

User Interfaces for Shape-Changing Devices in Smart Environments

Bachelor's Thesis at the
Media Computing Group
Prof. Dr. Jan Borchers
Computer Science Department
RWTH Aachen University

by
Yorck Immanuel Heilmann

Thesis advisor:
Prof. Dr. Jan Borchers

Second examiner:
Prof. Dr. Ulrik Schroeder

Registration date: 26.06.2026
Submission date: 28.09.2026

Eidesstattliche Versicherung

Declaration of Academic Integrity

Heilmann, Yorck Immanuel

Name, Vorname/Last Name, First Name

Matrikelnummer (freiwillige Angabe)

Student ID Number (optional)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende ~~Arbeit~~/Bachelorarbeit/
~~Masterarbeit~~* mit dem Titel

I hereby declare under penalty of perjury that I have completed the present paper/bachelor's thesis/master's thesis* entitled

User Interfaces for Shape-Changing Devices in Smart Environments

selbstständig und ohne unzulässige fremde Hilfe (insbes. akademisches Ghostwriting) erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt; dies umfasst insbesondere auch Software und Dienste zur Sprach-, Text- und Medienproduktion. Ich erkläre, dass für den Fall, dass die Arbeit in unterschiedlichen Formen eingereicht wird (z.B. elektronisch, gedruckt, geplottet, auf einem Datenträger) alle eingereichten Versionen vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

independently and without unauthorized assistance from third parties (in particular academic ghostwriting). I have not used any other sources or aids than those indicated; this includes in particular software and services for language, text, and media production. In the event that the work is submitted in different formats (e.g. electronically, printed, plotted, on a data carrier), I declare that all the submitted versions are fully identical. I have not previously submitted this work, either in the same or a similar form to an examination body.

Aachen, 25.09.2026

Ort, Datum/City, Date

Unterschrift/Signature

*Nichtzutreffendes bitte streichen/Please delete as appropriate

Belehrung:**Official Notification:****§ 156 StGB: Falsche Versicherung an Eides Statt**

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

§ 156 StGB (German Criminal Code): False Unsworn Declarations

Whosoever before a public authority competent to administer unsworn declarations (including Declarations of Academic Integrity) falsely submits such a declaration or falsely testifies while referring to such a declaration shall be liable to imprisonment for a term not exceeding three years or to a fine.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Strafflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

§ 161 StGB (German Criminal Code): False Unsworn Declarations Due to Negligence

(1) If an individual commits one of the offenses listed in §§ 154 to 156 due to negligence, they are liable to imprisonment for a term not exceeding one year or to a fine.

(2) The offender shall be exempt from liability if they correct their false testimony in time. The provisions of § 158 (2) and (3) shall apply accordingly.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

I have read and understood the above official notification:

Aachen, 25.09.2026

Ort, Datum/City, Date

Unterschrift/Signature

Contents

Abstract	xi
Überblick	xiii
Acknowledgments	xv
Conventions	xvii
1 Introduction	1
1.1 Motivation and Research Gap	1
1.2 Research Question and Overview	4
2 Preliminaries and Related Work	7
2.1 Shape-Changing Interfaces in Smart Environments	7
2.1.1 Application Areas in the Smart Home	8
2.2 Existing Control Platforms, Interfaces, and Research Gap	10
2.2.1 Device-Specific SCI Systems	10
2.2.2 General Smart-Home Platforms	11

2.2.3	Research Gap	12
2.3	Interaction Principles and User Control	13
2.3.1	State-Centric Interaction	13
2.3.2	Transition Awareness	14
2.3.3	Constraint Visibility	14
2.3.4	Levels of User Control	15
2.3.5	Intuitive Use and First-Use Experience	16
3	Research Approach and Design Requirements	17
3.1	Research Model	17
3.1.1	Iterative Study Design	17
3.1.2	Participants and Data Analysis	20
3.2	Initial Exploratory Interviews: User Needs and Design Requirements	21
3.2.1	User Expectations and Problems	22
	Simplicity and Access to Detailed Control	22
	Transparency and Feedback	22
	Rooms and Groups	23
	Guests and Safety	23
	Privacy and Data Use	24
	Conversational Assistance	24
3.2.2	Derived Design Requirements	24
3.2.3	Development of the Interaction Concepts	25

3.3	Iterative Interviews: Design Feedback and Final Interaction Concept	26
3.3.1	Feedback on the Initial Interaction Concept	27
	State-Centric Interaction	27
	Explanation and Preview of States	28
	Transitions, Constraints, and Safety	28
	Consistency Across Heterogeneous Devices	29
	Scalability	29
3.3.2	Final Interaction Concept	30
4	System Design and Implementation	31
4.1	Interaction and User Interface	31
4.1.1	Device Setup and Configuration	33
4.1.2	Control Workflows	34
	Individual Device Control	35
	Room and Group Control	36
4.1.3	Feedback and Recovery	37
	History and Restoration	38
4.2	Domain Model and State Representation	39
4.2.1	State-Based Configurations	39
4.2.2	Status Rules and Control History	41
4.3	System Architecture and Home Assistant Integration	42
4.3.1	Entity Mapping and Capability Restrictions	43

4.3.2	Scene Integration and Automation	44
4.4	Control Execution and Reliability	45
4.4.1	Validation and Execution	45
4.4.2	Error Handling and Limitations	46
4.5	Prototype Scope	47
5	Evaluation	49
5.1	Evaluation Method	49
5.2	Results	50
5.2.1	Navigation and Orientation	50
5.2.2	Configuration and State-Based Control	51
5.2.3	Feedback and Safety Rules	52
5.2.4	History and Deletion	52
6	Discussion	55
6.1	Contributions and Remaining Challenges	55
6.1.1	State-Based Control and Understandability	56
6.1.2	Transparency of States, Actions, and Constraints	57
6.1.3	User Control and Recoverability	58
6.1.4	Shared Control of Heterogeneous SCIs	59
6.2	Limitations of the Study and Prototype	60
6.2.1	Methodological Limitations	60
6.2.2	Technical Limitations and Scope	61

7 Conclusion and Future Work	63
7.1 Future Work	64
A Exploratory Interview Guide	67
A.1 Understanding of SCIs and Their Potential Uses in the Home	67
A.2 Challenges When Adopting and Configuring Devices	68
A.3 Direct, Remote, and Automatic Control of Transformations	68
A.4 Trigger-Action Automations	68
A.5 Adjustability of Physical Properties and Behaviors	69
A.6 Coordinated Control of Multiple SCIs	69
A.7 Sensor Data and Logs for Understanding Device Behavior	69
A.8 Long-Term Adaptation of States and Device Configurations	70
A.9 Safety Constraints, Perceived Control, and Guest Access	70
A.10 Conversational Assistance	70
A.11 Structure of the SCI Control Interface	71
B Concept Interview Guide	73
B.1 Initial Impression and Comprehensibility	73
B.2 Tasks	74
B.2.1 Adding a New Shape-Changing Device	74
B.2.2 Locating a Specific Device and Function	74
B.2.3 Modifying a Device State	75
B.2.4 Responding to a Blocked Action	75

B.3	State-Centric Interaction	75
B.4	Explanation and Preview of States	75
B.5	Transitions, Constraints, and Safety	76
B.6	Consistency Across Heterogeneous Devices	76
B.7	Scalability and Long-Term Use	76
B.8	Supplementary Concepts	77
B.9	Concluding Question	77
C	Prototype Evaluation Guide	79
C.1	Prerequisites	79
C.2	Evaluation Tasks	80
D	Design Sketches	83
E	Prototype Screenshots	93
	Bibliography	99
	Index	103

List of Figures and Tables

2.1	Examples of shape-changing interfaces	9
3.1	Iterative research and development process	18
3.2	Conventional and 3D room-view concepts	19
4.1	LineFORM's construction and lamp application	32
4.2	Navigation through the prototype views	33
4.3	System architecture and component responsibilities	42
D.1	Comparison of the general overview sketches	84
D.2	Comparison of the room-state sketches	85
D.3	Comparison of the individual room-control sketches	86
D.4	Comparison of the stationary device sketches	87
D.5	Comparison of the individual device-control sketches	88
D.6	Comparison of the transition sketches	89
D.7	Comparison of the blocked-action sketches	90
D.8	Shared information and history view	91

D.9	Comparison of the conversational-assistant sketches	92
E.1	Prototype room overview	94
E.2	Devices within a room	94
E.3	Device states and individual settings	95
E.4	Device setup and entity mapping	95
E.5	Status-rule configuration	96
E.6	Confirm device action	96
E.7	Combined room control	97
E.8	History and restoration	97

Abstract

Shape-changing interfaces (SCIs) can adapt physical properties such as form, orientation, or volume and thereby extend smart homes beyond conventional devices. However, their diversity also makes them difficult to control through a shared interface: devices often expose many different parameters, transitions have consequences in physical space, and users must understand both the intended result of an action and the reasons why it may be unavailable.

This thesis investigates how user interfaces for heterogeneous SCIs in smart-home environments can support understandability, transparency, and user control. We followed an iterative, user-centered process, beginning with exploratory interviews to identify how prospective users expected to control SCIs and which concerns arose in smart-home environments. Based on these findings, we created two sets of low-fidelity interface sketches and discussed them in a second interview phase to refine the requirements. Building on this concept, we developed a Home Assistant add-on that lets users control SCIs through named states and preview the resulting configurations. It also helps them coordinate multiple devices, understand why certain actions are unavailable, and undo unwanted changes.

The evaluation indicates that named states provided an understandable starting point for controlling heterogeneous SCIs without requiring users to interpret every parameter. Participants nevertheless valued direct access to selected settings for fine-tuning and exploration. Blocking messages helped participants understand restrictions, while active-state indicators and restoration functions helped them follow and reverse changes. At the same time, device setup and rule configuration remained demanding as soon as interface terms were too technical. Participants also had difficulty interpreting the amount of detail shown in the history.

These findings suggest that shared SCI control should combine state-based abstraction with progressively accessible parameters, contextual explanations, and reversible actions. This thesis contributes empirically grounded design requirements, an interaction concept, a functional prototype, and design implications for future SCI control interfaces.

Überblick

Shape-Changing Interfaces (SCIs) können ihre Form, Ausrichtung oder Volumen verändern und erweitern somit das typische Smart-Home. Gleichzeitig erschwert die Vielfalt dieser Geräte eine einheitliche Steuerung: Geräte stellen oft viele verschiedene Parameter bereit, Übergänge haben Auswirkungen im physischen Raum, und Nutzer müssen sowohl das Ergebnis als auch die Gründe verstehen, warum Aktionen möglicherweise nicht verfügbar sind.

Diese Arbeit untersucht, wie Benutzeroberflächen für heterogene SCIs gestaltet werden können, damit ihre Bedienung verständlich, transparent und benutzerorientiert bleibt. Den Ausgangspunkt bildeten explorative Interviews, um die Erwartungen und Bedenken von Benutzern zu verstehen. Mithilfe dieser Erkenntnisse erstellten wir zwei Varianten von einfachen Oberflächen, die in einer zweiten Interviewphase diskutiert und weiterentwickelt wurden. Im Anschluss entwickelten wir ein Home-Assistant-Add-on, mit dem Benutzer SCIs über Zustände steuern und über eine Vorschau angucken können. Darüber hinaus erlaubt das Add-on mehrere Geräte zu koordinieren, unerwünschte Änderungen rückgängig zu machen und zeigt, warum bestimmte Aktionen nicht verfügbar sind.

Die Evaluation zeigte, dass benannte Zustände einen verständlichen Einstieg in die Steuerung unterschiedlicher SCIs ermöglichen, ohne dass jeder einzelne Parameter bekannt sein muss. Für gezielte Anpassungen und zum Ausprobieren blieb der direkte Zugriff auf ausgewählte Einstellungen dennoch wichtig. Sperrmeldungen halfen den Teilnehmern, Einschränkungen zu verstehen, während eine Historie mit Wiederherstellungsfunktionen ihnen dabei half Änderungen nachzuvollziehen und rückgängig zu machen. Schwierigkeiten entstanden vor allem bei der Einrichtung von Geräten und Regeln, sobald die verwendeten Begriffe technisch wurden. Auch der hohe Umfang der Historie erwies sich als kompliziert.

Insgesamt legen die Ergebnisse nahe, dass eine gemeinsame SCI-Steuerung verständliche Zustände mit zugänglichen Detaileinstellungen, kontextbezogenen Erklärungen und Möglichkeiten zum Rückgängigmachen kombinieren sollte. Die Arbeit liefert empirisch fundierte Gestaltungsanforderungen, ein Interaktionskonzept, einen funktionsfähigen Prototyp und Empfehlungen für die Gestaltung zukünftiger SCI-Steuerungen.

Acknowledgments

I would first like to thank Prof. Dr. Jan Borchers and Prof. Dr. Ulrik Schroeder for examining this thesis and for giving me the opportunity to pursue this topic at the Media Computing Group.

Sarah, thank you for your guidance, thoughtful feedback, and the many discussions throughout this work. Your support helped me shape the research, refine the interaction concept, and bring the prototype and thesis together.

To everyone who participated in the interviews and the prototype evaluation, thank you for your time, openness, and detailed feedback. Your contributions were an essential part of this thesis and helped shape both the system and its evaluation.

Finally, to my family and friends: thank you for your patience, encouragement, and support throughout my studies, especially during the final month of this thesis.

Conventions

Throughout this thesis we use the following conventions:

- The thesis is written in American English.
- The first person is written in plural form.
- Unidentified third persons are described in female form.

Short excursuses are set off in colored boxes.

EXCURSUS:

Excursuses are set off in orange boxes.

Where appropriate, paragraphs are summarized by one or two sentences that are positioned at the margin of the page.

This is a summary of a paragraph.

Chapter 1

Introduction

Shape-changing interfaces (SCIs) are an emerging technology with potential applications in smart homes. Their adaptable physical behavior can extend established forms of connected control and automation. This thesis explores this potential by integrating heterogeneous SCIs into a smart-home platform and developing a shared interface for their configuration and control.

1.1 Motivation and Research Gap

Smart homes connect sensors and appliances to provide convenient control and support daily living [Sovacool and Furszyfer Del Rio, 2020]. Currently, most smart-home devices remain stationary and only change their digital states. SCIs, by contrast, can alter physical properties such as form, volume, orientation, or texture. This versatility enables a much wider spectrum of applications. It allows SCIs to serve as inputs and outputs as well as communicate not only information but also emotions through their physical environment [Alexander et al., 2018; Rasmussen et al., 2012].

Shape-changing
interfaces extend
physically static
smart-home devices.

SCIs create new opportunities for adaptable and physically expressive environments.

These possibilities are illustrated by existing SCI systems. For example, Togler et al. [2009] designed the Thrifty Faucet. It can change its posture to communicate feedback about water consumption and hygiene and demonstrates how shape change can embed persuasive feedback into the household. Additionally, participatory research by Hübner [2025] identified envisioned smart-home applications. Ideas ranged from adaptable furniture and environmental control to safety functions and multifunctional objects. Complementary research showed that SCIs are able to guide physical interaction and manipulate surrounding objects in the environment [Follmer et al., 2013]. SCIs can therefore perform practical functions and make otherwise abstract system states physically observable.

Shared SCIs must accommodate different users and spatial dependencies.

Smart-home environments are particularly relevant for SCIs because devices are embedded in shared spaces. An SCI does not need to serve the same purpose for every household member. Through different physical configurations, the same device can support distinct activities or assume different roles depending on the situation. Previous research argues that smart-home control should support domestic routines and relationships rather than focus exclusively on individual devices [Davidoff et al., 2006]. Long-term deployments further reveal difficulties caused by fragmented control [Brush et al., 2011]. These challenges become more consequential for SCIs because their actions produce visible and spatial changes in a shared environment. The home therefore provides both a meaningful application area and a demanding context for investigating adaptable SCI control.

Heterogeneous device capabilities make SCI control difficult to generalize.

Beyond differences in how a single SCI may be used, the capabilities of different SCIs can also vary substantially as they are not confined to a standardized form factor. Most SCIs differ in terms of physical construction, actuation mechanisms, transformation types, and exposed inputs and outputs [Rasmussen et al., 2012; Sturdee and Alexander, 2018]. While one device may provide multiple angle and speed settings, another may require an entirely different representation. Consequently, transferring the exposed parameters directly into a conventional smart-home user interface makes it highly complex and difficult

to operate. In the end, this creates an abstraction gap between the user's intent, such as preparing a room for relaxation, and the device-specific parameters required to realize it [Alexander et al., 2018; Omar, 2026].

One way to possibly bridge this gap is to combine low-level parameters into meaningful configurations like "Relax" or "Working." This approach replaces adjusting and understanding every setting with an outcome representing the user's goal. This activity-oriented approach aligns with Davidoff et al. [2006], who argue that smart-home control should support users' routines instead of concentrating exclusively on device control. Other research demonstrates the importance of clearly communicating the relationship between physical configurations and their underlying system states [Tiab and Hornbæk, 2016]. Nevertheless, predefined states are insufficient in certain situations: users may need to adapt or define their own personal states and intervene if a previously configured state produces unwanted results. SCI control must therefore not exclusively use one approach, but introduce a balance between abstraction and parameterized control.

Meaningful states
reduce complexity while
detailed controls
preserve flexibility.

Beyond representing target configurations, SCIs may be obstructed by nearby objects or ongoing activities as they move in physical space. This leads to constraints that must temporarily restrict otherwise valid transformations. Consequently, the interface must communicate unavailable actions to the user. Explaining the reasons why an action is blocked and cannot be executed follows established Human-Computer Interaction work on feedforward and system intelligibility [Lim et al., 2009; Vermeulen et al., 2013]. Furthermore, users ideally need to intuitively understand which position the SCI ends up in when selecting a certain state, as well as the transition leading to it. Prior work shows that the speed and form of a transition, as well as the relationship between user action and device movement, can substantially affect how shape change is experienced and understood [Rasmussen et al., 2016; Tiab and Hornbæk, 2016].

Physical transitions and
contextual constraints
must be
understandable.

Existing SCI research already provides taxonomies, interaction concepts, and numerous device-specific pro-

Existing work does not establish a shared control model for heterogeneous SCIs.

totypes [Rasmussen et al., 2012; Sturdee and Alexander, 2018]. Other work has investigated application ideas for SCIs, including applications for augmented living [Sturdee et al., 2015] and approaches for translating contextual information into device behavior [Omar, 2026]. However, the reviewed literature contains very little research on controlling multiple heterogeneous SCIs from one platform. Existing systems generally focus on individual artifacts and specific transformation technologies. General smart-home platforms can integrate heterogeneous entities. Unfortunately, their controls do not represent SCI states, physical transitions, contextual constraints, or spatial consequences [Alexander et al., 2018; Sturdee and Alexander, 2018].

1.2 Research Question and Overview

Focus on understandability, transparency, and user control.

Based on the described challenges, this thesis investigates the following research question:

RQ: How can user interfaces for shape-changing devices in smart-home environments be designed to support understandability, transparency, and user control?

Here, understandability concerns available actions and their physical effects, transparency covers system states and outcomes, and user control refers to choosing an appropriate level of detail.

Excursus:
Home Assistant

HOME ASSISTANT:

Home Assistant is an open-source home automation platform that integrates devices from different manufacturers and enables their monitoring, control, and automation through a shared interface.

The prototype integrates SCIs into the Home Assistant ecosystem.

This thesis addresses this gap by designing a web-based control platform that integrates with Home Assistant. Our prototype combines configurable device settings with meaningful device states, status rules, and a reversible con-

trol history. Its practical value lies in providing consistent interaction concepts across multiple heterogeneous SCIs within one application.

The scope of this thesis is the user-facing configuration and control of SCIs. Rather than standardizing communication across different SCIs, it models them through settings that can be directly linked to compatible Home Assistant entities. This approach allows us to study shared interaction concepts across different SCIs by connecting them to an established smart-home platform.

To answer the research question, this thesis follows an iterative, user-centered process. We firstly conduct initial exploratory interviews to investigate how prospective users expect to control SCIs and which concerns arise when such devices are placed in smart-home environments. Afterward, we translate the findings into design requirements and an initial interaction concept. In a second round of interviews, we discuss and refine this concept with participants. We then implement the resulting concept as a functional prototype. Finally, we examine whether the resulting prototype supports the intended qualities and identify remaining usability problems and trade-offs.

