Early Stage Probabilistic Software Project Schedule Estimation

Donghwoon Kwon
dkwon3@students.towson.edu

Robert J. Hammell II
rhammell@towson.edu

Department of Computer and Information Sciences
Towson University
Towson, MD 21252, USA

Abstract

This paper proposes a framework for the objective and accurate estimation of software project schedules in the proposal preparation stage, while taking into account project uncertainty. The project size, resource effort, and the Project Delivery Rate (PDR) value are fundamental to the software project schedule estimation process, and such factors are calculated and determined by function point analysis and the equations and data repository of the International Software Benchmarking Standards Group (ISBSG). Project uncertainty is accounted for so that numerous possibilities may be explored. The framework provides a probabilistic approach by using the @RISK tool which is based on Program Evaluation Review Technique (PERT) analysis. This approach generates a schedule estimation *range*; this range is then narrowed by applying the Central Limit Theorem (CLT) to the Work Breakdown Structure (WBS) which reduces the overall uncertainty and increases the schedule accuracy. WBS Chart Pro is used to create the project Work Breakdown Structure, and Microsoft Project 2010 is used to determine the project critical path.

Keywords: Software Project Schedule Estimation, Function Point, Effort Estimation, Central Limit Theorem

1. INTRODUCTION

According to the Project Management body of Knowledge (PMBOK) (Project Management Institute, 2008), project management is driven by 9 knowledge areas and 42 processes, and it consists of the following 5 process groups: initiating, planning, executing, monitoring and controlling, and closing. Among these groups, planning accounts for 20 out of 42 total processes (approximately 48%) so that its role is very significant; that is, regardless of project type, a good project plan in vital since poor project planning can lead directly to project failure. The research (Project Management Solutions, Inc., 2012) shows that only 47% of information technology (IT) projects are completed successfully and 37% of IT projects are repaired or cancelled. One of the reasons that IT projects are cancelled is that they frequently go over schedule (Emam & Koru, 2008), so schedule estimation is critical within the planning process group.

While poor planning and estimation as well as unrealistic schedules and budgets can be associated with projects of any kind, these issues have been specifically mentioned as challenges for software projects (Hughes & Cotterell, 2009). Other issues discussed as

unique to software development projects include inadequate quality controls, a lack of understanding between clients and developers, the volatility of software requirements, and problems associated with using ontologies in the requirement elicitation stage (Hughes & Cotterell, 2009, Ogwueleka, 2012). The work in this paper focuses on the issues associated with respect to schedule estimation.

It is important to note that schedule estimation is necessary prior to the actual initiation of a project (Gido & Clements, 2009); in other words, contractors must estimate the project schedule to prove that they are able to complete a project within the given time frame required in the Request for Proposal (RFP). However, it is difficult to estimate the project schedule due to the fact that uncertainty is inherent is all types of projects (Xiao Liu et. al., 2009). Furthermore, such uncertainty results from the fact that contractors depend on high level Statement of Work (SOW), requirements and deliverables in the proposal preparation stage. It is especially difficult for software project managers to estimate accurate project schedules because they should consider a variety of factors such as project size, resource effort, and so on, and it is very hard to figure out such factors without the agreed design documentation. This lack of information injects uncertainty into the planning process, causing timetable deviations (a major cause of overall project schedule overruns), and a chance for project failure or cancellation (Emam & Koru, 2008, Tesch et. al., 2007).

Another important fact is that project risks originate from uncertainty, and managing such risks is critical (Project Management Institute, 2008). There are two kinds of risks: known and unknown and they are presented using the quadrant form in Figure 1 (Douglas & Ra, 2010, Dobson & Leemann, 2010).

The major objective of the "knowns unknowns" quadrant is to transform *unknown unknowns* into *known knowns*, *known unknowns*, or *unknown knowns*. For the framework proposed herein, the risk type is *known unknowns* because the process used to estimate activity durations is known, but the outcome is unknown. Although the outcome is unknown, a project manager can create outcomes with probability using risk management tools and techniques.

The probabilistic approach keeps all possibilities in mind so that emphasis is placed on generating a schedule range as opposed to producing a point estimation. This is done based on inferential statistics, which is the fundamental concept of the Central Limit Theorem (CLT) (Smith & Wells, 2006).

Therefore, the thrust of this paper is aimed at probabilistic software schedule estimation in the proposal preparation stage of the project life cycle rather than in the planning stage. We propose methodologies such as the CLT, PERT analysis, International Software Benchmarking Standards Group (ISBSG) equations, and function point calculation based on International Function Point User Group (IFPUG) as well as tools such as WBS chart pro, @RISK, and Microsoft Project 2010. The rest of the paper is organized as follows: Section 2 provides a brief literature review. Section 3 defines the schedule estimation methodology, and Section 4 demonstrates how to estimate a software project schedule within the proposed framework. Lastly, Section 5 provides conclusions and future work.

2. LITERATURE RESEARCH

In this section, we briefly discuss aspects of software project scheduling from the literature that are related to our research objectives.

Initially, it was necessary to find out which factors and processes are crucial to estimate software project schedules, and the literature defined the following software project estimation processes sequentially: 1) requirements collections, 2) product size estimation, 3) effort estimation, 4) schedule creation, 5) cost estimation, 6) estimation approval, 7) and product development (Nasir, 2006). Since our work is concerned only through step 4 (schedule creation), the main factors about which we will be concerned are size and effort.

Based on the above results, it is clear that methodologies used to estimate size needed to be examined. The research (Malik, 2010) introduced six major size estimation categories: 1) expert judgment, 2) analogy-based estimation, 3) group consensus estimation, 4) decomposition, 5) probabilistic methods which refer to PERT sizing, and 6) hybrids of the previous categories. The research also mentioned how to measure size in terms of two categories: Function Point Analysis (FPA) and physical size measurement, the latter being

related to Source Lines of Code (SLOC) (Malik, 2010).

Due to the fact that project uncertainty in the early stage is very high, we believe that probabilistic methods have great potential for solving the size estimation problem with respect to project scheduling. Additionally, as it is mentioned in Figure 1, a project manager is aware of estimating the project schedule which is known as a process, but the outcome of schedule estimation is unknown. For this reason, we focused on the probabilistic method which refers to PERT sizing based on FPA. Also, FPA was selected over SLOC because function points can be more readily and accurately measured in the requirements phase (Nassif et. al., 2010, Lind & Heldal, 2010). This point has an important meaning because we focus on schedule estimation in the proposal preparation stage with high level requirements.

The other factor that clearly needed to be examined within the literature was how to estimate effort. There are a variety of models to estimate effort: analogy-based effort estimation (Chiu & Huang, 2007, Kocaguneli et. al., 2012, Cherjee et. al., 2009, Basha & Dhavachevan, 2010), regression equations, COCOMO, and so on. (Basha & P., 2010, ISBSG, 2010). Among them, we selected to use the regression equations which were generated by data analysis of the ISBSG repository based on IFPUG FPs (ISBSG, 2010). This is due to the fact that they are the most suitable in the early estimation stage (ISBSG, 2010). Moreover, COCOMO models such as COCOMO 81 and COCOMO II mostly use Line of Code (LOC) instead of FP for effort estimation; although COCOMO II uses FP, it possibly causes error in effort estimation (Basha & Dhavachevan, 2010).

According to the literature research above, we determined that software size and resource effort are fundamental factors for schedule estimation, and FPA and the ISBSG regression equations are necessary methodologies to measure and calculate project size and resource effort. Yet, those factors and methodologies should be performed in the pre-bid stage because accurate pre-bid estimation leads to successful project completion and better project progress (Nasir, 2006). This point corresponds with our study objective to produce better project schedule estimates in the proposal stage.

