

**TOWSON UNIVERSITY  
OFFICE OF GRADUATE STUDIES**

**OBJECTIVE DECISION SUPPORT TOOLS FOR IT PROJECT MANAGERS**

by

Donghwoon Kwon

A Dissertation

Presented to the faculty of

Towson University

in partial fulfillment

of the requirements for the degree

Doctor of Science

Department of Computer and Information Sciences

Towson University  
Towson, Maryland 21252

May 2015

**TOWSON UNIVERSITY  
OFFICE OF GRADUATE STUDIES  
DISSERTATION APPROVAL PAGE**

This is to certify that the dissertation prepared by Donghwoon Kwon entitled Objective Decision Support Tools for IT Project Managers has been approved by the dissertation committee as satisfactorily completing the dissertation requirements for the degree Doctor of Science.

  
Chairperson, Dissertation Committee: Robert J. Hammel II

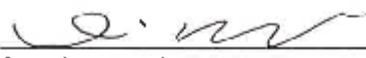
4/30/2015  
Date

  
Committee Member: Josh Jchlinger

4/30/2015  
Date

  
Committee Member: Scott Hillberg

4/30/2015  
Date

  
Committee Member: Wei Yu

4/30/2015  
Date

  
Dean of Graduate Studies

4-30-15  
Date

© 2015 By Donghwoon Kwon

All Rights Reserved

## Abstract

### Objective Decision Support Tools for IT Project Managers

Donghwoon Kwon

The definition of a project is a temporary endeavor undertaken to create a unique product, service, or result. In terms of the Information Technology industry, projects are generally aimed at developing or acquiring new or modified information systems hardware or software, or both. The literature claims that less than half of all IT projects are completed successfully.

The Project Management Body of Knowledge (PMBOK) published by the Project Management Institute indicates that the “planning” process group accounts for approximately 48% of all project management processes. The literature reveals that a major reason that IT projects fail or are cancelled is that they frequently go over schedule. Thus, schedule estimation is vital to the planning process; but it is well known that project uncertainty is typically very high in the early stages of a project, which is where accurate planning is particularly crucial. Additionally, analysis of the PMBOK shows that the highly subjective “expert judgment” technique is relied on much more than other tools and techniques, perhaps adding to the uncertainty and inaccuracy in the early project stages.

This research focuses on proposing objective tools and techniques to increase certainty and accuracy in the early project stages, concentrating on the primary research question “Can project determination and time / schedule estimation be improved by using objective tools and techniques?”. This question is answered by

attacking three sub-questions in three separate steps. In the first step, a framework is proposed for estimating an objective project schedule in the *proposal preparation* stage, and its effectiveness is demonstrated on an example project scenario. Function point analysis, probability, and project management techniques are used to reduce project risk by taking into account the uncertainty associated with project schedule estimation in this very early project stage.

The second step uses the above framework to address the issue of refining / verifying the previous schedule when the project reaches the *planning* stage. With the availability of design documents at this stage, additional detail is available to allow the function point analysis to be done with the more accurate Adjusted Function Point (AFP) count as opposed to the Unadjusted Function Point (UFP) used in the first step. The example project scenario from the first step is carried forward and extended to illustrate the usage of the framework in this step.

The final step develops a methodology using a decision tree-based framework to provide an objective technique for the early-stage comparison of software development project types. The process utilizes cost estimation based on the probabilistic project schedule from the previous newly-developed framework, expected monetary value, possibilistic success rate, and the decision tree approach to introduce more objectivity into deciding, during the early project stages, which software development type should be used. The methodology is illustrated by example, and one notable result is that the software development type selected using the project management viewpoint (relying on expected monetary value) may not be

the one that would be selected from the typical business management point of view (governed by net present value).

## Table of Contents

List of Tables .....	viii
List of Figures .....	x
Chapter 1 Introduction .....	1
1. 1 Outline of the Dissertation .....	4
Chapter 2 Background .....	7
2.1 Problem Identification .....	7
2.2 Research Objectives .....	13
2.3 Research Methodology .....	14
Chapter 3 Literature Review .....	16
3.1 Schedule Estimation Processes in the Early Stage (Proposal Preparation Stage) .....	16
3.2 Schedule Refinement or Verification Processes .....	34
3.3 Comparison Process for Determining a Software Project Type .....	38
Chapter 4 Research Methodology Demonstrations .....	44
4.1 Schedule Estimation in the Early Stage (Proposal Preparation Stage) .....	46
4.2 Schedule Refinement or Verification in the Planning Stage .....	83
4.3 Early-Stage Comparison of Software Development Project Types .....	122
Chapter 5 Conclusion, Limitations, and Improvements .....	136
Bibliography .....	139
Curriculum Vita .....	142

## List of Tables

Table 1 Knowledge Areas and Processes of using the Expert Judgment Technique .....	8
Table 2 Equation of Unadjusted Function Point Calculation .....	24
Table 3 Example Number of Functional Component Types .....	26
Table 4 The value of C, E1, and E2 in the PDR Equation (ISBSG, 2010).....	28
Table 5 Project Delivery Rate by Language Type – All Platforms (ISBSG, 2010) .....	28
Table 6 Project Delivery Rate by Language Type – Mainframe Platforms (ISBSG, 2010) .....	29
Table 7 Project Delivery Rate by Language Type – Midrange Platforms (ISBSG, 2010) .....	29
Table 8 Project Delivery Rate by Language Type – PC Platforms (ISBSG, 2010) .....	29
Table 9 Project Delivery Rate by Language Type – Multi Platforms (ISBSG, 2010) .....	29
Table 10 Project Delivery Rate by Language – All Platforms (ISBSG, 2010) .....	30
Table 11 Project Delivery Rate by Language – Mainframe Platforms (ISBSG, 2010).....	30
Table 12 Project Delivery Rate by Language – Midrange Platforms (ISBSG, 2010).....	31
Table 13 Project Delivery Rate by Language – PC Platforms (ISBSG, 2010).....	31
Table 14 Project Delivery Rate by Language – Multi Platforms (ISBSG, 2010).....	31
Table 15 ISBSG Regression Equations (ISBSG, 2010) .....	32
Table 16 Project Cost Estimation .....	41
Table 17 Success Rate Criteria and Equation (Maglyas, 2009).....	42
Table 18 The Expected Size Estimation of the First Work Package .....	48
Table 19 The Summarized UFP for Size Estimation of Each Work Package .....	77
Table 20 The Summarized Pessimistic Duration of WBS Level 3.....	78
Table 21 Final Schedule Estimate of WBS Level 2 .....	81
Table 22 Final Schedule Estimate of WBS Level 3 .....	81
Table 23 The Counted EI Number of Each Work Package .....	89



Table 24 Final Simulation Results .....	121
Table 25 Revised Cost Estimation Equations .....	125
Table 26 Cost Estimation of New Development .....	130
Table 27 Success Rate of Each Decision Alternative .....	131
Table 28 NPV for New Development Strong Demand .....	132
Table 29 NPV for New Development Weak Demand .....	133
Table 30 NPV for Reuse-based w/o Modification Strong Demand .....	133
Table 31 NPV for Reuse-based w/o Modification Weak Demand .....	133
Table 32 NPV for Reuse-based w/ Modification Strong Demand .....	134
Table 33 NPV for Reuse-based w/ Modification Weak Demand .....	134

## List of Figures

Figure 1 Map Objective Technique to Predictive Life Cycles (PMI, 2013).....	9
Figure 2 Knowns and Unknowns Quadrant Form .....	11
Figure 3 Difference between Deliverables-based and Phase-based WBS.....	20
Figure 4 IFPUG Function Point Architecture .....	22
Figure 5 Triangular Distribution .....	26
Figure 6 The concept of the sampling distribution of the mean .....	33
Figure 7 The concept of the CLT.....	34
Figure 8 Research Model for Schedule Estimation in the Early Stage.....	44
Figure 9 Research Model for Schedule Refinement / Verification.....	45
Figure 10 Research Model for Early-Stage Comparison of Software Development Project Types .....	45
Figure 11 Inventory Project WBS.....	49
Figure 12 WBS 1.1.1 Item Registration UFP .....	50
Figure 13 WBS 1.1.2 Item Modification UFP .....	51
Figure 14 WBS 1.1.3 Item Search & List UFP and WBS 1.1.4 Item Deletion UFP.....	52
Figure 15 WBS 1.1.5 Item Adjustment UFP .....	53
Figure 16 WBS 1.2.1 New Purchasing Order UFP .....	54
Figure 17 WBS 1.2.2 Purchasing Order Modification UFP .....	55
Figure 18 WBS 1.2.3 Purchasing Order Cancellation UFP.....	56
Figure 19 WBS 1.2.4 Purchasing Order Search & List, WBS 1.2.5 Order Acceptance, and WBS 1.2.6 Accepted Order Search & List UFP .....	57
Figure 20 WBS 1.3.1 New Invoice UFP.....	58
Figure 21 WBS 1.3.2 Invoice Modification UFP .....	59
Figure 22 WBS 1.3.3 Invoice Deletion UFP .....	60
Figure 23 WBS 1.3.4 Invoice Search & List UFP .....	61

Figure 24 WBS 1.4.1 New Vendor UFP.....	62
Figure 25 WBS 1.4.2 Vendor Modification UFP .....	63
Figure 26 WBS 1.4.3 Vendor Search & List UFP .....	64
Figure 27 WBS 1.4.4 Vendor Deletion UFP .....	65
Figure 28 WBS 1.4.5 Mailing Labels UFP .....	66
Figure 29 WBS 1.5.1 New Customer UFP .....	67
Figure 30 WBS 1.5.2 Customer Modification UFP.....	68
Figure 31 WBS 1.5.3 Customer Search and List UFP.....	69
Figure 32 WBS 1.5.4 Customer Deletion UFP .....	70
Figure 33 WBS 1.6.1 Inventory Report UFP.....	71
Figure 34 WBS 1.6.2 Purchasing Report UFP .....	72
Figure 35 WBS 1.6.3 Sales Report UFP .....	73
Figure 36 WBS 1.6.4 Other Reports UFP.....	74
Figure 37 WBS 1.7.1 Account UFP.....	75
Figure 38 WBS 1.7.2 Database UFP .....	76
Figure 39 Critical Activities and Duration of WBS Level 2 & Entire Project .....	79
Figure 40 Comparison Analysis of the Range of Schedule Estimation between WBS Level 2 and Level 3.....	82
Figure 41 Class Diagram for Item and Purchasing Order Functionality .....	85
Figure 42 Class Diagram for Vendor and Customer Functionality .....	85
Figure 43 Class Diagram for Sales, Setting, and Report Functionality .....	86
Figure 44 Database Relationship between Each Work Package .....	87
Figure 45 WBS 1.1.1 Item Registration Sequence Diagram .....	90
Figure 46 WBS 1.1.2 Item Modification Sequence Diagram .....	91
Figure 47 WBS 1.1.3 Item Search & List Sequence Diagram.....	92
Figure 48 WBS 1.1.4 Item Deletion Sequence Diagram .....	93

