Implementing Knowledge Management in an IT Startup: A Case Study

Dena Ahmed, Said A. Salloum, and Khaled Shaalan

Abstract Knowledge management (KM) comprises of managing, planning, deploying, collecting, storing, reusing, and distributing the knowledge in organizations in an organized planned way. Applying KM in an organization aims to allow for ensuring overall success as it is a method that simplifies the entire process of dealing with knowledge. Some organizations such as software development companies suffer from many issues that obstruct the KM process. This case study performed on an Egyptian IT startup aims to examine their business process and methodologies for dealing with knowledge. The results showed that they were wasn't utilizing the knowledge they had properly and suffered from a lack in KM in almost all aspects. All problems were examined and solutions to handle them were proposed to implement efficient KM.

Keywords Knowledge management (KM) · Knowledge management system (KMS) · Software engineering (SE) · Software development · Startup

1 Introduction

Software development is quite elaborate involving several people working in multiple stages and different activities. New technologies are constantly emerging, and different projects require different development techniques. Therefore, many new issues are faced and new experience is gained with each project and thus, new knowledge is created. That is why knowledge in software development is very

D. Ahmed · K. Shaalan
Faculty of Engineering and IT, The British University in Dubai, Dubai, UAE

S. A. Salloum (✉)
School of Science, Engineering, and Environment, University of Salford, Salford, UK
e-mail: ssalloum@sharjah.ac.ae

S. A. Salloum
Machine Learning and NLP Research Group, Department of Computer Science, University of Sharjah, Sharjah, UAE

M. Al-Emran et al. (eds.), *Proceedings of International Conference on Emerging Technologies and Intelligent Systems*, Lecture Notes in Networks and Systems 322,
https://doi.org/10.1007/978-3-030-85990-9_59

heterogeneous and diversified with an exponential growth rate [1, 2]. Companies, especially startups, face many difficulties while trying to keep track of what the knowledge they have is, where it is located, who has it, and how they can apply it. This further proves the need for some structured methodology to manage the knowledge at hand, helping the organization to advance forward. KM has a lot to offer to software companies as it takes an all-encompassing, exhaustive approach to the different activities of collecting and handling knowledge [3]. It considers the knowledge obtained from the software developers' minds as human assets and converts that into a company asset that can be used and shared by all the company's employees [2]. Having a KM system in a software company creates a shared language between the software developers, which facilitates their interactions with each other, thus allowing for more sharing of their experiences and knowledge. KM simplifies and advances the concept of organizational learning in the context of software development by helping with knowledge transfer throughout the organization [4], thus allowing for more success and improved performance. Many companies, especially startups, are unaware of how KM can immensely impact their quality of work and productivity [5], thus losing the chance of advancing. Several startups go out of business because of the lack of proper KM practices [6]. This case study aims to take a closer look at an Egyptian IT startup to identify the aspects that could benefit from applying KM, and show how important KM is to the success of software organizations.

The following points will be tackled by this study:

(1) What are the various types of knowledge processes used in the company?
(2) How important is applying KM in a software startup?
(3) What are the problems regarding KM in the company?
(4) What are some solutions and plans for applying proper KM?

This paper is divided as follows. Section 2 discusses KM in software organizations. Section 3 covers the methodology of data collection for the case study. Key issues, KM processes identified, and proposed potential solutions are presented in Sect. 4. Finally, Sect. 5 will conclude.