3. SCHEDULE ESTIMATION METHODOLOGY

The proposed model for software project schedule estimation is depicted in Figure 2 (See Figure B1 in Appendix 2 for the enlarged version). This section will provide a step-by-step explanation of the methodology.

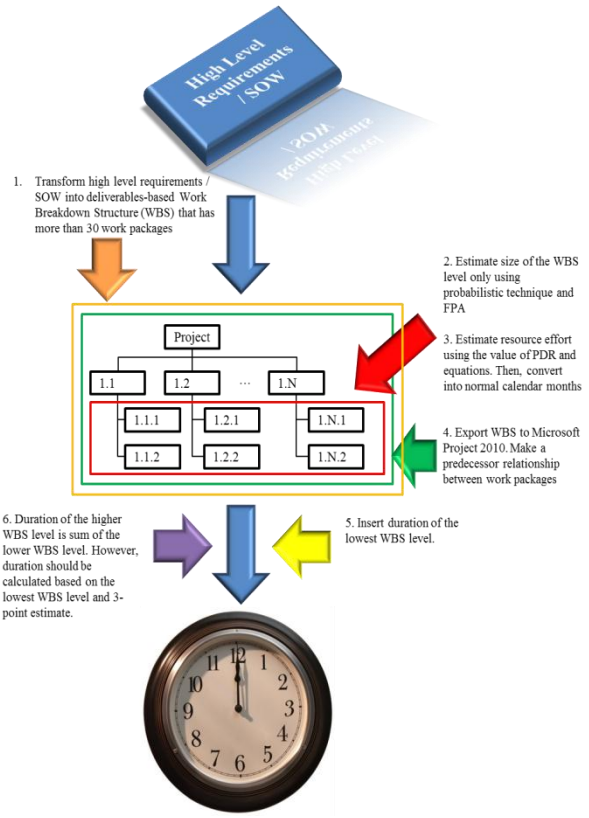

Figure 2 - Overall research model.

Size Estimation

Once the RFP is released, the first step towards schedule estimation is to identify and understand the stakeholder's requirements and to create a project Work Breakdown Structure (WBS) to represent the project scope. There are two kinds of WBS creation: phase-based and deliverables-based (Project Management Institute, 2008). For our purposes, deliverables-based WBS is more effective than phase-based because the major functionalities of the target system or application can be placed in the WBS level 2 and those functionalities have specific modules as work packages which can be placed in the WBS level 3. The research (Bottenfield, 2005) also defined that there are 2 types of WBS: product WBS and activity WBS. However, these basically map to deliverables-based WBS

and phase-based WBS, respectively. Since we follow the software functional size method of IFPUG, the deliverables-based WBS is needed so that we may apply function point analysis.

The functional size method uses five functional component types: External Inputs (EI), External Output (EO), External Inquiry (EQ), Internal Logical Files (ILF), and External Interface Files (EIF) (Cuadrado-Gallego et. al., 2010). Figure 3 depicts a deliverables-based WBS.

The second step is to count the function points of each WBS level. Our assumption is that each deliverable and work package follows the architecture as shown in Figure 4 so that a software project manager is able to count how many EIs, EOs, and EQs ILFs, EIFs are in each work package.

One thing to keep in mind is that since this estimation phase is performed in the proposal stage without design documentation to which all stakeholders agree, it is very difficult to figure out the exact number of EIs, EOs, EQs, ILFs, and EIFs (we assume a lack of historical records and experience for similar projects). For example, assume that a primary stakeholder wants to develop an online tire market and one of the functionalities is to search for tires. The number of EIs can be one if a tire is searched for by brand, but if a tire is searched for using brand, size, and vehicle model, the number of EIs is three. For this reason, the number of each EI, EO, EQ, ILF, and EIF should be measured in terms of a probabilistic technique (optimistic, most likely, and pessimistic), and then a software project manager is able to calculate Unadjusted Function Point (UFP) by the equations in the UFP calculation (Singhal & Srikrishna, 2008, Pressman, 2009).

However, there are two issues that complicate the size estimation. The first issue is size measurement at WBS level 2. The simplest way to calculate UFPs at WBS level 2 is for a software project manager to count the UFPs of the WBS level 3. However, we realized that this methodology may not make sense in some cases. For instance, assume that there is an employee management functionality that consists of 2 modules such as employee registration and employee deletion. The number of ILF relates to the number of database tables. Now, assume that both modules are performed using a single database table; thus, since both modules require only the same database table, the number of

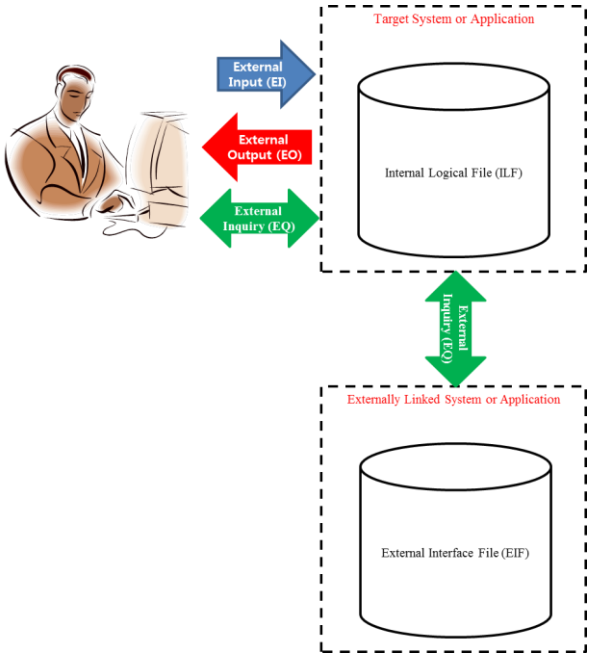

Figure 4 - Architecture of function point counting

ILFs in the employee management functionality is 1, not 2. For more details, consider an example in terms of database Structured Query Language (SQL). If an employee table consists of 3 fields such as Social Security Number (SSN), first name, and last name, the SQL command of employee registration and deletion is as follows based on SQL command syntax.

1. Syntax of insert command
   A. INSERT INTO table_name (table column1, column2, …) VALUES (data, data);
2. SQL command for employee registration
   A. INSERT INTO Employee (SSN, first_name, last_name) VALUES (123456789, 'Sam', 'Smith');
3. Syntax of delete command
   A. DELETE FROM table_name WHERE condition
4. SQL command for employee deletion
   A. DELETE FROM Employee WHERE SSN=123456789;

From the above, it is easy to figure out that the same database table is used for both modules. As a result, it clearly points out that the sum of counted UFPs from WBS level 3 does not always work properly for WBS level 2.

The second issue is that while, theoretically, the equations in the UFP calculation are valid, the

Expected Count value is a weighted average of a 3-point estimation, so it is considered as the mean value. The meaning of the mean value is a population mean that indicates only 50% of a probability distribution result, so it does not correspond to our final research objective that generates the range of schedule estimation with consideration of all possibilities based on inferential statistics. For example, suppose that the number of EIs, EOs, EQs, ILFs, and EIFs is as shown in Table 1.

Table 1 - Example number of functional component types

Category	3-Point Estimation		
	Op	ML	Pess
EI	8	10	13
EO	6	12	16
EQ	4	7	10
ILF	3	5	7
EIF	9	12	15

According to Table 1, each functional components type indicates a triangular distribution which is shown in Figure 5.

