Cognition, Technology & Work (2024) 26:83–106
https://doi.org/10.1007/s10111-023-00749-z

ORIGINAL ARTICLE

A conceptual framework for context-driven self-adaptive intelligent user interface based on Android

Mughees Ali[1] · Saif Ur Rehman Khan[2] · Atif Mashkoor[3] · Anam Taskeen[1]

Received: 6 September 2023 / Accepted: 27 November 2023 / Published online: 3 January 2024
© The Author(s) 2023

Abstract

Adaptive User Interface (AUI) can change its layout, appearance, and/or elements based on the needs of its user requirements and current usage context. The AUIs are used in state-of-the-art software products, applications for mobile devices, and websites. Moreover, AUI is an emerging research field in a mobile context, as it can enhance usability, performance, and user satisfaction. This study aims to propose a conceptual framework for developing a real-time self-adaptive user interface based on the Android Operating System (OS). Furthermore, the focus is on developing the core algorithms for the modules of the proposed framework. To evaluate the performance of the proposed framework, three case studies have been designed based on the daily and weekly activities of the user. Moreover, an expert-based validation approach is employed to obtain the expert's feedback regarding the proposed framework. The result indicates that the proposed framework helps improve user satisfaction and experience by making an intelligent mobile device interface. The results of the framework's evaluation and validation show the proposed framework's feasibility and effectiveness. We conclude that the current work is beneficial in filling the identified research gap. Moreover, this research shows the significance of an adaptive interface in an Android OS-based context. In addition, it not only helps in improving the user interest and satisfaction but also enhances the overall performance of the mobile device.

Keywords Adaptive user interface · Self-adaptive user interface · Interface design · Smartphone user interface · Android-based adaptive user interface

1 Introduction

Smartphones offer valuable features and portability. As a result, they have significantly improved our professional and personal lives (Hu et al. 2018). Due to the demand for smartphones in the market, the features in the smartphone Operating Systems (OS) and the number of smartphone application users and developers also increased greatly. At the same time, new challenges are emerging in each release of mobile OS due to user satisfaction. Smartphones are widely used, and different OS-based mobile phones are available. With a market share of over 80%, Android continues dominating the global smartphone market (Wu et al. 2021).

User expectations from the mobile OS, like adaptive user interface and different suggestions according to their interest, increase the desired search outcomes. The User Interface (UI) is a key component of any interactive system or application that allows the users to access the system or application features directly (Hussain et al. 2018). The UI links the user and the interactive system or device, and it should be easy to

Mughees Ali, Saif Ur Rehman Khan, Atif Mashkoor and Anam Taskeen contributed equally to this work.

✉ Atif Mashkoor
atif.mashkoor@jku.at

Mughees Ali
mughees.ali101@gmail.com

Saif Ur Rehman Khan
saif_rehman.ssc@stmu.edu.pk

Anam Taskeen
anamtaskeen573@gmail.com

[1] Department of Computer Science, COMSATS University Islamabad (CUI), Tarlai Kalan, Park Road, Islamabad 45550, Pakistan

[2] Department of Computing, Shifa Tameer-e-Millat University (STMU), Park Road Campus, Islamabad 44000, Pakistan

[3] Institute of Software Systems Engineering, Johannes Kepler University, Altenbergerstraße 69, 4040 Linz, Austria

use and understand (Yigitbas et al. 2019). The smartphone interface offers content and reliable communication between the device and its user. Smartphones also have a variety of user types and modes of interaction. Numerous high-quality applications have been published, but their interfaces are complex, unattractive, and frustrating, causing them to be overlooked by the user. This discomfort is caused by a non-adaptive interface for mobile devices, which directly impacts performance, usability, and satisfaction among different users. One of the major drawbacks of a non-adaptive interface is that it cannot be adapted to the essential changes even if the usage environment/context changes (Tanaka et al. 2019).

Adaptive User Interface (AUI) is a growing research field (Rathnayake et al. 2019), and it adapts to the user's profile/context and platform (Soui et al. 2017). The AUI interface changes its design and components based on the user's current usage context. Moreover, AUI is based on the user's behavioral patterns and changes automatically to provide a customized experience (Rathnayake et al. 2019). The ability of a framework to create an intelligent and customized interface based on the user's needs is referred to as interface adaptation (Soui et al. 2017). The AUI also aims to enhance user and UI interaction by adapting device layout distribution and available actions to the user's current goals and needs (Machado et al. 2018).

AUI's are used in software systems and by many popular e-commerce websites like eBay and Amazon to improve usability and customize the purchasing experience (Rathnayake et al. 2019). The interface adaptation has been promoted to solve usability problems and satisfy the user's needs and preferences (Soui et al. 2017). Based on the literature, modern systems have several usability issues. The interface's ambiguity and lack of flexibility are primarily responsible for these issues (Wesson et al. 2010). AUIs have the potential to help with these usability problems. The user specifications should be assessed regarding accessibility, learnability, understandability, effectiveness, performance, and objectivity while designing mobile phone interfaces. Normally, the users are provided with factory-set profiles or moods. The context information and external interference are ignored in adaptive interface design for mobile devices based on the traditional user model. The interface has become a bottleneck due to its inability to meet various usage contexts to improve software availability (Wang et al. 2014).

The UI is an essential part of any interactive device or application, and it is not independent of its context-of-use, defined by the user, platform, and environment (Yigitbas et al. 2017). Surely, unattractive, unreliable, and frustrating interfaces in a non-adaptive user interface for mobile devices directly impact usability, user satisfaction, and performance (Iqbal et al. 2018). The best solution to the above-mentioned issues is through the Self-Adaptive User-Interface (SAUI) (Machado et al. 2018). SAUI can be viewed as an interface that adjusts its structure or content based on the needs and usage contexts (Braham et al. 2019). The SAUI aims to enhance the user and UI interaction by adapting aspects, including device layout distribution and available actions, to the user's current goals and needs (Machado et al. 2018). Ideally, the mobile OS must adjust its interface and performance according to its current usage context. We acknowledge that significant work has been done in the UI software adaptation domain. However, limited work is performed on Android-based self-adaptive smartphone/mobile applications. Moreover, the current state-of-the-art lacks any framework or model in the Android OS-based SAUI context. This acts as the main underlying motivation to conduct the current research work. As the successful adaptation improves the overall usability and performance (Raheel 2016), there is a need for a conceptual framework that targets the Android operating system to make its interface intelligent and effective.

Figure 1 provides an overview of the working mechanism of self-adaptation in the Android OS (indicated through the pink line) and in the mobile-based application context (shown by the black line). Figure 1 represents the two main phases with several sub-steps. Phase 1 is related to the OS-based and any domain-specific application and is regarded as a Managed System phase. In an OS-based context, sensor data are gathered in a particular usage context (step 1). Similarly, a user employs a mobile application in a domain-specific application-based context. Then, the application updates its behavior according to the context, such as security, health-based apps, and so on (step 1). In contrast, phase 2 gives a high-level overview of the working of a self-adaptive mobile OS and mobile application and is regarded as a Managing System phase. The data from the self-adaptive OS or the application is collected (step 2). All the gathered data is sent to some adaptive scheme, technique, algorithm, or framework (step 3), which monitors changes, modifies them, and updates them according to their usage context (step 4). Finally, at the last step of Phase 2 (step 5), a successful adaptation is made according to their usage context (Fig. 1).

The main Research Contributions (RCs) of this work are:

- *RC1* Review the current state-of-the-art to identify the research gap in an adaptive user interface in a mobile context.
- *RC2* Report the widely-used self-adaptive models mentioned in the literature.
- *RC3* Propose a conceptual framework for Android OS-based SAUI context.
- *RC4* Develop the core algorithms supporting different modules of the proposed framework.

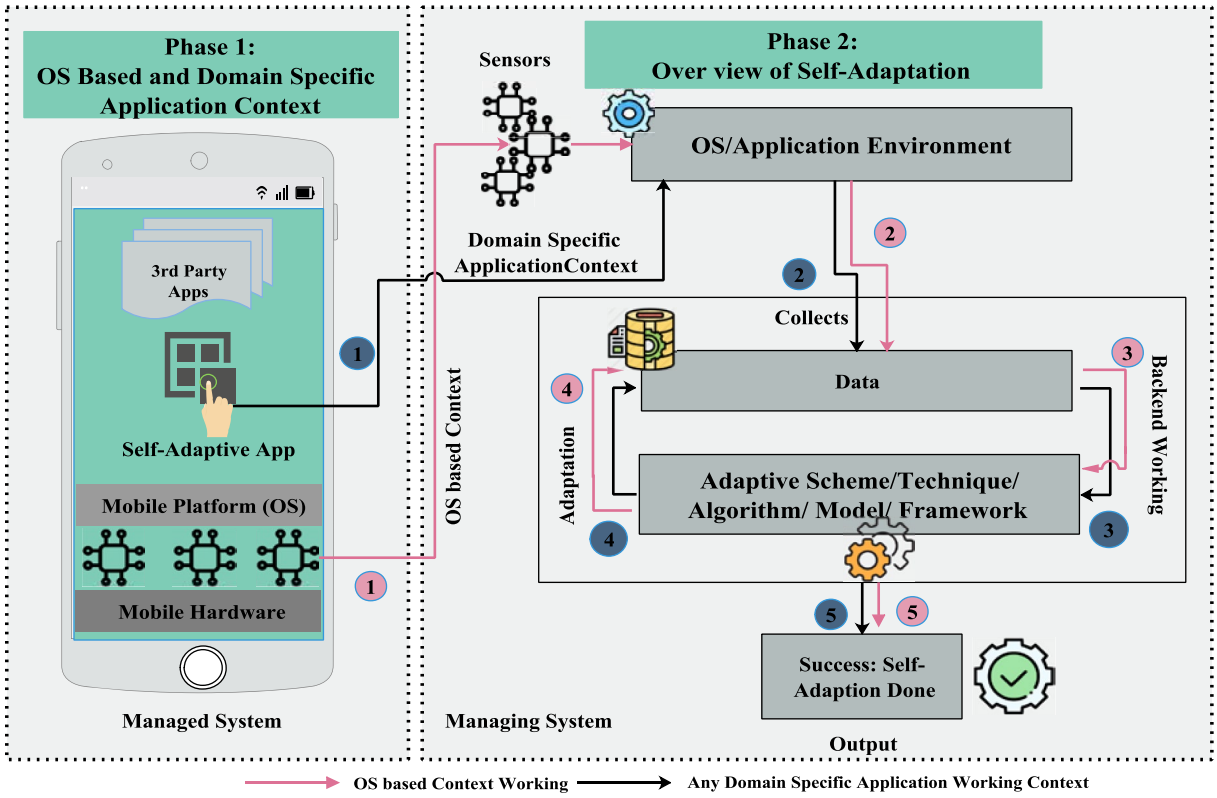


Fig. 1 General workflow of self-adaption in smartphone context

- *RC5* Evaluate the performance of the proposed framework through case studies and expert-based validation techniques.

The following Research Objective (RO) has been focused on in this work:

- *RO* To propose and validate the SAUI based conceptual framework.

Two main Research Questions (RQs) have been formulated related to RO, as follows:

- *RQ1* What is the attained performance of the proposed framework using the designed case studies?
- *RQ2* What is the practitioner's feedback regarding the proposed framework?

The rest of the article is organized as follows: Sect. 2 provides the related work about the self-adaptive models. The adapted research methodology is discussed in Sect. 3, while Sect. 4 presents the proposed framework. In contrast, Sect. 5 provides the evaluation and validation criteria. The results and discussion are mentioned in Sect. 6, while threats to validity are provided in Sect. 7. Section 8 is about the research implications, and finally, Sect. 9 concludes the current work and outlines potential future research directions.

2 Related work

The same authors have already performed a detailed systematic literature review on self-adaptive mobile-based applications (Ali et al. 2021). Current state-of-the-art shows the interest of researchers in the domain of the adaptive user interface in the smartphone context. Machado et al. (2018) proposed a conceptual framework for real-time adaptive interfaces for web-based applications. The authors addressed the common challenges older people face, such as vision loss and cognitive decline. However, they focused only on older people and targeted web-based applications in their work. Similarly, an adaptive framework is proposed by Wattearachchi et al. (2020), which recommends different functions based on the user's emotions and context parameters, including place, time, and current user behavior, that can be used to enhance the user experience. The authors focused on facial expressions in their proposed framework. The user experience may be affected due to the low visibility/light, age factors, and gender, as mentioned in their work. In contrast, Yigitbas et al. (2017) presented a model-driven

Integrated Development Environment (IDE). The authors named their IDE Adapt-UI, which provides integrated views for UI, context, and adaptation modeling. Moreover, the authors validated their IDE by designing a case study.