2 Knowledge Management in Software Organizations

Knowledge in SE is very dynamic and continuously evolving with new technologies developed every day, and so does the organization's culture. The processes in SE [7] are almost wholly dependent on the knowledge of the developers and team members. The fundamental principle in SE [8] is that the availability and efficient utilization of knowledge significantly enhances the quality of the developed software products. Multiple factors assist the KM process in software companies such as technology, leadership and organizational culture [9]. Technology connects developers together and creates a knowledge bank for the entire company; while

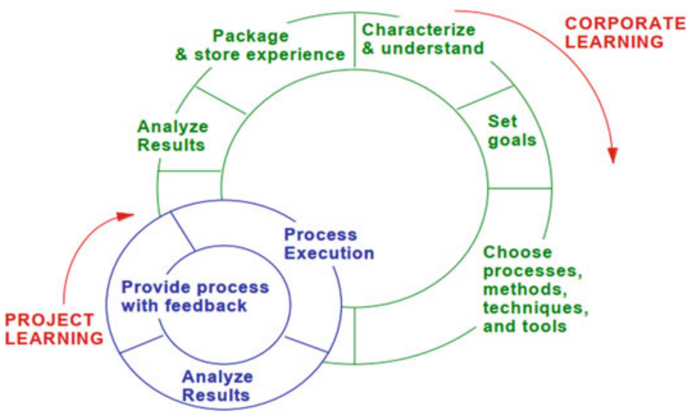

Fig. 1 The quality improvement paradigm

leadership is the force that pushes for applying KM in the work processes and services of the company. The organizational culture advocates knowledge sharing and distribution of expertise to advance innovation within the company [10]. There is a continuous learning process occurring during the lifecycle of software development, therefore all knowledge from projects must be appropriately gathered and documented in an organized manner to help with upcoming projects. This concept is also portrayed in the Quality Improvement Paradigm (QIP) [11, 9] where analysis and finding conclusions regarding the project's progress is done at each project level. Then, the results are further analyzed up another level and stored in what can be referred to as an experience database. This serves as knowledge to guide plans for upcoming projects such as selecting the business processes, methodologies and tools to be used [12]. Figure 1 shows the QIP flow.

3 Methodology

The startup under study will be referred to with a pseudo name "Tech-K" throughout the paper for confidentially purposes. Data about Tech-K was collected through interviews with all employees. Questions for the interviews were designed carefully to address the research variables to identify the degree of KM implementation in an IT startup.

Tech-K is an IT startup that started with 5 employees, and now currently 20 employees: 2 Android developers, 2 iOS developers, 2 front-end Web developers, 1 backend developer, 4 employees handling social media and marketing tasks, 2 account managers, 2 UI designers, 3 interns, 1 technical officer and the CEO. In the beginning, the focus was only on developing websites. This later expanded into developing basic mobile applications using drag & drop app builders, then native app development using Android then later iOS. In addition, social media marketing was also added to the company's activities. The CEO has a medical and business background, so his technical knowledge regarding software development is limited. Almost all employees are young, fresh graduates with limited experience.

4 Results and Discussion

4.1 Identified KM Processes and Issues

Acquiring Knowledge. Because of the small number of employees, meetings are usually informal where many different subjects are discussed. The outcome of these meetings is not being utilized at the organizational level so there is no proper attainment of knowledge to keep track and predict further developments. Experiences shared during meetings where developers discuss their progress, issues faced and plan for remaining tasks are the main source for acquiring knowledge but they are not recorded to benefit from. Requirements' gathering from clients is another source of gathering knowledge but it is also done informally through meetings and interviews without proper documentation or an organized procedure. When more developers join the company, the CEO and technical officer are going to be the only source of information. This knowledge centralization forces the technical officer to have multiple sessions and meetings to transfer their knowledge instead of actually focusing on more important pressing tasks. So, the CEO and the technical officer are subjected to loss of productivity because of the centralization of knowledge.

Codifying Knowledge. The majority of the employees don't know where to find knowledge about activities or projects done before or even the current knowledge regarding ongoing projects. Knowledge is not well recorded in most cases. A developer wanting to know about the logic of a specific feature in a mobile application has no documentation to go back to; but has to ask the technical officer or the UI designer. They don't have the answers to all the questions and they have to go back to ask the client sometimes. Occasionally, assumptions are just made without knowing if that assumption will match the client's needs. Employees think that there has to be a specific person responsible for documenting project requirements in a proper way, where they are continuously updated with any changes or modifications for an enhanced development environment.

