ISSN: 2754-6659

Journal of Artificial Intelligence & Cloud Computing

SCIENTIFIC Research and Community

Review Article

Open Access

Adaptive User Interfaces: Personalizing User Experience User Profiling and Modeling

Venkata Naga Sai Kiran Challa

USA

***Corresponding author**
Venkata Naga Sai Kiran Challa, USA.

Received: May 15, 2023; **Accepted:** May 19, 2023; **Published:** May 26, 2023

Data Collection and User Profiling

On the principle of AUIs, user profiling and user modeling are critical processes that play a critical role in the decision-making process as regards adapting the same to accommodate different users. The first step entails an extensive gathering of data that helps drive the understanding of users' behavior and intentions. Three primary data sources are utilized in this process: This form of big data includes interaction, contextual, and precisely defined user-provided data.

Interaction Data

Interaction data includes the user's activities and the other related happenings within the framework of the given system. This concerns click stream data, which are records of the path or sequence of clicks chosen by a user while going through the interface. Clickstream data is essential because it reflects the history of the user's click-through, who accessed the page or feature, and if it was viewed first or second. For example, clickstream data, as described by Bucklin and Sismeiro can inform about users' decision-making processes and reveal specific patterns of user behaviors that are beneficial for developing AU Interfaces [1].

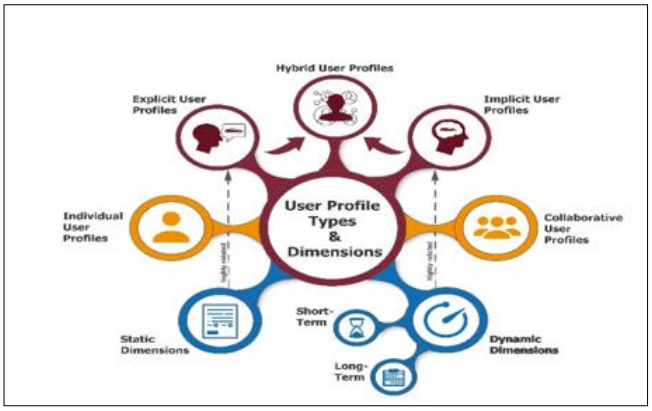

Figure 1: User Definition and Profiling

Contextual Data

Interaction context concerns the characteristics of the environment in which the interaction is taking place; for example, the type of device being used, the physical location of the interaction, or the time of access. Such data is vital primarily to understand the background of specific actions taken by the users and the interface adjustments necessary to suit them. For example, knowing if the user is on a mobile or a computer terminal substantially determines what is provided to them regarding interface design. In the study by Smith and Davenport, the authors stressed the significance of the contextual data. They pointed out that while using location-based services such as GPS, users can obtain more buoyant and valuable information depending on the user's position.

Explicit User-Provided Data

Implicit user data includes information provided willingly by the users and includes the choices, settings, and feedback about the system. It can be obtained by communicative activities performed during a particular service, such as when people fill in the forms or answer surveys. Explicit data are essential in this case as they represent the user's well-calculated stands and preferences, hence the high percentage of personalization possible. In a paper by Dinev and Hart, the authors explained that explicit user-provided data integrated with implicit data enhances the accuracy and efficiency of personal services [2].

Interaction, context, and, last but not least, the explicit input from the user defines the fundamental guidelines of user profiling and modeling. Specifically, any data source provides specific information, which creates a holistic perception of the user. This coordinated approach helps design and implement the reactive user interface, which can meet different users' different needs, thus improving satisfaction and the overall level of user interest. Adaptive user interfaces incorporating elaborate data collection techniques and analysis of users' profiles and models will be a steadily more relevant factor in the increasingly digital world.

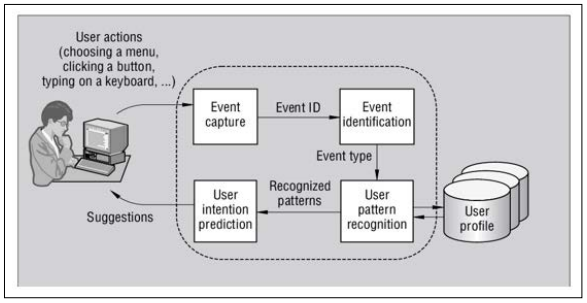

Figure 2: A Personalized Adaptive User Interface

Data Storage and Management

Adequate data storage and management are challenging due to the sheer amount of user data gathered when it comes to realizing an adaptive user interface. This process uses databases, data warehouses, and real-time data processing frameworks.

- **Databases:** Structured data requires proper storage, hence the importance of databases. Relational databases such as MySQL and PostgreSQL are suitable for storing user profiles, records of users' interactions, and other structured information. These databases enable SQL operations that query and manipulate the data efficiently. For instance, Kumar and Karthikeyan paid much attention to the use of relational databases in storing user data for web applications, which are under consideration owing to their robustness and capability of expanding as demand increases
- **Data Warehouses:** Tools like Amazon Redshift and Google BigQuery are the structure of a data warehouse where massive data from various sources are aggregated and analyzed. They support query and analytical processing, essential for data mining, which involves finding patterns in large datasets. Inmon further explains that data warehouses ensure that past data is stored, which can be used for complex analysis and reporting as it feeds into the evolution of the interface.
- **Real-Time Data Processing Frameworks:** Apache Kafka and Apache Flink are the real-time data processing frameworks that stream and analyze real-time data. These frameworks are valuable for the application, as they must be interactive and respond to the user interface in real time based on user actions. In their study, Stonebraker, et al. note that processing data in real-time for interactive applications or systems that should act according to the users' gestures is essential.

User Modeling

Feature Extraction

Finding the essence of the data is an essential step within the framework of modeling for adaptive interfaces. It aims to determine different measures of users' characteristics and then describe and prescribe the resulting profile. These features are generally classified as behavioral, contextual, and demographic.

Behavioral Features

Behavioral features refer to a user's behaviors in an interface and how they tend to perform activities. Such as the number of visits, the time spent on the site, and the pattern of the clicks. To be more specific, such behaviors as visit frequency and their specific click paths can be recognized as user interests. In the view of Chittaro and Montanari, behavioral data is valuable in identifying user preferences and behavior, hence helping in the design or development of adaptive interfaces [3]. Thus, such data can be used to align content and functions to the expectations of their users and enhance the experience.

Contextual Data

Contextual data, in this case, can be defined as the environmental circumstances that define the interaction of the users with the devices, place, and time. To refine this consideration and define where the changes are necessary, such information is essential to improve the interface in response to various contexts of use. For instance, the user using the service from a mobile gadget may need a different layout format than a PC. Dey and Abowd believe that contextual information is the key to augmenting the relevance and usefulness of adaptive systems, which are used to gain insights into the situational context of the user's interactions [4].

Demographic Features

Some of the demographic characteristics are age, gender, and any other characteristics that define the nature of the users and their tendencies. These features are frequently obtained from registration data or derived from the users' behavior. In their work, Nguyen et al. specifically stated that demographic information plays a factor in the user experience, stating that age and gender may affect the preferences and interactions of users [5]. Adding demographic data to the user models allows for designing interfaces that meet the unsaid needs of minority users.

