# eSPACE: Leveraging Theoretical Foundations for the End-User Development of Cross-Device and IoT Applications

AUDREY SANCTORUM and BEAT SIGNER, WISE Lab, Vrije Universiteit Brussel, Belgium

In the rapidly evolving landscape of cross-device computing and the Internet of Things (IoT), there is a need for intuitive and user-friendly solutions empowering end users to create and tailor their applications. To address this need, we analysed the different metaphors for end-user development (EUD) of cross-device and IoT applications. A key observation is the lack of addressing end user's mental models when designing interactions among devices, resulting in less intuitive EUD tools. To fill this gap, we introduce eSPACE, an end-user authoring tool facilitating the development of cross-device and IoT applications. In contrast to most existing tools, eSPACE is grounded on strong theoretical foundations, including a conceptual model, reference framework as well as design guidelines and suitable metaphors. The effectiveness of these theoretical foundations in creating an intuitive and user-friendly EUD platform has been validated in a user study. Our study confirms eSPACE's potential as a useful and easy to use tool for end-user application development. In addition, we present potential future research directions, including automation functionality, intelligibility and human-AI interaction. In discussing these future directions, we aim to foster and advance EUD research towards more end-user-accessible authoring environments for cross-device and IoT applications.

CCS Concepts: • **Human-centered computing → Interactive systems and tools**; **Systems and tools for interaction design**; *Ubiquitous computing*.

Additional Key Words and Phrases: end-user development, end-user authoring tool, Internet of Things, cross-device interaction, metaphors

**ACM Reference Format:**
Audrey Sanctorum and Beat Signer. 2025. eSPACE: Leveraging Theoretical Foundations for the End-User Development of Cross-Device and IoT Applications. (2025), 51 pages.
X

## 1 INTRODUCTION

With the proliferation of smart devices and Internet of Things (IoT) technologies, the number of applications used to control them has increased significantly. Users are confronted with a diverse range of applications, each offering different (often limited) functionality and user interfaces (UIs) that vary in appearance, resulting in fragmented control and a lack of uniformity. Navigating through these different interfaces—which possess unique sets of controls and interaction patterns—adds complexity and hinders the efficient management and control of smart devices and IoT technologies.

In response to this problem, on the one hand, research has been conducted in the fields of cross-device interaction (XDI) and distributed users interfaces (DUIs), aiming to improve the interaction across different smart devices, such as smartphones, tablets, and smartwatches [12]. Research in these domains falls under the umbrella of cross-device computing, a term that will be used throughout this paper to describe the overarching field of study. On the other hand, researchers have also

Authors' address: Audrey Sanctorum, audrey.sanctorum@vub.be; Beat Signer, beat.signer@vub.be, WISE Lab, Vrije Universiteit Brussel, Pleinlaan 2, Brussels, Belgium.


ACM XXXX-XXXX/2025/1-ART
https://doi.org/XXXXXXX.XXXXXXX

This is the author's version and the final article is available at:
Audrey Sanctorum and Beat Signer. eSPACE: Leveraging Theoretical Foundations for the End-User Development of Cross-Device and IoT Applications. ACM Transactions on Computer-Human Interaction (TOCHI), 2025 (to appear)

focused on providing users with tools to easily control and manage their IoT devices [14]. However, existing solutions tend to focus on either providing cross-device computing solutions or IoT control, but rarely provide comprehensive solutions bridging the cross-device and the IoT research domains. Additionally, there is often a notable oversight in designing solutions that align with the end users' mental models when it comes to performing cross-device or IoT interactions.

Our research tackles these gaps by proposing a unified and intuitive solution that consolidates control over all smart home devices and *things* in a single platform. The approach aims to empower end users to create custom smart applications for the seamless control and management of their smart environments based on their mental models. To achieve this, we adopt an end-user development (EUD) paradigm [50], enabling users without technical expertise to develop apps that meet their specific needs. While we aim to unify the cross-device computing and the IoT research fields, we mainly focus on selecting fitting metaphors that resonate with end users' mental models regarding smart home interactions. This leads us to our main research question:

> *How can appropriate abstractions be employed to hide technical complexity, empowering end users to effectively control their smart environments and create personalised applications for managing their smart devices and things?*

In order to address this research question, we have further divided it into four subquestions. In this paper, we focus on answering the last two questions, while the first two questions have already been addressed in previous publications. Together, they provide a comprehensive answer to our main research question.

**Research Question 1 (RQ1)** *What are the main requirements for the end-user authoring of unified cross-device and IoT user interfaces?* This question was answered by thoroughly analysing related work in cross-device and IoT end-user development research, resulting in an initial set of requirements. The requirements were further refined by describing a use case scenario for cross-device and IoT interactions. Based on our requirements and insights from related work, we opted for a model-based approach and analysed existing model-based solutions, leading to additional requirements. The complete list of requirements for cross-device and IoT end-user authoring tools is available in Appendix A and a detailed description can be found in [75].

**Research Question 2 (RQ2)** *What are the necessary concepts and methods to address the requirements resulting from answering RQ1?* Given the difficulty of reusing, adapting, and extending existing solutions to support the requirements of RQ1, we built on knowledge from our background research to present new conceptual foundations. These foundations support all identified requirements and form the basis for developing EUD authoring solutions for cross-device and IoT applications, as detailed in [79].

**Research Question 3 (RQ3)** *Which metaphors or abstractions should be used on top of our conceptual foundation to allow end users to visualise and create their unified cross-device and IoT interactions?* To gain an overview of potential metaphors and abstractions, in Section 2 we review existing work in the field of cross-device computing and IoT solutions. We define each metaphor and present a table summarising the solutions using these metaphors. In order to incorporate the users' perspective on which metaphors they find most appropriate, we conducted a user elicitation study to identify the metaphors for IoT and cross-device interaction that best fit users' mental models. This study, combined with an analysis of existing guidelines from related work, resulted in several design guidelines for creating cross-device and IoT EUD authoring tools. A more detailed description of the study and the resulting design guidelines can be found in [76].

**Research Question 4 (RQ4)** *How can we design a unified cross-device and IoT EUD authoring tool given the requirements from RQ1, the conceptual foundations from RQ2 and the guidelines including the appropriate metaphors found in RQ3?* This final part aims to create a compliant proof-of-concept EUD authoring solution based on the artefacts resulting from the previous research questions. To this end, we built the *eSPACE* end-user authoring tool, detailed in Section 3. This section includes a brief overview of its theoretical foundations, functionality, (authoring) views, and instructions on how to use it to create cross-device and IoT applications. eSPACE was not only developed to demonstrate the practical application of our conceptual foundation, but also to validate the applicability of the design guidelines derived from RQ3. Consequently, we conducted a user study to evaluate its usability and user experience, hypothesising that by adhering to the design guidelines, we would achieve high levels of user satisfaction and usability. This evaluation is presented in Section 4. Since eSPACE allows users to define the functionality of their applications using either the pipeline metaphor or the rules metaphor, we also considered it relevant to analyse which metaphor users preferred. The results of our study provided valuable insights, leading to future improvements for eSPACE. The study also showed good usability and high user experience ratings, thereby confirming the applicability of our design guidelines. Additionally, it offered insights into future research directions, which we discuss in Section 5.

We summarise the contributions of this paper as follows:

- We provide an extensive overview of related work in end-user development (EUD) within the cross-device and IoT research domains, along with a summary of the most common metaphors used in these fields.
- We present the eSPACE end-user authoring tool, grounded in strong theoretical foundations.
- We validate the applicability of these theoretical foundations by evaluating eSPACE, confirming good usability, along with high levels of user satisfaction and experience.
- We introduce future research directions for EUD in cross-device computing and IoT research based on findings from our user study and related work.

The remainder of this paper is organised as follows. Section 2 offers a comprehensive overview of related work, highlighting existing EUD research efforts in cross-device computing and the Internet of Things. Section 3 presents the theoretical foundations and design principles that underpin the eSPACE end-user authoring tool. This section also describes the different views of the eSPACE tool and showcases its key features. In Section 4, we outline the evaluation methodology of eSPACE, along with the results and highlight the benefits and limitations of our approach. Potential future research directions are explored in Section 5, and Section 6 concludes the paper with a summary of our key findings.

## 2 RELATED WORK

The field of end-user development (EUD) has gained considerable attention over the last years by offering a "*set of methods, techniques, tools, and socio-technical environments that allow end users to act as professionals in those ICT-related domains in which they are not professionals, by creating, modifying, extending and testing digital artefacts without requiring knowledge in traditional software engineering techniques*" [2]. In this section, we first present an overview of end-user development tools that facilitate cross-device interactions, highlighting the metaphors employed to assist users. We then explore solutions that empower users to have more control over their IoT devices, discussing the used metaphors as well. We conclude by discussing the limitations of these approaches and emphasising the importance of selecting intuitive metaphors.

While our focus is the analysis of the metaphors used in end-user development systems for cross-device computing and IoT environments, Baricelli et al. [2] provide a broader overview of

EUD, end-user programming (EUP) and end-user software engineering (EUSE), covering the used techniques, application domains and target users involved.

### Authoring Cross-Device Interactions and Applications

In [80], we presented an overview of existing approaches in the domain of cross-device and distributed user interfaces (DUIs), and demonstrated the advantages of adopting an end-user development approach in this domain. Throughout the years, various EUD research has been conducted to allow end users to create and compose DUIs and cross-device interactions in multi-device environments. A popular approach that emerged proposed web-based mashups, such as in *SmartComposition* [46] and *DireWolf* [45], which support the distribution of UI components across different devices. The components are presented as pre-built widgets which can be arranged in a grid-based layout using the *drag-and-drop* metaphor. Inter-widget communication further enables cross-device cooperation and synchronisation. For the sake of completeness, we include the definition of the drag-and-drop metaphor:

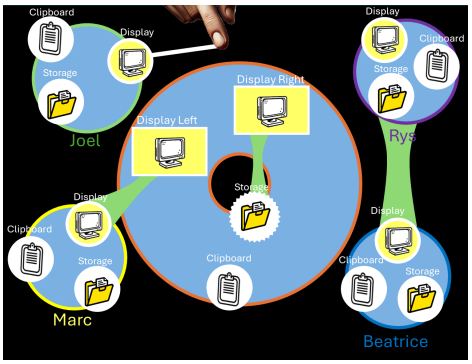
**Definition 2.1 – Drag-and-Drop Metaphor**

By using the drag-and-drop metaphor, one can interact with digital items by selecting, dragging, and dropping them to a target location.

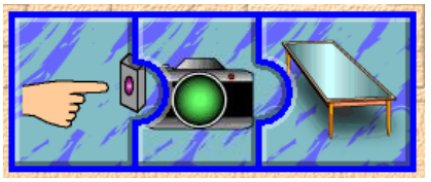
In addition to single-user mashup approaches, Ardito et al. [1] introduced an extension to their *EFESTO* mashup platform [26] supporting collaboration through distributed interactive workspaces (DIWs). While EFESTO already allowed CRUD operations on UI components and workspaces, the extension supports the publishing and versioning of resources on the common platform repository, as well as the definition of access rights. Sharing policies can be used to distribute entire DIWs or parts of them on multiple devices and among different users.

Other researchers also focused on improving collaboration in such distributed settings, including Blichmann et al. [10] who presented a permission management approach for the collaborative use of composite web applications. Thereby, users can share entire applications, components, specific features or data, while blocking some private data elements from being shared. Further, the authors provide an easy-to-understand visualisation of the existing permission assignments and the current state of all sharings. While no particular metaphor has been used to allow cross-device interaction or UI distribution, since these systems provide a (simple) point-and-click user interface, Blichmann et al. [10] explored a triple-based overview to facilitate the sharing mechanism for end users. This consists of allowing users to create sharing definitions by visually composing a new triple which indicates *who* will have access, *what* is shared and *how* this component will be shared (i.e. view only or edits possible).

Focusing on improving collaboration between multiple users and platforms, Pering et al. [70, 71] presented the *Platform Composition* approach to allow different services and resources of existing applications to be used in a distributed setting. The prototype has been showcased with a collaborative workspace table prototype, including four laptops, two embedded displays, and a central embedded control surface. In order to share resources between different devices, users can use a *join-the-dots* graphical user interface, which shows devices as large circles enclosing smaller circles representing the device's core services, as illustrated in Figure 1a. By simply drawing a line from a service to a target device, resources between different devices can be shared. This way, the clipboard service from a user's laptop could, for example, be connected to the clipboard of another user to easily copy/paste resources between their laptops. To break this connection, users




(a) Overview of the devices and services (based on [70]) available in the user's environment of the Platform Composition system which uses the join-the-dots metaphor



(b) Using the jigsaw puzzle metaphor to define the following interaction: upon ringing the doorbell, a picture is taken by a webcam and sent to a tabletop display [41]

Fig. 1. Metaphors used for cross-device computing

can simply drag a line across the service (metaphorically "cutting" the connection). The use of the join-the-dots metaphor has been described in the literature as follows:

**Definition 2.2 – Join-the-Dots Metaphor**

By using the join-the-dots metaphor, the environment is represented using circles or clusters that can be connected with each other. Each cluster represents a device containing services that are represented by smaller circles or dots. Connections between devices and services are created by drawing a line between the corresponding circles [23].

The main advantage of the join-the-dots metaphor is its simplicity. Further, the graphical visualisation of all available devices and their services in a single view represents an overview of the entire system state. However, its principal disadvantage is related to scalability. In environments with many devices and services, the visualisation can become crowded by showing all devices and services, even those that are currently not used.

A more recent mashup approach called *A-Mash* was proposed by Lee et al. [48]. In contrast to previously discussed mashups, A-Mash is a mobile platform allowing users to dynamically extract UI elements from existing apps and embed them in a wrapper app. Users can thus mash up parts of multiple applications into a single screen according to their preferences. This is done using the MashupRecorder which allows the users to navigate to their apps, perform a multi-finger gesture to enter selection mode and select which UI elements they want to extract to compose their custom mashup interface.

Similar work was presented by Oh et al. [66] introducing *FLUID*, which enables users to decompose mobile applications by migrating or replicating individual UI elements across multiple devices using the same UI selection procedure. FLUID requires the app to be installed on all mobile devices and the source code to be available to allow its distribution. Cui et al. [21] presented *PRUID* enabling UI distribution without these limitations. PRUID offers an app-independent UI agent that captures and extracts information from a UI element on the host device at run time and allows the

distribution to a guest device by reconstructing the UI element on that guest device. These tools offer the possibility to restructure and distribute mobile UI elements by selecting the UI elements they want to distribute to (de)construct app UIs according to their preference. No particular metaphors are offered to end users to perform this distribution.

Another tool, mainly introduced for facilitating cross-device user studies, is *XDBrowser* by Nebeling and Dey [60]. The tool allows end users to cut, copy and move parts of existing web interfaces between devices using a rectangular selection tool. A newer version of the tool also provides a semi-automatic distribution of UI components based on study results about cross-device usage for customising web pages [59].

Meskens et al. [56] presented *Jelly*, a tool for more advanced users, providing a design environment for designing UIs for multiple platforms in parallel by *copy and pasting* parts of a user interface from one device to another. When elements are copied to another device, the designer can select from a list of available widgets how elements should look on the target device. Jelly also introduces *linked editing* which keeps the content of UIs consistent across different devices. We group and define the copy-and-paste and cut-and-paste metaphor as follows:

**Definition 2.3 – Copy-and-Paste / Cut-and-Paste Metaphor**

By using the copy-and-paste or cut-and-paste metaphor, one can duplicate or transfer digital content by selecting it, copying or cutting it and then pasting it into a different location.

Focusing on allowing end users to define their own cross-device interactions, Chen and Li introduced *Improv* [16], which enables end users to define custom cross-device inputs by demonstration. The defined gestural inputs are used to manipulate existing applications. Improv was implemented as a browser extension and can, for example, be used to improvise a presentation clicker. This is done by first recording the target behaviour on the computer and then demonstrating the desired input method for this behaviour on the smartphone or another input device.

A different group of systems offers users the possibility to design cross-device applications from scratch, such as in the authoring environment proposed by Ghiani et al. [36], that enables the development of context-dependent cross-device user interfaces using trigger-action *rules*. The trigger can be a UI or contextual event, and the action can have an effect on the visibility of a UI component on a particular device. An example could be "*if user is near the widescreen, then assign 'shopping list content' to the widescreen device and unassign it from the mobile device*". Further, the authoring environment also allows rules to be previewed. Unfortunately, the authors' user study revealed that the tool is too complex for end users. Therefore, they created a domain-dependent extension offering a simplified view of the trigger-action rules (limited in functionality) for people working in the smart retail domain. The rules metaphor has been defined and used in the literature as follows:

**Definition 2.4 – Rules Metaphor**

By using rules, the system's behaviour is represented using if-then statements. Each statement expresses the way the system should behave given specific situations [67].

The rules metaphor allows for a wide range of automation by combining multiple triggers and actions and is more scalable as it permits the seamless addition of new devices and services

to broaden automation opportunities. Further, studies have shown that inexperienced users can quickly learn to use the rules metaphor [85]. However, as observed in the work of Ghiani et al. [36], the system's usability may decline as the number of available options increases, as the process of identifying the appropriate choice becomes more challenging for users.

In order to assist end users in reconfiguring their ubiquitous domestic environment, Humble et al. [41, 73] developed a user-oriented editor, allowing users to interconnect home devices according to their particular needs. Each jigsaw puzzle piece represents different components available in the user's domestic environment. By dragging and dropping compatible pieces together in the editing canvas, a connection is made between the corresponding components in the home, such as connecting and configuring lightweight sensors, computer devices and services. For instance, it is possible to establish an interaction where pressing the doorbell's push button triggers the webcam to take a picture, which is subsequently transmitted to a tabletop display. As shown in Figure 1b, this can be achieved by linking the puzzle pieces of the push button, webcam, and tabletop display device, and configuring them appropriately. The jigsaw puzzle metaphor has been defined as follows:

**Definition 2.5 – Jigsaw Puzzle Metaphor**

By using the jigsaw puzzle metaphor, applications are represented as a combination of multiple puzzle pieces. Each piece of a puzzle usually corresponds to a service, while the shape of the puzzle provides cognitive clues to discern the possible actions [23].

This metaphor offers an easy-to-understand and natural visual interface. While the jigsaw puzzle metaphor is one of the most used metaphors [23], its main disadvantage is that the expressiveness is limited by the number of sides of a puzzle piece.

### 2.1 Authoring Interaction Across Internet of Things Devices

In this section, we explore existing authoring tools for creating IoT interactions, examining the different metaphors employed to facilitate this process for end users. Similar to our approach in the preceding section, we discuss the advantages and disadvantages associated with each metaphor. Note that an overview of metaphors used in the IoT domain has also been presented by Paterno et al. in [68]. Several commercial solutions have been introduced to assist users in configuring their smart environments, most of them using Event-Condition-Action (ECA) rules. Some examples are *IFTTT*[1], *Atooma*[2], *Tasker*[3] or *Zipato*[4], and an overview of some IoT solutions can be found in [14].

The rules metaphor, together with the concept of trigger-action programming, has often been used by researchers as well. Some example tools are *ImAtHome* [30] and the *SmartFit Rule Editor* [4]. The former allows users to create complex rules for managing their smart home by creating rules containing reusable "scenes" (i.e. set of actions). The latter targets the *eWellness* domain and allows coaches and trainers to create rules for monitoring and analysing fitness and wellness data. A more advanced example is the *Trigger-Action Rule Editor (TARE)* presented by Ghiani et al. [37], which gives users continuous feedback on the rule that is being edited. TARE enables end users to create context-dependent behaviour in their smart environment by selecting triggers and actions by navigating through a hierarchy of concepts to compose a rule. Each concept is associated with a

[1]https://ifttt.com
[2]https://atooma.com
[3]https://tasker.joaoapps.com
[4]https://www.zipato.com

contextual dimension, being either “user”, “environment”, “technology” or “social”. Unlike most IoT EUD tools, TARE supports the modification and distribution of user interfaces across devices by using rules. However, it is unclear how TARE handles the UI distribution and whether it supports UI design. TARE’s user interface is shown in Figure 2.