Disseminating Knowledge. Because most of the knowledge is distributed through informal conversations, it is harder to supervise or control. Employees are of the opinion that informal conversations are used because there is not a better alternative. When knowledge is being recorded in an accessible system in a user-friendly way, clearly demonstrating who is working on what and which individual has the knowledge regarding a specific point, then the dependency on the informal meetings becomes less essential.

Almost all employees agree that a big problem at Tech-K is that the knowledge is stored only in the minds of the employees without being properly recorded. There is no doubt that this implicit knowledge is a great asset, but it would be of much greater benefit if the company was somehow less dependent on the actual presence of those employees. This is because this knowledge is lost when an employee leaves Tech-K which causes significant problems to the workflow, especially with being a startup with limited resources where each employee is handling important tasks that can't be delayed. There isn't a proper KMS to solve this. Hiring experienced senior developers in a startup is challenging as they need competitive salaries while there are limited financial resources. Senior developers also have more experience so they have worked with different technologies and bigger companies with better organization. So, even if offered a high salary, they usually leave after a few months because of the lack of a proper organizational structure. When a senior developer is working on a project, and then they leave the company before finishing it, all of his knowledge is not recorded. Even with a handover process, the complete knowledge inside the senior developer's head would be lost. Also, with no proper documentation of the code architecture, approach or techniques used in the code, the developer handling the rest of the project will face a lot of difficulties. This cripples the startup as the remaining junior developer would struggle with continuing the project on his own, and hiring another senior developer would definitely take a lot of time that the company can't afford. Some employees think that sharing knowledge with others will strengthen their standing in the company since you are likely to get back knowledge from others if you share your knowledge with them. Others at Tech-K finds sharing knowledge with colleagues to be very threatening. This is the case usually between developers or employees of the same level.

Developing & Applying Knowledge. It was found that at several times, the same knowledge is being developed at more than one place at Tech-K. This happens because of the absence of detailed information about colleagues' activities. For example, both front-end web developers would be working on a specific task at the same time because of some miscommunication or the lack of proper documented task allocation. This costs time and money, which is crucial for a startup. Developers also complain that they have to wait and keep asking for the designs from the UI developers and that slows down their progress. With the limited number of only 2 UI designers, they are under a lot of pressure and face difficulties in delivering their tasks on time. With the lack of resources and a prioritized task allocation system, there are a lot of informal requests. The same issue happens with the back-end developer as he is handling all web and mobile projects. Another complaint by the mobile developers was that the UI designers were too ambitious with their designs. The UI team always tries to create overly advanced designs with complicated patterns in an attempt to showcase their skills and impress both the client and the CEO. However, these designs are most of the time complicated for junior developers to translate into code and they take up a lot of time that could have been better utilized on coding main feature. Another issue is the constant change in requirements. Since most projects are obtained through personal connections of the CEO, there are not strict guidelines regarding the requirements of the project. For example, if the client wants a mobile application, the contract would only include the general broad functionality of the app without details. Also, in the spirits of being more flexible and satisfying clients to the best possible way, the CEO doesn't enforce any regulations regarding requirements change to maximize customer satisfaction. However, that caused many significant issues through the development process. Giving the client the freedom to constantly change the requirements whenever they want opened the door to an endless loop of modification requests that didn't make sense most of the time. Most of the time, clients don't have enough technical knowledge so they are not good judges of what is applicable and its impact on the overall project. There are several occasions where the client is confused and doesn't know what they want, so they come up with non-feasible unnecessary requirements. This affects the development process negatively in many ways such as delaying the deadlines affecting other future projects, having to destroy or completely change code architecture based on earlier requirements, changes across several departments and many more. Moreover, sometimes the CEO approves modifications without understanding the technical effort that would be needed by the developers to accommodate those changes. Some changes can completely change the entire structure and architecture serving as the building layer of the entire project which would require extensive effort and time.

