Towards End-User-Driven Generation of IoT Applications

AUDREY SANCTORUM* and BRENDA ORDOÑEZ LUJAN*, Vrije Universiteit Brussel, Belgium

In the rapidly evolving landscape of smart home devices and IoT applications, various end-user development solutions have emerged, empowering individuals without technical knowledge to manage and control all their devices. While many of these solutions revolve around creating custom functionality using trigger-action rules, some also offer the capability to design personalised user interfaces (UIs) for controlling their smart devices. This allows end users to design IoT applications tailored to their specific individual needs and preferences. However, the resulting UI designs often fall short due to the typical absence of UI design skills among end users. In order to address this challenge and elevate the quality of UI design for end users, we introduce an automatic IoT Application Generation Module. This module has been implemented as an extension to an existing end-user authoring tool, and allows end users to generate user interfaces based on their requirements and preferences. We conducted preliminary evaluations through expert assessment sessions, which yielded valuable insights for future improvements. Our findings indicate that the module facilitates the rapid creation of simple IoT applications, yet further refinements are essential to optimise the final design outputs.

CCS Concepts: • **Human-centered computing → User interface design**; **Interactive systems and tools**; **User interface toolkits**; *Graphical user interfaces*; Ubiquitous computing.

Additional Key Words and Phrases: User interface generation, user interface design, IoT applications generation, end-user development, IoT user interfaces, end-user IoT application generation, automatic IoT application generation

ACM Reference Format:
Audrey Sanctorum and Brenda Ordoñez Lujan. 2024. Towards End-User-Driven Generation of IoT Applications. In *Companion of the 16th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS Companion '24), June 24–28, 2024, Cagliari, Italy.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3660515.3661332

1 INTRODUCTION

Designing user interfaces (UIs) has traditionally been a complex and time-consuming task, often requiring design skills and programming knowledge. With the evolution of technology and the ever-changing needs of users, the task of UI design has become even more challenging. One approach to cope with this challenge is through the field of End-User Development (EUD), which enables individuals without programming expertise to create, adapt, or expand software applications that suit their specific needs [9].

In this paper, we focus our attention on EUD within the Internet of Things (IoT) domain. Over the years, numerous EUD tools have surfaced in this domain [2], enabling users to exercise greater control over their smart home devices. These tools commonly offer the ability to create trigger-action rules for creating custom behaviours. Some advanced tools take it a step further, also allowing users to construct custom UIs for controlling their IoT devices. However, these tools often pose challenges for end users, especially when designing UIs is involved. The UI design process can be time-consuming and intricate, making it less accessible for users with limited design experience.

*Both authors contributed equally to this research.

Manuscript submitted to ACM

In response to these challenges, this paper presents a novel solution that contributes to the field of EUD. Our approach aims to streamline the UI design process, making it more efficient for end users. Simultaneously, it aspires to enhance the quality of the end-user UI designs by automating most of the design process. Our proposed Automatic IoT Application Generation Module takes on this role.

The remainder of this paper is structured as follows. First, we provide a succinct overview of related work in the domain. Next, we introduce our IoT Application Generation Module. Following this, we present the results of an expert evaluation of the IoT Application Generation Module. Finally, we conclude with a summary of our findings and outline potential future work.

2 RELATED WORK

User interface software tools have evolved over the years to assist developers in designing and implementing user interfaces [12]. These tools exhibit a wide range of capabilities, spanning from those demanding users to design and implement the entire UI to solutions automating various aspects of the UI development process.

This diversity in capabilities has naturally led to distinctions in target users. Traditional code editors, addressing professional developers, offer advanced functionalities and require strong programming foundations. While tools like Sketch, Figma, and Adobe XD [29] tailored for design professionals, provide intuitive environments for creating wireframes and mockups, but still demand expertise in fundamentals of user interface design (UID). Although these three tools offer plugins to automate repetitive UID tasks, research has been done to further simplify the design process by introducing tools capable of automatically generating part or the entire UI. A popular approach for automating the UI design involves the use of model-based generation tools that construct a UI based on high-level models or specifications [11]. Several tools have adopted this approach, including Humanoid [25], Mecano [20], MASTERMIND [26], Teallach [6], SUPPLE [3, 5], Huddle [15], MARIA [17], DB-USE [28], UsiMAD [13] and MODUS [10]. A comprehensive overview, can be found in [21]. Typically designed for designers, these tools necessitate users to possess modelling skills.