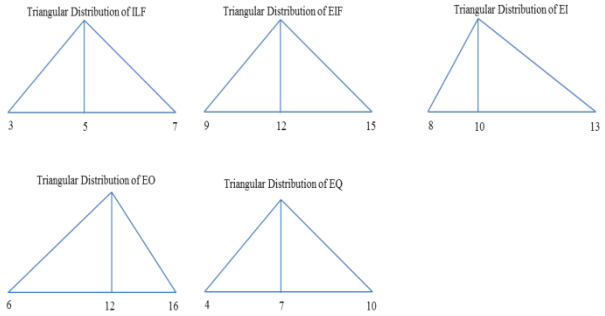

Figure 5 - Triangular distribution

There are many possibilities to pick any random numbers between the optimistic and pessimistic values of each functional component type; this relates to the Monte Carlo technique based on iterative simulations.

The key idea of using the Monte Carlo technique is that randomly chosen input values are used to transform the triangular distributions into normal probability distributions which will be calculated from the iterations (Project Management Institute, 2008). Therefore, it is required to simulate all cases hundreds or thousands times to calculate UFP for software size measurement.

In summary, the second step provides how to estimate project size using UFPs and requires the Monte Carlo technique for 3-point estimation which is based on triangular distribution.

Resource Effort Estimation

The third step of Figure 2 is to estimate the resource effort for each work package using the value of Project Delivery Rate (PDR). As it is mentioned in Sections 1 and 2, the ISBSG equations are used for effort estimation. The equations actually used in this paper are shown in ISBSG regression equations (See Table A1 in Appendix 1, ISBSG, 2010). However, there are two cases that could cause potentially cause a different PDR value.

The first case is that if all work packages are performed by one project team, the value of PDR is possibly the same because each of the work packages is able to have the same productivity rate. The second case is that if each work package is performed by different project teams, the PDR value could be different because a software project manager cannot expect same productivity from each different project team.

For this reason, we propose two solutions for determining the PDR value. The first solution is to find the appropriate fixed PDR value in the ISBSG data repository based on programming language and platform if a software project expects same productivity (ISBSG, 2010). The second solution (ISBSG, 2010) is to use the equation, $C*Size^{E1}*Maximum\ Team\ Size^{E2}$. The value of C, E1, and E2 should be found from the ISBSG data repository according to development type, platform, and programming language. The value of size is the calculated UFPs, and the maximum team size is the number of programmers who participate in a project. The selection of which method to use is up to the project manager based on the circumstances of the project.

Once the resource effort calculation is done, it must be converted into normal calendar months.

Export WBS to Microsoft Project 2010

The fourth, fifth, and sixth steps of Figure 2 involve exporting the created WBS to Microsoft Project 2010 and calculating the duration of the WBS level 2 by inserting durations of the WBS level 3 into the duration column. These steps are necessary due to the fact that since the sum of

counted UFPs from the WBS level 3 may not work properly for the size measurement of the WBS level 2 as mentioned earlier, it is not possible to directly calculate duration of the WBS level 2. That is, the duration of WBS level 2 definitely depends on the predecessor relationship between its work packages, so making the predecessor relationship of each work package should be conducted first through Microsoft Project 2010. Suppose that there are two modules as work packages in WBS level 3, and the duration of the first module and the second module is 1.5 months and 1 month, respectively. The duration of WBS level 2 can be 2.5 months or 1.5 months depending on the predecessor relationship of both modules; in other words, if both modules are on the critical path activities, the duration of WBS level 2 is 2.5 months. But if they are performed in parallel at the same time, the duration of WBS level 2 is 1.5 months because the first module has longer duration than the second module.

One final process is required before the duration estimate is finalized. Just as was done in the second step for *size*, a 3-point estimate should be used here for *duration*. To get the optimistic, most likely, and pessimistic duration for each work package, the UFP values from the second step that are associated with 25%, mean, and 75% probabilities, respectively, should be used. The Monte Carlo technique is also required to produce a normal probability distribution for the durations.

Comments on Applying the Concepts of the Sampling Distribution of the Mean and the Central Limit Theorem

It is important to note that the estimate of the final schedule uses two fundamental concepts: sampling distribution of the mean and the Central Limit Theorem (CLT). First of all, the key idea of the sampling distribution of the mean is that it has a mean μ and a standard deviation б/ $\sqrt{N}$ (N = sample size) if a population is given with mean μ and standard deviation б (Lane, 2007). Accordingly, the spread of the sampling distribution of the mean becomes narrower as long as the sample size increases (Lane, 2007). The main idea of the CLT comes from the concept of the sampling distribution of the mean. If the random samples are X1, X2, ⋯, Xn (n=sample size) with mean μ and variance $б^2$, the sample mean is:

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

(Thomas & Luk, 2008, Kim & Ra, 2011)

This means that as long as the sample size increases, the sampling distribution of the sample mean from random samples forms an approximate normal distribution no matter what the shape of the original distribution (Smith & Wells, 2006, Lane, 2007, Kim & Ra, 2011). Figures 6 and 7 graphically portray the idea of these two concepts.

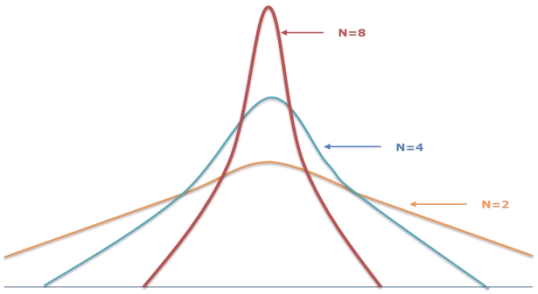

Figure 6 - The concept of the sampling distribution of the mean

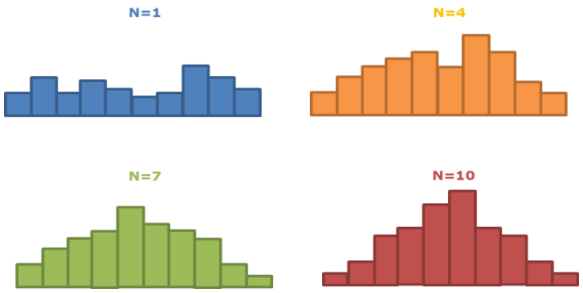

Figure 7 - The concept of the CLT

One important fact is that a sample size of more than 30 is required to generate the normal sampling distribution (Smith & Wells, 2006, Kim & Ra, 2011); based on this, we relate the number of work packages to the sample size. For this reason, the number of work packages should be more than 30 if a software project manager wants to apply the CLT to the project WBS.

To summarize this section, the idea is to have the WBS represent a detailed enough decomposition so that the number of work packages is greater than 30. This in turn generates a narrower normal distribution due to the concepts of the sampling distribution of the

mean and the CLT. Thus the estimation accuracy will be improved.

4. SIMULATION

In this section, we will use an inventory management software development project as an example to demonstrate the methodology; the demonstration will be explained step by step. This software is intended for large business, and it addresses the issue of management and tracking inventory. There are seven assumptions in this scenario. The first assumption is that all weight factors in the Unadjusted Function Point Calculation are "simple" (versus "average" or "complex"; See Table A1 in Appendix 1). The second assumption is that 3 programmers participate as a single team in the project and they perform the project with same project delivery rate (PDR). The third assumption is that project team members do not have historical records and experience of inventory system development so that they could compare an already developed inventory system which is currently available in the market to find out the number of EIs, EOs, EQs, ILFs, and EIFs. The fourth assumption is that the given timeframe is 30 months. The fifth assumption is that this project is based on multiplatform and uses the Java programming language. The sixth assumption is that all work packages are critical path activities, and the last assumption is to use 75% UFP value for calculating effort and duration.