Figure 49 WBS 1.1.5 Item Adjustment Sequence Diagram .....	94
Figure 50 WBS 1.2.1 New Purchasing Order Sequence Diagram .....	95
Figure 51 WBS 1.2.2 Purchasing Order Modification Sequence Diagram .....	96
Figure 52 WBS 1.2.3 Purchasing Order Cancellation Sequence Diagram .....	97
Figure 53 WBS 1.2.4 Purchasing Order Search & List Sequence Diagram .....	98
Figure 54 WBS 1.2.5 Order Acceptance Sequence Diagram .....	99
Figure 55 WBS 1.2.6 Accepted Order Sequence Diagram .....	100
Figure 56 WBS 1.3.1 New Invoice Sequence Diagram .....	101
Figure 57 WBS 1.3.2 Invoice Modification Sequence Diagram .....	102
Figure 58 WBS 1.3.3 Invoice Deletion Sequence Diagram .....	103
Figure 59 WBS 1.3.4 Invoice Search & List Sequence Diagram .....	104
Figure 60 WBS 1.4.1 New Vendor Sequence Diagram .....	105
Figure 61 WBS 1.4.2 Vendor Modification Sequence Diagram .....	106
Figure 62 WBS 1.4.3 Vendor Search & List Sequence Diagram .....	107
Figure 63 WBS 1.4.4 Vendor Deletion Sequence Diagram .....	108
Figure 64 WBS 1.4.5 Mailing Label Sequence Diagram .....	109
Figure 65 WBS 1.5.1 New Customer Sequence Diagram .....	110
Figure 66 WBS 1.5.2 Customer Modification Sequence Diagram .....	111
Figure 67 WBS 1.5.3 Customer Search & List Sequence Diagram .....	112
Figure 68 WBS 1.5.4 Customer Deletion Sequence Diagram .....	113
Figure 69 WBS 1.6.1 Inventory Report Sequence Diagram .....	114
Figure 70 WBS 1.6.2 Purchasing Order Report Sequence Diagram .....	115
Figure 71 WBS 1.6.3 Sales Report Sequence Diagram .....	116
Figure 72 WBS 1.6.4 Other Reports Sequence Diagram .....	117
Figure 73 WBS 1.7.1 Account Setting Sequence Diagram .....	118

Figure 74 WBS 1.7.2 Database Setting Sequence Diagram ..... 119

Figure 75 Decision Tree for Each Decision Alternative..... 129

Figure 76 Final Decision Tree ..... 135

## **Chapter 1 Introduction**

Despite the fact that software development in the Information Technology (IT) field is related to project management, the main problem is that many project managers, programmers, and even stakeholders are not aware of it. According to “A Guide to the Project Management Body of Knowledge (PMBOK) 5th Edition”, abbreviated as the PMBOK guide 5th Edition, the definition of project management is to apply knowledge, skills, tools and techniques to project activities to meet the project requirements (Project Management Institute, 2013). It is no doubt that project management is very important to complete a project successfully, but the problem is that many projects suffer from mismanagement so that they are cancelled in the middle of completing a project. For example, the software project success rate was only 32% in 2009. The research also shows that only 47% of IT projects are completed successfully and 37% of IT projects are repaired or cancelled (Project Management Solutions, 2012). There are various reasons regarding this issue including schedule overrun, over budget, too many requirements and scope change, and so on (Emam & Koru, 2008). One important fact is that these reasons are related to triple constraints which consist of scope, time, and cost. The importance of triple constraints is that since it is almost impossible to change only one constraint without affecting the other ones, triple constraints management is one of the key factors that lead to successful project completion. Project quality is also affected by the balance of triple constraints (Phillips, 2009). So problems result from a different view that coding, not project management, is the most important part of software development. Thus, stakeholders think nothing of changing project scope, time, and costs even though these directly affect success or failure of a project.

The success criteria for IT projects is not only based on the quality of an implemented program, but also on how well a project manager fulfills all stakeholders' requirements and delivers a program or project on time and on budget that matches the requirements because the software in a system feels the pressures of change (Brooks, 1986). This means that to satisfy the success criteria, a critical first step for a project manager is to objectively estimate project scope, cost, time, and so on so that it enables little or no changes during project performance. Objective estimation can occur in three different areas such as schedule estimation in the early stage (i.e. proposal preparation stage), schedule refinement / verification in the planning stage using design documentations, and decision making to determine the type of software development (i.e. new software development, reuse-based without modification, and reuse-based with modification) (Forselius, 2008).

Yet, the biggest problem of objective estimation is that it tends to heavily depend on domain specific knowledge, judgment, and experience; this means that a lot of risks are potentially inherent in the very early stage (PMI, 2013). Specifically, expert judgment is also used to estimate the project schedule and cost in the proposal stage and the planning stage; as a result, it has a chance to cause schedule / cost overruns which are one of the project cancellation reasons. Expert judgment may be the best technique for estimation of the project schedule and cost, but the issue is that expert judgment is highly subjective. Additionally, it goes without saying that subjective tools and techniques decrease the certainty of decision making for determining the type of software development as well as accuracy of schedule estimation. Therefore, this research focuses on proposing objective tools and techniques to increase such certainty and accuracy focusing on the primary research

question, “Can project determination and time / schedule estimation be improved by using objective tools and techniques?”.

The answer to this question is provided by developing objective methodologies to address three problems in the early stages of the project life cycle. The overall focus will be addressed by answering the following specific 3 questions:

- Research Question 1 (RQ1) – Can an objective technique / framework be developed to help estimate the project schedule in the proposal preparation stage?
- Research Question 2 (RQ2) – Can the estimated project schedule in the proposal preparation stage be refined / verified in the planning stage using design documentation?
- Research Question 3 (RQ3) – Can an objective technique / framework be developed for early-stage comparison of software development project types?

Initially, the objective technique is applicable to estimating the project schedule in the early stage (i.e. project proposal preparation stage). Since a contractor has a responsibility to prove the capability to complete a project on time, probabilistic schedule estimation will be beneficial to proposal preparation. This is due to two reasons: probabilistic schedule estimation is able to show the chance of completing a project within a given time frame, and it is very difficult to estimate the project schedule accurately using only the high level of Statement of Work (SOW) and project requirements (Emam & Koru, 2008; Tesch, Kloppenborg, & Frolick, 2007).

After the initial estimations, the objective technique is applicable to refinement / verification of early stage probabilistic software project schedules in the



planning stage using design documentations. Based on the additional detail from design documents at this stage, Function Point Analysis (FPA) can be done using the more accurate Adjusted Function Point (AFP) count versus the Unadjusted Function Point (UFP) count used in the proposal stage (ISBSG, 2010; Uemura, Kusumoty, & Inoue, 1999).

A third use for objective techniques is for early-stage comparison of software development project types. A software project type should be carefully determined with consideration of various factors; cost estimation based on probabilistic project schedule estimation, Net Present Value (NPV) and Expected Monetary Value (EMV), and the success rate (possibility). Those factors are major components for a Decision Tree tool. Especially, cost-benefit analysis should be conducted as an objective cost estimation technique based on quantitative project risk analysis.

## **1. 1 Outline of the Dissertation**

This dissertation is divided into the following chapters: Introduction, Background, Literature Review, Research Methodologies, Simulation Section, Results and Analysis, and Conclusion and Contribution.

This chapter, chapter 1, introduced the research problem and provided the research question and topic. It also provided a brief discussion on the objective tools and techniques in terms of project schedule estimation, refinement / verification, and early-stage comparison of software development project types.

Chapter 2, background section, describes why this research seeks to develop objective tools and techniques. It outlines the concept of five project process groups, processes, a project life cycle, knowledge areas, and estimation and determination

issues which result from uncertainty. This uncertainty results from expert judgment which is a subjective estimation technique. Probabilistic schedule estimation, refinement / verification of the estimated probabilistic schedule, and Decision Tree-based comparison of software development project types were chosen as the research methodology because uncertainty management is associated with quantitative risk analysis.

Chapter 3, literature review section, covers three main topics: how to perform probabilistic schedule estimation in the early stage, how to refine / verify the estimated probabilistic schedule in the planning stage, and how to objectively compare software development project types in the early-stage. This chapter concentrates on defining existing methodologies (or tools and techniques), justifying the necessity of this research, and delineating the main framework.

Chapter 4, demonstration section, shows how the research methodology and framework have been developed and applied and how they are beneficial to schedule estimation, schedule refinement / verification of the estimated project schedule, and comparison of software development project types based on a scenario and assumptions. This chapter also analyzes the demonstrated results and looks at how the demonstration result is different from the existing tools and techniques as well as methodologies.

Chapter 5, conclusion section, includes the following contents: contribution, limitation, and future work. The primary contribution of this research is to develop an objective methodology and framework for schedule estimation, refinement / verification of the estimated probabilistic project schedule, and early-stage

comparison of software development project types which can be applied to no matter what types of projects.

## **Chapter 2 Background**

The definition of a project is a temporary endeavor undertaken to create a unique product, service, or result. The example of this in terms of the IT industry is to develop or acquire a new or modified information system (hardware or software) (PMI, 2013). According to the PMBOK guide, 5th Edition, project management is driven by ten knowledge areas, forty-seven processes, and five process groups such as initiating, planning, executing, monitoring & controlling, and closing. Among these groups, planning accounts for twenty out of forty seven total processes (approximately 48%), so that its role is very significant; that is, regardless of project type, a good project plan is vital since poor project planning can lead directly to project failure. 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. Furthermore, each PM process (which consists of inputs, tools and techniques, and outputs) should be performed through a series of sequential or overlapping phases (PMI, 2013).

### **2.1 Problem Identification**

Analysis of the PMBOK guide, 5th Edition, also reveals that the processes rely heavily on the expert judgment technique compared to other tools and techniques; that is, the expert judgment technique is used in eight out of ten knowledge areas (80%) and twenty-six different processes as shown in Table 1, and it is used much more frequently in the processes compared to other tools and techniques (PMI, 2013).

