A Study on Probabilistic Schedule Estimation and Refinement of the Small and Medium-Sized Software Development Projects

Donghwoon Kwon[1], Ilnam Jeong[2], Geumchae Yoon[3], Bilal Abu Bakr[1], Ki Pyung Kim[4 *]

[1] Department of Computer Science and Information Systems, Texas A&M University-Commerce, Commerce, TX, USA

[2] Department of Computer Science, George Washington University, Washington D.C, USA

[3] ASTI Manufacturing Ltd., Farmers Branch, TX, USA

[4] School of Natural and Built Environments, University of South Australia, Adelaide, SA, Australia

* Corresponding author. Tel.: +61 (0)40 1917 519; email: ki.kim@unisa.edu.au

Manuscript submitted February 9, 2018; accepted March 9, 2018.
doi: 10.17706/jsw.13.3.201-211

Abstract: This paper proposes a research framework for objective and probabilistic project schedule estimation and refinement targeted at a small and medium-sized project, as well as, verification of the research framework. Graphic User Interface (GUI) and design documents, which are composed of a class diagram, sequence diagram, and normalized database, are created to count the accurate number of Function Point (FP). The counted FP, which is the main input for calculating Adjusted Function Point (AFP), is then used to estimate an accurate project size, recalculate human resource efforts, and yield single point-based project schedule estimation. According to programmer's subjective judgment, application development is supposed to be completed within 10 business days. Yet, objective 3-point schedule estimation shows that the success rate of completion within 10 business days is infeasible. Additionally, single point estimation shows that at least 1.313 months (approximately 30 business days) is required to complete application development if more than 80% success rate is expected. The effectiveness of the research framework is successfully demonstrated though development of the actual Android OS-based mobile application (EzMaintenance.apk).

Key words: Android application, probabilistic schedule, small to medium projects.

1. Introduction

To complete a project successfully, managing project triple constraint such as project scope, time, and cost is vital because one of these three cannot change without affecting the others [1]. Only 29% of all software projects were successful completed in 2015 according to the CHAOS report [2]. One interesting point in this report is that smaller projects have more success rate than larger projects while small to medium projects indicate relatively high failure rate 59%, which is more than a half [2]. Based on this data, this research starts from the point by asking the question of what causes such lower success rate of the small and medium-sized project.

There are seven reasons causing the failure of the IT projects, and these can be summarized into three major reasons: i) not adopting Software Development Life Cycle (SDLC) methodology; ii) poor planning; and iii) inappropriate human resource management [3]. An iterative process, which is composed of planning, analysis, design, implementation, and maintenance, is a part of the SDLC methodology. The main goal of a planning phase in the iterative process is to identify project scope, determine the size and level of effort, identify the Critical Path Activities (CPA), also known as Critical Path Analysis (CPA), etc. [4]. The most remarkable advantage of CPA in terms of management is to track and monitor project status focusing on the scheduled completion date of each critical path activity [4]. The key methodology to identify robust critical path activities prior to the planning phase is objective and accurate schedule estimation as early as possible. Additionally, the objectively and accurately estimated project schedule needs to be refined after initial schedule estimation. As a project has its own unique condition and constraint, it is unlikely to have a magical tools to conduct the probabilistic schedule estimation and its refinement, and most models focused on constructing a relationship between software size and human resource effort for either project schedule or cost estimation [5]-[7]. In addition, while the previous researches successfully proposed research frameworks for probabilistic schedule estimation and its refinement based on Unified Modeling Language (UML) diagrams, the biggest limitation was assumptions-based demonstration targeted at large-sized projects [8], [9]. This limitation injects a necessity into verification of the previously proposed research frameworks [8], [9] based on the project size and actual development. The rest of this paper is organized as follows: Introduction, Related Work, Methodology, Demonstration, and Conclusion.

2. Related Work

This section describes a short summary of literature review related to probabilistic schedule estimation and its refinement in this section from previous studies [8], [9].

2.1. Size Estimation

Project size estimation in the early stage is related to defining project scope using the Work Breakdown Structure (WBS). There are two different types of WBS such as phase-based WBS and deliverable-based WBS. The previous research demonstrated that deliverable-based WBS could be more powerful than phase-based WBS when Function Point Analysis (FPA) is applied.