Similarly, Mobile Enterprise Resource Planning (ERP) will improve the enterprise's long-term viability and quality of life for employees and people's economic prosperity. As a result of the numerous advantages, mobile ERP has become an important part of enterprises with ERP systems. Omar and Gomez present an adaptive framework architecture (Omar and Gómez 2017) to support automatic UI component adaptation in mobile ERP apps based on the context of use. However, due to the usability problem of ERP systems and the mobile sense of usage, their model suffers from low usability. An interface prototype is developed by Lee et al. (2011) for Android smartphones suggesting a variety of apps based on the user's current usage context. Moreover, based on a perceptual control theory, the authors (Wang et al. 2014) suggested an adaptive user interface model for mobile devices by analyzing the usage context, exporting user intent, and re-designing the context to satisfy the requirement of efficient control using perceptual control theory. Furthermore, an ontology-based approach is proposed by Soui et al. (2017) to automatically recommend AUIs based on the user's context using Semantic Web Rule Language (SWRL) rules. In contrast, Khan and Khusro (2019) designed a blind-friendly user interface framework for performing common smartphone tasks. Based on the user contextual profile, the proposed framework reorganizes the interface components, and the authors targeted a simple layout design for the interface in their work. Similar work is done by Khan et al. (2018) as blind persons having difficulty accessing and running mobile device elements such as finding a button, recognizing templates, and navigating the interface. The authors targeted simple applications and layout designs in their work and mentioned performing work on deep/complex application interfaces as future work.

An adaptive mechanism is proposed by Raheel (2016), which can track user behavior on a mobile phone and adjust the interface accordingly. Furthermore, Feng and Liu (2015) proposed a context-aware adaptive model for a mobile location-based service interface. In contrast, Deuschel and Scully (2016) designed different principles for the interface that can change automatically. The authors Iqbal (2022) presented a semantic context model for AUI frameworks that is based on four contexts: user, device, task, and environment. The model is created using Protégé and verified for consistency and data availability. The model is designed to be versatile, scalable, and semantically verified and can be mapped to individual UI displays through smart calculations for versatile UIs. The available SMS apps are not suitable for drivers due to safety concerns. The ConTEXT application is proposed by Khan and Khusro (2022), which addresses this by adapting interfaces based on driving conditions, improving safety, and reducing distractions. Empirical evaluation shows that ConTEXT minimizes risky interactions for drivers, enhancing safety.

Omar and Gómez (2017) examined how AUIs can help boost mobile app usability. The authors explored a few basic forms of adaptation that have been shown to improve mobile app usability significantly. Yigitbas et al. (2019) proposed a framework for creating AUIs which consist of numerous components that cover different aspects such as context monitoring and widget-level UI adaptation. Similarly, Yigitbas et al. (2019) presented a novel method for evaluating the usability of UI adaptation for mobile devices, emphasizing end-user satisfaction. The authors presented an on-the-fly usability testing solution and tried to solve the problems of existing methods for assessing user satisfaction with adaptive UIs.

Smart TVs are complex devices with a wide range of features, which can make their UIs difficult to use and navigate. Additionally, multiple people with different needs and preferences often use smart TVs. This makes it difficult to design a UI that is suitable for everyone. Khan and Khusro (2023) discuss the challenges of designing user interfaces (UIs) for smart TVs and propose a personalized adaptive UI as a solution. Similarly, a smart TV-based work is performed by Khan and Khusro (2023). The authors proposed a smart TV-based lifelogging system called SmartLog. SmartLog is a framework for logging specific information about watching experiences, such as the channels and programs watched, the times of day they are watched, and the duration of the viewing sessions. SmartLog also collects environmental data, such as the room temperature and light level. Another work done by Khan et al. (2022), as smart TVs have become increasingly complex in recent years, with new features and functionalities being constantly added. This has made it difficult for designers to create UIs that are both easy to use and visually appealing. The authors figure out that one of the biggest challenges in designing smart TV UIs is the diversity of users. Another challenge is the limited input methods available on smart TVs. Despite these challenges, the authors also mentioned some opportunities for designing better smart TV UIs.

The authors Khan and Khusro (2020) proposed a context-aware adaptive user interface framework called DriverSense to minimize driver distraction. DriverSense is implemented on the Android platform and adapts smartphone user interfaces based on real-time contextual factors such as driver preferences, environmental factors, and device usage using adaptation rules. In contrast, Braham et al. (2019) presented a framework that combines ontology-based models with UI design pattern methods to create adaptive mobile apps. Their proposed framework allows for the dynamic collection of appropriate UI design patterns based on user

needs. Moreover, an integrated model-driven development approach is presented by Yigitbas et al. (2020) for self-adaptive UIs based on traditional model-driven UI development approaches.

Table 1 shows that several studies target a single mobile application or a specific application category or provide literature-based general guidelines and principles. Moreover, several studies have not performed any validation process for their proposed models or frameworks. In addition, a user may want to undo or redo the adapted changes based on his or her interest, which is also lacking in the reported literature. User interest and involvement are important factors for generating AUIs, which help increase the user experience and improve satisfaction.

2.1 Self-adaptive model identification

Monitor Analyze Plan Execute-Knowledge (MAPE-K) is a self-adaptive systems' influential reference control model. From the literature, we also identified the MAPE-k model as the widely-used self-adaptive model in smartphone application context (Casquina et al. 2016; Omar and Gómez 2017; Yigitbas et al. 2019, 2020; Moghaddam et al. 2017; Cañete et al. 2020; Naqvi et al. 2016; Amoud and Roudies 2017; Velázquez-García et al. 2018; Evers and Geihs 2014; Arcaini et al. 2015).

MAPE-K model contains five main modules. The flow of the MAPE-K model is shown in Fig. 2. Monitor (step 1) is the first module that collects data about the environment and the managed resources, such as status, metrics, configuration settings, etc. Furthermore, the monitor compiles the data to identify symptoms/signs and sends these symptoms/signs to the Analyze module for further processing (Lara et al. 2019). The analyze module (step 2) includes the mechanism for analyzing and reasoning the data sent by the Monitor. Moreover, the analysis module determines whether there is a need for adaptation. Analyze may use prediction frameworks or models (Lara et al. 2019) for data analysis. If there is a need for adaptation, it sends a request to the Plan module. The Plan module (step 3) generates a plan for modifying the managed resource and then passes the plan to the Execute module (step 4). The Execute module executes a set of actions provided by the Plan module to adjust the managed resource following the generated plan (Lara et al. 2019). The knowledge module (step 5) saves and maintains all the information from other MAPE-K model modules (Fig. 2).

We consider the MAPE-K model for the proposed framework due to its flexibility, with the addition of two new modules based on the framework requirement. The Adaptation Controller module is one of the modules that help control adaptation rules and other adaptation-related issues. The second module is User Validation/Verification, which helps enhance user satisfaction. Thus, the proposed framework contains seven modules to effectively handle the user interface adaptation (detailed discussion in Sect. 4).

Table 1 Literature review on AUIs

References	Research work description	Methodology	Limitation	Target area
Yigitbas et al. (2020)	Proposed an approach that performs adaptation at run time	Model-driven self-adaptive UIs approach	Validation is missing, Knowledge is missing in their adaptation loop	Target single application
Iqbal et al. (2018)	Based on the survey, proposed a user behavior-centric model	Survey-based work	Provide guidelines for the AUI development, Validation is missing	Literature-based general guidelines
Yigitbas et al. (2019)	Provide a solution for evaluating the usability of mobile UI adaptations	Proposed on-the-fly usability testing solution	User is unable to undo the adaptive changes	Target single application
Braham et al. (2019)	Support of design patterns in generating adaptive mobile user interfaces	Adaptive user interface design patterns	Expert's validation is missing	Target single application
Machado et al. (2018)	A conceptual framework for creating an interactive User Interface for a web-based application has been proposed	Proposed a conceptual framework	Validation is missing	Target single application. Target a specific age of persons
Omar and Gómez (2017)	Proposed adaptive UI component-based architecture in mobile ERP apps	Proposed conceptual model	Poor usability of the model due to several challenges, Validation is missing	Target specific category of applications
Lee et al. (2011)	The authors conduct two literature reviews in the AUI domain and design 8 principles for adaptive user interfaces	Conducted two simple literature reviews	Validation is missing. Methodological evaluation lacks	Literature-based general guidelines

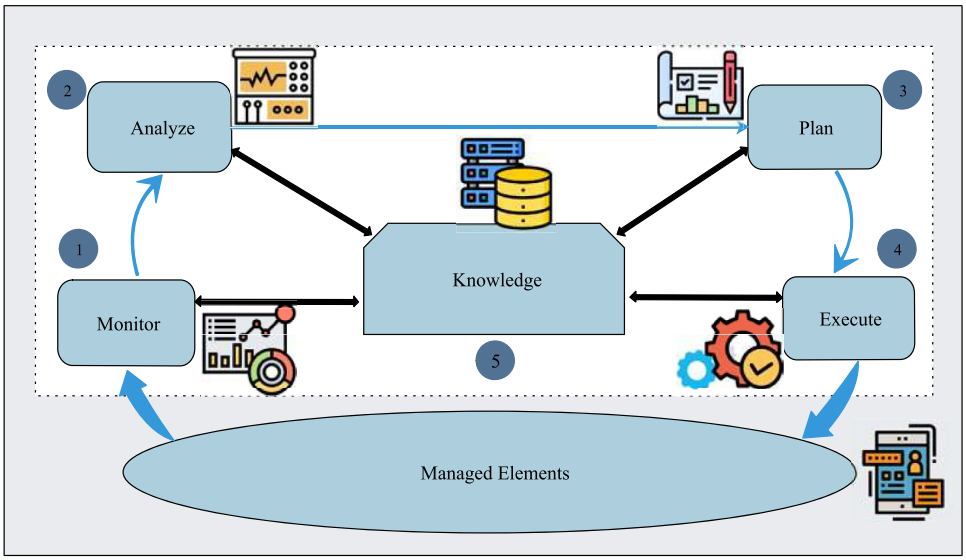


Fig. 2 MAPE-K model workflow

3 Research methodology

The adopted research methodology for the current work is based on three phases:

- In Phase 1, a literature review is conducted in the targeted domain to determine the limitations of the relevant work (discussed in Sect. 2).
- In Phase 2, an Android-based conceptual framework is proposed grounded on the limitations of the previously conducted work (discussed in Sect. 4).
- Phase 3 is related to the evaluation and validation of the framework. Evaluation is done by the state-of-the-art (by designing case studies). In contrast, validation is done by state-of-the-practice (expert-based validation technique by identifying the domain experts and validating the framework), as shown in Fig. 3.

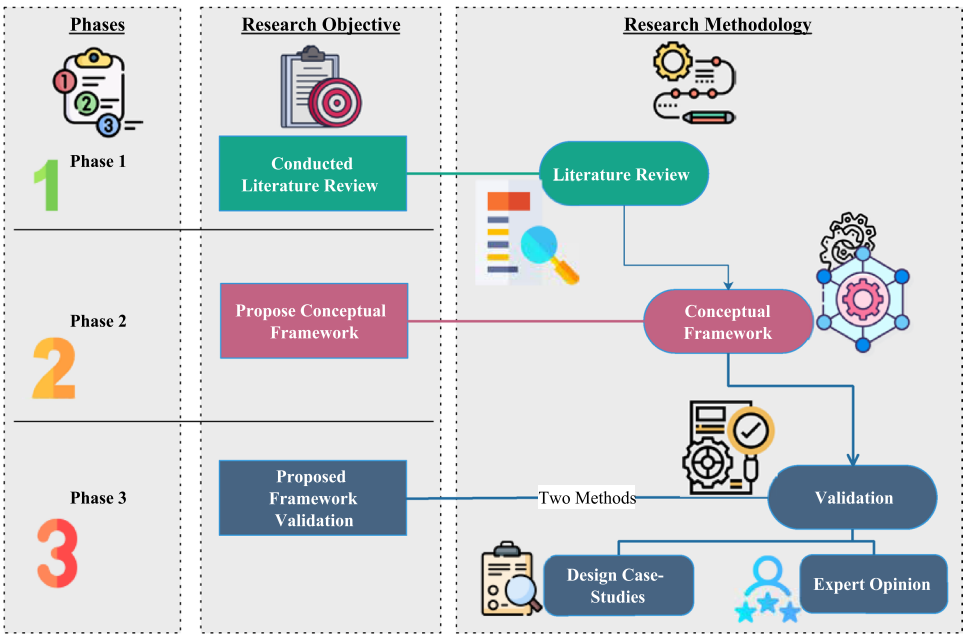


Fig. 3 Adopted research methodology

4 Proposed framework

This section provides a detailed discussion of the high-level view of the proposed framework, a detailed view of the framework, workflow steps, and general rules for UI adaptation.