In the context of best practices for creating a good user model, social behavior, context, and demographic data must all be considered and extracted well in feature extraction. These features can contribute to creating adaptive user interfaces that are highly specific to users since they address their specific requirements, which will improve the usefulness of applications with interfaces designed in this way and increase user satisfaction.

User Representation and Segmentation

User Representation

In adaptive user interfaces, user representation is one of the essential features that help gain insights about users and model the user behavior appropriately. Regarding the VSM, there is a recognizable approach to representing users, which may be referred to as user modeling. In the VSM methodology, users are depicted using vectors within the n-dimensional space in which each vector entity corresponds to a particular feature obtained from the user data sources. The primary technique used is the ability to model the distance or similarities between vectors and perform user profile comparisons. VSM was first proposed by Salton, Wong, and Yang in 1975 as an appropriate model for information retrieval and outlining the complexity of the structures. Employing the VSM, adaptive systems establish and operationalize higher-order data, embodying enhanced user interfaces. However, in cases where it is appropriate to describe the user actions' randomness and variability, probability models are used. These models include the Bayesian networks and the Hidden Markov Models (HMM), which depict the users' state and change probability. For example, HMM can be applied to sequences of users' actions and forecast further behaviors using observed patterns. Adaptive systems and probabilistic models are discussed by Ghahramani especially in classifiers that employ incomplete and noisy values [6]. There are many applications of probabilistic models, especially in cases where the behavior of users changes and is dependent on context since they will be able to make predictions more easily.

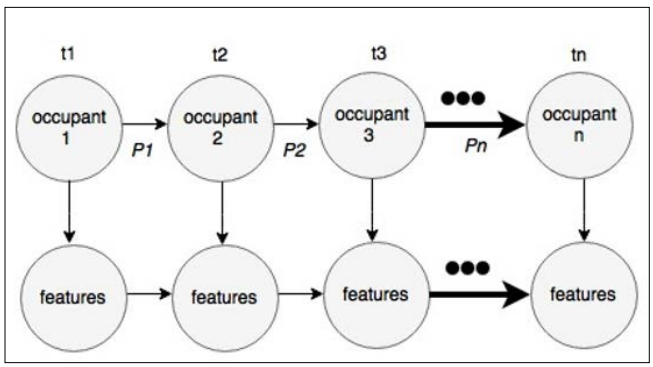

Figure 3: Hidden Markov Model (HMM) structure

User Segmentation

User segmentation is the division of users into several groups based on their characteristics and activity. Often, clustering algorithms are used for such a task. For example, K-means groups the users into k clusters by using feature similarities with the objective of reducing the variance of each cluster. This is common and efficient in handling most problems and is commonly used in adaptive interfaces. MacQueen further showed the efficiency of K means for clustering problems by noting that it is easy to implement [7].

Another clustering algorithm is DBSCAN (Density-Based Spatial Clustering of Applications with Noise): it finds the cluster based on the density of the points. It helps find clusters of arbitrary shapes and deal with noise. Ester et al. proposed DBSCAN in 1996, proving that the devised method for clustering is superior to most existing methods in terms of its ability to find informative patterns in big data.

Aside from clustering, latent class analysis (LCA) is used to extract latent user groups based on interactions and contextual information. LCA assumes that more hidden classes influence the observable data while segmenting the users, thus giving the model a probabilistic nature. Collins and Lanza outline using LCA in behavioral research, which is suggested to uncover structures in large datasets. LCA can contribute to discovering subtleties of the adaptive systems' users that are not exposed when applying clustering techniques, thus improving the capacity of the adaptive systems to focus on specific segments of users [8].

ML Algorithms

In adaptive user interfaces, machine learning algorithms process user data and provide improved and recommended features. They can be classified into three broad types: supervised learning algorithms, unsupervised learning algorithms, and reinforcement learning algorithms. All three have different methods and uses.

Supervised Learning

Supervised learning is learning from a set of data with known outcomes for given inputs. It is generally used for classification and regression to assign different weights to different instances in the training set.

- **Classification:** Since the goal is to bin the data of users into specific categories, methods like Support Vector Machines (SVM) and Random Forests are employed. SVM is beneficial in high-dimensional space and is famous for its efficiency in text and image categorization problems. According to Cortes and Vapnik, support vector machines maximize the margin between the different data classes, separating them and enhancing the classification capability [9]. Random Forests, in contrast, is an ensemble learning method based on decision trees, aiming to increase the latter's accuracy and decrease the impact of overfitting. According to Breiman, Random Forests are flexible and less sensitive to the characteristics of significant cases [10].
- **Regression:** Regression algorithms forecast continued variables depending on the input data imparted to them. Among the most widespread methods used to describe dependencies and make forecasts, linear regression, and its modifications are worth mentioning. For instance, as a basis for the application, using stats on how users have responded in the past on certain content can help make content available as per the user's choice. The paper under discussion also shows the use of regression analysis in various fields by Draper and Smith, proving this method's efficiency in quantitative estimations [11].

Unsupervised Learning

Unsupervised learning refers to working with data that does not have predefined labels. It can be used in tasks like clustering and reducing the data dimensions.

- **Clustering:** Analyzing the inputs of similar outliers is a job that clustering algorithms like the K-means algorithm can do because they ensure that similar users are categorized into one segment. As MacQueen observed, k-means categorizes the data set into k clusters with a criterion of minimizing the samples' variance, which qualifies it for users' segmentation [7].
- **Dimensionality Reduction:** A procedure like Principal Component Analysis and t-distributed Stochastic Neighbor Embedding is applied to cut down the size of a dataset's features while also retaining the general structure of the set. PCA rotates the data to a new reference frame so that the principal components remove the weakest components of data variations. In the same sentiment concerning the application of PCA for complexity reduction of the dataset, Jolliffe stated that PCA eases interpretation. T-SNE visualization was proposed by van der Maaten and Hinton to help visualize high-dimensional data to provide more noticeable patterns when transformed into a lower dimensionality [12].

Reinforcement Learning

Reinforcement learning is a learning process in which an agent faces an environment and makes decisions by receiving feedback as a reward for its actions. It is more effective in environments that are constantly changing as it picks up new interaction patterns with time.

- **Markov Decision Process (MDP):** MDPs describe decision scenarios in which some aspects are stochastic and others depend on a decision maker's choices. Sutton and Barto have defined MDPs as basic to reinforcement learning because they provide a way to achieve the most significant reward over time.
- **Deep Q-Learning:** Deep Q-Learning is a blend of Q-Learning, a value method that works best in large state spaces, and deep neural networks. Mnih et al. used Deep Q-Learning to effectively increase performance to a superhuman level in playing Atari games, exhibiting the capacity of Deep Q-Learning in decision-making [13].