One of the early model-based UI development processes that actively involved end users in the design cycle is the Model-based Interface Designer (Mobi-D) [19] environment, the successor of Mecano [20]. Mobi-D begins by capturing user tasks through an interactive tool, where end users input textual task descriptions. These descriptions are then translated into structured user-task information, aiding interface developers in constructing user-task and domain models. Based on these models, Mobi-D suggests presentation and interaction techniques to the developers for constructing the user interface, which is then ultimately tested by end users. Another model-based tool that integrates end-user input is ARNAULD [4], an extension of SUPPLE designed to generate UIs based on end users' preferences. It achieves this by presenting different user interface fragments to users and asking them to select their preferred option [5].

However, these tools tend to limit the end users' freedom compared to most EUD tools, where end users actively engage as non-professional developers to create, modify or extend a software artefact [9]. Since the rise of the Web 2.0. it became evident that end users sought greater involvement in user-generated content, expressing a desire to customise the behaviour and functionality of their applications [17]. Research in EUD has seen extensive exploration, ranging from web applications and mashups to smart objects and intelligent environments. Recent systematic reviews by Kuhail et al. [8] and Barricelli et al. [1] highlight the interest in EUD approaches. As outlined in these reviews, the emergence of the Internet of Things has led to numerous proposed EUD solutions. Nevertheless, the majority of these tools primarily focus on empowering end users to control their IoT device through the creation of trigger-action rules

or by defining routines, with only a handful offering the capability to create custom user interfaces for managing these devices [23].

Johnsson and Magnusson [7] have introduced one of the few solutions that provide end users with a user interface design environment for creating IoT applications. The authors' objective is to speed up the GUI design process by taking a novel approach called the inverted GUI development method. In contrast to the conventional procedure where users drag and drop UI components onto a canvas and subsequently add functionality, this approach allows users to drag and drop the desired functionality component and the tool then selects the suitable UI component to visually represent this functionality. Another notable end-user authoring tool for developing UIs for IoT applications is eSPACE [22]. The tool empowers end users to create GUIs by dragging and dropping UI components to an authoring canvas. To add functionality to these components, users create trigger-action rules, such as "if button is clicked, then turn on living room light". Additionally, there are a couple of commercial solutions providing GUI builders for IoT control, including the NEMA GUI Builder[1] and Embedded Wizard[2].

While these tools grant considerable freedom to end users and greatly facilitate the UID process, constructing IoT UIs and applications remains a time-consuming task. Moreover, research, such as the user study conducted on eSPACE [22], indicates that the resulting user interfaces may not consistently meet high standards of user-friendliness due to end users' limited design expertise. With these considerations in mind, we propose an innovative solution aimed at addressing these issues and streamlining the application development process while enhancing the user-friendliness of the created UIs. Our solution involves the automatic generation of IoT UIs and applications, guided by end users' specific requirements communicated through a multi-step wizard. This approach draws inspiration from ARNAULD [4] and will be elaborated upon in the following section.

3 THE IOT APPLICATION GENERATION MODULE

In this section, we introduce our IoT Application Generation Module which has been implemented as an extension to the eSPACE end-user authoring tool [22]. Therefore, we first provide a brief overview of eSPACE. Then, we delve into our automatic IoT Application Generation Module, highlighting its dialog-based interface and implementation details.

3.1 The eSPACE End-User Authoring Tool

With eSPACE, end users can create their own cross-device and IoT applications using its different authoring views in the web browser of their choice. Two of these authoring views are dedicated to the creation of interaction rules. One view offers a visual approach, using the pipeline metaphor, while the other provides a textual counterpart where users can create rules employing the IF-THEN notation. For instance, users can define rules such as "At 8am, turn on the bedroom light" or "IF the 'light' button UIe is clicked THEN toggle the bedroom light". The second example illustrates a rule involving a user interface element (UIe), specifying that when this UIe—in this case a button—is clicked, the bedroom light will be toggled on or off. For the creation of such UI elements, eSPACE offers a third authoring view called the *UI design* view, providing a user-friendly drag-and-drop interface for designing custom user interfaces (see Figure 1).

Lastly, eSPACE includes the *home* view and the *app* view. In the *home* view users can see all their created applications, rules and devices. Opening an application triggers eSPACE to launch the *app* view in a new browser tab, presenting the app's UI for immediate use. If an application features multiple UIs, individual tabs will open, each representing one UI.

[1]https://www.think-silicon.com/gui-builder/. Last accessed on April 29, 2024.
[2]https://www.embedded-wizard.de/. Last accessed on April 29, 2024.