Our evaluation suggests that the implemented interface supports understandable control of heterogeneous SCIs by organizing settings into meaningful device and room states. Participants valued these states as a way to apply intended outcomes. Individual settings remained accessible when predefined configurations did not meet users' needs. The reversible history provided a basis for exploring such adjustments: users could try different configurations, inspect their changes, and restore previous settings if needed. However, setting up these controls remained challenging because participants did not always understand which device settings the Home Assistant entities represented. Technical terminology also made it difficult to configure blocking rules. These findings support combining state-based control with individual adjustments and recovery, while highlighting the need to make configuration even more accessible.

The work examines the control layer rather than standardizing communication.

A user-centered process connects requirements, design, implementation, and evaluation.

State-based control and recovery support exploration, while configuration remains challenging.

Contributions towards requirements, interaction concepts, a prototype, and design implications.

The work makes four primary contributions:

- a set of empirically grounded requirements for controlling shape-changing devices in smart-home environments;
- an interaction concept combining state-centric control, transition awareness, and constraint visibility;
- a control platform that integrates heterogeneous SCI configurations with Home Assistant and supports device, group, room, rule, and history functionalities;
- an evaluation of the prototype together with implications for the design of future SCI control interfaces.

Chapters progress from foundations to design, implementation, evaluation, and implications.

The remainder of this thesis is structured as follows. Chapter 2 reviews relevant SCI and smart-home research. Chapter 3 presents the research process and derived requirements, while Chapter 4 describes the resulting system. Chapter 5 reports the evaluation, Chapter 6 discusses its findings and limitations, and Chapter 7 concludes and outlines future work.

Chapter 2

Preliminaries and Related Work

This chapter covers the conceptual and empirical basis for controlling SCIs in smart environments. We first define SCIs and identify the properties that control systems must represent. Afterward, we review device-specific SCI systems and well-known smart-home platforms to identify the gap this thesis targets. Finally, we derive certain interaction principles to make heterogeneous shape-changing devices understandable and controllable.

2.1 Shape-Changing Interfaces in Smart Environments

Unlike most interfaces, SCIs use physical changes and their materiality as input and output [Alexander et al., 2018]. As a result, they can cover a wide spectrum of displayed information and convey meaning or emotion by transforming into specific shapes. Depending on the interaction, these transformations may be initiated by the user, negotiated with the user, or controlled by the system [Rasmussen et al., 2016].

SCIs make
computational states
physically perceivable
and manipulable.

Shape change includes more than geometric deformation.

Based on an analysis of 44 systems, Rasmussen et al. [2012] classify shape changes into eight different categories: orientation, form, volume, texture, viscosity, spatiality, material addition or subtraction, and permeability. Later, Sturdee and Alexander [2018] add material- and construction-based categories such as bendable, inflatable, actuated, liquid, and hybrid interfaces. Together, these taxonomies demonstrate the diversity of device constructions a general platform must include.

Control must cover state, transition, capability, and context.

In addition to the final *state*, which represents a stable configuration and meaning, a *transition* may itself convey information as the device changes between states. Its *capabilities* define exposed parameters and value ranges, whereas its *context* captures spatial, situational, and safety-related conditions. Altogether, these properties describe what a device is currently doing, what it is capable of, and why a requested change may be restricted or blocked.

2.1.1 Application Areas in the Smart Home

Domestic SCIs can adapt spaces, objects, information, and behavior.

Especially at home SCIs could play a big role in the future, adding introducing versatile, emotion-conveying objects into daily life. The Thrifty Faucet, for example, transformed a faucet into an expressive interface, giving feedback like reward or denial through different postures [Togler et al., 2009]. Alexander et al. [2018] also address other purposes, including adapting affordances and physicalizing information. In a home context, these purposes may translate into furniture that adapts to different activities or people, and physical controls that appear only when needed.

inFORM, LineFORM and Zooids as examples.

inFORM, LineFORM, and Zooids illustrate how shape change can convey information and adapt an object's functionality. Although these systems were not designed specifically for smart homes, they demonstrate capabilities relevant to this context. inFORM uses a grid of pins that move up and down independently. These pins can form physical controls and offset objects placed on the surface [Follmer et al., 2013]. LineFORM consists of connected motorized

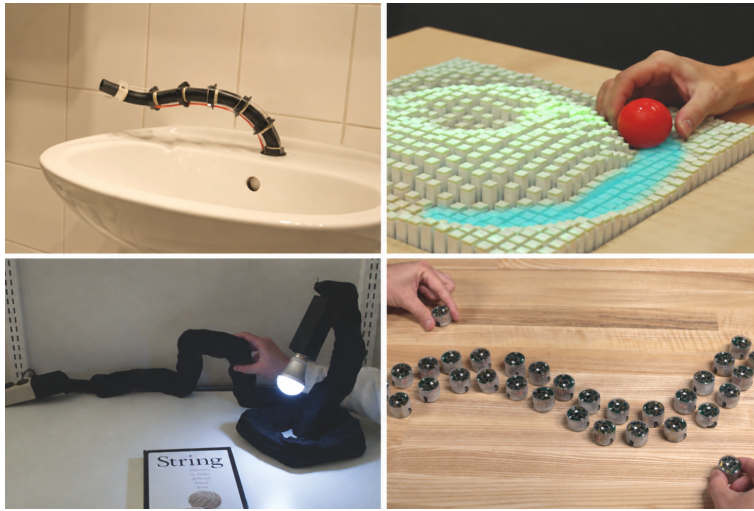


Figure 2.1: Examples of shape-changing interfaces: the Thrifty Faucet (top left), inFORM (top right), LineFORM (bottom left), and Zooids (bottom right). Images adapted from Follmer et al. [2013]; Le Goc et al. [2016]; Nakagaki et al. [2015]; Togler et al. [2009].

segments that bend into different shapes. It can display information, serve as a control, or restrict motion [Nakagaki et al., 2015]. Zooids consists of small mobile robots representing information. Users can rearrange these robots to provide input [Le Goc et al., 2016].

Hübner [2025] analyzed how people imagine SCIs in smart homes. To investigate this, participants were asked to propose ideas ranging from injury-preventing objects to adapting whole rooms to different situations. Participants also liked multifunctional design and clear mapping, such as a lampshade that grows proportionally to its brightness. Such changes can add visible status information to already established smart-home benefits like convenience, safety, and energy management [Sovacool and Furszyfer Del Rio, 2020].

Chamunorwa et al. [2024] demonstrate smart-home controls that are embedded in familiar objects like a vase, a plant, or a pillow. On the one hand, this can support overall aesthetics and routines; on the other hand, the added

User scenarios
emphasize
environmental control,
safety, and
multifunctionality.

Embedded controls
must remain
discoverable and easy
to use.

controls must remain discoverable. Therefore, the control should not interfere with the object’s primary purpose, but add to it. For a general SCI platform, this means providing a consistent interface across different devices while representing specific controls and their physical effects.

2.2 Existing Control Platforms, Interfaces, and Research Gap

Existing control approaches either target the mechanisms of individual SCIs or provide generic abstractions for heterogeneous smart-home devices. Comparing both perspectives reveals which aspects of SCI control are already supported and where a shared interaction model is still missing.

2.2.1 Device-Specific SCI Systems

Existing SCI controls
are tied to particular
hardware.

The reviewed prototypes describe control only in regard to their own mechanisms. While inFORM changes the height of hundreds of individual pins, LineFORM configures multiple joints, and Zooids changes the position of separate mobile elements [Follmer et al., 2013; Le Goc et al., 2016; Nakagaki et al., 2015]. Each command developed for one system cannot be transferred to another because their capabilities and context differ too much. Alexander et al. [2018] identify this problem and suggest cross-platform software and tools for end-user programming as an important next step for SCI research.

Emerging middleware
abstracts generation,
but still requires
device-specific
integration.

In his master’s thesis, Omar [2026] approached part of this abstraction problem by presenting Notiflex. It translates a high-level notification context into hardware-specific shape-change parameters and demonstrates the value of separating semantic intent from low-level actuation. However, Notiflex’s main purpose is to generate notifications, and still requires individual code for each device to interpret the generated parameters. The main distinction is that

Notiflex generates shape-change behavior for notifications, whereas this thesis develops a shared interface through which users can configure and control heterogeneous SCIs.

2.2.2 General Smart-Home Platforms

Outside the SCI space, smart-home platforms already integrate heterogeneous devices from different manufacturers. Two established platforms are Home Assistant and openHAB. The [Home Assistant documentation](https://www.home-assistant.io/docs/)¹ shows how device functions are represented as standardized entities and how the platform provides shared dashboards, scenes, custom scripts, and automations. Similarly, the [openHAB documentation](https://www.openhab.org/docs/tutorial/model.html)² describes it separates physical *Things* from user-facing *Items* and organizes them by location. Because these abstractions already hide communication protocols, provide consistent controls for heterogeneous devices, and optimally extend end-users' current smart homes, they provide a good foundation for this thesis. However, as already mentioned, they reach their limits when a device's physical configuration and context become part of the interaction.

Smart-home platforms
standardize
heterogeneous devices.

Dashboards in Home Assistant (or openHAB) can control exposed capabilities and react to predefined contextual restrictions but do not describe meaningful physical states, transitions, or overall spatial constraints with regard to SCIs. Interoperability is therefore provided at the entity level, while the interpretation of physical shape change still remains device-specific.

Entity models do not
capture the semantics
of shape change.

Automation systems add a second control layer. Trigger-action programming, for example, can express many desired smart-home behaviors that even inexperienced users can learn relatively easily [Ur et al., 2014]. Nevertheless, users must still translate their intentions into explicit conditions and device-level actions, which requires technical knowledge that cannot be assumed of the average user.

Automation increases
expressive power and
configuration
complexity.

¹ <https://www.home-assistant.io/docs/>, last accessed September 28, 2026

² <https://www.openhab.org/docs/tutorial/model.html>, last accessed September 28, 2026

Brush et al. [2011] reported additional barriers including poor manageability, unreliable behavior, and the effort to understand interconnected devices and parameters. Since SCIs often cause visible movement that takes up more space, these problems become more serious. Consequently, users must know if a change could interfere with nearby objects or people before applying it.

2.2.3 Research Gap

No reviewed system
combines
heterogeneous SCI
control with a shared
end-user interaction
model.

In conclusion, the reviewed systems address different parts of the control problem. At the hardware layer, each prototype, inFORM, LineFORM, and Zooids, provides controls specifically tailored to their actuation technology [Follmer et al., 2013; Le Goc et al., 2016; Nakagaki et al., 2015]. Smart-home platforms like Home Assistant or openHAB already tackle this abstraction problem by generalizing controls across devices and manufacturers. However, these platforms are not built to represent devices that change their physical shape. A standard device icon alone does not show the resulting configuration or the space it occupies. Also, users cannot ensure easily that each transition executes only when the context allows it. Notiflex addresses part of this problem by translating notification context into shape-change parameters, but is not designed for interactive home control [Omar, 2026]. In the reviewed literature, no platform allows users to create coordinated states across different SCIs while reliably explaining and preventing blocked or unavailable actions.

This thesis contributes
the missing interaction
and orchestration layer.

In this thesis, we address this gap by using the Home Assistant ecosystem and building upon it. We map device-specific entities into configurable SCI settings and group them into meaningful states. Afterward, we build a shared interaction layer supporting room and group orchestration, status rules, transition-aware interaction, and constraint communication. To achieve this, we use interviews and a prototype evaluation to investigate how to present the physical complexity of heterogeneous SCIs to end-users without requiring technical knowledge.

2.3 Interaction Principles and User Control

To achieve the goal of making heterogeneous SCIs accessible to non-technical users, we focus on three core interaction concepts: state-centric interaction, transition awareness, and constraint visibility. We additionally consider adaptable levels of user control and the first-use experience.”

2.3.1 State-Centric Interaction

The first concept we use is state-centric interaction. It refers to controlling multiple individual device parameters through meaningful configurations, building on activity- and routine-based smart-home control. It aligns with the arguments from Davidoff et al. [2006] that households seek control over activities and routines and do not want to be confronted with isolated devices and their controls. Similarly, Fogli et al. [2017] describe scenes as sets of actions across multiple devices that abstract hardware-specific controls with high-level commands. Applied to SCIs, multiple physical properties, including position or orientation, are combined into named states such as “Working” or “Relaxing,” forming a recognizable configuration that directly reflects the user’s intent.

States combine device settings into activity-related configurations.

At the same time, named states should not fully replace access to individual hardware-specific parameters. Zaidi et al. [2026] show that users commonly need to switch between centralized and direct control depending on convenience, time, and required precision. State-centric interfaces, therefore, are not enough and must combine efficient high-level control with access to detailed settings.

Abstraction must remain inspectable and editable.

2.3.2 Transition Awareness

Transition awareness connects intended states with observable physical changes.

Unlike instantaneous graphical updates, SCI transitions unfold over time and can affect nearby people and objects. Our second concept, transition awareness, concerns both aspects and addresses the final configuration as well as feedback during execution. Petersen et al. [2020] show that the perceived movement and its meaning are influenced by properties such as speed or trajectory. Furthermore, Tiab and Hornbæk [2016] show that fully working transitions can be interpreted as malfunctions if they are unclear. Before activation, the user therefore needs clear feedback on which target state will be reached and why a transition starts, completes, or fails to execute correctly.

Reversible actions and confirmations increase safety and encourage exploration.

To increase perceived control, users should be allowed to reverse actions by restoring a previous configuration. According to Shneiderman et al. [2016], reversible actions reduce anxiety and encourage exploration by making unintended actions easy to correct. For high-risk actions, the user should also confirm the action to reduce the likelihood of unintended or unsafe actions [Sattar and Dudek, 2011].

2.3.3 Constraint Visibility

Capabilities need to be restrictable depending on the environment.

SCIs may be placed in different locations throughout the home, each with its own spatial constraints. This implies that valid ranges must be configurable and specific to the current environment and context rather than being based only on the device's capabilities [Alexander et al., 2018; Rasmussen et al., 2012].

Constraints should explain what is possible before an action is attempted.

Moreover, persons or misplaced objects could occupy the needed space, preventing an otherwise possible action. According to Vermeulen et al. [2013], interfaces should therefore not only report errors afterward, but foresee and account for consequences before execution. In terms of SCI control, unavailable states must be disabled to prevent dangerous behavior.

Lim et al. [2009] found that the intelligibility of context-aware systems can be improved through simple “why” and “why not” explanations. A message like “The rear distance sensor indicates that the sofa is too close to the wall. Please move it away.” is therefore more useful than a generic failure. It connects the requested action to the sensed context and indicates how to resolve the problem.

Explanations turn blocked or modified actions into intelligible system behavior.

Making constraints visible clarifies which actions are currently possible, but it does not determine who should initiate them. This leads to the related question of how control should be distributed between the user and the system.

2.3.4 Levels of User Control

Rasmussen et al. [2016] separate SCI control into four different levels: *direct control*, *negotiated control*, *indirect control*, and *system control*. If a user explicitly causes the change, *direct control* takes place, while *system control* describes an automatically acting device. *Negotiated control* distributes the decisions between the user and the system, and *indirect control* directly observes the user’s behavior and infers appropriate changes. Current SCI research mainly focuses on direct and system control, while examples of *indirect* and *negotiated control* remain less common.

SCI control ranges from direct manipulation to autonomous behavior.

During daily life, interactions with objects often require different levels of control. Direct interaction supports quick adjustments, while negotiated behavior can account for safety and spatial constraints, and autonomous system control is well suited for routines. Studies of MoLux revealed that most participants preferred negotiated interaction, while broader smart-home research indicates that the control type depends heavily on the task and perceived reliability [Sørensen et al., 2022; Zaidi et al., 2026]. Negotiated control may therefore provide a useful default, but users should be able to switch control modes as their context and needs may change. However, such flexibility is only useful if users can understand the available forms of control from their first interaction.

Control levels should remain adaptable.

2.3.5 Intuitive Use and First-Use Experience

First-use interaction
should build on prior
knowledge.

Because SCI behavior is unfamiliar, users may not immediately understand how digital commands translate into physical transformations. Fischer et al. [2015] characterize intuitive use as the transfer of prior knowledge to a new device or interface. Conventional navigation, named activities, and previews can therefore reduce the effort required to learn the interface and let users focus solely on the SCI's unfamiliar physical behavior.

The principles define
the design space of this
thesis.

Together, state-centric interaction, transition awareness, constraint visibility, adaptable control levels, and first-use support provide the conceptual basis for the platform developed and evaluated in the following chapters.

Chapter 3

Research Approach and Design Requirements

To identify design requirements and develop the interaction concepts described above, we followed an iterative, user-centered approach. We used initial exploratory interviews to derive a first set of requirements before conducting a second interview phase to refine the overall concept. The prototype that resulted from this process is assessed separately in Chapter 5.

3.1 Research Model

This section first outlines the iterative study design and subsequently summarizes the participants and data-analysis procedure across the interview phases.

3.1.1 Iterative Study Design

There are only a few established conventions regarding the control of SCIs in smart homes. To better understand the needs for this type of control, we therefore decided to use a user-centered approach across three stages. Here, we

The research process connected empirical findings with successive design decisions.

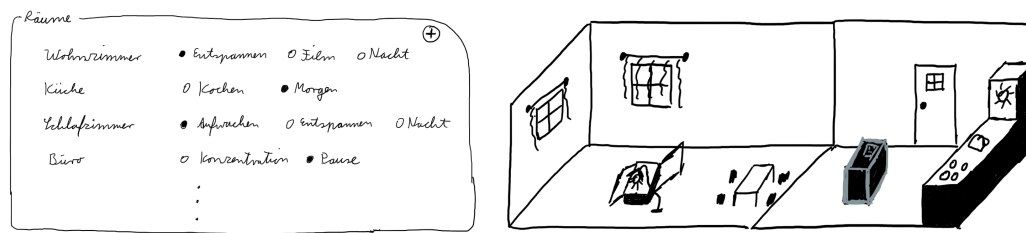


Figure 3.2: Two room-view concepts presented during the second interview phase. The conventional control concept lists rooms and their available states (left), while the alternative concept places devices within a spatial 3D representation of the room (right).

Based on the identified requirements, we created two sets of low-fidelity sketches. They overlapped partly but focused on different concepts: The first concentrated on conventional controls displaying meaningful states alongside checkboxes and sliders. The second set of sketches incorporated visual representations more extensively by operating on 3D models. Figure 3.2 illustrates this difference by comparing the conventional room view with the spatial 3D concept. Both versions were then presented to participants in the second interview phase. In this iteration, we asked participants to express their first impressions of the sketches. They had to describe the process for certain functions, like applying a state or deleting a room. In addition, they commented on consistency, scalability, and long-term use of the sketched interface. This allowed us to look at the individual interface elements and assess the importance of state-centric control, visualization, and access to detailed settings. Having obtained this feedback, we revised the concepts before implementing the prototype.

The second interview presented two sketch sets for better insights.

However, not all suggestions were implemented as-is. We compared statements from all participants. Afterward, we grouped them into common needs, concerns, and preferences. Lastly, we combined these findings with the reviewed literature in Chapter 2 to derive principles, accounting for the scope of this thesis and technical possibilities.

Statements were translated into general requirements rather than direct feature requests.

After refining the interaction concept based on the second interview phase, we implemented it as a functional prototype. The third and final phase then evaluated this proto-

The third phase evaluated the resulting prototype.

type. While the first two phases shaped and refined the design, the final phase assessed the resulting interface. Specifically, we examined whether it supported understandability, transparency, and user control during practical use. The concrete procedure and results of this phase are presented in Chapter 5.

3.1.2 Participants and Data Analysis

Purposive sampling
allowed for a variety of
insights.

In the first phase of the interviews, eight participants took part. Despite the small sample size, we captured a wide range of viewpoints through a purposive sampling method. Selecting participants based on their background allowed us to include participants aged 19 to 62, with an average age of 46. Furthermore, they had varying levels of smart-home experience and technical know-how. Participation was entirely voluntary and did not require any previous experience with shape-changing interfaces, since the study also aimed to include people hearing about SCIs for the first time. To build a foundation of understanding, every participant received a brief introduction to what SCIs are and what makes them special.