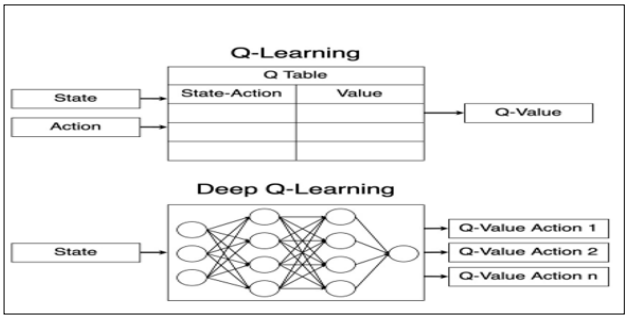

Figure 4: Q-Learning vs. Deep Q-Learning

The described use of the supervised, unsupervised, and reinforcement learning algorithms helps enhance the user interface's adaptation. These ML techniques allow applications to learn a user's behavior and respond, reason, and learn the user's actions in real time, thus improving the service quality.

Adaptive Mechanisms

Dynamic adaptations in UIs include several strategies devoted to adapting the content, layout, and functionality regarding the

information concerning the user. These are achieved to ensure that people use the phone in a tailored and effective manner.

Content Adoption

Recommendation Systems

Recommendation systems are very essential in creating personalized content for the users. These systems employ collaborative filtering techniques, either user-based or item-based. User-based CF identifies similar users and then recommends items that similar users prefer. While the former suggests to a user item similar to what he/she liked in the past, the latter filters items based on comparing them to a particular item. Konstan et al. explained that collaborative filtering was suitable for recommending content by observing users' behavior and preferences to increase user satisfaction and value [14].

Another type of filtering, content-based filtering, suggests items based on the properties of items the user has expressed an interest in. It is aimed at assessing the objects' features instead of the users. This hybrid approach integrates the features of both the collaborative and content-based methods, whereby recommendations are produced using the best aspects of the two systems, thus improving their efficiency. Burke said about hybrid recommendation systems that such systems enhanced the reliability of recommendations and their variety when using more than one filter [15].

Personalized Search

Personalized search enhances the search results depending on the user's previous search story of the user making. Learning to Rank (LTR) is the usual approach used to enhance the ranking of the search results per the user's expectation. Due to machine learning models, LTR algorithms employ an ordering that helps make items more relevant to the end user. Joachims also proved that with the use of LTR in(re)searching, the coherence of the obtained results can be boosted due to the learning process based on the interaction with the users' feedback and responses [16].

Layout Adoption

Responsive Design

Adaptive design is crucial in mobile application development since different applications should work well on all devices and all sizes of screens. Responsive design, now implied by most designers to mean the creation of interfaces for multiple resolutions, is implemented using CSS frameworks, including Bootstrap and Tailwind CSS. These frameworks contain a small number of pre-planned constituents and grid apparatus to inform the layout based on the screens of the viewed device. Marcotte suggested that website design was shifting towards what he named responsive design, which helped to make the websites qualitative and responsive in terms of accessibility and usability on different devices.

Dynamic Component Loading

Load time optimization refers to the toggling of UI elements based on the user's context and filtering of UI components that should not be rendered on the page the first time the page is loaded. This means that some low-priority resources are loaded only when they are required and not at the point of time when a page is opened, which in turn enhances the speed of page loading and, in turn, overall site performance. Conditional rendering adapts the components that will be shown to the application user depending on the determined context, which can be the type of the device being used, or the user's preferences. In the article, Grigoras et al. mentioned the advantages of dynamic component loading to show how this or that strategy affects the optimization of the performance and the enhancement of the user experience [17].

Functional Adoption

Feature Toggles

Feature flags are also called feature toggles because they help to turn a feature on and off without any need to release a new code. This mechanism helps do A/B testing, slow rollout of new features, and rollback of features if anything goes wrong. Products such as Launch Darkly contain strategies for properly using feature toggles. In their paper, Rahman et al. described the adoption of feature toggles when applied to software development, with the main concepts of continuous integration and delivery, decreasing the risk of deployment, and increasing the level of agility.

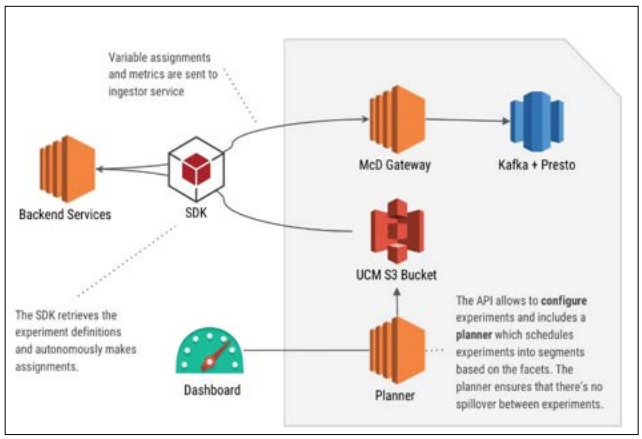

Figure 5: Feature Toggles and A/B Testing SDK

Contextual Menus

The contextual arrangements of the actions and the menu options vary according to the user's specialty and the task's current context. This adaptation makes it easier to organize tasks and enhances operational efficiency for some outcomes since the user is shown how this situation could turn out. For instance, while new controllers may observe basic options, advanced ones are available to proficient controllers. Contextual adaptation can lead to a massive improvement in usability, resulting in an overall better user experience. Johnson et al. included a study about the applicability of contextual menus in offering the stimuli of augmented user interactivity, asserting their ability to decrease the user's cognitive load and offer the most appropriate choices to the user during the execution of the tasks [18].

Adjustable interface techniques refer to enhancing the user interface by incorporating sophisticated procedures to affect the content, layout, and interactivity. Through recommendation systems, search engines that are specific to certain individuals, websites' most preferred layout, component libraries that are only loaded when needed, feature optimization options, and menus that only appear when relevant, it is possible to create user experiences that are highly specific to what the users need. These mechanisms increase users' satisfaction, engagement, and effectiveness; thus, creating adaptive interfaces has become critical in contemporary digital applications.

System Architecture

Adaptive user interfaces must always have an optimum system architecture since it is the linchpin of efficient, flexible, and responsive interface use. The following architecture is usually adopted, using client-side frameworks, server-side microservices, and midware.

Client-Side

On client-side MEAN stack technologies, which includes JavaScript frameworks like React, Angular, and Vue. Js are widely applied to create attractive and interactive front ends of web applications. These frameworks allow the generation of single-page applications (SPA) for which only a part of the content is reloaded. This capability is necessary for adaptive interfaces, especially if they respond quickly to the users' actions. For instance, React, Facebook, is well known for its rendering and state management functionalities for complex and interactive UIs [19]. Moreover, there are possibilities for local storage to be equipped with LocalStorage and IndexedDB to store preferences and to access them quickly. Cookies can be easily used for key-value storage, where the data is saved locally for the current session and can be accessed in the next session. Local storage also provides a key-value store, but for more extensive data and complex structures, there is IndexedDB, which is an asynchronous database. This local storage is quite helpful in caching preferences and the state of the user, which, in a way, boosts the application's response time and sustains the same user experience if the internet connection is unavailable. IndexedDB is considered vital for client-side storage based on Gaunt asserting that IndexedDB features flexibility regarding data types and client transaction handling.

