Journal of Scientific & Industrial Research
Vol. 82, November 2023, pp. 1184-1192
DOI: 10.56042/jsir.v82i11.3367

Uncovering Issues Impacting Mobile Banking Performance using ISM-MICMAC Approach

Mangesh Joshi
Shri Ramdeobaba College of Engineering and Management, Nagpur 440 013, Maharashtra, India

Received 28 June 2023; revised 12 September 2023; accepted 22 September 2023

Every day, a multitude of mobile apps are released or updated, resulting in millions of daily downloads, usage, and views. This creates a fascinating phenomenon of user community acceptance and rating of these apps. Some apps are well-received by users due to factors such as performance, compatibility, and cost. This study has examined eleven factors that affect the performance of mobile banking apps like Google Pay, Phone Pay, and Paytm. Factors are identified by conducting a literature survey, analysing user reviews, and seeking expert opinions. It is worth noting that users tend to reject or dislike apps that pose challenges to them due to issues in the apps. Furthermore, the authors have utilized an Interpretive Structure Modeling (ISM) approach to develop a hierarchical structure for improvement of individual factors, along with MICMAC (Matrice d'Impacts Croisés Multiplication Appliquée á un Classment) analysis, to categorize the identified issues into four groups. Numerous studies have addressed issues related to mobile apps, but the classification or grouping of these issues has often been inadequate. In divergence, this particular research delivers a well-organized classification of issues associated with mobile banking applications. The issues are grouped into appropriate categories.

Keywords: App issues, Mobile apps, Mobile banking apps, Mobile banking applications, User community

Introduction

Mobile applications have entered our day to day activities.[1] From ordering food, booking a taxi, to checking the weather, mobile apps have made our lives easier and more convenient. However, as with any technology, mobile apps are not immune to issues.[2] These issues can range from minor bugs and glitches to serious security breaches that can compromise user data.[3] On the other hand, Firms for app development are less in number/size.[4] The absence of a particular scientific method being employed by app developers is a significant contributing factor to the emergence of issues at various stages of app development.[5] Recent research has placed a growing emphasis on software engineering concerns within the context of mobile applications. For instance, the Mining Software Repositories (MSR) challenge has facilitated access to Android platform change and bug report data, prompting researchers to discover new insights regarding this platform.[6] Other studies have investigated challenges associated with code reuse and testing in mobile app development.[7]

One of the most common mobile app issues is platform-specific bugs.[8] Mobile apps can be developed for different platforms such as Android, iOS, and Windows, and each platform has its own set of bugs that can cause issues with the app.[9] For example, an app that works perfectly on an Android device may encounter issues on an iOS device. To address this issue, developers must test their apps on different platforms to ensure that they work seamlessly across all platforms. Another issue that can affect mobile app performance is device fragmentation. There are thousands of different types of mobile devices with varying screen sizes, hardware, and software configurations. This can create compatibility issues with the app.[10] For example, an app that is optimized for a specific screen size may not display properly on a device with a different screen size. To address this issue, developers must ensure that their apps are optimized for different device configurations. Network issues are also a common cause of mobile app issues. Mobile apps rely on internet connectivity to function properly. Poor network conditions such as low bandwidth, network congestion, and packet loss can cause issues with the app. To address this issue, developers can optimize their apps to work well even under poor network conditions. For example, apps can be designed to work offline or in a low bandwidth mode. App updates can also cause issues with mobile apps.[11] App

*Author for Correspondence
E-mail: joshimp@rknec.edu

updates can introduce new bugs or cause compatibility issues with other apps installed on the device. To address this issue, developers must thoroughly test their updates before releasing them to the public. Additionally, users must ensure that they update their apps regularly to ensure that they are running the latest version. User behaviour can also cause issues with mobile apps. User behaviour such as multitasking, switching between apps, and interrupting app processes can cause issues with the app.[12] For example, if a user receives a phone call while using an app, the app may crash or freeze. To address this issue, developers can design their apps to handle interruptions and multitasking. Battery life is another issue that can affect mobile app performance.[13] Mobile apps can drain the device's battery, and if the battery is low, it can cause issues with the app. To address this issue, developers can optimize their apps to minimize battery usage. Additionally, users can conserve battery life by closing apps that are not in use.