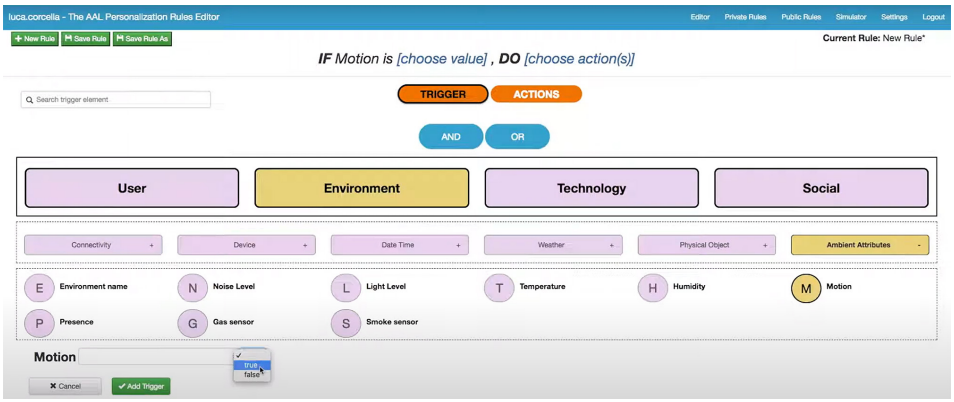



Fig. 2. The Trigger-Action Rule Editor (TARE) [37] user interface showing the different contextual dimensions to compose the triggers or actions of a rule. Notice the textual description of the rule at the top of the interface which is updated as the rule is being edited

More visual approaches such as *iCAP* [28] and *CMT* [84], offer a visual rule-based system for building context-aware applications as well. While iCAP provides a visual representation of the different rule elements, CMT focuses on providing a more visual representation of the rule composition using rule templates. While the rule authoring process for these tools is typically conducted using a computer, certain researchers offer dedicated tools that enable rule authoring through mobile phones, such as in *TouchCompozr* [47], *EPIDOSITE* [49] and *Keep Doing It* [25]. These three tools use programming by demonstration to define smart interactions. With TouchCompozr, users can demonstrate how they physically interact with tangible objects (e.g. a button), to create IF-THEN-ELSE rules. EPIDOSITE enables the demonstration of a particular behaviour through the manipulation of the smartphone’s IoT apps to create home automation scripts. Lastly, based on the users’ individual phone usage patterns Keep Doing It offers them recommended rules to automate smartphone tasks.

Another tool using programming by demonstration is *a CAPpella* introduced by Dey et al. [29]. In this approach, users repeatedly demonstrate a specific behaviour, including a situation and its corresponding action(s). Based on these demonstrations, a CAPpella learns to recognise this situation and performs the demonstrated actions when this situation is detected. The a CAPpella user interface employs a *timeline* metaphor, where users indicate the start and end time of the behaviour to show the system when the behaviour is demonstrated. The timeline metaphor is defined as follows:

**Definition 2.6 – Timeline Metaphor**

By using the timeline metaphor, temporal references can be provided, typically represented by a line on or above which different elements are (graphically) positioned. The elements correspond to events that are organised in chronological order [67].

The timeline metaphor is an intuitive visualisation, making it easy for users to grasp the sequence of interactions over time among devices given their familiarity with calendar and scheduling applications. Further, a timeline view can enable users to see emerging temporal behaviour patterns at a glance, facilitating retraceability of actions executed by the home (e.g. light turned off) [55]. In addition, the visualisation could also include information about the sensors that triggered them, revealing potential chains of causality, as seen in [17]. This allows users to deduce when certain rules will trigger and what state changes they will cause to connected smart home entities, improving the users' understanding of smart home behaviour. A potential limitation of the timeline metaphor is its inability to represent more complex interactions, as it might be challenging to accurately represent conditional events and concurrent actions.

The timeline metaphor has also been used in *AppsGate* [20], an end-user development environment that helps end users control and augment their homes. In AppsGate, users can program their smart home devices with a syntax editor by creating rules, actions and conditional statements using a pseudo-natural language. In contrast to other tools, AppsGate also provides debugging capabilities, by allowing users to test their rules using a virtual date and time. The *timeline* metaphor is used to monitor the smart home, and a *dependency graph* provides insights into the relationships between different entities of the smart home. The graph or pipeline metaphor can be defined as follows:

**Definition 2.7 – Pipeline Metaphor**

By using the pipeline or graph metaphor, applications are represented as directed graphs. The nodes of such a graph typically correspond to individual devices, services or activities, while links (i.e. pipelines) allow us to connect them. Binary logic operators and filter elements can be used to organise pipelines into more complex structures [23].

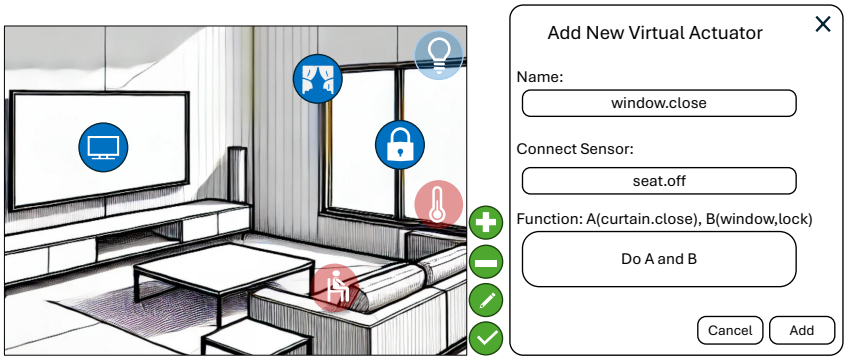
The pipeline metaphor allows us to model complex applications and their behaviour, offering a high degree of expressiveness. However, this metaphor often comes with a steep learning curve. Therefore, end-user tools should provide a simplified version or offer a high degree of automation [23].

García et al. [35] use the pipeline metaphor as well in the graphical editor of their IoT platform called *Midgar*. Using the web editor, users can interconnect heterogeneous objects to model the desired behaviour of the IoT application. The editor then generates the model of the application in a domain-specific language, which is then processed and transformed into a Java application that is then compiled and executed by Midgar.

Another modelling-oriented approach is presented by Koren [44] who introduced the *Direwolf Interaction Flow Modeler*, which builds upon the IFML language. The tool empowers end users to model web interfaces that incorporate Internet of Things sensors. However, it should be noted that end users require some prior training to effectively use this editor. The pipeline metaphor is also employed in the *IoT Mashup Application Platform (IoT-MAP)* [40] that dynamically discovers devices, downloads the necessary software modules and provides an end-user UI, called *Versatile*, for the mashup and composition of smart things based on Node-RED[5].

A different approach is proposed by Jeong et al. [43] who introduced virtual sensors and actuators that abstract physical *things*. The authors present the *AVIoT* web-based interactive framework for visualising and authoring IoT in indoor environments such as homes or small offices. Based

[5]https://nodered.org




(a) multiple selection (b) virtual node creation

Fig. 3. Representation of the AVIoT tool (based on [43]) demonstrating the following interaction: the user selects multiple nodes to define a new behaviour named "window.close"

on the user's input regarding their indoor architecture, the framework loads the closest match from their 3D model templates. The indoor 3D scene is configured to show attributes of available physical sensors and actuators. Given the unknown actual physical locations of IoT devices, they are initially placed arbitrarily within the scene, users can then relocate them to fit the real positions. The framework provides visual authoring methods allowing users to define certain behaviours. By, for example, selecting the seat sensor, window lock actuator and curtain actuator nodes on the 3D model, users could define that when someone is not seated on the sofa, the curtains and window should be closed as shown in Figure 3.

Moreover, AVIoT includes visual debugging by providing a hierarchical graph view that summarises the mappings between the sensors and actuators. Lastly, more complex behaviour definitions can be programmed using a scripting language. The main metaphor used in this approach is the floor plan metaphor, which we defined as follows:

**Definition 2.8 – Floor Plan Metaphor**

By using the floor plan metaphor, spatial references can be provided. Typically represented as a virtual model of a building's floor plan depicting a 2D or 3D layout, it provides a clear understanding of where elements are located within the physical space.

The main advantage of a floor plan is its spatial awareness, as it provides an intuitive understanding and overview of the position of the devices in the real-world. Users can quickly navigate through different rooms and areas of the smart environment, making it easier to manage devices across multiple locations, thereby allowing simplified location-aware behaviour authoring. A disadvantage of this approach is its maintainability since changes in the physical space should be reflected in the virtual one as well for maintaining consistency, which can be a challenging task. In AVIoT, it is up to the end users to update the 3D model when IoT devices are relocated.

Based on an elicitation study with computer science students, Desolda et al. [27] introduced and explored three rule creation interfaces called *E-Free*, *E-Wizard* and *E-Wired*. The *EFESTO mashup platform* [26] was used as a basis for the implementation of the prototypes. The E-Free prototype represents rules as events and actions with a split interface. The left side displays triggering events, and the right side shows the corresponding actions. Users can add new events and actions by clicking the + symbol, which opens a popup wizard for defining the event or action. The E-Wizard

prototype offers a more controlled environment. When users click the `new rule` button, they are guided through a wizard procedure to define a basic rule with one event and one action before they can add additional events or actions. Lastly, the E-Wired prototype utilises the pipeline metaphor for rule creation. Services are depicted as nodes and directed links represent cause-effect relationships. The UI includes a sidebar containing available services and a workspace area for linking services to create a rule. In a comparative study, the E-Free prototype emerged as the preferred choice, showing higher satisfaction levels and outperforming the other two prototypes as well as IFTTT.

Including some self-made tangible tokens to the IoT environment, Bellucci et al. [6] introduced *T4Tags 2.0*, an end-user toolkit for programming those tokens and other IoT devices using trigger-condition-action rules. These tokens enable the programming of intelligent behaviours on physical objects within the IoT environment. For instance, by placing a token on a door, one can detect when the door is being opened. The behaviour of these smart objects can be programmed using the T4Tags 2.0 user interface, a mobile-oriented web application. The UI menu resembles Atooma, offering IF and THEN options presented in a circular menu with icons. Users can create rules using this UI and share them publicly on a social platform. The created rules are displayed in a tile-based interface, similar to IFTTT rules, but with a more graphical representation that includes icons and pictures of the involved services.

Another mobile EUD approach, called *Puzzle*, is presented by Danado and Paternò [22]. Puzzle supports the development of IoT applications on smartphones, integrating web services, native phone functions and existing smart things. The applications can be created using the jigsaw puzzle metaphor. Each puzzle piece represents specific functionality that can be composed by connecting the pieces via a drag-and-drop interface. The shape and colour of the puzzle piece indicate the number of inputs and outputs, as well as the data that can be exchanged. It is worth noting that a commercial solution utilising the jigsaw puzzle metaphor is the Zipato Rule Creator, an online tool designed to create rules for the Zipato Home Automation System.

### 2.2 Discussion and Limitations

In the following, we discuss the key aspects and limitations associated with the selection of particular metaphors in end-user authoring tools for cross-device computing and the Internet of Things (IoT). It is important to note that our focus has primarily been on tools that provide different metaphors, and while we have presented a rich body of work, this is not meant to be an exhaustive literature review.

Throughout our exploration of end-user development tools in the domains of cross-device computing and IoT, we have observed a variety of approaches and metaphors. Some tools focus on the interaction between smart devices or *things* and their connection with each other, while others focus on elements that can be distributed among these smart devices, ranging from simple commands to parts or entire user interfaces transferred between devices.

The main goal of an end-user authoring tool is to hide the complexity of underlying technologies. Intuitive metaphors play a crucial role in achieving this goal by highlighting the main conceptual aspects and abstracting the technical details. For instance, when defining data transfers between devices, appropriate metaphors are used to hide the communication protocol, and show when and how data is transferred without requiring deep technical knowledge from the users [68]. Choosing the right abstractions and metaphors is not a straightforward task, as each comes with its own advantages and disadvantages. Various metaphors have been employed ranging from simple drag-and-drop to timelines and jigsaw puzzles. In Table 1, we provide an overview of the most dominant metaphors used by the related work discussed in this section. EUD solutions in the field of cross-device computing are denoted in black, while EUD solutions in the IoT field are highlighted in blue. We also included our eSPACE authoring tool, which is shown in red.

| Metaphor | Timeline | Rules | Pipeline | Jigsaw Puzzle | Join-the-dots | Floor Plan | Drag-and-drop | Copy/Cut-and-Paste | None |
|---|---|---|---|---|---|---|---|---|---|
| a CAPpella [28] | ✓ | | | | | | | | |
| A-Mash [48] | | | | | | | | | ✓ |
| ACCORD [41, 73] | | | | ✓ | | | | | |
| Ardito et al. [1] | | | | | | | | | ✓ |
| Atooma | | ✓ | | | | | | | |
| AppsGate [20] | ✓ | ✓ | ✓ | | | | | | |
| AVIoT [43] | | | ✓ | | | ✓ | | | |
| Blichmann et al. [10] | | | | | | | | | ✓ |
| CMT [84] | | ✓ | | | | | | | |
| DireWolf [45] | | | | | | | ✓ | | |
| Direwolf Interaction Flow Modeler [44] | | | ✓ | | | | | | |
| E-Free [27] | | ✓ | | | | | | | |
| E-Wired [27] | | ✓ | ✓ | | | | | | |
| E-Wizard [27] | | ✓ | | | | | | | |
| EPIDOSITE [49] | | | | | ✓ | | | | ✓ |
| eSPACE | | ✓ | ✓ | | | | | | |
| FLUID [66] | | | | | | | | | ✓ |
| Ghiani et al. [36] | | ✓ | | | | | | | |
| iCAP [29] | | ✓ | | | | | | | |
| IFTTT | | | | | | | | | |
| ImAtHome [30] | | ✓ | | | | | | | |
| Improv [16] | | | | | | | | | ✓ |
| Jelly [56] | | | | | | | | ✓ | |
| Keep Doing It [25] | | ✓ | | | | | | | |
| MashupEditor [38] | | | | | | | | ✓ | |
| Midgar [35] | | | ✓ | | | | | | |
| Platform Composition [70] | | | | | ✓ | | | | |
| PRUID [21] | | | | | | | | | ✓ |
| Puzzle [22] | | | | ✓ | | | | | |
| SmartComposition [46] | | | | | | | ✓ | | |
| SmartFit [4] | | ✓ | | | | | | | |
| T4Tags 2.0 [6] | | ✓ | | | | | | | |
| TARE [37] | | ✓ | | | | | | | |
| Tasker | | ✓ | | | | | | | |
| TouchCompozr [47] | | ✓ | | | | | | | |
| Versatile [40] | | | ✓ | | | | | | |
| XDBrowser 2.0 [59] | | | | | | | | ✓ | |
| Zipato | | | | ✓ | | | | | |

Table 1. Summary of used metaphors by existing solutions in alphabetical order. Cross-device solutions are shown in black, IoT solutions are shown in blue and our solution is depicted in red (eSPACE).

In the field of IoT, the rules metaphor is widely used by several tools [4, 6, 20, 25, 27, 29, 30, 37, 47, 84]. A few exceptions are *a CAPpella* [28] employing the timeline metaphor, *Puzzle* [67] using the

jigsaw puzzle metaphor and *DireWolf IFM* [44], *Midgar* [35] and *Versatile* [40] that are applying the pipeline metaphor. Finally, *EPIDOSITE* [49] does not leverage any specific metaphor. Moreover, note that some tools like *AppsGate* [20] are applying multiple metaphors, including the rule, timeline, and pipeline metaphors. Multiple metaphors are used in *AVIoT* [43] as well, containing the pipeline and the floor plan metaphor.

On the other hand, in the domain of cross-device computing, metaphors are less prevalent and more scattered among tools. These include the drag-and-drop [45, 46], the copy/cut-and-paste [38, 56, 59], rules [36], join-the-dots [71] and jigsaw puzzle [41, 73]. Finally, many tools do not provide any specific metaphors, such as *A-Mash* [48], Ardito et al.'s tool [1], Blichmann et al.'s tool [10], *FLUID* [66], *Improv* [16] and *PRUID* [21].

As mentioned before, each metaphor comes with its limitations. Many metaphors suffer from scalability issues. For example, the pipeline metaphor in diagrams becomes harder to interpret with an increasing number of elements and connections. Similarly, the jigsaw puzzle metaphor's scalability might be limited due to the cluttering of the screen with numerous puzzle pieces. Further note that the jigsaw puzzle metaphor is also limited by the number of sides of a puzzle piece, restricting the number of possible combinations (i.e. restricting its expressiveness) [23]. Certain metaphors, like rules, allow easy composition of events and actions by using Boolean operators (e.g. AND, OR, and NOT), but they may be prone to breaking and detecting conflicting actions can be difficult [29]. A potential solution is to restrict the choice of trigger combinations, as done in *T4Tags 2.0* [6], which also further guides users during the rule creation process to avoid inconsistent or invalid states. Others [19] focus on providing ways to debug trigger-action rules. Depending on the information needed, some metaphors might be more appropriate, as the timeline metaphor is practical for conveying time-related information, while the floor plan metaphor will be more efficient for conveying spatial references.

Considering the diverse advantages and limitations of metaphors, researchers strive to identify the most fitting metaphor(s) for their tools. This selection process typically involves reviewing related work, analysing commercial solutions and drawing on personal experience. Yet, a crucial aspect often overlooked is the end users' perspective on cross-device and IoT interactions. Their mental models, which shape how they understand and interact with technology, can significantly differ from those of developers, particularly in computational thinking contexts [68]. Therefore, adopting representations and concepts that resonate with end users' mental models is essential for enabling successful end-user authoring experiences. This ensures the development of tools that are not only functional but also intuitive and accessible to the intended audience. In other words, there is a need to investigate the end users' mental models regarding cross-device and IoT interactions to select the most intuitive metaphor(s). While Desolda et al. [27] conducted an elicitation study to identify suitable metaphors, their participants were computer science students, not the general end-user population. To the best of our knowledge, Dey et al. [29] are among the few researchers who directly engaged with end users to better understand their mental models, specifically in the context of context-aware applications. Based on insights from their study, they designed *iCAP*.

Lucci and Paternò [53] also performed a study to improve current designs and inform future designs of mobile context-dependent applications, which result influenced the requirements for *TARE* [37]. However, this study had a narrower focus than that of Dey et al. [29], centring on smartphone EUD applications. During the evaluation of the *Puzzle* prototype [22], a pre-test questionnaire was used to assess different visual metaphors. Results showed a preference for the jigsaw puzzle metaphor, followed by the pipeline metaphor.

A more exploratory study has been conducted by Brich et al. [11] comparing the rule-based and process-oriented paradigms[6]. During the study end users were told to follow either the rule-based or process-oriented paradigm and were given a pen and paper notation kit. They were asked to model whatever behaviour they would want in their ideal smart home. Brich et al. identified benefits and drawbacks of the two paradigms, suggesting that providing end users with a mix of the two approaches might be beneficial to address a broad variety of automation desires, as rule-based notations are sufficient for simple automation tasks but not flexible enough for more complex use cases.

While these studies have sought user preferences among predefined metaphors, there is a lack of research exploring users' spontaneous inclination towards metaphors when faced with a scenario A more open-ended approach has been presented in Sanctorum et al.'s [76] elicitation study, which allowed end users to freely depict smart behaviours of a given scenario using pen and paper. This method revealed a tendency among participants to use arrows for representing interactions between smart devices and *things*, aligning closely with the pipeline metaphor. From these observations, Sanctorum et al. derived design guidelines for developing or refining cross-device and IoT end-user authoring tools. The guidelines are shown in Table 2. One of them advocates for the use of the pipeline metaphor, while another also recommends employing rules. Based on these guidelines, we designed the eSPACE end-user authoring environments, which are introduced in the next section.

## 3 ESPACE END-USER AUTHORING TOOL

In this section, we introduce eSPACE, which stands for end-user Smart PlACE. eSPACE is an end-user authoring tool designed to empower non-technical users to create their own cross-device and IoT applications using various authoring environments. Users can, for example, create an application to control their smart TV by designing a user interface with buttons linked to specific TV functionality, such as turning off the TV with a button click. Alternatively, users can also create applications without user interfaces by defining rules that automate appliance behaviour, such as specifying that at 11 p.m the TV should turn off.

Our tool is the result of following the Design Science Research Methodology (DSRM) for information systems research, as defined by Peffers et al. [69], to address our main research question. DSRM is widely used by design science researchers to guide the production, presentation, and evaluation of research outcomes. eSPACE is built on top of theoretical artefacts also resulting from our DSRM process, which includes a reference framework, conceptual model and design guidelines. The DSRM process model is based on prior design science research principles and includes six activities. The first activity, problem identification and motivation (1), is presented in Section 1. The second activity involves defining the objectives of a solution (2). The third one is design and development (3), followed by the demonstration phase (4), which is described in this section. Next is the evaluation step (5), where we assess our artefacts through a user study outlined in Section 4. Finally, the communication phase (6) is accomplished through the publication of our work.

The specific objectives of this research are:

- Investigating the metaphors used by end-user development (EUD) tools in the contexts of cross-device (XD) and IoT environments (see Section 2).
- Introducing the necessary theoretical concepts, frameworks, and guidelines to abstract technical concepts for end-user authoring (presented in [76, 79], briefly described in Section 3.1).

[6]Note that the process-oriented notation of this work uses the pipeline metaphor.

- Developing a flexible and extensible end-user authoring tool that supports a wide variety of devices and enables cross-device and IoT interactions, informed by the defined theoretical foundations established in the previous objectives (see Section 3.2 and Section 3.3).
- Validating the applicability of the theoretical foundations and effectiveness of the proposed tool (see Section 4).

In the remainder of this section, we first discuss eSPACE's theoretical foundations and then elaborate on its design by presenting the different authoring views. Lastly, we show how end users can use our tool to create their own custom applications for managing and controlling their smart environment.

