wurde die

Integration dieses Anwendungsfalles übereinstimmend mit dem Zeitplan zuerst durchgeführt. Hierzu wurden die initial entwickelten Jupyter Notebooks nach dem Überführen in ein eigenständiges Python-Programm in Container Images verpackt. Dies ermöglicht allen Projektpartnern eine Ausführung die unabhängig der von QuinScape verwendeten Technologien möglich ist. Die von der Fachhochschule Dortmund verwendeten Datenmodelle waren während der Entwicklung bereits bekannt. Eine Übertragung der Daten zur Laufzeit wurde bilateral in einem iterativen Prozess definiert. Als Ergebnis der Planungsgespräche wurde vereinbart, dass die Nutzungshistorie des jeweiligen Nutzers und der aktuelle Kontext der Benutzung unmittelbar nach Identifizierung des Nutzers mittels eines REST-Aufrufes an die Reasoning-Engine übertragen wird. Hierzu wurde ein eigenes Datenmodell in Form von Data-Transfer-Objects entwickelt, um einen sogenannten Anti Corruption Layer zu formen. Dies hat für alle beteiligten Seiten den Vorteil, dass die intern verwendeten Datenmodelle zu jeder Zeit beliebig angepasst werden können, ohne dass die jeweils andere Sotwareseite dies ebenfalls tun muss. Als Ergebnis der Anfrage wird eine Liste der wahrscheinlichsten Anwendungsfälle in Abhängigkeit eines Scores, der die mutmaßliche Wahrscheinlichkeit angibt, übermittelt. Ab diesem Zeitpunkt ist der Reasoning-Prozess beendet und die weitere Verarbeitung obliegt den jeweiligen Projektpartnern.

Für die Anpassung des Bedienerlebnis auf die Bedürfnisse des jeweiligen Benutzers sind in der Erhebung kompliziertere Parameter benötigt worden, die teilweise auch auf dem aktiven Feedback von Benutzern beruhen. Es lag nahe ebendiesen Kommunikationsmechanismus auch für die Adaption der Benutzerintefaces zu verwenden. Im Laufe der Entwicklung wurde auf Basis von Erkenntnissen vom Projektkonsortium der Entschluss gefasst, einen für den Anwendungsfall spezialisierten Kommunikationsmechanismus neu zu entwickeln. Dieser sollte die gleichen Vorzüge des Anti Corruption Layers bei gleichzeitiger bidirektionaler Kommunikation bieten. Für die Realisierung wurde daher das gRPC-Framework zur Kommunikation für den WIE-Fall und dem lokalen Backend der Fachhochschule Dortmund erfolgreich adapiert und von QuinScape in die Reasoning-Engine implementiert. Durch eine unabhängig vom Quellcode definierbare Kommunikationsinfrastuktur wurde in mehreren Planungsworkshops ein initialer Mechanismus herausgearbeitet. Durch Einsatz des Framworks konnte sowohl die Fachhochschule Dortmund als auch QuinScape in den jeweils eingesetzten Programmiersprachen entsprechende Schnittstellen automatisch generieren und mit passender Geschäftslogik befüllen. Nachträgliche Anpassungen erforderten daher lediglich eine automatisierte Neugenerierung dieser Stubs. In der ersten Iterationsstufe wurden mittels gRPC erfolgreich Nutzungsmetriken wie der Abstand zum Display oder die Druckstärke an die hierfür von QuinScape entwickelte zweite Reasoning Engine übermittelt. Als Ergebnis des gRPC-Aufrufs wurden die passenden Parameter für das individuelle Nutzerinterface returniert. Aufgrund von Ergebnissen der anderen Projektpartner, sollten im weiteren Verlauf während der Interaktion Nutzer aktiv um Feedback gefragt werden. Dies machte es notwendig ein weiteres Frage-Antwort Protokoll zu entwerfen und in die bereits existente Kommunikationsinfrastruktur einzugliedern. Das Softwaresystem wurde in der Gesamtheit so befähigt, asynchron zu kommunizieren. So kann der Nutzer zu einem beliebigen Zeitpunkt zuvor durch

die Reasoning Engine gestellte Fragen beantworten, die in späteren Vorhersageprozessen passend berücksichtigt werden.