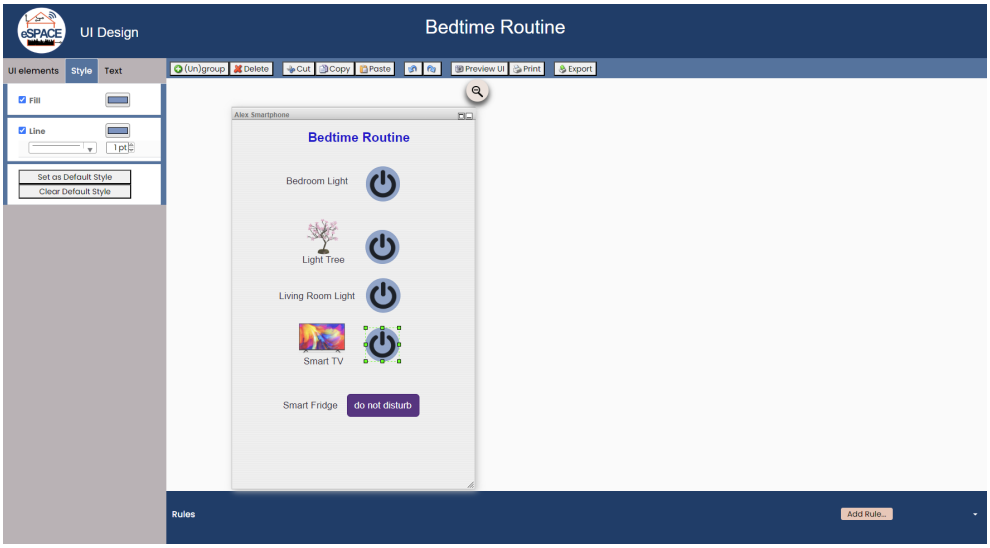

Fig. 1. UI design view of the eSPACE end-user authoring tool

In order to facilitate the UI development process, eSPACE adopts a model-based approach. Therefore, it adheres to the different layers outlined in the eSPACE reference framework [24], hereby decomposing the UI development process. In order to construct the final user interface (FUI) model, elements from the different layers are linked together. Typically, this involves linking user interface elements to "active components" which contain programming logic. For instance, a button UIe can be linked to the "turn on" functionality, allowing users to control a smart device using this button. Once the FUI is created, it is parsed and transformed into an executable program, a process executed within the *app* view and which effectively turns the model into a functional application. It is beyond the scope of this paper to provide further details, but a detailed explanation and an example of an application model can be found in Chapter 4 of Sanctorum's work [22].

While eSPACE's end-user evaluations consistently indicated favourable results in terms of usability and user satisfaction, user-generated UIs indicated room for improvement, which is natural given that end users do not possess the design expertise of dedicated UI designers. Additionally, feedback highlighted the time-consuming nature of the application creation process. In response, we developed an integrated application generation module to streamline this process and improve UI designs. In the following section, we present this module.

3.2 The Application Generation Module

The Application Generation Module enables end users to generate IoT applications tailored to their specific needs. The module serves a dual purpose, streamlining the application creation process and enhancing the quality of the UI designs. It is implemented using Spring Boot and seamlessly integrates with eSPACE's visual style and functionality. In order to access the module, users can simply click the "Generate App..." button on eSPACE's *home* page. This action redirects them to a multi-step form wizard—inspired by ARNAULD [4]'s active elicitation method—through which they can specify their IoT application requirements. Note that, in contrast to ARNAULD, which uses this method to elicit

end-user preferences among two UI options for refining an underlying cost function, our approach leverages it to build the UI from scratch, based on the end user's choices.

3.2.1 *Key UI Design Considerations.*

In the context of eSPACE, where users create IoT applications for various devices such as tablets, smart TVs, and smartphones, it is crucial to consider the diverse screen sizes and capabilities of these devices when designing the user interface. In this initial prototype, we mainly focus on addressing screen size variations. This involves adjusting the font size used in the UI according to the screen size of the device. We identified some relevant studies in this domain. One study, for instance, stated that a minimum font size of 22pt is well-suited for Smart TVs [18], while another, focusing on an older smartphone model (Siemens S45)—which screen width can be likened to contemporary smartwatches—suggested a font size of 12pt [30]. It is worth noting that various online guidelines offer recommendations for font selection and sizing tailored to specific devices as well. For our prototype, we adopted the fluid font size approach proposed by Taylor [27], a professional web developer. This approach establishes font sizes relative to the screen width, using the formula `15px + 0.390625vw`, where `vw` represents the viewport width. Table 1 presents a summary of font sizes based on specific screen widths. In order to include Ziefle's findings [30], we added the ≤ symbol to first row, indicating a minimum font size of 16px (= 12pt). We implemented this solution to set the minimum font size for each device and determined the maximum font size by starting with the minimum font size and incrementally adding 1px as long as space permits. If there is no white space, the font size remains at the minimum value. This ensures that the font size is displayed optimally on the screen, considering the screens' dimensions and available space.