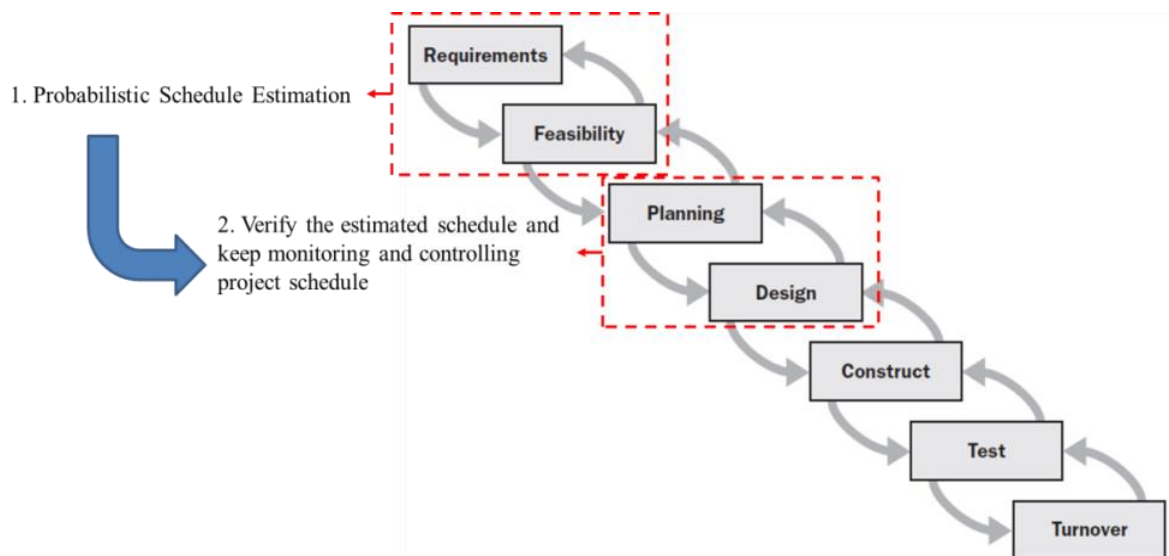
<b>Knowledge Area</b>	<b>Process</b>
<b>Project Integration Management</b>	Develop Project Charter
	Develop Project Management Plan
	Direct and Manage Project Work
	Monitor and Control Project Work
	Perform Integrated Change Control
	Close Project or Phase
<b>Project Scope Management</b>	Plan Scope Management
	Define Scope
	Create WBS
<b>Project Time Management</b>	Define Activities
	Estimate Activity Resources
	Estimate Activity Durations
<b>Project Cost Management</b>	Plan Cost Management
	Estimate Costs
	Determine Budget
<b>Project Communication Management</b>	Control Communications
<b>Project Risk Management</b>	Plan Risk Management
	Identify Risks
	Perform Qualitative Risk Analysis
	Perform Quantitative Risk Analysis
	Plan Risk Responses
<b>Project Procurement Management</b>	Plan Procurement Management
	Conduct Procurements
<b>Project Stakeholder Management</b>	Identify Stakeholders
	Plan Stakeholder Management
	Control Stakeholder Engagement

**Table 1 Knowledge Areas and Processes of using the Expert Judgment**

### **Technique**

Among eight knowledge areas, this research focuses on the project time management knowledge area because a primary reason that IT projects are cancelled is schedule overrun (Emam & Koru, 2008). It is a well-known fact that project schedule estimation is usually performed in a planning stage with consideration of

project scope, activities' relationship, and human resources (PMI, 2013), but it is actually possible to estimate project schedule before initiating a project. Figure 1 illustrates how the first two objectives of this research can be mapped to the general predictive life cycles model (PMI, 2013). That is, it is possible to estimate the project schedule in the requirements and feasibility stage which is considered as the proposal preparation stage and to refine / verify the estimated project schedule in the planning and design stage with design documentation.



**Figure 1 Map Objective Technique to Predictive Life Cycles (PMI, 2013)**

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 in all types of projects (Liu, Yang, Chen, & Li, 2009). Furthermore, such uncertainty results from the fact that contractors depend on high level Statement of Work (SOW)

tasks, requirements and deliverables in the proposal preparation stage. It is especially difficult for software project managers to estimate accurate project schedules because they need to 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).

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).

Another important fact is that project risks originate from uncertainty, and managing such risks is critical (PMI, 2013). There are two kinds of risks, known and unknown, and they are presented using the quadrant form in Figure 2 (Douglas & Ra, 2010; Dobson & Leemann, 2010).

		Outcomes	
		Knowns	Unknowns
Process	Known	Known Knowns (Easily identified from experience)	Known Unknowns (Outcomes can be created using probability in terms of risk management)
	Unknown	Unknown Knowns (Expert judgments are required)	Unknown Unknowns

**Figure 2 Knowns and Unknowns Quadrant Form (Douglas & Ra, 2010)**

The major objective of the “knowns unknowns” quadrant is to transform unknown unknowns into known knowns, known unknowns, or unknown knowns. The risk type for schedule estimation in this research 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 single point estimation. This is done based on inferential statistics, which is the fundamental concept of the Central Limit Theorem (CLT) (Smith & Wells, 2006). Once this is done, the next job for a project manager is to refine or verify the estimated schedule before actual project scheduling in the planning stage for mitigating or avoiding such risk. This point is also related to a predictive life cycle, as shown in Figure 1. Once a project manager estimates the probabilistic project schedule in the requirements and feasibility phase,



the next job is to refine or verify it using a class and sequence diagram as well as relational databases in design documents and to develop a project schedule in the planning and design phase (PMI, 2013; Smith & Wells, 2006). Yet, the question is how to accurately measure or verify the estimated schedule.

The last important issue studied in this work with respect to objective estimation is early-stage comparison of software development project types. According to the PMBOK guide, 5<sup>th</sup> Edition, a project life cycle consists of four different phases such as starting the project, organizing and preparing, carrying out the project work, and closing the project (PMI, 2013). These phases basically map to project life cycle effort which consists of needs identification, proposed solutions, project performance, and project termination (Gido & Clements, 2009). In the first phase of the life cycle, once one or more needs are identified, the next job is to perform project selection. The main objective of project selection is to evaluate a variety of needs or opportunities that can be advantages or disadvantages and to decide which of these should move forward as a project (Gido & Clements, 2009).

Additionally, software development project selection is further complicated by the fact that there are several different potential types of software development that may be undertaken. These include new development, enhancement, and so on (Forselius, 2008). Therefore, a fundamental framework for determining the right type of a software development project needs to be established.

## 2.2 Research Objectives

The main objective of this research is to determine objective methods, supportive tools and techniques, and outputs which can be applied to any types of projects and to improve certainty, accuracy, and lessen the reliance on purely subjective expert judgment. In other words, software development projects are the focus of this research, but fundamental methods, tools and techniques, and outputs which are determined by this research are not limited to software development projects, but can be applied to any project domain.

There is one primary research question; three specific sub-questions will be addressed to answer the main question.

- Primary Research Question - Can project determination and time / schedule estimation be improved by using objective tools and techniques?
- Research Question 1 (RQ1) – Can an objective technique / framework be developed to help estimates the project schedule in the proposal preparation stage?
- Research Question 2 (RQ2) – Can the estimated project schedule in the proposal preparation stage be refined / verified in the planning stage using design documentation?
- Research Question 3 (RQ3) – Can an objective technique / framework be developed for early-stage comparison of software development project types?

Based on all research questions above, this research will focus on providing objective frameworks for two specific critical areas for project managers: project scheduling and early stage comparison of software development project types. The scheduling domain will be subdivided into two separate problems; that of schedule estimation in the proposal stage and schedule refinement / verification in the planning stage, both focusing on a large-sized project. To fulfill such research objective, the scenario-based case will be used.

### **2.3 Research Methodology**

To investigate the three above research questions, this work uses multiple methodologies, techniques, and software applications.

The Central Limited Theorem (CLT), Program Evaluation and Review Technique (PERT) analysis, International Software Benchmarking Standards Group (ISBSG) equations, and function point calculation based on International Function Point User Group (IFPUG) are fundamental methodologies for estimating project schedule in the proposal preparation stage (RQ1).

All those methodologies are used for refinement / verification of the estimate project schedule in the planning stage (RQ2). However, Unified Modeling Language (UML) diagrams such as a class and sequence diagram and relational database are also fundamental methodologies for this research question.

The commonly used tools for both research questions are WBS chart pro, @RISK, and Microsoft Project 2010 (MSP 2010). WBS chart pro is used to make project Work Breakdown Structure (WBS), @Risk is utilized to generate probabilistic

project schedule based on the Monte Carlo technique, and MSP 2010 is used to make a relationship between each work package.

Additionally, a decision tree framework that includes quantitative risk management related to project cost and possible project success is utilized for early-stage comparison of software development project types (RQ3). The quantitative risk analysis consists of cost estimation, possibility (success rate), Net Present Value (NPV), and Expected Monetary Value (EMV).

## **Chapter 3 Literature Review**

The literature review is subdivided into three areas of research. The first and second area discusses aspects of software project schedule estimation and refinement / verification from the literature that are related to the research objective. The third area discusses how to objectively compare software development project types in the early-stage with consideration of project cost estimation, success rate (possibility), and so on.

### **3.1 Schedule Estimation Processes in the Early Stage (Proposal Preparation Stage)**

The PMBOK guide, 5<sup>th</sup> Edition, defines that project time management has total seven processes such as plan schedule management, define activities, sequence activities, estimate activity resources, estimate activity durations, develop schedule, and control schedule. Each process accepts outputs from previously performed processes as inputs and generates outputs which will be an input for other processes by performing appropriate tools and techniques (PMI, 2103). Yet, these processes belong to a planning process group, and necessary factors are project activities, predecessor and successor relationship between activities, and resources which are not suitable to estimate software project schedule in the early stage. For this reason, it initially needs to be examined which factors and processes are crucial to estimate software project schedules.

The literature defined seven software project estimation steps sequentially such as requirements collections, product size estimation, effort estimation, schedule creation, cost estimation, estimation approval, and product development (Nasir, 2006).

Since this research is concerned only through step 4 (schedule creation), the main concerned factors are user requirements, size, and effort.

Based on the above, the first step towards schedule estimation is to identify and understand the stakeholders' requirements. To do so, the best way is to create a project Work Breakdown Structure (WBS) to represent the project scope (PMI, 2013). Then, it is clear that methodologies used to estimate size needed to be examined. The literature (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 second step from above is concerned with project size estimation. 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, probabilistic methods have great potential for solving the size estimation problem with respect to project scheduling. Additionally, as it is mentioned in Figure 2, 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, this research focuses on the probabilistic method which refers to PERT sizing based on FPA. Also, FPA is selected over SLOC because function points can be more readily and accurately measured in the requirements phase (Nassif, Capretz, & Ho, 2010; Lind & Heldal, 2010).

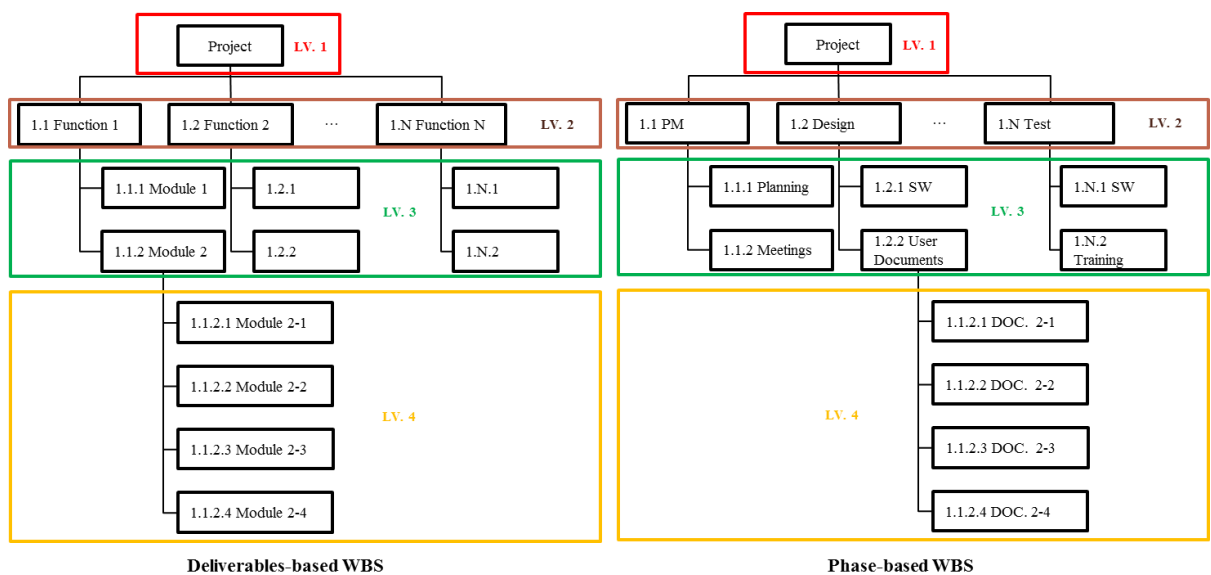
The third step examined within the literature is how to estimate effort. There are a variety of models to estimate effort: analogy-based effort estimation (Chiu & Huang, 2007; Kocaguneli, Menzies, Bener, & Keung, 2012; Cherjee, Mahanti, & Jumar, 2009; Basha & P, 2010), regression equations, COCOMO, and so on. (Basha & P., 2010; ISBSG, 2010). Among them, the regression equations which were generated by data analysis of the International Software Benchmarking Standards Group (ISBSG) repository based on International Function Point User Group (IFPUG) FPs are selected (ISBSG, 2010). This is because 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 & P, 2010).