Security issues are a major concern for mobile apps.[14] Mobile apps can be vulnerable to security breaches, which can cause issues such as data loss, unauthorized access, and malware infections.[15] To address this issue, developers must ensure that their apps are secure by implementing encryption, authentication, and other security measures. Additionally, users must be cautious when installing apps and only download apps from trusted sources. App design and development issues such as poor coding practices, inadequate testing, and insufficient user feedback can cause issues with the app. To address this issue, developers must ensure that they follow best practices[9] for app design and development. This includes testing the app thoroughly and soliciting user feedback to identify and address issues. User error is another common cause of mobile app issues. User error such as incorrect input, accidental deletions, and forgetting login credentials can cause issues with the app.[16] Additionally, users must be cautious when using apps and take care to input data correctly and store login credentials securely. Mobile apps often integrate with third-party services such as social media platforms, payment gateways, and analytics tools. Issues with third-party services can affect the functionality of the app. To address this issue, developers must ensure that their apps are compatible with third-party services and monitor their integration closely to identify and address issues.

Mobile app issues can have a significant impact on user experience and app performance. Developers must take a proactive approach to address these issues by testing their apps thoroughly, optimizing their apps for different platforms and device configurations, implementing security measures, and soliciting user feedback. By addressing these common mobile app issues, developers can ensure that their apps are reliable, secure, and user-friendly.

Why Interpretive Structural Modeling?

Interpretive Structural Modeling (ISM) is a technique that enables the development of a hierarchical model representing the inter-relationships among various factors in a complex system. The technique[17] was first proposed by Warfield (1974) and has since been used in numerous applications. The process involves a series of steps that help to identify the relationships between different elements of the system, and to identify the most important elements that affect the system's overall performance. ISM has been used in various domains to analyze complex systems. One of the key benefits of using ISM is that it can help researchers to identify the key drivers of a system's performance. By identifying these drivers, researchers can better understand the factors that are most important in determining the overall performance of a system. Recent studies have demonstrated the utility of ISM in a range of domains.[18] As each study identifies different significant factors, it is important to establish the relationship between these factors.

The structure of this paper consists of two primary components: firstly, the identification of issues that have a bearing on the success or failure of apps, and secondly, the utilization of ISM to reveal and explore contextual relationships among these issues. Therefore, ISM has been applied in prioritizing the performance factors in the further part of this paper.

Literature Review

The literature review was carried out in two parts for factor identification that impacts the performance of mobile apps. In the first part, a research papers were identified using scholarly articles available at science direct, Web of Science (WoS) database, and other Scopus indexed journals. The search utilized words/ or combination of words /phrases such as 'performance', 'app issues', 'app development', 'app performance', 'app design', and 'app factors considerations'. Through this review, a list of contributing factors (CFs) was identified.

In the second part, latest 5000 reviews were gathered on various mobile banking applications from Google Play Store and amassed approximately 50,000 user reviews across 10 mobile banking applications were collected as a sample. After data cleansing the text, we determined that only 30–50% of the reviews contained useful information. We clustered similar types of complaints to create problem areas. Finally, a discussion was held with a group of eight experts, including five academic experts and three industry experts in hybrid mode. This discussion led to the finalization of eleven relevant classified CFs from the list of eighteen initially identified factors for this research presented in Table 1.

Methodology

The ISM approach is based on the assumption that the components of a system are interdependent and that their relationships can be modeled as a hierarchy of levels. Each level represents a set of components that are related to each other and have a similar level of importance within the system. The components at the top of the hierarchy are the most important, while those at the bottom are the least important. The construction of a logical mental diagram involves a systematic procedure18 which is as follows:

1. Identify factors that are relevant to the specific problem through an extensive literature survey. Suggestions from practicing professionals are taken for finalizing the final list of factors.
2. Establish contextual relationships among the finalized contributing factors.
3. Develop a Structural Self-Interaction Matrix (SSIM) that shows the pair wise relationship between the problem factors.