Server-Side

Microservices architecture is preferred on the server side for its scalability when designing the solutions. Microservices decompose the application into smaller and independent services, which may be deployed separately, and these services interact using APIs. This modular scale ensures that maintenance can be done concurrently and that the rate of deploying the equipment is enhanced. As Lewis and Fowler noted, continuous delivery is promoted in the microservices architecture, and the applications can manage complexity and change well [20]. RESTful and GraphQL are some APIs deployed in the client-server communication processes. RESTful APIs mainly adhere to the stateless and client-server model and are accessible, easy to implement, and easily scaled. GraphQL is an open-source developed by Facebook that lets clients ask for what they need instead of receiving too much or too little information. This flexibility is essential in adaptive interfaces where data that has to be retrieved is dynamic and differs depending on the users' interaction [21].

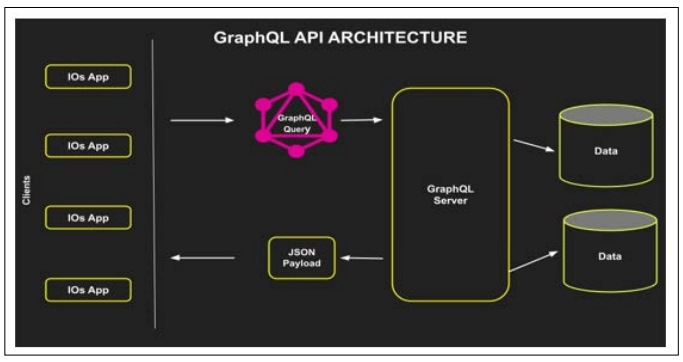

Figure 6: GraphQL vs REST

Middleware

Middleware components control the interactions between the client-side and the server-side services. Versatile middle tiers may alter their operation based on the existing context and conditions to provide optimal data transmission. Systems such as Apache Kafka and RabbitMQ support message passing by having services loosely coupled and guaranteeing the passing of the message. As Kreps et al. have noted, Kafka has high throughput and low latency, thus making it appropriate when handling real-time data streams, which are fundamental in Adaptive UIs that need real-time interaction [22].

Adaptive UI design mainly involves using a client-end JavaScript framework, server-side microservices, and an adaptive middleware layer. The components above collectively result in an effective and flexible structure that can easily adjust to suit each user's requirements.

Evaluation and Optimization

Assessment and fine-tuning are indispensable phases in designing and implementing user interfaces that can adapt to the interacting subjects. Such processes enable the interface to always be up to its challenge of addressing the users' needs as they dynamically transform.

A/B Testing

A/B or split testing, as it is also known, is a popular procedure that applies in assessing various versions of a user interface. This involves making two or more versions of a feature for different population groups to decide which version performs better. Specific manipulations are undertaken, often as experiments that are controlled in various ways and for which statistical tests are sought to determine whether a difference exists in the behavior of the users or the results. The authors Kohavi et al. give an application of A/B testing in web development and show that it helps to make rational data-driven decisions regarding improving the user experience [23].

Organizations like Meta often use A/B testing to optimize their features. One of the most typical examples is feature flags – the specific feature that can be allowed or blocked depending on the selected user's segments. All these flags are managed through a specific portal where characteristics such as gender, age, geographical location, or actions are specified and set. In this case, the specific feature to which the user meets these criteria is activated, and its effect can be quantified. This method allows for specific targeting of the features and avoidance of certain dangers within the process since feature implementation is often done step by step with the ability to revert if problems are identified.

User Feedback and Logging

The reception of clients' opinions is crucial to evaluating customers' satisfaction and discovering possible problem areas. Examples include surveys, user interviews, and feedback tools that are integrated into the application or website. Also, there are user analytical tools such as Google Analytics, which can provide detailed observations of users and their dispersion, activities, and interactions. Finstad states that one of the key objectives is the user feedback and analysis of the software product as the basis for subsequent iterations of the design process and its alignment with users' expectations and requirements convenient for them.

Continuous Learning

The learning also encompasses modifying the system continuously depending on new data and interacted users. It may be accomplished during online learning when the model's parameters are modified in parallel with the new data received. Periodic model update helps the system adjust to new users' behavior and their changing preferences. In particular, Bifet and Gavaldà touched on the subject of learning with continuous data streams, pointing to the fact that it is essential for adapting the prediction model to live data sources and keeping the model up-to-date, if necessary [24].

Evaluating and optimizing adaptive user interfaces through A/B testing, user feedback, and learning is indispensable for their success. These activities allow the interface to remain relevant to users, increasing their satisfaction level.

Comparison with Traditional Interfaces

Traditional Interfaces

Static-modeled UIs display a structured and fixed design and are easily comprehensive because of their generality. These interfaces are developed with a set-specific idea, where the layout and options are similar to every user regardless of their characteristics.

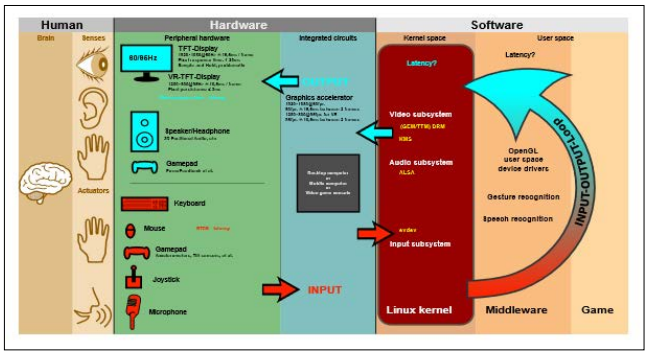

Figure 7: User interface design

Static Design

Traditional interfaces are inherently unidimensional, which implies that designing them is based on a GUI's specific characteristics that do not alter its design or behavior in response to users' actions. This is an unfavorable characteristic for member station interfaces because it suggests that the site will remain stagnant and incapable of meeting the diverse and changing needs and expectations of the users after some time, according to Nielsen. This characteristic hinders differentiation and extensibility to cater to diverse user needs, hence resulting in inferior usability.

Generalized Approach

All these interfaces are generic and have been developed with an obvious intent to address an ordinary user without necessarily considering that users may differ. A generalized approach has disadvantages as it can serve the purpose without considering the individual users' requirements and sensitivity. Shneiderman and Plaisant reported that the generalized interfaces lead to a situation where people claim that there is a "one-size-fits-none" since everyone cannot be appreciated or satisfied with the same user experience. While it is knowledge, it is often a broad focus that needs more detail regarding different types of users, thus bringing down the overall usability of an interface.

Predictable Experience

Traditional interfaces are mostly uniform to all users, which can be a strength or a weakness. The predictability of interfaces makes navigation easier and learning less demanding, but simultaneously, it hinders variability to satisfy all users. Norman pointed out that while on static interfaces, predictability may become a disadvantage when applied to a complex or varied user population as it cannot recognize a user's context or profile.

Adaptive Interfaces

Adaptive interfaces are based on collected user data, consumer feedback, and the pliability of interface elements.

Dynamic Evolution

Dynamic interfaces, also called adaptive ones, are defined to change according to the user's activity or their reactions to it. This attribute makes it possible to design and develop gadgets to address some basic customer requirements of the diverse users. To this end, Brusilovsky and Millán instilled the prospect of adaptability in such interfaces because they deliver more meaningful use experience to the ordinary user because the interfaces adapt to the users' needs in real-time [25].