4.1 High-level view of proposed framework

The proposed framework is divided into (i) a managed and (ii) a managing System. The process of adaptation is performed in the Managing System, such as data collection, performing analysis, generating plans, and so on. At the same time, the displayed UI is a part of the Managed System, and the sensors can be considered the intermediate between these two parts.

Figure 4 represents the high-level view of the proposed framework with the sequence of steps. Firstly, the Monitor module (step 1) is for context monitoring. The Analyzer (step 2) is the second module used to perform analysis. The third module is Planner (step 3) for generating plans. The fourth and fifth modules are adaptation Controller (step 4) and User Validation/Verification (step 5). The Executor module (step 6) is the sixth, and Knowledge (step 7) is the seventh module in which all the information about modules and adaptation is stored.

4.2 Detailed view of proposed framework

Modules in the proposed framework are: (i) Monitor Module, (ii) Analyzer Module, (iii) Planner Module, (iv) Adaptation Controller Module, (v) User Validation/Verification Module, (vi) Executer Module, and (vii) Knowledge Module. All the modules, submodules, and flow of the proposed framework are shown in Fig. 5. In addition, a detailed description of each module of the proposed framework is provided in Table 2.

4.3 Algorithms of proposed framework

From the reported literature, the authors cannot find any algorithm or pseudocode proposed in the Android SAUI context. As a result, the authors of this study also proposed core algorithms for various modules. The proposed framework comprises seven distinct components. Hence, to visually depict the core algorithms of each module separately, the authors followed the conventional approach of utilizing component diagrams, as shown in Fig. 6.

Moreover, the algorithmic details of all modules are discussed below:

4.3.1 Monitor Module Algorithm

The "Monitor Module Algorithm" takes input from a set of sensors denoted as *S*. Initially, all sensor values are set to 0. The algorithm then proceeds to identify sensor parameters labeled as *P*. Inside a while loop that continues as long as the user interacts with a phone user interface (UI), the sensors in set *S* collect multiple values of *P*. The algorithm further monitors the data environment and saves the collected data to a Knowledge Module.

The algorithm generally represents continuously collecting and monitoring sensor data based on user interactions with a phone's user interface and storing this data in a knowledge module.

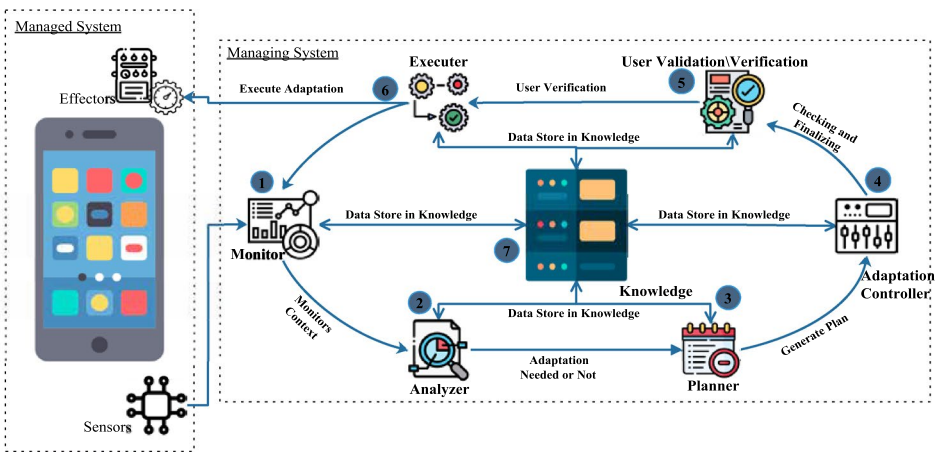


Fig. 4 High-level view of the proposed framework

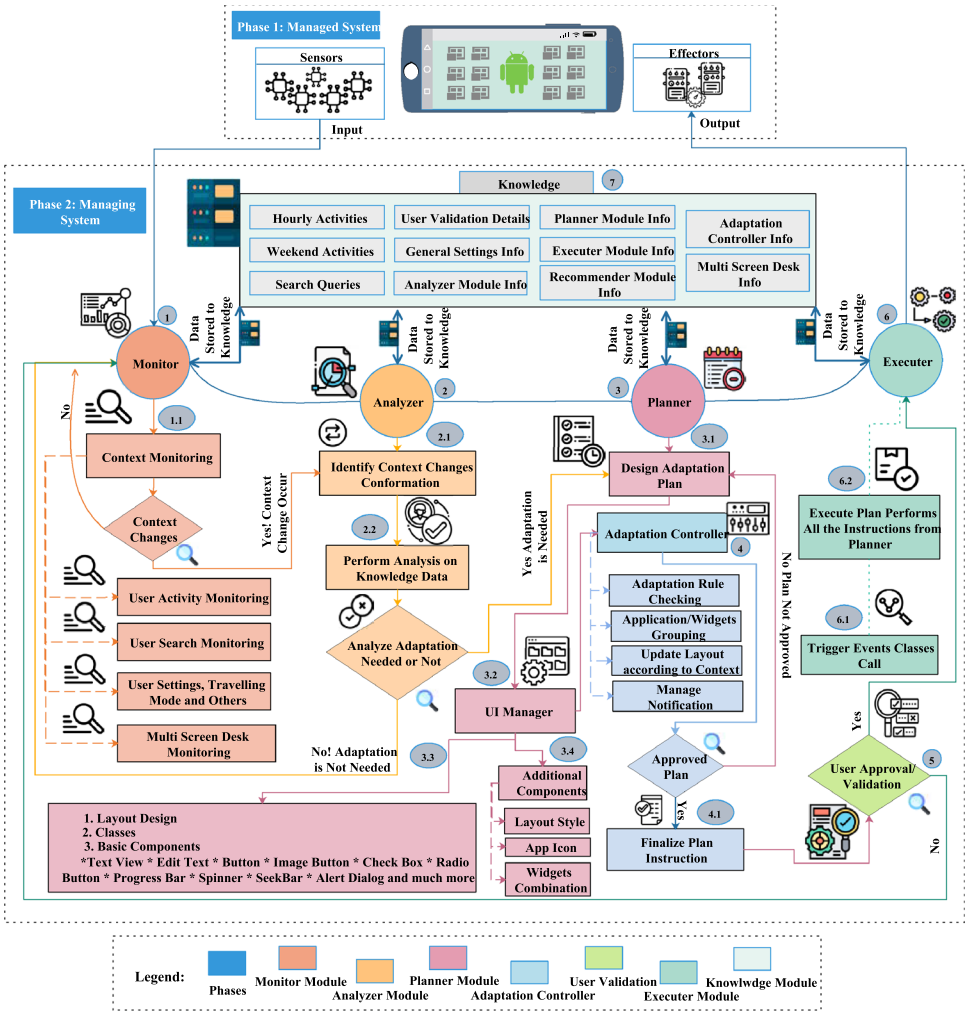


Fig. 5 Detailed view of the proposed framework

Algorithm 1 Monitor Module Algorithm

```
/* Dataset of user based sensors                                    */
+Data: Set of sensors S
/* Set all the sensors value to 0                                   */
+S = 0
  +Initially : Identification of sensor parameters P
  /* Now this is a While loop                                       */
while User interacts with phone UI do
  | +S collects multiple P value
  |   +Monitor data environment
  |   +Save data to KnowledgeModule
end
```

4.3.2 Analyzer Module Algorithm

The "Analyzer Module Algorithm" begins by selecting effective parameters denoted as $P*$, using some machine learning-based techniques (i.e., LSTM, SVM, NLP, and other Reinforcement Learning Gheibi et al. 2021). Once the parameters are selected, the algorithm performs an analysis of them. After that, the algorithm evaluates the analysis results, which can be "YES" or "NO." If the analysis results are "YES," it confirms adaptation is required

Table 2 Detail description of modules in the proposed framework

Sr.	Module name	Details
1	Monitor module	The monitor is the first module (Fig. 5, step 1), and the monitor's purpose is context monitoring (Fig. 5, step 1.1). Initially, the framework monitors for seven days as the daily smartphone usage differs from the weekend usage. This module monitors the context changes like hourly activities (which application user prefers in which hour) and weekend activities (monitor type of application usage and activities performed on weekends) and stores all the data in the Knowledge module. It also monitors the different search queries users commonly search on their smartphones. It will help in item recommendation/suggestions through notifications. Moreover, this module also monitors hourly general and advanced settings and the user traveling information to activate the Travelling Reader Mode accordingly. Furthermore, the concept of the multi-screen desk helps the user to share their phone with others in a controllable usage context, and the monitor module also monitors it. All the monitoring module data is collected through the sensors (Lee et al. 2011) and stored in the Knowledge module
2	Analyzer module	The analyzer is the second module of the framework (Fig. 5, step 2) which identifies the context change confirmation (Fig. 5, step 2.1). The main purpose of this module is to finalize whether the adaptation is needed by analyzing the data stored in the Knowledge module (Fig. 5, step 2.2). If the analyzer module accepts the need for adaptation by analyzing the gathered context monitor data, it proceeds to the Planner module
3	Planner module	The third is the Planner module (Fig. 5, Step 3) which generates the plan (Fig. 5, Step 3.1) for the interface adaptation through the UI Manager (Fig. 5, Step 3.2). UI Manager consists of the traditional layouts and basic components (Fig. 5, step 3.3) with additional components like layout style, application icon, and widgets combination folder/block (Fig. 5, step 3.4). The Planner module generates a planner and sends it to the Adaptation Controller module for further processing
4	Adaptation controller module	The Adaptation Controller is the fourth module (Fig. 5, step 4) and checks the rules designed for the proposed framework. It also verifies the exact widgets grouping, layout design, recommendation notification, and the finalized plan instructions (Fig. 5, step 4.1)
5	User Validation/Verification Module	User Validation/Verification is the fifth module of the proposed framework (Fig. 5, step 5). After finalizing all the tasks, this final plan is sent to the user according to its selected/specified time slot. A checklist of all the new and previous adaptations is shown to the user dividing them into two sections. The first section of the checklist is for the latest, while the second section is for the previous adaptation details. Users may check or uncheck (undo or redo) any new or old adaptation according to their interest and finally approve it
6	Executer Module	The sixth module is the Executer (Fig. 5, step 6). After the user verification of the planned SAUI, the next step is executing that particular plan through a set of event class calls of the system (Fig. 5, step 6.1). The executor module then sends this information to the mobile phone's effectors, and a complete SAUI process is done (Fig. 5, step 6.2)
7	Knowledge Module	This module maintains all the information and data related to hourly activities, weekend activities, search queries, user validation details, general information settings, analyzer module information, planner module information, executer module information, recommendation notification information, adaptation controller module information, and multi-screen desk monitoring information (Fig. 5, step 7). Moreover, if the user wants to switch the SAUI adaptation to turn it able or disable it, the user has the authority to do so. Furthermore, on turning ON the self-adaptive user interface adaptation option, the framework will provide all of the basic guidelines to the user relevant to the framework or provide a trial mode to understand the working of the proposed framework better

and proceeds to save data to the Knowledge Module. However, if the results are "NO," the algorithm instructs a jump back to the Monitor Module for further monitoring.

In general, the algorithm appears to be a part of a process where effective parameter selection and analysis play a crucial role in decision-making for the adaptation.