Screen Width	Font Size
≤ 320px	16px
768px	18px
1024px	19px
1280px	20px
1536px	21px
1920px	23px
2560px	25px

Table 1. Font size based on screen width [27]

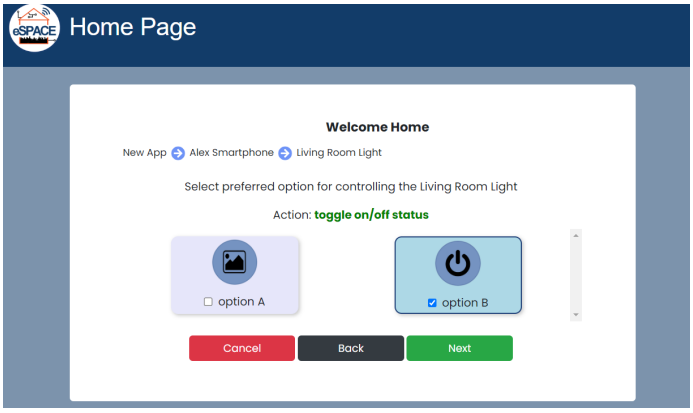

Fig. 2. Screenshot of the App Generation Module for selecting a UI component for the action toggle on/off

Beyond font size, it is important to suggest appropriate UI elements for selected actions. For instance, if a user wants to be able to set the colour of their smart light to "purple", recommending a toggle button would be inappropriate. In order to address this challenge, we employ a Smart Template approach as introduced by Nichols et al. [14]. This approach enriches UI elements with high-level semantic information, enabling interface generators to interpret and select the most suitable UI element(s) for each action. In our module, when end users select an action, they are presented with the most fitting UI elements for that action, as illustrated in Figure 2. This recommendation process is driven by a dedicated XML file containing "smart templates" that specify, for each action type, the appropriate UI elements. For

instance, actions involving adjusting values—–like altering a light's brightness—are typically matched with UI elements such as sliders or counters. Similarly, for "toggle" actions, as demonstrated in Figure 2, we have identified two suitable button types. Creating these templates is a manual process for each action. If a new action is configured and not yet incorporated into the templating system, default UI elements are provided, which are round and rectangular buttons containing either a label or an image. Note that this approach also increases the consistency across a wide range of applications.

Lastly, for creating the final user interface, accounting for screen size and selected UI elements, we currently offer two design alternatives. The first option centres all UI elements and stacks them vertically, while the second one divides elements into two columns, maintaining a centred alignment.

3.2.2 *Work-flow of the Multi-Step Form Wizard.*

In order to gather the user's application requirements, we employ a multi-step form wizard that guides them through a series of questions. The complete workflow of our multi-step form wizard is illustrated in Figure 3.

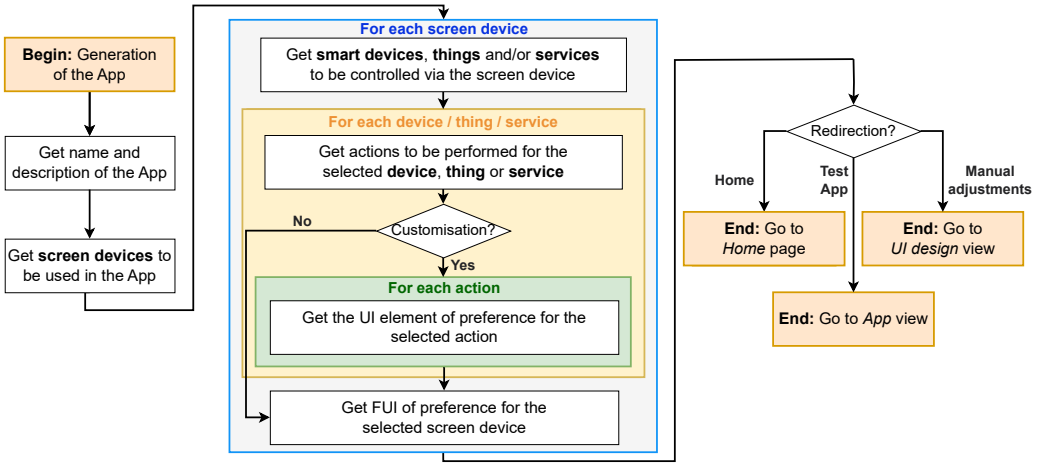

Fig. 3. Flowchart for the generation of IoT applications

The process begins by prompting the user to provide the name and description of their application. Next, the wizard proceeds by asking the user to specify the device(s) for which they wish to create a user interface within their application. It is important to note that in the flowchart, we explicitly mention *screen devices*, as our system can only design UIs for devices with screens. Given that users can create applications containing multiple user interfaces tailored for specific devices, they have the flexibility to select multiple devices during this step.