Table 1 — Details of factors identified after literature review

S. N	Factor related with	Brief Description	Consisting of
1	Data safety/Privacy[19,27]	The application may endanger the belongings of the user.	1. Security breaches 2. Data loss, Malware infections. 3. Unauthorized access
2	Hardware/software Compatibility[19,27,29]	There may be issues due to hardware-software compatibility.	1. Devices with varying screen sizes 2. Different hardware 3. Different software configurations.
3	Performance/Service[19,21,29]	Main performance factors are to be monitored for better user satisfaction.	1. Battery life, Heating issues 2. Speed 3. Consuming memory space
4	Consistency/Stability[19,21,25]	It refers to the ability of an application to behave in a predictable and uniform manner.	1. Runtime/ Dynamic errors 2. Lacking improvement after updates 3. Network connectivity
5	User interface[19,20,22]	User interface (UI) refers to the design and layout of the visual elements that a user interacts with.	1. UI problem 2. Unappealing UI 3. Too many ads
6	Features/Functionality[20,21, 24,28,29]	It refers to the specific capabilities and tasks that the app is designed to perform or enable for the user.	1. Uninteresting content 2. Language support 3. Feature addition/deletion
7	Start-up/Session[24,27,29]	This refers to problems appearing during application start up and during the session.	1. Connections and sync issue 2. Session crash/ auto logout issues 3. Login/ OTP issues
8	User Behavior[21,24]	It refers to the actions and interactions that users perform within the app, including their patterns of usage and engagement.	1. User Intentions 2. User parallel activities 3. Users' Know-how
9	Feedback / Grievance Handling[20–22]	It describes the mechanism of obtaining feedback, reviews from the customer and its handling.	1. Missing notifications 2. User Rating 3. Unhandled complaints
10	Cost[26,27]	This refers to purchase price of app and associated benefits user gets. Hidden costs or any recurring cost.	1. Open-source app 2. Paid app 3. Hidden recurring cost
11	App Testing and Service[21,23]	It refers to the processes and activities involved in testing the app's functionality, performance, usability, and security, as well as providing ongoing maintenance and support to ensure optimal user experience.	1. Bugs/Errors 2. Lack of functionality of some features 3. Maintenance

4. Using SSIM, a reachability matrix was formed and checked for transitivity issues. The term ‘Transitivity’ refers to dependence of one factor over another and another factor over many more factors. So the first factor in consideration will indirectly influence the other dependent factors.
5. Partition the reachability matrix into different levels.
6. Draw a directed graph (DIAGRAPH) based on the relationships in the reachability matrix and remove transitive links.
7. Replace factor nodes with statements to convert the resulting diagraph into an ISM.
8. Due to the complex structure of the problem, conceptual inconsistencies may arise. Therefore, reviewing is necessary to identify and make any necessary modifications to the ISM.

Structural Self-Interaction Matrix (SSIM)

To generate SSIM matrix, experts’ view was collected. This is done to map the relationships between the factors in consideration. The developed front end for data collection is shown in Fig. 1. Experts were asked to compare a single factor with another factor and indicate the relationship between them by pressing one of four buttons as shown. These relationships were then analysed using four variables to determine the direction of the relationship between the two factors (i and j) in the development of SSIM. The direction of the relationship between two factors (i and j) in the development of SSIM is determined using four variables. ‘V’ indicates that ith factor tries to achieve jth factor; A, indicates that jth factor tries to achieve ith factor; X, which indicates that I and j will help to improve each other; and O, indicate no relation between the factors. Final SSIM matrix prepared is shown in Table 2.

Reachability Matrix

To transform SSIM into a two-dimensional matrix known as the reachability matrix, V, A, X, and O were replaced with either 1 or 0 based on specific conditions. The substitution of 1 and 0 followed the following rules:

(i) If CellSSIM(i, j) == ‘V’, then CellRM (i, j) is set to 1 and CellRM (j, i) is set to 0.

(ii) If CellSSIM(i, j) == ‘A’, then CellRM (i, j) is set to 0 and CellRM (j, i) is set to 1.

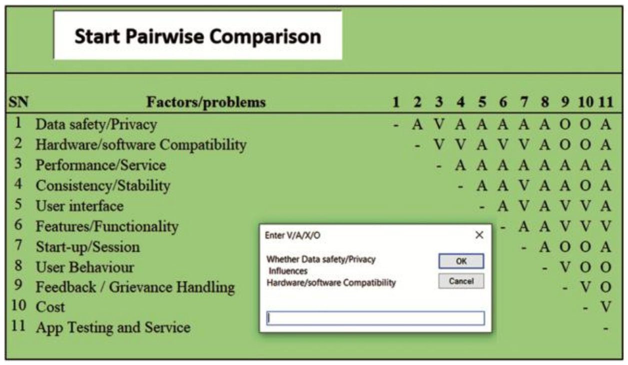
(iii) If CellSSIM(i, j) == ‘X’, then CellRM (i, j) is set to 1 and CellRM (j, i) is set to 1.

(iv) If CellSSIM(i, j) == ‘O’, then CellRM (i, j) is set to 0 and CellRM (j, i) is set to 0.