Da jede Veränderung eines Parameters sich unterschiedlich stark auf das zu generierende Userinterface auswirken kann, wurde die Anbindung der Reasoning Engine mit einer weiteren asynchronen gRPC-Methodik ausgestattet, sodass die Komponenten der Fachhochschule Dortmund aktiv nach möglichen Einflusskosten bei einer Veränderung des Interfaces gefragt werden konnten. Dieser Prozess erfolgt just in time während dem Reasoningprozess und startend bei Erkennung des Benutzers.

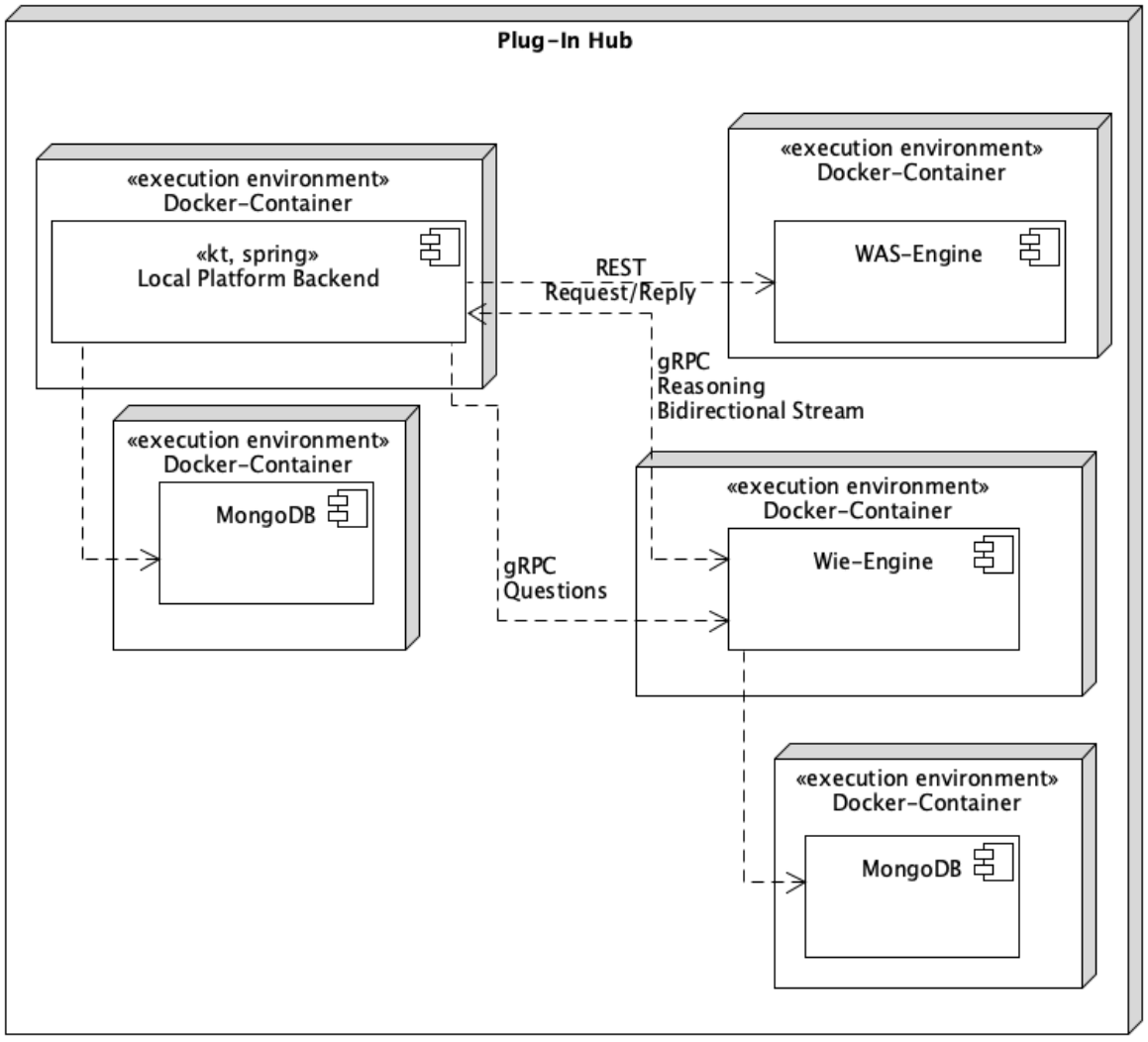

Abbildung 5 Reasoning-Architektur

Während der Entwicklung wurde von QuinScape eine von der Fachhochschule Dortmund zur Verfügung gestellte Versionsverwaltung mit angeschlossenem Buildsystem genutzt. Im Rahmen der iterativen sowie agilen Entwicklung wurde der bereits beschriebene Prozess zur Containerisierung vollständig automatisiert. Nach einer Änderung des Quellcodes wurde somit nach einem testweisen Build und einem Peer Review der Änderungen eine Bereitstellung

von Containerimages beider Reasoning-Engines vollautomatisch durchgeführt. Eine Integration in ein durchgehend laufendes Testsystem wurde ebenfalls durchgeführt.

AP14: Integration Schnittstellengenerator

Die Integration des Schnittstellengenerators besteht aus zwei Teilen. Der erste Teil ist das Backend, welches für die Generierung der Benutzeroberfläche mit Einfluss von Reasoning Daten zuständig. Es besteht aus den unterschiedlichen Teilen des Projekts und verknüpft die Adapter, Services und Datenbanken miteinander. Das Gegenstück zu dem Backend ist das Frontend und ist verantwortlich für die visuelle Darstellung der Benutzeroberfläche.