### 3.1 Theoretical Foundations

The eSPACE end-user authoring tool has been built based on several key theoretical principles, including requirements, a reference framework, a conceptual model, and design guidelines. First, in order to guide the development process, a set of functional and non-functional requirements has been selected. These requirements were derived from related work as well as from a use case scenario [75] and are outlined in Appendix A. Next, we opted for a model-based approach to facilitate the UI development and offer more flexibility and extensibility. Therefore, the approach described in [79] has been adopted. This approach utilises a reference framework which decomposes the UI development process into different layers of abstraction, as shown in Appendix B. By using such a compositional approach, elements contained in each layer can be linked to create a final user interface that is tailored to a certain device, user and context. The resulting UI model must then be parsed and transformed into an executable program in order to be used by users. The available elements of the reference framework are described through a conceptual model showing the relationship among them, which is illustrated in Appendix C. Further, some example use cases are presented in [79].

In contrast to most related work, which typically relies on existing research prototypes, commercial solutions and personal experiences to select metaphors for abstracting technical complexities, we based our approach on the findings of an elicitation study conducted by Sanctorum et al. [76]. This study investigated which metaphors people naturally employ to conceptualise cross-device and IoT interactions. The insights gained from the study and the review of related work led to the formulation of design guidelines for building cross-device and IoT end-user authoring tools. The set of design guidelines is shown in Table 2 and has informed the design of the authoring environments of our eSPACE end-user authoring tool.

| Guidelines | |
|---|---|
| G1 | Use pipeline or graph metaphor to represent interactions |
| G2 | Assign different arrow types to different types of interaction |
| G3 | Provide a realistic graphical device representation |
| G4 | Provide a graphical representation of users |
| G5 | Represent sequential interactions from left to right and group concurrent interactions |
| G6 | Provide textual as well as graphical representations for conditional statements |
| G7 | Support UI design |
| G8 | Include (custom) symbols and support for annotations |

Table 2. Design Guidelines based on prior work [76]

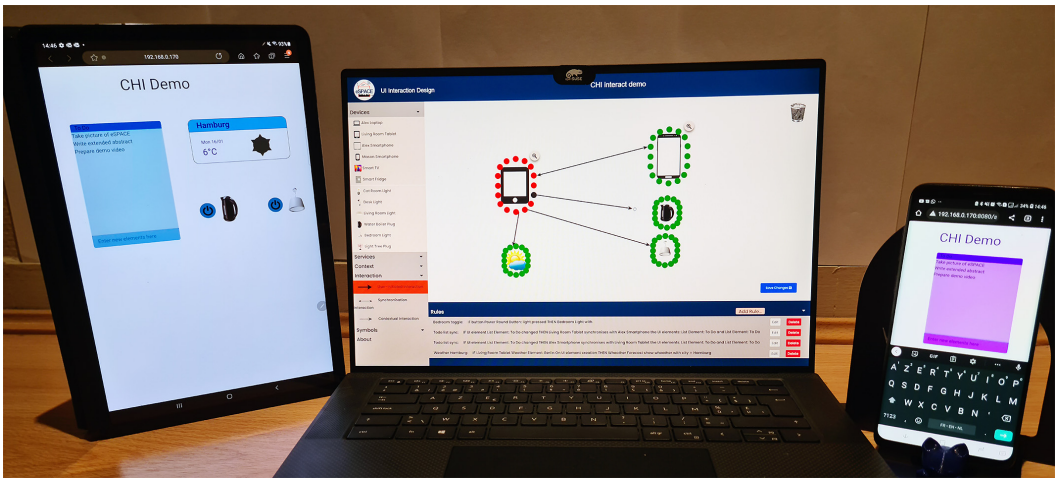



Fig. 4. The eSPACE end-user authoring tool in action, with the tool open on a laptop showing the *interaction* view, and example eSPACE applications running on a tablet and smartphone. The *interaction* view shows five elements on the authoring canvas (i.e. tablet, smartphone, water boiler, lamp and the weather forecast service). The user is currently creating an arrow between the tablet element on the left and the water boiler on the right by dragging the arrowhead towards a green dot surrounding the water boiler.

By building upon these theoretical foundations, eSPACE has been developed to meet the identified requirements and adhere to the model-based UI development process delineated in the reference framework [79], all while aligning with the design guidelines suggested in [76]. To the best of our knowledge, based on related work analysis in [76], eSPACE is the only XD and IoT EUD tool that supports all listed design guidelines. For instance, our second guideline (G2) is not supported by any existing solutions, and the seventh guideline (G7) is rarely supported in IoT tools, highlighting eSPACE's distinctiveness. We present the different (authoring) views of eSPACE in the next section. Please note that it is beyond the scope of this paper to go into more details about the theoretical foundations of eSPACE, but more information can be found in [75].

### 3.2 eSPACE Views

The eSPACE end-user authoring tool is a web-based tool, making it accessible from a wide range of devices. It has been implemented using Spring Boot, and consists of five views allowing users to author and use their own custom applications. An application can consist of multiple user interfaces. Each UI is tailored for a specific device screen and is represented as a web page. An example of the eSPACE tool in action is illustrated in Figure 4. On the laptop screen, one can see one of eSPACE's authoring views, while on the tablet and smartphone, a user-defined application is displayed. In the following sections, we provide a detailed description of each of eSPACE's views, including the *home* view, *interaction* view, *rules* view, *UI design* view, and *app* view. The navigation options between these different views are depicted in Figure 5.

*3.2.1 Home View.* The *home* view provides a dashboard-like overview, allowing users to view, edit and delete their user-defined applications, rules and devices. It also enables users to see and share their applications with other users. The *home* view can be accessed from any authoring view by pressing the eSPACE logo at the upper left corner of a page. When users create a new application or want to edit existing applications, they are redirected to the *interaction* view, which allows them

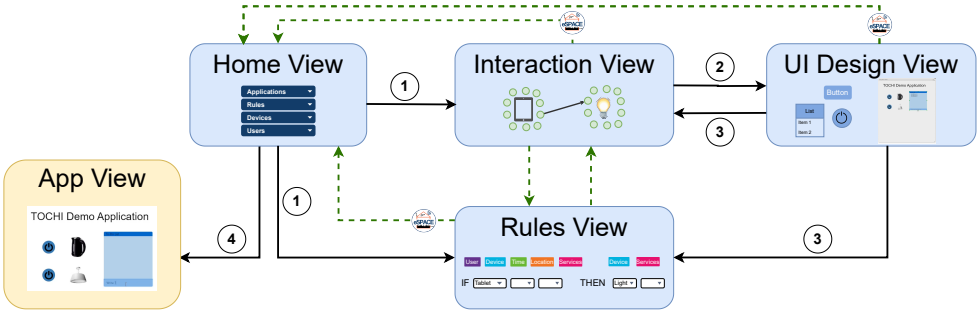



Fig. 5. Flowchart representing the navigation paths between the different (authoring) views of eSPACE for creating XD and IoT applications. Users start in the *home* view. To create an application, they can choose the *interaction* view for a visual approach or the *rules* view for a more textual approach (1). Both of these views can be used to define interaction rules. To design a UI for an application, users have to navigate to the *UI design* view (2). Afterwards, they can link functionality to UI elements by returning to the *interaction* or *rules* view (3). The created application can be accessed via the *app* view (4). Note that users can return to the *home* view from any authoring view, as shown by the green dashed arrows.

to define interaction rules for the selected application through a visual interface. Alternatively, a more textual interface to create and modify interaction rules is also provided. Users are redirected to this interface, called the *rules* view, when creating new rules or by clicking the edit button of an existing rule. The *home* view also allows users to add and edit devices. However, it currently does not include functionality for adding new users, although developers can use the RESTful API to add users to the system. In the future, it is expected that the *home* view will also enable users to add new friends using the web interface.

*3.2.2 Interaction View.* The *interaction* view is illustrated on the laptop's screen in Figure 4. It allows end users to graphically define interactions between devices by linking them together on the authoring canvas. In the left panel of the view, we grouped elements that can be added to the authoring canvas into five categories: devices (smart devices and *things*), services (like a weather forecast), contextual elements (user, time and location), interaction arrows, and symbols and annotations. The last category has been added to adhere to guideline G8 and enables end users to add custom elements to the authoring canvas. There are three types of interaction arrows, each representing different types of interactions as suggested by guideline G2. The regular arrows are used to represent all actions between devices except for synchronisation, which is represented by double-sided arrows. Dashed arrows are used for contextual interactions, which have a contextual element as a trigger. Contextual interactions therefore include either a time-, location or user-related component. Note that each component is graphically represented by an icon that—in line with guideline G4—symbolises the specific contextual element.

When devices are dragged from the side panel and dropped to the authoring canvas, they are represented with larger graphical representations (aligning with guideline G3). If the device is a screen device[7], a magnifying glass icon (🔍) appears above it on the right. This magnifying glass icon can be clicked to go to the *UI design* view, where users can design a UI tailored for the selected device screen, such as a smartphone or tablet screen.

[7]a device for which the end user can create a user interface

In order to create an interaction rule, devices can be linked with other elements on the authoring canvas using interaction arrows. To do so, we opted for a combination of the pipeline metaphor and the join-the-dots metaphor, hereby following guideline G1.

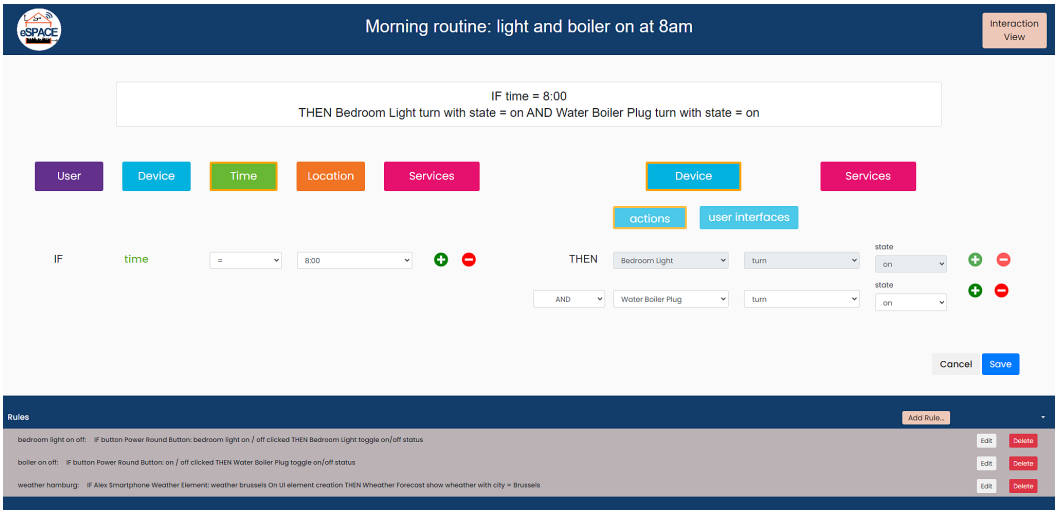



Fig. 6. *Rules* view of the eSPACE end-user authoring tool showing the rule: "IF time = 8:00 THEN Bedroom Light Turn On AND Water Boiler Plug Turn On"

When selecting an arrow in the side panel, elements that can be connected on the authoring canvas will be surrounded by green dots and the ones that cannot be connected by this type of arrow will be surrounded by red dots. After selecting a source element by clicking on a green dot, users can join the dots by selecting another green dot surrounding the target device. This process is illustrated in Figure 4, where the tablet element is being linked to the water boiler element using a regular arrow. Once linked, the arrow will appear between both elements and a popup window will ask the user to fill in additional parameters depending on the type of interaction. Example interactions consist of turning off a light at a certain time, or synchronising a specific folder between two laptops.

Further, note that all interactions are also shown in the form of textual trigger-action rules in the container at the bottom of this view. Via this container, users can access the *rules* view, where they can modify or add new interaction rules as well. All rules defined in the *interaction* view can be visualised in the *rules* view and vice versa, with both views providing a consistent graphical and textual representation of the interaction rules. This is in line with guideline G6, which suggests providing both graphical as well as textual representations of conditional statements that, in our case, are the interaction rules.

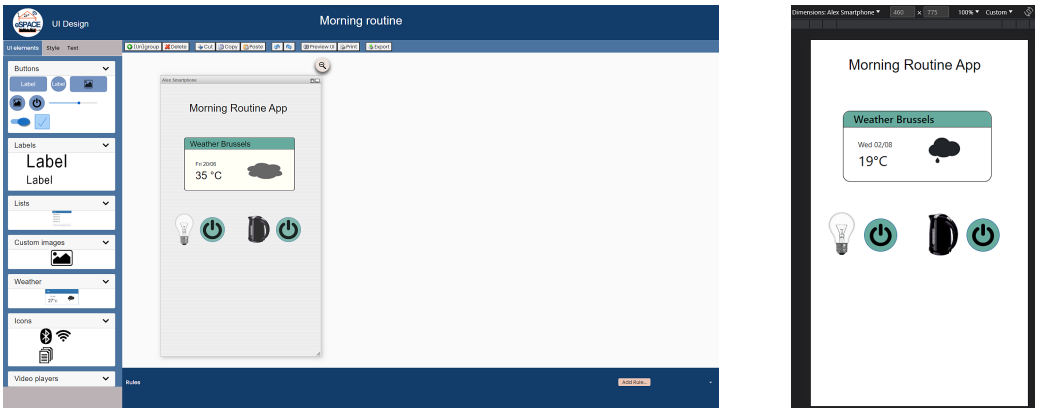
*3.2.3 Rules View.* In the *rules* view, users can define interaction rules in the form of trigger-action rules using the tile-based interface shown in Figure 6, which was inspired by TARE [37]. The tiles represent different categories including `devices`, `time`, `location`, `user`, and `services`. We named the tiles according to the Rule_5W model defined by Desolda et al. [27]. When selecting a tile, the options related to this tile will appear in the row of dropdowns located beneath the tiles. By selecting the different options in the dropdowns, users can formulate an if-then statement. The current rule defined via the dropdowns is described in a sentence at the top of the page. Note that the *rules* view allows users to define more complex expressions using the boolean `AND` and

OR operators. Lastly, by using the + and − symbols, users can add or remove triggers and actions. In order to keep it simple, the current version of eSPACE supports up to two triggers and two actions. An example interaction rule is shown in Figure 6 where the user defined that at 8 a.m., their bedroom light as well as their water boiler should turn on.

The *rules* view represents sequential interactions from left to right and concurrent actions are grouped vertically as suggested in guideline G5. Although the *interaction* view does not enforce this arrangement, the visual counterpart of every rule defined via the *rules* view will conform to this guideline. In other words, when viewing a rule that has been created in the *rules* view in the *interaction* view, all triggers will be vertically aligned on the left, while the actions will be grouped on the right.

All rules defined for the current application are displayed in a dedicated container at the bottom of the page, mirroring the design of the rules container found in the *interaction* view. Users can navigate to the *interaction* view from the *rules* view by clicking the `Interaction View` button, which is positioned in the top right corner of the page. When pressing the `Save` button to save the rule or the `Cancel` button, users will be taken back to the *home* view.

*3.2.4 UI Design View.* In the *UI design* view, users can design simple graphical user interfaces for their applications tailored for specific devices, hereby fulfilling guideline G7. The available UI elements are shown in a side panel on the left and can be dragged and dropped to a canvas on the right, which represents the device screen. In addition, this panel includes multiple tabs which allow users to further customise a UI element by, for example, changing its colour and font. The toolbar at the top provides copy/paste, undo/redo, delete, and grouping functionality. Although the size of the design canvas reflects the dimensions of the device screen for which the UI is being designed, the resulting UI can still be viewed on other compatible devices. Note that while this view enables the creation of a personalised user interface, it does not include any linked functionality for the UI elements. To add such functionality, users should use the *interaction* or *rules* view. An example UI design is shown in Figure 7a, where the user created an interface to see the weather and to control the light and the water boiler. The resulting application is illustrated in the *app* view in Figure 7b, which is explained in detail in the following section.




(a) *UI Design* view allowing users to create customised interfaces for their applications

(b) *App* view with user-created application

Fig. 7. *UI Design* view of the eSPACE authoring tool on the left and the resulting app on the right

*3.2.5 App View.* Via the *app* view, users can see and use their user-defined applications. When a user clicks the `Open` button of an application in the *home* view, eSPACE will open a number of browser tabs. Each tab shows one user interface of the application. As mentioned before, applications might be composed of multiple UIs, each tailored for a particular device. Therefore, these tabs are meant to be opened on the corresponding devices as shown in Figure 4, where the tablet-tailored UI is opened on a tablet, while the smartphone-tailored UI is used on the smartphone. Given that we follow a model-based approach, the functionality of the *app* view is created by parsing the UI model stored on the eSPACE server. This UI model has been constructed based on the UI created in the *UI design* view and the interaction rules defined in the *interaction* or *rules* view. Additionally, we support the distribution of UI elements across multiple devices using the *longpress* gesture on the desired element in the *app* view. This will open a popup window asking the user to which device they want to distribute the UI element. In order to see the distributed UI element on the target device, the user must refresh the *app* view. In a future version of eSPACE, we plan to display a popup window on the target device asking the user whether they want to accept the distributed element and show it without requiring a manual page refresh.

### 3.3 Creating User-defined Applications

This section demonstrates how to create a simple application including cross-device and IoT interactions by using eSPACE. We start by building an app to control two IoT devices, which is then extended with additional functionality, including list synchronisation across devices, a contextual interaction and the integration of a weather forecast service, as further illustrated in a video tutorial[8]. The application we are creating is designed for use on a tablet and will control the water boiler plug and smart light bulb in our bedroom. We want the app's interface to feature two buttons that act as switches to turn the plug and light on and off.

To create this application, we start in the *home* view and press the `Add App...` button. This opens a popup window asking for the application's name. After providing a name, we confirm our choice and are redirected to the *interaction* view. Here, we begin by dragging and dropping the `tablet` element on the authoring canvas. Next, we go to the *UI design* view by clicking the magnifying glass icon (🔍) at the top of the `tablet` element. In this view, we drag and drop two `power buttons` to our design canvas. We then return to the *interaction* view by pressing the magnifying glass icon again. In the *interaction* view, we can now link the `tablet` element to the water boiler and smart light to add functionality to the buttons. To do this, we drag and drop both the `water boiler` and `light` elements to the authoring canvas. Then, we select the regular interaction arrow, click on one of the green dots surrounding the `tablet` element, and drag the arrow towards the `water boiler` element as illustrated in Figure 4. After releasing the arrowhead on a green dot surrounding the `water boiler`, the source and target of the arrow are defined and a popup window appears to define the interaction details. We specify that when the water boiler power button is pressed, we want the water boiler to toggle on/off. We repeat this process to add functionality to the light button. Note that we could have used the *rules* view to define these interactions. By clicking the `Add Rule...` button in the rules container from the *interaction* or *home* view, we would be redirected to the *rules* view, where we could create interaction rules using if-then statements. For instance, we could define the water boiler interaction rule by selecting the appropriate tiles and indicating in the dropdowns that "*IF Tablet, Power Round Button: water boiler, clicked, THEN Water Boiler Plug, toggle on/off status*".

To facilitate cross-device interactions, eSPACE allows users to synchronise UI elements across devices. We demonstrate this by extending our application with a to-do list that is synchronised

[8]https://youtu.be/KuAwowYwsOQ

between our tablet and smartphone. This is done by adding a `list` UI element to the existing tablet user interface and creating a new user interface for our smartphone that only contains a `list` UI element. Therefore, in the *interaction* view, we add the `smartphone` element to the authoring canvas and navigate to the *UI design* view to create the smartphone's interface, which consists only of the to-do list. We then modify the tablet UI by adding a `list` UI element for the to-do list. We can now define the interaction between the two `list` UI elements using either the *interaction* view or the *rules* view. If we use the *interaction* view, we connect the `tablet` and `smartphone` elements with the double-sided interaction arrow, and then fill in the necessary information in the popup window.

eSPACE further enables contextual interactions using the dashed interaction arrow, which involves either a user-, time- or location-related element. For example, to turn on the bedroom light at 8 a.m., we can drag and drop the `time` contextual element in the *interaction* view and link it to the `light` element with a dashed arrow. Then, in the popup window, we specify that at 8 a.m., the light should turn on. Alternatively, we could create this rule in the *rules* view by selecting the `time` tile and filling in the dropdowns specifying "*IF time=8:00 THEN Bedroom Light Turn On*", similar to the rule shown in Figure 6.

Finally, eSPACE provides services that can be linked to UI elements, such as the weather forecast service. For example, we can integrate this service into our application by adding a weather component to display the weather in Brussels on the tablet UI. This is done by including the `weather` UI element to our tablet UI in the *UI design* view and linking it to the functionality "show weather in Brussels". This can be achieved either via the *interaction* view, using a regular arrow to connect the `tablet` element to the `weather` service, or through the *rules* view by creating the rule "*IF Tablet, Weather Element: Brussels, On UI element creation THEN Weather Forecast, show weather, city: Brussels*". In this case, the weather will be shown to the user upon opening the application, as we selected "On UI element creation" as the trigger.