According to the aforementioned guidelines, the initial reachability matrix (IRM) was constructed as in Table 3. The final version of RM as in Table 4 was generated after removing indirect links. Furthermore, Table 4 exhibits the driving and dependence power for each factor. The driving power refers to the control it has over entire factors.

Level Partitions

The reachability set was formed where there is 1 in the ith row and the antecedent set was prepared having 1 in jth column. The common factors in the



SN	Factors/problems	1	2	3	4	5	6	7	8	9	10	11
1	Data safety/Privacy	-	A	V	A	A	A	A	A	O	O	A
2	Hardware/software Compatibility		-	V	V	A	V	V	A	O	O	A
3	Performance/Service			-	A	A	A	A	A	A	A	A
4	Consistency/Stability				-	A	A	V	A	A	O	A
5	User interface					-	A	V	A	V	V	A
6	Features/Functionality						-	A	A	V	V	V
7	Start-up/Session							-	A	O	O	A
8	User Behaviour								-	V	O	O
9	Feedback / Grievance Handling									-	V	O
10	Cost										-	V
11	App Testing and Service											-

Whether Data safety/Privacy Influences Hardware/software Compatibility

Fig. 1 — Front End used for data collection

Table 2 — Structural self-interaction matrix (SSIM)

S. N	Factors/problems	1	2	3	4	5	6	7	8	9	10	11
1	Data safety/Privacy	—	A	V	A	A	A	A	A	O	O	A
2	Hardware/software Compatibility		—	V	V	A	V	V	A	O	O	A
3	Performance/Service			—	A	A	A	A	A	A	A	A
4	Consistency/Stability				—	A	A	V	A	A	O	A
5	User interface					—	A	V	A	V	V	A
6	Features/Functionality						—	A	A	V	V	V
7	Start-up/Session							—	A	O	O	A
8	User Behaviour								—	V	O	O
9	Feedback / Grievance Handling									—	V	O
10	Cost										—	V
11	App Testing and Service											—

Table 3 — Initial reachability matrix (IRM)

S N	**Factors/problems**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**DR.P.**
1	Data safety/Privacy	1	0	1	0	0	0	0	0	0	0	0	**2**
2	Hardware/software Compatibility	1	1	1	1	0	1	1	0	0	0	0	**6**
3	Performance/Service	0	0	1	0	0	0	0	0	0	0	0	**1**
4	Consistency/Stability	1	0	1	1	0	0	1	0	0	0	0	**4**
5	User interface	1	1	1	1	1	0	1	0	1	1	0	**8**
6	Features/Functionality	1	0	1	1	1	1	0	0	1	1	1	**8**
7	Start-up/Session	1	0	1	0	0	1	1	0	0	0	0	**4**
8	User Behaviour	1	1	1	1	1	1	1	1	1	0	0	**9**
9	Feedback / Grievance Handling	0	0	1	1	0	0	0	0	1	1	0	**4**
10	Cost	0	0	1	0	0	0	0	0	0	1	1	**3**
11	App Testing and Service	1	1	1	1	1	0	1	0	0	0	1	**7**
	DEPENDENCE POWER	**8**	**4**	**11**	**7**	**4**	**4**	**6**	**1**	**4**	**4**	**3**	**56**

Table 4 — Final reachability matrix (FRM)

S N	**Factors/problems**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**DR.P.**
1	Data safety/Privacy	1	0	1	0	0	0	0	0	0	0	0	2
2	Hardware/software Compatibility	1	1	1	1	1	1	1	0	1	1	1	10
3	Performance/Service	0	0	1	0	0	0	0	0	0	0	0	1
4	Consistency/Stability	1	0	1	1	0	1	1	0	0	0	0	5
5	User interface	1	1	1	1	1	1	1	0	1	1	1	10
6	Features/Functionality	1	1	1	1	1	1	1	0	1	1	1	10
7	Start-up/Session	1	0	1	1	1	1	1	0	1	1	1	9
8	User Behaviour	1	1	1	1	1	1	1	1	1	1	1	11
9	Feedback / Grievance Handling	1	0	1	1	0	0	1	0	1	1	1	7
10	Cost	1	1	1	1	1	0	1	0	0	1	1	8
11	App Testing and Service	1	1	1	1	1	1	1	0	1	1	1	10
	DEPENDENCE POWER	10	6	11	9	7	7	9	1	7	8	8	**83**

