JKAU: Comp. IT. Sci., **Vol**. **12** No. 2, pp: 17 – 29 (2023 A.D.)
DOI: 10.4197/Comp.12-2.2

Agile Recommendation System for Traditional Software Development Teams

Abdulrahman Alsari[1], Abdullah M. Algarni[1], Rizwan Qureshi[2], Abdullah Marish Ali[1]
Department of Computer Science[1], Department of Information Technology[2]
King Abdulaziz University
Jeddah, Saudi Arabia
aalsari0003@stu.kau.edu.sa, amsalgarni@kau.edu.sa, rmuhammd@kau.edu.sa, ammali@kau.edu.sa

***Abstract*—With the increasing demand and necessity for web and mobile applications, programmers are developing an increasing number of applications every day, and these applications have a high degree of deliverables. Therefore, developers must follow the highlighted software development processes as a standard practice. Traditional project management systems have shortcomings that must be addressed. While agile software development methodologies are widely used in the software industry, they are significantly more efficient and improve product quality if followed according to agile requirements. Traditional methods are not adaptable to changes and have defined life cycles. However, agile methods are more flexible in responding to changes, communicating with customers, and improving communication and collaboration. In this study, we present a recommendation system that uses a flow diagram to decide the appropriate agile methodology to be applied by teams based on the collected data after ensuring that the team is eligible for a change to agile. In addition, we presented a mobile application implemented to support the decision tree and project descriptions in which the tool was used. This leads to the greatest benefits from an agile methodology based on the business environment and customer requirements to improve the software industry in terms of cost, time, and other success factors.**

Keywords—Traditional methods; recommendation system framework; Agile methods; software development teams; project management

I. INTRODUCTION

Any new software or application requires a set of processes called the software development lifecycle (SDLC), which includes requirements elicitation, design and analysis, implementation, testing, and maintenance [1]. Without these activities, software developers cannot create software using scientific methodology. Therefore, developers use these processes and coding skills to build software that meets clients' expectations and needs while controlling the software project budget and knowing how the project is managed during its lifecycle. Then, the management type of software development, involving traditional and agile methods, plays a significant role in the IT business environment, products, and the organization structure.

Traditional methods are used to design and develop simple software or projects [2], and security is not a high priority. Traditional approaches are costly for small projects and teams because they require a large administrative overhead. This includes the time and effort required to maintain thorough documentation or plan every detail before proceeding. However, they have advantages in terms of knowledge of the project lifecycle and efficient implementation. Various traditional methods are available, including waterfall and Rational Unified Process (RUP) [3].

In software development, the term "agile" refers to a set of principles for solving design problems through effective collaborative methods. This challenge is solved through mature collaboration among diverse cross-functional and self-organizing teams. The English dictionary defines "agile" as the capacity to adjust to fast-changing technology, the software development industry, and customer requirements. This can be described as a gradual and iterative process that works directly with the end-user. Each iteration of this development process takes 1-3 weeks. Several inspections were required during this period before final approval and version release. This includes

several actions performed by cross-functional teams.

Agile is a catch-all term for a variety of methodologies such as extreme programming (XP), Scrum, Dynamic systems development method (DSDM), Crystal, Feature Driven Development (FDD), Lean Development, and Adaptive Software Development (ASD). These methodologies have many similarities, including short iterative life cycles, quick and frequent consumer feedback, and continuous learning. Scrum and XP/Scrum hybrids have been the most popular agile methodologies in the last decade.

Both traditional and agile methodologies define the workflow and organization of software development tasks. However, each method achieves the objectives differently: traditional methods sequentially employ a linear model (no overlap), whereas agile environments use iterations and short sprints (up to several weeks). Each sprint presents a set of deliverables that can be updated or rescheduled for the next sprint if they are unmet. The major goal of this strategy is to release functioning and complete components throughout all iterations [4], [5]. Furthermore, according to [6], several aspects such as cost, time, customer satisfaction, team capacity, and customer involvement contribute to the success of continuing agile software development projects.

Therefore, agile works best when the project's overall scope is not predetermined and/or cannot be adapted, iterations require greater user input (focused scenarios) and a team-based approach (therefore, more collaboration) to meet project constraints (E.g., budget, schedule). Iterative approaches allow for greater flexibility in dealing with requirements and modifications, though the level of difficulty increases during editing. Although agile development methods appear less focused on the project structure, traditional methods are frequently preferred. If team members are valued, the built-in flexibility enables changes to be made at any moment, resulting in faster error detection. The difficulty lies in avoiding the "spaghetti code" effect, which occurs after many iterations. However, if the project remains on track, it will have a mitigating effect because numerous components are provided in each sprint [5].

Current studies do not mention an automated tool that collects and analyzes user input to determine whether they are eligible for an agile environment and recommend the best method for switching to an agile environment. From the previous information, it becomes clear that it is important to have a recommendation system to transform the entire company from traditional methods to agile to obtain all the benefits during the software development lifecycle.

This study focuses on the proposed recommendation system framework to recommend traditional teams as the most suitable agile methodologies based on the collected data. In addition, we applied a mobile application as an implementation tool to support the proposed recommendation system by comparing some available projects and data descriptions via the application and determining whether the team is recommended to agile or not. This tool could support organizations with multiple teams to organize the most appropriate team and best utilize their capabilities.

The paper is organized as follows: Section 2 presents a brief background of the traditional and agile methods and their main properties and addresses some challenges while transforming organizations. Section 3 describes the research questionnaires and a survey report and presents the recommendation system, i.e., a flow diagram to decide the appropriate agile methodology to be applied by teams based on the collected data. Section 4 presents the tool implemented to support the decision tree and project descriptions in which the tool was used. Section 5 presents research challenges and limitations. Finally, Section 6 summarizes the conclusions and future work.

II. Background

Many different types of software development life cycle methodologies are utilized, depending on the nature and requirements of the project, and they fundamentally outline how the software development activity is organized. The traditional waterfall and agile software development processes are the two primary approaches. This is briefly detailed in the following sections.