According to the literature research, 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 the research objective to produce better project schedule estimates in the proposal stage.

### **3.1.1 Work Breakdown Structure (WBS) and WBS Chart Pro Tool**

According to the PMBOK guide, 5th Edition, one of processes in the Project Scope Management knowledge area is to create Work Breakdown Structure (WBS) (PMI, 2013). This is the process of decomposing major deliverables into smaller pieces of components which are called Work Packages, and the meaning of major deliverables in this research is major system functionalities. This WBS type is called a deliverables-based WBS, but one thing to keep in mind is that there is one more WBS type which is called a phase-based WBS (PMI, 2013). This is related to the WBS structure. So the fundamental difference between the two WBS types is that phase-based WBS uses phases of the project life cycle as the 2<sup>nd</sup> level of WBS and the product and major deliverables are located in the 3<sup>rd</sup> level of WBS. However, deliverables-based WBS uses major deliverables as the 2<sup>nd</sup> level of WBS, and the components of each major deliverable are located in 3<sup>rd</sup> level (PMI, 2013). The difference is also shown in Figure 3. Since we need to identify and depict at least 30 work packages (small components or modules) to invoke the Central Limit Theorem (CLT), the WBS chart pro tool is used to facilitate this.





**Figure 3 Difference between Deliverables-based and Phase-based WBS**

### 3.1.2 Function Point Analysis (FPA)

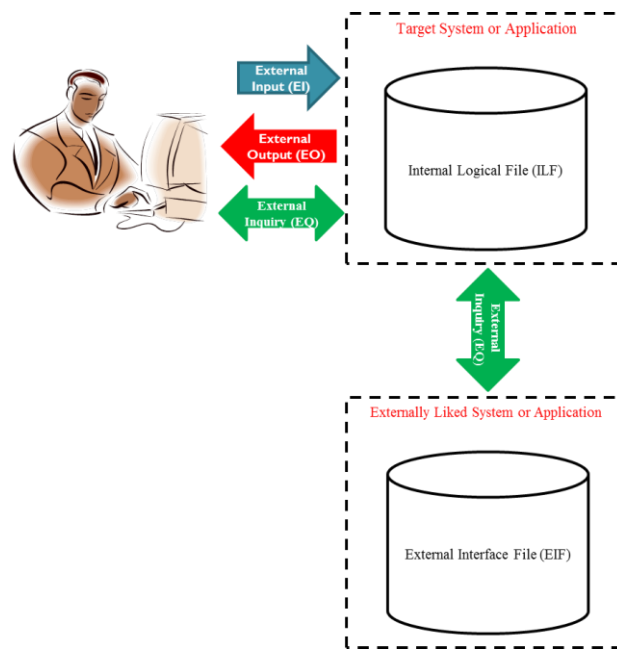
The Function Point Analysis (FPA) is firstly introduced by Allan J. Albrecht in the mid 1970s, and its main objective is to make up for the weakness of Source Lines of Code (SLOC) for measuring software size (Longstreet, 2005). There are a variety of standards for software size measurement based on FPA which is approved by the International Organization for Standardization (ISO) such as International Function Point Group (IFPUG), Finnish Software Measurement Association (FiSMA), Mark-II (MkII), Netherlands Software Metrics Association (NESMA), and Common Software Measurement International Consortium (COSMIC) (Total Metrics, 2007).

The main benefits of IFPUG FPA are to provide the greatest accuracy and flexibility for software size measurement and to quantitatively measure user requirements (Garmus, 2006). Additionally, the IFPUG FPA technique is that it has a long history and is the widely used technique for software size measurement. The

fundamental concept of the IFPUG FPA technique is to quantify the functions contained within software in terms that are meaningful to the software users. Additionally, system functions should be counted using functional component types such as External Input (EI), External Output (EO), External Inquiry (EQ), Internal Logical File (ILF), and External Interface File (EIF).

- External Input (EI): is an elementary process that data comes from user input or other applications. This data is used to manage or maintain one or more ILF (Longstreet, 2005).
- External Output (EO): is an elementary process that the derived data moves out from ILF to users (Longstreet, 2005).
- External Inquiry (EQ): is an elementary process that retrieves data from one or more ILF and ELF (Longstreet, 2005).
- Internal Logical File (ILF): is logically and related dataset and used to managed and maintained through EI (Longstreet, 2005).
- External Interface File (EIF): is also logically and related data set, but used for reference only (Longstreet, 2005).

Based on the fundamental concept of each functional component above, the architecture of IFPUG FPA is depicted in Figure 4. This concept is applied to measure the size of each work package for this research, and the PERT technique should be used to calculate Unadjusted Function Point (UFP).



**Figure 4 IFPUG Function Point Architecture**

### 3.1.3 Adjusted Function Point (AFP)

Once each functional component is counted and the value of UFP is determined, the next step is to determine the Value Adjusted Factor (VAF) and to calculate Adjusted Function Point (AFP). This is basically associated with 5 steps to count function points: Determine the type of count, identify counting scope and application boundary, determine the UFP, determine the VAF, and calculate the AFP (Alexander, 2011).

The VAF consists of 14 general system characteristics such as data communication, distributed data processing, performance, and so on, and the influence degree of each characteristic ranges on a scale 0 to 5 (0: No influence, 5: Strong influence) (ISBSG, 2010; Uemura et al., 1999). Then, the value of AFP can be calculated using the equation:

$$AFP = \text{Count total (UFP)} \times [0.65 + 0.01 \times \Sigma(F_i)] \text{ (Azath \& Wahidabanu, 2008)}$$

where the value of  $\Sigma(F_i)$  is sum of influence scale.

### 3.1.4 Program Evaluation and Review Technique (PERT) Analysis

The fundamental concept of PERT analysis comes from the weak point of single point estimation; in other words, the accuracy of single point estimates may be improved by considering uncertainty and risk (PMI, 2013). A high level of uncertainty and risk may be reduced using the PERT technique because it considers three estimates such as optimistic, most likely, and pessimistic to define an approximate range. Basically, there are two equations:

Expected Value

$$= \frac{(\textit{Optimistic Value} + \textit{Most Likely Value} + \textit{Pessimistic Value})}{3}$$

Expected Value

$$= \frac{\{(1 * \textit{Optimistic Value}) + (4 * \textit{Most Likely Value}) + (1 * \textit{Pessimistic Value})\}}{6}$$

The first equation is used for Triangular Distribution, and the second one is used for Beta Distribution (PMI, 2013). Theoretically, both distribution types are very similar; that is, they emphasize the most likely value to provide better estimation based on probabilities, but the biggest difference between them is that Beta Distribution may generate more realistic probability distribution (Gido & Clements, 2009). For this reason, the Beta Distribution of the PERT technique is applied to this research. This concept is used to calculate the UFP value, which is the main input of project size and project duration. The number of each functional component type (EI,

EO, EQ, ILF, and EIF; these are defined above) will be measured by a probabilistic technique (optimistic, most likely, and pessimistic), and this concept will be also used to produce a project schedule range with probabilities. The equation for calculating the number of UFP is shown in Table 2 (Pressman, 2009).

Category	3-Point Estimation			Expected Count	Weight Factors			Sub Total
	OP	ML.	Pess.		Simple	Average	Complex	
EI	OP Est.	ML Est.	Pess Est.	$\{ 1*(Pess.)+4*(ML.)+1*(OP.) \} / 6$	3	4	6	(Expected Count) * (One of Weight Factors)
EO	Same as above	Same as above	Same as above	Same as above	4	5	7	Same as above
EQ	Same as above	Same as above	Same as above	Same as above	3	4	6	Same as above
ILF	Same as above	Same as above	Same as above	Same as above	7	10	15	Same as above
EIF	Same as above	Same as above	Same as above	Same as above	5	7	10	Same as above
<b>Grand Total (UFP)</b>								Sum of sub total

**Table 2 Equation of Unadjusted Function Point Calculation**

### 3.1.5 Issues of Counting UFPs

There are two remarkable issues regarding UFP counting and calculation. The first issue is that the sum of counted UFPs from the work packages may not work properly for the size measurement of the higher WBS level. For example, assume that there is WBS which consists of 3 levels. 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 and to add it up. However, 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 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;

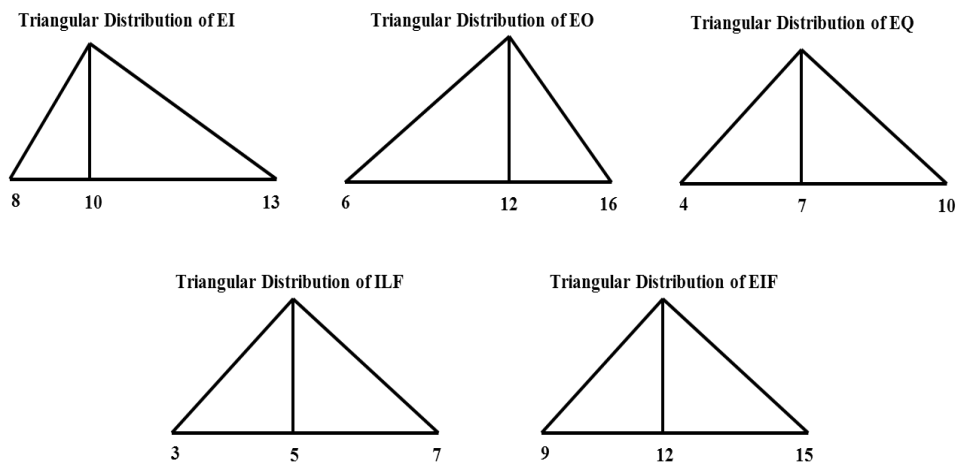
From the above, it is easy to figure out that the same database table is used for both modules. As a result, it clearly points out that the sum of counted UFPs from WBS level 3 does not always work properly for WBS level 2.

The second issue is that the equations in Table 2 are theoretically valid, but the problem is that the Expected Count value is a weighted average of a 3-point estimation. So it is considered as the mean value. This means that the mean value is a population mean that indicates only 50% of a probability distribution result, so it does not correspond to the research objective that generates the range of schedule estimation with consideration of all possibilities based on inferential statistics. For instance, suppose that the number of EIs, EOs, EQs, ILFs, and EIFs is as shown in Table 3.