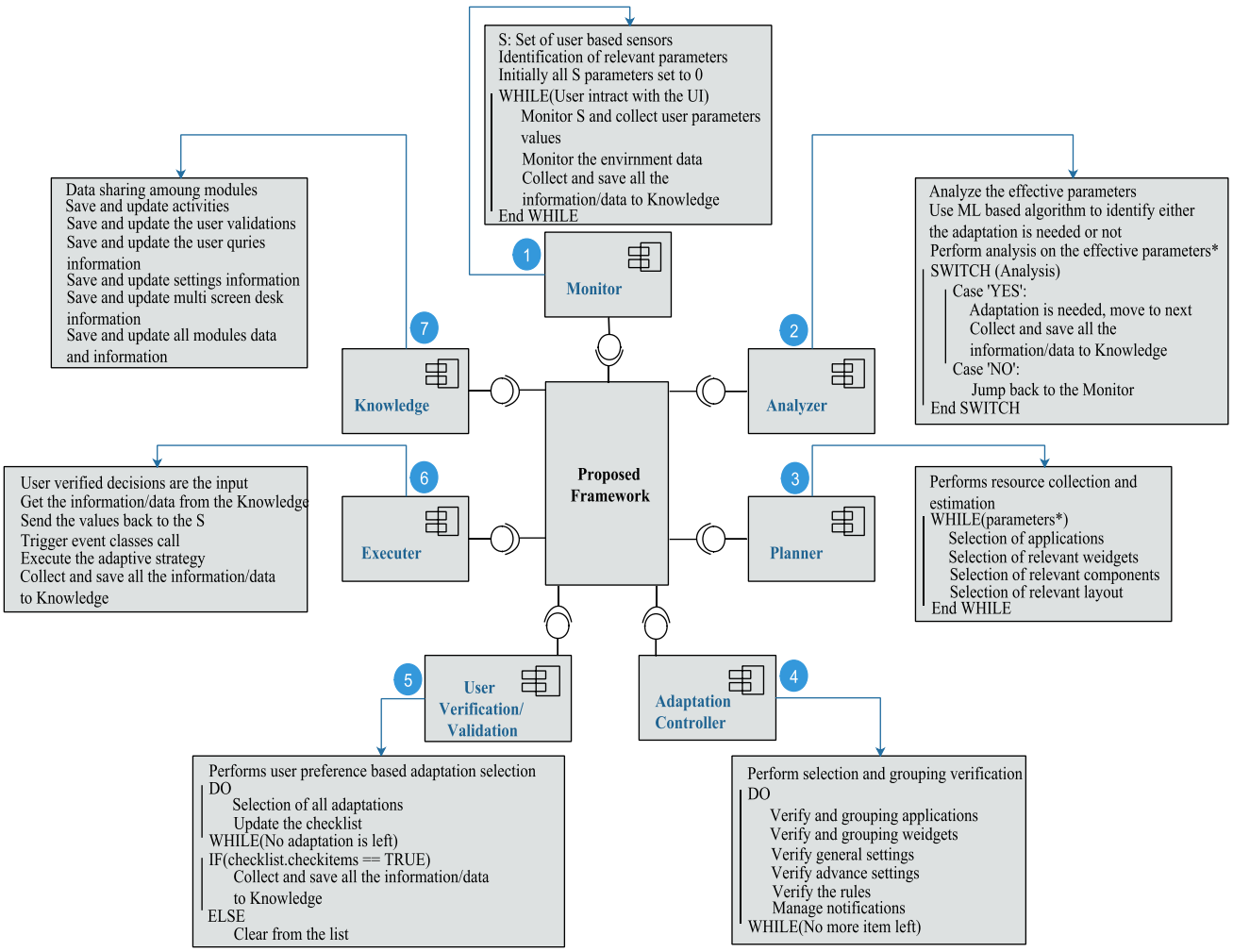


Fig. 6 Proposed conceptual framework flow, core algorithms based on the component diagram

Algorithm 2 Analyzer Module Algorithm

```
/* Selection of effective parameters using ML                    */
Selection of effective parameters P*
  /* Perform analysis on selected effective parameters           */
P* analysis = DONE
 switch Analysis results (YES/NO) do
    case YES do
       Adaptation confirmation = True
        Save data to the Knowledge Module
    end
    case NO do
       Jump back to the Monitor Module
    end
end
```

4.3.3 Planner Module Algorithm

The "Planner Module Algorithm" is designed to collect and estimate device resources. It begins with the collection of device resources. Then, the algorithm enters into a loop while P^* is true. This loop performs a series of steps, including selecting applications, relevant widgets, components, and layouts.

In general, algorithm plays a vital role in planning and resource management.

Algorithm 3 Planner Algorithm

```
/* Resources collection and estimation                                    */
+ Collection of device resources = DONE
 while P* do
  | + Selection of applications
  |  + Selection of relevant widgets
  |  + Selection of relevant components
  |  + Selection of relevant layouts
end
```

4.3.4 Adaptation Controller Module Algorithm

The "Adaptation Controller Module Algorithm" outlines a procedure for making final selections and grouping verification of device resources. The process is executed in a loop until no items are left to process. Various tasks are performed within the loop, including verifying and grouping applications and widgets, checking general and advanced settings, confirming rules, and managing notifications.

The algorithm ensures proper adaptation by systematically examining and categorizing various components and settings.

Algorithm 4 Adaptation Controller Module Algorithm

```
/* Perform final selection and grouping verifications                     */
while Item left = FALSE do
  | + Verifying and grouping applications
  |  + Verifying and grouping widgets
  |  + Verify general settings
  |  + Verify advance settings
  |  + Verify the rules
  |  + Manage notifications
end
```

4.3.5 User verification/validation Module Algorithm

Initially, the user preference-based selection is started. Subsequently, it enters a "do-while" loop that repeats the following steps as long as the condition "Adaptation left = FALSE" holds. Inside the loop, it selects the user-based adaptation and updates the checklist.

Following the loop, an "if-else" statement, if the condition "checklist.checkitems == TRUE" evaluates to true, the algorithm collects and saves data into the Knowledge Module. Conversely, the algorithm removes items from the list if the condition is false.

In general, the algorithm represents user verification and validation. It leverages user preferences and a checklist to govern the adaptation process.

Algorithm 5 User verification/validation Module Algorithm

```
/* Performing user selection-based adaptation                    */
+ user preference based selection = started
 while Adaptation left = FALSE do
 | + Selection of user-based all adaptation
 |  + Updating the checklist
end
if checklist.checkitems == TRUE then
 | + Collect and save data into Knowledge
end
else
 | + Clear from the list
end
```

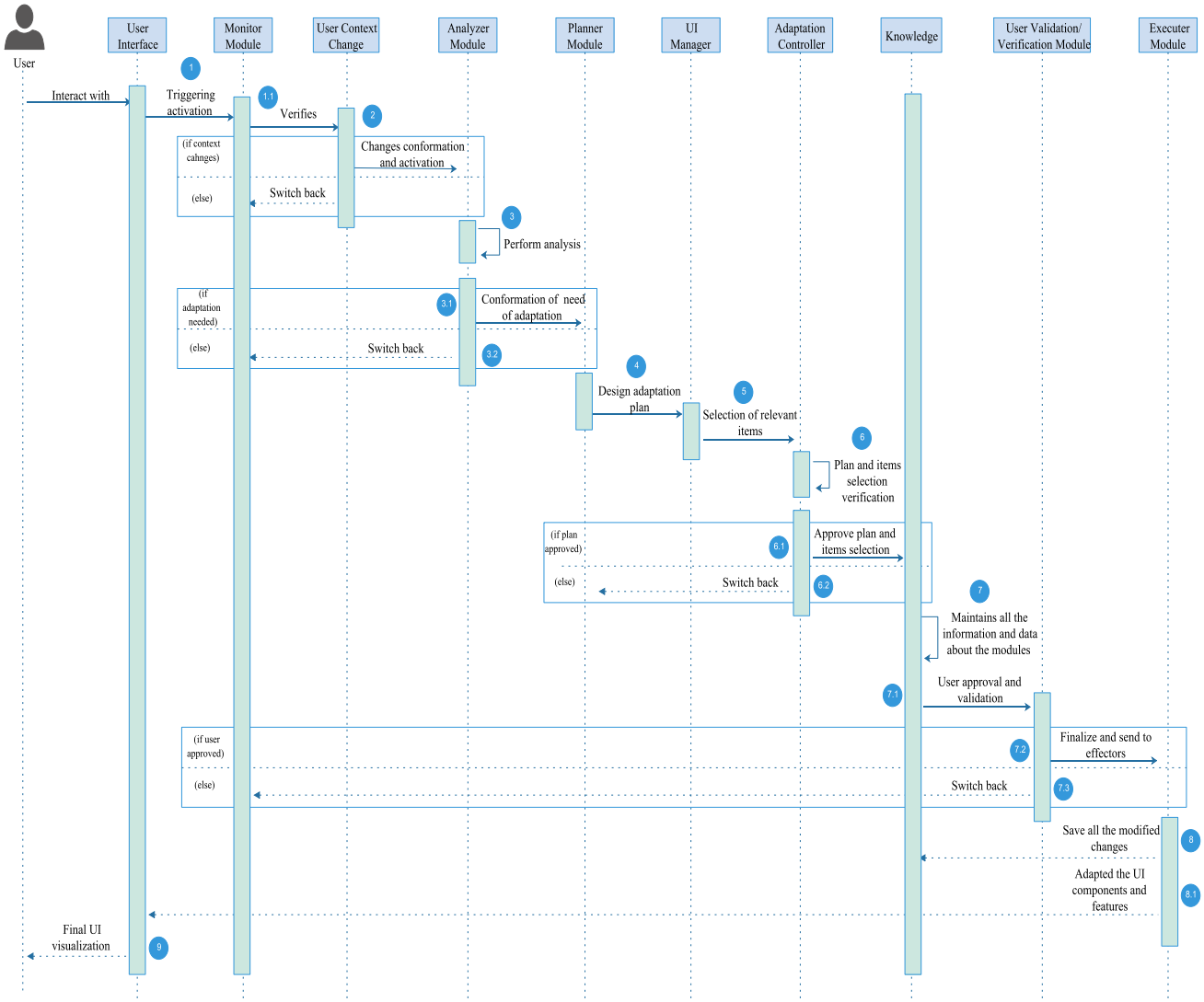


Fig. 7 Sequence diagram representing the proposed framework

4.3.6 Executer Module Algorithm

The "Executer Module Algorithm" performs a series of tasks. It begins by receiving user verification decisions, denoted as D. Then, it retrieves information from the Knowledge Module and sends these values back to S."

Following this, it triggers relevant events and classes and also executes an adaptive strategy. Finally, it concludes by collecting and saving the most recent data into the Knowledge Module, presumably to update or maintain knowledge data.

Algorithm 6 Executer Module Algorithm

```
/* Performing the expectations                                          */
+Data: User verification's decisions D
+Getting the relevant data from the Knowledge Module
 +Sending the values back to S
 +Triggering the relevant events and classes = Started Executed
 +Execute the adaptive strategy
 +Collecting and saving the latest data into the Knowledge module
```

Table 3 UI adaptation rules

Rule	Title	Description	Type
Rule-1	Basic tutorial UI	Basic and additional SAUI information and navigation information are shown if a first-time user wants to learn about the framework	Basic Task Features
Rule-2	Low light	Whenever the surrounding lighting condition is low, the user prefers a different light scheme. Then change the UI color scheme and contrast to improve readability in the dark	Layout
Rule-3	Normal light	If the surrounding lighting condition is low and the user prefers a different mobile light scheme. Then change the UI color scheme and contrast to default mode	Layout
Rule-4	High light	Similarly, when the surrounding lighting condition is high, and the user prefers a different light scheme than normal conditions, then improve the readability by increasing the UI contrast	Layout
Rule-5	Day mode	As the day begins (clock time), all UI elements are displayed according to user usage context in the day mode layout	Layout
Rule-6	Night mode	When the day is over (clock time), all UI elements are displayed in the night mode style based on the usage context	Layout
Rule-7	Vision problem	The font size of the UI elements will be adjusted based on the user context if the user has vision issues	Layout
Rule-8	Battery low	To save energy, the UI is shown in a black and white color scheme when the mobile's battery level is low	Layout
Rule-9	Normal battery	Rule "Battery Low" are deactivated when the mobile's battery level is normal	Deactivation
Rule-10	SAUI on option	The self-adaptive user interface option can be turned on by the user	Activation
Rule-11	SAUI off option	The self-adaptive user interface option can be turned off by the user	Deactivation
Rule-12	Traveling mode on	When the user travels (accelerometer), increase the font and color contrast, making it easy to read and interact with the system	Layout
Rule-13	Traveling mode off	When the user travels (accelerometer) and does not prefer the reader mode, switch it off until the user allows it	Deactivation
Rule-14	Multi-desk switching	Users can switch from one screen desk to another	Layout
Rule-15	User feedback	The adaptation verification list is based on user feedback they prefer	Verification
Rule-16	Previous adaptation update	The user will make changes to the adaptation list that they previously verified	Verification

4.3.7 Knowledge Module Algorithm

The "Knowledge Module Algorithm" outlines a procedure for managing and maintaining data for future adaptations. This algorithm involves several key tasks, including data sharing among different modules, saving and updating activities, user preference-based validations, user query information, setting information, multi-screen desk information, and all modules' data and information. This process seems to be essential for knowledge management within, ensuring that data is saved and updated as needed to support various functionalities and user interactions.