There are also two different techniques to estimate the project size of software development, i.e. counting Source Lines of Code (SLOC) and FPA. The main difference between SLOC and FPA is that more accurate size estimation can be expected by FPA rather than SLOC. Additionally, FPA is composed of five functional components such as External Input (EI), External Output (EO), External Inquiry (EQ), Internal Logical File (ILF), and External Interface File (EIF). So, the fundamental concept of FPA is to count the accurate number of each functional component. Yet, since ILF is associated with database tables, not database, it needs to count the number of database tables for counting the number of ILF.

Size estimation at the early stage is challenging task because only high level requirements, not very specific stakeholders' requirements, are available. Given the lack of information, counting the accurate number of FP is the main task for size estimation, but unfortunately, there are no magical tools and techniques to count very accurate FP with insufficient software specifications. For this reason, 3-point estimation associated with Central Limited Theorem (CLT) and Monte Carlo simulation came to the fore.

Based on the 3-point estimation, initial project size estimation forms a triangular distribution so that such triangular distribution needs to be transformed into a normal distribution using CLT and Monte Carlo simulation. Since our initial research focused on the large-sized project, the required number of samples was at least thirty due to the concept of CLT. Note that thirty samples were considered as thirty work packages, and this point was the key concept to apply CLT. Once all the mentioned concepts are performed, the Unadjusted Function Point (UFP) is calculated for size estimation, and this result will be used for resource estimation.

2.2. Resource Estimation

Resource estimation is aimed at estimating the resource effort of each work package using the value of Project Delivery Rate (PDR) from the International Software Benchmarking Standards Group (ISBSG) repository. Two different values of PDR can be obtained: (i) find the fixed value of PDR from the ISBSG repository and use it or (ii) calculate it by the equation:

$$C^{*}SizeE1^{*}Maximum\ \ Team\ \ SizeE2 \tag{1}$$

where the value of C, E1, and E2 can be obtained from the ISBSG repository based on the development type, development platform, and programming language.

Keep in mind that either PDR should be selected by the circumstances of the project or project manager's judgment. Furthermore, since the calculated resource effort is Person Months (PM), this value needs to be converted into normal calendar months.

2.3. Probabilistic Schedule Estimation

Once both size and resource estimations are conducted, probabilistic project schedule can be generated after making the predecessor relationship of the work packages. Note that the duration of the upper WBS level and the entire project can be automatically calculated based on the duration and the predecessor relationship of the lowest WBS level. Furthermore, the concept of CLT and Monte Carlo simulation should be applied again to estimate final probabilistic project schedule.

2.4. Refinement of the Estimated Probabilistic Schedule

The main motivation of this section is because the estimated probabilistic project schedule in the early stage resulted from 3-point estimation in conjunction with Monte Carlo simulation in terms of size and final probabilistic schedule estimation is no longer valid in the planning phase. This is due to the fact that each functional component of FPA can be accurately counted through the UML-based design documents focusing on one-point estimation.

There are sixteen UML diagrams categorized into two groups such as static and behavior. Among them, a class diagram and sequence diagram with consideration of a relational database need to be used for counting five functional components. Yet, there is one issue with respect to counting EIs using the class diagram; that is, the number of EIs should be carefully determined based on class operations. In addition, Adjusted Function Point (AFP), not UFP, should be used this time. AFP is calculated by the following equation:

$$\text{Count total (UFP)} \times [0.65 + 0.01 \times \Sigma(F_i)] \tag{2}$$

where F_i is Complexity Adjustment Values (CAV). Note that the summation of F_i is obtained by answering fourteen general system characteristics using a scale from 0 to 5. Once AFP is calculated for size estimation, resource effort needs to be recalculated and then, the final one-point estimation-based project schedule can be calculated.

In summary, the overall research framework with necessary tools and techniques referring to all those concepts above is depicted in Figure 1 below.