The participants
partially overlapped
between interviews.

The second stage of interviews followed the same approach. We again applied purposive sampling, trying to include the same participants as in the first stage. In the end, six participants took part. Four of them had already been interviewed in the first stage, while two had no prior knowledge of SCIs and needed an introduction. In the second interview phase, participants were aged 21 to 61, with an average age of 36.

Most semi-structured
interviews were
conducted online.

In both phases, we used semi-structured interviews as a basis. A predefined set of questions ensured the same central topics with each participant. Additional follow-up questions clarified individual statements. Most interviews were conducted remotely and lasted approximately 45 minutes on average. They were documented through handwritten notes taken during the session. All sessions were conducted in German, and the statements reproduced in this thesis were translated into English.

After completing all interviews in one phase, we categorized the responses. First, we grouped the statements according to the topics selected in the interview guide. Afterward, we related and combined repeatedly mentioned topics into broader categories. In the first phase, we then linked each category to a possible design response. In the second phase, we further related the feedback to the interface component or mechanism under discussion. Given the small sample size, this analysis focused on recurring patterns to inform design decisions rather than making statistical claims.

The analysis grouped responses into recurring concerns and possible design responses.

3.2 Initial Exploratory Interviews: User Needs and Design Requirements

As briefly mentioned, the first interview phase explored the participants' needs. Our main goal was to identify the control requirements associated with shape-changing devices in smart homes. Additionally, we wanted our concept to tackle existing smart-home system problems. We therefore opened the interview by inquiring about past experiences. Afterwards, we asked how controls must differ to support SCIs. The interview covered the following main areas, while the complete interview guide is provided in Appendix A:

The first interviews examined expectations.

- prior smart-home experiences and potential SCI applications;
- preferences for control, including adjustable properties and coordinated multi-device actions;
- expectations regarding automations, histories, and long-term adaptation;
- safety constraints, system feedback, perceived control, and guest access;
- conversational assistance and the overall organization of the control interface.

3.2.1 User Expectations and Problems

The interviews revealed recurring expectations and concerns across several aspects of SCI control.

Simplicity and Access to Detailed Control

Participants wished for a simple default interface without permanently losing detailed control.

The responses showed that interface use could be divided into two scenarios. During everyday use, participants expected to open the application, quickly change a device's state, and resume what they were doing. They did not want the short interaction to be interrupted by configuration options. More detailed control mattered, however, as soon as a device needed a particular adjustment or acted unexpectedly. Several participants linked this issue to existing smart-home applications cluttered with parameters, making essential settings hard to find. The original design therefore left everyday controls in the default view. Configuration and troubleshooting options were placed on a separate device settings page instead.

Transparency and Feedback

Unexpected physical movement creates discomfort.

Participants expressed hesitation or rejection when discussing unexpected SCI movements. These concerns relate particularly to indirect and system control, where transformations can occur without an explicit user command. Almost all described such behavior as confusing and uncomfortable. One participant even said it could feel like a stranger had altered something in the house. To prevent this kind of feeling, participants suggested different approaches.

Movement should be announced beforehand and remain traceable afterward.

Most wanted shape-change to happen only after a direct interaction, with a clear visual or acoustic indication beforehand. To trace already performed movements, including those by other users, participants additionally proposed a history feature. For transparency, the log should include the triggered action, who initiated the movement,

and when it was executed. To clarify the expected result before triggering a transition, it was deemed useful to provide some sort of preview.

Rooms and Groups

Most participants clearly favored a central interface organized into rooms. Comments such as “it matches the spatial organization of the room” indicated that participants regarded it as an intuitive starting point. Device-level pages could then provide a second level to check and adjust individual devices.

Room-based organization provided an intuitive entry point.

Another expectation was the ability to control several devices together. More than half had already experienced the pain of repeating the same settings across several devices. Two even noted they had to switch between apps because devices with the same capabilities came from different manufacturers. Our initial concept tried to remove these inconveniences by including groups. These groups could then apply a shared change to several devices without sacrificing individual access.

Groups should allow control of several devices at once.

Guests and Safety

When asked about past experiences, participants described problems associated with having guests. Simple actions often required insider knowledge that guests were unable to obtain during a brief introduction. Participants even observed that guests were reluctant to press ordinary buttons because they were afraid of triggering an unwanted action. To prevent this problem, they proposed a restricted guest mode that disables most, or even all, smart functionalities.

Guests require easy-to-use basic controls without access to critical settings.

Privacy and Data Use

Participants accepted action logs but opposed sensor-data storage.

As stated before, participants supported a history of executed transitions. Nevertheless, participants were skeptical about including context-related sensor data. Some considered local sensor histories useful for diagnosing failures. Others rejected storing it entirely because of privacy concerns. For example, recording which transitions were blocked by a person's presence could reveal where and when someone was in the home.

The history supports traceability without exposing contextual sensor data.

To address these privacy concerns, the initial history concept only retained information needed to trace executed actions or restore a previous state. Failures, potentially revealing information about the device's surroundings, were excluded from the log.

Conversational Assistance

A conversational assistant was seen as an optional extra rather than a main interface.

Views concerning the conversational assistant varied. A few participants found natural-language control useful. They explained use-cases for coordinating a number of devices or setting the desired atmosphere easily. However, most doubted its practicality or were worried about actions being carried out without their knowledge.

We included the conversational assistant in the initial concept, supplementing the primary interface by proposing inspectable changes. To preserve control, users must actively select desired changes before execution.

3.2.2 Derived Design Requirements

Overall, we consolidated the initial requirements as follows:

- DR1: State-centric everyday control.** Instead of technical parameters, meaningful states or intended outcomes should express common actions.
- DR2: Progressive access to detail.** The default interface exposes only frequently used controls. Detailed parameters remain available for configuration and diagnosis.
- DR3: Room- and device-based navigation.** Rooms should provide the primary spatial overview.
- DR4: Group control.** Users should be able to control several devices together.
- DR5: Transition transparency.** The interface should communicate target configurations, active transitions, executed actions, and failures.
- DR6: Explainable constraints.** Blocked actions should communicate their cause and indicate a possible resolution.
- DR7: Recoverability.** Users should be able to trace and reverse unwanted actions.
- DR8: Traceable automation.** Automated actions must remain identifiable and should indicate the rule, event, or user action that triggered them.
- DR9: Limited data use.** Sensor data should only be processed and not remain in the history.
- DR10: Optional assistant support.** Conversational assistance should propose inspectable changes that require explicit confirmation.

3.2.3 Development of the Interaction Concepts

Based on these requirements, we prepared two sets of simple sketches. They partially overlapped in terms of history, the assistant, and other minor parts. The general organization also followed the same idea, keeping rooms as the primary organizing structure. States such as “Relaxing”

We created two partially overlapping sketch sets.

or “Movie Night” represented meaningful outcomes that could be applied to one or more devices.

The first sketches used familiar controls.

The first sketches used a more conventional approach. Users could navigate through rooms and devices and see the corresponding states. They could then apply the states through familiar lists and controls. To accommodate non-technical users, individual settings were fully hidden during daily use. To access them, a user had to navigate to the device-specific page. On each page, static target previews were given as soon as a state was pre-selected.

The second sketches emphasized visual previews and navigation.

The second set of sketches concentrated on showing visual displays of the room and its shape-changing devices. It proposed a much more detailed target preview to make physical changes easier and faster to understand. The navigation was also refined. At the start, users would see their exact home layout. Clicking a room zoomed in on it, to show all devices in their current state and position. From this view, users could select and preview different room states, changing multiple devices at once. Selecting a specific device inside the room opened its configuration options.

Both concepts included a history and a conversational assistant.

Both concepts also covered a history view for current and previous transitions. It linked each action to the devices it affected and its trigger to explain why movement occurred. The assistant view explored how a user request could be translated into suggested device states. The sketch presented these suggestions as action buttons, each identifying a device and its proposed state. Users had to explicitly select a suggestion before it took effect.

The complete set of sketches presented during the interviews is included in Appendix D.

3.3 Iterative Interviews: Design Feedback and Final Interaction Concept

Initial feedback helped to find new aspects.

In the second interview phase, we presented all created sketches to each interviewee. We wanted to evaluate

whether the concepts were understandable and balanced simplicity with control. Therefore, participants were asked to review the sketches and identify unclear elements. After three minutes, they described their first impressions. This initial feedback helped us to identify which aspects were already intuitive and which still needed improvement.

In the next part of the interview, participants had to solve various tasks within the sketches. They had to describe a sequence of actions to add a new SCI, locate a specific device, or modify states. Further questions addressed other areas, such as blocked actions and scalability. The complete interview guide and its tasks are provided in Appendix B.

Participants completed tasks and answered follow-up questions.

3.3.1 Feedback on the Initial Interaction Concept

Participant feedback addressed both individual interaction elements and the overall organization of the concept.

State-Centric Interaction

All six participants responded positively to the named states. They described them as faster, less intimidating, and generally better suited for everyday routines. One participant from the first interview explained that this was exactly what he had imagined: selecting the outcome rather than “battling with hundreds of settings.” Based on this, we retained named states as the primary control mechanism.

Participants preferred meaningful outcomes to parameterized control.

Nevertheless, participants still considered easier access to the parameters necessary. They stated it remained essential for exploration and quickly editing states. Technical users also made clear that parameters were indispensable to debug unexpected behavior. In our revised concept, we did not want to display all settings permanently. This would ignore the need derived from the first interview. We therefore decided to use a hybrid approach. Each setting included an option determining whether it should be displayed during everyday use or hidden on a separate device settings page.

Parameters were still needed, and the concept was improved.

Explanation and Preview of States

Static previews were favored by participants.

Before activating a state, participants still needed to understand the outcome of the transition. Preferences on how to display this end-state divided the participants into two groups. Most preferred static previews as shapes were easier to recognize. Animations were harder to interpret and did “not add any benefit.” Others favored animations, appreciating the ability to view a device from all sides. Both groups saw textual descriptions as beneficial, acknowledging that previews cannot display all relevant information.

We decided to use static images along with a description.

For the final concept, we chose static images supplemented with descriptions, as participants who preferred them expressed stronger objections towards animations. In addition, static previews offered more flexibility. First, images could show the SCI in its surroundings. Second, users could choose the viewing angle and replace the image without having to create or modify an animation.

Transitions, Constraints, and Safety

Transition feedback should prioritize outcomes and explain deviations.

Consistent with static previews, participants considered the target state more important than the transition itself. Technical users recognized transitions as relevant, but most found them too “distracting” and “hectic.”

Blocked actions require clear causes and possible solutions.

For blocked actions, participants were presented with two options: a generic error message and a specific error message including technical explanations. Both options also included a solution to resolve the issue. All participants agreed that the specific error message is only beneficial and has no downsides.

Safety constraints require severity-dependent override rules.

They also agreed that critical safety constraints must remain binding if they could cause damage or injury when ignored. However, opinions differed on whether less severe safety restrictions should be overridable. Some argued that sensors could fail and needed to be overridden from time to time. Others insisted that even the potential of causing problems should be enough to fully block actions. To ac-

commodate both mindsets, the revised concept offers both possibilities. New rules default to blocking actions as soon as they are active, but technical users can always disable that option.

Consistency Across Heterogeneous Devices

As mentioned, SCI controls vary greatly, making it hard to create consistent sketches. During the interview, participants pointed these inconsistencies out across different device views. As a solution, they suggested reusing the same layout and elements.

Participants expected a similar layout across SCIs.

In our concept, we therefore standardized the navigation, title, state selection, and status communication. This consistent look was intended to bridge intuitive navigation and device-specific controls.

Standardized elements were used to improve consistency.

Scalability

In general, participants described the presented room-based structure as intuitive. However, many were concerned that the 3D room view could become cluttered, with an increasing number of devices, making it “useless.” Together with it being potentially complex to implement, this concern led us to choose the other approach for our prototype.

However, in this design, two participants expected additional ways to navigate faster. Both suggested a global device search, while one also proposed a favorites function. For our prototype we decided to implement the global search, as it addressed the need shared by both participants and offered more versatile access to devices regardless of the current view.

Global search complements room-based navigation in larger installations.

Group control provided a second mechanism for larger installations. Participants liked adjusting multiple device states by just clicking one button. Repeating adjustments across multiple pages felt tedious and annoying.

Groups and room states support scalable control across devices.

3.3.2 Final Interaction Concept

The second interview phase translated user feedback into concrete design revisions.

Concluding these findings, we revised the interaction concept as follows:

- Named states remain the primary interaction mechanism.
- Individual parameters can be shown or hidden.
- Static previews communicate target configurations with supporting textual descriptions.
- Blocked actions provide specific explanations and possible solutions.
- Critical safety constraints remain binding.
- Less severe restrictions can be overridden explicitly.
- Navigation, state selection, and status communication are standardized across devices.
- A room-based structure remains the primary form of navigation.
- A global device search supports navigation in larger installations.
- Groups enable coordinated changes across several devices.

Participants also reacted positively to the guest-access concept and to the proposed version of conversational assistance. However, both were considered supplementary features rather than core mechanisms for controlling SCIs and therefore remained outside the scope of this thesis.

Chapter 4

System Design and Implementation

This chapter describes how we developed the prototype for controlling SCIs in smart homes. It translates the requirements derived from the first two interview stages (see Chapter 3) into a specific interface connected to Home Assistant. At first, we present the interaction from setup to everyday use. Afterward, we describe the underlying representation and its connection to Home Assistant. Finally, we cover execution checks and the prototype's scope.

4.1 Interaction and User Interface

The following simplified example illustrates how the LineFORM device is represented and controlled in the prototype. It demonstrates the interaction flow by showing how a user transforms the device into a predefined configuration. Additional interface views from the prototype complementing this example are provided in Appendix E.

Consider a user who wants to transform LineFORM, depicted in Figure 4.1, into a reading lamp. To achieve this, a chain of motorized segments must bend into a specific shape. First, the user opens the living room and selects

LineFORM demonstrates the prototype's control workflow.

Users apply a predefined reading configuration.

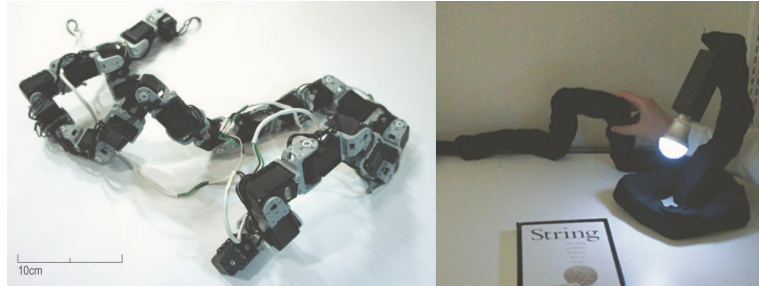


Figure 4.1: LineFORM’s physical construction and an example application. The uncovered prototype shows the series of connected servo motors (left), while the shape-changing cord forms a lamp with a physical lever for controlling its brightness (right). Images adapted from Nakagaki et al. [2015].

LineFORM. Several states are presented, including a reading configuration. Each state has its own card, with an image showing the final state and a short clarifying description. Upon applying this state, LineFORM moves all individual joints into their set target angles. The result is a reading lamp, perfectly adjusted to its environment. Figure 4.2 illustrates this navigation sequence from the overview through the room view to the device view.

Configurations can be adjusted, saved, and restored.

If the lamp’s final position needs an adjustment, the user can change individual joint angles and apply them directly. If desired, they can then save the resulting configuration as a new state for later use. To return to the previous configuration—for example, because the user prefers that one or made a mistake—they can restore it from the history.

Actions are validated, executed, and recorded.

From the user’s perspective, no further actions or considerations are required. However, in the background, the application will run several tests to provide a safer experience. If an active rule blocks the action, the user will be notified. Otherwise, it validates the requested target angles against their permitted ranges. If all checks pass, it sends the parameters to Home Assistant, which forwards them to the device. Once the reported values match, the application records the change in its history and displays the estimated remaining transition time.

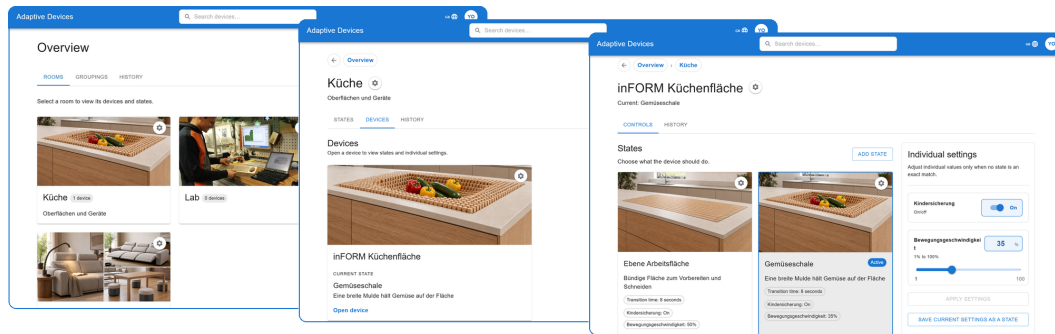


Figure 4.2: Navigation through the prototype from the overview to the room and device views. The first two views are partially overlaid, while the complete device view shows the activation and transition of a predefined state.

4.1.1 Device Setup and Configuration

ESP32 AND ESPHOME:

The ESP32 is a compact microcontroller commonly used in embedded systems. ESPHome is a software framework that integrates such microcontrollers into home automation systems.

Excursus:

ESP32 and ESPHome

Before an SCI can be included in the application, its properties must be exposed to Home Assistant. This requires a suitable integration between the SCI and Home Assistant. For instance, an ESP32 running ESPHome can be used to expose joint angles as number entities and context data through sensor entities. Our prototype, however, just depends on the exposed entities and supports every protocol that Home Assistant understands.

SCI properties are exposed through Home Assistant entities.

As soon as these entities are available, the process of setting up the device can begin. First, users enter a device name and select which Home Assistant entities represent the properties of the SCI. To locate the relevant entities, they can carry out a search and use filters for the setting types. This flexible mapping allows the prototype to technically support every SCI.

Users link entities to device settings.

Settings are adapted to the device environment.

The chosen entities must then be adapted to match the device's surroundings. Individual options of categorical settings may be disabled, while numeric settings can be limited to a smaller range within the specified capabilities. For example, an angle setting could be restricted in order to stop LineFORM from hitting a nearby wall. In this step, users can also choose which settings stay in the configuration dialog and which appear directly on the device page. Later on, this configuration dialog allows them to change the device name, assign it to a new room, and add or remove linked entities.

Status rules restrict potentially unsafe actions.

Status rules are configured as the second part of this process. Users select a sensor entity and define the condition under which users are notified, need to confirm their action, or cannot execute it at all. As status rules are a central part of our prototype, their behavior and role during control are described in more detail later in this chapter.

States store reusable device configurations.

Once the device settings and rules have been set up successfully, users should create states. Each state contains a name together with one target value for every linked setting. An image, a description, and an estimated transition duration communicate the expected outcome. Usually, users would need to set all target values manually. Instead, we wanted them to explore the device and apply changes to it. We therefore implemented an alternative way in which users can save the device's current configuration after adjusting it to their liking.

4.1.2 Control Workflows

The home view organizes devices by room.

Once setup is completed, everyday interaction begins in a home view organized around rooms. Each room is represented by a card that includes its name and an image. To give a good understanding of how the room currently looks, the image normally displays a collage of the preview images of the devices in the room. However, if users prefer a static image, they can upload a custom one. Separate tabs lead to groupings and to the global history.

BREADCRUMBS:

Breadcrumbs show the user's current position within an interface, for example, *Home → Living Room → Reading Lamp*.

Excursus:
Breadcrumbs

Selecting a room opens its states, devices, and room-specific history. Breadcrumbs indicate the current position and provide a clickable path back to the home view. This approach realizes DR3 and guides users step by step from the overview to a specific device.

Room views and breadcrumbs guide navigation.

On the device page, the states are shown alongside the settings. The history remains in a different tab. Most settings stay hidden to keep the states in focus. However, we still retain access to the most important ones by displaying the selected settings directly on the page, implementing DR2.

Device pages prioritize states and selected settings.