The first step in this example is to transform the high level requirements into a deliverables-based WBS; the WBS of this project is presented in Figure 8 (See Figure B2 in Appendix 2 for the enlarged version). Note that since 30 work packages are placed in WBS level 3 it enables us to use the concept of the CLT, and further decomposition is not required.

The second step is to estimate the size of the lowest WBS level (WBS level 3) using FPA and the probabilistic technique. For each work package, the number of EIs, EOs, EQs, ILFs, and EIFs is counted in terms of a 3-point estimation, and then the *expected count* is calculated (see Table A1 in Appendix 1) (Pressman, 2009). Next, using the "simple" weighting factor (as per the first assumption above), the *sub total* for each function point is determined; then all the subtotals are summed to arrive at the *grand total* for each work package – this gives the expected size for each work package. The

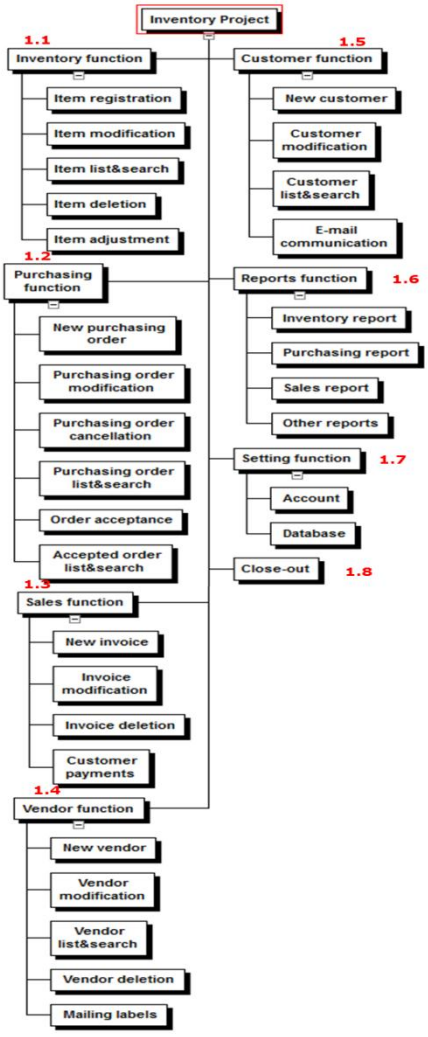

Figure 8 – Inventory Project WBS

@RISK tool is then used to invoke the Monte Carlo technique by picking 1000 random numbers from the triangular distribution range (that is, between the pessimistic and optimistic values of the range), which produces a normal distribution. The value determined at the 75% probability range (see the "75% UFP" column in Table A4, Appendix 1) is used for the work package UFP.

The third step is to calculate the resource effort based on the ISBSG regression equations (see Table A2 in Appendix 1). Per Table A3 in Appendix 1, the fixed value of 8.1 is used as the PDR (based on a Java platform and the 75% column). Then, for each work package, PDR is multiplied by the 75% UFP value calculated for

the work package in step 2 – this produces effort in terms of hours. This value is converted into months per the "if team size is known" equation in Table A2 of Appendix 1 (the team size is 3 per the second assumption). The resulting durations for each work package are shown in the "Duration" column of Table A4 in Appendix 1. Note that these are the expected durations based on the pessimistic (75% value) UFP.
The fourth and fifth steps are to export the WBS to Project 2010 and to form the predecessor relationships between each work package. The calculated durations based on the three programmers are simply put in the WBS level 3 duration column.

As shown in Figure B3 in Appendix 2, the duration of WBS level 2 and the entire project (the sixth step) is calculated automatically based on the duration and the predecessor relationship of WBS level 3. The calculated duration of the entire project is 28 months.

However, as implied by the comment at the end of step 3, this total project duration is based on a single-point duration estimate for each work package using the pessimistic UFP value. To complete the sixth step, a 3-point estimate in conjunction with the Monte Carlo technique should be used to generate a normal probability distribution for the duration estimate (as was done with the *size* calculations in step 2).

To accomplish this, the 25%, mean, and 75% values for UFP from Table A4 in Appendix 1 are used to calculate optimistic, most likely, and pessimistic duration values, respectively, for each work package. The @RISK tool is then used again to provide a Monte Carlo simulation by picking 1000 random numbers in the optimistic to pessimistic range; this produces normal distributions for the work package durations, which can then be summed to produce a probabilistic estimate of the overall project duration. The final schedule estimate using WBS level 2 (a sample size of 7 work packages) is shown in Table 2; the final estimate at WBS level 3 (using all 30 work packages) is shown in Table 3. Note that the expected total duration reflected in both tables is less than the 28 months that was calculated based on the single-point estimate.

As an example of how to use the information in Tables 2 and 3, suppose a customer wants to know the chance of completing this project within 26.13 months. The project manager can see that the answer is either 40% (Table 3) or 45% (Table 2). Our claim is that the estimate information in Table 3 (based on more work packages) is more accurate. Indeed, the values in Table 2 may be suspect since the sample size is less than 30, thereby not ensuring a normal distribution due to the requirements of the CLT.

It is important to observe that the range of schedule estimation becomes narrower by applying the CLT in terms of inferential statistics. This is shown graphically in Figure 9. The taller, narrower schedule estimation range is produced when the 30 work packages (samples) from WBS level 3 are used, whereas the shorter, wider range is based on using only the 7 work packages reflected at WBS level 2.

Table 2 - Final schedule estimate of WBS level 2

Simulation Results		**Probability Range**	
Min Dur.	25.33	10%	25.81
Mean Dur.	26.16	15%	25.87
Max Dur.	26.94	20%	25.93
N/A		25%	25.99
		30%	26.02
		35%	26.06
		40%	26.10
		45%	26.13
		50%	26.17
		55%	26.20
		60%	26.24
		65%	26.27
		70%	26.30
		75%	26.33
		80%	26.40
		85%	26.43
		90%	26.50
		95%	26.61
		100%	26.94

Table 3 - Final schedule estimate of WBS level 3

The effectiveness of this framework can also be viewed as a method to assist with monitoring and controlling overall project performance. For instance, projects are usually seen as being constrained by the three main factors of scope, schedule, and cost (Gido & Clements, 2009); it is obvious that a change in one of these will impact the other two. Since our approach focuses on developing a more accurate schedule, the chance for budget or scope changes caused by schedule problems is significantly reduced.

Table 2 - Final schedule estimate of WBS level 2

Simulation Results		Probability Range	
Min Dur.	25.33	10%	25.81
Mean Dur.	26.16	15%	25.87
Max Dur.	26.94	20%	25.93
N/A		25%	25.99
		30%	26.02
		35%	26.06
		40%	26.10
		45%	26.13
		50%	26.17
		55%	26.20
		60%	26.24
		65%	26.27
		70%	26.30
		75%	26.33
		80%	26.40
		85%	26.43
		90%	26.50
		95%	26.61
		100%	26.94

Table 3 - Final schedule estimate of WBS level 3

Simulation Results		Probability Range	
Min Dur.	25.74	10%	25.98
Mean Dur.	26.16	15%	26.01
Max Dur.	26.6	20%	26.04
N/A		25%	26.06
		30%	26.09
		35%	26.11
		40%	26.13
		45%	26.14
		50%	26.16
		55%	26.18
		60%	26.20
		65%	26.22
		70%	26.24
		75%	26.27
		80%	26.28
		85%	26.31
		90%	26.34
		95%	26.38
		100%	26.60

5. CONCLUSION AND FUTURE WORK

The goal of this research was to develop a methodology to provide more accurate software project schedule estimates early in the project life cycle (perhaps in the proposal preparation stage).