Table 5 — Level partitioning

S N	Factors/problems	Reachability Set	Antecedent Set	Level
1	Data safety/Privacy	1,3	1,2,4,5,6,7,8,9,10,11	II
2	Hardware/software Compatibility	1,2,3,4,5,6,7,9,10,11	2,5,6,8,10,11	VI
3	Performance/Service	3	1,2,3,4,5,6,7,8,9,10,11	I
4	Consistency/Stability	1,3,4,6,7	2,4,5,6,7,8,9,10,11	III
5	User interface	1,2,3,4,5,6,7,9,10,11	2,5,6,7,8,10,11	VI
6	Features/Functionality	1,2,3,4,5,6,7,9,10,11	2,4,5,6,7,8,11	VI
7	Start-up/Session	1,3,4,5,6,7,9,10,11	2,4,5,6,7,8,9,10,11	III
8	User Behaviour	1,2,3,4,5,6,7,8,9,10,11	8	VII
9	Feedback / Grievance Handling	1,3,4,7,9,10,11	2,5,6,7,8,9,11	V
10	Cost	1,2,3,4,5,7,10,11	2,5,6,7,8,9,10,11	IV
11	App Testing and Service	1,2,3,4,5,6,7,9,10,11	2,5,6,7,8,9,10,11	IV

reachability set and antecedent set is then computed for all the factors which is called as intersection set. The ith factor having similar elements in reachability and intersection sets are assigned a level I. Later, the ith factor is omitted and again intersection set is formed. The same procedure is repeated until all the factors are portioned into appropriate levels. The initial iteration of level partitioning results in level I is demonstrated in Table 5. These levels aid in the construction of the diagraph and the final ISM model.

Formation of ISM

The data in the Table 5 contain the level partition tables used to generate the ISM (Fig. 2). All the factors are joined as per their relationship with the other factor.

Micmac Analysis

The MICMAC (Matrice d'Impacts Croisés Multiplication Appliquée á un Classement or Cross-Impact Matrix Multiplication Applied to Classification) analysis is employed to identify and evaluate the driving power and dependence of the

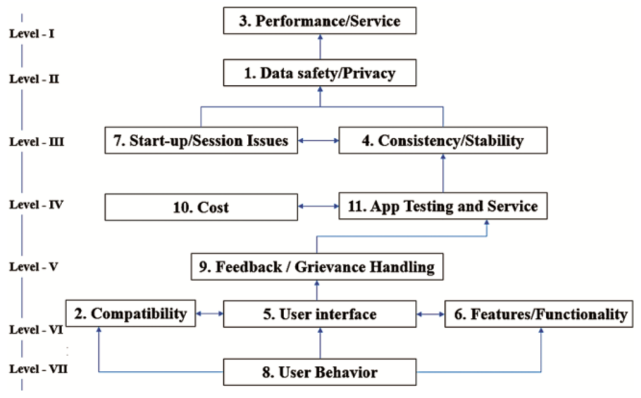


Fig. 2 — ISM Model

factors that contribute to the problem or ultimate goal. According to the MICMAC principle, which is based on matrix multiplication properties, if element X has a direct influence on element Y represented by (x -> Y), and Y -> Z then indirectly X -> Z.

The MICMAC analysis classifies the factors into four categories like Autonomous, dependent, Linkage and Driving as illustrated in Fig. 3.

Results and Discussion

The hardware and software that are not designed with adequate security features can put sensitive data at risk of unauthorized access, theft, or hacking. In order to ensure data safety and privacy, it may be necessary to use specific hardware and software that have been designed with security in mind. On the other hand, compatibility issues between hardware and software can also affect data safety and privacy. If a hardware device or software application is not compatible with the system in which it is being used, it could potentially cause security vulnerabilities or data loss. This is why it's important to ensure that any hardware or software being used is compatible with the system and meets any necessary security requirements.

In terms of performance and service, systems that prioritize data safety and privacy may have additional security features or protocols in place that could impact performance. For example, encryption can add an additional layer of security but may also slow down data processing speeds. On the other hand, systems that prioritize performance and service may not have adequate security measures in place, which could lead to data breaches or other security risks. Furthermore, data safety and privacy can also impact service quality. For example, a system that prioritizes data privacy may require additional steps for user