The algorithm generally describes the process or operations of saving and updating various data types for future adaptations.

Algorithm 7 Knowledge Module Algorithm

```
/* Saving and updating the latest data for future adaptations */
+ Data sharing among the different modules
 + Save and update activities
 + Save and update the user preference-based validations
 + Save and update the user query information
 + Save and update the setting information
 + Save and update the multi-screen desk information
 + Save and update all modules' data and information
```

4.4 Workflow of the proposed framework

For a better understanding of the proposed framework, there is a need for a diagram in which the flow of the framework can be explained with the sequence of steps. With the help of the sequence diagram, the flow of the proposed framework is shown in Fig. 7, which provides a complete workflow of the framework step by step. When the user interacts with the smartphone user interface, it triggers the activation of the Monitor module (Fig. 7, step 1) and verifies the user context change (Fig. 7, step 1.1). Based on the user context changes (Fig. 7, step 2), it is to decide either to go to the Analyzer module (Fig. 7, step 2.1) or switch back to the Monitor (Fig. 7, step 2.2). The Analyzer analyzes the gathered data (Fig. 7, step 3) to decide whether the UI adaptation is needed. If there is a need for the UI adaptation based on the performed analysis, then switch to the next module (Fig. 7, step 3.1); otherwise, switch back to the Monitor module (Fig. 7, step 3.2).

On the confirmation of the need for the UI adaptation, the next step is the planning to design the adaptation plan (Fig. 7, step 4). Based on the adaptation plan, the next step is selecting the relevant items for UI adaptation (Fig. 7, step 5). The purpose of the Adaptation Controller is to check and verify the selected items (Fig. 7, step 6). With the approval, all the information is saved to the Knowledge (Fig. 7, step 6.1), and on the rejection, it switches back to the Planner to redesign the appropriate plan (Fig. 7, step 6.2). The Knowledge module maintains all the modules' information and data (Fig. 7, step 7). After the approval, the plan is sent to the user for verification (Fig. 7, step 7.1). On user approval, the updated plan is sent to the effectors (Fig. 7, step 7.2). Based on the rejection, the framework switches back to the Monitor module (Fig. 7, step 7.3). All the updated information is also stored in the Knowledge (Fig. 7, step 8). UI components and features are adapted and updated accordingly (Fig. 7, step 8.1). Finally, the user has intelligently updated the UI based on interest and satisfaction (Fig. 7, step 9).

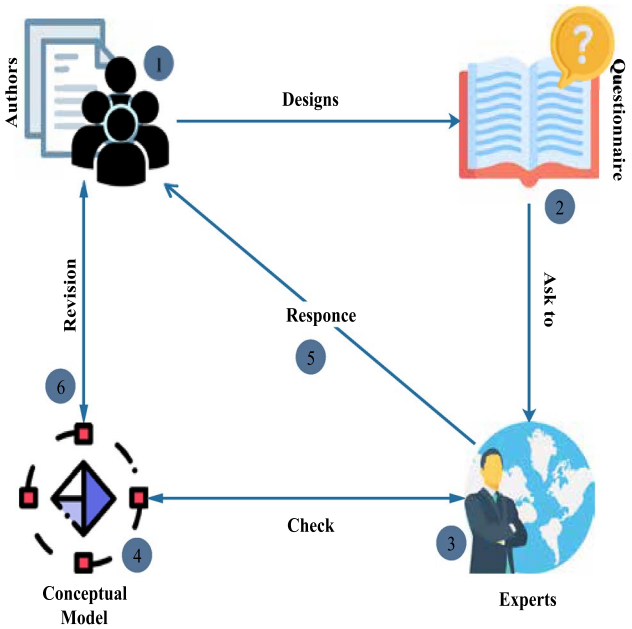


Fig. 8 Expert-based validation technique process

4.5 General rules for UI adaptation

The rules provide control over adaptive behavior, and for efficient working of any framework, the rules must be evaluated at run time based on the latest usage context information. Rules should be defined in a universal and portable representation dynamically loaded at run time from various sources (Yigitbas et al. 2019). Inspired by Yigitbas et al. (2019), the authors also introduced 16 UI adaptation rules to cover various adaptation dimensions by dividing them into four main adaptation strategies. The strategies are:

(i) Basic Task Features is about the overall tutorial for the user to give complete information to the user about how the system is going to perform work,
(ii) Activation/Deactivation refers to the activation of features as well as the feature's deactivation,
(iii) The layout allows UI changes to adjust the appearance of the UI, (iv) Verification is based on the user feedback (user want to redo or undo).

UI adaptation rules that cover various adaptation dimensions are discussed in Table 3.

5 Evaluation and validation criteria

This section is based on the discussion of the evaluation criteria of the proposed framework. Moreover, validation criteria for the proposed framework are briefly discussed in the same section.

5.1 Evaluation criteria

For the evaluation of the proposed model or framework, several studies have designed their case studies (Yigitbas et al. 2019; Machado et al. 2018; Wesson et al. 2010; Yigitbas et al. 2017; Khan et al. 2018; Yigitbas et al. 2019). Similarly, three case studies have been designed and developed to evaluate the framework. The first case study is based on daily activities, while the second is based on the weekly activities of the user. Both case studies represent the feasibility and effectiveness of the framework.

5.2 Validation criteria

The framework is validated by industrial experts, who may help determine which additional changes are required to

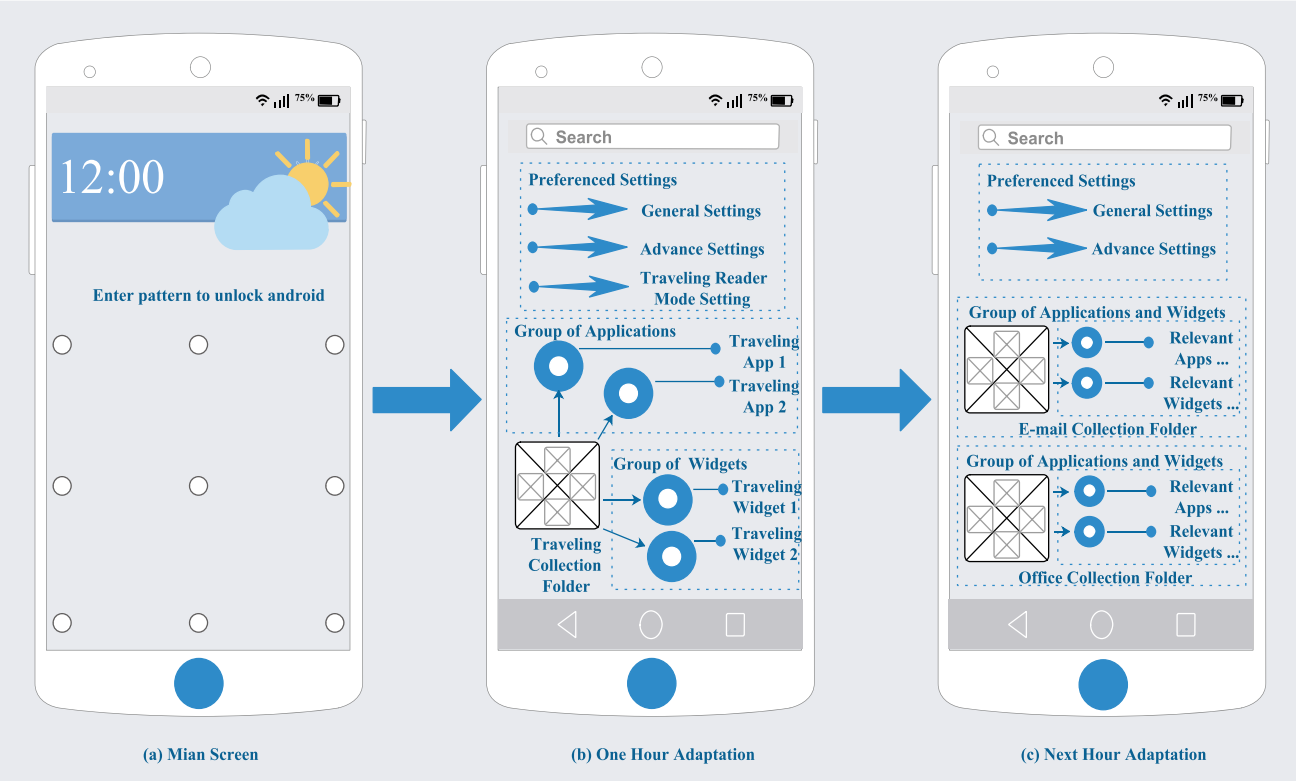


Fig. 9 Proposed framework evaluation based on the case study 1

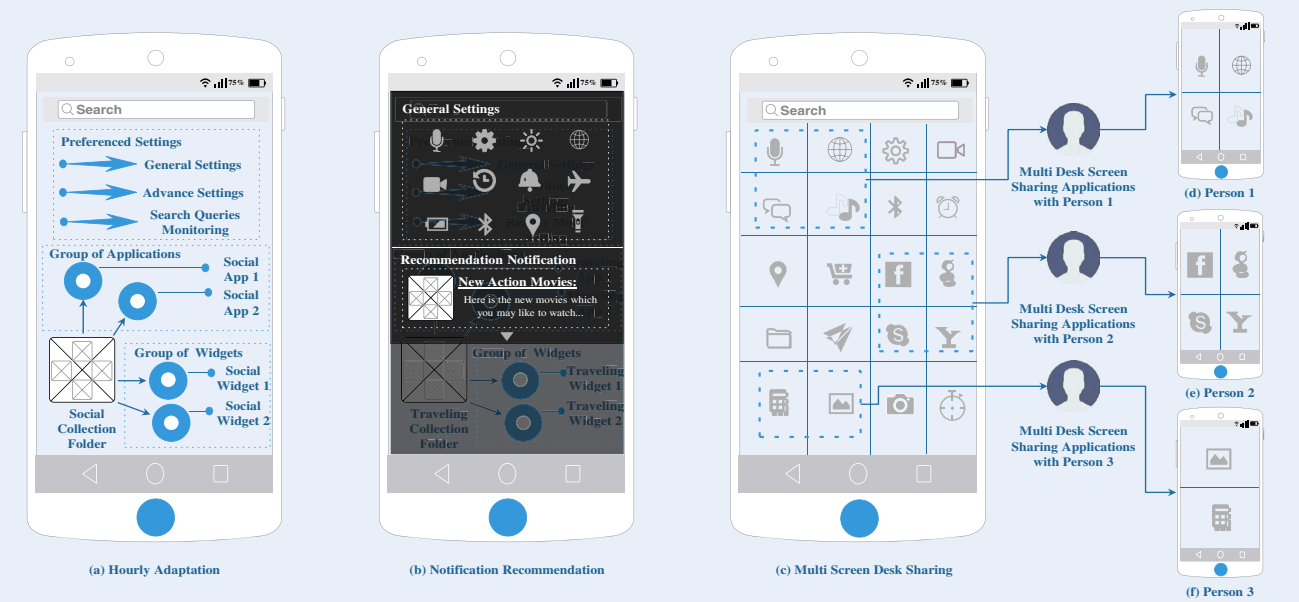


Fig. 10 Proposed framework evaluation based on the case study 2

improve the framework. We designed the criteria to figure out the valid experts, and the criteria are:

- The experts must be working as "Android Developers/ Development."
- Experts must have at least three years of experience in the targeted domain.
- Experts must be from different regions.

A questionnaire was designed and shared with the identified experts. The following link provides the questionnaire used for expert-based validation: https://forms.gle/c2WWDCnqFw7mz3GW7. Moreover, complete details about the experts are available on the provided link below: The flow of the adopted methodology for the expert-based validation technique is shown in Fig. 8. First, the authors designed the questions for the questionnaire (step 1). Then, the questionnaire was distributed among the identified experts (step 2). Next, the experts read the questionnaire (step 3), validated the framework (step 4), and provided their feedback to the authors (step 5). Finally, based on the comments and suggestions of the experts, the authors modified the framework (step 6).