This is a difficult task due to the uncertainty that exists in the early stages. A major contribution of this paper is the development of a framework for such estimates, and the demonstration that it is possible to generate greater probability accuracy by making the range of schedule

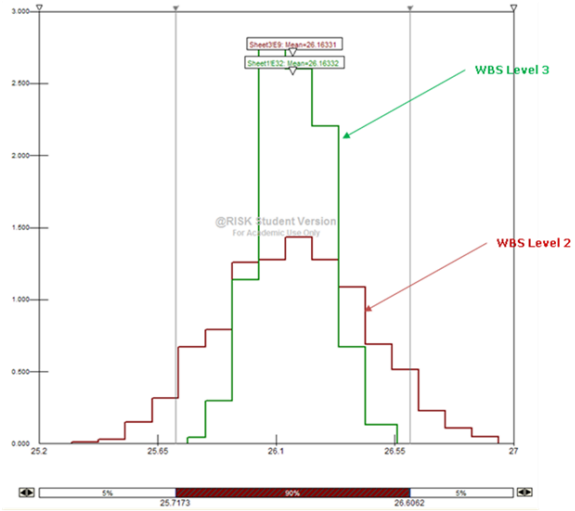

Figure 9 - Comparison analysis of the range of schedule estimation between WBS level 2 and level 3

estimation narrower using the CLT concept. We believe that probabilistic schedule estimation based on inferential statistics is able to assist in reducing project risks by taking into account the uncertainty associated with project schedules, especially in the early project stages.

Another contribution of this paper is that our framework is applicable to not only IT projects, but also projects of other types. While the methods used to estimate size and effort may vary with non-software development projects, the overall methodology can still be utilized. Further, the proposed method is not only useful for the initial schedule generation, but is equally applicable in any required re-estimation efforts to adjust the schedule in later stages of the project life cycle.

The use of the proposed estimation methodology was demonstrated on an example project. However, to more accurately examine the efficacy of the framework and to verify that it can be used to produce more accurate estimations, the methodology must be applied to real-world case studies. This will be an important next step.

Additionally, there are certainly improvements that should be considered for future work. First, redundancy in the work packages or modules must be taken into account while creating the project WBS. Further, schedule estimation should be conducted in the planning stage using actual design documentation when possible;

even further, size and resource requirements of previously performed similar efforts should be able to be used when available. Calculating Adjusted Function Points (AFPs) with the Value Adjustment Factor (VAF) should be also paired with schedule estimation in the planning stage.

6. REFERENCES

Basha, S., & P, D. (2010). Analysis of Empirical Software Effort Estimation Medels. *International Journal of Computer Science and Information Security Vol. 7 No. 3*, 68-77.

Bottenfield, R. (2005, May 14). *Project Effort and Schedule Estimation Modeling Applied to Software Localization.* Retrieved from MSSE 2005 Plan B Project:http://www.msse.umn.edu/system/files/capstone_project_files/Project+Effort+and+Schedule+Estimation+Modeling+Applied+to+Software+Localization.pdf

Cherjee, V. B., Mahanti, P. K., & Jumar, S. (31st Jul 2009). Complexity Metric for Analogy Based Effort Estimation. *Journal of Theoretical and Applied Information Technology Vol. 6 No. 1*, 1-8.

Chiu, N.-H., & Huang, S.-J. (2007). The adjusted analogy-based software effort estimation based on similarity distances. *The Journal of Systems and Software Volume 80 Issue 4*, 628-640.

Cuadrado-Gallego, J. J., Rodríguez-Soria, P., & Hakimuddin, S. (2010). Early Functional Size Estimation with IFPUG Unit Modified. *9th IEEE/ACIS International Conference on Computer and Information Science* (pp. 729-733). IEEE.

Dobson, M. S., & Leemann, T. (2010). Creative Project Management: Innovative Project Options to Solve Problems on Time and under Budget. McGraw-Hill, New York.

Douglas, J., & Ra, J. (2010). State Government Virtual Project & PMO Collaboration Guided Discussion. *Washington DC: PMI Global Congress.*

Emam, K. E., & Koru, A. G. (2008). A Replicated Survey of IT Software Project Failures. *IEEE*, (pp. 84-90).

Gido, J. & Clements, J. P. (2009). *Effective Project Management, 4th Edition.* Mason: SOUTH-WESTERN CENGAGE Learning.

Hughes, B. & Cotterell (2009). Software Project Management, 5th Edition. McGraw-Hill, London.

ISBSG. (2010). Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort & Duration. McGraw-Hill, New York.

Kim, K.-P., & Ra, J. (2011). Applying Central Limit Theorem to Project Cost Estimation. *Project Management Review*, (pp. 29-35).

Kocaguneli, E., Menzies, T., Bener, A. B., & Keung, J. W. (March / April 2012). Exploiting the Essential Assumptions of Analogy-Based Effort Estimation. *IEEE Transactions on Software Engineering Vol. 38 No. 2* (pp. 425-438). IEEE.

Lane, M., David (2007). *HyperStat Online Statistics Textbook.* Retrieved from http://davidmlane.com/hyperstat/sampling_dist.html

Lind, K., & Heldal, R. (2010). On the Relationship between Functional Size and Software Code Size. *Proceeding of the 2010 ICSE Workshop on Emerging Trends in Software Metrics* (pp. 47-52). ACM.

Liu, X., Yang, Y., Chen, J., Wang, Q., & Li, M. (2009). Achieving On-Time Delivery: A Two-Stage Probabilistic Scheduling Strategy for Software Projects. *ICSP*, (pp. 317-329).

Malik, A. A. (2010). Quantitative and Qualitative Analyses of Requirements Elaboration for Early Software Size Estimation. 7-13.

Nasir, M. (2006). A Survey of Software Estimation Techniques and Project Planning Practices. *Proceeding of the Seventh ACIS International Conference.* SNPD'06.

Nassif, A. B., Capretz, L. F., & Ho, D. (2010). Enhancing Use Case Points Estimation Method Using Soft Computing Techniques. *Journal of Global Research in Computer Science(JGRCS) Volume 1 No. 4*, 12-21.

nee'Singhal, N. B., & Srikrishna, C. V. (2008). A Case Study to Assess the Validity of Function Points. *World Academy of Science, Engineering and Technology*, (pp. 224-227).

Ogwueleka, N. F. (2012). Requirement elicitation problems in software development - A case study of a GSM service provider. *Indian Journal of Innovations and Development Vol. 1, No. 8,* 599-605

Pressman, R. S. (2009). Software Engineering: A Practitioner's Approach 7th Edition. McGraw-Hill, New York

Project Management Institute. (2008). A Guide to the Project Management Body of Knowledge (PMBOK Guide 4th Edition) (pp. 17, 119-120, 275-276). PMI, Newtown Square, PA

Project Management Solutions, Inc. (2012, March 26). *Strategies for Project Recovery: A PM Solutions Research Report.* Retrieved from http://www.pmsolutions.com/collateral/research/Strategies%20for%20Project%20Recovery%202011.pdf

Smith, Z. R., & Wells, C. S. (2006, October 18-20). *Central Limit Theorem and Sample Size.* Retrieved from The annual meeting of the Northeastern Educational Research Association: http://www.umass.edu/remp/Papers/Smith&Wells_NERA06.pdf

Stevens, M. I., Hogendoorn, K., & Schwarz, P. M. (2012, May 8). *Evolution of sociality by natural selection on variances in reproductive fitness: evidence from a social bee.* Retrieved from BMC Evolutionary Biology: http://www.biomedcentral.com/content/pdf/1471-2148-7-153.pdf

Tesch, D., Kloppenborg, T. J., & Frolick, M. N. (Summer 2007). IT Project Risk Factors: The Project Management Professionals Perspective. *Journal of Computer Information Systems*, 61-69.