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

**Table 3 Example Number of Functional Component Types**

According to Table 3, 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 fundamental concept of the Monte Carlo technique will be explained in the section 3.1.7.

### **3.1.6 International Software Benchmarking Standards Group (ISBSG)**

#### **Equations**

ISBSG equations are defined by the ISBSG, and they are used to calculate resource effort. There are two major inputs to get project duration: Project Delivery Rate (PDR) and Resource Effort. The PDR value can be calculated by two cases. The first case is to use the equation,

$$\text{PDR} = C * \text{Size}^{E1} * \text{Maximum Team Size}^{E2} \text{ (ISBSG, 2010)}$$

if a project manager cannot expect the same productivity from all team members, and the second case is to use the fixed PDR value if a project manager expects same productivity. Furthermore, the value of C, E1, and E2 in the first equation can be found from the ISBSG data repository according to development type, platform, and programming language, and the fixed PDR value in the second case can be also found based on programming language and platform. The applicable values for the variables in the above equation as well as for the fixed PDR value are shown in Table 4 through Table 14 (ISBSG, 2010). After that, resource effort can be calculated by PDR times UFP.



<b>Class</b>	<b>C</b>	<b>E1</b>	<b>E2</b>
All	57.39	-0.558	0.710
Enhancement	79.12	-0.616	0.692
New Development	37.48	-0.496	0.759
Midrange	60.76	-0.664	0.960
Multi	34.49	-0.510	0.875
3GL	51.74	-0.526	0.693
4GL	32.90	-0.468	0.692
New & Midrange	35.09	-0.597	1.080
New & Multi	37.41	-0.463	0.736
Enh & Midrange	115.90	-0.759	0.872
Enh & Multi	38.97	-0.566	0.951
New & 3GL	39.40	-0.489	0.762
Enh & 4GL	64.10	-0.605	0.728
MR & 3GL	42.94	-0.605	0.994
MR & 4GL	56.86	-0.664	0.967
Multi & 3GL	36.44	-0.491	0.832
Multi & 4GL	9.35	-0.282	0.801
Enh & MR & 3GL	81.76	-0.647	0.785
Enh & ER & 4GL	162.70	-0.865	0.963
New & Multi & 3GL	72.34	-0.530	0.666
New & Multi & 4GL	6.72	-0.228	0.839
Enh & Multi & 3GL	25.63	-0.462	0.909
Enh & Multi & 4GL	13.98	-0.372	0.829

**Table 4 The value of C, E1, and E2 in the PDR Equation (ISBSG, 2010)**

	<b>Min</b>	<b>10%</b>	<b>25%</b>	<b>Median</b>	<b>75%</b>	<b>90%</b>	<b>Max</b>	<b>Mean</b>
2 <sup>nd</sup> gen Language	3.8	4.6	8.2	15.7	23.0	34.8	45.8	17.9
3 <sup>rd</sup> gen Language	0.6	3.5	5.9	11.4	22.5	37.9	79.7	16.7
4 <sup>th</sup> gen Language	1.2	3.2	5.4	8.7	14.5	21.9	55.5	11.3
5 <sup>th</sup> gen Language	6.4	8.5	9.8	16.1	22.2	25.5	37.1	17.1
Application Generator	4.7	5.6	8.2	10.8	16.1	26.9	48.3	14.7

**Table 5 Project Delivery Rate by Language Type – All Platforms (ISBSG, 2010)**

	Min	10%	25%	Median	75%	90%	Max	Mean
3 <sup>rd</sup> gen Language	0.6	3.7	7.8	15.3	27.5	42.2	79.7	19.9
4 <sup>th</sup> gen Language	1.2	3.2	4.6	7.3	17.5	29.2	52.5	12.0
Application Generator	4.7	5.9	8.9	11.5	17.5	27.7	48.3	15.3

**Table 6 Project Delivery Rate by Language Type – Mainframe Platforms**

(ISBSG, 2010)

	Min	10%	25%	Median	75%	90%	Max	Mean
3 <sup>rd</sup> gen Language	1.3	4.4	7.0	10.8	20.8	33.8	74.2	16.2
4 <sup>th</sup> gen Language	2.0	5.5	7.8	13.2	19.9	28.8	55.5	16.0

**Table 7 Project Delivery Rate by Language Type – Midrange Platforms (ISBSG,**

**2010)**

	Min	10%	25%	Median	75%	90%	Max	Mean
3 <sup>rd</sup> gen Language	1.0	2.8	4.6	8.6	12.6	22.7	60.1	10.9
4 <sup>th</sup> gen Language	1.2	2.4	2.8	5.9	13.1	16.4	33.8	8.5

**Table 8 Project Delivery Rate by Language Type – PC Platforms (ISBSG, 2010)**

	Min	10%	25%	Median	75%	90%	Max	Mean
3 <sup>rd</sup> gen Language	0.8	3.6	5.6	9.1	18.0	31.1	61.8	13.7
4 <sup>th</sup> gen Language	1.4	3.6	5.5	8.2	11.5	17.1	35.7	9.5
5 <sup>th</sup> gen Language	6.5	8.8	11.5	17.2	22.0	25.1	31.8	17.4

**Table 9 Project Delivery Rate by Language Type – Multi Platforms (ISBSG,**

**2010)**

	<b>Min</b>	<b>10%</b>	<b>25%</b>	<b>Median</b>	<b>75%</b>	<b>90%</b>	<b>Max</b>	<b>Mean</b>
ABAP	4.2	7.0	7.9	11.3	15.6	21.4	34.3	13.0
Access	1.6	2.4	2.7	7.1	8.7	13.0	14.5	6.8
ASP	1.8	2.6	3.6	6.7	9.9	15.6	30.6	8.6
Assembler	3.8	4.6	8.2	15.7	23.0	34.8	45.8	17.9
C	1.8	3.6	8.3	13.6	24.4	41.0	76.5	18.8
C++	1.0	4.9	8.2	14.8	31.3	54.2	78.7	23.1
C#	1.9	6.1	9.6	15.1	25.1	39.7	49.8	18.8
COBOL	0.8	4.2	6.7	15.3	28.1	48.6	79.7	21.1
Cool:gen	4.7	5.6	8.2	10.8	16.1	26.9	48.3	14.7
HTML	1.0	3.5	4.3	13.7	22.3	40.3	48.0	17.2
Java, J2EE	1.9	4.8	5.9	8.0	15.6	29.4	74.2	13.3
Lotus Notes	1.2	1.5	2.7	3.7	5.1	9.5	12.2	4.7
Natural	3.4	5.1	5.7	10.2	13.9	25.0	35.3	12.2
Oracle	1.2	3.0	4.7	8.2	15.7	23.8	33.8	11.2
PL/1	0.6	2.9	5.7	16.0	22.9	34.3	61.8	16.9
PL/SQL	0.8	1.3	1.7	4.6	9.7	26.4	42.1	9.4
Powerbuilder	4.2	5.0	6.4	9.3	14.1	18.6	23.6	10.9
Scripting language	1.4	3.7	5.1	7.6	13.2	22.5	61.8	11.7
SQL	2.4	3.9	6.2	11.4	16.7	27.2	55.5	13.5
Visual Basic	0.6	2.4	4.1	8.5	18.1	34.7	69.4	13.5
Other 3GL	4.0	6.7	10.2	14.2	22.5	30.1	43.1	16.8
Other 4GL	3.6	6.0	7.8	9.2	13.2	19.2	35.7	11.6
5GL	6.4	8.5	9.8	16.1	22.2	25.5	37.1	17.1
Other	0.7	2.5	5.3	8.6	15.0	24.1	59.3	11.6

**Table 10 Project Delivery Rate by Language – All Platforms (ISBSG, 2010)**

	<b>Min</b>	<b>10%</b>	<b>25%</b>	<b>Median</b>	<b>75%</b>	<b>90%</b>	<b>Max</b>	<b>Mean</b>
C	5.6	9.2	11.7	15.5	27.5	42.5	60.4	21.6
C++	5.8	10.7	17.1	32.2	49.6	53.3	75.2	34.3
COBOL	0.8	4.2	7.5	16.8	32.3	54.6	79.7	23.0
Cool:gen	4.7	5.9	8.9	11.5	17.5	27.7	48.3	15.3
Java	3.1	5.1	11.4	18.1	27.4	29.4	31.6	18.1
Oracle	1.2	2.9	4.3	6.6	18.5	29.7	31.7	12.1
PL/1	0.6	2.3	4.2	13.2	22.2	28.5	55.1	14.9
Scripting language	1.4	5.7	9.1	13.2	22.1	29.5	61.8	17.5
Visual Basic	0.6	3.6	18.4	27.4	30.3	38.9	54.6	24.7
Other 3GL	4.8	7.2	10.2	13.3	19.5	31.3	43.1	16.6
Other	0.7	2.9	6.4	10.7	16.0	31.5	52.5	13.9

**Table 11 Project Delivery Rate by Language – Mainframe Platforms (ISBSG,**

**2010)**

	Min	10%	25%	Median	75%	90%	Max	Mean
C	3.6	8.4	13.1	15.1	22.4	29.5	34.2	17.6
C++	1.3	3.9	5.1	7.9	15.0	19.0	49.6	11.8
Java	4.2	4.5	7.3	9.3	20.9	60.1	74.2	19.0
Oracle	2.0	3.0	5.8	9.0	14.6	24.4	28.8	11.3
SQL	4.1	5.8	9.6	13.3	20.1	29.2	55.5	16.8
Other	3.5	5.3	7.1	10.5	20.8	33.8	42.1	15.1

**Table 12 Project Delivery Rate by Language – Midrange Platforms (ISBSG, 2010)**

	Min	10%	25%	Median	75%	90%	Max	Mean
ASP	2.2	2.6	2.7	5.9	7.8	9.5	14.3	6.0
C++	4.0	8.6	9.3	11.4	18.5	27.8	60.1	16.5
COBOL	2.8	4.2	5.2	10.4	19.7	24.0	35.1	12.7
Java	1.9	3.0	5.7	7.7	10.9	19.0	25.3	9.3
Oracle	1.2	2.3	3.7	9.0	13.5	19.8	33.8	10.6
Visual Basic	1.0	1.9	3.2	7.2	9.5	13.8	24.4	7.4
Other	1.0	2.2	3.6	7.3	14.6	25.6	49.8	11.2

**Table 13 Project Delivery Rate by Language – PC Platforms (ISBSG, 2010)**

	Min	10%	25%	Median	75%	90%	Max	Mean
ASP	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	13.0	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	8.0	13.7	22.8	32.2	48.8	16.7
Java	3.1	5.0	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	8.0	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
Other 3GL	4.8	7.8	10.9	17.3	22.7	30.0	38.0	17.8
Other 4GL	3.6	6.0	7.8	8.7	12.5	19.2	35.7	11.3
5GL	6.5	8.8	11.5	17.2	22.0	25.1	31.8	17.4
Other	1.1	3.1	4.8	7.4	10.4	14.8	27.4	8.7

**Table 14 Project Delivery Rate by Language – Multi Platforms (ISBSG, 2010)**

There are two ways to get project duration. If the team size is known, use the equation: (Effort/team size) / Number of hours worked per person per month. If team size is unknown, use the equation:  $0.370 * \text{Effort}^{0.328}$ . The PDR, effort, and duration calculations are summarized in Table 15 below.