Personalized Experience

An adaptive interface is rich in personalization, delivering an experience based on users' profiles and activities. Thus, such interfaces can present contents, layouts, and functions tailored to the user data that they have captured. Jameson pointed out that personalized tools increase customers and productivity based on relevant and context-adapted information and options.

Flexibility

Another feature of adaptive interfaces is high adaptability, which allows them to shift to a new context and suit the user's experience. This helps the developers cater to the best user experience on different gadgets and occasions. Gajos et al. also showed that concepts of flexible interfaces enhanced accessibility and usability as they changed in line with the user's requirements and tasks [26]. At the same time, this flexibility prescribes the best fit for the interface to solve a given problem no matter what environment it is applied to and does not limit it when it is not needed.

It could be concluded that although traditional interfaces have certain benefits, such as predictability and simplicity, a modern application cannot satisfy the heterogeneity of the contemporary user. Dynamic, personalized, and flexible interfaces make it easier for the user to use and learn since they are intelligent and function according to the user's techniques.

Pros and Cons

Pros

Enhanced User Experience

Organizational user interfaces (OUIs) improve user journeys by introducing adaptability. It results in better user experience and interaction as interfaces that are developed fond of and sensitive to users' preferences and habits perform better. McKinsey & Company concluded in their report published in 2016 that measures towards personalization of users' digital experience could lead to a 20% improvement in satisfaction among users [27]. This makes the interface more relevant and valuable to the user and increasing the focus and usage of the interface and, thus, the AUI.

Increased Efficiency

Adaptive interfaces enable the user to become more productive by completing tasks faster and with less effort. Since AUIs filter the options that are likely helpful to the users and align the most utilized paths, usage time is considerably reduced. Shneiderman and Plaisant quo cited, suggest that improvements to the UIs can boost the productivity of systems by a significant degree since the users spend reduced periods searching for the information they need and more time engaged in positive work.

Accessibility

Using AUI offers an opportunity to design solutions for disabled or specially needed users and make digital content more approachable. Reducing user dependencies on specific systems outcomes in embracing different inputs, screens, and assistive technologies, and as such, AUIs can be more accommodating. For instance,

Gajos et al. showed that adaptive interfaces benefited users with Motor disabilities and allowed the change of the interface as per the user's preference [26]. This flexibility guarantees optimum user interaction with the system despite the user's disabilities.

User Retention

Customization can increase the utility and traffic of the site by increasing patrons' engagement and return visits. In this way, users believe an interface is designed to meet their needs and expectations. Thus, they will stay loyal to the product and not switch to other similar products. According to Accenture, 91% of consumers were willing to shop more with the relevant brands [28]. This could mean personalization is central to customers' engagement in such settings.

Cons

Complexity

Developing effective AUIs requires the most efficient algorithms and substantial user information. A disadvantage, though, is the relative difficulty of constructing and supporting such systems, which are grounded in modern AI technologies such as machine learning and data stream processing. Out of all the various facets of adaptive systems, Hearst notes that knowing the technology and the people and finding a way to balance the two may be difficult [29].

Privacy Concerns

The use of user data presents various questions regarding its collection and application, mainly in the aspects of privacy and data protection. Concerns can be raised about whether users know how their data is being processed and protected enough. In the EU, the General Data Protection Regulation (GDPR) indicates high standards for data privacy. As Acquisti, Brandimarte, and Loewenstein noted it is tough to balance adopting personalization and privacy for adaptive systems [30].

Resource Intensive

Due to real-time adaptation, AUIs demand large computational power. The capability of handling massive data on a typical daily basis can put pressure on the system and add up to expenses. In this regard, Dean and Ghemawat claim that the underlying infrastructure must be scaled and efficient to foster the success of adaptive systems [31].

User Trust

People may have concerns about automated systems that self-optimize according to the users' actions. If the machine learning algorithms are black-boxed, the users are likely to get suspicious of their way of making a particular decision, or they find that the interactive interface has suddenly changed. Parasuraman and Riley noted that to build trust in the system, the users must feel they have adequate control, and anything that depicts the opposite will erode this trust [32]. Developing responsive and easy-to-understand adaptive systems is therefore essential to enhance the users' confidence.

Current Deployments

AUIs are extensively used in different sectors where personalization application is used to improve interaction and performance.

E-commerce Platforms

In electronic commerce, adaptive user interfaces are used to customize products that are suggested to shoppers and search results. Sites like Amazon and eBay use artificial intelligence to study website usage and buying patterns and recommend products based on user preferences. This personalization enhances the chances of the customer purchasing, raising customer satisfaction. Smith and Linden show that Amazon's networking increases sales due to its ability to recommend the most suitable products based on information regarding the user.

Social Media

Companies like Facebook and Twitter utilize AUIs to choose what to display on a user's news feed and the advertisement that should be shown to the user. These platforms analyze user engagements and present content that the users are interested in, optimizing engagement. For example, in the case of social sites such as Facebook, the mean filtering satisfies user needs in that it sorts posts and ads for the particular user in increasing order of the user's likelihood of interest in them based on their activity history. According to the study by Bakshy, et al. using a news feed that shows a user more of the content they like leads to more engagement and better ads [33].

Healthcare

AUIs adapt human interfaces and health suggestions in healthcare according to past care and personal desires. Implementations like EHRs and patient portals adjust the interface to inform and remind the patients of their health state. Blumenthal and Tavenner noted that implementing the EHRs with the manufacturing elements of an individual patient aids in increasing the patient's health status by providing the recommendations at the right time [34].

E-learning

Online learning systems like Coursera and Khan Academy can change the content delivered and the layout so that they align with the student's learning style and learning achievement. Based on student performance and activity data, they provide individual learning pathways and materials. According to the authors Brusilovsky and Millán, adaptive educational systems increase learning effectiveness and learner satisfaction by adapting to students' learning styles [25].

Smart Home Devices

Voice assistants and smart thermostats or the so-called smart home devices learn from the users and their habits. Smart home devices like the Nest thermostat are characterized by their ability to gain information regarding users' daily programs and choices to ensure efficient resource utilization. Mozer et al. have shown that adaptive home automation energy use is enhanced considerably with improved comfort levels through learning and user preferences [35].

Key Considerations

Data Collection and Analysis

Decent AUIs depend on sound data capture mechanisms and sophisticated algorithms to analyze users' behaviors. This entails collecting information on users from various aspects, such as interactions, the environment, and user-provided information. Mayer-Schönberger and Cukier have noted that the idea and possibility of handling Big Data constitute one of the fundamentals that enable the creation of adaptive and relevant systems [36].

Algorithm Selection

There is a need to select proper machine learning algorithms to run the adaptive features optimally. Multiple algorithms are appropriate for classification, regression, clustering, and recommender systems. The overall success of an AUI is based on adequately implementing these algorithms to analyze the user data and correctly forecast preference. Domingos has mentioned that the selection of the particular algorithm affects the ability and the efficiency of the system to learn and personalize [37].

User Consent and Transparency

For ethical considerations, it is crucial to ensure the users know the data being collected and get their permission. An organization should use user data only with the user's approval and acknowledgment of the data's legal requirements by adhering to regulations such as GDPR. According to Acquisti, et al. there is a need to combine user consent and transparency so that ethical benchmarks are upheld when designing adaptive systems [30].