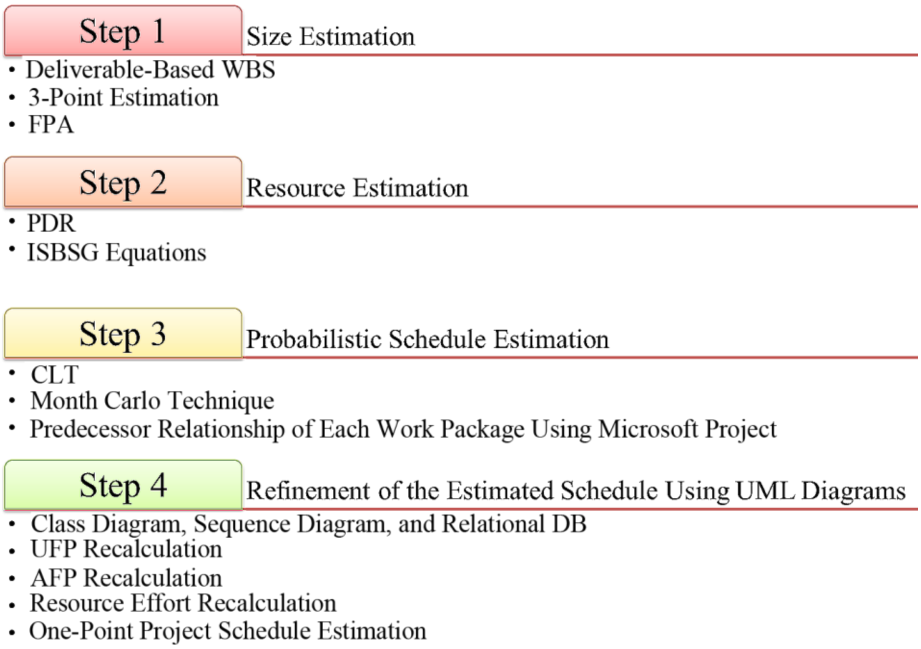

Fig. 1. Overall research framework of probabilistic schedule estimation and refinement.

3. Methodology

Through the literature review, the following major limitations have been identified: (i) the research frameworks related to probabilistic schedule estimation and refinement were successfully verified, but only large-sized project which had more than thirty work packages was considered, (ii) assumptions and simulation-based demonstration was performed for verifying the effectiveness of the research frameworks and, (iii) database was not designed based on the normalization concept for counting ILF.

Based on the identified limitations, the following research design has been devised: (i) probabilistic-based technique for schedule estimation and refinement targeted at small and medium-sized projects will be used and (ii) actual software development based on design factors such as the class diagram, sequence diagram, normalized relational database, and Graphic User Interface (GUI) will be conducted. Therefore, this research will cover four stages: the proposal preparation stage, planning stage, design stage, and development stage.

After setting up the research direction, one question is on the rise: “What is the criterion to differentiate large-sized projects and small and medium-size projects?”

To answer this question, we focused on the number of work packages. As mentioned earlier, large-sized projects have more than thirty work packages so that the number of work packages for a small and mid-sized project is absolutely less than thirty [8]. Yet, this point leads another issue regarding CLT and normal distribution. If there are less than thirty work packages (samples), it is impossible to apply the CLT concept to generate the normal sampling distribution [8]. However, since distributions such as a normal, triangular, or Beta distribution commonly belong to a probability distribution, it is possible to produce the probability through Monte Carlo simulation [10]. This point has a significant meaning because if it is possible to estimate probabilistic project size, other steps, i.e. resource estimation (step 2), probabilistic schedule estimation (step 3), and refinement of the estimated probabilistic schedule (step 4) in Figure 1 can be also performed. Therefore, the proposed and revised research framework is depicted in Figure 2 below.