Users who already know which device they want to reach can bypass the view hierarchy through the global search in the header. Preview images and room names help to distinguish similar results. The header also contains the language selection and access to the Home Assistant user profile.

Global search provides direct access to devices.

To support smaller screens, like phones, the design also includes responsive adjustments to a certain degree. Cards rearrange according to the available width, and a sticky action bar keeps the relevant apply button accessible while users scroll. However, not all features are supported on smaller screens.

Responsive adjustments support smaller screens.

Individual Device Control

The saved states are the primary control on the device page. Each state is displayed as a card including its name, description, and preview image. The card also includes indicators representing the estimated transition duration and the target values for the settings displayed on the device page. This fulfills DR1 by allowing users to choose an intended outcome instead of adjusting every parameter individually.

Saved states provide outcome-based device control.

Preview images show the expected physical result.	The preview image communicates the expected physical result. Clicking on it opens a larger view in which the device can be inspected more clearly. Since the users supply the images themselves, this preview can display the SCI in its actual environment. An image of LineFORM may therefore communicate both its posture and its position relative to nearby objects.
Selection separates inspecting and applying a state.	To give users the opportunity to inspect changes before applying them, we separated those steps into different actions. The first click on the state card selects it, while a different button applies it. For repeated use, the state can also be double-clicked to directly apply it.
Individual settings enable direct parameter control.	If no saved state suits the current activity, the individual settings can be used. Numeric settings combine a slider with a value field that supports keyboard input. Boolean settings use ordinary switches, and categorical settings present the permitted options in a dropdown menu. All settings match the limits that were defined during setup.
Reported values determine the active state.	To prevent commands from being sent to Home Assistant at every slider position, modified values are only sent after the user applies them. Once Home Assistant reports the updated values, the application compares them with the saved states. If the configuration matches a state, it is marked as active. If no state represents the configuration, the applied state will be set as individual with a generic preview image.

Room and Group Control

Room states control multiple devices at once.	The same state-based interaction also applies to the control of multiple devices. Room states combine multiple devices located within one room. Each room state therefore has a name and saves each device together with its target state. Applying this state triggers the transitions for all selected devices at once, skipping the back-and-forth of preparing a room for an activity.
---	--

Groupings follow the same principle but extend it across room boundaries. They can therefore present broader routines, such as a morning routine. Just as with room states, users reuse existing device states instead of entering individual parameters. Room states and groupings therefore address DR4 by combining several device changes into one action.

Groupings extend combined control across rooms.

Their cards identify the included devices and corresponding target states. Composite previews reuse the available state images, while per-device indicators distinguish reached targets, estimated transitions, and differing configurations. Users can inspect this information before applying the combined action.

Cards preview targets and transition status.

The commands are issued as one requested action, but the underlying integrations determine how each device executes them. Simultaneous physical movement across all included devices can therefore not be guaranteed.

Device integrations determine the actual execution.

4.1.3 Feedback and Recovery

Whenever users prepare an action, status messages indicate conditions that require attention or prevent movement. Each message belongs to a configured rule and can explain both the cause and a possible resolution. Connection problems are shown separately from contextual restrictions because an unavailable entity requires a different response from a condition reported by a sensor.

Status messages explain restrictions and connection problems.

The configured severity determines how a message is presented. Informational messages communicate context without interrupting the action. Non-blocking warnings and errors require acknowledgment, whereas an active blocking rule prevents the action entirely and cannot be overridden. These explanations address DR6, although their usefulness depends on the description entered during configuration.

Severity determines how rules affect an action.

Rules are refreshed
and rechecked before
execution.

Rule information is refreshed every few seconds while the device page remains open. Changes will therefore become visible shortly after they occur. Before executing an action, the backend evaluates all blocking rules again. This prevents a delayed interface update from allowing an action that should be blocked.

History and Restoration

The history records
successful device
changes.

After a successful action, the history records what changed. It is available globally and from individual rooms and devices, where the corresponding filters are applied automatically. Each entry identifies the time, initiating actor, affected devices, and their previous and resulting values.

Filters and search help
locate relevant
changes.

Further filters narrow the history by room, device, action type, actor category, or time interval. Separate active and undone views distinguish changes that can still be restored from those already reversed. A search field supports cases in which users know which entry they need. Along with previews and transition indicators, these records contribute to the transparency required by DR5. Failed actions appear through error messages instead of being added as successful history entries.

Only the latest device
change can be
restored.

Restoration can only apply the latest active recorded change for each device. This avoids silently replacing an older entry with a newer configuration. When an action affected several devices, eligible devices can be restored separately without reversing the remaining changes.

Restorations are
validated and recorded
as new events.

An undo request sends the recorded previous values through the same checks as any other control action. It may therefore be rejected if a setting no longer exists, a value is no longer permitted, or a blocking rule is active. A successful restoration creates a new history event rather than deleting the original one. This preserves both actions and addresses DR7 by making unwanted changes traceable and reversible.

4.2 Domain Model and State Representation

The domain model separates spatial organization from device capabilities. A room has a name and a stable identifier used for navigation. Optionally, it can include a description and an image. Apart from its own attributes, rooms contain an arbitrary number of devices. Each device can only be assigned to exactly one room. Room states and groups simply establish new relationships and access device information to do so.

Rooms organize devices independently of their capabilities.

To support heterogeneous SCIs, devices are defined as a collection of settings, not a hardware class. Each setting has a type, presentation metadata, and a reference to a Home Assistant entity. Currently, the supported types are numeric, boolean, and categorical. Joint angles and pin heights can therefore both be presented as numeric settings. Their names then identify the individual setting, while their units allow angles to be distinguished from distances.

Settings represent heterogeneous SCI capabilities.

Stored setting values provide a local representation of the configuration. Live values and capabilities are synchronized with Home Assistant. This distinction matters during disconnection: retaining a stored value preserves context, but cannot establish the current physical condition of the device. On a device refresh, available live values update the local representation.

Stored values preserve context between live updates.

4.2.1 State-Based Configurations

A device state stores a target value for every setting linked to that device, making each activation independent of the others. Without complete assignments, the same state could produce different results depending on the previous configuration. The state's metadata, like the name, supplies the human-readable component that a sequence of settings lacks.

Device states store complete target configurations.

New settings are added to existing states.	When a setting is added to an existing device, we append its current value to each existing state. This preserves completeness, but the initial value may not express the intended meaning of every state. Best practice would therefore be to define all settings before creating states. If not, existing states may require review afterward.
The current state remains independent of frontend selection.	The current state reference identifies a configuration represented by the device's values. It is distinct from the selection that a user is currently inspecting in the frontend. Selecting a card consequently does not change the stored current state.
Live values determine the active state.	The active state is determined by retrieving the device's current values from Home Assistant. These values are then compared with the target values of all saved states. If all values match a saved state, we set the corresponding state to active. A complete set of available settings must be provided for this check.
Combined configurations reference existing device states.	Room states and groupings only store references to device states. They do not maintain independent copies of the same parameter sets. A change to a referenced state therefore also changes the target used by configurations that include it. This avoids divergent copies and prevents multiple adaptations when changing a state.
Transition metadata estimates the expected duration.	Transition metadata records an estimated start and end based on the state's configured duration. It supports a temporary transition indicator without requiring every device to expose movement progress. The estimate does not describe the movement path or confirm that a mechanism has physically stopped. Its role is to communicate the expected interval associated with an activation. However, before a transition can be executed, the device's context is checked through status rules.

4.2.2 Status Rules and Control History

Status rules link devices to Home Assistant sensor or binary sensor entities. A rule specifies a comparison and the value that triggers it where needed. Supported comparisons include equality, numeric thresholds, and binary conditions. Unavailability can be configured as a trigger itself. In contrast to the initial concept, severity and blocking behavior are configured independently: the severity selection controls the presentation, while a separate flag determines whether a triggered rule blocks control.

Status rules define contextual triggers and control effects.

These rules are not related to limiting settings. Limits describe permitted target values; status rules depend on the currently reported context. This means a target may fall within its numeric range and still be rejected because an active rule blocks the device. Saving sensor readings inside the history would contrast with DR9, possibly revealing information about the environment. Therefore, we had to evaluate sensor readings without copying them into the log.

Limits constrain targets, while rules evaluate live context.

History events instead only store the initiating action and contain a separate item for each affected device. Each item records before and after values, along with the associated state information. Names and other identifying details are saved as snapshots so that older events remain understandable after configuration changes. The undo status is recorded per item as well as per event. A combined event is fully undone only after all its items have been restored.

History events preserve changes and configuration snapshots.

As a Home Assistant add-on, the prototype's history can only cover actions handled by the application itself. Changes made outside the application, for example directly in Home Assistant, unfortunately cannot be tracked consistently. To still support restoration, the application saves the last known device values before each action. Undo uses these saved values, although they may be outdated if an external change has not yet been detected.

External changes can make restoration values outdated.

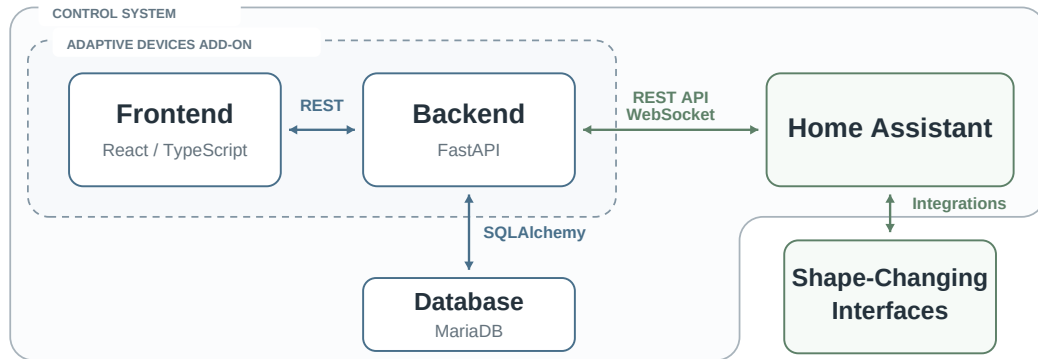


Figure 4.3: System architecture, component responsibilities, and communication interfaces of the implemented prototype.

4.3 System Architecture and Home Assistant Integration

The add-on combines a React frontend and FastAPI backend.

As depicted in Figure 4.3, the prototype executes as a Home Assistant add-on. The frontend is implemented using React and TypeScript with Material UI components. A Python FastAPI backend handles configuration and control requests. To centralize validation, all communication goes through this backend instead of sending device commands directly to Home Assistant. Additionally, the backend serves the frontend, deploying the interface with the application API. Both the frontend and backend are packaged in one container and require access to Home Assistant and MariaDB.

MariaDB persists the domain model and history.

To persist the domain model in MariaDB, the backend relies on SQLAlchemy. This includes state definitions and configuration relationships, as well as the control history. Uploaded images are kept in the persistent add-on storage, with their metadata in the database. For custom setups, the database connection can be configured through the add-on options. At every startup, Alembic then executes database migrations and brings the schema to the expected version. This allows updating the add-on to its newest version without causing unexpected problems. After that, the web service starts and the backend checks the database connectivity.

Users access the add-on through Home Assistant ingress. It forwards current user information from Home Assistant to the prototype, allowing us to track who triggers what action. Currently, we fully rely on the forwarded information and do not maintain a separate account system.

Home Assistant ingress provides access and user information.

4.3.1 Entity Mapping and Capability Restrictions

To communicate directly with entities from Home Assistant, we map them to our settings. Numeric settings map to number or input_number entities. Boolean settings use switch or input_boolean entities, and categorical settings use select or input_select entities. This mapping defines the integrations that the prototype supports.

Settings map to corresponding Home Assistant entities.

The backend then reads the entity attributes to determine available values. For numeric settings, the effective interval is the intersection of the exposed bounds and the application's configured bounds. Additionally, Home Assistant exposes a step size which restricts valid targets. For categorical settings, we remove the options disabled by the users from the choices offered by Home Assistant. Configuration can therefore narrow the available controls without extending the entity's exposed capabilities.

Entity capabilities and configuration determine valid values.

The same checks are required when an existing state is activated. A state that was valid when created may become invalid after the integration changes its range or removes an option. Reading current capabilities before execution prevents this problem.

State targets are revalidated before execution.

This abstraction supports a common interface for different devices, but it does not translate arbitrary descriptions of shape into actuator commands. Device-specific integration still remains responsible for exposing suitable entities to Home Assistant.

Device integrations must expose suitable control entities.

4.3.2 Scene Integration and Automation

States and groupings are exposed as Home Assistant scenes.	To support automated control, like notifications, all states and groupings are exposed as scenes in Home Assistant. This makes the configurations available to Home Assistant automations without requiring users to recreate their target values separately. To distinguish configurations, scene identifiers include readable names and database identifiers.
Scene activations follow the standard validation process.	Activating a scene changes a shared helper that identifies the requested configuration. A backend listener receives the corresponding Home Assistant event and calls the appropriate backend endpoint. As a result, scenes use the same target validation and blocking rules as manual activation. To ensure consistency, scene names are updated as soon as the underlying configuration changes. Manual adjustment of the scenes is prevented by Home Assistant.
Scene outcomes are recorded without their automation context.	Successful scene activations enter the control history as system actions. The recorded actor identifies scene-based execution, but does not preserve the particular automation rule or triggering event. The integration also emits success or failure events to Home Assistant. These events can then trigger new actions or notify users of the failure and how to resolve it. This partially addresses DR8 by identifying scene activations in the history. However, it does not explain which automation or event initiated them.
Direct entity control bypasses validation and history tracking.	Actions that control certain entities and are directly triggered in Home Assistant, bypassing scenes, also skip the backend. Home Assistant does provide events containing previous and updated entity states. However, the current prototype does not check these events, as it would require routing all requests through the application. The prototype can retrieve the resulting values later, but cannot reconstruct the entire history from them. As these requests skip the add-on's backend, we also cannot enforce validation and blocking rules on them.

4.4 Control Execution and Reliability

This section describes how control requests are validated and executed before examining the system’s error handling and remaining reliability limitations.

4.4.1 Validation and Execution

Validation begins with the request structure and the referenced application objects. The backend first checks whether the requested configuration belongs to the relevant device or room. It then evaluates blocking status rules and prevents the action if necessary. Otherwise, it resolves the requested state into concrete settings.

Requests are validated before targets are resolved.

For a target batch, the backend checks current entity availability and setting types. Numeric values must satisfy the effective range and step size. Categorical values must remain among the allowed options. These checks also apply to individual settings, so not using states does not remove the restrictions. If a batch assigns conflicting values to the same Home Assistant entity, the backend rejects it before sending commands.

Target batches must satisfy all entity constraints.

After validation, the backend selects the service corresponding to each entity type and sends the requested targets. It then reads the entities repeatedly for a bounded number of attempts, checking whether their reported values match. A successful service call alone is insufficient to complete the operation. Confirmation is required before the action is committed as a successful configuration change with a history entry.

Reported values confirm successful configuration changes.

This confirmation concerns the state Home Assistant reports. Some integrations may report a requested setpoint before the physical movement finishes. Confirmation and the estimated transition interval therefore serve different purposes: one establishes agreement with the entity value, while the other represents the configured expectation about duration.

Entity confirmation does not confirm completed movement.

Multi-device restoration
is validated and
executed sequentially.

Restoration allows users to return a device to the recorded values. Restoring the values sends a request to the backend, which again performs all previously mentioned checks. When restoring several devices, these checks and subsequent changes are performed one device at a time. A failure could therefore occur after another device has already been restored.

4.4.2 Error Handling and Limitations

Failure messages
identify causes and
affected elements.

When an action fails, the interface explains the cause and identifies the affected device or setting where possible. The messages distinguish invalid configurations from connection problems and target values that Home Assistant has not confirmed.

Control requests are not
retrieved automatically.

For temporary connection problems, the application may retry requests that only retrieve information. The frontend does not automatically resend control requests, as the original command may already have reached the device despite a missing response. Nevertheless, stored configurations remain intact during a disconnection. If access fails because the Home Assistant session has expired, the interface identifies this as an authentication problem.

Partial failures can
leave devices in mixed
configurations.

A failure during execution can leave some devices at their requested targets while others remain unchanged. In this case, the application reports the failure. Commands already sent are not reversed, so users may need to inspect the current values before continuing.

Status rules do not
replace device-level
safety.

Status rules can also only provide limited protection, as they check conditions before execution. These conditions could change after the checks, and the application cannot continuously monitor movement or stop devices when an obstacle suddenly appears. Physical collision detection and emergency stopping must therefore be handled by each device's underlying control system.

4.5 Prototype Scope

As we did not identify an existing interface that provided the required control mechanisms for heterogeneous SCIs, we implemented the prototype described in this chapter specifically for this thesis. ChatGPT by OpenAI was used as a supporting tool to generate, revise, and debug selected code fragments. We reviewed, adapted, and tested all AI-assisted code before integrating it into the final implementation. Home Assistant and the introduced software frameworks provide the underlying infrastructure.

The prototype was implemented to address the research question.

For ease of testing, the implementation includes demonstration configurations for simple adaptive furniture in several rooms. They exercise different setting types and named states within the interface. Supporting helpers automatically create Home Assistant entities and link them to the demonstration settings. This provides controllable entity values without requiring a physical actuator for each illustrated device.

Demonstration configurations simulate devices without physical actuators.

The resulting configuration supports demonstration of the interaction and integration mechanisms. It does not establish the movement characteristics of physical furniture. In particular, helper values cannot validate collision handling or the accuracy of a configured transition duration. Physical behavior still depends on the hardware, as we focused only on a shared control environment.

The demonstrations cannot validate actual physical behavior.

Overall, almost all design requirements have been translated into concrete features of the prototype. DR8 remains partially implemented. Scene activations are recorded as system actions, but the history does not identify the particular automation or event that initiated them. Improving this attribution would help users understand why an automated change occurred and distinguish it from other actions. The optional assistant described in DR10 is the only requirement without an implemented feature. Its integration would have required decisions about model selection and token usage costs. Safety concerns also required consideration, particularly how to handle incorrect or ambiguous suggestions involving physical movement. We there-

The prototype implements almost all design requirements.

fore focused on the core control interface. A future assistant would need to present proposed changes for explicit confirmation and use the same validation and blocking rules as other control requests.

The implementation enables evaluation of the control concept.

The implemented features provide the basis for evaluating whether the interface offers the intended clarity and control during use.

Chapter 5

Evaluation

This chapter presents the evaluation of the prototype that we implemented. It looks at how participants used the interface and which parts were still difficult to understand. We first describe the study and its relationship to the previous interview phases. Afterward, we report the findings derived from the participants' feedback.

5.1 Evaluation Method

Again, we selected participants based on their backgrounds to include a range of technical knowledge and experience with smart homes. In total, seven participants aged 21 to 62 took part, with an average age of 42; five of them had already participated in at least one previous interview phase. This time, all interviews were conducted online. The process of documenting the interviews and carrying out the qualitative analysis was the same as in the earlier phases.

In this final phase, we concentrated on understandability, transparency, and user control during practical use. Participants were asked to independently explore the implemented interface and complete a set of tasks. This allowed us to examine their opinions of the available functions and the difficulties they encountered.

Seven participants took part, five of whom had attended an earlier interview.

The final study examined practical interaction with the prototype.

Home Assistant helpers represented the devices during the study.

To solve the tasks, participants operated the interface themselves. Due to the online format, no physical SCIs were connected. Instead, the prepared test data and Home Assistant helpers served as digital copies of SCIs. Participants could therefore not observe actual physical movements.

All participants completed the full set of ten tasks.

The same task catalog was completed by all seven participants. In total, there were ten tasks, which became progressively more difficult and technical. The starting task was to create a room with a device, while later tasks included creating blocking rules to prevent damage. Additionally, participants saved and applied device states, created a room state, and restored a configuration through the history. Finally, they deleted the devices and rooms which had been created during the study. The complete evaluation guide and task descriptions are provided in Appendix C.

We then grouped the collected feedback into topics using the same approach described in Chapter 3. The following section presents the resulting findings.

5.2 Results

Overall, participants said the interface was clearly organized and easy to learn. They valued the replacement of parameters with states and liked the recovery functions. Nevertheless, the participants also mentioned some difficulties. They described some inconsistencies during selection steps, unclear configuration workflows, and overly technical explanations.