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 15 ISBSG Regression Equations (ISBSG, 2010)**

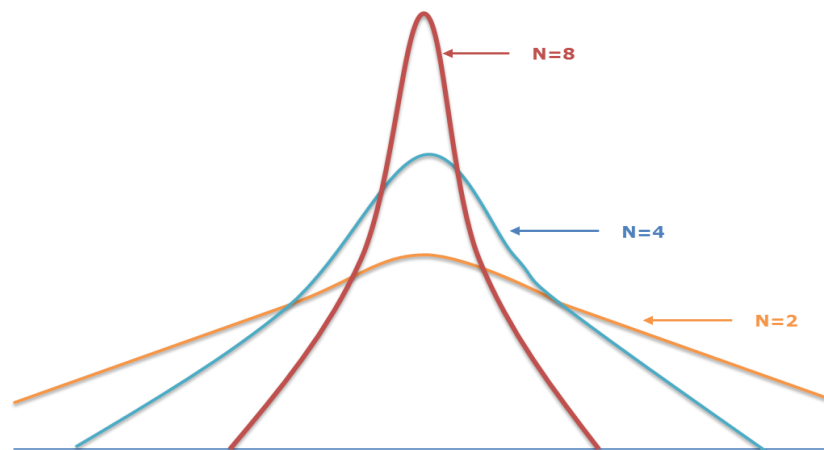
### 3.1.7 @ Risk

The Palisade Company developed the @ Risk software which performs risk analysis using the Monte Carlo technique to show many possible outcomes (Palisade Corporation, 2010). The key idea of using the Monte Carlo technique is that since the number of each functional component is measured in terms of 3 point estimates, there are many possibilities to pick any random numbers between the optimistic and pessimistic values of each functional component type. So randomly chosen input values are used to transform the triangular distributions into normal probability distributions which will be calculated from the iterations and using this software (PMI, 2013). Therefore, it is required to simulate all cases hundreds or thousands times to calculate UFP for software size measurement for the research.

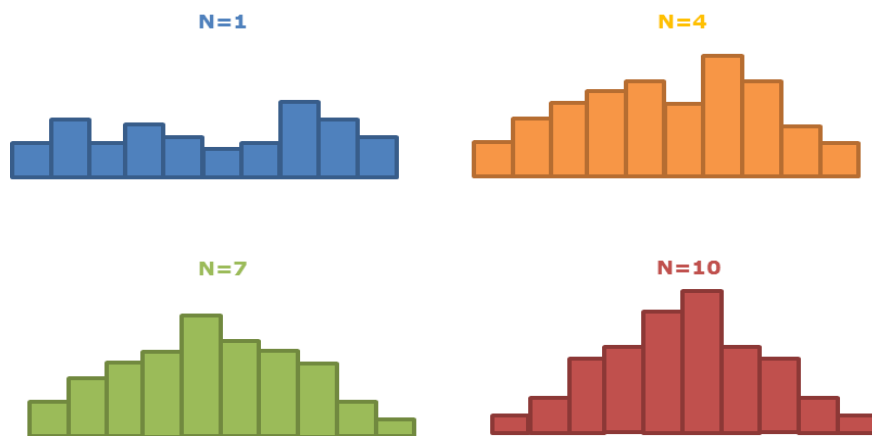
### 3.1.8 Central Limited Theorem (CLT)

Final schedule estimation uses two fundamental concepts: sampling distribution of the mean and Central Limit Theorem (CLT). The key idea of the sampling distribution of the mean is that it has a mean  $\mu$  and a standard deviation  $\sigma / \sqrt{N}$  ( $N$  = sample size) if a population is given with mean  $\mu$  and standard deviation  $\sigma$

(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  $X_1, X_2, \dots, X_n$  ( $n$ =sample size) with mean  $\mu$  and variance  $\sigma^2$ , the sample mean is:  $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$  (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). All the explanations above are depicted in Figure 6 and Figure 7.



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



**Figure 7 The concept of the CLT**

### **3.1.9 Microsoft Project 2010**

Microsoft Project 2010 is project schedule management software. The reason of using this software is that after exporting the created WBS to Microsoft Project 2010, it is required to make a predecessor relationship between work packages. This should be performed due to the issue that the schedule of the higher WBS level may vary according to the predecessor relationship between work packages.

### **3.2 Schedule Refinement or Verification Processes**

The key point of the section above is that schedule estimation is very difficult in the early stage due to a high level of uncertainty; the premise is that probabilistic schedule estimation can increase the accuracy of these early estimates. Yet, since schedule estimation in the early stage is performed based on the high level stakeholders' requirements and Statement of Work (SOW) without actual design documents, refinement / verification of the estimated probabilistic schedule is needed.

Schedule refinement / verification should be performed because the size estimation process in the early probabilistic schedule estimation is based on 3-point estimate. Basically, UFP, which are computed by 3-point estimates and the Monte Carlo technique, provide the primary input for project size estimation. The reason of using 3-point estimates in the size estimation in the early stage is because of uncertainty which results from a lack of information, historical records, and experience for similar projects.

However, in the planning stage the 3-point estimates are no longer necessary because the number of functional components such as External Input (EI), External Output (EO), External Inquiry (EQ), Internal Logical File (ILF), and External Interface File (EIF) can be accurately determined through the UML diagrams (Irawati & Mustofa, 2012). Since it is possible to count the exact number of the five functional components, it is not necessary to rely on the 3-point estimates at this stage. Additionally, Adjusted Function Points (AFP) can be used as the primary input in the size estimation process instead of using the UFPs alone. The AFP is calculated by multiplying the UFP by a Value Adjustment Factor (VAF) (ISBSG, 2010; Uemura et al., 1999). The VAF is derived by considering fourteen general system characteristics: data communications, distributed data processing, performance, heavily used configuration, transaction rate, online data entry, end-use efficiency, online update, complex processing, reusability, ease of installation, ease of operation, multiple sites, and facilitation of change; the influence degree of each characteristic ranges on a scale from 0 to 5 (0: No influence, 5: Strong influence) (ISBSG, 2010; Uemura et al., 1999).



The next area to be examined within this section is related to which UML diagrams support the determination of the accurate number of each function point category. The research defined three UML diagrams as useful for base functional component analysis: use case diagram, the class diagram, and the sequence diagram (Iorio, 2004). The author explained that both the transaction functions (such as EI, EO, and EQ) and the data functions (such as ILF and ELF) can be obtained from the use case diagram, but it is hard to achieve a sufficient level of detail. Therefore, two diagrams (such as the class diagram and the sequence diagram) should be used to provide detailed information in terms of the data functions and the transaction functions (Iorio, 2004). The publication of the Netherlands Software Metrics Users Association, "FPA applied to UML/Use cases Version 1.0", explains the same theory. The research pointed out that a combination of a use case diagram, a class diagram, and a sequence diagram is very suitable for FPA and provides a detailed count of each functional component (The Netherlands Software Metrics Users Association, 2008).

The last area to review that relates to this section involves relational databases. This is because a problem might occur when counting functional components using only class and sequence diagrams; that is, it may cause an inaccurate number for the Internal Logical File data function. As the section 3.1.5 above discussed the issue of function point counting, the number of Internal Logical File (ILF) is related to the number of database tables. Thus, it is necessary to figure out the accurate number of database tables because that number directly affects the calculation of resource effort and project duration. If the relationship of database tables is not considered, it may cause redundancy and result in an inaccurate number of ILF. So it is important to

understand how to efficiently design the database structure from a relational database viewpoint.

Research proposes that a class diagram is associated with the database structure, and a database table can be designed using a class which consists of class name and attributes (Ibrar, 2013). The class name becomes the name of the database table which is same as an entity, and attributes become table columns or fields which is same as the attributes of an entity. Once the database tables and attributes are determined, it is possible to build relational databases using a primary key and foreign key.

There is, of course, other work related to the use of UML models for software size estimation. In this section, size estimation is only the very beginning piece; the final goal is to produce more accurate schedule estimate with related probabilistic chances of completion.

Other differences exist as well. For example, both researches (Lavazza & Robiolo, 2010; Levesque, Bevo, & Cao, 2008) address estimating software size with UML models, but their works use the COSMIC method whereas this research is based on the IFPUG function point analysis. Additionally, they both start with use case diagrams to create the class and sequence diagrams while this research begins with the work breakdown structure (WBS). So WBS-based IFPUG function point analysis in this research allows the creation of a “unified” class diagram across work packages that eliminates potential redundancy in EIs and more accurately represents the required database(s).

Other researches (Zivkovic, Rozman, & Hericko, 2005; Zhou, Yang, Xu, Leung, & Zhou, 2014) are also related to size estimation based on UML diagrams. The main contribution of the first research is an object-oriented to FPA mapping (Zivkovic et al., 2005). Further, their desired “final estimation” is produced much later in the project phase than the “basic estimation” on which this research focuses. The method presented in the second research (Zhou et al., 2014) relates to estimating the source lines of code (SLOC) for object-oriented systems, which is an alternate to the functional size measurement this research uses for effort estimation.

The above items from the literature guided this research into probabilistic schedule estimation and refinement for large-sized projects. Relational databases and UML diagrams from design documents, such as a class diagram and a sequence diagram, are determined to count each functional component, and they are used to refine the estimated project schedule. This point also corresponds with the research objective to refine/verify the estimated project schedule in the planning stage.

### **3.3 Comparison Process for Determining a Software Project Type**

Before determining a software project type, it is required to examine what types of software development projects are available. There are several different software development project types, including customer specific new development, software product new development, software version enhancement, information and communication technology development, package software configuration, data conversion, and software integration development (Forselius, 2008). These project types can be grouped into the following three more general categories:

- Entirely new development: customer specific new development projects and software product new development projects belong to this group.
- Reuse-based with modification: software version enhancement projects, information and communication technology development projects, and software integration development projects are included here.
- Reuse-based without modification: package software configuration projects and data conversion projects are associated with this category

The above classification is used in this research because the enumerated categories are the most commonly used software development project types (Basili, Bailey, Rombach, & Joo, 1987).

The next consideration is the evaluation of a potential development project with respect to the merits of the three above types. There are a variety of criteria to use for evaluation, such as resource availability, project size and duration, technical difficulty and risks, potential benefits, and so on (Hoffer, George, & Valacich, 2010). The typical evaluation methodology depends heavily on expert judgment; however, the overarching goals of this research are to replace subjectivity with objectivity whenever possible. For this reason, this research seeks to utilize objective (quantitative) techniques to evaluate a potential development project.

There are basically two types of models that support this research's aims: numeric and non-numeric (Asaka, Aila, Odera, & Abongo, 2012). A non-numeric model is usually used if a project does not expect huge amounts of resources while a numeric model is used to establish project feasibility (Asaka et al., 2012). Thus, the

numeric model supports the conduct of the evaluation before, or as part of, project selection and can deal with various aspects of the project including economic feasibility, technical feasibility, operational feasibility, schedule feasibility, legal and contractual feasibility, and political feasibility (Hoffer et al., 2010). The numeric model category is comprised of five sub-models: profitability model, cost benefit analysis, scoring model, goal programming, and the weighted factor scoring model (Asaka et al., 2012).