Continuous Improvement

The concept of AUIs also implies that they should be dynamic and successively enhanced regarding user feedback and newer technologies. This involves putting in place measures that make the system incorporate constant learning and adjustment to maintain its effectiveness. McKinsey & Company further emphasizes that updating the range and depth of personalization methods pays for continually stronger user engagement and satisfaction results [27].

Future Direction

Adaptive user interfaces are set to become a more distinct category, with advanced technologies and users' expectations driving further developments. These include advanced Artificial Intelligence, platform compatibility, Emotional and Behavioral adaptation, and Privacy and Security issues.

Advanced AI Integration

Deep learning and other types of neural networks may improve the existing user modeling and personalization approaches. These technologies allow the systems to process large amounts of information and search for the nonlinear relationships in the users' actions. Other advanced structures of DL, including CNNs and RNN, can improve the functionality of AUIs to deliver a personal touch. Goodfellow et al. pointed out that deep learning could significantly revolutionize aspects of AI, such as adaptive interfaces, due to increased accuracy and capabilities of predictive computations [38]. Incorporating these advanced AI approaches will help make AUIs even smarter by enriching and making them context-sensitive, thus improving their performance.

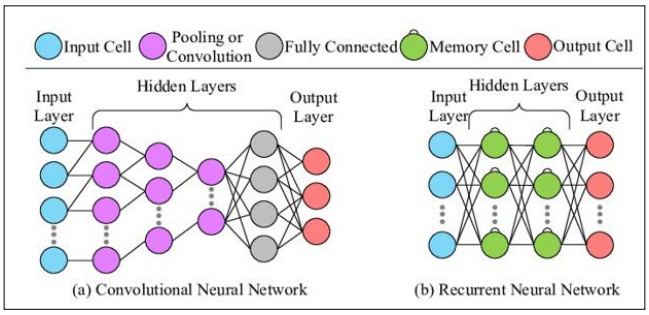

Figure 9: General structures of CNN and RNN

Cross-Platform Adaptation

Continuity of adaptive experience across devices and platforms is another important research area for AUIs in future development. Customers engage with services more through personal computers, tablets, smartphones, laptops, and wearable electronics. The integration of interfaces means designing interfaces that can respond to variations in the size of the displays, modes of input, and the overall surroundings of operation. The advantages of this approach include improved user interaction and satisfying design uniformity across devices. According to Nielsen, responsive web design, which aims to create web pages compatible with multiple devices, enhances the website's usability [39]. Employing the envisioned future, AUIs of the near future will utilize qualitatively improved techniques to ensure proper integration of various user interfaces, thus offering a harmonious experience.

Emotional and Behavioral Adaptation

Since emotion recognition and behavioral analysis may be easily implemented in AUIs, more natural and organic adaptations may be achieved. Emotion recognition technologies identify emotions from readers' facial expressions, voice pitch, and physiological responses, whereas behavioral analysis involves the study of patterns in which users engage with a system. Recognizing the ECM and BCM, AUIs can now adjust the response and the interfaces provided to the users. For instance, Picard proposed the idea of affective computing, which is acknowledgment or a move to close the gaps between human emotion and computing experiences [40]. Integrating these concepts into AUIs may improve their capacity to produce empathetically adapted responses pertinent to the user's context. This possibility can positively affect satisfaction and user engagement.

Enhanced Privacy Measures

With the development of personalization, security measures are also in high demand. It is critical to devise strategies for building privacy-preserving technologies to strike a proper equilibrium between the flexibility associated with membership personalization and the security of its customers' data. Innovations like differential privacy, federated learning, and secure multi-party computation can shield personal data even when delivering personalized content. Dwork elaborated that differential privacy is the most effective means for preserving privacy in extensive data analysis [41]. Future AUIs will have to use such and other innovative methods to retain customers' confidence and adhere to higher levels of data protection control.

Future development is focused on further stabilizing AI, providing cross-platform support, allowing emotional and behavioral adjustments, and answering security questions. Such developments will help AUIs enhance user satisfaction, usability, and security as users' needs and expectations constantly rise.

Citation: Venkata Naga Sai Kiran Challa (2023) Adaptive User Interfaces: Personalizing User Experience User Profiling and Modeling. Journal of Artificial Intelligence & Cloud Computing. SRC/JAICC-386. DOI: doi.org/10.47363/JAICC/2023(2)369

Use Case

Several adaptive interfaces were created and integrated during a stay in Meta or Facebook. A perfect example is the settings page in the facebook. The team had unrestricted user data and classification, including generic, marketing, and page users. In addition to this, the use of interaction data was also taken into account. Thus, by using this information, including geographical data and other factors, the settings page changed its design to correspond to each user's preferences. The frame was created to sort and rearrange parameters and reconstruct such tasks into a single function that had previously required two or more clicks or taps. This adaptive approach helped make the settings most frequently used for the client easily accessible, thus increasing usability.

Backend Setup

Data Collection and User Profiling

```
from flask import Flask, request, scalar
from flask_sqlalchemy import SQL Alchemy
import jilbab
import pandas as pd

app = Flask( name )
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///users.db'
db = SQL Alchemy(app)

class User(db. Model):
    id = db. Column(db. Integer, primary key=True)
    username = db. Column(db. String(80), unique=True, nullable=False)
    age = db. Column(db. Integer, nullable=False)
    preferences = db. Column(db. JSON, nullable=False)
    interaction history = db. Column(db. JSON, nullable=False)  # Interaction history to track navigation paths

@app.route('/update profile', methods=['POST'])
def update profile():
    data = request. Json
    user = request. Json(username=data['username']).first()
    if not user:
        user = User(username=data['username'], age=data['age'], preferences=data['preferences'], interaction history=data['interaction history'])
        request. Json(user)
    else:
        user. Age = data['age']
        user. Preferences = data['interaction history']
        user. Preferences()
    return scalar({'message': 'Profile updated'}), 200

if __name__ == '__main__':
    user. Preferences()
    apparent(debug=True)
```

ML Model Training - A combination of clustering and sequence modeling (e.g., LSTM) for this task.

```
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
import numpy as np
import pandas as pd
import joblib

# Sample data: User profiles and interaction history
data = [
    {'username': 'user1', 'age': 25, 'preferences': {'privacy': 'high'}, 'interaction_history': ['privacy', 'notifications']},
    {'username': 'user2', 'age': 30, 'preferences': {'privacy': 'medium'}, 'interaction_history': ['theme', 'privacy']},
    # Add more user data
]

df = pd.DataFrame(data)
df['interaction_sequence'] = df['interaction_history'].apply(lambda x: ' '.join(x))
df['privacy'] = df['preferences'].apply(lambda x: 1 if x['privacy'] == 'high' else 0)

X = df[['age', 'privacy']].values
scaler = StandardScaler().fit(X)
X_scaled = scaler.transform(X)

# Clustering users into different segments
kmeans = KMeans(n_clusters=2, random_state=42).fit(X_scaled)
df['cluster'] = kmeans.labels_

# Save the model and scaler for future use
joblib.dump(kmeans, 'kmeans_model.pkl')
joblib.dump(scaler, 'scaler.pkl')
```