A. Traditional Development Method

The waterfall is a well-known traditional methodology and document-driven method with

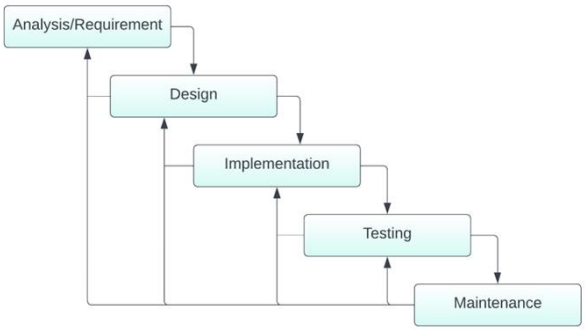

Fig. 1. Waterfall phases.

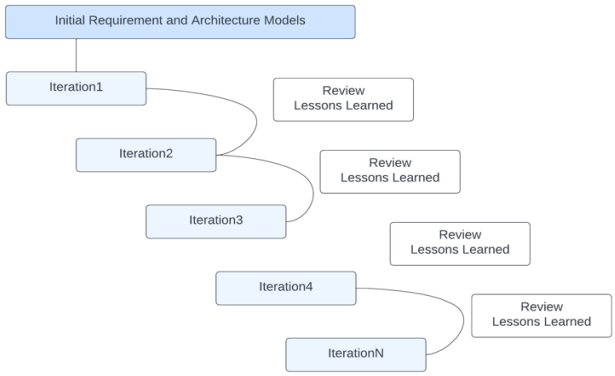

Fig. 2. Agile Development Life Cycle [1].

well-defined and sequential project phases [7]. In practice, a phase begins after the previous phase is completed (Fig. 1). It consists of four basic phases: requirement specification, software design, implementation, and testing [8]. As evidence of the strongly document-oriented approach of this method, it is sufficient to consider the following (phase, output) pairs: (user requirements, user requirements document), (software requirements, software requirements document), (software design, architectural design document), etc. [4]. Thus, the comprehensive planning of each phase facilitates the definition of deliverables and milestones. This is appropriate for implementing large and complex projects of critical systems with a high level of fault tolerance.

B. Agile Development Methods

The agile methodology is deemed more effective in projects where requirements change rapidly or where product field rivalry is heightened. New features must be added quickly because of its improved flexibility in implementing client requirements and frequent product releases [7], [9]. The agile model encourages the team to work simultaneously in different project phases (Fig. 2). Scrum, XP, and Lean are three prominent types of software development management in the agile model:

- Scrum: This emphasizes process adaptability and customer satisfaction through the speedy delivery of a functional software product [10]. The entire process of developing a product, from design to development, is based on a set of principles for controlling that development. The focus on business value (more than any other feature) and project is divided into packets with it testing and documentation. It mixes iterative and incremental process models to overcome the difficulties with standard models [11].
- Extreme Programming (XP): It is a lightweight, flexible, low-risk software development technique that can handle ambiguous or fast-changing requirements. [12]. XP focuses on code review and the intensive use of refactoring, and applications requiring onsite presence (colocation) often use this method because unit tests are executed daily. XP is designed for small to medium-sized teams and does not require large teams [13].
- Lean: The identification and elimination of process waste is the major emphasis and guiding principle of Lean to focus on producing customer value [14]. Lean is about getting more done with less [15] or producing in a third of the time, at a third of the cost, and with a third of the defect rate [16].

These three agile methodologies share many similarities, such as short iterative life cycles, quick and frequent client input, and continuous learning, but they also have many differences that distinguish them. For example, Scrum works in iterations that run for two to four weeks, whereas XP models work in iterations that last one to two weeks since XP is intended for faster deliverables and rapid feedback, and Lean emphasizes continuous improvement. In addition, no changes are permitted in Scrum after the completion of an iteration, whereas XP is eager to make any necessary changes because early modifications positively affect the quality of products. Furthermore, while Scrum focuses on working

product increments, Lean is concerned with improving process output and lowering costs [17].

C. *Main Properties of Traditional Methods and Agile Methods*

There are important properties to be considered before moving from the current traditional method to agile. Agile is an incremental development process, while traditional development methodologies are more formal development approaches. Properties are the primary factors that should be understood and considered as the basis for transformation. However, Table I describes the top ten properties that allow organizations to stay using the current traditional methods or may transform to agile methods [16].

TABLE I. PROPERTIES OF TRADITIONAL METHODS VS. AGILE METHODS

Properties	**Traditional Methods**	**Agile Methods**
Attitude	Predictive	Adaptive
Project size	Large	Small
Team size/mindset	Large/disciplined	Small/innovative
Project management model	Autocratic	Decentralized
Change attitude changes	Resistant against	Embracing changes
Documentation	Comprehensive	Light and abstract
Upfront planning	Comprehensive	Limited
Life cycle	Tied and bound	Unlimited iteration
Organizational culture	Command and control	Leadership and collaboration
Return of Investment	At the end of the project	Early stages

D. *Agile Requirements and Challenges for Teams and Methods*

During the transformation process from the traditional methods to agile, there are several challenges in the four main domains that organizations may face and should address before moving forward. These challenges are described as follows:

Organization and management:

- The style of management, adaptability, and responsiveness association will be changed from "order and control" to "administration and cooperation" [18].
- The project manager's role has changed from an organizer and controller to a chief and facilitator [19].
- Group decision-making and cooperative choice are important issues in improving assets and performing group assignments [20].
- Documentation in agile is limited, as most of the information is understood because one of the primary goals in agile is to write software rather than documentation [21].

People:

- Pair programming in XP for senior developers requires skilled team selection and provides fundamental preparation and training to do enough work rehearses that advance procedure greatness [22].
- Role of the customer(s), their assumptions, and the importance of their cooperative response, approval, and education Currently, conventional undertaking leaders cannot fit new circumstances [23].
- Awareness of team members requires sufficient instruction and guidance. It is important for supervisors to consider an expert mentor in their group.