The numeric model is used in this work to compare the three general project types before or during project selection. This research focuses on economic and technical feasibility associated with project risk factors and uses a decision tree structure as provided in (PMI, 2013). From the five numeric model sub-models, this research adopted cost-benefit analysis because it is one of the most widely used techniques for large scale project evaluation (Nickel, Ross, & Rhodes, 2009). While cost-benefit analysis can include both tangible and intangible benefits, following the spirit of this work only tangible benefits will be considered since intangible benefits do not lend themselves to quantitative (objective) measurement (Hoffer et al., 2010).

To conduct cost-benefit analysis, the first step is to estimate accurate project costs and to find out which approach will be less expensive. There are several techniques and equations to help project cost estimation, such as model-based techniques, expertise-based techniques, and so on (Boehm & Abts, 2000). The equations this research uses to estimate project costs align with project types and come from (K.S. & Vasantha, 2008); the equations are shown in Table 16. One common mistake that many stakeholders make is that they are only concerned with

seeking immediate gains without bearing in mind potential profits that can be indicated by considering things such as Net Present Value (NPV), Return on Investment (ROI), Break Even Analysis (BEA), and so on. This research takes into account potential benefits by using Expected Monetary Value (EMV) coupled with a possibility (or chance nodes) which assesses project risks in the technical feasibility area (PMI, 2013).

Category	Equation	Note
Development cost w/ no-reuse	Labor costs + Hardware purchasing expense	N/A
Reuse-based w/o modification	$Cost_{search} + (1-P) * Development_{no-reuse}$	<ol style="list-style-type: none"> <li>1. <math>Cost_{search}</math>: the cost of searching the desired components or modules</li> <li>2. P: probability to find the desired components or modules from the library</li> </ol>
Reuse-based w/ modification	$Cost_{search} + Cost_{adapt} + (1-P) * Development_{no-reuse}$	<ol style="list-style-type: none"> <li>1. <math>Cost_{search}</math> and P are same as above</li> <li>2. <math>Cost_{adapt}</math>: the total amount costs of adapting components or module</li> </ol>

**Table 16 Project Cost Estimation**

While EMV and possibility associated risks are elements that have been found to assist in determining the right project type, two questions still remain. One relates to what objective tools and techniques can support the project type selection, and the other is how to objectively measure or calculate possibility associated risks (success rate) and the technical difficulty of software development. For the first question, this research proposes a decision tree tool because the factors of quantitative risk management (such as project costs and possibility) will be fundamental factors to calculate EMV (PMI, 2013). To address the second question, (Maglyas, 2009)

provides ten criteria and an equation for the success measurement of software development. Table 17 shows the proposed criteria and the equation.

1. User involvement	6. Executive management support
2. Clear statement of requirements	7. Proper planning
3. Realistic expectations	8. Smaller project milestones
4. Competent staff	9. Ownership
5. Clear vision and objectives	10. Hard-working, focused staff
Success Rate = $\sum_{i=1}^{10} w_i \cdot A_i$	
1. $w$ is a set of weighting coefficients which consist of the following numbers: 19, 16, 15, 11, 10, 9, 8, 6, 3, 3	
2. $A_i$ is either of two numbers (0 or 1).	

**Table 17 Success Rate Criteria and Equation (Maglyas, 2009)**

The above items from the literature guided this research into decision making for project development type comparison in the early stages of the project life cycle. This research determined that a decision tree and the concept of success rate are required to make a useful comparison framework. This point corresponds with the study objective to produce better project selection in the early phase and it helps to decide which among the three project types should be selected.

### 3.3.1 Decision Tree

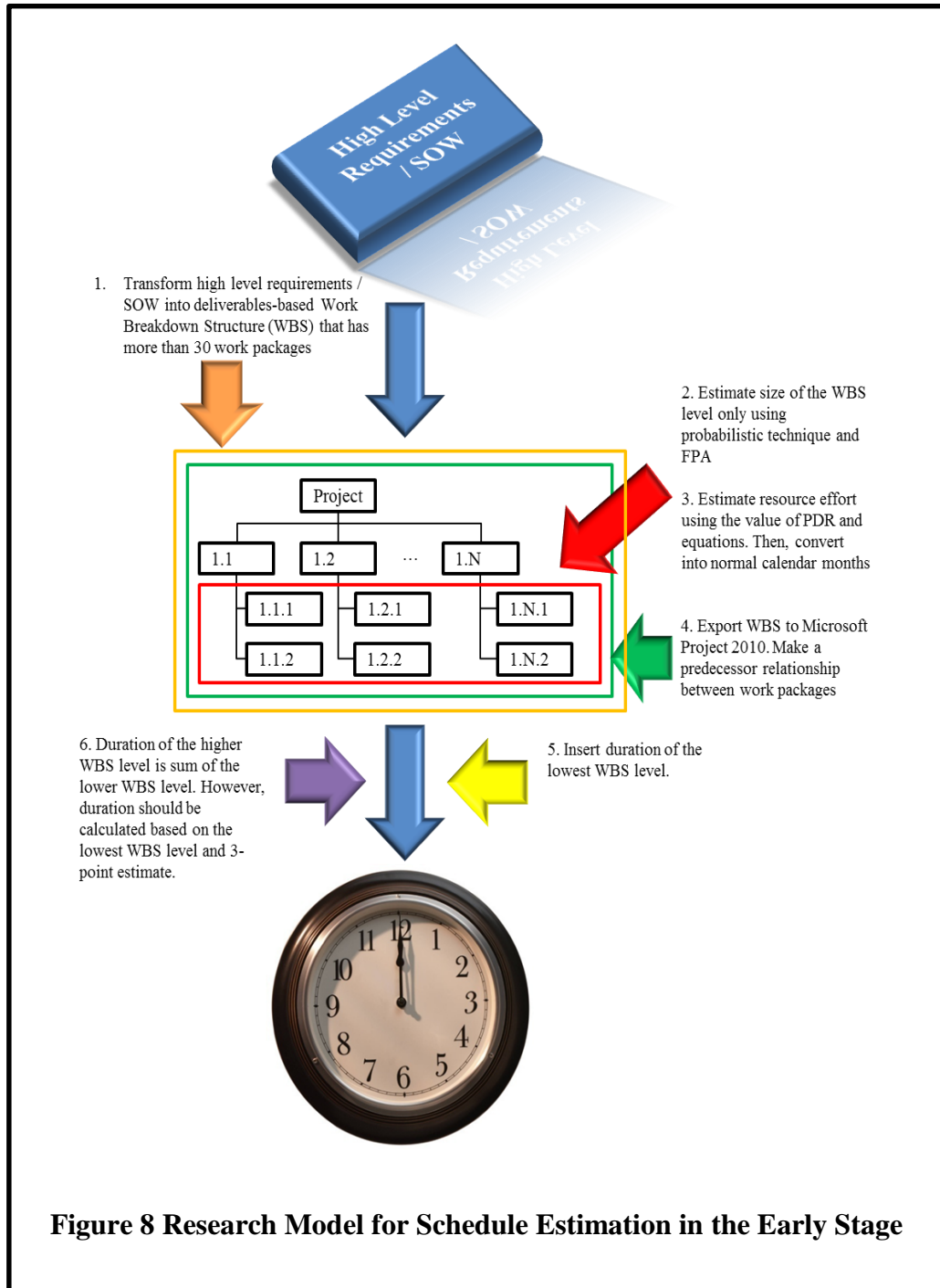
The major objective of a decision tree tool is that it is used for modeling and optimization of probabilistic decision-making problems (Haines et al., 1990). Additionally, it is used to convert a very complex problem into an understandable and comprehensible model (Haines et al., 1990). The fundamental methodology of a decision tree is that it is driven by a probability and Expected Monetary Value (EMV)

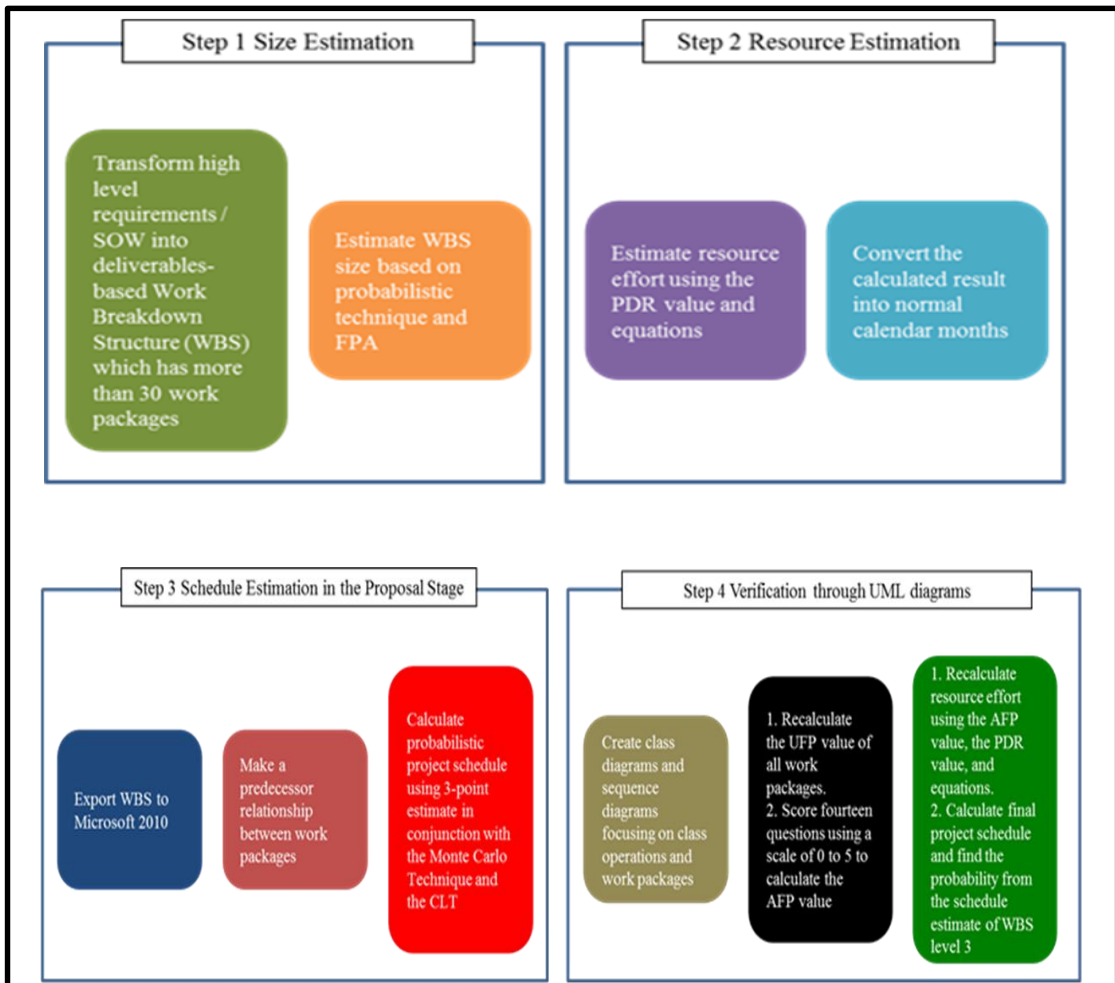
based on quantitative risk analysis (PMI, 2013). The decision tree model is used in this research to provide the objective model for early-stage comparison of software development project types.