In the next section, we delve into the evaluation of the eSPACE end-user authoring tool and provide a comprehensive analysis of its outcomes.

## 4 USER STUDY

In this section, we present the user study conducted to evaluate the usability and user experience of the eSPACE end-user authoring tool, thereby achieving our last objective and addressing our main research question. This study also aims to assess the practical application of our design guidelines, outlined in Table 2. While the guidelines are derived from findings in related work and our elicitation study [76], no existing solutions, to our knowledge, fully implement all of these guidelines as noted in [76]. Therefore, we seek to further validate the guidelines through this user evaluation of eSPACE. Our guidelines advocate for the use of both the pipeline metaphor (G1) and rules metaphor (G6), primarily based on the results of our elicitation study [76] and supported by Brich et al's exploratory study [11], which compared both metaphors. Similar to their work, we found it relevant to investigate user preferences between these metaphors in a practical setting, rather than in a pen-and-paper format as used in Brich et al's study. Additionally, we aim to evaluate how easily and successfully participants can create interaction rules for controlling a smart home using eSPACE, while also identifying areas for improvement. In summary, this study aims to achieve the following:

- Validate the practical application of our design guidelines outlined in Table 2.
- Identify user preferences between the pipeline and rules metaphors in the context of XD and IoT application creation.
- Assess the usability and user experience of eSPACE.

- Evaluate the ease of the application creation process.
- Identify potential areas for improvement.

Note that a pilot evaluation of eSPACE was conducted with eight participants in an online setting[9]. The goal was to gather preliminary feedback and capture initial impressions from participants. Based on the outcomes of this study, refinements, bug fixes and enhancements were applied to both the functionality and user interface of the tool. A complete description of the study can be found in Chapter 7 of [75]. While this initial study indicated promising results, we aimed to further assess eSPACE with a more comprehensive evaluation which is detailed below.

Fig. 8. Setup for the user study: a laptop and an external display showing the eSPACE home page; a second external display showing the house plan of the smart home. A hypothetical participant is seated on the right, while the study supervisor is positioned to the left.

The remainder of this section is organised as follows. We first present the user study setup, followed by a brief description of our participants. Subsequently, we elaborate on the study protocol, which replicates the protocol used in the pilot study, involving participants performing a series of tasks to create cross-device and IoT applications using eSPACE. Afterwards, we discuss the study results and conclude this section with a summary and discussion of potential avenues for improvements.

### 4.1 Setup

The study took place in our research lab, where we set up a large table with a laptop, two external displays, a keyboard and a mouse. The laptop's screen was duplicated on an external display and the other display was used to show a map of the smart home, showing the status of the different smart devices. This map was created explicitly for the study to simulate a smart home environment, allowing participants to interact with devices without physically setting up multiple IoT rooms. Participants were seated in front of the two external displays and used a keyboard and mouse to navigate the eSPACE tool as illustrated in Figure 8.

[9]prompted by the Covid-19 social measures during that period

### 4.2 Participants

A total of 22 participants (7 males and 15 females) with ages ranging from 20 to 47 years (average 26) participated in the study. Participants were recruited via flyers distributed across the university campus and nearby buildings. None of the participants had a computer science background or programming skills. Each of them was awarded a €20 voucher to compensate for their participation.

The demographic characteristics of the participants were gathered via a questionnaire that collected information on their gender, age, country of origin, highest level of education, technological proficiency, awareness of the term "Internet of Things" (IoT), and possession of smart devices. Analysis of the responses showed that the majority, comprising 14 participants, were either pursuing or had obtained a Master's degree. In contrast, 3 participants were pursuing Bachelor's degrees, and 5 were enrolled in or had completed a PhD programme. Most of the participants were familiar with the term IoT; only 2 participants reported they had "never heard of it", and 2 had simply "heard of it". Responses from the remaining participants ranged from "know it well" to "know a fair amount". Regarding the ownership of devices, 7 participants reported having no IoT devices at home, whereas the others had at least one IoT device. Further, all participants owned between 2 and 7 smart devices. Lastly, 13 participants self-assessed their technological proficiency as "good", while 6 positioned themselves in the "neutral" category. The remaining 2 participants considered themselves as having "bad" proficiency, and 1 rated their proficiency as "very good".

This demographic information provides a basis for understanding the participants' background and technological familiarity, which is essential for analysing the study's findings.

### 4.3 Protocol

Participants first received a consent form and a short oral explanation of the study procedure. Each participant was given a few pages of explanation about the different devices available in our smart home scenario and a brief explanation of the different views offered by the tool and was invited to watch a five-minute demo video of the tool. After that, they were asked to complete three tasks, during which they created different applications using the eSPACE authoring tool. Finally, we concluded the user study with an interview and a post-study questionnaire. The different steps are detailed below and all user study documents are available online[10].

*4.3.1 eSPACE Presentation (10 mins).* Each participant received a four-page document describing a scenario and visually detailing our smart home setup, the available devices, and different views of the eSPACE authoring tool. Next, a short demonstration video[11] that explained how to use eSPACE through the creation of a small example application was shown on their screen to watch before starting the study. The documents and video were available at all times during the study. Participants were also told to think aloud during the study and ask questions if they had any or were stuck during the study.

*4.3.2 Task Execution (20–50 mins).* During the evaluation, participants took on the role of user *Alex* who lives in the smart home environment described in the provided document. Alex wishes to create a couple of applications to facilitate her daily routine and enhance the interactions across her various smart devices and *things*. To provide participants with a clear understanding of Alex's smart home, a visual representation was displayed on a separate external display, as illustrated in Figure 8. A detailed overview of the smart home is shown in Figure 9, displaying each smart device's status and allowing participants to monitor changes when controlling devices via eSPACE. Note that this smart home visualisation is not part of eSPACE but was created specifically for this

[10]https://doi.org/10.5281/zenodo.10777381
[11]https://youtu.be/NnQ9auXKj68

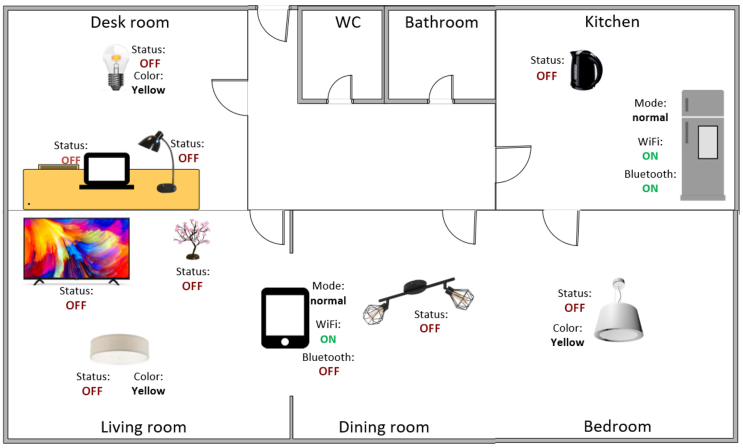



Fig. 9. House plan of Alex's smart home, specifically created for performing our user study. It displays various smart devices and *things*, along with their statuses, which participants can control using eSPACE.

user study to fit the study's scenario. During the evaluation, participants were asked to create three applications using eSPACE, referred to as follows:

**App1**. The Simple Controller Application
**App2**. The Grocery List Application
**App3**. The Morning Routine Application

The first one is a simple application that is meant to be used on a tablet and consists of three buttons to control different IoT devices located in the living room. For the second application, participants have to create two user interfaces, one for Alex's smartphone and one for the smart fridge. Each user interface contains a grocery list that should be synchronised. Further, the UI of the smart fridge should also display some family pictures. The last application is a smartphone application that facilitates Alex's morning routine by showing her the weather of Brussels once the application is opened and triggers the light of her bedroom as well as her water boiler plug to go on at 8 a.m. Next, the app also allows Alex to control the water boiler plug and bedroom light manually via two buttons.

These three applications were defined to cover most of the functionality offered by our authoring tool, ranging from simple buttons to control smart appliances to the synchronisation of UI elements and the use of a contextual element (time) as well as an external service (weather forecast). In order to create the three applications, participants had to use the three different arrow types offered in the *interaction* view or use the corresponding functionality in the *rules* view, which allowed us to evaluate how easy it was to use our metaphors. We summarise the different applications and their related rules in Table 3. We categorise the rules into four distinct rule schemes. The first one entails an interaction triggered by a button click (RS1). The second one employs a synchronisation interaction (RS2). The third one involves the use of a service (RS3) and the last one includes a contextual interaction (RS4). Note that there are two possible approaches to create the time-related rule, one using the `AND` operator and the other by creating two separate rules, as shown in the table.

Once participants read the documentation about eSPACE, watched the demo video and read the description of the three applications to be created, they were given access to the authoring tool via the web browser. Given that we also aimed to find out whether users preferred the pipeline metaphor or the rules metaphor for creating rules, we divided our participants into two groups.

| App ID | Rule Schema | IF-THEN Rule |
|---|---|---|
| App1 | RS1 | **IF** Living Room Tablet *label button: TV* clicked **THEN** Smart TV Turn On |
| | | **IF** Living Room Tablet *label button: Light* clicked **THEN** Living Room Light Turn On |
| | | **IF** Living Room Tablet *label button: Tree* clicked **THEN** Light Tree Plug Turn On |
| App2 | RS2 | **IF** Alex Smartphone *list element: Phone Groceries* changed **THEN** Alex Smartphone synchronises UIe *list element: Phone Groceries* with Smart Fridge *list element: Fridge Groceries* |
| | | **IF** Smart Fridge *list element: Fridge Groceries* changed **THEN** Smart Fridge synchronises UIe *list element: Fridge Groceries* with Alex Smartphone *list element: Phone Groceries* |
| App3 | RS3 | **IF** Alex Smartphone *weather element: forecast* on UIe creation **THEN** Weather Forecast Service show weather in city: Brussels |
| | RS1 | **IF** Alex Smartphone *label button: Boiler* clicked **THEN** Water Boiler Plug Turn On |
| | | **IF** Alex Smartphone *label button: Light* clicked **THEN** Bedroom Light Turn On |
| | RS4* | **IF** Time = 8:00 **THEN** Water Boiler Plug Turn On **AND** Bedroom Light Turn On |
| | RS4 | **IF** Time = 8:00 **THEN** Water Boiler Plug Turn On |
| | | **IF** Time = 8:00 **THEN** Bedroom Light Turn On |

Table 3. Overview of applications and their corresponding rules. Smart devices and *things* are coloured in blue, services in pink and contextual elements in green.

The first group was asked to use the *interaction* view (pipeline metaphor) to create their first rule and then switch to the *rules* view (rules metaphor) to create their second rule. The second group was asked to start with the *rules* view for their first rule and then use the *interaction* view for their second rule. This approach allowed us to compare preferences for the two metaphors by observing how each group adapted to and evaluated both methods.

The study supervisor followed the participants' actions during the whole study on the duplicated screen and noted down when they had to intervene, help or clarify certain things. Lastly, for every finished application, participants were told to try out the application in order to see whether it had the expected behaviour.

*4.3.3 Interview and Questionnaire (15 mins.)* In order to capture the participants' experience and impressions of eSPACE, we used the Microsoft Reaction Card method [7], which consists of giving participants a set of words from which they have to select the ones that best describe the tool they just used. The word set contains negative, neutral and positive words. While the original set contains 118 cards, it is recommended to shorten this list depending on which words are more relevant for the tool being evaluated. We decided to retain the 40 words shown in Table 4, selected from the reduced product reaction cards list, prioritising those terms that held the most relevance for our research. We maintained the initial ratio of 60% positive words to 40% words that were either negative or neutral. Participants could select between 5 to 10 of these words and classify them either as 'highly relevant' or just 'relevant' with a maximum of 5 words per category. Note that the set of words has been randomised for each participant (by placing them differently in the table). Next, participants had to fill in the Post-Study System Usability Questionnaire (PSSUQ)[12] in order to have an indication of eSPACE's usefulness, information quality and interface quality. This questionnaire consists of 16 questions to be answered via a 7-point Likert scale and an N/A option. The lower the score, the better. After completing the PSSUQ, participants were asked to provide some additional feedback. This included rating the ease of creation of each app, indicating their preferred view (*interaction* or *rules* view), and sharing any comments or suggestions for improving the tool. Additionally, we conducted short interviews to understand participants' rationale for selecting specific words from

[12]https://uiuxtrend.com/pssuq-post-study-system-usability-questionnaire

the reaction cards and briefly discussed their preferred view and suggestions to enhance our tool. Finally, we collected demographic data through a questionnaire that covered standard questions such as age, gender, country of birth, and more. In this questionnaire, participants were also asked about their ownership of smart devices and IoT devices at home, as well as their self-perceived level of technological proficiency on a scale of 1 to 5. This information aimed to explore any potential relationship between participants' technological skills and their preferred view (i.e. metaphor).

| | | | | |
|---|---|---|---|---|
| Advanced | Creative | Exciting | Irrelevant | Straight Forward |
| Annoying | Desirable | Familiar | Organized | Stressful |
| Approachable | Difficult | Flexible | Overwhelming | Time-consuming |
| Attractive | Dull | Helpful | Predictable | Trustworthy |
| Comfortable | Easy to use | Inconsistent | Relevant | Unattractive |
| Complex | Effective | Ineffective | Reliable | Undesirable |
| Confusing | Efficient | Innovative | Satisfying | Unpredictable |
| Consistent | Engaging | Inspiring | Simplistic | Useful |

Table 4. Set of 40 Reaction Cards used in the user study alphabetically ordered. With 60% of the cards being positive words and 40% being negative or neutral words.

### 4.4 Study Results

In the following, we analyse the outcomes of our user study. We start by discussing how participants executed the different tasks they were given, along with the challenges they faced in creating each application. Following this, we share noteworthy observations made during the study. Next, we present our analysis of the reaction cards selected by participants. We then continue by describing the results of our post-study questionnaires and conclude by reporting on the insights gained from interviews on participants' suggestions for eSPACE.

*4.4.1 Task Execution.* All participants successfully completed the tasks and created the three applications as requested, which operated as intended. If participants or the study supervisor noticed that the application did not function according to the task description, the participants were prompted to rectify the incorrect behaviour. The average time to finish all three applications was 35 minutes. Notably, the first application took an average of 16 minutes, the second 8 minutes, and the third 11 minutes. In the subsequent paragraphs, we will detail our observations for each task and application.

For the *first task*, participants were required to create the "Simple Controller Application", a basic app comprising three buttons, each controlling a smart appliance. The purpose of this task was to familiarise participants with our authoring tool. Note that the app can almost be created entirely by following the demo video. The demo video demonstrates how one button can be used to turn on the Smart TV and Decorative Light Tree. In contrast, participants were asked to create three separate buttons, each controlling a different device and enabling both turning on and turning off functions. While observing participants perform this task, we noticed that all of them easily created the three buttons. However, almost all participants (20) associated the "turn on" functionality with these buttons, possibly influenced by the demo video they viewed earlier. It is plausible that the pre-selection of the "turn on" functionality in the option dropdowns in our tool contributed to this trend as well. Additionally, some participants were not familiar with the term "toggle" and, as a result, did not immediately recognise the option. This lack of familiarity with the term resulted in some participants requiring assistance during this part of the task. Seven participants received

some hints while creating the app. They mainly required hints to find the "toggle" functionality when changing their buttons' behaviour. Interestingly, 5 participants attempted to address the issue by creating a rule "IF button clicked THEN Smart TV Turn On AND Smart TV Turn Off", which, of course, did not result in the expected behaviour. Remarkably, despite being able to customise their applications, most participants (17) used simple rectangular blue "label buttons", like the ones shown in the demo video. During this task, 14 participants watched the demo video again to re-find specific passages, one participant used it as follow along video. Lastly, as mentioned earlier, half of our participants were asked to use the *interaction* view for creating the first rule and the *rules* view for the second rule, while the other half followed the opposite order. Despite this intentional division, when it came to creating the third rule, the majority (17) chose to use the *interaction* view. This initial observation suggested a preference towards the *interaction* view regardless of the initial view participants used, but a more detailed discussion is presented in Section 4.4.2.

For the *second task*, participants were given the flexibility to choose the view they preferred to create the synchronisation rule(s). The process of synchronising interfaces or folders is often desired to occur bidirectionally, ensuring that changes on one device are reflected on the other device and vice versa. To facilitate this "bi-directional" synchronisation, eSPACE provides a checkbox allowing users to indicate whether they want the synchronisation to occur both ways. When checked, the authoring tool automatically creates two rules: one for synchronising changes from the source device and another for triggering synchronisation when changes occur on the target device. In this task, participants were tasked with synchronising a grocery list between Alex's Smartphone and the Smart Fridge. Given that all participants left the checkbox checked, only one rule needed to be created, simplifying and expediting the task completion. Out of the 22 participants, only 6 opted to use the *rules* view for creating the synchronisation rule, and only one of them did not require assistance during the rule creation process. All 6 participants encountered challenges when filling in the devices on the "THEN" side of the rule. Based on observations from our pilot study, where participants struggled with this rule, we anticipated potential difficulties. As a result, we made adjustments and equipped the authoring tool with a warning mechanism to assist users. When detecting an incorrect device entry on the "THEN" side, the tool displayed a message advising users that it is generally better to use the same device as both the trigger and action device as illustrated in Table 3. While one participant was able to correct the rule based on the warning message, others still required some guidance from the study supervisor. The concept of trigger and action devices seemed unfamiliar to participants and proved non-trivial to grasp. Further note that, to gain space, the abbreviation "UIe" was employed for "user interface element", which unintentionally caused confusion among some participants. This issue will be addressed in Section 4.4.6. The 16 remaining participants used the *interaction* view for creating the synchronisation rule. They quickly understood the necessity of employing the synchronisation arrow. The popup window for filling in the rule information appeared to be more user-friendly, as no participant encountered significant difficulties with it. This popup window is shown in Figure 10. Lastly, we noticed that many participants did not consider creating the second rule and did not pay attention to the checkbox. This oversight may be attributed to potential confusion with the wording, since the word "target UI" might have been unclear, as most participants appeared to read the text. Only 4 participants actually asked about the bidirectional behaviour, and the study supervisor explained the purpose of the checkbox.

The *last application* presented a higher level of complexity, combining familiar rule structures with the integration of new elements, such as a weather forecast service and a contextual element (i.e. time). The first interaction described in the task description involved the weather forecast: "the app should show the weather in Brussels". To create this interaction, participants had to use the weather UI element in the *UI design* view and connect it to the weather forecast service. Given

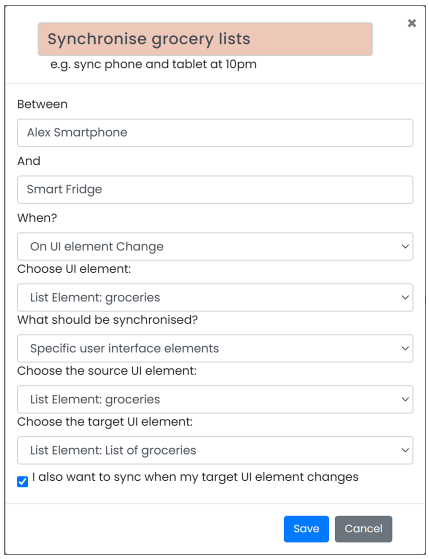



Fig. 10. Popup appearing when connecting the smartphone and smart fridge with a synchronisation arrow

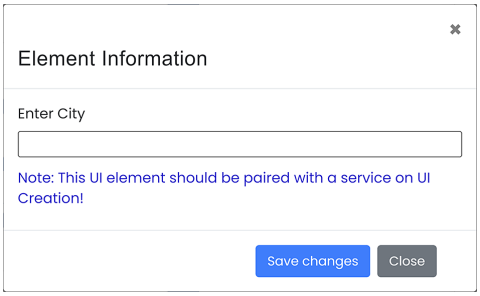



Fig. 11. Popup window appearing after dropping the weather UI element on the canvas

that the weather UI element should always be linked to such a service, dropping the element on the canvas prompted a note instructing participants to link it with a service as shown in Figure 11. Among the participants, 12 used the *interaction* view to create this rule. Although most of them efficiently found the weather forecast service, some confusion arose concerning the selection of the appropriate arrow for creating the interaction. Specifically, 8 participants believed that the synchronisation arrow should be used, while 6 participants mistakenly drew the interaction arrow in reverse, designating the weather element as the source instead of the target. This observation highlights the need for revisions to improve the interaction definition process for this particular case (see Section 4.4.6). The 10 remaining participants who used the *rules* view to create this interaction, created the rule very quickly. Only one participant experienced initial struggles, mistakenly designating the weather service as a trigger on the "IF" side instead of as a target on the "THEN" side. Based on our observations, it can be asserted that participants found the *rules* view easier to use for this particular rule. Regarding the creation of the functionality for controlling the light and water boiler using interface buttons, a slight majority of the participants opted for the *interaction* view. Notably, 13 participants employed the *interaction* view to create the interaction rule for controlling the water boiler, while 14 participants used it for the light control interaction. A solitary participant took a mixed approach, creating one interaction with the *interaction* view and the other with the *rules* view, while the rest consistently used the same view for both interactions. Finally, for the time-related rule, participants had the option to create either a single rule using the "AND" operator on the "THEN" side in the *rules* view, or create two separate rules, as shown in Table 3. Note that the demo video demonstrated the use of the AND operator to compose a THEN side, but no examples of contextual or service interactions were presented in the video. The purpose of this design was to assess whether users could intuitively identify this functionality without explicit instructions. 12 participants used the *rules* view to create the time interaction. Out of these, 7 employed the AND operator to generate a single rule, while the others created two

rules, one for turning on the light at 8 a.m. and the other one to turn on the water boiler plug. Conversely, 10 participants opted for the *interaction* view to create these interactions. During the process, 3 participants initially intended to use the synchronisation arrow but later realised that the contextual arrow should be used. Additionally, some participants told us they expected the time to come from the smartphone, leading to some confusion when they understood it was not the case.