Thomas, D. B., & Luk, W. (2008). Estimation of sample mean and variance for Monte-Carlo simulations. *ICECE Technology 2008* (pp. 89-96). Taipei: IEEE.

Appendices

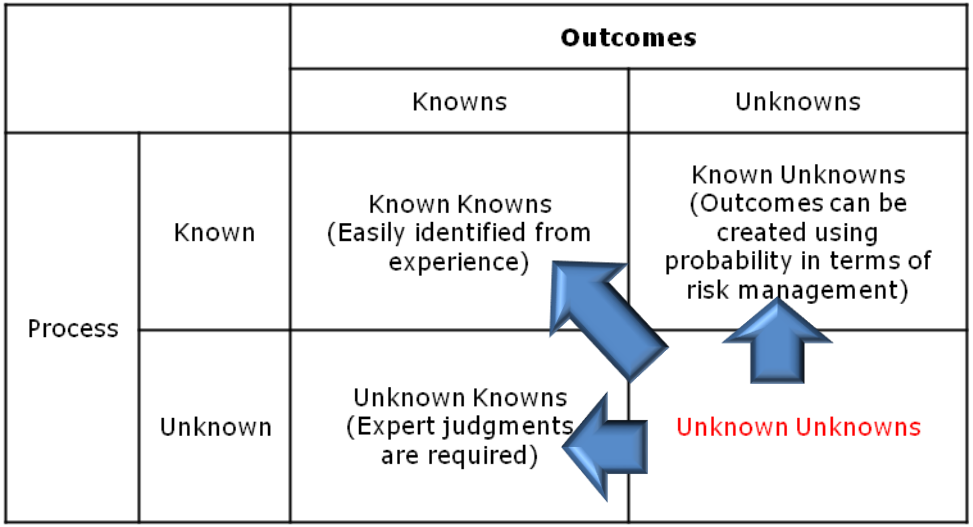

Figure 1 - Knowns and unknowns quadrant form

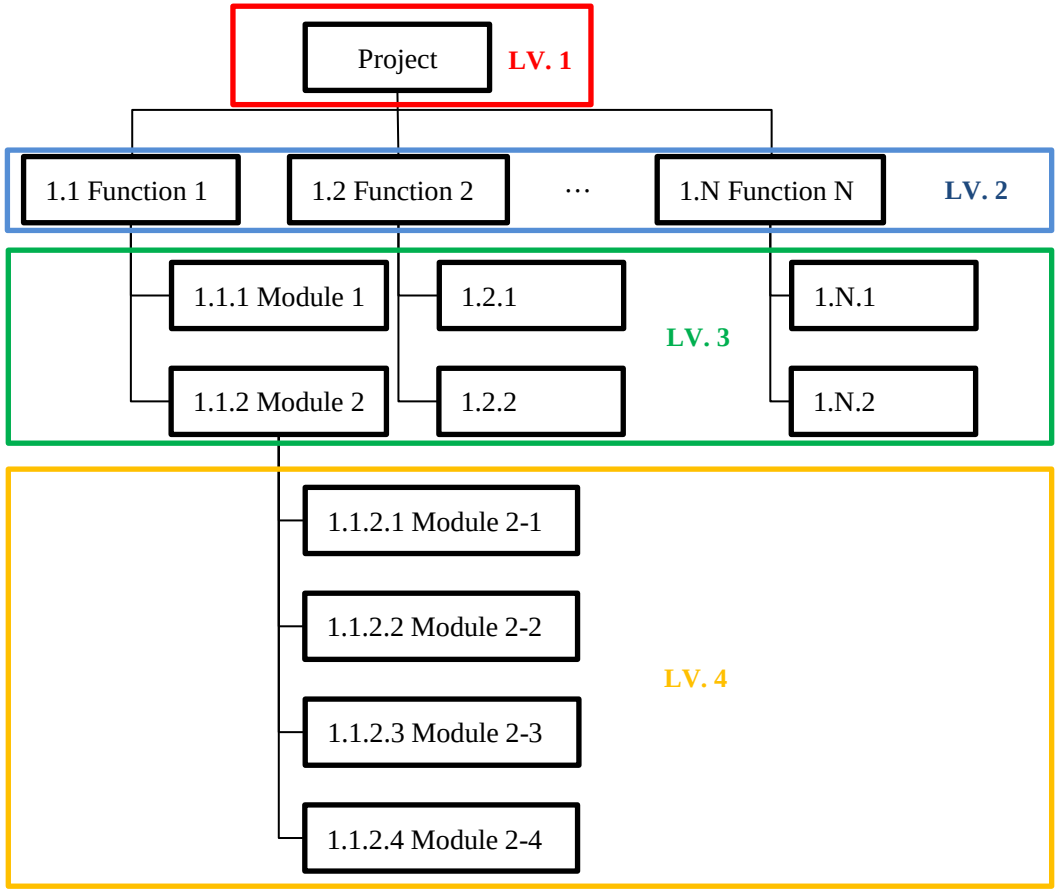

Figure 3 - Deliverables-based WBS

Appendix 1

Table A1 - Unadjusted Function Point Calculation (Pressman, 2009)

Category	3-Point Estimation			Expected Count	Weight Factors			Sub Total
	OP	ML.	Pess.		Simple	Average	Complex	
EI	No.	No.	No.	{1*(Pess.)+4*(ML.)+1*(OP.)} / 6	3	4	6	(Expected Count) * (One of Weight Factors)
EO	No.	No.	No.	Same as above	4	5	7	Same as above
EQ	No.	No.	No.	Same as above	3	4	6	Same as above
ILF	No.	No.	No.	Same as above	7	10	15	Same as above
EIF	No.	No.	No.	Same as above	5	7	10	Same as above
Grand Total (UFP)								Sum of sub total

Table A2 – ISBSG Regression Equations (ISBSG, 2010)

Category		Equations	Values
Project Delivery Rate (PDR)		$C*Size^{E1}*Maximum\ Team\ Size^{E2,}$ OR the fixed value of PDR.	Number of hours per FP
Effort		PDR*UFP	Hours
Duration	If team size is known	(Effort / team size) / Number of hours worked per person per month	Calendar months
	If team size is unknown	$0.370*Effort^{0.328}$	

Table A3 – Project Delivery Rate based on Programming Language and Multi-Platform (ISBSG 2010)

	Minimum	10%	25%	50%	75%	90%	Maximum	Mean
ABAP	4.2	6.5	7.8	9.6	14.6	20.3	34.3	12.1
C	1.8	1.9	2.2	3.9	10.1	13	31.3	7.7
COBOL	3.4	4.7	8.3	20.3	37.8	43.2	49.1	22.8
C#	1.9	5.7	8	13.7	22.8	32.2	48.8	16.7
Java	3.1	5	5.7	6.4	8.1	11.8	17.1	7.4
Lotus Notes	1.5	1.9	2.9	3.7	5.1	7.8	11.9	4.5
PL/1	8	12.5	15.6	20.8	26.8	46.8	61.8	24.9
PL/SQL	0.8	1.4	1.7	4.2	6.7	10.7	14.3	5.1
Visual Basic	0.9	2.5	4.2	8.6	18.6	36.8	60.9	14.1
3rd Generation Language	4.8	7.8	10.9	17.3	22.7	30	38	17.8
4th Generation Language	3.6	6	7.8	8.7	12.5	19.2	35.7	11.3
5th Generation Language	6.5	8.8	11.5	17.2	22	25.1	31.8	17.4
Other	1.1	3.1	4.8	7.4	10.4	14.8	27.4	8.7