Process:

- Predefined standards: Standard exercises and estimates are used in common techniques, while doubtful exercises are used as dexterous strategies to help speedy improvement and top-notch production [24].
- Implementing agile activities and performing some agile exercises, such as continuous coordination, generating forthright test codes, and continuous testing in traditional programming engineers is difficult.
- Selecting a suitable agile method. As each agile strategy differs in terms of needs, execution, venture and group size, cycle time, code ownership, and other factors. Regrettably, there is no intelligent approach that can be linked together; thus, organizations must choose the best agile strategy for them.

Tools:

- Agile methods have different measuring tools and measurements compared to traditional methods [25]. The assessment of software size in agile methodologies is a team-driven measurement, regardless of traditional approaches [26], and the size estimation process is performed for each iteration separately [27]. Using a standard method for size estimation in agile methods, such as Agile COSMIC, could be beneficial [28].
- Using rigid tools and hardware, businesses should use devices that allow for slow development, consistent coordination, rework, executive rendition, and other significant advancements [29]. They should consider cutting-edge technologies, such as fog computing, GPU hardware technology, and supercomputing [30].

E. Agile Transformation Frameworks

Several new existing frameworks focus on agile transformation. These frameworks are team-oriented, and they transform the entire organization to adopt agile values and principles. Agile practices are applied to an environment that can embrace—and thrive in—a flexible, collaborative, self-organizing, fast-changing environment.

Hamed et al. [33] investigated a real-world instance of an IT organization attempting to transition from a traditional project management office (PMO) to an agile PMO. They examined the current concerns and challenges and designed a transformation approach known as the scalable agile transforming process (SATP). The implemented procedure has been successfully finished; however, it must be applied to different project types in order to develop common criteria for the SATP process. The overall major challenge for any organization is structure transformation.

Durisic et al. [34] discussed the use of the scaled agile framework (SAFe) on various Volvo car systems. SAFe was created for large organizations to use in order to implement the agile principle across different departments. Agile principles were successfully deployed across numerous teams; yet, team coordination, communication, and dedication to working together were significant problems due to the presence of some centralized teams.

AlAli et al. [35] provided a customized roadmap architecture for a digital transformation strategic plan based on sprints. As most major enterprises transition to digital transformation, they require a well-thought-out strategic plan. Their method is divided into four stages: planning, prototyping, production and transition, and realization. Each phase has specific goals that must be met. This framework can be customized in terms of the interested domain, roadmap architecture, and roadmap procedure.

Signoretti et al. [36] investigated a real-world application of the Combined Approach in software development. The case study was conducted with a multinational information technology corporation, and data was gathered through interviews, focus groups, questionnaires, and observations. The study's findings emphasized two primary strategies: assigning a dedicated team to drive the transformation and providing the necessary toolbox, as well as incorporating the role of product designer into a cross-functional team and leveraging experimentation.

III. A Proposed Recommendation System for Agile Methods

This section presents an appropriate recommendation system to advise traditional teams about the suitable agile methodology based on the available data. This system allows all the current teams to check if they meet the agile requirements or need to train themselves to meet them.

A. Dataset

This work focuses on the survey report results conducted in our previous research [17]. We created a questionnaire list based on three agile methodologies: Scrum, XP, and Lean. The questionnaires were divided into three key modules: design and strategic learning requirements, profiling modules, and team building, each of which had multiple questions.

We collected and analyzed profile attributes against agile experience, position, team size, sector, and location in the profile-mapping module. According to the responses, the majority of participants were interested in following agile

methods, but they did not deal with it, and 27% of the participants were playing project manager roles. Also, most firms had team sizes ranging from 1 to 50 people, with the majority associated with software houses, IT, and government sectors.

This section focuses on the results of the second questionnaire, a design and learning module for the team. Our first point was concerning the team's regular value production for their stakeholders; the majority of responses were “yes” or "categorically yes." Most participants claimed they did not have a hybrid team collaboration environment, demonstrating clearly that a strong collaborative environment is the most important aspect of agile implementation. Also, some questions addressed were the organization's willingness to "dedicate a full-time business expert," "dedicate a full-time delivery team," "provide a business analyst for Just-in-Time (JIT)," "time box each iteration," "assign the right person at right role," support a collaborative environment," and "implement the mandatory discipline." It was noticed that around half of the participants were unwilling to devote full time to business expertise.

Similarly, our study conducted a third survey report on team factor questionnaires and examined five aspects: value, validation, stakeholders, self-organization, and reflective improvement. People seek ways to improve their business processes, and 77 percent believe in agility and adoption. Furthermore, 59 percent of respondents stated that their team had a product owner who represented the stakeholder community. They collaborate on an as-needed basis with certain stakeholders, notably domain specialists. Contrastingly, almost 60% of the respondents said that task assignments should be handed directly to the team by the project manager rather than through self-effort. In terms of daily meeting organizations, half of the people do it at the start of the day and discuss the agenda based on "what is done" and "what to do," as well as responsibilities assigned to the team correspondingly.

In contrast, it was noticed that just 9% of participants use and construct collaborative environment infrastructure. In the final parameter of the team building module, "reflective improvement," half of the participants measured and tracked their progress daily at the end of the day.

B. A Proposed Framework

The proposed framework is presented in detail, which helps recommend the existing traditional method to select the appropriate agile methods. The proposed framework is divided into three key phases: the first is pre-processing, in which the system examines the organizational initial input data to determine whether the system is eligible to use the agile methodology. Once the data has been processed, it is transmitted to the next phase, where predefined concerns are compared with user input data to determine which agile model is most suited for the intended organization. The entire dataset is sent to the next module, an auto analyzer, which checks the terms and conditions of agile methods such as Lean, Scrum, and Extreme Programming to ensure an appropriate agile recommended model. After comparing the data to the outcomes of various agile models, the proposed framework produces a final output in the form of an agile-recommended model. Fig. 3 depicts a thorough data flow diagram that illustrates the processes involved from the initial demand to the outcome, while the block diagram depicts the system's blocks connected by lines that demonstrate the relationships between these blocks (Fig. 4). Furthermore, all the phases are discussed in detail to understand how the entire system executes its objectives and how each component reacts to the system through mutual interaction. More information is available in the following:

- Pre-processing: This phase is the first phase of the proposed recommendation system framework for agile methods, in which the system initializes the project to gather the system's configuration and settings. Next, the system gathers the organization's fundamental team information, such as the team members’ domain, skills, experience, number of team members, and other essential information for all members. This collected information is then linked to the main agile requirements for adopting the model. If the submitted input meets the agile requirements, it is forwarded to the next phase, "Concerns Analysis." Else, it is recommended that hands-on training is provided by the organization itself to address weak areas in order to meet the agile requirements. The tool will not recommend any specific or required training; it will only

show the user that the provided input does not meet agile requirements.