*4.4.2 General Observations.* Throughout the user study, we recorded a variety of noteworthy observations. One observation pertained to confusion between clicking and dragging actions. For instance, when using the *interaction* view, 15 participants attempted to drag and drop the interaction arrow onto the canvas rather than clicking on the arrow to trigger the appearance of green dots. Conversely, when dragging and dropping devices from the left sidebar to the canvas, 5 participants inadvertently clicked on the devices instead, causing some confusion due to the lack of support for this action.

Another trend in the *interaction* view was that participants (15) often dragged and dropped all necessary elements for application creation before progressing to UI creation process. In terms of rule creation, almost all participants (except two) at least once overlooked filling in the name of the rule, indicating a need for improved visibility for this field. Another concern arose when 9 participants overlooked the red dots while attempting to establish interactions between two elements. These red dots indicate that certain canvas elements cannot be connected using the selected interaction arrow. To address this, an informative tooltip could be incorporated in future iterations instead of merely preventing the selection of the dots. Further, we also observed that 16 participants encountered difficulties in selecting the dots, being unsure whether they should drop the arrow's tip onto the dot or their cursor. This problem had been observed during the pilot user study of eSPACE, we therefore enlarged the dots and employed a hand-shaped cursor to indicate dropping, but this was not enough to solve the problem. Further solutions for this issue will be discussed in Section 4.4.6. In the current version of eSPACE, the *interaction* view does not yet allow rule editing, but 7 participants still attempted this by (double) clicking an arrow, suggesting potential implementation in future versions.

Regarding the *rules* view, we noted interesting behaviours. Strikingly, many participants (16) did not use the rules container in the *home* view to access the *rules* view. Instead, they navigated to the *interaction* view and used the rules container in that view to go to the *rules* view. Additionally, 8 participants did not immediately grasp that the rules dropdown at the bottom of the *interaction* view was expandable. Within the *rules* view, after creating a rule and saving it, participants often found the redirection to the *home* view slightly time-consuming. As observed, 13 participants pressed the close button of the popup window before the page could redirect. An improvement in comparison to our initial eSPACE prototype was the inclusion of a navigation button within the *rules* view that directs users to the *interaction* view. This addition proved valuable as 8 participants used this button.

The *UI design* view yielded notable observations as well, for example, 17 participants exclusively used simple label buttons instead of exploring our other button options. Furthermore, only one participant used the `style` tab to customise their UI, editing the colour of the list element and label font sizes. Given that these customisation options were not showcased in the demo video, most participants were unaware of this possibility and did not discover this on their own.

We also observed that during the study, participants often sought explanations for certain concepts. For instance, 8 participants inquired about the meaning of "toggle", and 11 participants asked about various aspects related to "UI", including its definition, and the meaning of "UIe creation" and "UI change".

Various interesting rule mistakes were observed as well. For instance, 8 participants attempted to create UI-related rules before designing the UI. During rule creation, 12 participants at least once inadvertently chose the wrong button for some interaction. Lastly, the most common mistake—made by 20 participants—was the selection of the "turn on" action instead of the "toggle" action during the first task.

Given that our tool is still at the prototyping stage, we also identified some bugs during the study. Although these were mostly minor and easily addressable, they may have impacted the user experience. For example, connecting green dots that were close to each other sometimes led to accidental creation of two interaction arrows instead of one (4 participants). Another issue involved saving an empty rule when the participant initiated rule creation but then cancelled (4 participants). Finally, the warning message concerning the syncing interaction in the *rules* view occasionally appeared without cause (6 participants). While the study supervisor clarified that these were not the participant's fault, these bugs may still have influenced their overall experience.

*4.4.3 Reaction Cards Analysis.* After the creation and testing of the three applications by the participants, we proceeded by applying the Microsoft Reaction Card Method [7]. Therefore, we asked participants to choose a maximum of ten words that best described their experience with our tool, words that they could categorise either as *relevant* or *highly relevant* (max. 5 per category). Based on the selected words and their categorisation by our participants, we created the word cloud shown in Figure 12.


Fig. 12. Word cloud based on the selected reaction cards

The term "Useful" was the most frequently selected word, chosen by 17 participants. Participants mentioned that they found the tool useful and could see themselves configuring smart devices and *things* at their homes with it. For instance, P6 stated, "*certainly useful, saves you time and it is simply handy to fix all these domotics*". Moreover, several participants (P7, P11, P13, P16, P18) recognised the tool's usefulness even though they did not yet own smart *things* at home. One, for example, expressed appreciation for its capability to control various devices, and particularly highlighted their liking for the synchronisation feature of the grocery list across devices (P16).

Following closely were the words "Innovative" and "Easy to use", selected by 11 participants. Many of them found the tool innovative as they had never encountered something similar before. Some mentioning the possibility to accomplish various tasks without requiring any programming skills. P22 commented, "*innovative because you can still build something in an 'approachable' way*". Furthermore, the tool sparked interest in IoT technology and inspired ideas to improve their daily routines. Participants who chose "Easy to use" expressed that once they got used to the tool, they could easily perform the required actions, and further mentioned that it quickly became intuitive.

For instance, P4 stated, "*once you know how it works you can find things easily by simply searching a bit*".

Other recurrent words were, "Exciting", "Satisfying", "Engaging", "Efficient", "Approachable" and "Effective", chosen by 9, 8 or 7 participants. Nine participants used "Exciting" to describe their experience with the tool, expressing that it was really nice to "*play around*" with and thinking "*oh my god, I can do this*" (P7). Many participants (8) found it "Satisfying", especially when witnessing their apps in action on the house plan. P5, for example, remarked that they could say: "*hey, I created this!*". The words "Efficient" and "Effective", chosen by 8 participants, highlight their belief that the tool served its purpose efficiently and effectively. P15 commented, "*it is effective, efficient, it does what it does and it helped me set the rules very quickly so it is very efficient*". Nine participants selected the word "Engaging", one even likening it to a game (P6). The term "Approachable" was chosen by 7 participants, with five placing it in the "highly relevant" category. They explained finding the tool very approachable as "*anyone could use it if they see the (demo) video*" (P4).

A few negative words were also mentioned. "Confusing" and "Complex" were picked by 7 and 6 participants, respectively. Some mentioned that it was a bit confusing at first because they expected to find certain elements at certain places, so it is a bit searching at first but once you find it you get used to it (P8, P9, P17). Others initially struggled to find the right steps to follow, for example, first create the UI and then add functionality (P10, P21). Certain participants mentioned that it was "Complex", mainly due to their lack of experience with such tools and their unfamiliarity with the vocabulary (P10, P20, P21). The presence of multiple ways to achieve the same outcome also contributed to the complexity (P16, P22). Lastly, it is worth noting that 5 participants found the tool "Time Consuming". P1, for example, remarked that the process of modifying rules took some time, others mentioned they needed time to get used to the tool (P10, P16).

Overall, participants provided overwhelmingly positive feedback, perceiving eSPACE as a useful, innovative, easy to use tool, which is approachable, engaging, efficient and satisfying to use, which is a gratifying result.

*4.4.4 Post-Study Usability Questionnaire.* Upon completing their selection and explanation of the reaction cards, participants proceeded to fill out the Post-Study System Usability Questionnaire (PSSUQ). The analysis of the PSSUQ results will provide insights into the usability of eSPACE, contributing to the validation of the applicability of our design guidelines outlined in Table 2. This step is crucial for confirming that the guidelines, rooted in users' mental models, effectively enabled us to implement an authoring tool with good user-friendliness and usability.

The PSSUQ gives an overall score corresponding to the user's perceived satisfaction with the tool. The questionnaire is composed of 16 questions employing a 7-point Likert Scale (with an additional N/A option). A mean score of 4 indicates a "neutral" response, and the lower the value, the more favourable the result. The questions can be categorised into three subgroups: the system usefulness (SYSUSE) (average scores of questions 1 to 6), the information quality (INFOQUAL) (average scores of questions 7 to 12) and the interface quality (INTERQUAL) (average scores of questions 13 to 15). Before delving into these categories, we present an analysis of the responses to each individual question. Figure 13 provides an overview of the positive versus negative responses for each question.

One striking observation concerns the negative responses received for question Q7 (mean: 5.15). This question pertains to whether the "system provided error messages that clearly guided me in resolving issues". Given that eSPACE currently offers limited error messages, we expected such a response. Participants evaluated this question unfavourably mainly due to the absence of error messages. It is also worth mentioning that the warning message related to rule creation for the synchronisation interaction in the *rules* view was perceived as an error message by participants. Given

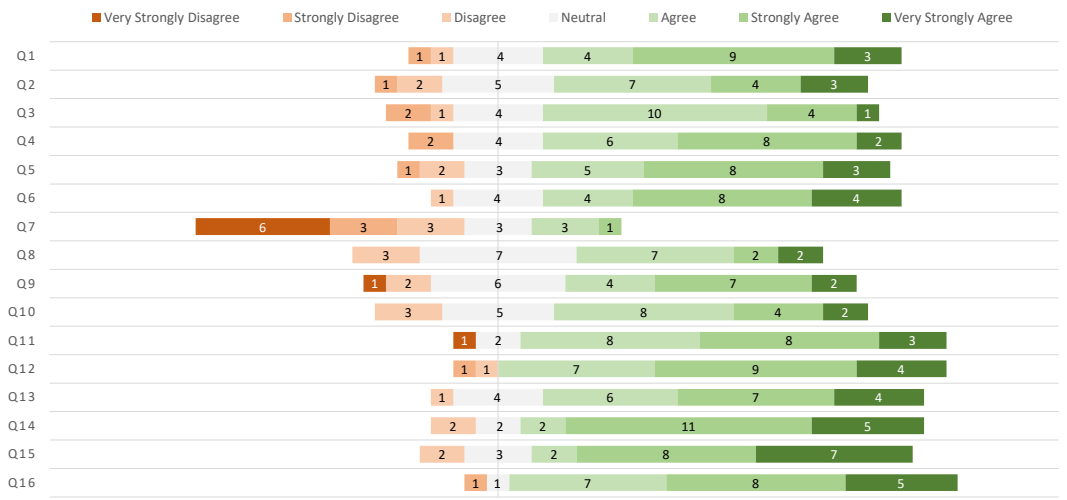



Fig. 13. Diverging stacked bar chart showing participants' responses to each question of the Post-Study System Usability Questionnaire (PSSUQ). Responses are shown on a scale ranging from 'Very Strongly Disagree' to 'Very Strongly Agree'

that this message was hard to understand by participants, it also contributed to the unfavourable rating. Further note that, three participants marked this question as "not applicable".

Regarding the remaining questions, the mean values lie between 2 and 3.5 on the 7-point Likert scale, indicating that participants leaned towards positive responses. Question Q8 (mean: 3.33), asked participants whether they could easily and quickly recover from their mistakes. This question stood out as the next least-favourably rated. One possible explanation is that participants found the process of navigating between views to rectify an action (e.g. correcting a "turn on" action to a "toggle" action) somewhat cumbersome. Notably, as observed in Section 4.4.2, almost all participants did not use the rules container in the *home* view to correct rules. Instead, they navigated to the *interaction* view to locate the relevant rule in the rules container of that view, which prolonged the correction process. Additionally, question Q2 related to being "able to complete the tasks and scenarios quickly using this system", received a close mean rating of 3.27. In contrast, the most positively rated questions were Q14 and Q15 (means: 2.32), which pertain to participants' appreciation of the eSPACE interface and its alignment with anticipated functions and capabilities.

Upon categorisation of the questions into the aforementioned subcategories, it becomes apparent that interface quality garnered the best mean score. This outcome suggests that our user interface, based upon the design guidelines delineated in Table 2, yielded commendable user satisfaction and experience. Notably, this category also displays the lowest standard deviation of 0.89 as shown in Table 5, indicating a high level of agreement among participants.

Next, the system usefulness scored a mean value of 2.88, highlighting participants' comfort and satisfaction with the tool's user-friendliness and learnability. As anticipated, eSPACE attained a slightly lower rating for information quality, primarily due to question Q7. Nonetheless, a mean value of 3.23 still represents a positive outcome. In summary, the overall usability score for eSPACE is 2.90, indicative of its good usability. To enhance interpretation, we compared the means per subcategory with the norms established by Sauro and Lewis [81] based on 21 studies involving 210 participants. The comparison is presented in Table 5, along with the corresponding standard deviations. Lower values are indicative of better outcomes.

Taking a closer look at this table reveals that our means are comparable to those of Sauro and Lewis, meaning that eSPACE exhibits robust user satisfaction. It is noteworthy to mention that during the study, a participant (P21) expressed discomfort throughout the session, likening the experience to taking an examination. This sentiment was evident in their chosen reaction cards and questionnaire responses. Although P21's case could potentially be treated as an outlier and excluded, we chose to retain this instance as it may mirror real-world users who may encounter stress while using the tool. However, for the sake of completeness, we also conducted the analyses excluding P21. The mean values without P21 are depicted in the final two columns of Table 5. This exclusion led to decreased standard deviations in all categories and an improvement in overall scores, aligning even more closely with Sauro and Lewis's norms.

Based on these results, we can validate the use of the design guidelines for building a user-friendly and usable authoring tool. In the subsequent section, we will further assess the extent to which the pipeline an rules metaphor have been positively received by end users for creating the interactions between devices.

| | Sauro and Lewis [81] | eSPACE's Means | eSPACE's SD | eSPACE's Means (without P21) | eSPACE's SD (without P21) |
|---|---|---|---|---|---|
| **SYSUSE** | 2.80 | 2.88 | 0.98 | 2.76 | 0.80 |
| **INFOQUAL** | 3.02 | 3.23 | 0.99 | 3.17 | 0.97 |
| **INTERQUAL** | 2.49 | 2.41 | 0.89 | 2.29 | 0.69 |
| **Overall** | 2.82 | 2.90 | 0.81 | 2.79 | 0.67 |

Table 5. Means and standard deviations (SDs) per subcategory compared to results of Sauro and Lewis

*4.4.5 Post-Study Tool Questionnaire.* After completing the post-study system usability questionnaire, participants were also asked to provide their input on a few additional questions regarding their perceived ease or difficulty in creating each application, their preferred view, as well as any suggestions or general comments to enhance the tool.

Ultimately, all three applications received an average rating of "neutral" on a 5-point scale ranging from "very easy" to "very difficult". None of the applications were classified as "very difficult" to create by participants. Overall, participants reflected that in hindsight, the first app was the easiest to create. However, since they were still acquainting themselves with the tool, many refrained from marking this particular app as "very easy" to create. Regardless, 12 participants still marked it as "easy" (9) or "very easy" (3). The "Grocery List Application"'s average rating leaned the most towards an "easy" scoring, with 13 participants rating it as such (9 for "easy" and 4 for "very easy"). Notably, the 3 participants who perceived this app as "difficult" were those who employed the *rules* view to define the synchronisation rule. The final app received a "neutral" assessment as well, which can be attributed to its incorporation of multiple new elements (i.e. time and weather forecast). 8 participants categorised it as neutral in terms of difficulty. In summary, juxtaposing the creation of these three apps did not yield substantial distinctions, considering that participants continually had to familiarise themselves with novel elements when transitioning to the next application.

The most interesting query revolved around the participants' preferred view, either the classical *rules* view commonly present in most IoT applications or the *interaction* view mainly based on design guidelines G1 and G2 inferred from people's mental models. The results were evenly divided: 10 participants favoured the *rules* view, 9 leaned toward the *interaction* view, and 3 expressed equal fondness for both views.

To explore whether the starting view impacted the preferred view, given that half of the participants began with the *interaction* view and the other half with the *rules* view, a chi-square test

was conducted. The null hypothesis was that there is no correlation between users' preferences and their initial view usage. The obtained p-value was *0.0948*, suggesting that, at the conventional significance level of 0.05, we do not have sufficient evidence to reject the null hypothesis. This implies that users' preference for a specific view is not significantly related to the first view they used.

Furthermore, the potential influence of participants' technical knowledge on their view preference was assessed as well. The null hypothesis for this test was that there is no association between users' technical knowledge level and their preference for either the *rules* view or the *interaction* view. The obtained p-value for this test was *0.44*, indicating no significant correlation between users' technical knowledge and view preference.

During interviews with participants who favoured the *interaction* view, several key points emerged. P19, for example, noted that they found the popup window easier to comprehend. P22 also found the *interaction* view easier for directly selecting necessary elements, which they considered simpler than in the *rules* view. Some participants noted the complementary nature of the two views. P20 mentioned preferring the *interaction* view while still intending to utilise the *rules* view for its added functionality. Participants who preferred the *rules* view, mentioned to like the textual nature of the view, like P21 who favoured the *rules* view due to their affinity for textual representation and enabling them to read the complete rule more effectively. P14 indicated an appreciation for connecting devices using arrows as well, yet expressed a preference for rule creation within the *rules* view. They suggested being redirected to the *rules* view after creating the arrow, while also proposing to use the *interaction* view primarily for a more visual representation, which was also mentioned by P17. Lastly, P7 who appreciated both views, shared that they preferred the rule creation in the *interaction* view and subsequent editing in the *rules* view.

*4.4.6 Interview on Suggestions and Discussion of Improvements.* During the post-study discussions, participants offered a range of valuable suggestions that merit some attention and based on which we describe potential refinements for a future iteration of eSPACE. Many participants expressed the need for additional explanations pertaining to elements not covered in the demo video. Some put forth the idea of adding informative tooltips when hovering over elements (P8, P9, P20) and/or proposed the creation of concise instructional videos targeting specific functionalities of eSPACE (P5, P8, P10, P17, P18). Additionally, a glossary containing complex terms and abbreviations (e.g. UIe) was suggested by participants to ease understanding (P6).

Initially intended to facilitate the testing of IoT applications, the house plan received favourable feedback from participants, some of whom recommended its direct incorporation into the tool (P6). The integration of a customisable house plan feature into eSPACE would not only provide real-time insights into the status of each device in the smart home, but also enable direct control over the smart devices through the house plan interface, similar to Jeong et al. [43]'s work. This new feature would enrich the user experience as well as further contribute to the support of guideline G3 related to providing realistic graphical representations of devices.

Regarding the navigation, a subset of participants found transitioning between views somewhat challenging (P3, P9, P19). Notably, our observations revealed that many participants accessed the *rules* view from the *interaction* view rather than the *home* view. To address this, enhancing navigation visibility, such as automatically expanding the rules dropdown in the *home* view, could offer an effective solution. Furthermore, some participants disliked being redirected to the *home* view after rule creation in the *rules* view. P1 mentioned a preference for immediately initiating the creation of a new rule within the *rules* view, noting this approach would save time. Consequently, it may be beneficial to offer users the option, upon saving a rule, to either return to the *home* view or proceed directly with the creation of another rule.

We also received feedback concerning the process of creating rules in the *interaction* view (i.e. by connecting the green dots surrounding canvas elements). Both P22 and P13 pointed out that refining the connection between these dots could enhance the user experience by eliminating the need for precise dot selection. During our observations we noted challenges when selecting a dot surrounding the targeted canvas element as well. Therefore, we aim to simplify the selection process by introducing a "halo" mechanism around the canvas elements. This design involves encircling actionable elements with a green halo and non-actionable elements with a red halo, thereby expanding the selection area and reducing the need for precision in selecting connection points. Figure 14 illustrates the proposed design, which is inspired by the Platform Composition [71] system. By surrounding the elements on the canvas with a halo instead of multiple dots, participants can simply drop the arrow target inside the halo area to connect elements on the canvas, which demands less precision on the user's part.