6 Results and discussions

Several studies evaluate their proposed models or frameworks by designing case studies in the literature. The authors of this study are not only designing case studies for evaluation, but we also validate the proposed framework through an expert-based validation technique.

6.1 State-of-the-art: case studies based (RQ-1)

This section aims to investigate the performance of the proposed framework by designing and developing different case studies. In other words, this section focuses on answering the devised RQ-1.

Three case studies have been designed and developed. The first case study is based on daily context monitoring, the second is based on weekend context monitoring, and the third is focused on emergency situations. The following sections provide a brief description of the considered case studies:

- Case Study 1: Based on daily activities

  Case study 1 is based on an example scenario by considering a user as a job holder who goes to the office daily. The user prefers an online transportation/traveling service (such as an Uber application) to book a ride from home to work in the morning. As a result, traveling-based applications take precedence over the rest of the applications installed on the user's smartphone at that particular hour. As the widgets are separately available from the application, sometimes it is difficult for the user to search out a specific widget properly and timely. Providing all the relevant widgets of particular applications for an exact hour in a separate folder/block helps the user to search out and perform their tasks quickly and easily and also helps manage time efficiently.

  Therefore, the proposed framework will prioritize all traveling applications and relevant widgets to travel applications in a separate folder. Moreover, the frame-

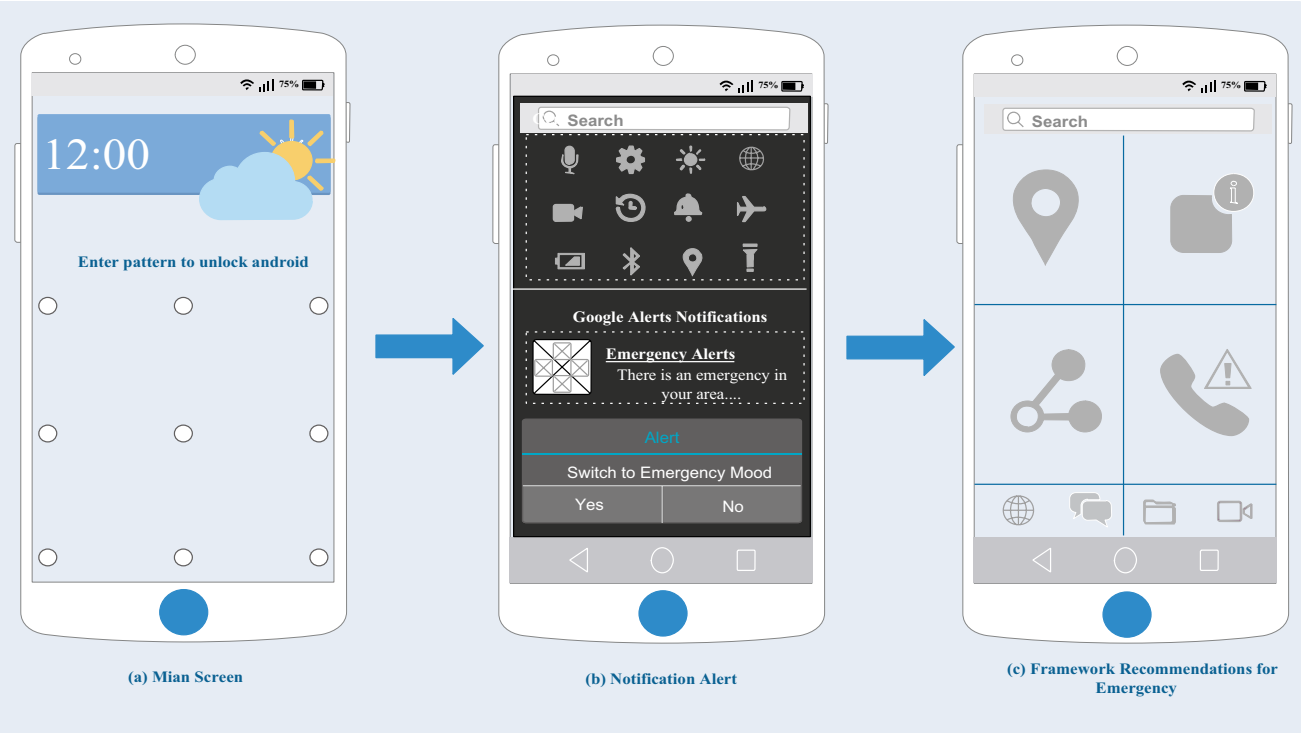


Fig. 11 Proposed framework evaluation based on the case study 3

work will also manage the smartphone's general and advanced settings, such as profile mode switching, activation of the traveling reading mode, network connectivity, screen brightness, volume level, screen layout, and so on, as shown in Fig. 9b.

Similarly, on the arrival of the user at the office, the first thing the user typically does on their smartphone is check for new emails and use other office-related applications. As a result, the priority of the traveling application is lower than that of the email applications and other office-related applications at that particular hour(s). For such hour(s), the framework intelligently updates the system's layout. It provides email-related applications with all relevant widgets in a separate folder and other office-based applications in a different folder. Moreover, the framework manages the rest of the smartphone settings accordingly, as shown in Fig. 9c. This case study shows how effectively and efficiently the system is adapted on an hourly basis, and the adaptation is based on the user behavior and its usage context.

- Case Study 2: Based on the weekend activities

  Case study 2 is based on a hypothetical example, as the weekend activities differ slightly from the daily routine. Therefore, the framework monitors the weekend activities and works accordingly. On weekends, social application usage increases compared to the daily routine. So, such applications have higher priority than email or office-based applications. The system will update the UI hourly, as shown in Fig. 10a, where the group of all social applications and their relevant widgets are shown in a single folder based on the user's hourly preference.

Table 4 Framework design validation

Expert ID	Five different categories				
–	Very poor	Poor	Fair	Good	Excellent
E1	–	–	–	Yes	–
E2	–	–	Yes	–	–
E3	–	–	–	–	Yes
E4	–	–	–	–	Yes
E5	–	–	–	Yes	–
E6	–	–	–	–	Yes
E7	–	–	–	Yes	–
E8	–	–	Yes	–	–
	0 Expert	0 Expert	2 Experts	3 Experts	3 Experts

In addition, monitoring search queries is a part of the proposed framework, which helps provide relevant item recommendations to the user through notifications. Considering the user prefers watching movies on weekends. So, the framework also recommends the latest movies according to their interest by notifying the user (Fig. 10b).

Moreover, user can sometimes share their smartphone with others (kids, friends, and so on) who have complete access to the device. The idea of multi-screen desk sharing is based on sharing a smartphone under user-defined usage control conditions and restrictions. As shown in Fig. 10c, the user can share their smartphone device using a multi-screen desk option, depending on which application they want to share with others (sharing your smartphone with kids or other colleagues). For example, consider a user who wants to share the device with three different people (Person 1, Person 2, and Person 3), as seen in Fig. 10d–f. Instead of sharing the complete device, the user creates a new screen desk, allowing Person 1 to use only four applications (Fig. 10d). In addition, the user sets up a different screen desk for Person 2 and restricts him to use four different applications (Fig. 10e). Similarly, the user creates another screen desk for Person 3 so that they offer to use just two applications for that particular person (Fig. 10f). Furthermore, the second case study demonstrates how efficiently and effectively the device UI is adapted based on the user's weekly behavior and usage context. Moreover, the second case study shows how the user can easily manage the mobile phone to share with others.

Table 5 Module's relevance information

Expert ID	Different categories		
–	All are relevant	Some may be not relevant	Some are not relevant
E1	–	Yes	–
E2	–	Yes	–
E3	Yes	–	–
E4	Yes	-	–
E5	Yes	-	-
E6	Yes	-	–
E7	–	Yes	–
E8	Yes	–	–
	5 Experts	3 Experts	0 Expert

- Case Study 3: Based on the emergency situations

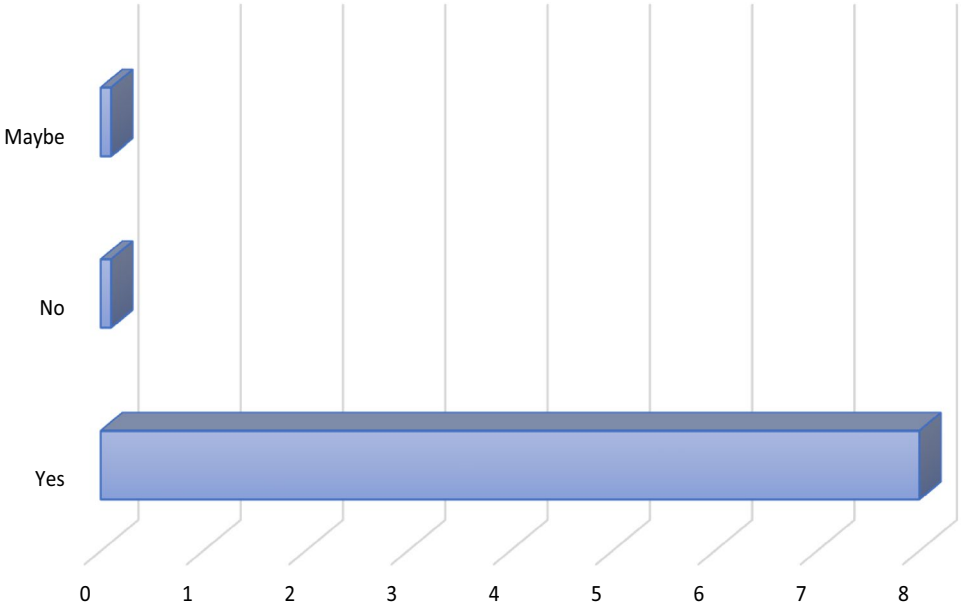
  Case study 3 is based on an example scenario considering emergencies such as earthquakes, floods, etc. Google develops the Android operating system, and Android is providing Google alerts for earthquakes. Considering the user is in an area prone to natural disasters. In this setting, staying informed and ready to respond quickly is essential for personal safety. The proposed framework monitors emergency alerts. When Google Alerts flag a significant event, the framework springs into action, ensuring the user receives timely, life-saving information (as shown in Fig. 11b). When a notification alert appears, if the user allows switching to emergency mode, the interface will be updated to a simplified one-tap access based on the user's preference (Fig. 11c).

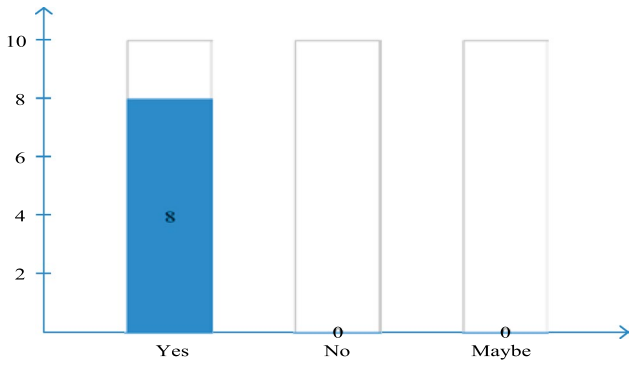
  The interface is transformed to prominently display details about the detected disaster, including its location and severity. Additionally, interactive maps are presented, offering evacuation routes and locations of nearby emergency shelters. Communication applications