This entry marks the beginning of a series of loops in our workflow:

1. For each selected screen device, the user is prompted to specify the smart devices or *things* they wish to control using that particular screen device. This step initiates our second loop:

1.1 For each selected device or *thing*, the user is asked to indicate the actions they want to perform. After that, the user has the option to either immediately customise the representation (UI element) of these actions, or let the module decide for them by selecting the "try luck" button. If the user opts to customise the actions' representation, the third loop is initiated:

1.1.1 For each action, the user is requested to choose their preferred UI element (e.g. see Figure 2).
1.2 For each selected screen device, the user is required to specify their preferred final user interface from two available choices.

Once this final loop concludes, the user is given the option to proceed to one of three different views. They can navigate to the *home* page, access the *UI design* view (see Figure 1) for further UI modification, or request the system to open the generated application (*app* view).

3.2.3 Summary and Limitations.

As our module is an extension of the eSPACE end-user authoring tool, it aligns with eSPACE's model-based approach, ensuring both flexibility and extensibility. The generated applications are optimised for the specific devices they are designed for, yet they remain compatible with any device that has a web browser. However, an application tailored for a smartphone may not render as well on a tablet, though it will remain fully functional. Additionally, the module's integration with eSPACE enables modifications to these applications through the various authoring views that eSPACE provides.

While our initial prototype does not yet include all of eSPACE's features, it serves as a first proof of concept, empowering end users to quickly and easily create diverse smart home UIs. Nevertheless, our module inherits certain constraints, notably a limited selection of UI elements. Additionally, at this stage, we offer only two final design alternatives. We plan to address these limitations in future iterations. In order to ascertain the module's direction and its capacity to streamline the application creation process while delivering user-friendly UIs, we conducted an initial expert evaluation. This evaluation not only provides a primary assessment, but will also help us identifying concrete areas for early improvement, as detailed in the following section.

4 EXPERT REVIEW

In order to gain preliminary insights into the effectiveness of our IoT Application Generation Module, we conducted an expert evaluation involving six design specialists, including three females and three males. The participants, aged between 26 and 49 years (average: 36 years), were selected based on their extensive experience in UX design, with varying levels of UID experience spanning from 1 to 24 years (average: 11 years). Additionally, all participants possessed hands-on experience with at least one smart home application, making them well-versed in the domain. None of them had prior experience in using the eSPACE end-user authoring tool.

4.1 Setup and Protocol

To ensure a comprehensive evaluation, we established a structured protocol for the expert review. Participants were provided access to our IoT Application Generation Module on a laptop, with an overview of a fictional smart home displayed on an external screen. During the study, participants were given two predefined tasks to complete. These tasks were intentionally designed to assess the module's functionality, usability, and its ability to generate user-friendly UI designs according to the users' preferences, while at the same time showcase the diverse functionality of our module.

The study started by asking participants to complete the consent form. After that, they were given a brief explanation of the module's purpose. Note that, we deliberately omitted a tutorial to assess the clarity of information within the multi-step wizard. Participants were then requested to complete the two tasks, each involving the creation of an IoT application. In these tasks, designers assumed the role of an end user named Alex. For the first task, participants were asked to create the "*Bedtime Routine*" application involving the design of a UI for Alex's smartphone. This UI

should facilitate control (on/off) over four smart appliances within the smart home and provide the option to set the smart fridge to 'do not disturb' mode. For the second task, participants were asked to create the "*Home Arrival*" application which involved the creation of two UIs: one for the smartphone and another for the smart fridge. The smartphone UI should allow users to turn on the living room light and the water boiler smart plug. Meanwhile, the smart fridge UI had to display the weather information for Brussels and Tokyo, and provide users with the option to change the colour of the living room light to three pre-defined colours.

Throughout the evaluation process, participants were encouraged to think aloud and/or note down comments to capture their thoughts and impressions in real-time. After completion of the two tasks, participants were asked to fill out a post-study questionnaire to gather additional insights. The questionnaire included questions regarding task completion speed, the clarity and organisation of information within the multi-step wizard, satisfaction with the generated app designs, encountered difficulties, and any suggestions or comments. The complete study protocol is available online[3]

4.2 Results

The outcomes of the expert review provided valuable insights into the strengths and weaknesses of our IoT Application Generation Module. In this section, we present our key findings in terms of usability, functionality, UI designs and efficiency.

4.2.1 Usability. Experts found the module to be generally user-friendly, with an intuitive interface that allowed for easy navigation and task completion. Participant P3 noted that, "it (the wizard) guides you easily through the process" (P3). All participants agreed that the organisation of information on the system screens was clear. Responses to this question on a 7-point Likert scale—where 1 represents "strongly disagree" and 7 represents "strongly agree"—yielded a mean score of 5.8.