In addition, there are no dedicated testers to test the projects before submitting them to the customer. The developers & other employees play the tester role by trying different scenarios on the product to identify bugs. Some mobile apps require testing on devices with different specifications, operating systems and screen sizes; and that is hard because of the limited resources available at Teck-K. Thus, employees test the apps on their personal mobile phones. Having no dedicated testers result in poor quality products and unsatisfied clients because of bugs or crashes.

Knowledge Systems. Teck-K uses Slack (*Slack* n.d.) for communication between employees. Employees don't feel like it is sufficient for communication and prefer a more comprehensive platform with more focus on project management. Developers also use Git (*Git* n.d.) as a version control system to monitor modifications in a project among several developers. It also allows developers to push their code changes to a centralized remote repository on the server. Keeping track of the history of changes made to all files in the code allows going back to a previous working version in case of bugs, and helps in identifying who is the responsible party if needed. However, developers don't utilize Git correctly. Most of them don't regularly push their new changes to the remote repository, and when they do, they don't describe the changes done. This also causes conflicts between developers in case a developer made some changes on his machine locally but didn't push to the remote copy, and another developer continued working on the old version without those updates.

Issues Summary. Issues faced by Tech-K can me summarized to: Not learning lessons from previous projects, flawed team selection and task allocation, centralization of knowledge, lack of important knowledge among key employees, lack of proper knowledge sharing and transfer, reluctancy of some employees to share knowledge, staff turnover causing loss of knowledge/failure of projects, poor requirements gathering from customers, lack of a proper change control system for modifications in requirements, project management problems including poor management skills, lack of proper KMS, misuse & underutilization of systems used, scheduling and deadline estimation problems, lack of proper testing of software products, and lack of a proper knowledge map. All of these issues cause significant damage to the startup including unsatisfied clients, frustrated exhausted employees, poor quality products, delayed delivery, poor reputation and loss of money. If these problems are not properly addressed, Tech-K won't be able to grow and would eventually fail.

4.2 Proposed Potential Solutions

Create a Knowledge Sharing Culture and a proper KMS. A knowledge repository similar to the portal in [13] is recommended. This system should contain all the information, expertise, help, guides and support needed for all tasks' domains. This could include IDE custom settings, authentication keys, libraries used, frequently encountered issues with documented solutions. Template code files can also be included for commonly used tasks across several apps and complex code components can be documented to be reused instead of creating everything from scratch. This saves effort, money and time by reusing the previously gained knowledge for more efficient advancement. This will promote the knowledge sharing culture at Tech-K and encourage employees to share their knowledge since they are benefiting from it as well.

Implement a Change Control/Management System. With the rapid and continuous change in technology and increase in competition, clients are not sure of what they need in a software product [14], resulting in many unreasonable and often unnecessary requirements. Therefore, this process has to be controlled by a proposed requirement change model. The process should start with creating a "change request", then Tech-K would start "assessing the change", where developers analyze the change request, identify the system components to be changed, estimate the cost required for this change and pinpoint potential conflicts with other requirements. The next step is "deciding on the change", where a decision is reached whether the change will be implemented or not depending on the knowledge assessed including rationale, relationships between requirements and design components, and impacts of change on the development process. If the changes are approved, they move to the next stage "planning the change", where the developers divide the new modifications into tasks and allocate them to responsible parties. At last, in "executing the change" stage, the changes are executed and then further validated after completion.

Implement Proper Testing/Quality Control. As discussed earlier, Tech-K doesn't have dedicated testers and the employees test the software products themselves before submitting to the clients. This results in undiscovered bugs or crashes leading to problems with the client. The proposed solution is to hire dedicated testers and apply a comprehensive testing methodology where all identified bugs are categorized with detailed steps to reproduce and actual vs expected behaviors thoroughly documented. Bug fixes should be updated and documented as well with multiple testing cycles. This will improve the quality of the products and enhance the reputation of Tech-K.