5.2.1 Navigation and Orientation

The room-device-settings hierarchy was intuitive.

Moving from the overview to rooms and finally to devices and their settings was found to be easy and intuitive. Participants said that the overview offered a clear starting point that was then “drilled into” for more specific actions. Opening the device settings then finally gave access to the

most granular and technical view of the SCI. This organization was perceived as very good for daily use, while still allowing for fine-tuning settings when needed. Overall, this hierarchy supported ease of use while retaining access to detailed settings when needed.

Additionally, participants appreciated the implementation of the global search. Combining the device name with the preview image and the room name helped to identify devices “at a glance”, while the search itself prevented the “back and forth” needed to switch between devices.

The global search provided quick access to devices.

5.2.2 Configuration and State-Based Control

From the device view, participants found the adjustable device settings very useful. They welcomed the ability to adapt each SCI to their specific home environment. However, they remarked that some technical terms remained unclear to them. For example, entities exposed by Home Assistant were not always understandable, making it unclear which device setting they represented.

Configurable ranges and display names were useful, but some terminology remained unclear.

To explore configured settings and then be able to save them as a state with a single click was another valuable feature for participants. They liked the ability to fine-tune the current configuration at any time without needing to scroll through all settings. After all, users could choose which settings remained visible on the device page for direct access. With regard to numeric settings, participants appreciated having a combination of a slider and a number field. While the slider supported quick adjustments, the number field allowed for precise inputs to “get the decimals right.”

Named states supported recurring activities, while parameters supported fine-tuning.

Participants did not always recognize how to select and apply a saved state. Multiple participants noted that clicking the card opened its image and found it unintuitive that they had to click the bottom part of the card to select it. Some also questioned why selecting and applying a state required separate steps. Instead, they proposed a dedicated button on each card that would either select or instantly apply the state.

State cards did not clearly communicate how to select and apply a state.

Coordinated control was useful, but room states and groupings needed explanation.

Beyond individual device states, participants thought it was practical to control several devices through a shared action. However, the difference between a room state and a grouping was often unclear and required explanation. One participant even questioned why room states were needed at all, as correctly configured groupings could replace them.

5.2.3 Feedback and Safety Rules

Success messages and active-state indicators helped participants follow changes.

After carrying out an action, the participants found success messages to be helpful. Furthermore, they liked the “Active” indicator for identifying the current configuration instead of just a highlighted outline. The transition indicators, showing which device was currently expected to be moving, were also viewed positively.

Blocking messages clearly communicated restrictions.

The displayed blocking messages were noted as clear and helpful, explaining why control was blocked and what needed to change before the device could be operated again. Technical participants also valued the option to add additional sensors that were not built into the SCI. For example, a separate presence sensor could prevent a device from moving while someone was in the room. On the negative side, participants observed that certain messages remained visible when switching between devices. Here, they were unsure which device the feedback referred to.

Creating the relevant rules was more difficult. Terms such as “backend” and “read-only” were “too technical”, causing trouble during configuration. The distinction between a rule’s severity and whether it blocked control also required explanation.

5.2.4 History and Deletion

The history made changes traceable and supported recovery.

The history was also found to be helpful when reviewing earlier changes. Participants appreciated being able to inspect what an action changed and restore the previous set-

tings. In particular, for group and room actions, they appreciated the ability to reverse one device without affecting the others. However, the controls for reversing an entire action or an individual device were not always easy to distinguish. Despite this initial uncertainty, participants found it reassuring that an entry was marked as undone after a change had been successfully reversed.

Difficulties with the history mainly stemmed from the level of detail available. Participants considered the amount of information “too much to deal with,” and understanding it took “considerable attention.”

The history displayed too much information at once.

Regarding deletion confirmation, participants had mixed feelings. They valued the additional confirmation steps to protect against accidental deletion and the accompanying warning messages. However, most deletion notices were considered to be very general. For example, the room deletion warning always stated that all devices in it would be deleted alongside. Two participants found this alarming and double-checked whether the room was actually empty before continuing.

Deletion confirmations helped guard against accidental actions.

Chapter 6

Discussion

In this chapter, we discuss what the prototype contributes to understandable, transparent, and controllable SCI interaction, and which challenges still remain. Taken together, the prototype and its evaluation support an approach that combines meaningful states with access to individual settings and explanations of device behavior. The findings also show that these mechanisms fall short in certain areas, particularly during configuration and when interpreting detailed feedback. We first examine these contributions in relation to previous research. Afterward, we discuss the methodological and technical limits.

6.1 Contributions and Remaining Challenges

The evaluation revealed contributions and remaining challenges across four aspects of the control concept. Whereas the evaluation chapter organizes observations by the interface areas examined during the study, this discussion relates findings across those areas to the broader themes of the research question.

6.1.1 State-Based Control and Understandability

Named states provide a meaningful starting point for everyday control.

Across the concept interviews and the final evaluation, named states received very positive feedback. This suggests they provide a useful abstraction for SCIs by combining their parameters into configurations. The room—device—settings hierarchy further supported this approach by providing intuitive and easy-to-use navigation. Together, these findings support organizing everyday interaction around recognizable outcomes. This aligns with Davidoff et al. [2006], who argue that smart-home control should accommodate domestic routines and their adaptation.

States and individual parameters serve complementary purposes.

In addition to states, participants continued to support individual adjustments with parameters. States and parameters therefore served two parts of the same activity: states allowed users to apply an outcome easily and quickly, while parameters supported creating and fine-tuning it. Keeping only selected settings visible connected these two forms of control and prevented users from needing to understand every parameter. Consistent with Zaidi et al. [2026], switching between states and individual parameters should remain straightforward. This allows an established state to serve as a starting point for adaptation when a user's needs change.

Simpler everyday interaction still depends on successful configuration.

However, the benefit of states still heavily depends on someone preparing the settings and states. The described difficulties with entity names show that the prototype still requires users to connect technical properties with their physical meaning. The state-based abstraction shifts this technical decision-making almost entirely to the initial setup. This should therefore be carried out by someone with enough technical knowledge. The interview findings suggest that, once the setup is complete, other users can control the SCIs with little to no technical knowledge. Beyond selecting a prepared state, users also need to understand whether it can currently be applied and what the displayed responses mean.

6.1.2 Transparency of States, Actions, and Constraints

In the evaluation, we found that feedback was particularly helpful when its meaning was explicit. The active-state indicator identified the current configuration and set it apart from other states. Blocking messages additionally explained why control was unavailable and what needed to change. These observations connect to the questions about system behavior examined by Lim et al. [2009], who investigate how different explanations affect understanding. Our findings show that participants found these explanations very helpful when controlling SCIs. However, their usefulness depends heavily on how clearly the supplied descriptions explain the condition and how to resolve it.

Feedback was useful when it answered a concrete question.

Providing these clear descriptions is part of the rule configuration process, which participants found more difficult than understanding the resulting messages. Certain terms used in the editor were too technical and required explanation. Additionally, users needed clarification on the difference between severity levels and blocking behavior, and where they differ. Future rule editors should therefore focus on making this process clearer for non-technical users.

Understanding a restriction does not imply being able to configure it.

Another boundary emerged in the history. Participants valued the ability to review earlier changes but found the level of detail demanding. Providing more information did not automatically make those changes easier to understand. A possible improvement could be to first show what happened and which devices were affected, with detailed parameter changes available on demand. Beyond the history, feedback messages also need a clear device reference. Messages that remained visible after navigation created uncertainty about which device they described. Letting users adjust how long messages remain visible could accommodate different reading needs, while keeping each message clearly linked to its device.

Transparency depends on relevant information and an unambiguous context.

For SCIs, feedback must additionally connect displayed configurations with physical behavior. In the concept interviews, participants favored static previews together with

A target preview cannot explain the complete physical transition.

descriptions. The final evaluation showed that participants viewed this representation positively. Nevertheless, these static images do not anticipate movement between two states. Tiab and Hornbæk [2016] investigate how shape-changing mechanisms relate to perceived state and feedback, emphasizing the importance of this connection. Because our prototype provides only the target representation and an estimated transition interval, it does not show the SCI's movement path. The interface should therefore make clear that these indicators describe an expected transition and do not confirm the device's actual progress or completion.

6.1.3 User Control and Recoverability

Deliberate activation
must make its purpose
apparent.

In our prototype, we separated selecting a state from applying it so users could inspect target configurations beforehand. However, participants did not always understand how to select a state or why applying it required another step. A clearly labeled apply button could make this distinction easier to understand. Confirmation dialogs caused similar uncertainty when their warnings did not reflect the current situation, such as when the interface warned about deleting all devices in a room even though the room was already empty. They should therefore clearly describe what the requested action will change.

Automation needs
explanations of its
triggers as well as its
outcomes.

With automation, users instead define in advance when a change should happen, allowing the system to act without confirmation. This shifts some control to the system, as described by Rasmussen et al. [2016], and makes it important to explain afterward what triggered a change. Our prototype supports automations through Home Assistant that can activate states independently. However, the history records these activations only as system actions and does not identify the automation. Users can therefore see what changed but may not be able to trace why. Including the triggering automation in the history would help users trace unexpected changes and identify which rule to adjust to prevent them from happening again.

Regardless of who initiated a change, users may want to reverse its outcome. The implemented history supported this by making previous configurations available for restoration. Participants particularly liked that they could restore only one of the devices affected by a group or room state. This suggests that users may want to retain most of a configuration while only reversing one part of it. Such flexibility fits the emphasis on exceptions and breakdowns in domestic routines described by Davidoff et al. [2006]. That the distinction between singular and complete undo controls was partly unclear nevertheless demonstrates that this must be displayed clearly.

Recovery makes recorded changes available for corrective action.

6.1.4 Shared Control of Heterogeneous SCIs

In general, the prototype supports all devices that can be controlled through a microcontroller and a protocol supported by Home Assistant. Together with room and group actions, this addresses part of the need for cross-platform software and end-user programming tools identified by Alexander et al. [2018]. Its contribution therefore lies in connecting configuration, activation, explanation, and recovery within one interface.

The prototype provides a common control model above device integration.

Connecting multiple configurations was valued by all participants, but the difference between room states and groupings was not clear to everyone. Both concepts reuse device states, and their spatial scopes did not make their differing purpose apparent. This suggests that supporting more combinations can introduce additional conceptual work. Brush et al. [2011] describe the difficulties of coordinating and managing household technology. Our findings suggest that bringing device controls together can also introduce new complexity if they partly overlap. A single standardized workflow for combining device states might prove to be easier to understand.

Common controls still need distinguishable organizational concepts.

6.2 Limitations of the Study and Prototype

The limitations of this work arise from both the study design and the technical scope of the implemented prototype. The following subsections distinguish methodological constraints affecting the interpretation and generalizability of the findings from technical constraints affecting device support, reliability, and functionality.

6.2.1 Methodological Limitations

The small sample and repeated participation limit generalizability.

Across the studies only small groups of participants were involved. Even though the participants were carefully selected, five of the seven participants in the last study had already attended an earlier interview. Their familiarity with parts of the concept may have helped them to understand the interface. In addition, earlier design decisions may also have influenced their opinions about certain parts of the prototype. The results therefore provide useful feedback on the interface, but may not fully reflect how new users would experience it.

Note-taking and interpretation influenced the recorded findings.

The findings also depend on how we documented and analyzed the interviews. We took handwritten notes during the interviews and grouped the recorded statements into recurring topics afterward. This helped identify common concerns across participants, but we may have missed some details during the sessions. The resulting topics also depended on our interpretation of the notes, which may have given more attention to certain observations than others.

The online study covered interface use without physical devices or long-term use.

The study setting further limited which aspects we could assess. Participants, for example, operated on digital representations of SCIs. They therefore could not observe the actual movements or experience their effects on nearby people and objects. Testing the interface in an actual household would also introduce situations that the study did not cover, including shared devices, changing routines, and installations that grow over time. How these situations affect the use of the interface remains open.

Regarding the evaluation, we focused on how participants used the prototype and which features they found helpful or unclear. To achieve this, we used qualitative feedback to understand their experiences and identify opportunities to improve the design. In other words, we focused on SCI control design and did not use standardized rating scales and performance measurements for comparison. Assessing performance relative to other interfaces would therefore require a separate comparative study.

The evaluation focused on practical use and participant feedback.

6.2.2 Technical Limitations and Scope

Beyond the study design, the prototype still cannot support all devices. As described in Chapter 4, each SCI must expose compatible entities via Home Assistant to be configured. We support commonly used parameters like numeric, boolean, and categorical settings. For now, devices that require other types can therefore only be represented partially. Additional types could be added, but would require corresponding controls, validation rules, and Home Assistant entity mappings. Nevertheless, the prototype supports all SCIs that depend on these settings.

Device support depends on an existing Home Assistant integration.

However, as we allow all protocols that Home Assistant supports and do not enforce specifications, an SCI can be implemented in various ways. An SCI could, for example, report the requested value while still moving or fail to confirm that the target state has been reached. Likewise, the configured transition is only an estimate. This limited access to actual physical data also limits the provided protection. Blocking rules check conditions before execution, but cannot detect new obstacles during movement or send an emergency stop command. These functions must be provided by the underlying device control system. However, we have no way of verifying whether this has actually happened.

Reported settings do not confirm completed or safe physical movement.

Even the checks and records that we provide in the prototype depend on the underlying device implementation. Unreported device changes, or certain unrecorded events triggered through Home Assistant, may not appear in the

External changes can leave gaps in the history and affect restoration.

history as they cannot be detected. In this case, the history values are outdated, and restoring a previous configuration would not produce the expected result. The history can therefore support recovery for recorded changes, but it cannot be treated as a complete log of all device activity.

Guest access and
conversational
assistance remain
outside the evaluated
scope.

Restricted guest access and the conversational assistant were mentioned during the early interviews but remained outside the implemented scope. We prioritized developing and evaluating the core functions for state-based control, feedback, and recovery. Guest access would require separating which controls different users can use, while a conversational assistant would need to translate requests into changes. Both features would require additional design, implementation, and evaluation and remain possible extensions for future work.

Chapter 7

Conclusion and Future Work

This thesis investigated how understandability, transparency, and user control of SCIs can be established in smart-home environments. As SCIs are largely heterogeneous, it is difficult to design a shared interface. At the same time, users still need to understand what a configuration means, why actions are blocked, and how to make them available again.

To answer the research question, we conducted a series of iterative, user-centered interviews. The exploratory interviews established design requirements based on participants' expectations. A second interview phase built on the sketches and refined the initial interaction concept derived from the first phase. We then used this concept to create a web-based Home Assistant add-on. The resulting prototype combines named states, access to individual settings, room and group control, configurable status rules, and a history with restoration functions. The final evaluation examined the practical use and benefits through tasks.

The feedback from the evaluation phase shows that states provide an accessible starting point for everyday interaction. Participants especially liked being able to apply configurations without understanding every parameter of the SCI, while still having easy access to the most important

The thesis investigates understandable, transparent, and controllable SCI interaction.

Three study phases informed the interaction concept and prototype.

Participants valued named states, accessible settings, configurable rules, and recovery.

settings. They also valued the configurable rules, which allowed them to include additional sensors and restrict control to their specific environment. The ability to restore previous configurations was seen as a useful addition that encouraged exploration.

Configuration and detailed feedback remained more demanding.

However, the evaluation also revealed difficulties with technical terms, state selection and activation, and the distinction between room states and groupings. The history presented too much information at once, and its different restoration controls were not always clear. These findings indicate that providing controls and feedback is insufficient unless their purpose and context are easy to understand. The small, partly recurring sample and the evaluation without physical SCIs further limit the conclusions to the studied interface interactions.

The thesis connects design requirements with an implemented and evaluated control concept.

Taken together, the findings provide empirically grounded guidance for designing shared control interfaces for heterogeneous SCIs. The contributions suggest an answer to the research question. Meaningful states can make everyday control understandable. At the same time, easy access to important settings allows users to adapt these configurations to their needs. To support informed decisions, the interface should also explain restrictions in context. If a change produces an unwanted result, recovery functions help users to restore the previous configuration. The prototype combines these mechanisms, providing a foundation for future work.

7.1 Future Work

The findings point to several directions for future work, ranging from immediate interface improvements to extensions that require evaluation with physical SCIs.

Reusable configurations could reduce the technical knowledge required.

First, future work should simplify device and rule configuration by replacing technical terminology with more familiar descriptions. Reusable device configurations could further reduce this effort. Manufacturers or other users could provide templates with entity mappings, setting names,

and predefined states, thereby completing most of the device setup. End-users could then import the template instead of configuring each device manually.

Several interaction workflows also require improvement. State cards should communicate more clearly how users select and apply a configuration. Combining room states and groupings into a common workflow could make coordinated actions easier to understand. The history could first present a brief overview of each change. Individual parameter values would then be available on demand with an additional click. Recording the triggering automation or event would additionally help users understand why an automated change occurred.

The first two interview phases also indicated interest in guest access and conversational assistance. These supplementary features could extend the prototype. Guest access could furthermore be expanded with more detailed permissions and temporary device reservations. For example, a user could reserve a shape-changing pillow while sleeping on it, preventing others from changing its state during use. A conversational assistant could translate requests into inspectable state suggestions that users must confirm before execution, as explored in the initial sketches.

These extensions should then be evaluated with physical SCIs in actual households. Long-term deployments could also reveal how device sharing and changes in routines affect the use of the interface. Finally, a comparison with conventional parameter-based interfaces could examine to what extent state-based interfaces simplify control. Together, these studies could determine how well the findings transfer to the everyday use of physical SCIs.

Clearer workflows could make everyday control easier to understand.

Guest access and conversational assistance could extend the core controls.

Household studies could examine how the approach works in everyday life.

Appendix A

Exploratory Interview Guide

The interviews were conducted in German and followed a semi-structured format. The questions below have been translated into English and served as a guide, while their wording and order could vary. Most questions were addressed in each interview. However, individual questions were skipped when they had already been answered during the conversation or were not relevant to the participant. Additional follow-up questions were asked when clarification was needed.

A.1 Understanding of SCIs and Their Potential Uses in the Home

- If you had a smart home, would you integrate shape-changing objects into it, and for what purposes could they be useful?
- What was the most unusual or unfamiliar device you have ever used?
- Have you ever felt afraid or uncomfortable because of a device you encountered in a smart home?

- Have you seen devices in someone else's home that you were skeptical about and would not install yourself?

A.2 Challenges When Adopting and Configuring Devices

- Have you encountered any difficulties when setting up a new device after adding it to your smart-home system?
- What kind of support or guidance did you need to set up the device successfully?

A.3 Direct, Remote, and Automatic Control of Transformations

- Are there situations in which you would prefer to control a device using your smartphone rather than interacting with it directly?
- How do you feel about passive triggers that automatically perform actions without requiring direct interaction?
 - *Follow-up:* Would person recognition be useful in this context?
- Would you trust a device to perform the requested action, or would you expect direct feedback? How should an upcoming physical transformation be announced, and would a preview of its expected result be helpful?

A.4 Trigger-Action Automations

- Which recurring activities in your home would you like to automate, and would a simple if-this-then-that logic be sufficient for these automations?

- Have you experienced situations in which an if-this-then-that rule did not produce the intended result?
 - *Follow-up:* Who or what was responsible for the problem, should the system have prevented it, and how should it communicate which rule or event triggered the action?

A.5 Adjustability of Physical Properties and Behaviors

- Which properties or settings of a device do you usually adjust after its initial configuration?
- How much control or adjustability do you expect, and should devices behave differently depending on the time of day, for example more gently in the evening than in the morning?

A.6 Coordinated Control of Multiple SCIs

- Do you find it frustrating to configure devices individually, and would you like to create groups of devices to which the same settings can be applied simultaneously?

A.7 Sensor Data and Logs for Understanding Device Behavior

- Which usage data would you consider relevant for a shape-changing cushion? Should an action history record what happened, when it happened, and who or what triggered it, or should it also retain contextual information, such as when someone leaned against the cushion? Should such a history allow users to trace and reverse unwanted actions?
- Do you frequently use multiple sensors to determine whether a particular action should be triggered?

A.8 Long-Term Adaptation of States and Device Configurations

- Do you regularly readjust your devices, or do you configure them once and then continue using them without further changes?
- Which settings should remain adjustable during long-term use?