Table 6 Evaluation of concepts

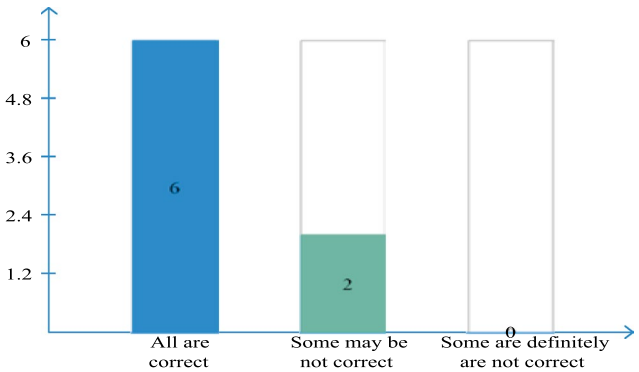
Expert ID	Multiple concept	Five different categories				
–	–	Very poor	Poor	Fair	Good	Excellent
E1	Multi-screen desk	–	–	✓	–	–
	Apps and widgets combo	–	–	–	✓	–
E2	Multi-screen desk	–	–	✓	–	–
	Apps and widgets combo	–	–	–	✓	–
E3	Multi-screen desk	–	–	–	–	✓
	Apps and widgets combo	–	–	–	–	✓
E4	Multi-screen desk	–	–	–	–	✓
	Apps and widgets combo	–	–	–	–	✓
E5	Multi-screen desk	–	–	✓	–	–
	Apps and widgets combo	–	–	–	✓	–
E6	Multi-screen desk	✓	–	–	–	–
	Apps and widgets combo	–	–	–	–	✓
E7	Multi-screen desk	–	–	✓	–	–
	Apps and widgets combo	–	–	–	✓	–
E8	Multi-screen desk	–	–	✓	–	
	Apps and widgets combo	–	–	–	✓	–
Total	Multi-screen desk	1 Expert	0 Expert	5 Experts	0 Expert	2 Experts
	Apps and widgets combo	0 Expert	0 Expert	0 Expert	5 Expert	3 Experts

Fig. 12 Readability validation of the framework

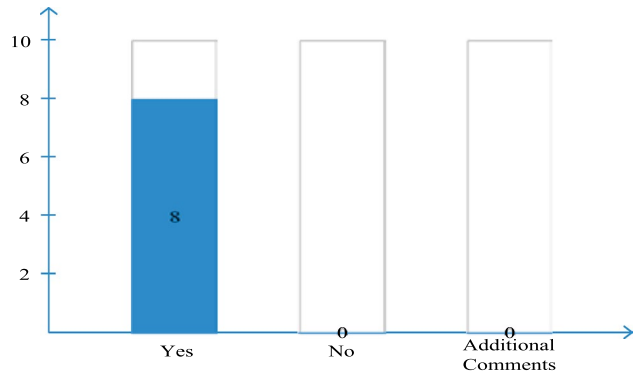




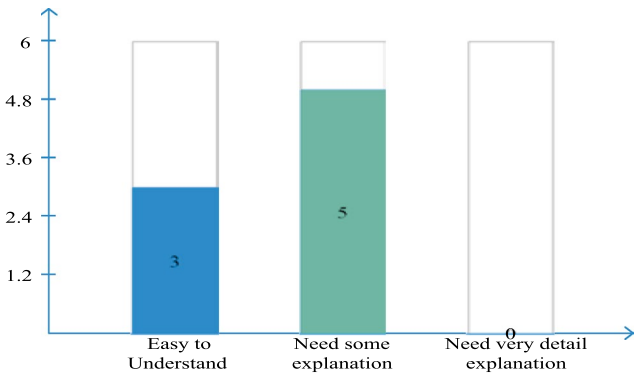
(a) Logical relationships and flows between the modules



(c) Correctnessof identified module



(b) Order of modules correct or need to changed



(d) Understandability of proposed conceptual framework

Fig. 13 Validation of correctness, logic, understandability, and ordering of modules

take the front seat, with one-tap access to messaging, location-sharing features, and emergency broadcasts. This empowers the user to instantly coordinate with family and friends, request assistance, and remain informed about the situation as it unfolds.

By combining the framework's capabilities with Google Alerts, users are better prepared and equipped

with the most up-to-date information, enhancing their safety and resilience during natural disasters. The integration of real-time alerts from Google ensures that users are well-prepared and able to make critical decisions in a time-sensitive crisis situation.

6.2 State-of-the-practice: expert-based validation technique (RQ-2)

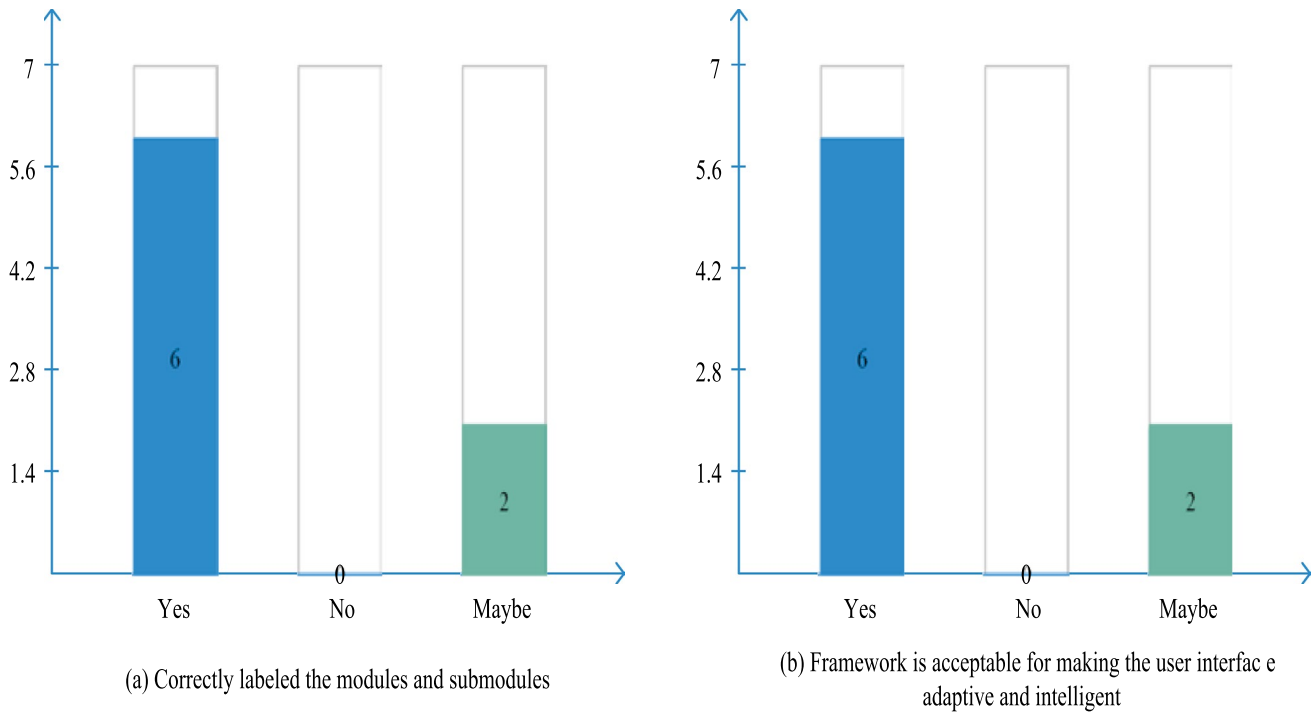
This section focuses on answering the devised RQ-2 to investigate the effectiveness of the proposed framework by validating industrial experts and getting their feedback regarding the framework.

A detailed questionnaire is designed and shared with the experts. There are seven critical sections of the developed questionnaire. The first section is based on basic information to determine whether the selected expert is the right person to validate the framework. The framework diagram and design are validated in the second section. The framework's readability is discussed in the third section. The relevance of all the framework modules was questioned in the fourth section of the questionnaire. The fifth section focused on the correctness of flows and the relationships between modules and sub-modules. The sixth section of the questionnaire asks for any missing information or details in the framework. Finally, the seventh section focuses on module labeling and proposed framework acceptance. Additional suggestions and comments were optional for every expert to fill in. All the sections of the designed questionnaire are discussed as follows:

- Basic information of experts
  Basic information (name, place to live, years of experience, current job/position, education, and current company name) is provided in Appendix A.
- Design validation of framework
  Framework design is validated with five different categories (very poor, poor, fair, good, excellent) from the experts, and their responses are shown in Table 4.

- Framework readability validation
  The experts validate the readability of textual information from the framework diagram discussed in this section. For example, Fig. 12 shows that all the experts have no readability issue (all the experts select "Yes," which means that the textual information is entirely readable for them).

Furthermore, based on the comments of all experts, there is no readability issue; experts also provide their positive feedback on this question. For example, E4 stated that "Framework is very well written." Moreover, E5 suggests adding more details to the diagram, but the authors consider that adding more information makes the framework diagram complex and may confuse the reader. So, the authors provide



(a) Correctly labeled the modules and submodules

(b) Framework is acceptable for making the user interfac e adaptive and intelligent

Fig. 14 Correct labeling and framework acceptance

a detailed description of different modules and submodules based on the other step numbers defined in the proposed framework diagram (Fig. 5).

- Framework relevance validation

  The relevance of the information and concepts is discussed in this section. Based on the feedback from the experts regarding that particular question, Table 5 shows that the experts agreed that all the information related to modules, submodules, and other items presented in the framework diagram is relevant.

Furthermore, E2 reported, "I think arranging the module vertically is enough, and color coding them may not be needed." In addition, according to the E3, "Everything is clear and readable, good choices on colors and fonts and style in general." Moreover, E4 stated that "All modules have the relevant information." Finally, E1 suggested, "A little congested. Maybe make it wider and add white spaces in between". So, the authors agreed to provide different colors to different modules and their respective sub-modules to differentiate between them easily.

Furthermore, questions related to significant concepts were asked (multi-screen desk, app, widgets combo) to get expert feedback. Validation of the concepts is shown in Table 6.

It can be observed that the experts appreciate the proposed concepts of the framework. According to E3, "Excellent work." Moreover, E2 commented that "Sounds like a good idea." In addition, E1 stated the concepts; "Very relevant and useful feature I would like to have in my phone." E7 also appreciated by commenting, "This is a good idea."

- Framework correctness validation

  Correctness, logic, orders of different modules, and framework understandability are discussed in this section. Figure 13a is based on the logical correctness of the relationship and flows between the modules and submodules. All the experts approved that the relationship and flows between the modules and submodules are logical. Moreover, Fig. 13b is about the correctness of the order of the modules. It can be easily observed from Fig. 13b that all the experts agreed that the order of the modules and sub-modules is correct. Furthermore, Fig. 13c is based on the correctness of the module identification. The majority of the experts agreed that all modules are correctly identified. Finally, Fig. 13d is about the easy understandability of the framework. Some experts stated that the framework is easy to understand, while some suggest that there is a need to add some explanation. Hence, the authors decided to provide a descriptive explanation.

Moreover, E3 commented, "Excellent work." Similarly, E1 stated, "Not too much aware of the details." In addition, E5 suggested that "knowledge data will be saved locally on the machine, which will contain time series data and aggregated data. Will there be a module to save the preferences and data to the cloud". The authors appreciate this valuable suggestion that E5 makes and consider this suggestion as a future work to explore the workings of cloud computing in smartphones and the data synchronization process.

- Framework missing information

  This section is related to investigating any missing information in the proposed framework. E1 stated, "I think this is a good proposal." Moreover, according to E8, "It is sufficient." E3 commented, "Everything is covered sufficiently. Bravo!"'. Similarly, E7 reported that "This is fine." E4 also stated that "Information provided is sufficient." In addition, E2 confirmed that "Idea and design seem good."
- Acceptance and suggestions

  Correct labeling of the modules, sub-modules, and the overall acceptance of the framework is discussed in this section. Figure 14a is based on correctly labeling the modules, submodules, and other items. The experts agree on the correctness of labeling modules, sub-modules, and other items. Figure 14b states the acceptance of the framework. From Fig. 14b, it can be easily noticed that the majority of the experts choose "Yes" to accept the framework, which approves the correctness and effectiveness of the proposed framework to make the UI of the Android OS adaptive and intelligent.
- Additional suggestions and comments of experts

  Overall, experts appreciate and encourage the efforts behind the proposed framework. E1 stated, "Something I haven't seen in a long time. It's innovative and bold. Go ahead". Moreover, E3 suggests, "Just keep up the good work; your work is Fantastic."

7 Threats to validity

This section provides the validity threats related to the targeted research work.

Construct validity This threat is related to the fact that relevant studies for this research work are covered. To address this threat, a research protocol was adopted for selecting the relevant studies published in the previous few years. Different electronic databases were targeted for the selection of papers. Finally, the potentially relevant studies were selected for this research work.

External validity This threat derives from selecting studies that may or may not represent the entire study. The snowballing technique is applied for the addition of the relevant studies. The authors of this work looked into the references of the selected studies and tried to determine other relevant studies to mitigate this threat.

Internal validity This threat concerns the conceptual framework, as an intelligent, self-adaptive user interface based on the Android OS is presented. We evaluate the proposed framework by designing case studies to overcome this threat. Moreover, the proposed framework is validated through an expert-based validation technique.