However, the experts suggested including previews when selecting different options, such as providing a visual preview when choosing a new button colour. Additionally, they recommended enabling users to revisit specific steps in the process after viewing the final user interfaces, which would allow for minor adjustments without the need to navigate back through the entire process.

4.2.2 Functionality. The module effectively generated IoT applications based on user preferences, demonstrating its capability to automate the design process. Participants reported that the module performed its intended functions well and allowed them to test applications using the fictional smart home view that displayed device statuses. Some participants did suggest improving the feedback messages associated with pressing power buttons. Specifically, they suggested providing a more detailed indication, such as specifying whether a light was turned on or off, as opposed to the current message "light toggled".

4.2.3 User Interface Designs. Most of the experts were satisfied with the generated UI designs. Nonetheless, there was some critique regarding the lack of originality, with one expecting more futuristic UI designs given the IoT context (P4). Another participant mentioned that the arrangements of UI elements seemed rudimentary (P5). As previously mentioned, the module currently offers a limited variety of UI elements and layouts, a point highlighted by participants as well.

An important observation pertained to the final design, which did not reflect the current status of smart appliances. For instance, the absence of visual cue, such as highlighting the power button in green to indicate that the light was on.

[3]The study materials and results are available on Zenodo. Accessible at: https://doi.org/10.5281/zenodo.11080010.

This issue relates to the feedback discussed in the previous subsection. Last but not least, participants also suggested providing feedback about colour choices during the authoring process to prevent issues with invisible labels and poor colour combinations.

4.2.4 Efficiency. Participants noted that the module streamlined the application design process, saving both time and effort compared to manual UI development. Responses to the question about the speed of app creation with our module ranged from "agree" to "strongly agree". On a 7-point Likert scale—where 1 represents "strongly disagree" and 7 represents "strongly agree"—the mean score was 6.2.

The average time to complete the first app was 7.5 minutes, while the second app took an average of 5 minutes. It is important to emphasise that these times serve as upper limits, as experts engaged in discussions and provided comments during the evaluation process. Nevertheless, they still demonstrate substantial time savings compared to traditional authoring methods, exemplified by eSPACE, where the creation time for apps with less functionality averaged 15 minutes [22].

4.3 Summary and Discussion of Future Work

The expert review yielded promising results, indicating that our IoT Application Generation Module holds significant potential. The positive feedback regarding usability, functionality, UI designs and efficiency suggests that the module is on the right track. However, the evaluation also identified areas for improvement, including suggestions to refine specific features and enhance the overall user experience. In the following, we detail the specific aspects for improvement based on the expert reviewers' input and outline our plans for implementing these enhancements.

4.3.1 Adding Visual Previews. One key suggestion is to incorporate visual previews when users make design choices, such as selecting buttons and colours. Additionally, we plan to provide an overall preview of the UI as it is constructed in real-time, giving users an immediate sense of the final UI's appearance. Further investigations will be conducted to determine the most effective approach.

4.3.2 Improved Recovery Mechanism. Currently, users must rely on the back button to correct mistakes made in previous steps, which is not the most efficient method. To enhance user experience, we are exploring mechanisms that would allow users to make quick changes when viewing the final UI preview. Suggestions from experts include the option to directly click on the UI element for modification or allowing users to revisit specific steps by providing a clickable overview of their selected options during the process.

4.3.3 Enhanced Colour Scheme Management. Participants found the colour selection process somewhat tedious, especially when selecting colours repeatedly throughout the process. One participant mentioned, "at some point you need to do the same all the time" (P5). To simplify this, we are considering requesting colour selection only at the end. Additionally, we plan to introduce pre-defined colour schemes to help users choose harmonious colour combinations. Lastly, we also aim to assist users further by adding tips during the process to prevent poor colour choices, such as displaying a warning message when selecting white text on a white background.

4.3.4 Diverse UI Designs. As highlighted by experts, we acknowledge that our current UI designs are rather simplistic. To address this, we intend to diversify the available UI elements and layout options. In doing so, we also plan to expand our "smart templates" by adding new matches between device actions and UI elements. Further, we will draw inspiration from existing research, including the work by Oulasvirta et al. [16], to optimise graphical UI designs.

4.3.5 Minor Revisions. The expert review also revealed some straightforward improvements. For instance, the term "turn" was found to be confusing, leading to differing interpretations. We plan to rename it to "turn on or off" for clarity. Participants also noted that the proximity of the back button to the cancel button could result in accidental clicks. We will address this issue by increasing the spacing between these buttons. Finally, two experts highlighted the absence of a back button in the last step, which we will include in future versions.