Usage of Knowledge Management Systems/Tools. UI designers at Tech-K used Adobe Photoshop to create full designs, then cut individual graphical assets with the different sizes and specs, then send them to the developers through email or using a flash drive which is not optimal by any means. Using Zeplin (*Zeplin* n.d.), they can turns designs into specifications and related snippets of code corresponding to design components for each screen, with the individual graphic components such as icons, fonts, color schemes, different sizes and many more are ready for the developers to download and use. Slack was used at Tech-K, but it was not efficient and didn't need the required needs. The proposed tool is Basecamp (*Basecamp* n.d.), which is a project management application that provides all tools to create to-do lists, message team members, create schedules, upload documents, check-in and create groups/projects. It enhances collaboration and work flow management by organizing communications and projects. Different projects can be created where all related information is saved including full documentation, any changes in requirements, deadlines, milestones, task allocations to developers, client feedback, to-do lists, testing results, and many more. Different teams can also be created where team members can communicate easily and share related knowledge. Schedules, announcements and many other features are also present. This helps in keeping all the information and knowledge about projects in an organized way that will always be available as assets that employees can always go back to.

Currently, the back-end developer at Tech-K creates the API requests and sends the links of each request through messages on Slack. This is a poor technique since they can't be tested and are hard to find later. The proposed tool is Postman (*Postman* n.d.), which is a collaborative tool for creating RESTful APIs for the projects, allowing the back-end developer and other developers to test the requests/responses of the links to make sure they are in the required format and working as expected. It is much easier than having to write separate unit tests to confirm that the API is working correctly. It enables the organization of APIs and it has a user-friendly user interface making it easy to use. Moreover, developers need proper training for Git usage. Detailed commit messages describing the changes done in the code should be added. Also, bulk commits with unrelated files should be avoided, and only related files with relevant modifications should be included in the same commit. This will enable better tracking of changes and makes looking at the files' history easier.

5 Conclusion

This study demonstrated the usage, benefits and impact of KM in SE based on a case study conducted in an Egyptian IT startup, Tech-K. The activities in the lifecycle of software development were examined. The effect of knowledge sharing, culture, technology, expertise, leadership, resources' utilization was also evaluated. The results show that the techniques and methodologies currently used at Tech-K were poor and inadequate to apply KM in an effective manner that would benefit the company. Several problems were identified including not learning from previous mistakes in older projects, lack of knowledge sharing, staff turnover leading to loss of knowledge, poor requirements gathering, lack of a change control system, poor documentation, lack of testing leading to poor quality software products and generally a lack of a proper knowledge map. All of these issues affect the performance of Tech-K negatively, threaten their presence in the market and stand in the way of their growth or stabilization. Potential solutions were proposed to apply KM properly at Tech-K. The first solution is to create a knowledge repository to encourage knowledge sharing and reuse the existing knowledge by documenting frequently encountered errors with solutions, templates, external libraries, code snippets and other useful knowledge sources. Another solution is implementing a change control system to manage the non-stop change of requirements from the clients and reign it in to avoid any negative impact on the projects or the company, thus making sure that only the necessary and beneficial requirements are approved. Implementing a proper testing methodology is also proposed by hiring a dedicated tester and performing clear testing documentation with all faced bugs and issues. The last proposed solution was using some tools and applications to facilitate the flow of knowledge at Tech-K such as Zeplin, Basecamp and Postman. Applying these solutions will significantly improve KM at Tech-K, enhance the quality of software products and boost the entire company's performance leading it to the path of becoming a successful stable organization.