Conclusion validity This threat concerns the conclusion aspects. To mitigate this threat, the proposed framework is evaluated through case studies and validated by industrial experts. Moreover, the valuable comments and suggestions of the experts regarding the different aspects of the proposed framework show the effectiveness of the framework.

8 Research implications

The research implication section can be viewed from the two facts: (i) State-of-the-Art and (ii) State-of-the-Practice.

8.1 State-of-the-art

The researchers have performed significant work in the domain of AUI in a software systems context. Furthermore, in recent studies, the researchers have shown their interest in a mobile-based application context to make the UI of mobile applications adaptive. However, less work is reported in the mobile application domain. Similarly, we presented an SAUI based on the Android OS to make the UI of the mobile device adaptive. It not only helps improve user interest and satisfaction but also improves the overall performance of the mobile device. In addition, the authors of this study also proposed the core algorithms of different modules of the proposed framework. So, this study provides a foundation for future research in the field of SAUI in a mobile OS-based context. Moreover, this study allows the researchers to improve the proposed core algorithms and contribute to research by making the UI more intelligent and adaptive.

8.2 State-of-the-practice

This conducted study not only facilitates the researchers but also facilitates the practitioners in different ways. This research guides the developers who wish to work in the domain of AUI in the mobile context. Moreover, a basic understanding of the development of the AUI is precisely described in this work. Furthermore, the concepts discussed in the proposed framework also help the developers develop adaptive mobile launchers for different mobile devices. In addition, core algorithms and the detailed view of the modules help develop and understand how other modules are interlinked and perform their tasks, respectively.

9 Conclusion and future work

This research proposes a conceptual framework for a self-adaptive user interface based on the Android OS. The framework helps improve user satisfaction and experience by making the mobile interface intelligent and effective. A self-adaptive model was identified from the reported literature, and new modules were added based on the requirements of the proposed framework. The authors also proposed core algorithms for the various modules of the framework. The proposed framework is evaluated through state-of-the-art (designing case studies) and validated through state-of-the-practice (expert-based validation technique). Three case studies were designed and developed, demonstrating how effectively and efficiently the device UI is adapted based on user behavior and usage context. Moreover, industrial experts from different regions were also identified. Finally, the proposed framework is shared with the identified experts for validation through a questionnaire based on several sections.

This research provides a new way for researchers to work on SAUI in the context of the Android OS. Moreover, this research work also provides a way for the researchers to operate and improve the core algorithms proposed in this research work.

In the future, we plan to enhance this work by exploring the monitor module's data handling. Moreover, we plan to identify which machine learning technique is better for such a data type, which helps identify the need for adaptation. Additionally, we also plan to improve the proposed algorithms. Moreover, the work performed by authors (Ahlström et al. 2020) can be considered for future work, as the authors proposed tracking the phone usage in traffic through the mobile application. It can be considered for adaptation in such a way that based on over usage while riding or driving, the smartphone will continuously notify and can make some UI updates which ultimately helps the user not to use the smartphone but to focus on the road.

Acknowledgements The authors thank the Software Reliability Engineering Group (SREG) members for their continuous support and feedback throughout this research. Moreover, they appreciate the experts who validated the proposed conceptual framework and provided valuable responses.

Author contributions All authors contributed to the study's conception and design. Material preparation, data collection, and analysis were performed by MA, SURK, AM, and AT. The first draft of the

manuscript was written by MA, and all authors commented on previous versions. All authors read and approved the final manuscript.

Funding Open access funding provided by Johannes Kepler University Linz. The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data availibility The datasets generated during and/or analyzed during the current study are not publicly available but are available from the corresponding author at reasonable request.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Consent for publication The authors have consented to the submission of the manuscript to the journal.

Ethics approval and consent to participate All participants provided written informed consent before enrolment in the study.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Ahlström C, Wachtmeister J, Nyman M, Nordenström A, Kircher K (2020) Using smartphone logging to gain insight about phone use in traffic. Cogn Technol Work 22(1):181–191. https://doi.org/10.1007/s10111-019-00547-6

Ali M, Khan SUR, Hussain S (2021) Self-adaptation in smartphone applications: current state-of-the-art techniques, challenges, and future directions. Data Knowl Eng 136:101929

Amoud M, Roudies O (2017) Dynamic adaptation and reconfiguration of security in mobile devices. In: 2017 International conference on cyber incident response, coordination, containment & control (Cyber Incident). IEEE, pp 1–6

Arcaini P, Riccobene E, Scandurra P (2015) Modeling and analyzing MAPE-K feedback loops for self-adaptation. In: 2015 IEEE/ACM 10th international symposium on software engineering for adaptive and self-managing systems. IEEE, pp 13–23

Braham A, Buendía F, Khemaja M, Gargouri F (2019) Generation of adaptive mobile applications based on design patterns for user interfaces. Multidiscip Digit Publ Inst Proc 31(1):19

Cañete A, Horcas JM, Ayala I, Inmaculada L (2020) Energy efficient adaptation engines for android applications. Inf Softw Technol 118:106220

Casquina JC, Eleuterio JDA, Sandim R, Cecilia MF (2016) Adaptive deployment infrastructure for android applications. In: 2016 12th European dependable computing conference (EDCC). IEEE, pp 218–228

Deuschel T, Scully T (2016) On the importance of spatial perception for the design of adaptive user interfaces. In: 2016 IEEE 10th international conference on self-adaptive and self-organizing systems (SASO). IEEE, pp 70–79

Evers C, Geihs K (2014) Enabling active user participation in self-adaptive applications. In: Socio-technical design of ubiquitous computing systems. Springer, pp 71–87

Feng J, Liu Y (2015) Intelligent context-aware and adaptive interface for mobile LBS. Comput Intell Neurosci 2015:5–5

Gheibi O, Weyns D, Quin F (2021) Applying Machine Learning in Self-adaptive Systems: A Systematic Literature Review. ACM Trans Auton Adapt Syst. https://doi.org/10.1145/3469440

Hu XL, Zhang LC, Wang ZX (2018) An adaptive smartphone anomaly detection model based on data mining. EURASIP J Wirel Commun Netw 2018(1):1–10

Hussain J, Ul Hassan A, Bilal HSM, Ali R, Afzal M, Hussain S, Bang J, Banos O, Lee S (2018) Model-based adaptive user interface based on context and user experience evaluation. J Multimodal User Interfaces 12(1):1–16

Iqbal MW, Ahmad N, Shahzad SK, Feroz I, Mian NA (2018) Towards adaptive user interfaces for mobile-phone in smart world. In: International journal of advanced computer science and applications, vol 9(11), Science and Information (SAI) Organization Limited

Iqbal MW, Naqvi MR, Khan MA, Khan F, Whangbo T (2022) Mobile devices interface adaptivity using ontologies. Comput Mater Contin 71:4767–4784. https://doi.org/10.32604/cmc.2022.023239

Khan A, Khusro S (2019) Blind-friendly user interfaces—a pilot study on improving the accessibility of touchscreen interfaces. Multimedia Tools Appl 78(13):17495–17519

Khan I, Khusro S (2020) Towards the design of context-aware adaptive user interfaces to minimize drivers' distractions. Mob Inf Syst. https://doi.org/10.1155/2020/8858886

Khan I, Khusro S (2022) ConTEXT: context-aware adaptive SMS client for drivers to reduce risky driving behaviors. Soft Comput 26(16):7623–7640. https://doi.org/10.1007/s00500-021-06705-1

Khan M, Khusro S (2023) Towards the design of personalized adaptive user interfaces for smart TV viewers. J King Saud Univ Comput Inf Sci. 35(9):101777. https://doi.org/10.1016/j.jksuci.2023.101777

Khan M, Khusro S (2023) SmartLog: a smart TV-based lifelogging system for capturing, storing, and visualizing watching behavior. J Hum Comput Interact Int. https://doi.org/10.1080/10447318.2023.2250054

Khan A, Khusro S, Alam I (2018) Blindsense: an accessibility-inclusive universal user interface for blind people. Eng Technol Appl Sci Res 8(2):2775–2784

Khan M, Khusro S, Alam I, Ali S, Khan I (2022) Perspectives on the design, challenges, and evaluation of smart TV user interfaces. Sci Program. https://doi.org/10.1155/2022/2775959

Lara E, Aguilar L, Sanchez MA, García JA (2019) Adaptive security based on mape-k: a survey. In: Applied decision-making. Springer, pp 157–183

Lee H, Choi YS, Kim YJ (2011) An adaptive user interface based on spatiotemporal structure learning. IEEE Commun Mag 49(6):118–124

Machado E, Singh D, Cruciani F, Chen L, Hanke S, Salvago F, Kropf J, Holzinger A (2018) A conceptual framework for adaptive user interfaces for older adults. In: 2018 IEEE international conference on pervasive computing and communications workshops (PerCom Workshops). IEEE, pp 782–787

Moghaddam FA, Simaremare M, Lago P, Grosso P (2017) A self-adaptive framework for enhancing energy efficiency in mobile applications. In: 2017 Sustainable Internet and ICT for sustainability (SustainIT). IEEE, pp 1–3

Naqvi NZ, Devlieghere J, Preuveneers D, Berbers Y (2016) Mascot: self-adaptive opportunistic offloading for cloud-enabled smart mobile applications with probabilistic graphical models at runtime. In: 2016 49th Hawaii international conference on system sciences (HICSS). IEEE, pp 5701–5710

Omar K, Gómez JM (2017) An adaptive system architecture for devising adaptive user interfaces for mobile ERP apps. In: 2017 2nd International conference on the applications of information technology in developing renewable energy processes & systems (IT-DREPS). IEEE, pp 1–6

Raheel S (2016) Improving the user experience using an intelligent adaptive user interface in mobile applications. In: 2016 IEEE international multidisciplinary conference on engineering technology (IMCET). IEEE, pp 64–68

Rathnayake N, Meedeniya D, Perera I, Welivita A (2019) A framework for adaptive user interface generation based on user behavioural patterns. In: 2019 Moratuwa engineering research conference (MERCon). IEEE, pp 698–703

Soui M, Diab S, Ouni A, Essayeh A, Abed M (2017) An ontology-based approach for user interface adaptation. Advances in intelligent systems and computing. Springer, Berlin, pp 199–215

Tanaka S, Iwata H, Shirogane J, Fukazawa Y (2019) Development support of user interfaces adaptive to use environment. In: Proceedings of the 2019 8th international conference on software and computer applications, pp 223–228

Velázquez-García FJ, Halvorsen P, Stensland HK, Eliassen F (2018) Autonomic adaptation of multimedia content adhering to application mobility. In: IFIP international conference on distributed applications and interoperable systems. Springer, pp 153–168

Wang X, Hua QY, Zou F, Guo L (2014) An adaptive user interface model for mobile devices based on perceptual control theory. In: 2014 IEEE 5th international conference on software engineering and service science. IEEE, pp 908–911

Wattearachchi WD, Hewagamage K, Hettiarachchi E (2020) A framework to decide adaptive functionalities by considering user emotions and the context. In: 2020 20th International conference on advances in ICT for emerging regions (ICTer). IEEE, pp 178–183

Wesson JL, Singh A, Tonder BV (2010) Can adaptive interfaces improve the usability of mobile applications? In: IFIP human–computer interaction symposium. Springer, pp 187–198

Wu D, Gao D, Lo D (2021) Scalable online vetting of Android apps for measuring declared SDK versions and their consistency with API calls. Emp Softw Eng 26(1):1–32

Yigitbas E, Sauer S, Engels G (2017) Adapt-UI: an IDE supporting model-driven development of self-adaptive UIs. In: Proceedings of the ACM SIGCHI symposium on engineering interactive computing systems, pp 99–104

Yigitbas E, Hottung A, Rojas SM, Anjorin A, Stefan SS, Engels G (2019) Context-and data-driven satisfaction analysis of user interface adaptations based on instant user feedback. In: Proceedings of the ACM on human-computer interaction, vol 3, no: EICS. ACM New York, pp 1–20

Yigitbas E, Josifovska K, Jovanovikj I, Kalinci F, Anjorin A, Engels G (2019) Component-based development of adaptive user interfaces. In: Proceedings of the ACM SIGCHI symposium on engineering interactive computing systems, pp 1–7

Yigitbas E, Jovanovikj I, Biermeier K, Sauer S, Gregor E (2020) Integrated model-driven development of self-adaptive user interfaces. Softw Syst Model 19(5):1057–1081

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.