5 CONCLUSIONS

In this paper, we introduced a novel approach to empower end users in generating IoT applications based on their needs and preferences. Our method employs a multi-step form wizard and is built as an extension module to the eSPACE end-user authoring tool. The primary goal is to streamline the application creation process, making it faster and accessible to individuals without design expertise while ensuring user-friendly designs. Our initial prototype has shown promise in an expert review, earning recognition for its clear and well-structured step-by-step wizard, robust functionality, efficient app creation process, as well as positive and constructive feedback on UI designs. This evaluation has validated our approach and provided valuable insights to guide future enhancements. For future work, we plan to refine our module based these insights, as detailed in the preceding section. Additionally, we envision conducting user studies with end users to assess real-world effectiveness and gather direct user feedback. Over time, we aim to develop a rich templating system that can be shared with and used by other UI generation tools to enhance their designs. Last but not least, through this paper, we hope to stimulate further research in the field of end-user-driven UI generation.

ACKNOWLEDGMENTS

The research of Audrey Sanctorum has been funded by an FWO Postdoc Fellowship (1276721N) of the Research Foundation Flanders.

REFERENCES

[1] 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. *Journal of Systems and Software* 149 (2019), 101–137. https://doi.org/10.1016/j.jss.2018.11.041

[2] 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

[3] Krzysztof Gajos and Daniel S. Weld. 2004. SUPPLE: Automatically Generating User Interfaces. In *Proceedings of IUI 2004, 9th International Conference on Intelligent User Interfaces*, Jean Vanderdonckt, Nuno Jardim Nunes, and Charles Rich (Eds.). ACM, Madeira, Portugal, 93–100. https://doi.org/10.1145/964442.964461

[4] Krzysztof Gajos and Daniel S. Weld. 2005. Preference Elicitation for Interface Optimization. In *Proceedings of UIST 2005, 18th Annual ACM Symposium on User Interface Software and Technology*, Patrick Baudisch, Mary Czerwinski, and Dan R. Olsen (Eds.). ACM, Seattle, WA, USA, 173–182. https://doi.org/10.1145/1095034.1095063

[5] Krzysztof Z. Gajos, Daniel S. Weld, and Jacob O. Wobbrock. 2010. Automatically Generating Personalized User Interfaces with SUPPLE. *Artificial Intelligence* 174, 12-13 (2010), 910–950. https://doi.org/10.1016/j.artint.2010.05.005

[6] Tony Griffiths, Peter J. Barclay, Norman W. Paton, Jo McKirdy, Jessie B. Kennedy, Philip D. Gray, Richard Cooper, Carole A. Goble, and Paulo Pinheiro da Silva. 2001. Teallach: A Model-Based User Interface Development Environment for Object Databases. *Interacting with Computers* 14, 1 (2001), 31–68. https://doi.org/10.1016/S0953-5438(01)00042-X

[7] Björn A. Johnsson and Boris Magnusson. 2020. Towards End-User Development of Graphical User Interfaces for Internet of Things. *Future Generation Computer Systems* 107 (2020), 670–680. https://doi.org/10.1016/j.future.2017.09.068

[8] Mohammad Amin Kuhail, Shahbano Farooq, Rawad Hammad, and Mohammed Bahja. 2021. Characterizing Visual Programming Approaches for End-User Developers: A Systematic Review. *IEEE Access* 9 (2021), 14181–14202. https://doi.org/10.1109/ACCESS.2021.3051043

[9] 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

[10] Marina Machado, Rui Couto, and José Creissac Campos. 2017. MODUS: Model-based User Interfaces Prototyping. In *Proceedings of EICS 2017, Symposium on Engineering Interactive Computing Systems*, José Creissac Campos, Nuno Nunes, Pedro Campos, Gaëlle Calvary, Jeffrey Nichols, Célia Martinie, and José Luís Silva (Eds.). ACM, Lisbon, Portugal, 111–116. https://doi.org/10.1145/3102113.3102146

[11] Gerrit Meixner, Fabio Paternò, and Jean Vanderdonckt. 2011. Past, Present, and Future of Model-Based User Interface Development. *i-com* 10, 3 (2011), 2–11. https://doi.org/10.1524/icom.2011.0026

[12] Brad A. Myers, Scott E. Hudson, and Randy F. Pausch. 2000. Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction* 7, 1 (2000), 3–28. https://doi.org/10.1145/344949.344959

[13] Thanh-Diane Nguyen, Jean Vanderdonckt, and Ahmed Seffah. 2016. Generative Patterns for Designing Multiple User Interfaces. In *Proceedings of MOBILESoft 2016, International Conference on Mobile Software Engineering and Systems*. ACM, Austin, Texas, USA, 151–159. https://doi.org/10.1145/2897073.2897084