QuinScape hat als Hauptverantwortliche für das AP 14 die visuelle Integration des Schnittstellengenerator mit Unterstützung der Fachhochschule Dortmund entwickelt. Die Integration erfolgte über eine Electron Anwendung mit dem Webframework React und verwendet die in AP 7 und 8 entwickelten HTML-Komponenten. Diese wurden verwendet, um das Metamodell der Benutzungsschnittstellenbeschreibungssprache umzusetzen. Als Basis verwendet die visuelle Integration eine von dem Schnittstellengenerator generierte JSON. Diese beinhaltet alle auf dem jeweiligen Bildschirm anzuzeigenden Elemente, sowie dessen Meta-, Positions- und Größendaten. Insgesamt wurde bis zum Ende der Projektlaufzeit diese Software auf einem von der Fachhochschule Dortmund entwickelten Prototypen, bestehend aus einem Raspberry Pi mit speziell angefertigter Hardware, verwendet. In Kooperation mit Projektexternen Sozialwissenschaftler*innen der Technischen Universität Dortmund wurde der Prototyp auch außerhalb des Projektkontextes evaluiert. Durch diese Kooperation konnte die visuelle Integration über Workshops hinaus aktiv getestet und entwickelt werden.

AP15: Beispiel Mehrwertservices

Moderne Haushaltsgeräte beinhalten einen großen Anteil an Software. So ist es perspektivisch möglich, existierende und im Feld befindliche Haushaltsgeräte nachträglich mit erweiterten Funktionen im Rahmen der existierenden Hardwarelimitierungen zu versehen.
Gerätehersteller oder auch Drittanbieter von Softwareprodukten haben ein natürliches Interesse daran, nachträgliche Aufwände ggf. zu monetarisieren. Das Plug-In Backend wurde durch ein gemeinschaftlich entwickeltes Konzept dahingehend erweitert, als dass Endbenutzer*innen einzelne Softwaresysteme nachträglich lizensieren können. Mittels eines Lizensierungsservers, welchen die Fachhochschule Dortmund unter Beratung von QuinScape entwickelt hat, können dem Smarthome System dynamisch Features erlaubt werden. Diese Features können wahlweise durch den jeweiligen Schnittstellengenerator des Herstellers oder durch Generatoren von Dritten zur Verfügung gestellt werden.