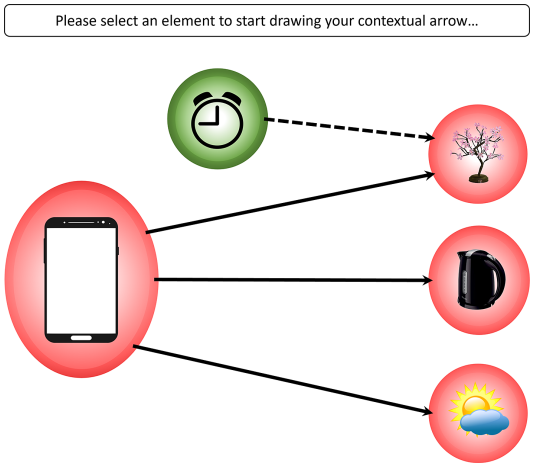



Fig. 14. Suggested improvements of the *interaction* view showing the canvas state upon selection of the contextual arrow which shows the compatible and non-compatible canvas elements surrounded by a green or red halo respectively

In addition, another improvement involves adding a directive text prompt at the top of the *interaction* view. This prompt would guide users in the selection of an element (the trigger) to start drawing the interaction arrow, clarifying that users should not attempt to drag and drop the arrow onto the canvas element, but rather initiate the process by selecting it.

As for enhancements to the *rules* view, we plan to incorporate small icons in the tiles and alongside dropdown options to improve comprehensibility of the different options and further enhance user experience. A suggestion that has been made by P9.

Certain participants also proposed to add automation into the tool. Currently, our tool already provides some level of automation, as illustrated with the synchronisation rule. One should normally have to create two rules, one for synchronising when a change happens on the source device and one for triggering synchronisation when a change happens on the target device. In eSPACE, upon creation of one rule, the tool proposes the automatic creation of the second rule. This automation feature has proven to be highly effective, as all participants used it, even if often unconsciously. We plan to introduce additional automation features. P8 proposed a feature where placing a button on the *UI design* view would triggers suggestions for linking associated functionality. For example, when a UI element that requires linkage to a specific component, like a weather forecast widget

needing connection to a weather service, is placed on the canvas, the tool could recommend this linkage. Upon user confirmation, it would automatically establish the connection or guide the user to the appropriate view for further action. This aspect warrants further exploration. Moreover, as mentioned by P11, one could also already pre-select the UI option in the dropdown menus when they navigate from the *UI design* view to the *rules* view, avoiding the need to manually look after the correct UI elements in the dropdown options in order to link functionality to them. Furthermore, P7 expressed a desire for users to receive assistance for common errors through a feedback loop mechanism, which is something to consider as well.

Last but not least, P14 suggested a time-saving approach for editing specific application rules. They proposed allowing direct authoring when accessing the app on a smartphone or tablet. This approach would eliminate the need to return to the authoring tool when identifying rule errors while using the app. The idea arose from a situation during the user study's first task, where altering multiple "turn on" actions to "toggle" actions proved time-consuming after the error was noticed while using the app. We find this idea valuable and intend to explore the inclusion of this new authoring mode in future iterations.

With these planned improvements for the next version of eSPACE, our goal is to further enhance its usability and explore new end-user authoring opportunities. The proposed suggestions also provided us with interesting ideas for future research directions, which will be presented in Section 5.

### 4.5 Summary and Limitations

The user study on eSPACE produced insightful results, validating the applicability of the design guidelines outlined in Table 2 and confirming the effectiveness of the pipeline metaphor in supporting the user-friendly creation of interaction rules. Additionally, the study offered a rich set of ideas for future enhancements aimed at improving user experience, which have been discussed in the previous section. In this section, we briefly summarise our general findings and limitations of the user study.

All participants successfully completed the assigned tasks, demonstrating their capacity to create three functional applications. For the first application, we noticed that participants frequently mistakenly selected the "turn on" action instead of "toggle" to control IoT devices. This pattern invites further investigation to ascertain whether this phenomenon stems from vocabulary nuances, instructional ambiguities, or mere inadvertence. From the observations gathered during the creation of the second application, we could see that the *interaction* view (which used the pipeline metaphor) seemed easier to use for defining the synchronisation rule than the *rules* view. Improvements should be made to simplify the rule creation in the *rules* view for the synchronisation interaction. Therefore, further research or studies should be performed investigating how end users would formulate such a rule using an IF-THEN statement, similar to the study performed by Dey et al. [29] for context-aware applications. For the third application, which included two new components not covered in the demo video, most participants figured out how to use these new components, yet some required extra guidance. This suggests that there is a need for some short demo videos explaining specific functionality as well as extra information in the tool about the different elements that are present. As seen in many recent tools, we should integrate on-hover tooltips providing additional information for each element.

The participants' selection of reaction cards overwhelmingly indicated positive feedback for eSPACE, with terms like "Useful", "Innovative" and "Easy to use" being the most selected for describing their user experience. Additionally, eSPACE also scored well with the post-study system usability questionnaire (PSSUQ). Its best score was on the interface quality (2.41), followed by the system usefulness (2.88) and lastly the information quality (3.23). The lower score for information quality reflects the previously mentioned issues regarding insufficient explanations and details

about the tool's functionalities. The high score in interface quality underscores the effectiveness of the design guidelines for implementing a user-friendly interface for the eSPACE authoring tool. With an overall score of 2.90, eSPACE demonstrated robust performance and garnered considerable user satisfaction.

A particularly notable outcome from the additional post-survey questionnaire was the divided preference among participants regarding the *rules* versus *interaction* view. As 10 participants preferred the *rules* view, 9 the *interaction* view and 3 expressed equal fondness for both views. This result validates design guideline G6, which recommends providing both options in an end-user authoring tool, and also aligns with the findings of Brich et al. [11]. Despite conducting a statistical analysis using the chi-square method, no significant relationship was found between a participant's technical proficiency and their preferred view nor between the participant's first view usage and view preference. However, insights gathered during interviews suggested that IF-THEN rules may be more appealing to advanced users who desire greater control over the behaviour of applications.

We conclude this section with a brief overview of our study's limitations. One notable limitation is the sample of participants, most of whom come from Western cultures and are therefore accustomed to left-to-right reading and interface layouts. It would be valuable to conduct the study with a more diverse group, including participants from cultures where reading occurs from right to left, to ensure the findings are more representative of a global population. We will explore further localisation research possibilities in Section 5.4. Additionally, regarding the study design, it would have been ideal to set up eSPACE in participants' own homes for an extended period, giving them more time to familiarise themselves with the tool and experience how home automation would function in a real-life context. Direct interaction with the actual benefits and challenges of the technology might alter their perception. Despite these limitations, our results are highly positive and demonstrate that eSPACE successfully answers our main research question by employing appropriate abstractions to hide technical complexity, enabling end users to effectively control their smart environments and create personalised applications for managing their devices and things. The study also provided valuable insights for future research directions, which we will discuss in the next section.

## 5 FUTURE RESEARCH DIRECTIONS AND CHALLENGES

Inspired by related research outcomes and guided by observations and participants' suggestions during our user study, we have identified seven potential future research directions in the context of End-User Development (EUD) tools for cross-device and IoT interactions, along with their associated challenges. We present opportunities for enhancing the usability and effectiveness of such tools and aim to enrich the landscape of end-user authoring tools, offering insights that contribute to their broader adoption and utility. In the following, we discuss each of these future research directions in detail, highlighting their significance and potential impact on the field.

### 5.1 Exploration of Multiple Authoring Modes

While we believe that the most efficient approach for authoring cross-device (XD) and IoT applications is via a desktop or laptop interface offering more screen space than smartphones, the incorporation of additional authoring modes should be considered. As pointed out by a participant during our user study, there is potential to extend authoring capabilities to the devices on which the created applications will operate. This would empower users to perform quick adjustments, such as rectifying rule errors or fine-tuning UI element positions, without the need for a computer.

In existing research, tools have been introduced that advocate authoring through tablet [6] and smartphone interfaces [22]. However, these tools are confined to mobile authoring interfaces. Therefore, a potential area for future research could be to investigate the feasibility of providing end

users with multiple authoring modes. This would lead to the exploration of optimal strategies for unifying various authoring methods into a single, cohesive tool that seamlessly blends desktop and mobile authoring modes. Note that a similar approach is chosen in XDStudio [61], a tool facilitating the development of cross-device web interfaces for web designers. XDStudio features two authoring modes: a simulated mode and an on-device mode. More research could be conducted to extend the approach to the context of end-user development for smart home applications.

Additionally, we have noticed an emerging research trend involving augmented reality (AR) in the authoring process [87], which might also present an additional avenue for EUD authoring. Investigating the harmonisation and integration of these diverse modes within a unified and cohesive tool for end users poses a compelling area of investigation for the future, which could increase adoption and enable faster and more innovative authoring methods.

### 5.2 Classification of Different Interaction Rules

The classification of interaction rules into distinct categories poses a challenge. Our initial classification into *device*, *contextual*, and *synchronisation* interactions in the *interaction* view requires some revision. As revealed by our study, the device interaction category did not always fit the user's expectations, particularly exemplified by the interaction involving displaying the weather forecast. Concretely, the rule "IF Alex Smartphone weather element: forecast on UIe creation, THEN Weather Forecast Service show weather in city: Brussels" (see Table 3), was often not perceived as a device interaction since participants often wanted to use the contextual or synchronisation interaction arrows instead. This experience underscores the need for meaningful categorisation. Furthermore, the complexity of categorising interaction types increases as the functionality of the tool expands. A study conducted by Ghiani et al. [37] on their TARE authoring tool, which inspired the rules view of eSPACE, also revealed that users had difficulties in finding the right trigger for composing their rule. This was due to a mismatch between the user's mental model and the trigger tree structure. The trigger categorisation did not correspond to what the users had in mind. These findings confirm that the challenge of creating a coherent classification framework for various rule types (as well as trigger and action types) warrants further examination.

This challenge parallels the diverse vocabulary used across the IoT application landscape, where tools and applications deploy varying terms to convey analogous concepts. Investigating an effective categorisation approach could also pave the way for a standardised vocabulary in trigger-action rules, as discussed in more detail in the next section.

### 5.3 Investigating End-User-friendly XD, IoT and UID Terminology

Finding the right terminology for end-user authoring tools remains a critical challenge. Designing tools for users with limited technical knowledge requires not only metaphorical resonance, but also clear and appropriate terminology [14, 88]. The proliferation of different terms for similar concepts in IoT applications makes it even more difficult to create user-friendly tools. This challenge has also been identified by Lucci and Paternò [52] who proposed the development of vocabularies that are easy for end users to understand. The authors further propose the use of the card sorting technique to understand how users think about and categorise different concepts that characterise IoT and context-aware applications. Other studies have been performed in order to find the users' conceptual models regarding context-awareness, investigating how users think about rule creation and composition as well as the related terminology. For instance, Dey et al. [29] gathered 371 application descriptions from user interviews which informed the design of iCAP, a visual EUD tool for context-aware applications. Although the primary focus was not on the vocabulary, the accumulated data potentially offers insights into the language employed by participants. Instead of solely initiating new studies, a constructive approach might involve collating data from existing

end-user studies. Leveraging this existing work not only provides new insights on existing data but also offers a time-saving advantage. Such an analysis whether on existing studies or by performing new ones is essential for developing user-friendly vocabularies for EUD tools.

During our user study, we already have highlighted certain terms posing comprehension challenges for end users. Noteworthy among these are user interface-related terms like "user interface element" and its abbreviation "UIe", and rule-focused terms such as "trigger" and "action device". While some terms could be reformulated based on the findings from end-user interviews and the identification of a user-friendly vocabulary, others might be more difficult to replace. To bridge this comprehension gap, other techniques will be needed to help users understand those terms. As suggested by some participants and used in many tools, explanatory tooltips and short dedicated tutorial videos could be integrated into EUD tools. However, this calls for additional investigation to see whether these techniques are effective in helping users understand the tool and its vocabulary.

While there is no one-size-fits-all solution, finding the right terminology is an ongoing challenge. Therefore, it is important to continue researching effective ways to introduce new terminology in a way that is easy for users to understand as well as using as much as possible terminology that closely matches with the users' mental models.

### 5.4 Cultural Sensitivity in End-User Authoring Tool Design

Taking into account cultural aspects is important when developing more user-friendly and inclusive tools. As discussed earlier, the choice of terminology and the use of appropriate vocabulary are critical in designing authoring tools for users without any technical background. When selecting this terminology, one should also consider the user's cultural background, as the meanings and connotations of terms might vary depending on the cultural context. The concept of GILT (Globalisation, Internationalisation, Localisation, and Translation) provides a comprehensive framework for addressing these cultural variations [32]. It involves designing adaptable products, refining them for specific linguistic and cultural needs, and ensuring usability across diverse contexts.

Most research in the IoT domain has focused on conducting user studies with participants from varying technological backgrounds (e.g. engineering, research, health, sales, or administration) as seen in [11, 14, 29, 52, 76]. While this focus is valuable, it would also be interesting to investigate how cultural differences influence a user's knowledge about this XD and IoT end-user authoring tool terminology. For instance, a study by Miraz et al. [58], which analysed multilingual UI design issues on the BBC website, highlighted the need for greater attention to text creation, particularly in relation to users' cultural-linguistic heritage. Applying the GILT principles could help in addressing these multilingual challenges.

Beyond terminology, future research could explore other aspects of authoring tools that may require adjustments to better align with users' cultural backgrounds. For instance, a master's thesis by Ridderstråle and Sigfridsdotter [72] compared UIs designed for Swedish and Japanese users, leading to valuable design guidelines. These guidelines revealed cultural preferences, such as Swedish users' inclination towards list layouts and Japanese users' preference for card layouts. Furthermore, aesthetic choices, such as colour schemes, were found to vary significantly between cultures. Ridderstråle and Sigfridsdotter highlighted that cool and warm blue tones in Swedish design evoke calmness and relaxation, while in Japanese culture, similar effects are achieved using cool and warm green tones. These findings underscore the importance of investigating how various elements of authoring tools might be (automatically) customised based on the user's cultural background, while also allowing for further (manual) personalisation. Aspects such as layout preferences and colour schemes could be tailored to different cultural contexts to enhance user experience. Miraz et al. [57] presented a literature review discussing the concept of universal

usability, the plasticity of UI design, and how adaptable interfaces hold promise for achieving better cross-cultural usability.

We acknowledge that the design guidelines we currently rely on were largely derived from studies conducted in Western Europe, and their validation was primarily carried out with individuals from or residing in Western European countries. Therefore, future studies should aim to include participants from diverse cultural backgrounds. This could lead to the development of expanded design guidelines that are applicable to countries with specific cultural backgrounds. For instance, it would be valuable to investigate whether global versus localised visual representations of common icons, such as those representing "time", "devices", and "users" in eSPACE, are needed in different cultural contexts. Bühler et al. [13] proposed a human-centred design process that designers and researchers can follow to create more suitable and intuitive visual representations.

Designing interfaces that adapt to the cultural needs of end users requires further investigation, as these needs can be translated at various levels of end-user authoring tools, including presentation, language, dialogue design, and interaction design [39]. This represents a promising area of ongoing and future research, with the potential to develop systems that adapt to the cultural preferences of users, thereby reducing mental workload and increasing user satisfaction.

### 5.5 Multimodal End-User Authoring Approach

In a preceding section, we highlighted an emergent interest in using augmented reality technologies for authoring applications. Nevertheless, the integration of gesture-based inputs into authoring tools does not necessitate the exclusive use of AR technologies. For instance, as demonstrated in *Improv* [16], gestures can be captured via a touchsceen and then used as new input methods. One could explore the possibility of integrating gestures as potential triggers, or alternatively, examine the feasibility of incorporating gesture interactions as an additional authoring mode.

Moreover, including voice interactions also offers an extra dimension to enhance authoring tools. The growing popularity of voice interfaces, exemplified by the widespread adoption of voice assistants like Amazon's Alexa, and the Google Assistant, underscores a need for investigating effective voice-driven interactions [8]. Given that voice interactions, much like gestures, are characterised by their "hidden" nature, they present similar challenges to those identified by Norman and Nielsen regarding concealed gesture interactions [64]. These well-documented challenges and concerns create opportunities for exploring the integration of multimodal support into our end-user authoring tools. Whether they are introduced as part of the authoring process or as trigger/action options in the rule creation, each approach presents its unique set of challenges. In the former, users must adapt to new interaction modes, necessitating research into how these new authoring modes can enhance the user experience and contribute value to the authoring process. It is essential to investigate whether there is indeed a demand for multimodal authoring modes and how they can be optimally leveraged.

In the latter case, where multimodal support is integrated into the rule creation process itself (e.g. using gestures as triggers or voice as actions), new visualisation methods need to be explored to present these diverse triggers and actions in a way that is comprehensible to end users. One potential approach worth exploring is the incorporation of gesture catalogues within the rule creation pages, potentially combined with the vocabularies discussed in the previous section.

The integration of multimodal support in EUD tools represents an ongoing and relatively unexplored research area, offering substantial potential for future research opportunities. Note that promising results in this domain have already been reported recently by Barricelli et al. [3] in their exploration of a multimodal approach combining vision, speech, and touch for creating smart speaker routines. The use of voice for creating trigger-action rules has also been investigated by Roffarello and Russis [74], who's study outcomes suggested that a balance of automation and human

dialogue is needed when designing an intelligent personal assistant for voice-based IoT automation composition. Additionally, the study revealed that their multimodal prototype—using voice for defining the trigger and physical demonstrations for action—yielded a superior success rate in comparison to a prototype reliant solely on voice commands.

### 5.6 Intelligibility

Another emerging trend in the IoT domain is adding more *intelligibility*. As previously explained, users often find it challenging to understand the capabilities of a system and assess whether their created rules work as intended. Research in the field of intelligibility aims to address these challenges by informing users of a system's capabilities and understandings, providing them with feedback so that they become aware of what the system knows, how it knows it, what it is doing about it, and ensuring that users remain in control over a system's actions [5].

Over the years, intelligibility has been included in various context-aware research [51, 86] and is now becoming more important in IoT research as well [17, 18, 42]. During our user study, many participants selected the "turn on" action instead of the "toggle" action for the first application. Some participants directly understood their mistake while others needed more time to understand how to change the buttons' actions so that they could be used to toggle on and off the IoT devices. By adding some intelligibility features, one could help users grasp the actions they created while using their applications as well as help users understand while authoring the outcome of the rules they are creating.

However, ensuring that these explanations are easy to understand remains a significant challenge. Additionally, we need to consider the issue of intrusiveness, as we do not want to overload users with unnecessary notifications about their smart homes. Furthermore, as identified by Jakobi et al. [42], intelligibility needs evolve over time, as users transition from a desire to learn about the system towards wanting to monitor the system and ensure its proper functioning.

Hence, this field remains inherently complex, offering many promising future research opportunities. These opportunities lie in the exploration into the seamless integration of intelligibility within end-user authoring tools tailored for cross-device and IoT interactions, while accommodating the evolving user needs.

### 5.7 Human-AI Interaction

The field of human-AI interaction is gaining more attention, especially with the introduction of ChatGPT and similar AI-powered systems. Recent publications have also begun to explore the impact of AI within the EUD research domain [31]. The integration of human-AI interaction in EUD authoring tools presents a multitude of possibilities. It can serve as a guiding hand for users during the authoring process and provide assistance when they encounter roadblocks. Several papers have already emerged examining the use of ChatGPT for rule creation [9, 34]. Within this section, we propose three compelling ways to incorporate AI components into EUD authoring tools.

Firstly, AI models could enhance the intelligibility features outlined in the preceding section. Personalised feedback and system information tailored to individual users could be generated based on detected usage patterns.

Secondly, an AI component could be integrated into the authoring tool to offer suggestions to users [78]. During our user study we observed that many participants only employed label buttons, without exploring further design options. This behaviour was primarily influenced by the fact that our demo video solely featured label buttons, and did not showcase other UI elements or the possibility to customise font sizes and colours. To solve this problem, participants suggested the inclusion of supplementary tutorial videos. However, an alternative approach could involve the

incorporation of a recommendation and exploration engine directly within the EUD authoring tool. Such an engine could introduce users to novel UI designs and help them refine their own. Considering that end users typically lack design expertise, the incorporation of such a recommendation engine could significantly assist them in enhancing their designs. Further, this concept has been well-received in design tools aimed at professionals, such as Sketchplore [83], DesignScape [65] and GRIDS [24]. While these tools cater to designers, investigating the potential of implementing such a component in an EUD tool is a promising prospect. In addition, these recommendations could extend to guiding users in creating interaction rules. For example, if a user selects the "turn on" function as an action for a "power button", the AI recommendation engine could suggest the use of a "toggle" function instead. Some preliminary work on integrating intelligent support for end users in the IoT rule creation process can be found in [54].

Lastly, going beyond suggestions, some aspects of the authoring process could potentially be automated, as suggested by participants during interviews. For instance, when a user drops a "weather" UI element on the canvas and labels it as "Brussels", the tool could propose to automatically create a rule for displaying the weather forecast in Brussels. This opens up the prospect of exploring further automation opportunities that could not only save users time, but also enhance their overall efficiency when using the tool.