A.9 Safety Constraints, Perceived Control, and Guest Access

- How do guests interact with the functions of your smart home? Do they encounter any problems?
- What safeguards would you like to have so that guests cannot damage anything and do not feel uncomfortable using the smart home? Under which circumstances should an action be blocked, and how should the system explain the cause and a possible resolution?
- Would you feel afraid or uncomfortable if objects in your home moved without you actively initiating the movement? What information or controls would help you remain in control?

A.10 Conversational Assistance

- Would you prefer to control individual settings yourself, use predefined moods or states, or interact through a chatbot?
- Should a chatbot execute changes directly or first present an inspectable suggestion for confirmation?

A.11 Structure of the SCI Control Interface

- What do you expect from the user interface: a simple and uncluttered default view with only frequently used controls, or immediate access to all available settings, even if some of them might be difficult to understand?
- Would you prefer to see all devices from the same physical room together in one virtual room, with separate pages for individual devices, or should each device have its own separate virtual space?

Appendix B

Concept Interview Guide

The interviews were conducted in German and followed a semi-structured format. The questions and tasks below have been translated into English and served as a guide, while their wording and order could vary. Most questions were addressed in each interview. However, individual questions were skipped when they had already been answered during the conversation or were not relevant to the participant. Additional follow-up questions were asked when clarification was needed.

Participants were shown two sets of low-fidelity sketches (see Appendix A) representing alternative interaction concepts. They were initially given approximately three minutes to examine the sketches independently. Because the sketches were not interactive, participants completed the tasks by pointing to interface elements and describing the sequence of actions they would perform.

B.1 Initial Impression and Comprehensibility

- I am going to show you a set of interface sketches. Please describe your first impression and identify

anything that appears unclear or confusing at first glance.

- If you were looking for a particular function, do you think you would be able to find it intuitively within the presented interface?

B.2 Tasks

Participants completed the following tasks after examining the sketches for approximately three minutes. For each task, they were asked to point to the relevant interface elements and describe each interaction step.

B.2.1 Adding a New Shape-Changing Device

Imagine that you have obtained a new shape-changing device and want to add it to your smart-home system. Starting from the overview shown in the sketches, point to the interface elements you would use and describe the complete sequence of actions you would take. Explain which information and configuration options you would expect during the setup process and how you would determine that the device had been configured successfully.

B.2.2 Locating a Specific Device and Function

Starting from the main overview, locate the device specified by the interviewer and navigate to its controls. Then locate the requested device function or setting. Describe every step you would take and explain where you would expect to find the relevant control if it is not immediately visible.

B.2.3 Modifying a Device State

Select one of the states shown for the device specified by the interviewer. First, explain how you would determine what the state will do before applying it. Then describe how you would access and modify its underlying settings and how you would save the resulting configuration as an updated or new state.

B.2.4 Responding to a Blocked Action

Imagine that you attempt to apply a device state, but the action is blocked because a person or object is in the device's movement area. Using the sketches, explain how you would identify why the action was blocked and what you would do to resolve the problem.

B.3 State-Centric Interaction

- The interface focuses on meaningful device states rather than technical parameters. Would you prefer this approach, or would you rather configure every parameter individually?
- In which situations would you still want to access or adjust the underlying parameters?

B.4 Explanation and Preview of States

- When selecting a state, how certain would you want to be about what the device will do afterward? Would you prefer a textual description, a static image of the target state, an animation, or the option to try the state directly?
- What advantages or disadvantages would these different forms of preview have for you?

B.5 Transitions, Constraints, and Safety

- While a device is changing its shape, should the interface display its current intermediate state, or would showing only the start and target states be sufficient?
- If an action cannot be performed, would a generic error message be sufficient, or should the interface provide a specific explanation of the cause? How technical should that explanation be, and should it include a possible solution?
- If an action is blocked because it could damage a person, the device, or another object, should the restriction remain binding or should users be allowed to override it?
- Should less severe restrictions be treated differently, for example when a sensor might be incorrect or temporarily malfunctioning?

B.6 Consistency Across Heterogeneous Devices

- Would you expect different shape-changing devices to be controlled through interfaces with the same general appearance and structure, or would it be acceptable for the interface to vary between devices?
- Which interface elements should remain consistent, even when the available device functions differ?

B.7 Scalability and Long-Term Use

- Imagine that you had 20 different shape-changing devices in your home. Would the presented interface still remain clear and manageable?
- How would you expect to find a specific device or control several devices together in a larger installation? Which additional navigation or coordination features would you need?

- Which aspects of the interface might become frustrating over time? Could any interaction become too slow or require too many steps?
- Would you want optional modules or modes providing additional settings over time, while keeping the initial interface simple?

B.8 Supplementary Concepts

- Would you prefer automatically suggested states, for example based on the time of day, or would you rather control everything manually?
- Would you give the system or a conversational assistant more control if it proved reliable, or would you still want to inspect and confirm every suggested change?
- How do you assess the presented guest-access concept? Would it be a central part of the interface or a supplementary feature used only in particular situations?

B.9 Concluding Question

- What would you change about the interface before it was released?

Appendix C

Prototype Evaluation Guide

The prototype evaluation was conducted in German. The task descriptions below have been translated into English. All participants completed the same ten tasks in the listed order. The tasks were designed to cover the central interaction flows of the prototype and generally became progressively more difficult and technical, followed by a final cleanup task.

Participants interacted with the functional prototype themselves. Prepared test data and Home Assistant helpers represented the shape-changing devices because the remote study setup did not include physical devices. Consequently, participants could interact with the complete control interface but could not observe actual physical transformations. Names and entity identifiers shown in quotation marks reproduce the labels used in the test environment.

C.1 Prerequisites

Before beginning the evaluation, the following test environment was prepared:

- The prepared test data was loaded.
- Home Assistant was running and connected to the prototype.
- A helper representing the desk height, named “Studie - Schreibtisch Höhe”, was available.
- A sensor representing a blocked movement area, named “Studie - Schreibtisch Kollision”, was available.

The tasks had to be completed in the order presented below.

C.2 Evaluation Tasks

1. Create a Room and Add a Device

Create a new room named “S-Arbeitszimmer”. Then add a device named “S-Schreibtisch” to this room. Connect at least its desk-height setting to the corresponding Home Assistant entity “Studie - Schreibtisch Höhe”.

Completion criterion: The room and device have been created, and the desk-height helper has been connected to the device.

2. Find a Device Using the Global Search

You want to move the bed into its reading position without first navigating through the room overview. Search for “Adaptives Bett”, open the corresponding device, and apply the state “Lesen”.

Completion criterion: The bed has been found using the global search and the state “Lesen” has been applied.

3. Control a Device Individually

You want to work while standing at the existing device “Adaptiver Tisch” in the living room. Set its height to 105 cm, enable collision protection, and select the slow movement speed. Then apply the settings.

Completion criterion: All three values have been applied as the current settings of the table.

4. Configure a Device Setting

The device “S-Schreibtisch” should only be used within a safe height range. Adjust its configuration so that the desk height can only take values between 70 and 120 cm. Rename the setting to “Tischhöhe” and configure it to appear among the device’s individual settings.

Completion criterion: The setting has been renamed, is visible among the individual settings, and is restricted to values between 70 and 120 cm.

5. Create and Apply a Device State

You want to reuse the configuration of “Adaptiver Tisch” created in Task 3. Save its current settings as a new state named “S-Fokusarbeit” and add a short description, such as “Test”. First apply the state “Standard”, and then apply the newly created state “S-Fokusarbeit”.

Completion criterion: The state “S-Fokusarbeit” exists and has restored the table height to 105 cm.

6. Create and Apply a Room State

Several devices in the living room should be configured together for a movie night. Create a room state named “S-Filmabend”. Configure “Adaptives Sofa” to use the state “Entspannung” and “Adaptiver Tisch” to use the state “Standard”. Then apply the room state.

Completion criterion: Both devices have changed to their assigned states.

7. Group Devices from Different Rooms

You want to prepare a configuration that affects devices across multiple rooms. Create a group named “S-Barrierearmer Alltag”. Configure “Adaptive Arbeitsfläche” in the kitchen to use the state “Barrierearm” and “Adaptives Bett” in the bedroom to use the state “Lesen”. Then apply the group.

Completion criterion: Both devices have changed to their assigned states despite being located in different rooms.

8. Trace and Reverse a Change

You notice that “Adaptiver Tisch” has accidentally been adjusted. Find the most recent relevant history entry for the device. Use the search and filter functions if necessary. Then reverse only the change made to this individual device.

Completion criterion: The relevant history entry has been found and the change to the individual device has been reversed.

9. Create a Safety Rule

The device “S-Schreibtisch” in “S-Arbeitszimmer” should no longer move when an obstacle is detected. Create a status rule for the device and connect it to the sensor entity “Studie - Schreibtisch Kollision”. Use the message “Bewegungsbereich blockiert. Alle Gegenstände unter dem Tisch müssen entfernt werden”. When the rule is active, it must prevent the desk from moving.

Completion criterion: The rule has been saved, the desk can no longer be controlled while the rule is active, and the blocking message is displayed.

10. Remove Content That Is No Longer Needed

The room created during the first task is no longer needed. Delete the device “S-Schreibtisch” and then delete the room “S-Arbeitszimmer”.

Completion criterion: Neither the device nor the room remains in the system.

Appendix D

Design Sketches

This appendix presents the sketches created after the first interview phase. The sketches are shown in the order in which the interaction can be followed: from the overview to room-level and device-level, followed by views for history and the conversational assistant. The labels inside the sketches remain in German because these were the versions presented to participants.

The collection documents the main interaction concepts. It does not include every device view shown to the participants, as this would result in a large number of sketches. Arrows and buttons illustrate the intended navigation and actions.

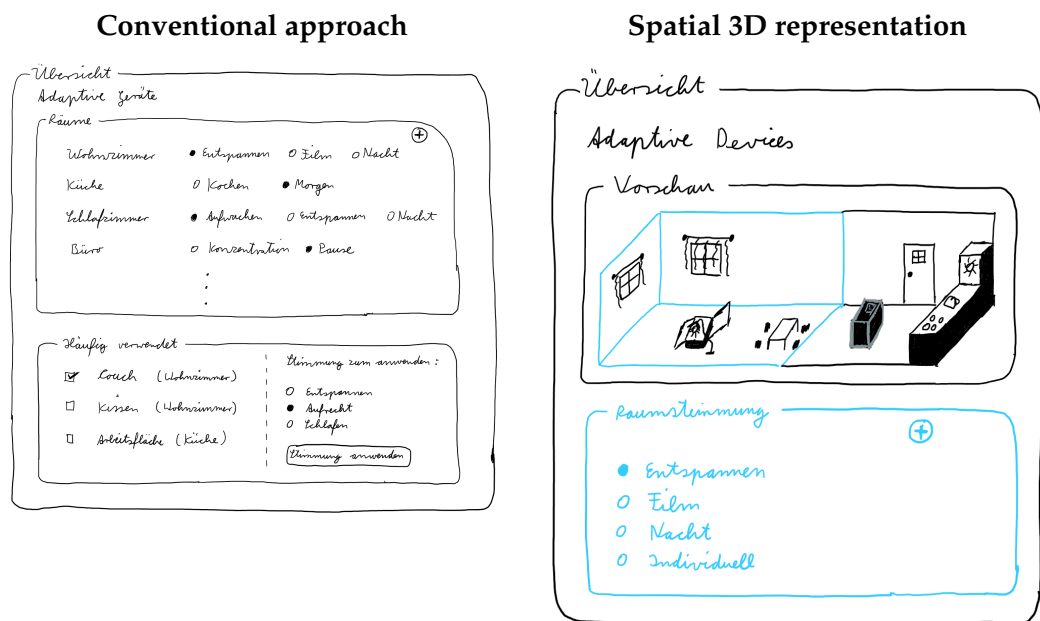


Figure D.1: Comparison of the general overview. The state-centric variant organizes rooms, states, and frequently used devices as conventional controls, while the 3D variant uses a spatial home preview as its primary entry point.

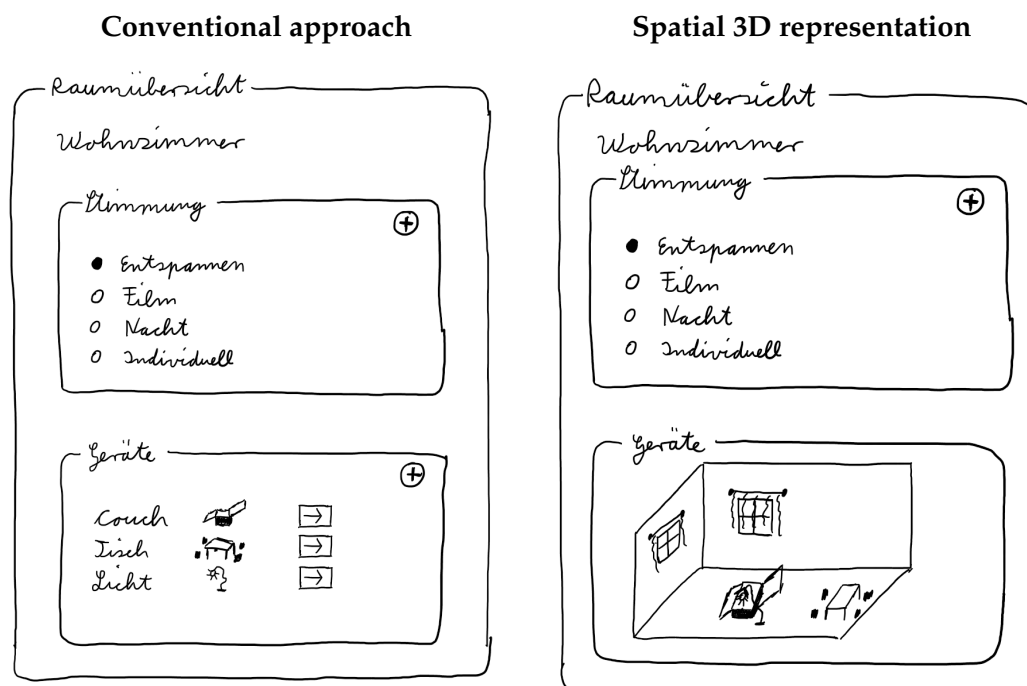


Figure D.2: Comparison of the room view with a room state selected. Both variants retain the same state-centric interaction while presenting the devices either as entries or within a spatial preview.

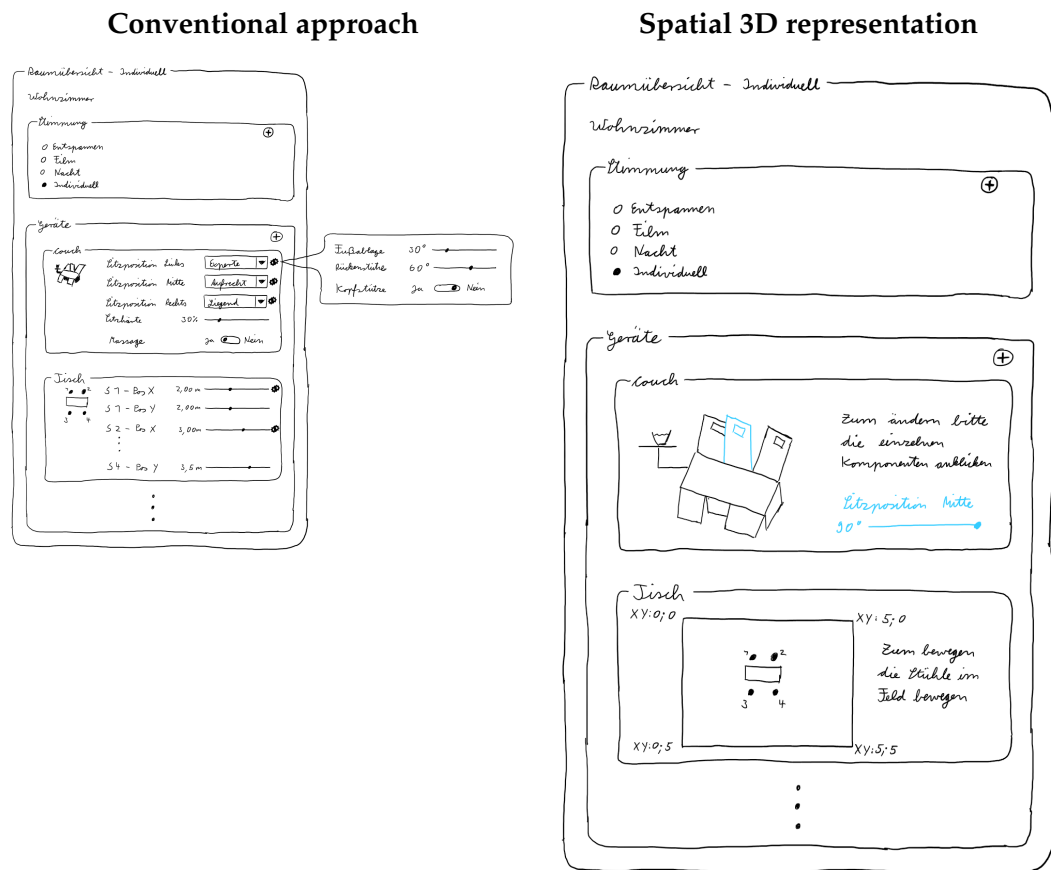


Figure D.3: Comparison of individual control at room level. The state-centric variant exposes conventional controls for several devices, whereas the 3D variant connects the controls directly to visual device components and positions.

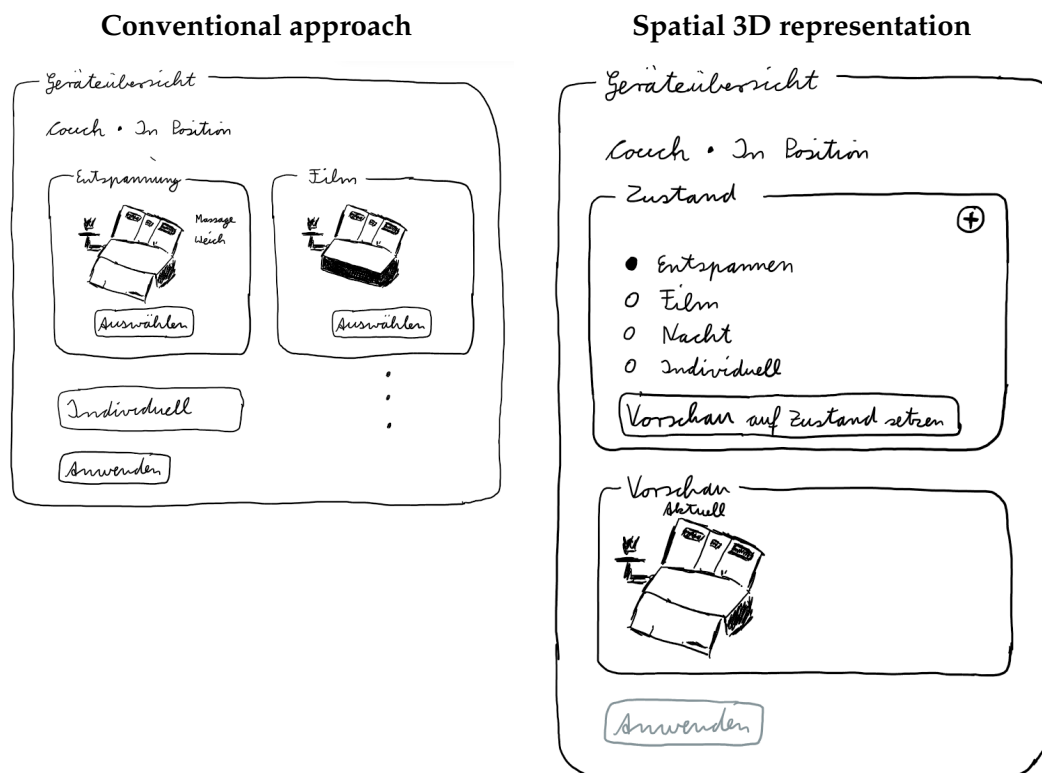


Figure D.4: Comparison of the stationary device view. The state-centric variant presents target states as cards, while the 3D variant combines state selection with a larger device preview.