2.2. Voraussichtlicher Nutzen / Verwertbarkeit der Ergebnisse und Erfahrungen

Wie in der Teilvorhabensbeschreibung dargestellt, ist QuinScape kein Produkthersteller, sondern ein Dienstleistungsunternehmen. Wirtschaftlich liegt der Mehrwert für QuinScape also nicht unmittelbar in der Produktion/Vertrieb von Plug-In Adaptern, sondern vielmehr in der Anwendung des dort erlangten Know-Hows.
Der Trend zur Individualisierung bzw. Personalisierung im Bereich der modernen Softwarelösungen hat sich in den letzten Jahren insb. vor dem Hintergrund des aufstrebenden Bereichs AI/KI weiter verstärkt. Vor diesem Hintergrund konnte QuinScape innerhalb des Projektes durch die Fertigstellung des Designs wertvolle Erfahrungen im Bereich der UI/UX von anpassbaren Oberflächen sammeln, die auch auf Domänen wir Automotive oder Pharmazie übertragen werden können.
Die Erschließung neuer Marktsegmente und Anbahnung von potenziellen Kundenkontakten erhofft sich QuinScape wie in der Vorhabensbeschreibung erklärt, starke Synergien und Impulse aus der Kombination von Data Science für besser UI/UX.
Insbesondere die Übertragbarkeit von persönlichen Interfaces über Gerätegrenzen hinweg scheint zukünftig ein spannender Markt zu sein. Aktuell beziehen sich die Erkenntnisse aus dem Projekt jedoch nur auf den Bereich Haushaltsgeräte.
QuinScape ist ein Unternehmen zu dessen Kundenstamm Unternehmen im DAX 30 Umfeld gehören. Endanwender*innen sind in der Regel eine projektspezifisch bestimmte Zielgruppe, die in der Planungsphase festgelegt wird. Durch die gewonnene Expertise aus den Designworkshops kann das Verständnis von Zielgruppen und den dazugehörigen Bedürfnissen projektübergreifend verbessert werden. Durch die geänderte Methodik und die nun vorhandenen quantitativen Daten erhofft sich QuinScape die Möglichkeit, bestimmte Bedienvorlieben bei adaptiven Interfaces an bestimmte Merkmale von Endanwender*innen koppeln zu können.

3. Literaturverzeichnis

Browne, D. (2014). *Adaptive User Interfaces*. Elsevier Science.

Hubert, M., Blut, M., Brock, C., Zhang, R. W., Koch, V., & Riedl, R. (2019). The influence of acceptance and adoption drivers on smart home usage. *European Journal of Marketing*, *53*(6), 1073–1098. https://doi.org/10.1108/EJM-12-2016-0794

Lavie, T., & Meyer, J. (2010). Benefits and costs of adaptive user interfaces. *International Journal of Human-Computer Studies*, *68*(8), 508–524. https://doi.org/10.1016/j.ijhcs.2010.01.004

Marsá-Maestre, I., López-Carmona, M. A., Velasco, J. R., & Navarro, A. (2008). Mobile Agents for Service Personalization in Smart Environments. *Journal of Networks*, *3*(5), 30–41. https://doi.org/10.4304/jnw.3.5.30-41

Saunders, J., Syrdal, D. S., Koay, K. L., Burke, N., & Dautenhahn, K. (2016). "Teach Me–Show Me"—End-User Personalization of a Smart Home and Companion Robot. *IEEE Transactions on Human-Machine Systems*, *46*(1), 27–40. https://doi.org/10.1109/THMS.2015.2445105

Vaportzis, E., Giatsi Clausen, M., & Gow, A. J. (2017). Older Adults Perceptions of Technology and Barriers to Interacting with Tablet Computers: A Focus Group Study. *Frontiers in Psychology*, *8*, 1687. https://doi.org/10.3389/fpsyg.2017.01687