- Agile Model Recommendation: The third phase of the proposed framework is the agile model recommendation, in which the input

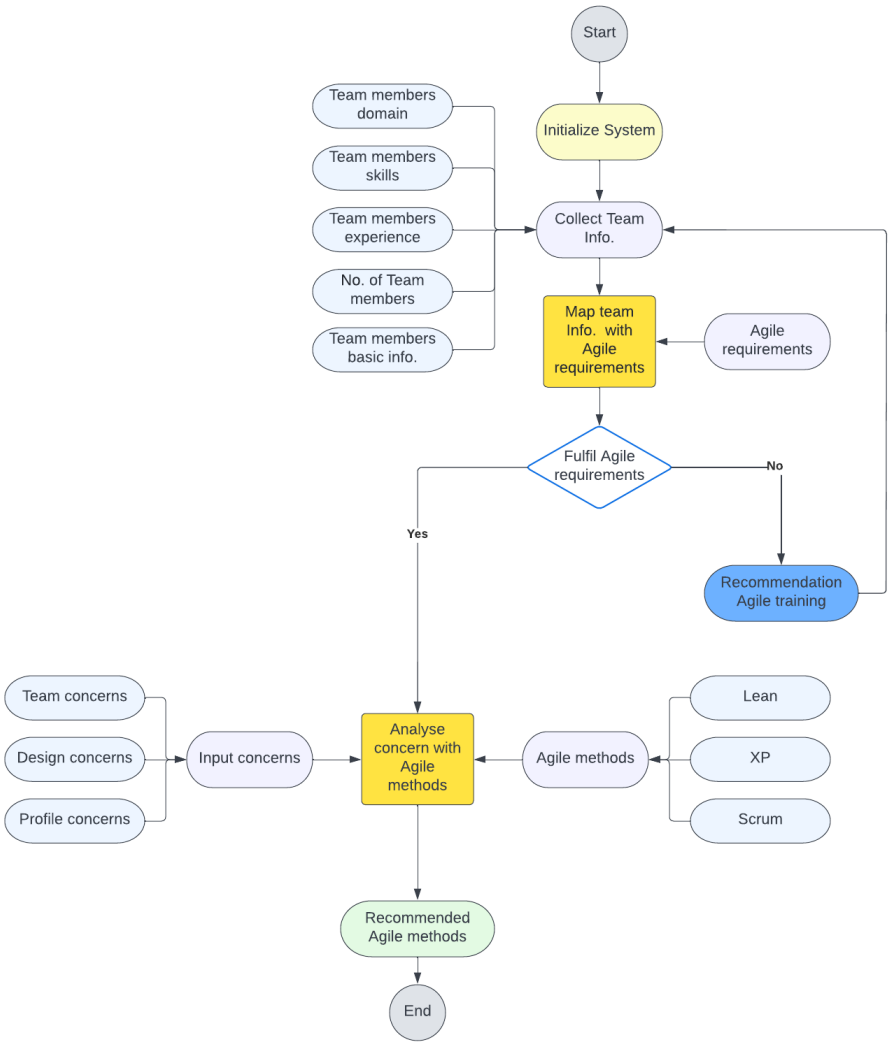

Fig. 3. Flow Diagram for the Proposed Recommendation System Framework for Agile Methods.

- Concerns Analysis: In this phase, the proposed framework includes team-related, design-related, and profile-related concerns for any organization. Fig. 3 shows a list of the specific concerns. These issues or concerns provide precise details about any organization to match a suitable agile model. Contrastingly, the proposed framework considers the aforementioned concerns as input raw data, which is then processed to provide refined information and then transmitted to the following phase, "Agile Model Recommendation," to reach a final agile methodology recommendation.

data from the previous two phases are analyzed in depth to produce a final recommendation model. This phase entails recommending one of the agile models, such as Lean, XP, or Scrum. All input data were compared to the terms and circumstances of these models. In this case, our smart tool chooses one of the most appropriate models and recommends it to the chosen organization based on the weight of the questions about the concerns; each question has weight, and a method will be recommended to the user based on the total weight. Thorough information and description of these agile models have

already been contributed to the system and preserved for interactive use with the Firebase database. The proposed framework allows additional flexibility in using the stored data with any other application accessible from interactive database data.

A. *Implementation of Proposed Recommendation System Framework*

The proposed framework was implemented on a mobile platform, and an Android application was created and developed using the Android Studio tool 3.7 version and the JAVA programming language (Fig. 5). We used the rapid processing-based Firebase database, which provides real-time interaction with the apps, to create a more user-friendly interface. The proposed model was tested on several Android devices, including the NEXUS 6, Samsung Galaxy 8 S, and Huawei P10. The proposed model outperformed on all fronts, implying that the agile model is a recommendation.