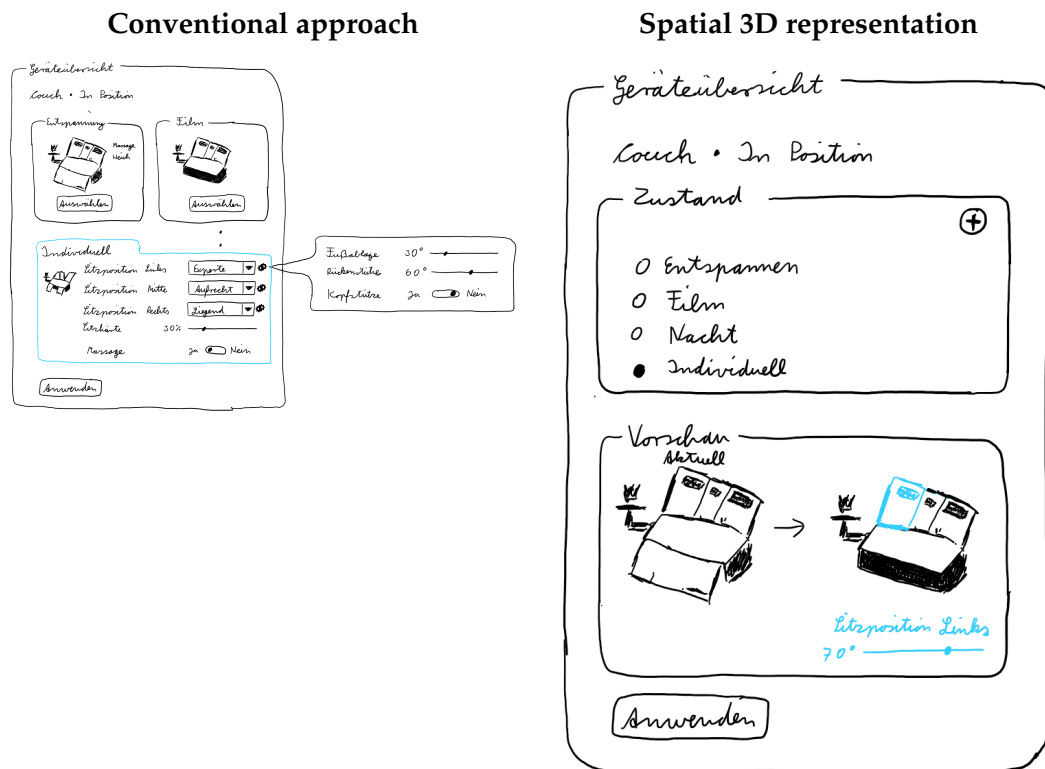


Figure D.5: Comparison of individual device control. Both variants provide access to detailed parameters but differ in whether these are presented as conventional controls or linked to the visual representation of the device.

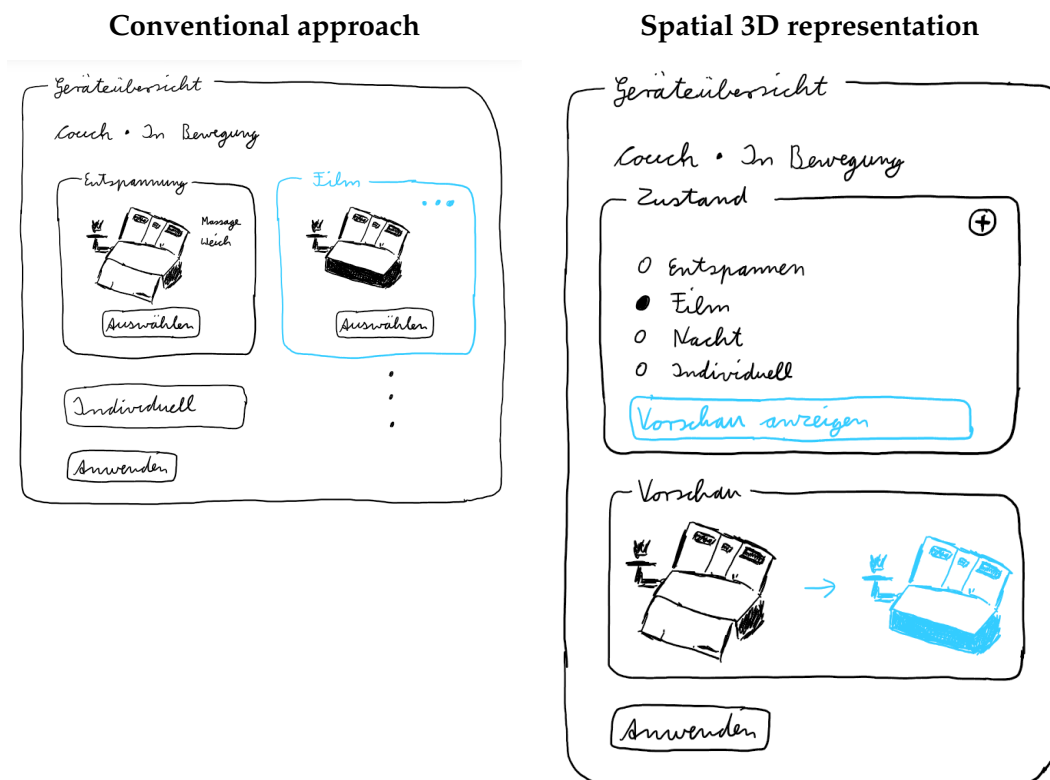


Figure D.6: Comparison of the device view during a transition. Both variants highlight the selected target state and communicate that the physical transformation is still in progress.

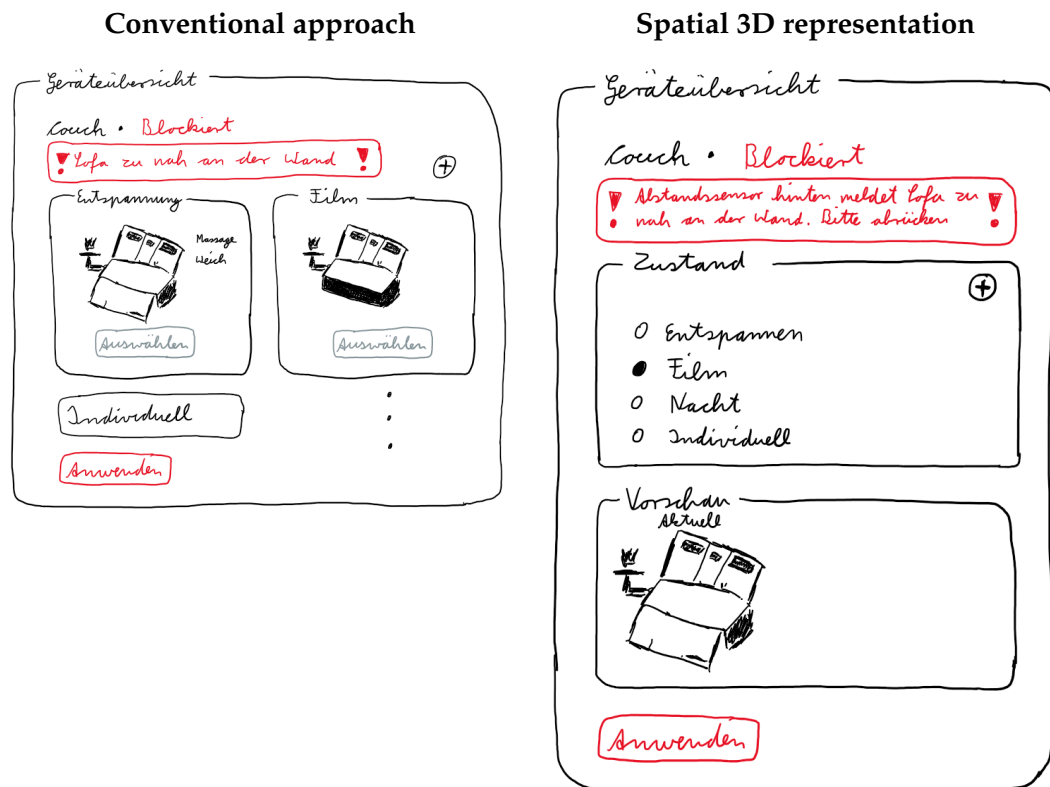


Figure D.7: Comparison of a blocked device action. Both variants explain the restriction and disable the affected controls while retaining their respective state-centric and spatial presentations.

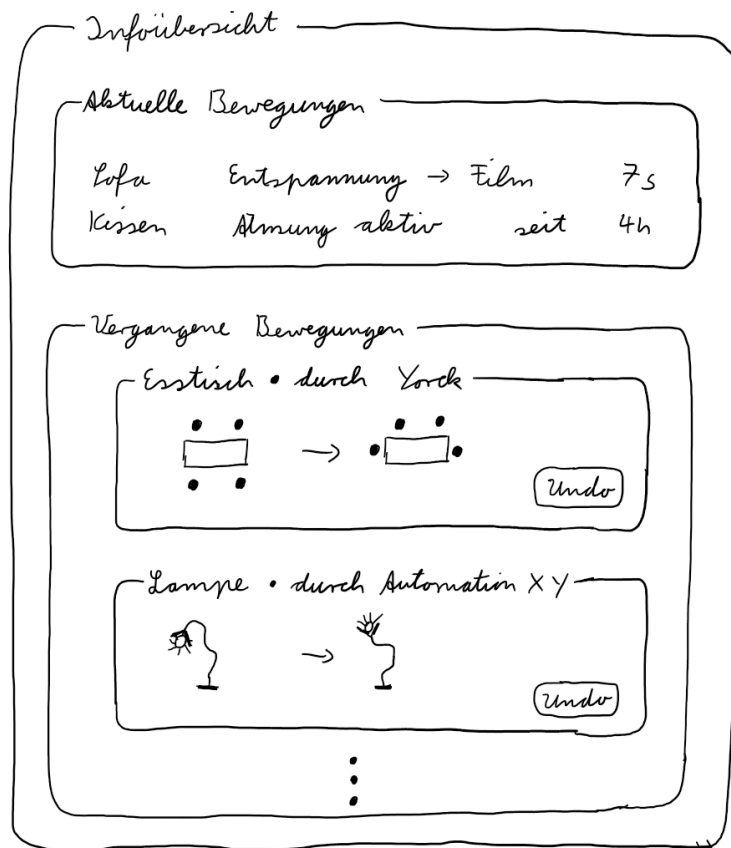


Figure D.8: Shared information and history view showing ongoing transitions and previous actions. This view was used with both interface variants and therefore appears only once.

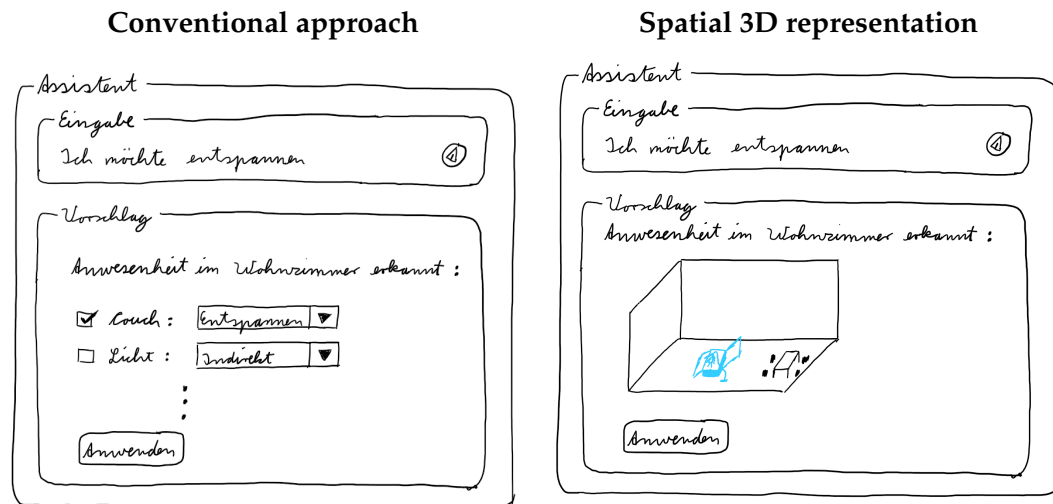


Figure D.9: Comparison of the conversational assistant. Both variants translate a natural-language request into inspectable suggestions but present the proposed changes through their respective conventional and spatial representations.

Appendix E

Prototype Screenshots

This appendix presents selected views of the implemented prototype. The screenshots document the central workflows discussed in Chapter 4. The interface language is set to English, while user-defined room and device names as well as their descriptions remain in German. Figures E.1–E.3 illustrate navigation from the overview to individual device control. Figures E.4–E.6 show device configuration and contextual restrictions, while Figures E.7–E.8 document combined control and recovery.

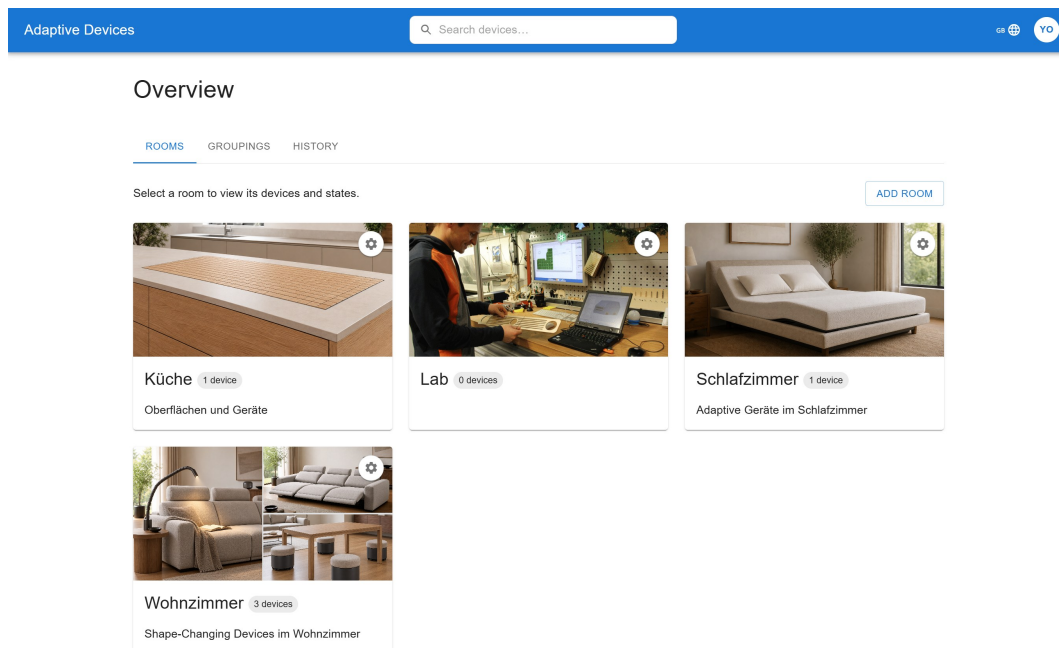


Figure E.1: Prototype overview organizing the configured devices by room.

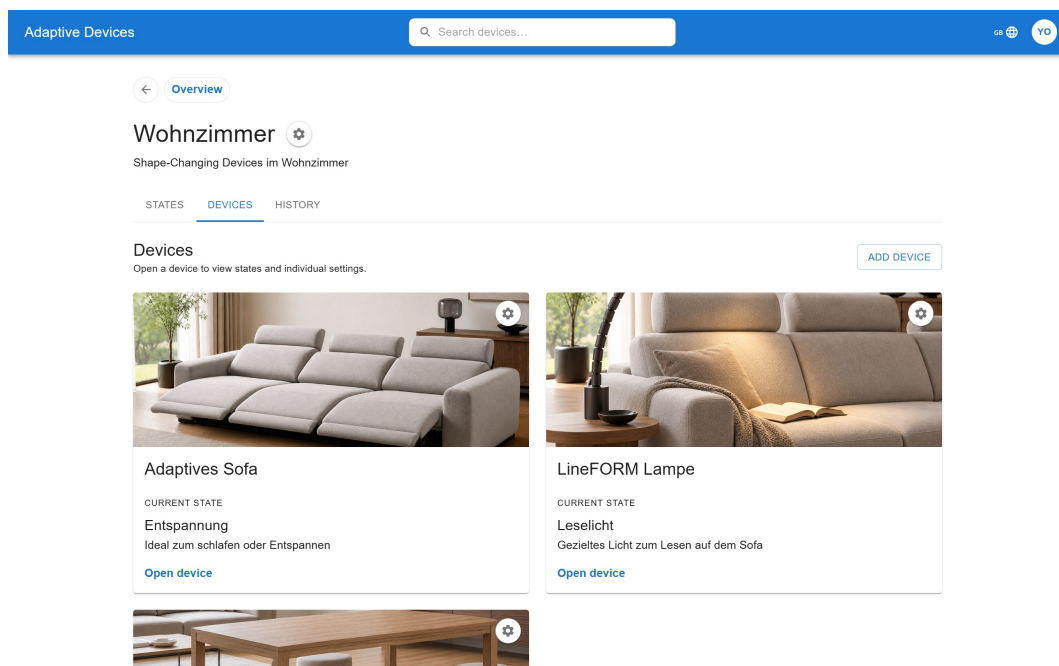


Figure E.2: Room view showing the devices assigned to the selected room.

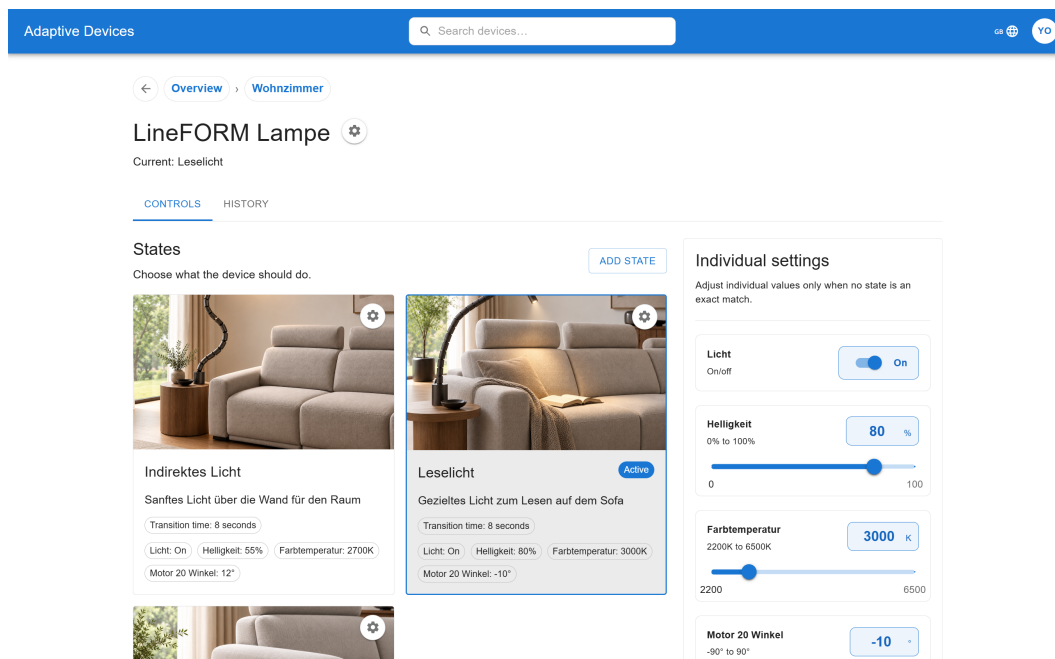


Figure E.3: Device view combining named states with direct access to selected individual settings.