Table A4 - Simulation Results for Size Estimation and Pessimistic Duration of WBS Level 3

WP	Min UFP	25% UFP	Mean UFP	75% UFP	Max UFP	Effort (Hours)	Duration (Month)
1.1.1	40.07	55.33	59	62.46	73.42	506	1.0
1.1.2	47.76	61.39	65	68.79	83.15	557	1.1
1.1.3	31.47	45.96	50	54.07	67.12	438	0.8
1.1.4	31.47	45.96	50	54.07	67.12	438	0.8
1.1.5	28.55	40.90	44	47.10	57.07	382	0.7
1.2.1	50.5	61.87	65	68	77.75	551	1.0
1.2.2	57.33	70.14	74	77.68	91.83	629	1.2
1.2.3	37.14	44.38	47	49.53	58.11	401	0.8
1.2.4	22.76	31.57	34	36.45	46.63	295	0.6
1.2.5	22.76	31.57	34	36.45	46.63	295	0.6
1.2.6	22.76	31.57	34	36.45	46.63	295	0.6
1.3.1	93.95	110.94	115.67	120.37	136.72	975	1.8
1.3.2	64.82	83.03	89	94.66	112.12	767	1.5
1.3.3	22.76	31.57	34	36.45	46.63	295	0.6
1.3.4	54.55	63.39	66	68.49	76.15	555	1.1
1.4.1	81.7	95.61	100	104.57	119.22	847	1.6
1.4.2	62.83	81.22	88	94.42	113.96	765	1.4
1.4.3	23	35.81	40	44.19	56.84	358	0.7
1.4.4	20.12	28.42	31	33.66	42.31	273	0.5
1.4.5	22.81	35.88	40	43.97	58.09	356	0.7
1.5.1	80.71	90.90	94	97.13	106.94	787	1.5
1.5.2	84.69	98.21	102	105.53	115.94	855	1.6
1.5.3	24.62	37.68	43	48.04	65.36	389	0.7
1.5.4	26.19	37.04	42	46.64	61.47	378	0.7
1.6.1	23	35.81	40	44.19	56.84	358	0.7
1.6.2	22.49	31.30	34	36.74	45.03	298	0.6
1.6.3	35.83	43.42	46	48.49	55.75	393	0.7
1.6.4	20.12	28.42	31	33.66	42.31	273	0.5
1.7.1	54.51	73.10	80	86.95	107.64	704	1.3
1.7.2	22.49	31.30	34	36.74	45.03	298	0.6

Appendix 2

Figure B1 – Overall research model

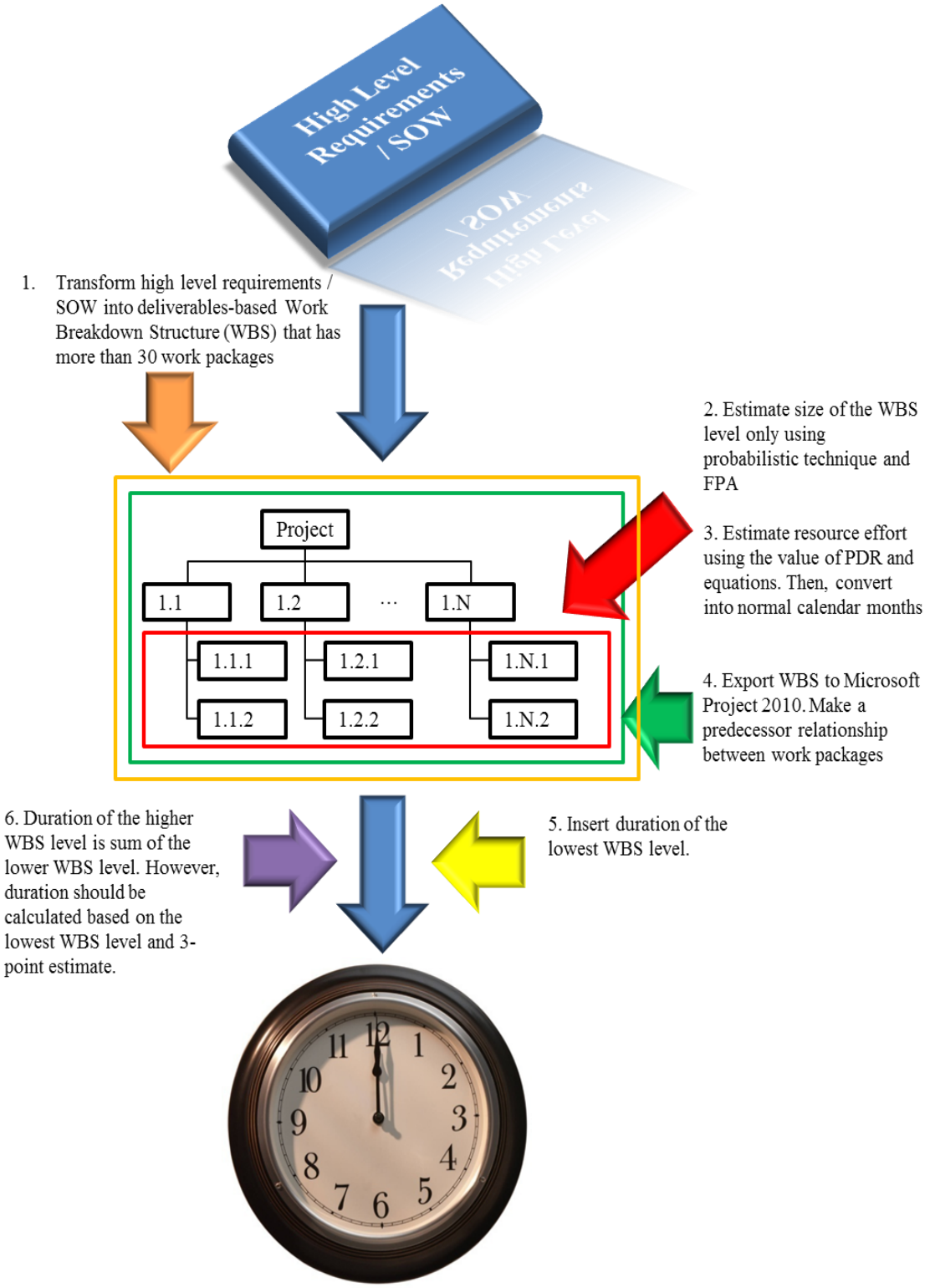
High Level Requirements / SOW
1. Transform high level requirements / SOW into deliverables-based Work Breakdown Structure (WBS) that has more than 30 work packages
2. Estimate size of the WBS level only using probabilistic technique and FPA
3. Estimate resource effort using the value of PDR and equations. Then, convert into normal calendar months
4. Export WBS to Microsoft Project 2010. Make a predecessor relationship between work packages
5. Insert duration of the lowest WBS level.
6. Duration of the higher WBS level is sum of the lower WBS level. However, duration should be calculated based on the lowest WBS level and 3-point estimate.
Project
1.1
1.2
…
1.N
1.1.1
1.1.2
1.2.1
1.2.2
1.N.1
1.N.2