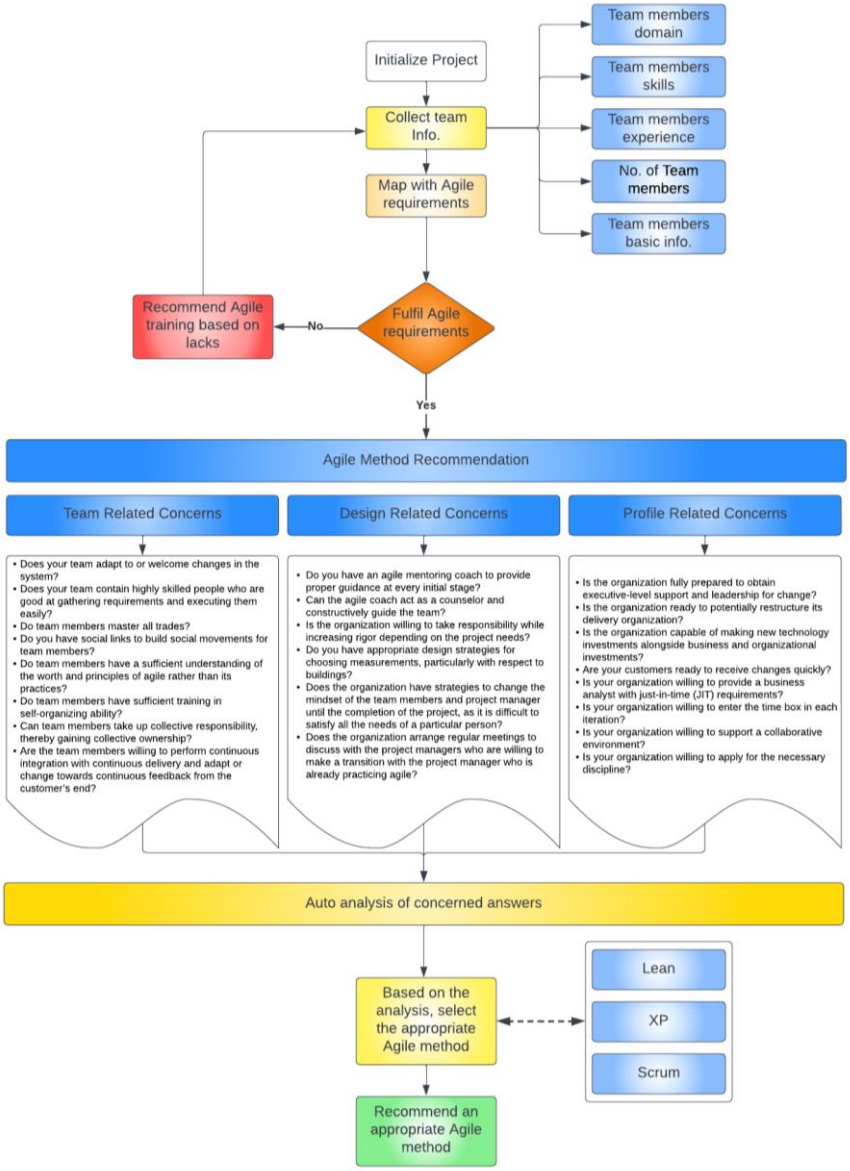

Fig. 4. Block Diagram for the Proposed Recommendation System Framework for Agile Methods.

IV. Implementation and Validation

The proposed recommendation system requires a tool implementation that can check whether traditional teams are eligible to move to agile and assist them in deciding which agile framework to apply. The mobile application and its validation by comparing it with several other projects are described as follows.

B. *Validation with Comparing Discussion*

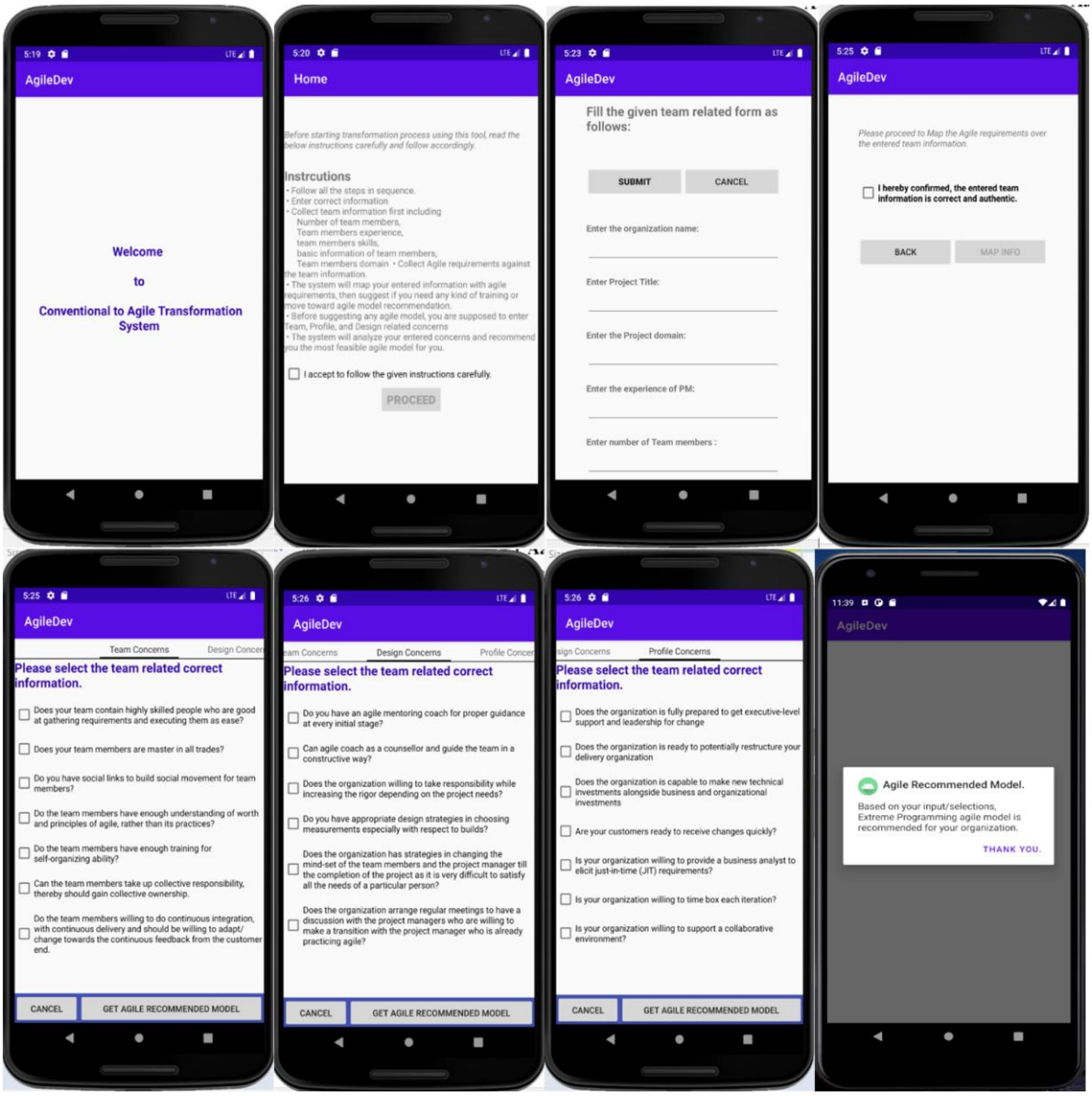

Fig. 5. Snapshots of our mobile application.