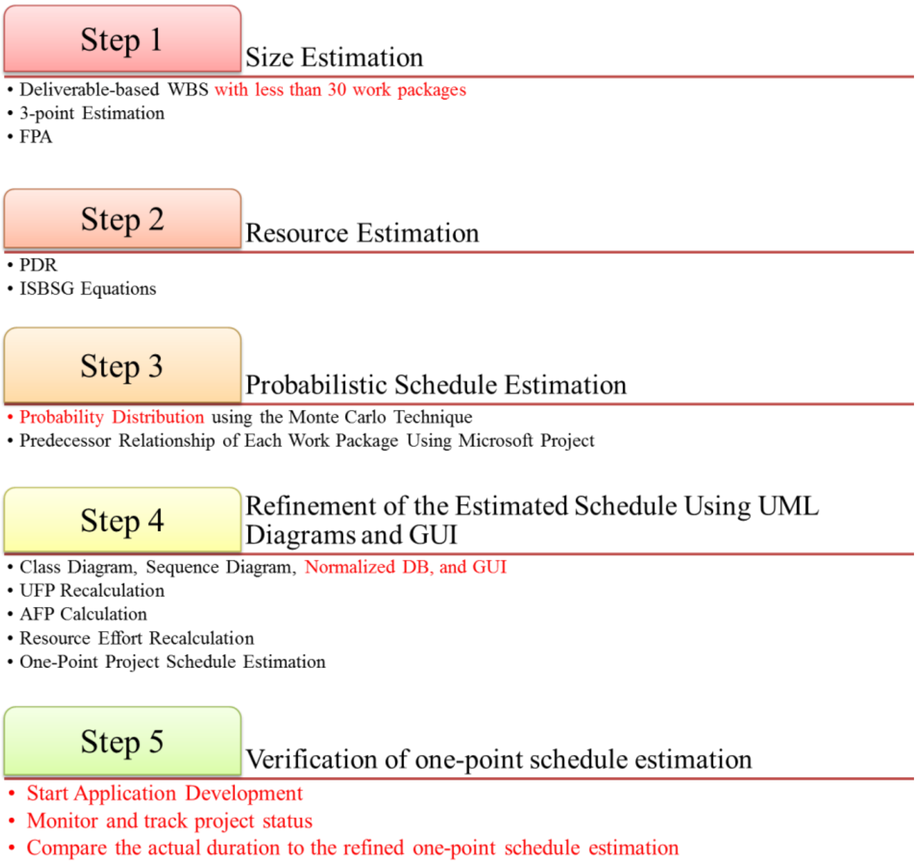

Fig. 2. Revised research framework.

Note that differences between Fig. 1 and Fig. 2 are highlighted in Red in Fig. 2.

4. Demonstration

To demonstrate the framework, an Android OS-based application has been developed. This application was intended for a small project, and it addressed the issue of vehicle maintenance. One of the technological requirements for this application was to use SQLite, which was employed to implement the SQL database engine for the Android application. Moreover, the application allowed users to maintain their vehicle on time and on mileage by listing alerts.

As shown in Fig. 2, we first focused on performing steps 1 through 3 for probabilistic project schedule estimation based on four functionalities such as vehicle registration, update vehicle maintenance, maintenance guideline, and maintenance alerts. The Precedence Diagraming Method (PDM) of each functionality was Start-to-Start so that only activity which had the longest duration was considered as a critical path activity. In this case, the vehicle registration function became the critical path. The given values for probabilistic schedule estimation were as follows: (i) all weight factors in the equation of UFP calculation were "simple". The weight factor values of EI, EO, EQ, ILF, and EIF were 3, 4, 3, 7, and 5. Once each FP component is counted based on 3-point estimation, UFPs are calculated by the equation, the counted number of each FP component times those numbers, (ii) the value of $\Sigma(Fi)$ is 13 resulted from answering fourteen questions, (iii) the fixed PDR value was used, and its value was 8.0 because of all platform-based and Java programming language to calculate resource effort and, (iv) only one programmer participated in the project so that same PDR could be expected.

The estimated probabilistic project schedule based on descriptions and values above is shown in Table I below.

Table 1. Estimated Probabilistic Project Schedule

Simulation Results		**Probability**	
Minimum	1.151	1%	1.167
Maximum	1.388	5%	1.188
Mean	1.270	10%	1.204
N/A		15%	1.216
		20%	1.226
		25%	1.235
		30%	1.243
		35%	1.250
		40%	1.257
		45%	1.264
		50%	1.270
		55%	1.276
		60%	1.283
		65%	1.290
		70%	1.297
		75%	1.305
		80%	1.314
		85%	1.324
		90%	1.336
		95%	1.352
		99%	1.373

Once probabilistic schedule is estimated, the next step is to recount each function component focusing on single point estimation. The number of EIs based on single point can be obtained by the class diagram and GUI shown in Fig. 3 and 4, respectively, and other functional components such as EOs, ILFs, and EIFs can be identified by the normalized database and sequence diagram shown in Fig. 5 and 6, respectively. Note that we included a part of GUI due to space limitations.