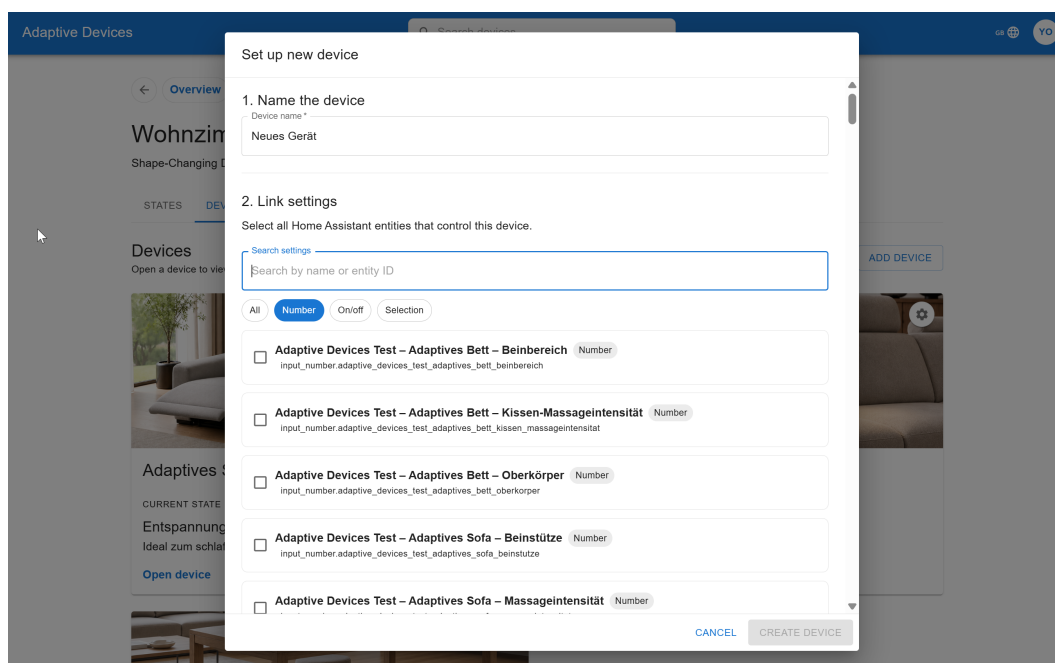


Figure E.4: Device configuration view for linking Home Assistant entities to SCI settings.

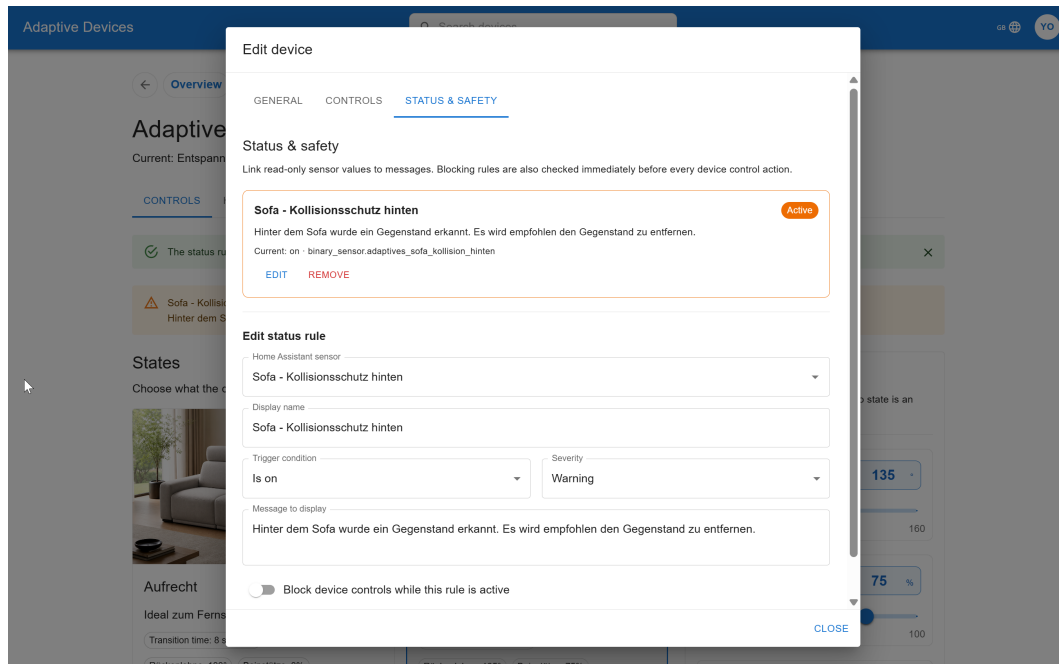


Figure E.5: Configuration of a contextual status rule and its effect on device control.

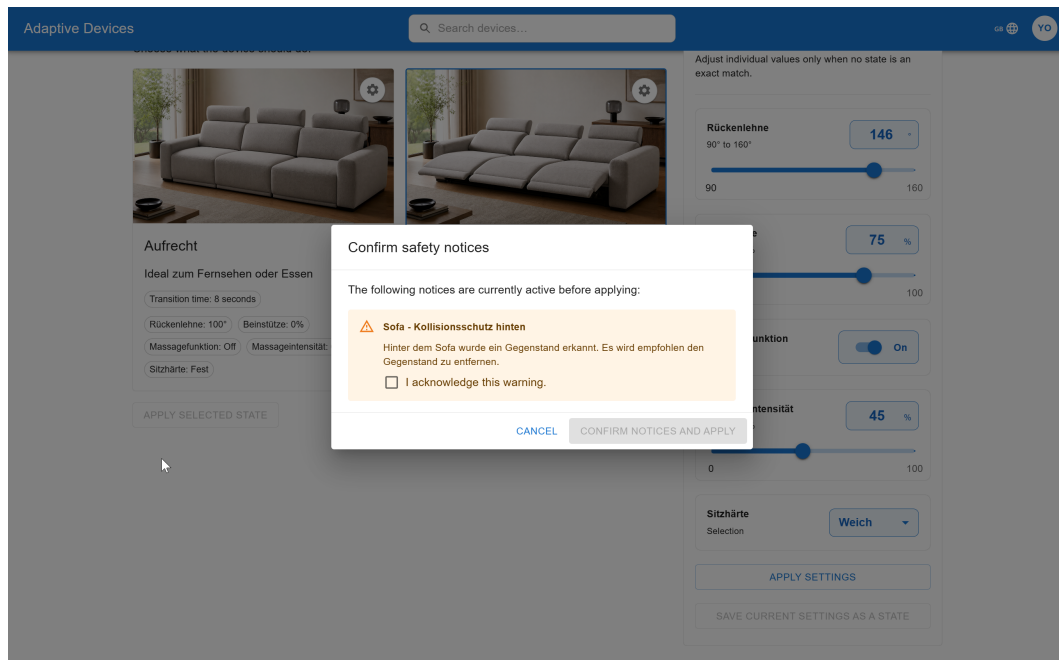


Figure E.6: Interface feedback explaining why a requested device action needs confirmation.

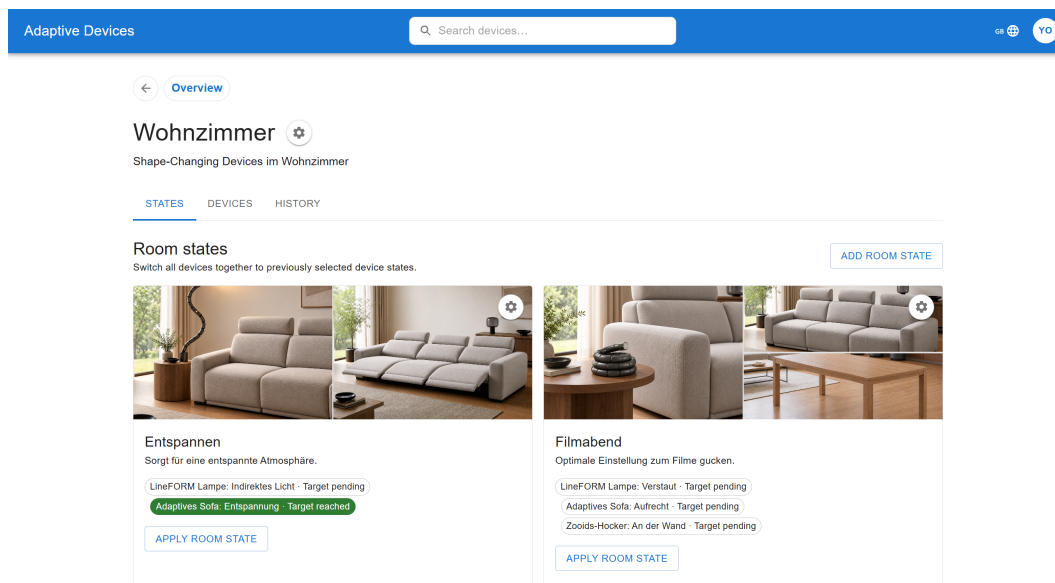


Figure E.7: Room-state view combining target states for multiple devices.

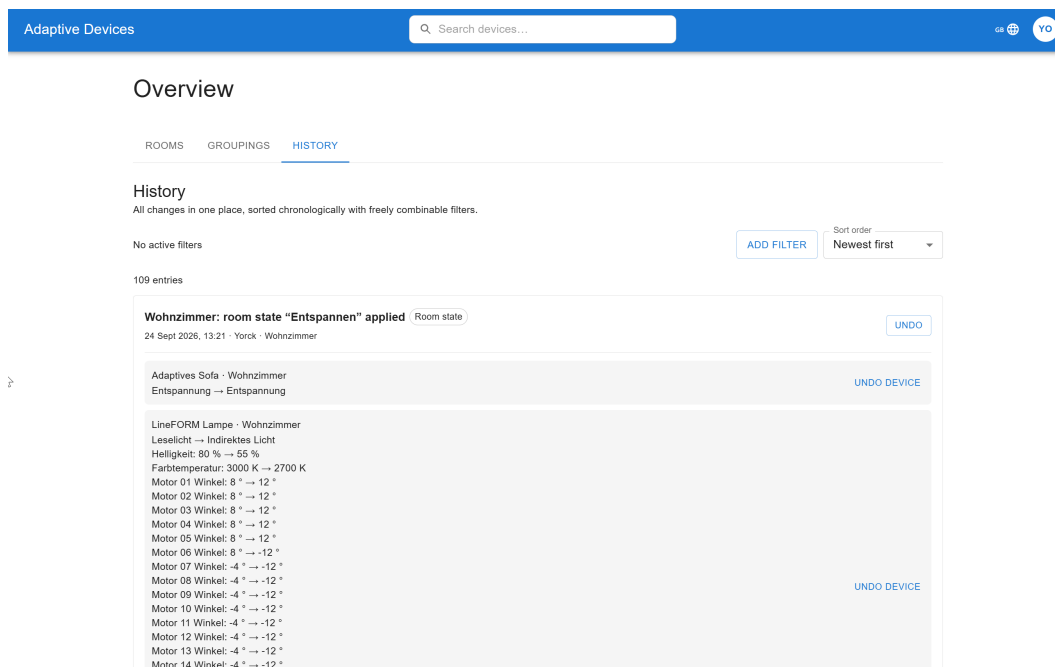


Figure E.8: Control history showing a previous configuration change and its restoration option.

Bibliography

- [1] Jason Alexander, Anne Roudaut, Jürgen Steimle, Kasper Hornbæk, Miguel Bruns Alonso, Sean Follmer, and Timothy Merritt. Grand Challenges in Shape-Changing Interface Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, New York, NY, USA, 2018. Association for Computing Machinery. doi.org/10.1145/3173574.3173873.
- [2] A. J. Bernheim Brush, Bongshin Lee, Ratul Mahajan, Sharad Agarwal, Stefan Saroiu, and Colin Dixon. Home Automation in the Wild: Challenges and Opportunities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2115–2124, New York, NY, USA, 2011. Association for Computing Machinery. doi.org/10.1145/1978942.1979249.
- [3] Michael Chamunorwa, Heiko Müller, and Susanne Boll. The Hidden Potential: Reimagining Household Items as Smart Home Interfaces. In *Proceedings of the International Conference on Mobile and Ubiquitous Multimedia*, MUM '24, pages 519–522, New York, NY, USA, 2024. Association for Computing Machinery. doi.org/10.1145/3701571.3703371.
- [4] Scott Davidoff, Min Kyung Lee, Charles Yiu, John Zimmerman, and Anind K. Dey. Principles of Smart Home Control. In *UbiComp 2006: Ubiquitous Computing*, volume 4206 of *Lecture Notes in Computer Science*, pages 19–34, Berlin, Heidelberg, 2006. Springer. doi.org/10.1007/11853565_2.
- [5] Sarah Fischer, Makoto Itoh, and Toshiyuki Inagaki. Prior Schemata Transfer as an Account for Assessing the Intuitive Use of New Technology. *Applied Ergonomics*, 46:8–20, 2015. doi.org/10.1016/j.apergo.2014.05.010.
- [6] Daniela Fogli, Matteo Peroni, and Claudia Stefini. ImAtHome: Making Trigger-Action Programming Easy and Fun. *Journal of Visual Languages & Computing*, 42:60–75, 2017. doi.org/10.1016/j.jvlc.2017.08.003.
- [7] Sean Follmer, Daniel Leithinger, Alex Olwal, Akimitsu Hogge, and Hiroshi Ishii. inFORM: Dynamic Physical Affordances and Constraints through Shape and Object Actuation. In *Proceedings of the 26th Annual ACM Symposium on*

- User Interface Software and Technology*, UIST '13, pages 417–426, New York, NY, USA, 2013. Association for Computing Machinery. doi.org/10.1145/2501988.2502032.
- [8] Franca Hübner. Co-Designing Shape-Changing Interface Scenarios in Smart Homes. Bachelor's thesis, RWTH Aachen University, Aachen, Germany, 2025.
- [9] Mathieu Le Goc, Lawrence H. Kim, Ali Parsaei, Jean-Daniel Fekete, Pierre Dragicevic, and Sean Follmer. Zooids: Building Blocks for Swarm User Interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 97–109, New York, NY, USA, 2016. Association for Computing Machinery. doi.org/10.1145/2984511.2984547.
- [10] Brian Y. Lim, Anind K. Dey, and Daniel Avrahami. Why and Why Not Explanations Improve the Intelligibility of Context-Aware Intelligent Systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 2119–2128, New York, NY, USA, 2009. Association for Computing Machinery. doi.org/10.1145/1518701.1519023.
- [11] Ken Nakagaki, Sean Follmer, and Hiroshi Ishii. LineFORM: Actuated Curve Interfaces for Display, Interaction, and Constraint. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology*, UIST '15, pages 333–339, New York, NY, USA, 2015. Association for Computing Machinery. doi.org/10.1145/2807442.2807452.
- [12] Muhammad Omar. From Context to Shape: AI-Driven Translations for Shape-Changing Interfaces. Master's thesis, RWTH Aachen University, Aachen, Germany, 2026.
- [13] Marianne Graves Petersen, Majken Kirkegård Rasmussen, and Johan Trettvik. Affordances of Shape-Changing Interfaces: An Information Perspective on Transformability and Movement. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference*, DIS '20, pages 1959–1971, New York, NY, USA, 2020. Association for Computing Machinery. doi.org/10.1145/3357236.3395521.
- [14] Majken K. Rasmussen, Esben W. Pedersen, Marianne G. Petersen, and Kasper Hornbæk. Shape-Changing Interfaces: A Review of the Design Space and Open Research Questions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 735–744, New York, NY, USA, 2012. Association for Computing Machinery. doi.org/10.1145/2207676.2207781.
- [15] Majken Kirkegård Rasmussen, Timothy Robert Merritt, Miguel Bruns Alonso, and Marianne Graves Petersen. Balancing User and System Control in Shape-Changing Interfaces: A Designerly Exploration. In *Proceedings of the 10th*

- Anniversary Conference on Tangible Embedded and Embodied Interaction*, TEI '16, pages 202–210, New York, NY, USA, 2016. Association for Computing Machinery. doi.org/10.1145/2839462.2839499.
- [16] Junaed Sattar and Gregory Dudek. Towards Quantitative Modeling of Task Confirmations in Human-Robot Dialog. In *2011 IEEE International Conference on Robotics and Automation*, pages 1957–1963, 2011. doi.org/10.1109/ICRA.2011.5979633.
- [17] Ben Shneiderman, Catherine Plaisant, Maxine Cohen, Steven Jacobs, Niklas Elmqvist, and Nicholas Diakopoulos. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Pearson, 6 edition, 2016. ISBN 978-0-13-438038-4.
- [18] Marie Louise Stisen Kjerstein Sørensen, Andreas Becker Bertelsen, Ninna Nordmark Hoffmann, Marianne Graves Petersen, and Eve Hoggan. MoLux: Negotiating Control with a Shape-Changing Lamp at Home. In *Participative Computing for Sustainable Futures: Adjunct Proceedings of the 12th Nordic Conference on Human-Computer Interaction*, NordiCHI '22, New York, NY, USA, 2022. Association for Computing Machinery. doi.org/10.1145/3546155.3547271.
- [19] Benjamin K. Sovacool and Dylan D. Furszyfer Del Rio. Smart Home Technologies in Europe: A Critical Review of Concepts, Benefits, Risks and Policies. *Renewable and Sustainable Energy Reviews*, 120:109663, 2020. doi.org/10.1016/j.rser.2019.109663.
- [20] Miriam Sturdee and Jason Alexander. Analysis and Classification of Shape-Changing Interfaces for Design and Application-Based Research. *ACM Computing Surveys*, 51(1):2:1–2:32, 2018. doi.org/10.1145/3143559.
- [21] Miriam Sturdee, John Hardy, Nick Dunn, and Jason Alexander. A Public Ideation of Shape-Changing Applications. In *Proceedings of the 2015 International Conference on Interactive Tabletops and Surfaces*, ITS '15, pages 219–228. Association for Computing Machinery, 2015. doi.org/10.1145/2817721.2817734.
- [22] John Samy Selim Tiab and Kasper Hornbæk. Understanding Affordance, System State, and Feedback in Shape-Changing Buttons. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 2752–2763, New York, NY, USA, 2016. Association for Computing Machinery. doi.org/10.1145/2858036.2858350.
- [23] Jonas Togler, Fabian Hemmert, and Reto Wettach. Living Interfaces: The Thrifty Faucet. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, TEI '09, pages 43–44, New York, NY, USA, 2009. ACM. doi.org/10.1145/1517664.1517680.

-
- [24] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman. Practical Trigger-Action Programming in the Smart Home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 803–812, New York, NY, USA, 2014. Association for Computing Machinery. doi.org/10.1145/2556288.2557420.
- [25] Jo Vermeulen, Kris Luyten, Elise van den Hoven, and Karin Coninx. Crossing the Bridge over Norman's Gulf of Execution: Revealing Feedforward's True Identity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 1931–1940, New York, NY, USA, 2013. Association for Computing Machinery. doi.org/10.1145/2470654.2466255.
- [26] Ali Zaidi, Anna Karanika, Ti-Chung Cheng, Yi-Shyuan Chiang, Camille Cobb, Indranil Gupta, and Karrie Karahalios. Control in Context: How Smart Home Users Navigate Proxy-Based Schemes. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*, CHI '26, New York, NY, USA, 2026. Association for Computing Machinery. doi.org/10.1145/3772318.3790549.

Index

Automations	44
Backend	42
Blocking status rules	37
Breadcrumbs	35
Collision detection	46
Constraint visibility	14, 37
Control history	38
Conversational assistance	24, 62
Database	42
Design requirements	24
- DR01: state-centric control	25, 35
- DR02: progressive access to detail	25, 35
- DR03: room-based navigation	25, 35
- DR04: group control	25, 37
- DR05: transition transparency	25, 38
- DR06: explainable constraints	25, 37
- DR07: recoverability	25, 38
- DR08: traceable automation	25, 44, 47
- DR09: limited data use	25, 41
- DR10: assistant support	25, 47
Device capabilities	8, 39
Device context	8, 41
ESPHome	33

Frontend.....	42
Global device search.....	35
Group control.....	37
Guest access.....	23, 62
Heterogeneous SCIs.....	2, 39
Individual settings.....	36
inFORM.....	8
Intuitive use.....	16
LineFORM.....	8
Named states.....	13, 39
Notiflex.....	10
Preview images.....	36
Recoverability.....	14, 59
Responsive design.....	35
Restoration.....	38
Room states.....	36
Room-based navigation.....	34
Scenes.....	44
Shape-changing interfaces (SCIs).....	1, 7
Smart-home environments.....	2, 11
State selection.....	36
State-centric interaction.....	13, 56
Status messages.....	37
Status rule severity.....	37
Status rules.....	34, 41
Target validation.....	43
Thrifty Faucet.....	8
Transition awareness.....	14, 40

Trigger-action programming	11
User control (direct, negotiated, indirect, system)	15
Users	43
Zooids	8