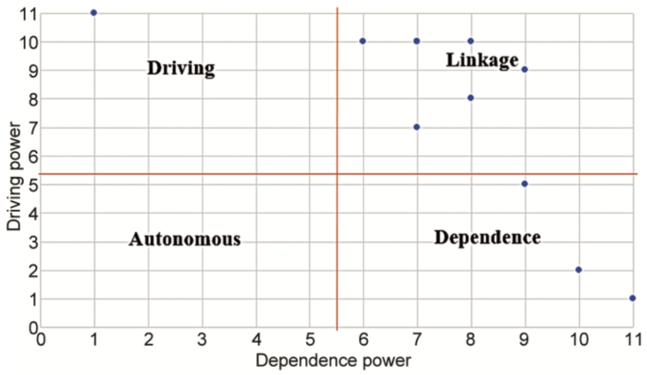


Fig. 3 — MICMAC analysis

verification or authentication, which could potentially increase wait times and decrease user satisfaction. However, if data privacy is not prioritized, users may be at risk of identity theft or other security breaches, which would negatively impact service quality. Overall, data safety and privacy and performance/ service are interdependent and both need to be carefully considered in order to ensure a secure and effective system that meets the needs of users. It's important to find a balance between these competing priorities to create a system that provides both adequate security and optimal performance/service.

In terms of consistency, systems that prioritize data safety and privacy may have additional checks and verifications in place to ensure that data is accurate and consistent. This could impact the speed and performance of the system, as additional steps may be necessary to ensure data consistency. However, ensuring consistency is important for maintaining the integrity of the data and reducing the risk of errors or discrepancies. Similarly, data safety and privacy can also impact the stability of a system. If a system is not designed with adequate security measures, it could be vulnerable to hacking, data breaches, or other security risks that could impact the stability of the system. On the other hand, a system that prioritizes data safety and privacy may require additional resources or maintenance to maintain the security of the system, which could also impact stability. Overall, data safety and privacy and consistency/stability are interrelated and both need to be carefully considered in order to ensure a secure and stable system that provides accurate and consistent data. It's important to find a balance between these priorities to create a system that is both secure and stable while also providing accurate and consistent data.

When designing UI, it is important to consider the potential privacy and security risks associated with the collection and processing of user data. This can impact the design decisions in several ways. The user interface should be designed in a way that clearly communicates to the user what data is being collected and how it will be used. For example, the user should be informed about the types of data being collected, how long the data will be retained, and whether it will be shared with third parties. The user experience should be designed in a way that minimizes the collection of unnecessary data and ensures that sensitive data is collected and processed securely. For example, the system may use encryption to protect sensitive data or require strong passwords to ensure that only authorized users have access. The user should be given control over their data privacy through privacy settings. This can include options to opt-out of certain data collection or to delete data that has already been collected. Designing a UI with data safety and privacy in mind can help to build trust with users and ensure that the system is compliant with relevant regulations.

The ISM technique is comprehensible to a diverse set of users across interdisciplinary teams. It offers a means of unifying varying viewpoints among the participating groups, and is capable of handling the many components and relationships inherent in complex systems. Its heuristic evaluation of model formulation adequacy yields valuable insights into system behaviour. Furthermore, the ISM method is user-friendly and available to a wider audience. Due to these characteristics, the ISM approach has gained widespread adoption.

When software developers and designers consider ways to improve their work, a multitude of factors come into play. Determining which factors to prioritize can be challenging due to the various constraints within the organization. To address this issue, the authors of this study utilized the ISM methodology and MICMAC analysis to examine the crucial performance related aspects of mobile apps. The goal is to achieve effective and efficient performance improvement and design. After conducting a thorough literature review, eleven factors were identified and an inter-relationship model is developed.

The data in the Table 3 shows the Reachability Matrix without transitivity, accounting for driving and dependence power. The factors with the highest driving power are 'User behaviour'. It is followed by 'Hardware/software Compatibility', 'Features/ Functionality', 'Start-up/Session', and 'App Testing and Service'. Factors with greater driving power have a stronger influence on other system factors. Improving driving factors can positively impact the contribution of other related factors. On the other hand, factors with high dependence power, such as 'Performance/Service', or 'Data safety/Privacy ', have a stronger impact on other factors. Changes or improvements to these factors may have little effect on the other factors.

The data in the Fig. 3 and Table 6 illustrates that no factors is located in Cluster I, which includes autonomous factors with low driving and dependence power. These factors can be considered separate and gets influenced. Cluster II comprises the factors such as 'Consistency/Stability', 'Data safety/Privacy', and 'Performance/Service' which are dependent factors. The critical success factors fall within Cluster III are 'Hardware/software Compatibility', 'User interface', 'Features/Functionality', 'Start-up/Session issues', 'Feedback', 'Cost', and 'App Testing and Service'. Linkage factors consists of both strong driving and dependence power. These factors are unstable in that any changes to them will affect other factors and result in a feedback effect on themselves. Remaining factor i.e. 'User Behaviour' is classified in cluster IV. The critical success factors' substantial driving power and limited dependence necessitate treating them as vital to success. Policymakers and software developers should prioritize addressing these critical success factors from Level VII – Level I as depicted in Fig. 2 to attain a substantial performance development in the current scheme.