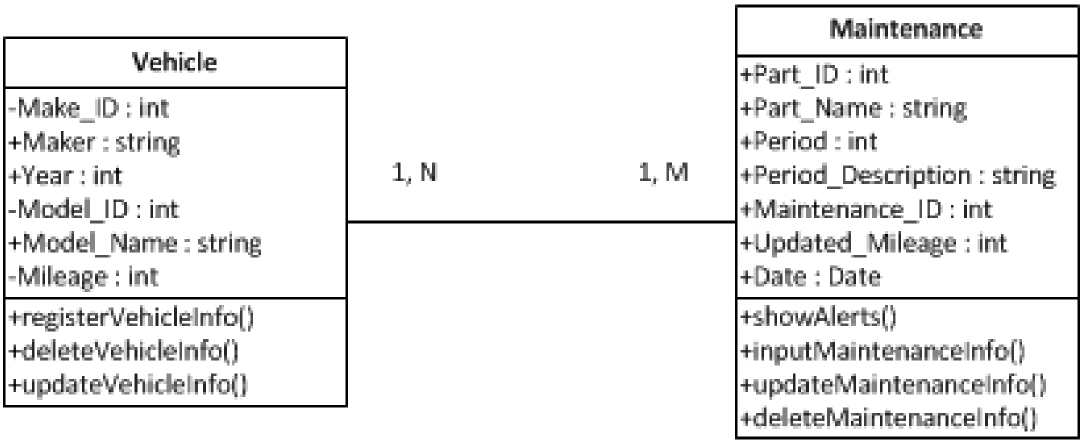

Fig. 3. Class diagram for the application.

Fig. 4. Graphic user interface (GUI) for the app.

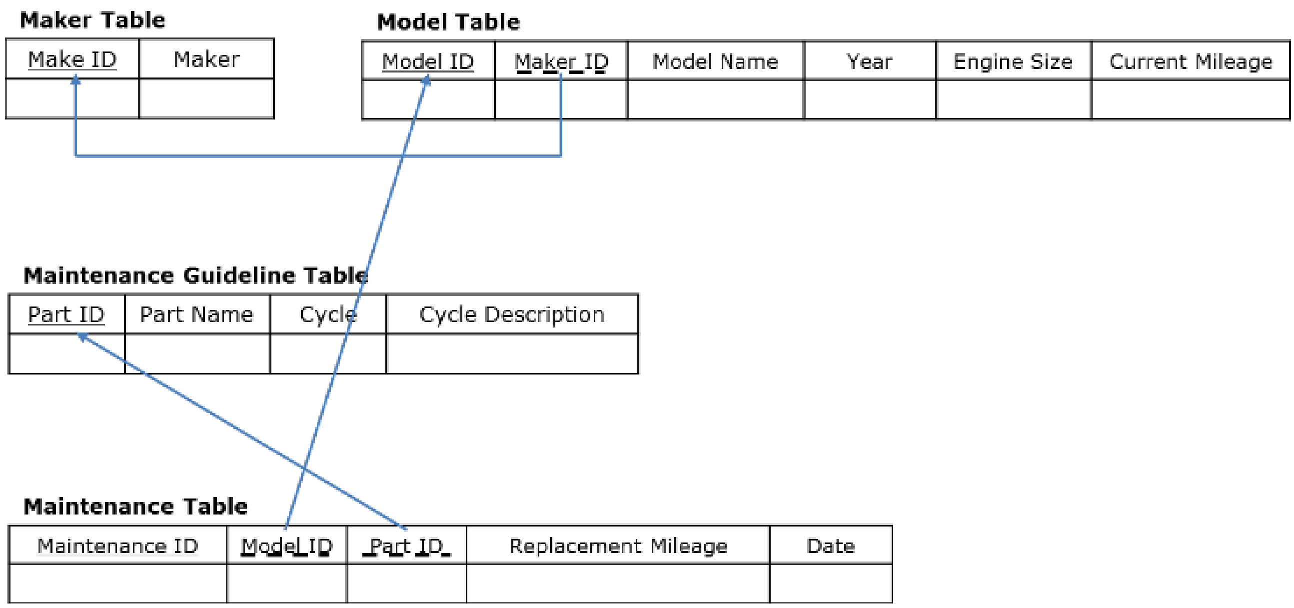

Fig. 5. Normalized database for the application.

According to the design documents including UML diagrams and GUI, details of the functional components for each functionality are:

1) Vehicle Registration Function
   A. EI: Maker_ID: Integer & Maker: String & Year: int & Model_ID: Integer & Model_Name: String & Mileage: Integer
   B. EO: returnSavedVehicleInfo.()
   C. ILF: Maker Table & Model Table
   D. EQ & EIF: None
2) Update Maintenance History
   A. EI: Part_ID: Integer & Part_Name: String & Maintenance_ID: Integer & & Updated_Mileage: Integer & Date: Date
   B. EO: returnUpdatedMaintenanceHistory()
   C. ILF: Maintenance Table & Maintenance Guideline Table
   D. EQ & EIF: None
3) Maintenance Guideline
   A. EI: Part_ID: Integer & Part_Name: String & Period: Integer & Period_Description: String
   B. EO: returnMaintenanceGuideline()
   C. ILF: Maintenance Guideline Table
   D. EQ & EIF: None
4) Maintenance Alert
   A. EI: Maintenance_ID: Integer & Updated_Mileage: Integer & Date: Date
   B. EO: showAlert()
   C. ILF: Model Table & Maintenance Table & Maintenance Guideline Table
   D. EQ & EIF: None

In summary, the recounted functional components and recalculated UFPs are shown in Table 2 below. Note that the sub total of each function point is calculated based on the “simple” weighting factor as described earlier.