The integration of AI to guide users, enhance intelligibility, and provide suggestions or even generate interaction rules is a subject that demands further research exploration. Such assistance has the potential to elevate UI designs and speed up the application creation process. While these concepts are prevalent in designer-oriented tools, their adaptation for end users merits investigation, presenting an opportunity for AI to empower end-user cross-device and IoT application creation.

### 5.8 Application Generation

Building upon the ideas discussed in the previous section, AI can play a significant role in suggesting and even generating interaction rules for end users. However, we can take this a step further by considering the development of a complete application generation module within the end-user authoring tool. It is worth noting that such a module does not necessarily require AI for its proper functioning. It can be developed using predefined constraints and rules grounded in established design principles, ensuring the creation of effective user interfaces. While some researchers have explored tools for generating UIs and applications [33, 62, 63], these solutions are often not designed for end users or not specifically designed with a focus on IoT or cross-device interaction. Hence, further research is needed to explore the nuances of cross-device and IoT application generation, with a particular focus on empowering end users throughout this creative process. In response to this need, we recently developed an extension module for the eSPACE end-user authoring tool, which allows end users to generate IoT applications by specifying their requirements through a multi-step form wizard [77]. Although this module received positive feedback from an expert review, further evaluations with end users are still necessary to fully assess its effectiveness and usability.

## 6 CONCLUSIONS

In this paper, we provided a comprehensive overview of relevant research in the field of end-user development, with a specific focus on cross-device computing and IoT research. We analysed and categorised the metaphors commonly used in this context, providing a structured overview of approaches in the domain of EUD. Further, we introduced eSPACE, an end-user authoring tool designed to empower end users to create and modify their own cross-device and IoT applications. We also briefly discussed the theoretical concepts behind eSPACE and the design guidelines that

informed its development. Further, we demonstrated the functionality of our tool by introducing its different authoring views and explaining the application creation process.

In order to validate the model-based approach and the applicability of our design guidelines serving as the foundation for eSPACE, we conducted a user study of eSPACE involving 22 participants. The findings from this study revealed that eSPACE is a useful, innovative, easy-to-use tool that is approachable, engaging, efficient, and satisfying to use. Additionally, eSPACE received commendable scores on the post-study system usability questionnaire, asserting its high user satisfaction and usability rating. This empirical validation not only confirmed that the implementation of the design guidelines successfully yielded a user-friendly EUD tool, but also highlighted the potential of using a simplified version of the pipeline metaphor for rule definition, which offers a more visual alternative to conventional textual if-then statements. Lastly, our user study provided valuable insights for refining eSPACE and unveiled future research directions, which have been outlined in the preceding section.

While potential enhancements based on participants' feedback were discussed in Section 4.4.6, along with future research directions in Section 5, numerous opportunities for further enhancements of eSPACE exist. Like many advanced tools in related work, eSPACE could include debugging functionality and detection of rule conflicts. Further, it might be interesting to extend eSPACE with support for collaborative rule creation, akin to platforms like IFTTT, enabling users to share their rules and applications. Although eSPACE currently offers limited multi-user support, this extension would require enhanced intelligibility features to ensure that applications and rules are comprehensible to users who may not have participated in their creation. Many challenges are associated with this future work, underscoring the rich potential for continued research in this field, with eSPACE serving as a foundational tool to facilitate such investigations.

In presenting these findings, we addressed our main research question, illustrating how eSPACE effectively applies suitable abstractions to hide the complexity of the cross-device and IoT application creation process, informed by design guidelines derived from users' mental models. Furthermore, our insights aim to inspire the research community and foster the exploration of new research directions within the domain of end-user development for cross-device computing and IoT solutions.

## ACKNOWLEDGMENTS

This work has been funded by an FWO Postdoc Fellowship (1276721N) of the Research Foundation Flanders.

## REFERENCES

[1] Carmelo Ardito, Paolo Bottoni, Maria Francesca Costabile, Giuseppe Desolda, Maristella Matera, and Matteo Picozzi. 2014. Creation and Use of Service-based Distributed Interactive Workspaces. *Journal of Visual Languages & Computing* 25, 6 (2014), 717–726. https://doi.org/10.1016/j.jvlc.2014.10.018

[2] Barbara Rita Barricelli, Fabio Cassano, Daniela Fogli, and Antonio Piccinno. 2019. End-User Development, End-User Programming and End-User Software Engineering: A Systematic Mapping Study. *The Journal of Systems and Software* 149 (2019), 101–137. https://doi.org/10.1016/J.JSS.2018.11.041

[3] Barbara Rita Barricelli, Daniela Fogli, Letizia Iemmolo, and Angela Locoro. 2022. A Multi-Modal Approach to Creating Routines for Smart Speakers. In *Proceedings of AVI 2022, International Conference on Advanced Visual Interfaces*, Paolo Bottoni and Emanuele Panizzi (Eds.). ACM, Frascati, Rome, Italy, 37:1–37:5. https://doi.org/10.1145/3531073.3531168

[4] Barbara Rita Barricelli and Stefano Valtolina. 2017. A Visual Language and Interactive System for End-User Development of Internet of Things Ecosystems. *Journal of Visual Languages and Computing* 40 (2017), 1–19. https://doi.org/10.1016/j.jvlc.2017.01.004

[5] Victoria Bellotti and Keith Edwards. 2001. Intelligibility and Accountability: Human Considerations in Context Aware Systems. *Human-Computer Interaction* 16 (December 2001). Issue 2. https://doi.org/10.1207/S15327051HCI16234_05

[6] Andrea Bellucci, Andrea Vianello, Yves Florack, Luana Micallef, and Giulio Jacucci. 2019. Augmenting objects at home through programmable sensor tokens: A design journey. *International Journal of Human-Computer Studies* 122 (2019),

211–231. https://doi.org/10.1016/j.ijhcs.2018.09.002

[7] Joey Benedek and Trish Miner. 2002. Measuring Desirability: New Methods for Evaluating Desirability in a Usability Lab Setting. *Proceedings of Usability Professionals Association* 2003, 8-12 (2002), 57.

[8] Frank Bentley, Chris Luvogt, Max Silverman, Rushani Wirasinghe, Brooke White, and Danielle M. Lottridge. 2018. Understanding the Long-Term Use of Smart Speaker Assistants. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 3 (2018), 91:1–91:24. https://doi.org/10.1145/3264901

[9] Giorgio Bimbatti, Daniela Fogli, and Luigi Gargioni. 2023. Can ChatGPT Support End-User Development of Robot Programs?. In *Joint Proceedings of the Workshops, Work in Progress Demos and Doctoral Consortium at the IS-EUD 2023 co-located with the 9th International Symposium on End-User Development (IS-EUD 2023) (CEUR Workshop Proceedings, Vol. 3408)*, Andrea Bellucci, Luigi De Russis, Paloma Díaz, Anders I. Mørch, Daniela Fogli, and Fabio Paternò (Eds.). CEUR-WS.org, Cagliari, Italy. https://ceur-ws.org/Vol-3408/short-s2-03.pdf

[10] Gregor Blichmann, Carsten Radeck, Robert Starke, and Klaus Meißner. 2016. Enabling End-Users to Individually Share Parts of Composite Web Applications. In *Proceedings of WEBIST 2016, 12th International Conference on Web Information Systems and Technologies (Lecture Notes in Business Information Processing, Vol. 292)*, Valérie Monfort, Karl-Heinz Krempels, Tim A. Majchrzak, and Paolo Traverso (Eds.). Springer, Rome, Italy, 164–184. https://doi.org/10.1007/978-3-319-66468-2_9

[11] Julia Brich, Marcel Walch, Michael Rietzler, Michael Weber, and Florian Schaub. 2017. Exploring End User Programming Needs in Home Automation. *ACM Transactions on Computer-Human Interaction* 24, 2 (2017), 11:1–11:35. https://doi.org/10.1145/3057858

[12] Frederik Brudy, Christian Holz, Roman Rädle, Chi-Jui Wu, Steven Houben, Clemens Nylandsted Klokmose, and Nicolai Marquardt. 2019. Cross-Device Taxonomy: Survey, Opportunities and Challenges of Interactions Spanning Across Multiple Devices. In *Proceedings of CHI 2019, Conference on Human Factors in Computing Systems*, Stephen A. Brewster, Geraldine Fitzpatrick, Anna L. Cox, and Vassilis Kostakos (Eds.). ACM, Glasgow, Scotland, UK, 562. https://doi.org/10.1145/3290605.3300792

[13] Daniel Bühler, Fabian Hemmert, Jörn Hurtienne, and Christer Petersen. 2022. Designing Universal and Intuitive Pictograms (UIPP) - A Detailed Process for More Suitable Visual Representations. *International Journal of Human-Computer Studies* 163 (2022), 102816. https://doi.org/10.1016/J.IJHCS.2022.102816

[14] Danilo Caivano, Daniela Fogli, Rosa Lanzilotti, Antonio Piccinno, and Fabio Cassano. 2018. Supporting End Users To Control Their Smart Home: Design Implications from a Literature Review and an Empirical Investigation. *Journal of Systems and Software* 144 (2018), 295–313. https://doi.org/10.1016/j.jss.2018.06.035

[15] Gaëlle Calvary, Joëlle Coutaz, L. Bouillon, M. Florins, O. Limbourg, L. Marucci, F. Paternò, C. Santoro, N. Souchon, D. Thevenin, and J. Vanderdonckt. 2002. The CAMELEON Reference Framework. CAMELEON Project Deliverable 1.1.

[16] Xiang 'Anthony' Chen and Yang Li. 2017. Improv: An Input Framework for Improvising Cross-Device Interaction by Demonstration. *ACM Transactions on Computer-Human Interaction* 24, 2 (2017), 15:1–15:21. https://doi.org/10.1145/3057862

[17] Sven Coppers, Davy Vanacken, and Kris Luyten. 2020. FORTNIoT: Intelligible Predictions to Improve User Understanding of Smart Home Behavior. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4 (2020), 124:1–124:24. Issue 4. https://doi.org/10.1145/3432225

[18] Sven Coppers, Davy Vanacken, and Kris Luyten. 2022. FortClash: Predicting and Mediating Unintended Behavior in Home Automation. *Proceedings of the ACM on Human-Computer Interaction* 6, EICS (2022), 154:1–154:20. https://doi.org/10.1145/3532204

[19] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. [n. d.]. Empowering End Users in Debugging Trigger-Action Rules. In *Proceedings of CHI 2019, Conference on Human Factors in Computing Systems*, Stephen A. Brewster, Geraldine Fitzpatrick, Anna L. Cox, and Vassilis Kostakos (Eds.).

[20] Joëlle Coutaz and James L. Crowley. 2016. A First-Person Experience with End-User Development for Smart Homes. *IEEE Pervasive Computing* 15, 2 (2016), 26–39. https://doi.org/10.1109/MPRV.2016.24

[21] Menglong Cui, Mingsong Lv, Qingqiang He, Caiqi Zhang, Chuancai Gu, Tao Yang, and Nan Guan. 2021. PRUID: Practical User Interface Distribution for Multi-surface Computing. In *Proceedings of DAC 2021, Design Automation Conference*. IEEE, San Francisco, CA, USA, 679–684. https://doi.org/10.1109/DAC18074.2021.9586162

[22] José Danado and Fabio Paternò. 2014. Puzzle: A Mobile Application Development Environment using a Jigsaw Metaphor. *Journal of Visual Languages & Computing* 25, 4 (2014), 297–315. https://doi.org/10.1016/j.jvlc.2014.03.005

[23] Oleg Davidyuk, Iván Sánchez Milara, Ekaterina Gilman, and Jukka Riekki. 2015. An Overview of Interactive Application Composition Approaches. *Open Computer Science* 5, 1 (2015). https://doi.org/10.1515/comp-2015-0007

[24] Niraj Ramesh Dayama, Kashyap Todi, Taru Saarelainen, and Antti Oulasvirta. 2020. GRIDS: Interactive Layout Design with Integer Programming. In *Proceedings of CHI 2020, Conference on Human Factors in Computing Systems* (Honolulu, HI, USA). ACM, 1–13. https://doi.org/10.1145/3313831.3376553

[25] Rodrigo de A. Maues and Simone Diniz Junqueira Barbosa. 2013. Keep Doing What I Just Did: Automating Smartphones by Demonstration. In *Proceedings of MobileHCI 2013, 15th International Conference on Human-Computer Interaction with Mobile Devices and Services*, Michael Rohs, Albrecht Schmidt, Daniel Ashbrook, and Enrico Rukzio (Eds.). ACM, Munich, Germany, 295–303. https://doi.org/10.1145/2493190.2493216

[26] Giuseppe Desolda, Carmelo Ardito, and Maristella Matera. 2015. EFESTO: A Platform for the End-User Development of Interactive Workspaces for Data Exploration. In *Rapid Mashup Development Tools - First International Rapid Mashup Challenge in ICWE 2015 (Communications in Computer and Information Science, Vol. 591)*, Florian Daniel and Cesare Pautasso (Eds.). Springer, Rotterdam, The Netherlands, 63–81. https://doi.org/10.1007/978-3-319-28727-0_5

[27] Giuseppe Desolda, Carmelo Ardito, and Maristella Matera. 2017. Empowering End Users to Customize their Smart Environments: Model, Composition Paradigms, and Domain-Specific Tools. *ACM Transactions on Computer-Human Interaction* 24, 2 (2017), 12:1–12:52. https://doi.org/10.1145/3057859

[28] Anind K. Dey, Raffay Hamid, Chris Beckmann, Ian Li, and Daniel Hsu. 2004. a CAPpella: Programming by Demonstration of Context-Aware Applications. In *Proceedings of CHI 2004, Conference on Human Factors in Computing Systems*. Vienna, Austria, 33–40. https://doi.org/10.1145/985692.985697

[29] Anind K. Dey, Timothy Sohn, Sara Streng, and Justin Kodama. 2006. iCAP: Interactive Prototyping of Context-Aware Applications. In *Proceedings of PERVASIVE 2006, Conference on Pervasive Computing*. Dublin, Ireland, 254–271. https://doi.org/10.1007/11748625_16

[30] Daniela Fogli, Matteo Peroni, and Claudia Stefini. 2017. ImAtHome: Making Trigger-Action Programming Easy and Fun. *Journal of Visual Languages and Computing* 42 (2017), 60–75. https://doi.org/10.1016/j.jvlc.2017.08.003

[31] Daniela Fogli and Daniel Tetteroo. 2022. End-User Development for Democratising Artificial Intelligence. *Behaviour & Information Technology* 41, 9 (2022), 1809–1810. https://doi.org/10.1080/0144929x.2022.2100974

[32] Debbie Folaron. 2006. A Discipline Coming of Age in The Digital Age. *Perspectives on Localization* (2006), 195–219.

[33] Krzysztof Z. Gajos, Daniel S. Weld, and Jacob O. Wobbrock. 2010. Automatically Generating Personalized User Interfaces with SUPPLE. *Artificial Intelligence* 174, 12 (2010), 910–950. https://doi.org/10.1016/j.artint.2010.05.005

[34] Simone Gallo, Alessio Malizia, and Fabio Paternò. 2023. Towards a Chatbot for Creating Trigger-Action Rules based on ChatGPT and Rasa. In *Joint Proceedings of the Workshops, Work in Progress Demos and Doctoral Consortium at the IS-EUD 2023 co-located with the 9th International Symposium on End-User Development (IS-EUD 2023) (CEUR Workshop Proceedings, Vol. 3408)*, Andrea Bellucci, Luigi De Russis, Paloma Díaz, Anders I. Mørch, Daniela Fogli, and Fabio Paternò (Eds.). CEUR-WS.org, Cagliari, Italy. https://ceur-ws.org/Vol-3408/short-s4-01.pdf

[35] Cristian González García, B. Cristina Pelayo García-Bustelo, Jordán Pascual Espada, and Guillermo Cueva-Fernandez. 2014. Midgar: Generation of Heterogeneous Objects Interconnecting Applications. A Domain Specific Language Proposal for Internet of Things Scenarios. *Computer Networks* 64 (2014), 143–158. https://doi.org/10.1016/j.comnet.2014.02.010

[36] Giuseppe Ghiani, Marco Manca, and Fabio Paternò. 2015. Authoring Context-dependent Cross-device User Interfaces based on Trigger/Action Rules. In *Proceedings of MUM 2015, Conference on Mobile and Ubiquitous Multimedia*. Linz, Austria, 313–322. https://doi.org/10.1145/2836041.2836073

[37] Giuseppe Ghiani, Marco Manca, Fabio Paternò, and Carmen Santoro. 2017. Personalization of Context-Dependent Applications Through Trigger-Action Rules. *ACM Transactions on Computer-Human Interaction* 24, 2 (2017), 14:1–14:33. https://doi.org/10.1145/3057861

[38] Giuseppe Ghiani, Fabio Paternò, Lucio Davide Spano, and Giuliano Pintori. 2016. An Environment for End-User Development of Web Mashups. *International Journal of Human-Computer Studies* 87 (2016), 38–64. https://doi.org/10.1016/j.ijhcs.2015.10.008

[39] Rüdiger Heimgärtner, Andreas Holzinger, and Ray Adams. 2008. From Cultural to Individual Adaptive End-User Interfaces: Helping People with Special Needs. In *Proceedings of ICCHP 2008, 11th International Conference on Computers Helping People with Special Needs (Lecture Notes in Computer Science, Vol. 5105)*, Klaus Miesenberger, Joachim Klaus, Wolfgang L. Zagler, and Arthur I. Karshmer (Eds.). Springer, Linz, Austria, 82–89. https://doi.org/10.1007/978-3-540-70540-6_11

[40] Sehyeon Heo, Sungpil Woo, Janggwan Im, and Daeyoung Kim. 2015. IoT-MAP: IoT Mashup Application Platform for the Flexible IoT Ecosystem. In *Proceedings on IOT 2015, Conference on the Internet of Things*. Seoul, South Korea, 163–170. https://doi.org/10.1109/IOT.2015.7356561

[41] Jan Humble, Andy Crabtree, Terry Hemmings, Karl-Petter Åkesson, Boriana Koleva, Tom Rodden, and Pär Hansson. 2003. "Playing with the Bits" User-Configuration of Ubiquitous Domestic Environments. In *Proceedings of UbiComp 2003, 5th International Conference on Ubiquitous Computing*. Seattle, Washington, USA, 256–263. https://doi.org/10.1007/978-3-540-39653-6_20

[42] Timo Jakobi, Gunnar Stevens, Nico Castelli, Corinna Ogonowski, Florian Schaub, Nils Vindice, Dave Randall, Peter Tolmie, and Volker Wulf. 2018. Evolving Needs in IoT Control and Accountability: A Longitudinal Study on Smart Home Intelligibility. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2 (12 2018),

1–28. Issue 4. https://doi.org/10.1145/3287049

[43] Yuna Jeong, Hyuntae Joo, Gyeonghwan Hong, Dongkun Shin, and Sungkil Lee. 2015. AVIoT: Web-based Interactive Authoring and Visualization of Indoor Internet of Things. *IEEE Transactions on Consumer Electronics* 61, 3 (2015), 295–301. https://doi.org/10.1109/TCE.2015.7298088

[44] István Koren. 2020. *DevOpsUse: Community-driven Continuous Innovation of Web Information Infrastructures*. Ph. D. Dissertation. RWTH Aachen University, Germany. https://publications.rwth-aachen.de/record/793376

[45] Dejan Kovachev, Dominik Renzel, Petru Nicolaescu, and Ralf Klamma. 2013. DireWolf - Distributing and Migrating User Interfaces for Widget-Based Web Applications. In *Proceedings of ICWE 2013, Conference on Web Engineering*. Aalborg, Denmark, 99–113. https://doi.org/10.1007/978-3-642-39200-9_10

[46] Michael Krug, Fabian Wiedemann, and Martin Gaedke. 2014. SmartComposition: A Component-Based Approach for Creating Multi-screen Mashups. In *Proceedings of ICWE 2014, Conference on Web Engineering*. Toulouse, France, 236–253. https://doi.org/10.1007/978-3-319-08245-5_14

[47] Thomas Kubitza and Albrecht Schmidt. 2017. meSchup: A Platform for Programming Interconnected Smart Things. *IEEE Computer* 50, 11 (2017), 38–49. https://doi.org/10.1109/MC.2017.4041350

[48] Sunjae Lee, Hoyoung Kim, Sijung Kim, Sangwook Lee, Hyosu Kim, Jean Young Song, Steven Y. Ko, Sangeun Oh, and Insik Shin. 2022. A-Mash: Providing Single-App Illusion for Multi-App Use Through User-Centric UI Mashup. In *Proceedings of MobiCom 2022, Annual International Conference on Mobile Computing and Networking*. ACM, Sydney, NSW, Australia, 690–702. https://doi.org/10.1145/3495243.3560522

[49] Toby Jia-Jun Li, Yuanchun Li, Fanglin Chen, and Brad A. Myers. 2017. Programming IoT Devices by Demonstration Using Mobile Apps. In *Proceedings of IS-EUD, Conference on End-User Development*. Eindhoven, The Netherlands, 3–17. https://doi.org/10.1007/978-3-319-58735-6_1