Figure B2 – Inventory project WBS

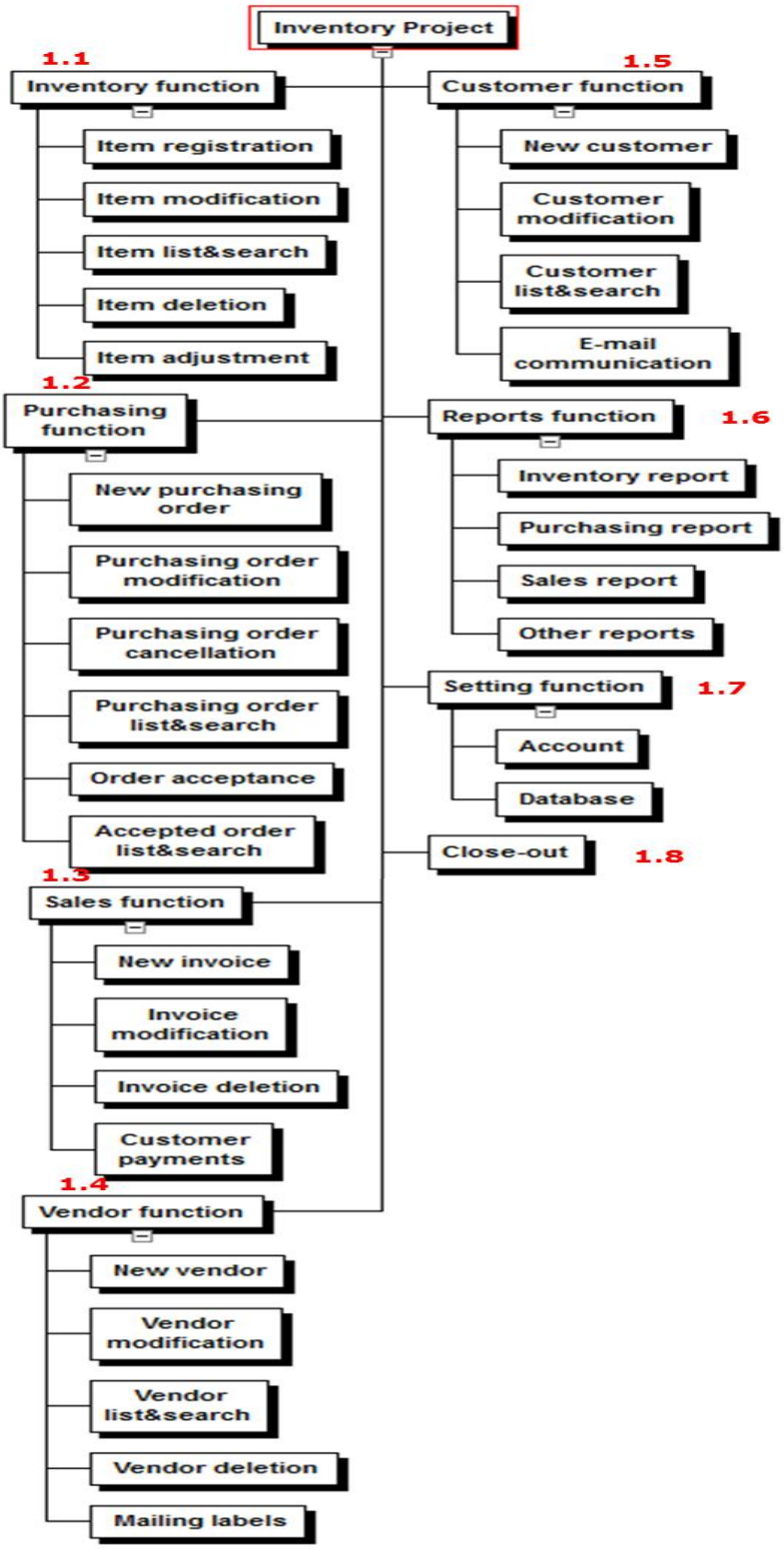

Figure B3 – Critical activities and duration of WBS level two & entire project

Task Name	Duration	Start	Finish	Predecessors	Resource Names
Inventory Project	**28 mons**	**2012-05-01**	**2014-06-23**		
Inventory function	**4.4 mons**	**2012-05-01**	**2012-08-30**		
Item registration	1 mon	2012-05-01	2012-05-28		John,Mike,Sam
Item modification	1.1 mons	2012-05-29	2012-06-27	3	John,Mike,Sam
Item list&search	0.8 mons	2012-06-28	2012-07-19	4	John,Mike,Sam
Item deletion	0.8 mons	2012-07-20	2012-08-10	5	John,Mike,Sam
Item adjustment	0.7 mons	2012-08-13	2012-08-30	6	John,Mike,Sam
Purchasing function	**4.8 mons**	**2012-08-31**	**2013-01-11**		
New purchasing order	1 mon	2012-08-31	2012-09-27	7	John,Mike,Sam
Purchasing order modification	1.2 mons	2012-09-28	2012-10-31	9	John,Mike,Sam
Purchasing order cancellation	0.8 mons	2012-11-01	2012-11-22	10	John,Mike,Sam
Purchasing order list&search	0.6 mons	2012-11-23	2012-12-10	11	John,Mike,Sam
Order acceptance	0.6 mons	2012-12-11	2012-12-26	12	John,Mike,Sam
Accepted order list&search	0.6 mons	2012-12-27	2013-01-11	13	John,Mike,Sam
Sales function	**5 mons**	**2013-01-14**	**2013-05-31**		
New invoice	1.8 mons	2013-01-14	2013-03-04	14	John,Mike,Sam
Invoice modification	1.5 mons	2013-03-05	2013-04-15	16	John,Mike,Sam
Invoice deletion	0.6 mons	2013-04-16	2013-05-01	17	John,Mike,Sam
Customer payments	1.1 mons	2013-05-02	2013-05-31	18	John,Mike,Sam
Vendor function	**4.9 mons**	**2013-06-03**	**2013-10-16**		
New vendor	1.6 mons	2013-06-03	2013-07-16	19	John,Mike,Sam
Vendor modification	1.4 mons	2013-07-17	2013-08-23	21	John,Mike,Sam
Vendor list&search	0.7 mons	2013-08-26	2013-09-12	22	John,Mike,Sam
Vendor deletion	0.5 mons	2013-09-13	2013-09-26	23	John,Mike,Sam
Mailing labels	0.7 mons	2013-09-27	2013-10-16	24	John,Mike,Sam
Customer function	**4.5 mons**	**2013-10-17**	**2014-02-19**		
New customer	1.5 mons	2013-10-17	2013-11-27	25	John,Mike,Sam
Customer modification	1.6 mons	2013-11-28	2014-01-10	27	John,Mike,Sam
Customer list&search	0.7 mons	2014-01-13	2014-01-30	28	John,Mike,Sam
E-mail communication	0.7 mons	2014-01-31	2014-02-19	29	John,Mike,Sam
Reports function	**2.5 mons**	**2014-02-20**	**2014-04-30**		
Inventory report	0.7 mons	2014-02-20	2014-03-11	30	John,Mike,Sam
Purchasing report	0.6 mons	2014-03-12	2014-03-27	32	John,Mike,Sam
Sales report	0.7 mons	2014-03-28	2014-04-16	33	John,Mike,Sam
Other reports	0.5 mons	2014-04-17	2014-04-30	34	John,Mike,Sam
Setting function	**1.9 mons**	**2014-05-01**	**2014-06-23**		
Account	1.3 mons	2014-05-01	2014-06-05	35	John,Mike,Sam
Database	0.6 mons	2014-06-06	2014-06-23	37	John,Mike,Sam
Close-out	0 mons	2014-06-23	2014-06-23	38	John,Mike,Sam