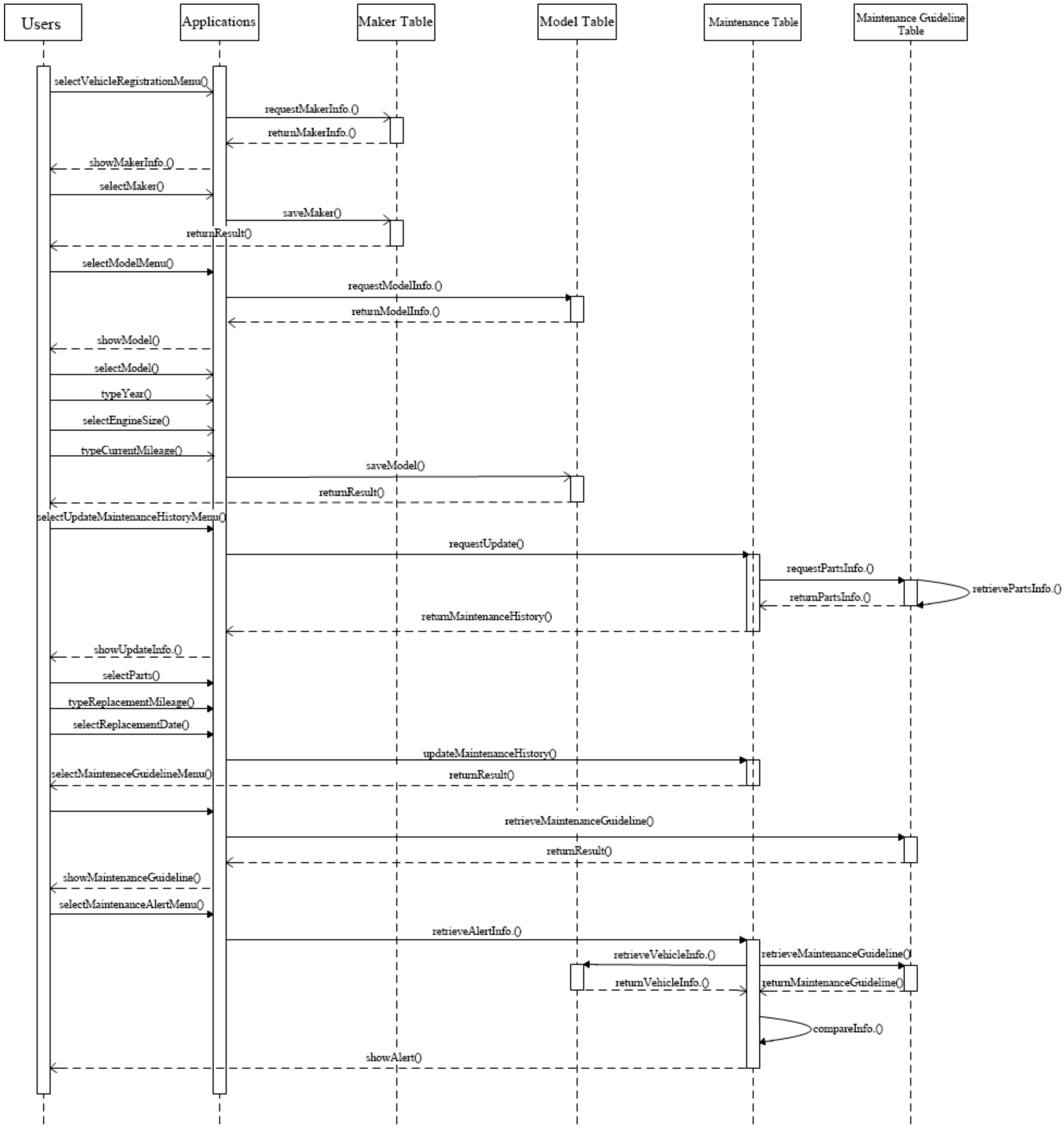

Fig. 6. Sequence diagram.