[14] Jeffrey Nichols, Brad A. Myers, and Kevin Litwack. 2004. Improving Automatic Interface Generation with Smart Templates. In *Proceedings of IUI 2004, 9th International Conference on Intelligent User Interfaces*, Jean Vanderdonckt, Nuno Jardim Nunes, and Charles Rich (Eds.). ACM, Funchal, Madeira, Portugal, 286–288. https://doi.org/10.1145/964442.964507

[15] 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

[16] Antti Oulasvirta, Niraj Ramesh Dayama, Morteza Shiripour, Maximilian John, and Andreas Karrenbauer. 2020. Combinatorial Optimization of Graphical User Interface Designs. *Proceedings of the IEEE* 108, 3 (2020), 434–464. https://doi.org/10.1109/JPROC.2020.2969687

[17] Fabio Paternò, Carmen Santoro, and Lucio Davide Spano. 2009. MARIA: A Universal, Declarative, Multiple Abstraction-Level Language for Service-Oriented Applications in Ubiquitous Environments. *ACM Transactions on Computer-Human Interaction* 16, 4 (2009), 19:1–19:30. https://doi.org/10.1145/1614390.1614394

[18] Emmanouil Perakakis and Gheorghita Ghinea. 2015. Responsive Web Design for the Internet Connected TV: The answer to more Smart TV Content?. In *Proceedings of ICCE-Berlin 2015, 5th International Conference on Consumer Electronics*. IEEE, Berlin, Germany, 38–42. https://doi.org/10.1109/ICCE-Berlin.2015.7391286

[19] Angel R. Puerta. 1997. A Model-Based Interface Development Environment. *IEEE Software* 14, 4 (1997), 40–47. https://doi.org/10.1109/52.595902

[20] Angel R. Puerta, Henrik Eriksson, John H. Gennari, and Mark A. Musen. 1994. Beyond Data Models for Automated User Interface Generation. In *Proceedings of HCI 1994, People and Computers IX*, Gilbert Cockton, Stephen W. Draper, and George R. S. Weir (Eds.). Cambridge University Press, Glasgow, UK, 353–366.

[21] Jenny Ruiz, Estefanía Serral, and Monique Snoeck. 2019. Evaluating User Interface Generation Approaches: Model-based Versus Model-driven Development. *Software and Systems Modeling* 18, 4 (2019), 2753–2776. https://doi.org/10.1007/s10270-018-0698-x

[22] 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.

[23] 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, Proceedings of the 11th 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

[24] Audrey Sanctorum and Beat Signer. 2019. A Unifying Reference Framework and Model for Adaptive Distributed Hybrid User Interfaces. In *Proceedings of RCIS 2019, 13th 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

[25] Pedro A. Szekely, Ping Luo, and Robert Neches. 1992. Facilitating the Exploration of Interface Design Alternatives: the HUMANOID Model of Interface Design. In *Proceedings of CHI 1992, Conference on Human Factors in Computing Systems*, Penny Bauersfeld, John Bennett, and Gene Lynch (Eds.). ACM, Monterey, CA, USA, 507–515. https://doi.org/10.1145/142750.142912

[26] Pedro A. Szekely, Piyawadee Noi Sukaviriya, Pablo Castells, Jeyakumar Muthukumarasamy, and Ewald Salcher. 1995. Declarative Interface Models for User Interface Construction Tools: the MASTERMIND Approach. In *Proceedings of IFIP 1995, Working Conference on Engineering for Human-Computer Interaction (IFIP Conference Proceedings, Vol. 45)*, Leonard J. Bass and Claus Unger (Eds.). Chapman & Hall, Yellowstone Park, USA, 120–150.

[27] Matthew James Taylor. 2022. Responsive Font Size (Optimal Text at Every Breakpoint). https://matthewjamestaylor.com/responsive-font-size. Accessed: 2023-10-03.

[28] Vi Tran, Jean Vanderdonckt, Manuel Kolp, and Yves Wautelet. 2010. Generating User Interface for Information Applications from Task, Domain and User models with DB-USE. In *Proceedings of UsiXML 2010, 1st International Workshop on User Interface eXtensible Markup Language*. Berlin, Germany, 184–194.

[29] Junfeng Wang, Zhiyu Xu, Xi Wang, and Jingjing Lu. 2022. A Comparative Research on Usability and User Experience of User Interface Design Software. *International Journal of Advanced Computer Science and Applications* 13, 8 (2022). https://doi.org/10.14569/IJACSA.2022.0130804

[30] Martina Ziefle. 2010. Information Presentation in Small Screen Devices: The Trade-Off Between Visual Density and Menu Foresight. *Applied Ergonomics* 41, 6 (2010), 719–730. https://doi.org/10.1016/j.apergo.2010.03.001 Special Section: Selection of papers from IEA 2009.