References

1. Foote A, Halawi LA (2018) Knowledge management models within information technology projects. J Comput Inf Syst 58(1):89–97. https://doi.org/10.1080/08874417.2016.1198941
2. Rong G, Liu X, Gu S, Shao D (2018) A goal-driven framework in support of knowledge management. In: Proceedings—Asia-Pacific software engineering conference, APSEC, 2018, vol 2017 Dec, pp 289–297. https://doi.org/10.1109/APSEC.2017.35
3. Razzak MA, Mite D (2015) Knowledge management in globally distributed agile projects—lesson learned. In: Proceedings—2015 IEEE 10th international conference on global software engineering, ICGSE 2015, pp 81–89. https://doi.org/10.1109/ICGSE.2015.22
4. Ochoa SF, Robbes R, Marques M, Silvestre L, Quispe A (2017) What differentiates chilean niche software companies: business knowledge and reputation. IEEE Softw 34(3):96–103. https://doi.org/10.1109/MS.2017.64

5. Pham QT, Nguyen DT (2017) An empirical investigation of knowledge management in Vietnamese SMEs. In: Proceedings of the 2017 17th international conference on computational science and its applications, ICCSA 2017. https://doi.org/10.1109/ICCSA.2017.8000016
6. Zafar I, Shaheen A, Nazir AK, Maqbool B, Butt WH, Zeb J (2018) Why Pakistani software companies don't use best practices for requirement engineering processes. In: 2018 IEEE 9th Annual information technology, electronics and mobile communication conference, IEMCON 2018, 2019, pp 996–999. https://doi.org/10.1109/IEMCON.2018.8614913
7. Riaz MN, Buriro A, Mahboob A (2019) The effect of software development project team structure on the process of knowledge sharing: an empirical study. In: 2019 2nd International conference on computing, mathematics and engineering technologies, iCoMET 2019. https://doi.org/10.1109/ICOMET.2019.8673504
8. Al Hafidz MU, Sensuse DI (2019) The effect of knowledge management system on software development process with scrum. In: ICICOS 2019—Proceedings of 3rd international conference on informatics and computational sciences: accelerating informatics and computational research for smarter society in the era of industry 4.0. https://doi.org/10.1109/ICICoS48119.2019.8982506
9. Vasanthapriyan S, Tian J, Xiang J (2015) A survey on knowledge management in software engineering,. In: Proceedings—2015 IEEE international conference on software quality, reliability and security-companion, QRS-C 2015, pp 237–244. https://doi.org/10.1109/QRS-C.2015.48
10. Rodrigues LLR, Mathew AO (2019) Knowledge management technology, knowledge sharing and learning—a case study. In: 2019 International conference on automation, computational and technology management, ICACTM 2019, pp 273–276. https://doi.org/10.1109/ICACTM.2019.8776779
11. Jahan MS, Talha Riaz M, Arif KS, Abbas M (2019) Software project management and its tools in practice in IT Industry of Pakistan. In: 2019 2nd International conference on computing, mathematics and engineering technologies, iCoMET 2019. https://doi.org/10.1109/ICOMET.2019.8673535
12. Ouriques R, Britto R, Wnuk K, Ouriques JF, Gorschek T (2019) A method to evaluate knowledge resources in agile software development. In: International symposium on empirical software engineering and measurement, vol 2019, Sept 2019. https://doi.org/10.1109/ESEM.2019.8870167
13. Vasanthapriyan S, Tian J, Zhao D, Xiong S, Xiang J (2017) An ontology-based knowledge sharing portal for software testing. In: Proceedings—2017 IEEE International conference on software quality, reliability and security companion, QRS-C 2017, pp 472–479. https://doi.org/10.1109/QRS-C.2017.82
14. Chaisricharoen R, Srimaharaj W, Boonyanant P, Laohapensaeng T (2019) Virtual workspace for a small-scale software development company in developing countries case of Thailand. In: International symposium on wireless personal multimedia communications, WPMC, vol 2019, Nov 2019. https://doi.org/10.1109/WPMC48795.2019.9096187