Table 2. Recalculated UFP Based on Recounted Functional Components

Functions	EI	EO	EQ	ILF	EIF	UFP
Vehicle Registration	6	1	0	2	0	36
Update vehicle maintenance	5	1	0	2	0	33
Maintenance guideline	4	1	0	1	0	23
Maintenance alerts	3	1	0	3	0	34
					Grand Total	126

Among four functionalities, the vehicle registration functionality is selected for further processes because this functionality is the critical path activity as described earlier. So the calculated AFP is 28.08 resulted from the equation,

$$\text{AFP} = \text{Count total (UFP)} \times [0.65 + 0.01 \text{ X } \Sigma(\text{Fi})] \tag{3}$$

where the value of Σ(Fi) is 13 as mentioned earlier. The last process is to recalculate resource effort and project duration using ISBSG regression equations. Since the fixed PDR value of 8.0 is used, the recalculated resource effort is 224.64 (28.08 times 8.0). In addition, 1 programmer works 176 hours per month (8 hours per day times 22 days per month) so that the calculated final schedule was 1.28 month (224.64 divided by 176).

5. Discussion

Based on the recalculated final project schedule (1.28 month equals to approximately 28 business days), its success rate is between 55% and 60% as shown in Table 1. To validate the research framework and estimated project schedule, we started to develop the mobile application. We also measured the duration of actual development and confirmed that the programmer actually took 30 business days. Note that the programmer initially estimated 2 weeks (10 business days) to complete development.

The actual duration of application development has a significant meaning because 2 days were delayed from our schedule estimation. The first reason for delaying 2 days was that the developer took a time to fix the errors of the application database. The second reason was that the developer could not fully assign 8 hours in a development period. This means that we did not successfully monitor, track, and control the estimated schedule in terms of risk and human resource management. Therefore, our suggestion through this actual development and research is that risk and human resource management has to be performed once a project is initiated, and extra time, which indicates more than 80% of the success probability, should be given with consideration of all possible contingencies.

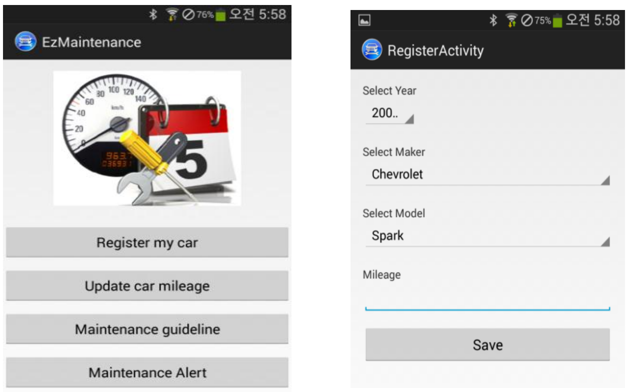

Fig. 7. Part of the developed application.

6. Conclusion

The goal of this research was to develop a methodology of how to accurately estimate and refine the probabilistic project schedule targeted at small and medium-sized projects. A major contribution of this research is that we determined a framework for schedule estimation and refinement of the small and medium-sized projects and demonstrated it through actual application development based on design factors such as UML diagrams, GUIs, and normalized database design.

However, there are certain improvements that should be considered for future study. Since schedule estimation plays a vital role as a guideline, schedule monitoring and controlling with consideration of project risk and human resource management needs to be examined for accurate and successful schedule management. In addition, we realized that software development heavily relies on the programmer's capability so that measurement of coding skills needs to be examined.

The actual duration of application development has a significant meaning because 2 days were delayed from our schedule estimation. The first reason for delaying 2 days was that the developer took a time to fix the errors of the application database. The second reason was that the developer could not fully assign 8 hours in a development period. This means that we did not successfully monitor, track, and control the estimated schedule in terms of risk and human resource management. Therefore, our suggestion through this actual development and research is that risk and human resource management has to be performed once a project is initiated, and extra time, which indicates more than 80% of the success probability, should be given with consideration of all possible contingencies.

References

[1] Wyngaard, C. J. V., Pretorius, J. H. C., & Pretorius, L. (2012). Theory of the triple constraint-a conceptual review. *Proceedings of the 2012 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)* (pp. 1991-1997). Hongkong, China.

[2] Vachon, J. S. (2016). *Why Projects Fail*. White Paper, Inteloom.