[50] Henry Lieberman, Fabio Paternò, and Volker Wulf (Eds.). 2006. *End User Development: An Emerging Paradigm*. Springer. https://doi.org/10.1007/1-4020-5386-X

[51] Brian Y. Lim and Anind K. Dey. 2013. Evaluating Intelligibility Usage and Usefulness in a Context-Aware Application. In *Proceedings of HCI International 2013, International Conference on Human-Computer Interaction*. Las Vegas, USA. https://doi.org/10.1007/978-3-642-39342-6_11

[52] Gabriella Lucci and Fabio Paternò. 2014. Understanding End-User Development of Context-Dependent Applications in Smartphones. In *Proceedings of HCSE 2014, 5th IFIP WG 13.2 International Conference Human-Centered Software Engineering*:. Springer, Springer Berlin Heidelberg, Paderborn, Germany, 182–198. https://doi.org/10.1007/978-3-662-44811-3_11

[53] Gabriella Lucci and Fabio Paternò. 2015. Analysing How Users Prefer to Model Contextual Event-Action Behaviours in Their Smartphones. In *Proceedings of IS-EUD 2015, 5th International Symposium on End-User Development (Lecture Notes in Computer Science, Vol. 9083)*, Paloma Díaz, Volkmar Pipek, Carmelo Ardito, Carlos Jensen, Ignacio Aedo, and Alexander Boden (Eds.). Springer, Madrid, Spain, 186–191. https://doi.org/10.1007/978-3-319-18425-8_14

[54] Andrea Mattioli and Fabio Paternò. 2021. Recommendations for Creating Trigger-Action Rules in a Block-based Environment. *Behaviour & Information Technology* 40, 10 (2021), 1024–1034. https://doi.org/10.1080/0144929X.2021.1900396

[55] Sarah Mennicken, David Kim, and Elaine May Huang. 2016. Integrating the Smart Home into the Digital Calendar. In *Proceedings of CHI 2016, Conference on Human Factors in Computing Systems*. ACM, San Jose, CA, USA, 5958–5969. https://doi.org/10.1145/2858036.2858168

[56] Jan Meskens, Kris Luyten, and Karin Coninx. 2010. Jelly: a Multi-Device Design Environment for Managing Consistency Across Devices. In *Proceedings of AVI 2010, International Conference on Advanced Visual Interfaces*. Roma, Italy, 289–296. https://doi.org/10.1145/1842993.1843044

[57] Mahdi H. Miraz, Maaruf Ali, and Peter S. Excell. 2021. Adaptive User Interfaces and Universal Usability Through Plasticity of User Interface Design. *Computer Science Review* 40 (2021), 100363. https://doi.org/10.1016/J.COSREV.2021.100363

[58] Mahdi H. Miraz, Peter S. Excell, and Maaruf Ali. 2016. User Interface (UI) Design Issues for Multilingual Users: A Case Study. *Universal Access in the Information Society* 15, 3 (2016), 431–444. https://doi.org/10.1007/S10209-014-0397-5

[59] Michael Nebeling. 2017. XDBrowser 2.0: Semi-Automatic Generation of Cross-Device Interfaces. In *Proceedings of CHI 2017, Conference on Human Factors in Computing Systems*. Denver, Colorado, USA, 4574–4584. https://doi.org/10.1145/3025453.3025547

[60] Michael Nebeling and Anind K. Dey. 2016. XDBrowser: User-Defined Cross-Device Web Page Designs. In *Proceedings of CHI 2016, Conference on Human Factors in Computing Systems*. San Jose, California, USA, 5494–5505. https://doi.org/10.1145/2858036.2858048

[61] Michael Nebeling, Theano Mintsi, Maria Husmann, and Moira C. Norrie. 2014. Interactive Development of Cross-Device User Interfaces. In *Proceedings of CHI 2014, Conference on Human Factors in Computing Systems*, Matt Jones, Philippe A. Palanque, Albrecht Schmidt, and Tovi Grossman (Eds.). ACM, Toronto, ON, Canada, 2793–2802. https:

//doi.org/10.1145/2556288.2556980

[62] Thiago Nepomuceno, Tiago Carneiro, Paulo Henrique M. Maia, Muhammad Adnan, Thalyson Nepomuceno, and Alexander Martin. 2020. AutoIoT: a Framework Based on User-driven MDE for Generating IoT Applications. In *Proceedings of SAC 2020, 35th ACM/SIGAPP Symposium on Applied Computing*, Chih-Cheng Hung, Tomás Cerný, Dongwan Shin, and Alessio Bechini (Eds.). ACM, online event, 719–728. https://doi.org/10.1145/3341105.3373873

[63] Jeffrey Nichols, Brandon Rothrock, Duen Horng Chau, and Brad A. Myers. 2006. Huddle: Automatically Generating Interfaces for Systems of Multiple Connected Appliances. In *Proceedings of UIST 2006, 19th Annual ACM Symposium on User Interface Software and Technology*, Pierre Wellner and Ken Hinckley (Eds.). ACM, Montreux, Switzerland, 279–288. https://doi.org/10.1145/1166253.1166298

[64] Donald A. Norman and Jakob Nielsen. 2010. Gestural Interfaces: A Step Backward in Usability. *Interactions* 17, 5 (2010), 46–49. https://doi.org/10.1145/1836216.1836228

[65] Peter O'Donovan, Aseem Agarwala, and Aaron Hertzmann. 2015. DesignScape: Design with Interactive Layout Suggestions. In *Proceedings of CHI 2015, 33rd annual ACM Conference on Human Factors in Computing Systems*. ACM, Seoul, Republic of Korea, 1221–1224. https://doi.org/10.1145/2702123.2702149

[66] Sangeun Oh, Ahyeon Kim, Sunjae Lee, Kilho Lee, Dae R. Jeong, Insik Shin, and Steven Y. Ko. 2019. Fluid: Flexible User Interface Distribution for Ubiquitous Multi-Device Interaction. *GetMobile: Mobile Computing and Communications* 23, 4 (2019), 25–29. https://doi.org/10.1145/3400713.3400719

[67] Fabio Paternò and Carmen Santoro. 2017. A Design Space for End User Development in the Time of the Internet of Things. In *New Perspectives in End-User Development*. 43–59. https://doi.org/10.1007/978-3-319-60291-2_3

[68] Fabio Paternò and Carmen Santoro. 2019. End-User Development for Personalizing Applications, Things, and Robots. *International Journal of Human-Computer Studies* 131 (2019), 120–130. https://doi.org/10.1016/j.ijhcs.2019.06.002

[69] K. Peffers, T. Tuunanen, M. Rothenberger, and S. Chatterjee. 2007. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems* 24 (January 2007), 45–77. https://doi.org/doi.org/10.2753/MIS0742-1222240302

[70] Trevor Pering, Kent Lyons, Roy Want, Mary Murphy-Hoye, Mark Baloga, Paul Noll, Joe Branc, and Nicolas De Benoist. 2010. What Do You Bring To the Table?: Investigations of a Collaborative Workspace. In *Proceedings of UbiComp 2010, 12th International Conference on Ubiquitous Computing*. Copenhagen, Denmark, 183–192. https://doi.org/10.1145/1864349.1864389

[71] Trevor Pering, Roy Want, Barbara Rosario, Shivani Sud, and Kent Lyons. 2009. Enabling Pervasive Collaboration with Platform Composition. In *Proceedings of PerCom 2009, 7th International Conference on Pervasive Computing*. Nara, Japan, 184–201. https://doi.org/10.1007/978-3-642-01516-8_14

[72] Madeleine Ridderstråle and Louise Östling Sigfridsdotter. 2023. *Exploring Cultural Differences in UI Usage: A Comparative Study of Swedish and Japanese Culture*. Ph. D. Dissertation. Chalmers University of Technology, University of Gothenburg.

[73] Tom Rodden, Andy Crabtree, Terry Hemmings, Boriana Koleva, Jan Humble, Karl-Petter Åkesson, and Pär Hansson. 2004. Configuring the Ubiquitous Home. In *Proceedings of COOP 2004, 6th International Conference on the Design of Cooperative Systems*. Hyères Les Palmiers, France, 227–242. https://hdl.handle.net/20.500.12015/3031

[74] Alberto Monge Roffarello and Luigi De Russis. 2023. Defining Trigger-Action Rules via Voice: A Novel Approach for End-User Development in the IoT. In *Proceedings of IS-EUD 2023, 9th International Symposium on End-User Development (Lecture Notes in Computer Science, Vol. 13917)*. Springer, Cagliari, Italy, 65–83. https://doi.org/10.1007/978-3-031-34433-6_5

[75] Audrey Sanctorum. 2020. *Conceptual Foundations for End-User Authoring of Cross-Device and Internet of Things Applications*. Ph. D. Dissertation. Vrije Universiteit Brussel. https://wise.vub.ac.be/sites/default/files/theses/PhDThesisAudreySanctorum.pdf.

[76] Audrey Sanctorum, Suzanne Kieffer, and Beat Signer. 2020. User-driven Design Guidelines for the Authoring of Cross-Device and Internet of Things Applications. In *Proceedings of NordiCHI 2020, Nordic Conference on Human-Computer Interaction*, David Lamas, Hegle Sarapuu, Marta Lárusdóttir, Jan Stage, and Carmelo Ardito (Eds.). ACM, Tallinn, Estonia, 84:1–84:12. https://doi.org/10.1145/3419249.3420136

[77] Audrey Sanctorum and Brenda Ordoñez Lujan. 2024. Towards End-User-Driven Generation of IoT Applications. In *Companion Proceedings of the 16th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS Companion 2024, Cagliari, Italy, June 24-28, 2024*, Michael Nebeling, Lucio Davide Spano, and José Creissac Campos (Eds.). ACM, 66–73. https://doi.org/10.1145/3660515.3661332

[78] Audrey Sanctorum, Luka Rukonic, and Beat Signer. 2021. Design Requirements for Recommendations in End-User User Interface Design. In *Proceedings od IS-EUD 2021, 8th International Symposium on End-User Development*, Daniela Fogli, Daniel Tetteroo, Barbara Rita Barricelli, Simone Borsci, Panos Markopoulos, and George A. Papadopoulos (Eds.). Virtual Event, 204–212. https://doi.org/10.1007/978-3-030-79840-6_14

[79] Audrey Sanctorum and Beat Signer. 2019. A Unifying Reference Framework and Model for Adaptive Distributed Hybrid User Interfaces. In *Proceedings of RCIS 2019, International Conference on Research Challenges in Information Science*, Manuel Kolp, Jean Vanderdonckt, Monique Snoeck, and Yves Wautelet (Eds.). IEEE, Brussels, Belgium, 1–6. https://doi.org/10.1109/RCIS.2019.8877048

[80] Audrey Sanctorum and Beat Signer. 2019. Towards End-User Development of Distributed User Interfaces. *Universal Access in the Information Society* 18, 4 (2019), 785–799. https://doi.org/10.1007/s10209-017-0601-5

[81] Jeff Sauro and James R. Lewis. 2016. *Quantifying the User Experience: Practical Statistics for User Research* (2nd ed.). Morgan Kaufmann.

[82] Beat Signer and Moira C. Norrie. 2007. As We May Link: A General Metamodel for Hypermedia Systems. In *Proceedings of ER 2007, International Conference on Conceptual Modeling*. Auckland, New Zealand. https://doi.org/10.1007/978-3-540-75563-0_25

[83] Kashyap Todi, Daryl Weir, and Antti Oulasvirta. 2016. Sketchplore: Sketch and Explore with a Layout Optimiser. In *Proceedings DIS 2016, Conference on Designing Interactive Systems* (Brisbane, Australia). ACM, 543–555. https://doi.org/10.1145/2901790.2901817

[84] Sandra Trullemans, Lars Van Holsbeeke, and Beat Signer. 2017. The Context Modelling Toolkit: A Unified Multi-layered Context Modelling Approach. *Proceedings of the ACM on Human-Computer Interaction* 1, EICS (2017), 8:1–8:16. https://doi.org/10.1145/3095810

[85] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman. 2014. Practical Trigger-Action Programming in the Smart Home. In *Proceedings of CHI 2014, Conference on Human Factors in Computing Systems*. Toronto, ON, Canada, 803–812. https://doi.org/10.1145/2556288.2557420

[86] Jo Vermeulen, Kris Luyten, and Karin Coninx. 2013. Intelligibility Required: How to Make Us Look Smart Again. In *Proceedings of RoCHI 2013, Romanian Human-Computer Interaction Conference*. Cluj-Napoca, Romania.

[87] Tianyi Wang, Xun Qian, Fengming He, Xiyun Hu, Yuanzhi Cao, and Karthik Ramani. 2021. GesturAR: An Authoring System for Creating Freehand Interactive Augmented Reality Applications. In *Proceedings of UIST 2021, 34th Annual ACM Symposium on User Interface Software and Technology*, Jeffrey Nichols, Ranjitha Kumar, and Michael Nebeling (Eds.). ACM, Virtual Event, 552–567. https://doi.org/10.1145/3472749.3474769

[88] Massimo Zancanaro, Giuseppe Gallitto, Dina Yem, and Barbara Treccani. 2022. Improving Mental Models in IoT End-User Development. *Human-centric Computing and Information Sciences* 12 (2022), 48. https://doi.org/10.22967/HCIS.2022.12.048

# A REQUIREMENTS FOR THE END-USER AUTHORING OF CROSS-DEVICE AND IOT USER INTERFACES

| | |
|---|---|
| **Requirement 1 (R1)** | Provide an **overview** of the smart technologies, environments and applications |
| **Requirement 2 (R2)** | **Interaction** support |
| *Requirement 2.1 (R2.1)* | Support for interaction across multiple smart technologies |
| *Requirement 2.2 (R2.2)* | Support for creation, customisation and distribution of cross-device and IoT user interfaces |
| *Requirement 2.3 (R2.3)* | Offer fine granularity user interface distribution |
| **Requirement 3 (R3)** | **Shareability** |
| *Requirement 3.1 (R3.1)* | Support for sharing and integration of apps in a central smart apps repository |
| *Requirement 3.2 (R3.2)* | Enable sharing of applications, user interfaces or parts of a user interface with specific users |
| **Requirement 4 (R4)** | **Extensibility** |
| *Requirement 4.1 (R4.1)* | Offer extensibility at the level of communication protocols, devices and user interfaces |
| *Requirement 4.2 (R4.2)* | Enable the integration of third-party applications |
| *Requirement 4.3 (R4.3)* | Offer extensibility of adaptive behaviour and distribution configurations |
| **Requirement 5 (R5)** | **Reuseability** |
| *Requirement 5.1 (R5.1)* | Support for reuse and combination of existing user interfaces |
| *Requirement 5.2 (R5.2)* | Support for reuse and combination of existing functionality |
| **Requirement 6 (R6)** | **Portability** |
| *Requirement 6.1 (R6.1)* | Offer platform independence |
| *Requirement 6.2 (R6.2)* | Support for context awareness |
| **Requirement 7 (R7)** | Support for **end-user development** |

Table 6. List of requirements for building XDI and IoT end-user authoring tools, derived from related work as well as from a use case scenario; more information can be found in [75]

## B ESPACE REFERENCE FRAMEWORK

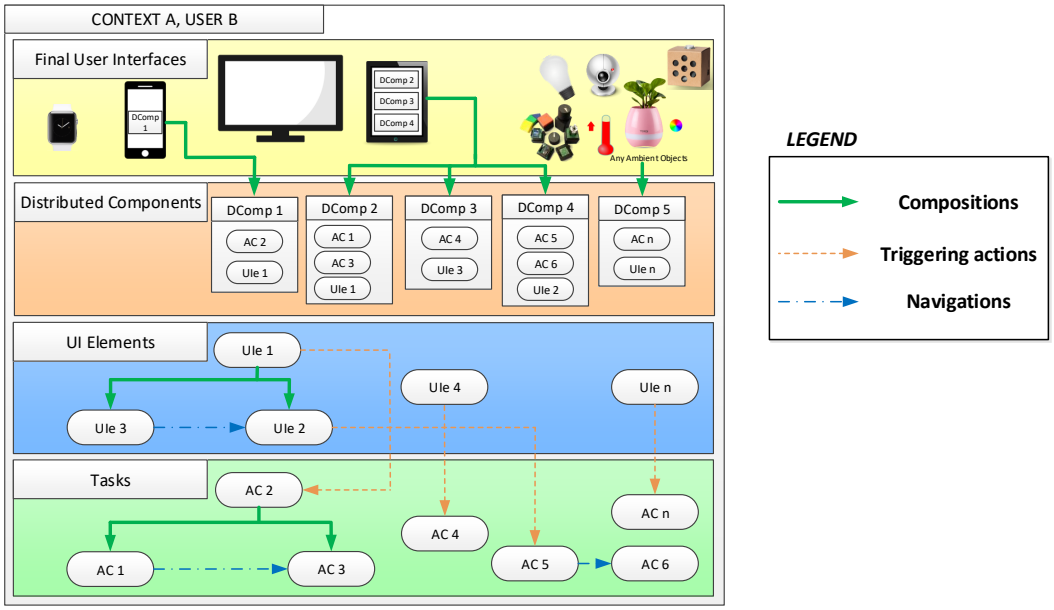



Fig. 15. The eSPACE reference framework breaks down the UI development process into distinct layers of abstraction in order to facilitate the development of cross-device and IoT user interfaces. Inspired by the CAMELEON Reference Framework (CRF) [15], it simplifies the UI design process by using a compositional approach without requiring transformations between abstraction layers. The framework comprises four layers: the *Tasks* layer, where Active Components (ACs) [82] represent functionality; the *User Interface Elements* layer, containing reusable UI components; the *Distributed Components* layer, combining tasks and UI elements into distributable units; and the *Final User Interface* layer, tailoring the interface to specific devices and contexts. This framework promotes reusability, flexibility, and extensibility, allowing users to link and adapt components to create customisable cross-device and IoT applications. A comprehensive description along with example use cases is available in [79].

## C ESPACE CONCEPTUAL MODEL

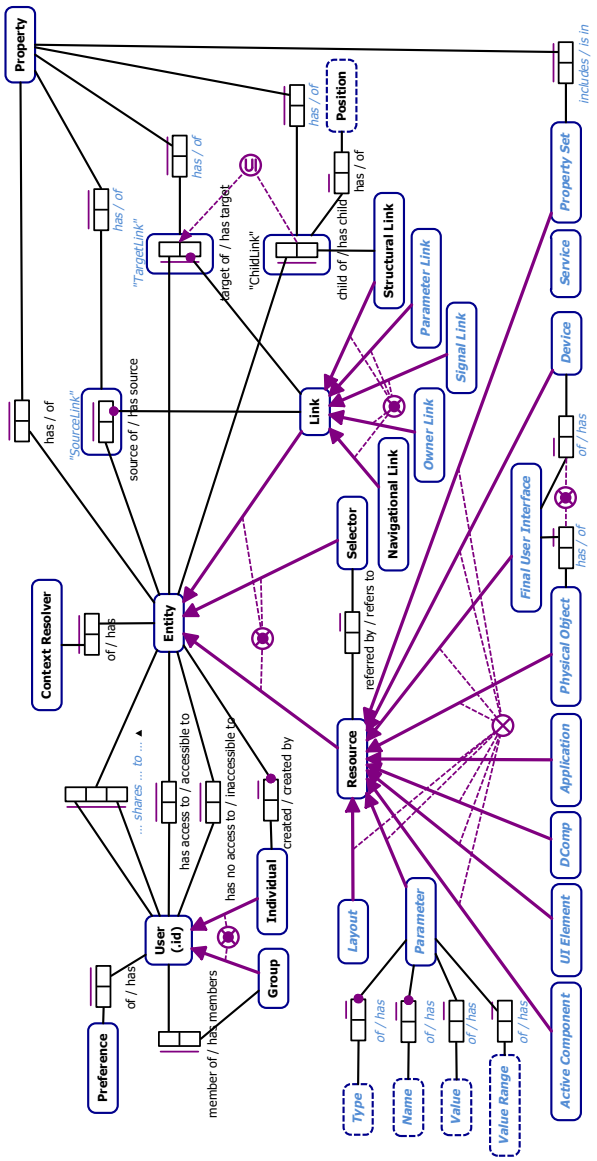



Fig. 16. The eSPACE conceptual model describes the elements available within the eSPACE reference framework and the relationships between them. Rooted in the Resource-Selector-Link (RSL) hypermedia metamodel [82], it extends the model with domain-specific adaptations. The model leverages *Resources* (e.g. Applications, UI Elements, Active Components), *Selectors* (targeting parts of resources), and *Links* (e.g. structural, navigational, and parameter links) to define relationships and interactions. By supporting reuse, context-awareness, and shareability, the model facilitates dynamic interactions such as triggering events or adapting UIs based on user preferences, time, or environment. Additionally, it incorporates extensions like parametrised inputs/outputs and ownership relationships, enhancing flexibility and scalability for cross-device and IoT applications. A detailed description of our model can be found in [79].