Selected IT-related projects were chosen and assessed using our mobile application to validate the framework. These chosen projects are now being used in international standards projects being considered for implementation in the proposed framework. The identities of the organizations were not revealed for privacy reasons, but the details are as follows. Technically, these projects are referred to as the dataset used to test the framework. However, during implementation, different datasets were chosen and run via the proposed solution mobile application, yielding the findings listed in Table II. This process is described in the following:

- T-Shirt Designing: It is an Internet business developed by a software development company. Online printed T-shirt designing allows new users to register and log in. This product has an admin login that checks for new requests and orders, with a member login that allows users to alter configurations using a wide range of colors and designs and submit mass requests for the same. Initially, the product structure allowed customers to select a shirt plan. Subsequently, the client selected the desired tone. Clients can choose from a variety of designs and texts to be put on the shirt using the Internet. After the client completes the configuration, the cost

is calculated, and the client can pay for it online. This pre-ordered T-shirt gets delivered to their front door.

By comparing our proposed framework, which uses factual data provided by the T-Shirt designing team, we found that the project (according to them) belongs to software development for mobile apps, whereas the development team was skilled in different web application development platforms. However, because the minimum requirements of agile transformation were not met, our tool does not recommend Agile transformation.

- City Tour guide: This Android-based city tour guide project provides visitors with a city map based on the location entered by an Android telephone client. This information assists tourists in identifying the best places to visit. Furthermore, it contains all the details of those areas or how to get to the area and other crisis conveniences, such as medical clinics, foundations, and transportation stops. Yet it provides fundamental data required to choose the spots to visit. This task is mostly useful for travelers who have no idea where they want to go.

This application is based on a Geographic Information System (GIS) and provides numerous advantages, such as allowing the client to see the necessary area on a map and estimate the time required to reach the final destination. The framework provides essential details that will be required, e.g., a picture of that location, with fundamental details such as location and contact. Based on the analysis and scrutiny of our tool, it was recommended that the city tour guide project team use the Extreme Programming (XP) agile method.

TABLE II. INPUT DATA AGAINST THE SELECTED PROJECT

Project Title	Project Domain	Number of Team members	Experience years of team members	Team Member's Domain	Team member's skills	Fulfill Agile requirement	Agile Model Recommendation
T-Shirt Designing	Software Dev	5	3 to 6	Software development	.NET, php, Adobe Flex	No	Not Recommended
City-Tour guide	Mobile app Dev	50	5 to 15	Software Development, Testing, & Project Management	P. management, S. Develop, S. Testing	Yes	XP
SR Train app	Mobile app Dev	18	5 to 20	Software development, & Mobile development SW testing	Android, IOS, Windows app	Yes	Scrum
ERP App Migration	Software Dev	12	5 to 15	Software development	SAP dev, Linux dev	Yes	XP
ERP DBs Migration	Software Dev	9	5 to 15	Software development	SAP dev, S/4 HANA dev	Yes	Scrum

- SR Train app: The SR Train application is a mobile application development project. The train app includes booking, ticket management, seat selection, and online payments. It should come with a user manual and a guide. The app includes a registration screen, an admin screen, and multiple language options. This application should be a single application built on the Hybrid development model for all platforms, including Android, iOS, and Windows, and should support mobile phones and tablets. App compatibility with all available screen resolutions, portraits, and landscape orientations. PUSH notifications are used to alert users to mobile applications. The necessary information was gathered for our agile transformation tool, and recommended Scrum method.
- ERP Program (Application & Database migration): Enterprise resource planning (ERP) infrastructure components stored in several data centers have reached the end of their useful lives and require a technological refresh. The goal of this infrastructure renovation project is to replace and reduce obsolete hardware, legacy processors, and proprietary operating systems that host the ERP and associated systems, as well as to

add capabilities to ensure the continued reliability of equipment with improved speed and capacity using modern tools and technologies. This project was designed to eliminate unforeseen service outages, shrink the IT footprint, streamline operations, improve security, and facilitate expansion. It includes installing, configuring, and maintaining hardware, hypervisors, operating systems, SAP software, and migrating databases for all applications on physical and virtual servers.

The proposed framework applies to two projects related to this program: application and database migration. The first project is for application migration from existing physical and virtual servers to new servers, while the second project is for database migration. The migration for both projects went into multiple phases, with each phase having several applications and databases. Based on the given input, our tool recommends XP for the application migration team and Scrum for the DB migration team.

V. Challenges and Limitations

This paper concerns recommendations for transforming conventional systems into agile methodologies with particular aspects. However, it was necessary to improve the proposed recommendation system to make it more comprehensive, effective, and accurate. For example, other agile methodologies are neither analyzed nor included. In addition, some potentially inaccurate responses to the recommendation questions can affect the accuracy of the recommendations; thus, it is critical to enter sufficient details for each team member. Also, the system does not currently validate team members' skills and experiences, which can negatively affect the resultant recommendations.

However, the proposed framework recommends a single agile model that is the most appropriate for the team, but a team may need to know what percentage of requirements were fulfilled for all agile models to provide additional transformation choices. Therefore, the current recommendation system does not recommend any required training for teams, and this would be a distinct feature if it must be provided by the system.

VI. Conclusions and Future Work

Traditional methodologies have some limitations, such as accepting the changes, bounding for defined cycles, and releasing the product at the end of the development cycle. In contrast, agile methods face many challenges and limitations by interacting with the customer, releasing versions in each iteration, responding to changes, and powerfully communicating and collaborating. Therefore, we present a recommendation system to allow eligible teams or organizations to recommend transforming from the current traditional methods to agile methods based on the results of the analysis of our survey report. In addition, we implement a mobile application to validate the successful operation of the proposed framework. This leads in the highest benefits from an agile methodology based on the business environment and customer requirements in terms of cost, time, and other success criteria for the software industry.