The eleven factors were ranked using a simple formula that divides a factor's driving power by its dependence power. A higher ranking is achieved with more driving power and less dependence. Table 7 displays the rankings after the MICMAC analysis. Ranking of eleven factors is found to be as follows:

Table 6 — Cluster formation as per FRM

Cluster No	Clusters	Factors
I	Autonomous	—
II	Dependence	Consistency/Stability, Datasafety/Privacy, Performance/Service
III	Linkage	Hardware/software Compatibility User interface Features/Functionality Start-up/Session issues Feedback Cost App Testing and Service
IV	Driving	User Behaviour

Table 7 — MICMAC ranking of the factors

Factor	Factors/problems	Dependence power	Driving power	Driving/ Dependence power	MICMAC Rank
F8	User Behavior	1	11	11.00	1
F2	Hardware/software Compatibility	6	10	1.67	2
F5	User interface	7	10	1.43	3
F6	Features/Functionality	7	10	1.43	3
F11	App Testing and Service	8	10	1.25	5
F7	Start-up/Session	9	9	1.00	6
F9	Feedback / Grievance Handling	7	7	1.00	6
F10	Cost	8	8	1.00	6
F4	Consistency/Stability	9	5	0.56	9
F1	Data safety/Privacy	10	2	0.20	10
F3	Performance/Service	11	1	0.09	11

F8>F2>F5>F6>F11>F7>F9>F10>F4>F1>F3

Conclusions

This study utilized an extensive literature review and consultation with experts to determine the influencing factors and formulate an interpretive structural model of eleven Critical Success Factors (CSFs) for evaluating mobile app performance issues. The proposed model effectively displays the interrelationships among these CSFs and can aid decision makers in understanding their relative importance for effective software development. Based on the model's hierarchy diagram, the User behaviour is the most important as it drives other CSFs and aids in strategic planning. It is recommended that policy makers and app developing authorities prioritize the factors to improve upon the app performance. The results of this analysis can play a crucial role in enhancing the quality of mobile applications and their adoption among the app-user community. While the paper itself does not suggest a particular strategy for developing apps, software developers can still leverage the findings outlined therein to enhance their overall app development processes. One way they could do this is by conducting a correlation analysis that compares issue types to mobile app ratings, which would help them gauge the significance of each.

A case study with real industry data can be conducted to improve performance and thereby validating the current study. Validation is beyond the scope of this paper serving as limitation of current study and scope for further research.

References

1 Vaupel S, Taentzer G, Gerlach R & Guckert M, Model-driven development of mobile applications for Android and iOS supporting role-based app variability, *Proc- Int Conf Softw LNI*, (2016), 99–100.

2 Kathuria A & Gupta A, Challenges in Android application development: A case study, *Int J Comput Sci Mob Computing*, **4(5)** (2015) 294–299.

3 Zahra S, Khalid A & Javed A, An efficient and effective new generation objective quality model for mobile applications, *Int J Mod Educ Comput Sci*, **4** (2013) 36–42.

4 Flora H K, Wang X & Chande S V, An investigation on the characteristics of mobile applications: A survey study, *Int J Inf Technol Comput Sci*, **11** (2014) 21–27.

5 Corral L, Sillitti A & Succi G, Software assurance practices for mobile applications, *Computing*, **97(10)** (2015) 1001–1022.

6 Shihab E, Kamei Y & Bhattacharya P, Mining challenge 2012: The android platform, *Proc Int Work Conf Min Softw Repos*, (2012) 112–115.

7 Ruiz I J M, Nagappan M, Adams B & Hassan A E, Understanding reuse in the Android market, *Int Conf Prog Compre*, (2012) 113–122.

8 Adipat B & Zhang D, Interface design for mobile applications, *Con Inf Syst*, (2005) 1–11.

9 Gatsou C, Politis A & Zevgolis A, The importance of mobile interface icons on user interaction, *Int J Comput Appl*, **9(3)** (2012) 92–107.