Making Predictions and Generate Dynamic Settings

```
@app.route('/get_dynamic_settings', methods=['GET'])
def get_dynamic_settings():
    username = request.args.get('username')
    user = User.query.filter_by(username=username).first()

    if not user:
        return jsonify({'error': 'User not found'}), 404

    user_data = [user.age, user.preferences['privacy'] == 'high']
    user_data_scaled = scaler.transform(user_data)
    cluster = kmeans.predict(user_data_scaled)[0]

    # Define different settings layouts for each cluster
    settings_layouts = {
        0: ['privacy', 'notifications', 'theme'],
        1: ['theme', 'privacy', 'notifications']
    }

    dynamic_settings = {
        'reordered_settings': settings_layouts[cluster],
        'new_settings': generate_new_settings(user.interaction_history)
    }

    return jsonify(dynamic_settings), 200

def generate_new_settings(interaction_history):
    # Example logic for generating new settings based on interaction history
    new_settings = []
    if 'privacy' in interaction_history:
        new_settings.append('advanced_privacy')
    if 'notifications' in interaction_history:
        new_settings.append('notification_schedule')
    return new_settings
```

Frontend Setup (Sample Code)

```
// App.js
import React, { useState, useEffect } from 'react';
import axios from 'axios';

const App = () => {
  const [settings, setSettings] = useState(null);
  const username = 'user1'; // For demonstration, hardcoded username

  useEffect(() => {
    axios.get(`/get_dynamic_settings?username=${username}`)
      .then(response => {
        setSettings(response.data);
      })
      .catch(error => {
        console.error('Error fetching settings:', error);
      });
  }, [username]);

  if (!settings) return <div>Loading...</div>;

  return (
    <div>
      <h1>Settings Page</h1>
      {settings.reordered_settings.map(setting => (
        <SettingComponent key={setting} setting={setting} />
      ))}
      {settings.new_settings.length > 0 && (
        <div>
          <h2>New Settings</h2>
          {settings.new_settings.map(setting => (
            <SettingComponent key={setting} setting={setting} />
          ))}
        </div>
      )}
    </div>
  );
};

const SettingComponent = ({ setting }) => {
  switch (setting) {
    case 'privacy':
      return <PrivacySetting />;
    case 'notifications':
      return <NotificationSetting />;
    case 'theme':
      return <ThemeSetting />;
    case 'advanced_privacy':
      return <AdvancedPrivacySetting />;
    case 'notification_schedule':
      return <NotificationScheduleSetting />;
    default:
      return null;
  }
};

const PrivacySetting = () => (
  <div>
    <label>Privacy</label>
    <select>
```

```
        <option                                                    value="high">High</option>
        <option                                                  value="medium">Medium</option>
        <option                                                     value="low">Low</option>
      </select>
    </div>
);

const                NotificationSetting                =                ()                =>                (
    <div>
      <label>Notifications</label>
      <select>
        <option                                                   value="email">Email</option>
        <option                                                     value="sms">SMS</option>
        <option                                                    value="push">Push</option>
      </select>
    </div>
);

const                ThemeSetting                =                ()                =>                (
    <div>
      <label>Theme</label>
      <select>
        <option                                                    value="dark">Dark</option>
        <option                                                   value="light">Light</option>
      </select>
    </div>
);

const                AdvancedPrivacySetting                =                ()                =>                (
    <div>
      <label>Advanced                                                                Privacy</label>
      <select>
        <option                                                    value="strict">Strict</option>
        <option                                                  value="relaxed">Relaxed</option>
      </select>
    </div>
);

const                NotificationScheduleSetting                =                ()                =>                (
    <div>
      <label>Notification                                                            Schedule</label>
      <select>
        <option                                                     value="daily">Daily</option>
        <option                                                    value="weekly">Weekly</option>
        <option                                                   value="monthly">Monthly</option>
      </select>
    </div>
);

export default App;
```

This idea can be used anywhere. For example, in automobile industry, the infotainment home screen which users can go in a tap/click can populate dynamically according to the user's behavior/metadata so that the drivers can find the app/option they are looking for in a second or two so that they don't have to take their eyes off the road for long.

Enhancing Adaptive User Interfaces with IPFS Node Cluster for Data Replication

Integrating an IPFS (InterPlanetary File System) node cluster into adaptive user interfaces enhances the personalization of user experiences through improved data replication, availability, and security. This section explores the benefits of combining user profiling and modeling with a decentralized data replication system.

Adaptive user interfaces (AUIs) dynamically adjust their content and layout to match individual user preferences and behaviors. Effective AUIs rely on accurate user profiling and modeling, which require robust and reliable data storage and retrieval systems. Incorporating IPFS for data replication can significantly enhance these systems by ensuring data integrity, availability, and security.

Adaptive User Interfaces (AUIs)

AUIs aim to create personalized user experiences by adapting to user behaviors and preferences. This requires collecting and analyzing large amounts of user data, which is used to build detailed user profiles and models.

IPFS Overview

IPFS is a decentralized storage protocol that enables peer-to-peer data sharing and storage. Its content-addressed, decentralized nature makes it an ideal solution for enhancing data replication and availability in various applications.

Proposed Integration

Data Replication with IPFS Node Cluster

Integrating IPFS into the AUI framework involves deploying an IPFS node cluster to handle data replication. This setup ensures that user data is distributed across multiple nodes, enhancing redundancy, availability, and security. The following steps outline the integration process:

User Data Collection

Collect user data through various interaction points within the AUI. This data includes user preferences, behaviors, and interaction histories.

Data Storage and Replication

Store the collected user data in an IPFS node cluster. The decentralized nature of IPFS ensures that data is replicated across multiple nodes, providing high availability and fault tolerance.

User Profiling and Modeling

Utilize the replicated data to build and update user profiles and models. IPFS ensures data integrity, enabling accurate and reliable user profiling.

Real-Time Data Access

Implement real-time data access and retrieval mechanisms to fetch user data from the IPFS cluster. This ensures that the AUI can adapt in real-time based on the latest user data.

Personalized User Experience

Use the user profiles and models to deliver a highly personalized user experience. The AUI can dynamically adjust content, layout, and interactions based on individual user preferences and behaviors.

Benefits of IPFS Integration

Scalability

- IPFS allows horizontal scaling by adding more nodes to the cluster, efficiently handling increasing volumes of user data.

Data Availability and Redundancy

- Data stored in IPFS is replicated across multiple nodes, ensuring high availability and resilience against node failures.

Security

- IPFS provides inherent data integrity checks through content addressing, ensuring that user data remains secure and unaltered.

Cost Efficiency

- By distributing data across a decentralized network, IPFS can reduce storage costs compared to traditional centralized storage solutions.

Performance

- Decentralized storage can reduce latency in data retrieval, enhancing the responsiveness of AUIs.

User Experience

- Improved data availability and reliability directly enhance the quality of personalized user experiences delivered by the AUI.

Challenges and Considerations

Network Latency

- Although IPFS can reduce latency, network configurations and node locations can affect performance. Optimizing node

placement is crucial.

Data Consistency

- Ensuring data consistency across IPFS nodes can be challenging. Implementing robust consensus mechanisms is necessary to maintain consistency.