Although the proposed framework can recommend any organizational or team transition from traditional to agile methods based on the input details, recommendations are limited to IT-related organizations, and the proposed framework was static for the three agile methods used in this study. However, to overcome this issue in the future, we want to work on a fully fledged adaptive and dynamic framework [35] that deals with more agile methodologies and has a database for team members' profiles to evaluate their skills and experiences using defined parameters and to recommend the required training or skill development programs to enroll them into agile methodologies. This will allow large enterprises to have a dynamic tool for choosing the best team for a specific project to fully utilize the team's capabilities, and to address the market's increasing need for agile adoption, which might assist interested firms in enhancing their business processes, as well as how well their teams cooperate internally [36]. In addition, our framework can be improved to recommend multiple suitable agile models for the traditional software team or individual IT project arranged according to the percentages of fulfillment of agile requirements.

REFERENCES

[1] S. Balaji, and M. S. Murugaiyan, "Waterfall vs. V-Model vs. Agile: A comparative study on SDLC," International Journal of Information Technology and Business Management, vol. 2, no.1, pp. 26-30, 2012.

[2] A. Lawal, and R. C. Ogbu, "A Comparative Analysis of Agile and Waterfall Software Development Methodologies," BAKOLORI JOURNAL OF GENERAL STUDIES, vol. 11, no.2, pp. 1-2, 2021.

[3] Y. Bassil, "A simulation model for the waterfall software development life cycle," International Journal of Engineering and Technology (iJET), vol. 2, no.5, pp. 1-7, 2012.

[4] D. M. Mahmud, and N. A. S. Abdullah, "Reviews on agile methods in mobile application development process," in 2015 9th Malaysian Software Engineering Conference (MySEC), pp. 161-165, 2015. https://doi.org/10.1109/mysec.2015.7475214

[5] Feggins Jr, R. (2011, March 2). Comparing Waterfall and Rational Unified Process. Reedy Feggins Jr. https://theAgileblueprint.wordpress.com/2011/03/02/comparing-waterfall-and-rational-unified-process/

[6] C. Tam, E. J. da Costa Moura, T. Oliveira, and J. Varajão, "The factors influencing the success of on-going agile software development projects," International Journal of Project Management, vol. 38, no.3, pp. 165-176, 2020. https://doi.org/10.1016/j.ijproman.2020.02.001

[7] A. M. Dima, and M. A. Maassen, "From Waterfall to Agile software: Development models in the IT sector, 2006 to 2018. Impacts on company management," Journal of International Studies, vol. 11, no.2, pp. 315-326, 2018. https://doi.org/10.14254/2071-8330.2018/11-2/21

[8] I. Sommerville, (2016). "Software Engineering Global Edition", 10th ed., Marcia Horton, Ed. England: Pearson Education Limited, 149.

[9] T. Stober, and U. Hansmann, "Agile Software Development: Best Practices for Large Software Development Projects," Springer, 2010. https://doi.org/10.1007/978-3-540-70832-2

[10] K. Schwaber, and M. Beedle, "Agile software development with scrum. Series in agile software development", Vol. 1. Upper Saddle River: Prentice Hall, 2002.

[11] J. Newkirk, "Introduction to agile processes and extreme programming," in Proc. ICSE, ICSE 2002, IEEE, pp. 695-696. 2002. https://doi.org/10.1145/581441.581450

[12] E. Mnkandla, and B. Dwolatzky, "A survey of agile methodologies," Transactions of the South African Institute of Electrical Engineers, vol. 95, no.4, pp. 236-247, 2004.

[13] T. Ohno, and N. Bodek, "Toyota production system: beyond large-scale production," 1st ed., Productivity press, 2019. https://doi.org/10.4324/9780429273018

[14] C. Martin, and D. R. Towill, "Supply chain migration from lean and functional to agile and customized," Supply Chain Management: an international journal, vol. 5, no.4, pp. 206-213, 2000. https://doi.org/10.1108/13598540010347334

[15] R. N. Charette, "Challenging the fundamental notions of software development," Cutter Consortium, Executive Rep, 4, 2003.

[16] P. Salza, P. Musmarra, and F. Ferrucci, "Agile methodologies in education: A review," in Agile and lean concepts for teaching and learning, pp. 25-45, 2019. https://doi.org/10.1007/978-981-13-2751-3_2

[17] A. Alsari, R. Qureshi, and A. Algarni, "Agile framework to transform traditional team," in 2020 IEEE Frontiers in Education Conference (FIE), IEEE, pp. 1-9, 2020. https://doi.org/10.1109/fie44824.2020.9274240

[18] H. Yang, S. Huff, and D. Strode, "Leadership in software development: Comparing perceptions of agile and traditional project managers," in AMCIS 2009 Proceedings. 184, 2009.

[19] N. B. Moe, T. Dingsyr, and O. Kvangardsnes, "Understanding shared leadership in agile development: A case study," in 2009 42nd Hawaii international conference on system sciences, IEEE, pp. 1-10. 2009. https://doi.org/10.1109/hicss.2009.480

[20] N. B. Moe, A. Aurum, and T. Dybå, "Challenges of shared decision-making: A multiple case study of agile software development," Information and Software Technology, vol. 54, no.8, pp. 853-865, 2012. https://doi.org/10.1016/j.infsof.2011.11.006

[21] M. Levy, and O. Hazzan, "Knowledge management in practice: The case of agile software development," in 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering, IEEE, pp. 60-65, 2009. https://doi.org/10.1109/chase.2009.5071412

[22] J. Srinivasan, K. Lundqvist, and C. Norström, "Studying software organizations: in search of a method," in Proceedings of the 3rd India software engineering conference, pp. 51-60, 2010. https://doi.org/10.1145/1730874.1730885

[23] K. Conboy, S. Coyle, X. Wang, and M. Pikkarainen, "People over process: key people challenges in agile development", IEEE Software, vol. 21, no.6, pp. 70-75, 2011. https://doi.org/10.1109/ms.2010.132