10 Necmiye N G & Abran A, A systematic literature review: Opinion mining studies from mobile app store user reviews, *J Syst Softw*, **125** (2017) 207–219.

11 Mcllroy S, Ali N, Khalid H & Hassan A E, Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews, *Empir Softw Eng*, **21(3)** (2016) 1067–1106.

12 Palomba F, Vasquez M L, Bavota G, Oliveto R, Penta M D, Poshyvanyk D & Lucia A D, User reviews matter! Tracking crowd sourced reviews to support evolution of successful apps, *IEEE Int Conf Soft Maint Evol*, (2015) 291–300.

13 Man Y, Gao C, Liu M R & Jiang J, Experience report: Understanding cross-platform app issues from user reviews, *IEEE Int Symp Softw Reliab Eng*, (2016) 138–149.

14 Khalid H, Shihab E, Nagappan M & Hassan A E, What do mobile app users complain about?, *IEEE Softw*, **32(3)** (2015) 70–77.

15 Perez B M, Diez I D L T & Coronado M L, Privacy and security in mobile health apps: A review and recommendations, *J Med Syst*, **39(1)** (2017) 181.

16 Bayomi H, Sayed N A, Hassan H & Wassif K, Application-based usability evaluation metrics, Int J Adv Comput Sci Appl, 13(7) (2022), doi: 10.14569/IJACSA.2022.0130712.

17 Warfield J, Developing interconnection matrices in structural modeling, *IEEE Trans Syst Man Cybern*, **4(1)** (1974) 81–87, https://doi.org/10.1109/TSMC.1974.5408524.

18 Joshi M P & Deshpande V, Application of interpretive structural modelling (ISM) for developing ergonomic workstation improvement framework, *Theor Issues Ergon Sci*, **24(2)** (2022) 1–23, https://doi.org/10.1080/1463922X.2022.2044932.

19 Man Y, Gao C, Lyu M R & Jiang J, Experience report: Understanding cross-platform app issues from user reviews, *IEEE 27th Int Symp Softw Reliab Eng,* (2016) 138–149.

20 Gao C, Zeng J, Lyu M R & King I, Online app review analysis for identifying emerging issues, *Proc - Int Conf Softw*, (2018) 48–58.

21 Gao C, Zheng W, Deng Y, Lo D, Zeng J, Lyu M R & King I, Emerging app issue identification from user feedback: Experience on We Chat, *Proc - Int Conf Softw*, (2019) 279–288, doi: 10.1109/ICSE-SEIP.2019.00040.

22 Chen Q, Chen C, Hassan S, Xing Z, Xia X & Hassan A E, How should I improve the UI of my app? a study of user reviews of popular apps in the Google Play, *ACM Trans Softw Eng Methodol*, **30(3)** (2021) 1–38, https://doi.org/10.1145/3447808.

23 Gao C, Zeng J, Lo D, Lin C Y, Lyu M R & King I, Infar: Insight extraction from app reviews, *Proc 2018 26th ACM Joint Meeting Eur Softw Eng Conf Symp Foundations Softw Eng ESEC/FSE,* (2018) 904–907.

24 Sayed N A, Hassan H, Wassif K & Bayomi H, Application-Based Usability Evaluation Metrics, *Int J Adv Comput Sci Appl*, **13(7)** (2022) 84–91, doi: 10.14569/IJACSA.2022.0130712.

25 Gao C, Wang B, He P, Zhu J, Zhou Y& Lyu M R, Paid: Prioritizing app issues for developers by tracking user reviews over versions, *IEEE 26th Int Symp Softw Reliab Eng*, (2015) 35–45.

26 Pandey M, Litoriya R & Pandey P, An ISM approach for modeling the issues and factors of mobile app development, *Int J Softw Eng Knowl Eng*, **28(07)** (2018) 937–953, https://doi.org/10.1142/S0218194018400119.

27 Gui J, Mcilroy S, Nagappan M & Halfond W G, Truth in advertising: The hidden cost of mobile ads for software developers, *IEEE/ACM 37th IEEE Int Conf Softw Eng*, **1** (2015), 100–110.

28 Xu X, Dutta K & Datta A, Functionality-based mobile app recommendation by identifying aspects from user reviews, *Proc Int Conf Inform Sys,* (2014) 1–10.

29 Pandey M, Litoriya R & Pandey P, Perception-based classification of mobile apps: A critical review, *Smart Comput Strategies: Theor Pract Aspects,* (2019) 121–133, https://doi.org/10.1007/978-981-13-6295-8_11.