Integration Complexity

- Integrating IPFS with existing AUI systems requires careful planning and execution to ensure seamless operation.

Conclusion

Adaptive user interfaces (AUIs) significantly improve from the static user experience methods within different domains. As complex data gathering and analysis systems, AUIs can adapt the content, layout, and functionality of products to the user's requirements and patterns of use. The flexibility created in this case leads to higher satisfaction, productivity, and accessibility rates for users and, therefore, higher user retention. While the environment of today's applications is complex, privacy is paramount, and the overall effort of implementing each AUI is understandably costly, the advantages remain apparent. Future developments for AUIs include the highly prevalent implementation of AI, cross-platform adaptability, emotional and behavioral adaptability, and privacy-shield mechanisms, making AUIs more precise and sensitive to user needs. With the advancement of technology, AUIs will play an even more crucial role in delivering user-centered experiences in the digital application. Through repeatability and analysis of these systems, developers can effectively engage and enhance the AUIs to make the digital spaces more effective [42-45].

References

1. Bucklin RE, Sismeiro C (2003) A model of web site browsing behavior estimated on clickstream data. Journal of Marketing Research 40: 249-267.
2. Dinev T, Hart P (2006) An extended privacy calculus model for e-commerce transactions. Information Systems Research 17: 61-80.
3. Chittaro L, Montanari A (2003) Evaluating dynamic user interfaces with large user models. International Journal of Human-Computer Studies 58: 533-555.
4. Dey AK, Abowd GD (2000) Towards a better understanding of context and context-awareness. Proceedings of the 2000 Conference on Human Factors in Computing Systems 79-80.
5. Nguyen DT, Hui PC, Harper FM, Terveen L, Konstan JA (2007) Exploring the filter bubble: The effect of using recommender systems on content diversity. Proceedings of the 18th ACM Conference on Hypertext and Hypermedia 11-20.
6. Ghahramani Z (2001) An introduction to hidden Markov models and Bayesian networks. International Journal of Pattern Recognition and Artificial Intelligence 15: 9-42.
7. MacQueen J (1967) Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability 1: 281-297.
8. Collins LM, Lanza ST (2010) Latent class and latent transition analysis: With applications in the social, behavioral, and health sciences. John Wiley & Sons.
9. Cortes C, Vapnik V (1995) Support-vector networks. Machine Learning 20: 273-297.
10. Breiman L (2001) Random forests. Machine Learning 45: 5-32.
11. Draper NR, Smith H (1998) Applied regression analysis. John Wiley & Sons 326.
12. Jolliffe IT (2002) Principal component analysis (2nd ed.). Springer Series in Statistics https://link.springer.com/book/10.1007/b98835.
13. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, et al. (2015) Human-level control through deep reinforcement learning. Nature 518: 529-533.
14. Konstan JA, Miller BN, Maltz D, Herlocker JL, Gordon LR, et al. (1997) GroupLens: Applying collaborative filtering to Usenet news. Communications of the ACM 40: 77-87.
15. Burke R (2002) Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction 12: 331-370.
16. Joachims T (2002) Optimizing search engines using clickthrough data. Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 133-142.
17. Grigoras R, Fratu O, Moldoveanu F (2015) Performance analysis of lazy loading mechanism for mobile web applications. Procedia Computer Science 52: 1261-1266.
18. Johnson H, Johnson P, Waddington R (2003) Task-related knowledge structures: Analyzing knowledge structures to inform system design. International Journal of Human-Computer Studies 58: 515-545.
19. (2013) React: A JavaScript library for building user interfaces. Facebook https://reactjs.or g/.
20. Lewis J, Fowler M (2014) Microservices: A definition of this new architectural term. MartinFowler.com https://martinfowler.com/articles/microservices.html.
21. (2015) GraphQL: A data query language. Facebook https://graphql.org/.
22. Kreps J, Narkhede N, Rao J (2011) Kafka: A distributed messaging system for log processing. Proceedings of the NetDB 11: 1-7.
23. Kohavi R, Henne RM, Sommerfield D (2009) Practical guide to controlled experiments on the web: Listen to your customers not to the HiPPO. Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 959-967.
24. Bifet A, Gavaldà R (2007) Learning from time-changing data with adaptive windowing. Proceedings of the 2007 SIAM International Conference on Data Mining 443-448.
25. Brusilovsky P, Millán E (2007) User models for adaptive hypermedia and adaptive educational systems. Adaptive Web 3-53.
26. Gajos KZ, Czerwinski M, Tan DS (2006) Adaptive Graphical User Interfaces: The Reality of Today and the Promise of Tomorrow. CHI '06 Extended Abstracts on Human Factors in Computing Systems 1310-1313.
27. (2016) The value of getting personalization right—or wrong—is multiplying. McKinsey & Company https://www.mckinsey.com/business-functions/marketing-and-sales/our-insights/the-value-of-getting-personalization-right-or-wrong-is-multiplying.
28. (2018) Personalization Pulse Check. Accenture https://www.accenture.com.
29. Hearst MA (2011) Search user interfaces. Cambridge University Press https://dl.acm.org/doi/10.5555/1631268.
30. Acquisti A, Brandimarte L, Loewenstein G (2015) Privacy and human behavior in the age of information. Science 347: 509-514.
31. Dean J, Ghemawat S (2008) MapReduce: Simplified data processing on large clusters. Communications of the ACM 51: 107-113.
32. Parasuraman R, Riley V (1997) Humans and automation: Use,

misuse, disuse, abuse. Human Factors 39: 230-253.

33. Bakshy E, Messing S, Adamic LA (2015) Exposure to ideologically diverse news and opinion on Facebook. Science 348: 1130-1132.
34. Blumenthal D, Tavenner M (2010) The "meaningful use" regulation for electronic health records. New England Journal of Medicine 363: 501-504.
35. Mozer MC, Vidmar L, Dodier RH (2010) The neurothermostat: Adaptive control of residential heating systems. In Advances in Neural Information Processing Systems 929-936.
36. Mayer-Schönberger V, Cukier K (2013) Big data: A revolution that will transform how we live, work, and think. Houghton Mifflin Harcourt https://books.google.co.in/books/about/Big_Data.html?id=uy4lh-WEhhIC&redir_esc=y.
37. Domingos P (2012) A few useful things to know about machine learning. Communications of the ACM 55: 78-87.
38. Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT Press https://mitpress.mit.edu/9780262035613/deep-learning/.
39. Nielsen J (2013) Mobile Usability. New Riders https://www.nngroup.com/books/mobile-usability/.
40. Picard RW (1997) Affective Computing. MIT Press https://mitpress.mit.edu/9780262661157/affective-computing/.
41. Dwork C (2008) Differential privacy: A survey of results. Proceedings of the 5th International Conference on Theory and Applications of Models of Computation 1-19.
42. Ohm Patel (2019) Building Data Replication System Replication System IPFS Nodes Cluster. International Journal of Science and Research (IJSR) 8: 2057-2069.
43. Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining 226-231.
44. Finstad K (2006) The system usability scale and usability testing: A methodological review. Journal of Usability Studies 4: 89-92.
45. Gaunt M (2014) IndexedDB - The Devil is in the details. Web.devhttps://developers.google.com/web/ilt/pwa/working-with-indexeddb.