[24] N. P. Singh, and R. Soni, "Agile software: Ensuring quality assurance and processes," in International Conference on High Performance Architecture and Grid Computing, pp. 640-648, Springer, Berlin, Heidelberg, 2011. https://doi.org/10.1007/978-3-642-22577-2_86

[25] B. Tanveer, L. Guzmán, and U. M. Engel, "Effort estimation in agile software development: Case study and improvement framework," Journal of Software: Evolution and Process, vol. 29, no. 11, e1862, 2017. https://doi.org/10.1002/smr.1862

[26] E. D. Canedo, D. P. Aranha, M. D. O. Cardoso, R. P. Da Costa, and L. L. Leite, "Methods for estimating agile software projects: Systematic literature review," in Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE, pp. 34-39, 2018. https://doi.org/10.18293/seke2018-031

[27] T. Javdani, H. Zulzalil, A. A. A. Ghani, A. B. M. Sultan, and R. M. Parizi, "On the current measurement practices in agile software development," arXiv preprint arXiv:1301.5964, 2013

[28] M. Salmanoglu, T. Hacaloglu, and O. Demirors, "Effort estimation for agile software development: Comparative case studies using COSMIC functional size measurement and story points," in Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement, pp. 41-49, 2017. https://doi.org/10.1145/3143434.3143450

[29] P. S. Brockmann, and T. Thaumuller, "Cultural aspects of global requirements engineering: An empirical chinese-german case study," in 2009 Fourth IEEE International Conference on Global Software Engineering, IEEE, pp. 353-357, 2009. https://doi.org/10.1109/icgse.2009.55

[30] M. U. Ashraf, F. A. Eassa, A. A. Albeshri, and A. Algarni, "Toward exascale computing systems: An energy efficient massive parallel computational model," International Journal of Advanced Computer Science and Applications, vol. 9, no. 2, 2018. https://doi.org/10.1109/access.2018.2823299

[31] R. M. Haj Hamad and M. Al Fayoumi, "Scalable Agile Transformation Process (SATP) to Convert Waterfall Project Management Office into Agile Project Management Office," in 2018 International Arab Conference on Information Technology (ACIT), Werdanye, Lebanon, Nov. 2018, pp. 1–8. doi: 10.1109/ACIT.2018.8672701.

[32] D. Durisic and A. Berenyi, "Agile System Architecture in Large Organizations: An Experience Report from Volvo Cars," in 2019 IEEE International Conference on Software Architecture Companion (ICSA-C), Hamburg, Germany, Mar. 2019, pp. 33–36. doi: 10.1109/ICSA-C.2019.00014.

[33] A. G. Al-Ali and R. Phaal, "Design Sprints for Roadmapping an Agile Digital Transformation," p. 6, 2019.

[34] P. Gregory and K. Taylor, "Defining Agile Culture: A Collaborative and Practitioner-Led Approach," in 2019 IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), Montreal, QC, Canada, May 2019, pp. 37–38. doi: 10.1109/CHASE.2019.00016.

[35] M. Qureshi, and M. Kashif, "Adaptive Framework to Manage Multiple Teams Using Agile Methodologies," International Journal of Modern Education & Computer Science, vol. 9, no. 1, 2017. https://doi.org/10.5815/ijmecs.2017.01.06.

[36] P. Paterek, "Agile transformation framework in software project organization," in ICMLG 2018: 6th International Conference on Management, Leadership and Governance, pp. 258-267, 2018.

نظام توصية رشيق لفرق تطوير البرمجيات التقليدية

عبدالرحمن السري [١] ، عبدالله القرني [١] ، رضوان قريشي [٢] ، عبدالله مارش[1]

[١] قسم علوم الحاسبات، كلية الحاسبات وتقنية المعلومات، جامعة الملك عبد العزيز، جدة ، المملكة العربية السعودية

[2] قسم تقنية المعلومات ، كلية علوم الحاسب والمعلومات ، جامعة الملك عبد العزيز، جدة ، المملكة العربية السعودية

aalsari0003@stu.kau.edu.sa, amsalgarni@kau.edu.sa, rmuhammd@kau.edu.sa, ammali@kau.edu.sa

المستخلص. مع تزايد الطلب والضرورة لتطبيقات الويب والجوال، يقوم المبرمجون بتطوير عدد متزايد من التطبيقات كل يوم، وتتمتع هذه التطبيقات بدرجة عالية من الإنجازات. لذلك، يجب على المطورين اتباع عمليات تطوير البرامج المميزة كممارسة قياسية. أنظمة إدارة المشاريع التقليدية بها أوجه قصور يجب معالجتها. بينما تُستخدم منهجيات تطوير البرمجيات الرشيقة (Agile) على نطاق واسع في صناعة البرمجيات، كونها أكثر كفاءة بشكل ملحوظ وتحسن جودة المنتج إذا تم اتباعها وفقًا لمتطلبات أجايل. الأساليب التقليدية غير قابلة للتكيف مع التغييرات ولها دورات حياة محددة. ولذلك، فإن الأساليب الرشيقة أكثر مرونة في الاستجابة للتغييرات والتواصل مع العملاء وتحسين التواصل والتعاون. في هذه الدراسة، نقدم نظامًا للتوصية يستخدم مخططًا للتدفق لتحديد منهجية (Agile) المناسبة التي سيتم تطبيقها من قبل الفرق بناءً على البيانات التي تم جمعها بعد التأكد من أن الفريق مؤهل للتغيير إلى (Agile). بالإضافة إلى ذلك، قدمنا تطبيقًا للهاتف المحمول تم تنفيذه لدعم شجرة القرار ووصف المشروع الذي تم استخدام الأداة فيه. هذا يؤدي إلى أكبر الفوائد من هذه المنهجية الرشيقة التي تعتمد على بيئة الأعمال ومتطلبات العملاء لتحسين صناعة البرمجيات من حيث التكلفة والوقت وعوامل النجاح الأخرى.

الكلمات المفتاحيةـ الطرق التقليدية؛ إطار نظام التوصية؛ طرق رشيقة؛ فرق تطوير البرمجيات؛ ادارة المشاريع