[3] Dorsey, P. (2005). *Top 10 Reasons Why Systems Projects Fail*. Dulcian Inc: Woodbridge, NJ.

[4] Rhodes, D. L. (2012). *The Systems Development Life Cycle (SDLC) as a Standard: Beyond the Documentation*. Washington D.C: U.S. Census Bureau.

[5] Sehra, S. K., Brar, Y. S., & Kaur, N. (2011). Soft computing techniquies for software project effort estimation. *International Journal of Advanced Computer and Mathematical Sciences*, 160-167.

[6] Bashir, H. A., & Thomson, V. (2001). Estimating effort and time for design projects. *Canadian Society of Value Analysis*, 1-9.

[7] Borade, J., & Khalkar, V. R. (2013). Software project effort and cost estimation techniques. *International Journal of Advanced Research in Computer Science and Software Engineering*, 730-739.

[8] Kwon, D., & Hammll, R. J. (2013). Early stage probabilistic software project schedule estimation. *Journal of Information Systems Applied Research*, 31-48.

[9] Kwon, D., & Hammell, R. J. (2014). Refinement / verification of early stage probabilistic software project Schedules in the Planning Stage. *Proceedings of the 15th IEEE / ACIS Software Engineering, Artificial Intelligence, Networking and Parallel / Distributed Computing (SNPD)* (pp.239-244).

[10] Kwak, Y., & Ingall, L. (2009). Exploring monte carlo simulation applications for project management. *IEEE Engineering Management Review*, 83-91.

Donghwoon Kwon received his D.Sc degree in applied information technology from Towson University, MD. He has been working as an assistant and adjunct professor in the Department of Computer Science at Texas A&M University-Commerce. The areas of research interests are internet of things (IoT), cloud computing, big data with machine learning, and mobile application, as well as, IT / IS project management in conjunction with software engineering and database management. He participated in the smart connected car research project from August 2016 to November 2017 and developed the deep learning-based blind spot detection system.

Ilnam Jeong received the B.E. degree in information engineering from the National Institute for Lifelong Education, Seoul, Korea, in 2008, and the Master's degrees in computer science from Towson University, Towson, MD, in 2011 and He is currently pursuing Ph.D. degree in computer science at the George Washington University, Washington, D.C.

In 2016, he joined the Public Safety Network research team at National Institute of Standards and Technology(NIST), Gaithersburg, MD as a guest researcher. During the time, he researched overall studies about vehicle-to-everything (V2X) technology specially LTE-based V2X technology which can enable inter-vehicle communication to support various safety features. He has been researching in the evolved multimedia broadcast multicast services on LTE-based V2X network. V2x communication between vehicles, a large amount of data is transmitted. As the number of vehicle increases, the amount of data increases exponentially in some scenarios. He is interested in software algorithm to improve systems and machine learning.

Geumchae Yoon received her M.S in computer science from Texas A&M University-Commerce and B.A in sustainability from Arizona State University. She has been working as a data forecast analyst at ASTI manufacturing LTd., TX, USA since 2017. The areas of research interests are database management, project management, statistics, and business analytics.

Dilal Abu Bakr received his Ph.D in computer science from Western Michigan University in 2011 and three M.S degrees in mathematics and computer science from Western Michigan University and University of the Punjab, India. He is currently an assistant professor in the Department of Computer Science at Texas A&M University-Commerce. The areas of research interest span from wireless sensor networks, parallel and distributed systems, and cloud system. Prior to joining in Texas A&M University-Commerce, he was an Assistant Professor at Alcorn State University.

Ki Pyung Kim is a PhD, PMP, PMI-SP, MCTS, is a lecturer in the construction project management program at University of South Australia. He obtained his PhD for research on integrating Building Information Modelling (BIM) into housing refurbishment to determine the most affordable refurbishment solution based on LCC and LCA. His research interests and expertise lie in BIM system and application, sustainable housing refurbishment, data/information management and construction time management. He was awarded Highly Commended Academic Paper at the Pacific Association of Quantity Surveyors Congress 2016, Excellence in Volunteer Recognition for PMI Knowledge Asset Localization in South Korea 2015, and the Best Mention Prize at the BIM Competition 2013 hosted by buildingSMART Singapore. Recently, he was awarded the Australia-Korea Foundation Research Funding 2017 and, with David Ness, an Arup Global Research Challenge Grant 2017 to develop a Cloud BIM Data Platform to enable life-cycle stewardship and reuse of building components and their provision as a service.