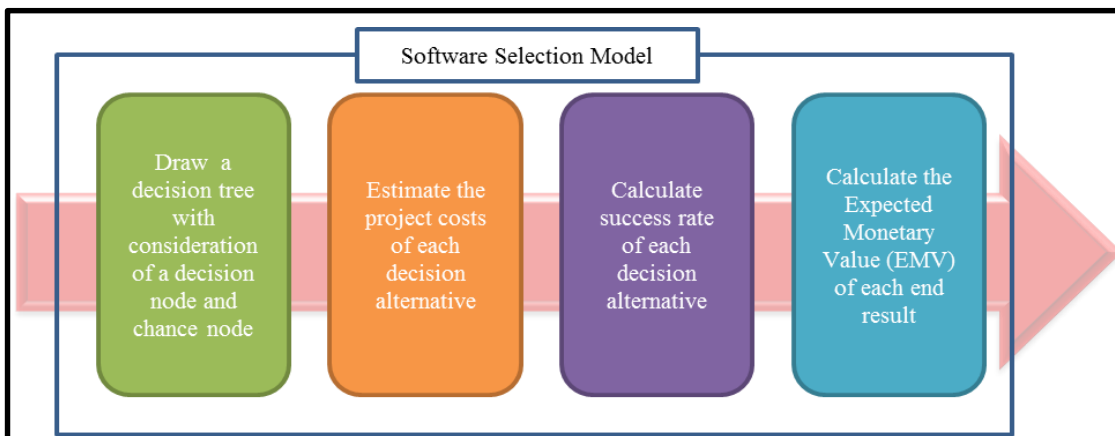
**Chapter 4 Research Methodology Demonstrations**

Based on the literature review, research models for the three research topic areas are depicted in Figure 8 through Figure 10. Note that illustrative examples are used to demonstrate the effectiveness of three research models.





**Figure 9 Research Model for Schedule Refinement / Verification**



**Figure 10 Research Model for Early-Stage Comparison of Software**

**Development Project Types**

#### **4.1 Schedule Estimation in the Early Stage (Proposal Preparation Stage)**

To demonstrate this research model, a fictitious large-sized inventory management software development project example is used; the demonstration will be explained step by step. There are seven assumptions in the first scenario for schedule estimation in the early stage, and these assumptions are made by our own discretion.

- The first assumption is that all weight factors in the Unadjusted Function Point Calculation are “simple” (versus “average” or “complex”; see Table 2).
- The second assumption is that three 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
- The seventh assumption is to use 25%, Mean, and 75% UFP value for calculating optimistic, most likely, and pessimistic effort and duration.

#### **4.1.1 WBS Creation for Project Scope**

According to the research model shown in Figure 8, 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 11. 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.

#### **4.1.2 Size Estimation**

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 according to Table 2 above (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. For example, if the number of each functional component of the first work package (Item Registration) is counted in terms of a 3-point estimation and based on the first assumption, it is calculated as shown in Table 18. The number displayed under each category for 3-point estimation is an illustrative example number.

Category	3-Point Estimation			Expected Count	Weight Factors			Sub Total
	OP	ML	Pess.		Simple	Average	Complex	
EI	7	10	13	$\{(1*7)+(4*10)+(1*13)\} / 6 = 10$	3	4	6	30
EO	1	2	3	$\{(1*1)+(4*2)+(1*3)\} / 6 = 2$	4	5	7	8
EQ	0	1	2	$\{(1*0)+(4*1)+(1*2)\} / 6 = 1$	3	4	6	3
ILF	1	2	3	$\{(1*1)+(4*2)+(1*3)\} / 6 = 2$	7	10	15	14
EIF	0	0	0	$\{(1*0)+(4*0)+(1*0)\} / 6 = 0$	5	7	10	0
<b>Grand Total (UFP)</b>								<b>55</b>

**Table 18 The Expected Size Estimation of the First Work Package**

The @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 simulated UFP results of all 30 work packs are shown in Figure 12 through Figure 38. The summary of simulation results for size estimation is shown in Table 19, and the value determined at the 75% probability range (see the “75% UFP” column in Table 19) is used for calculating the resource efforts of each work package. Note that this 75% UFP is only used for calculating pessimistic effort and duration of each work package, and the detailed explanation about using 25%, mean, and 75% UFP will be discussed in the Section 4.1.5.

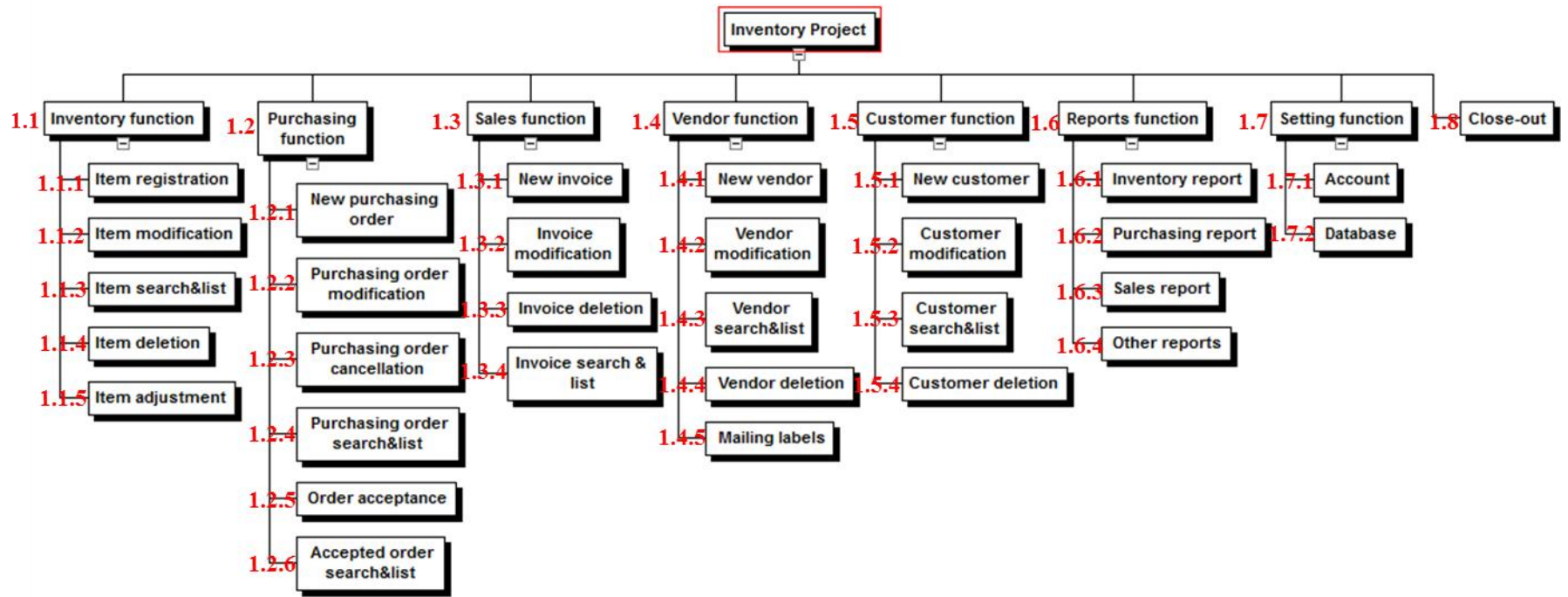


Figure 11 Inventory Project WBS

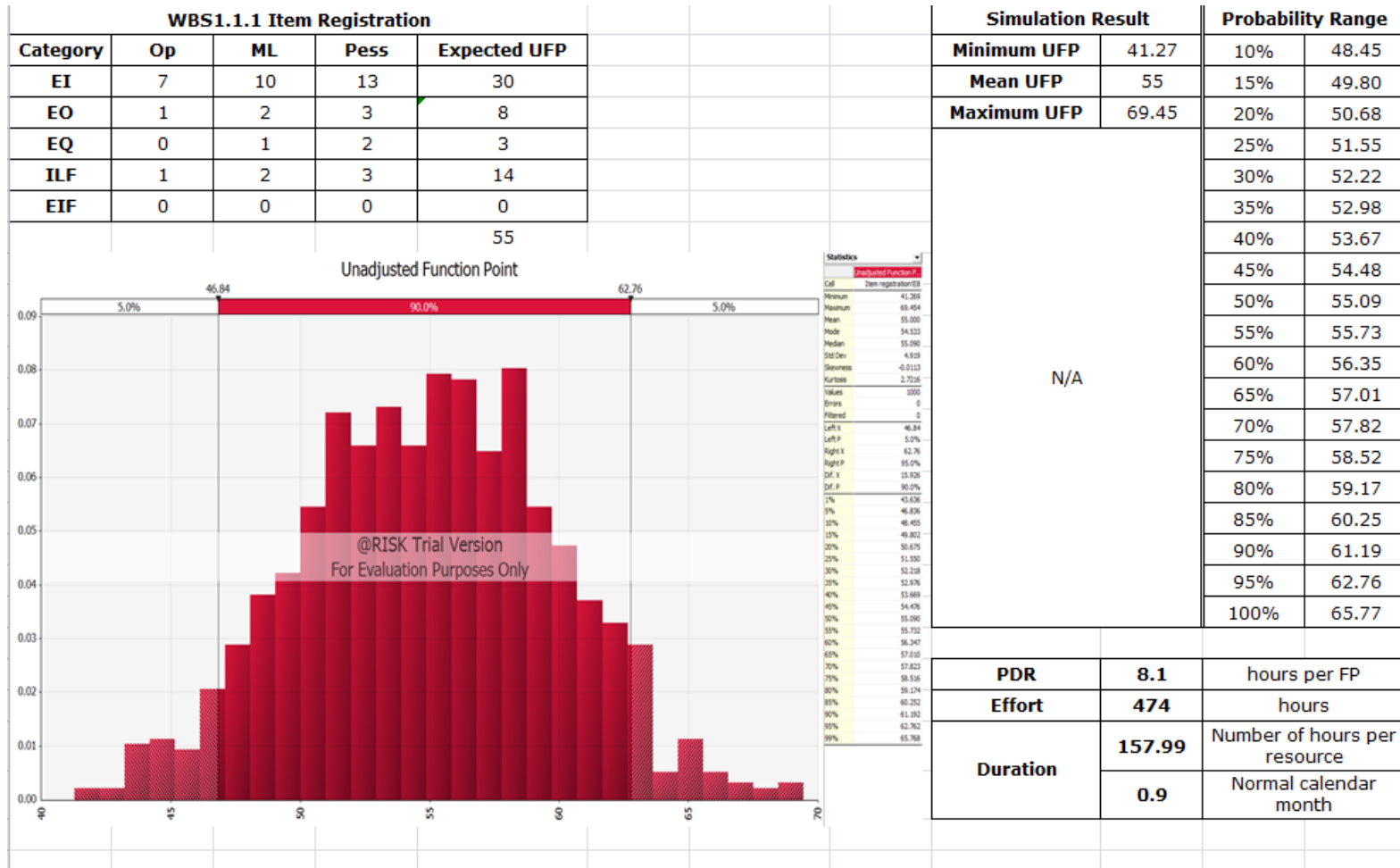


Figure 12 WBS 1.1.1 Item Registration UFP

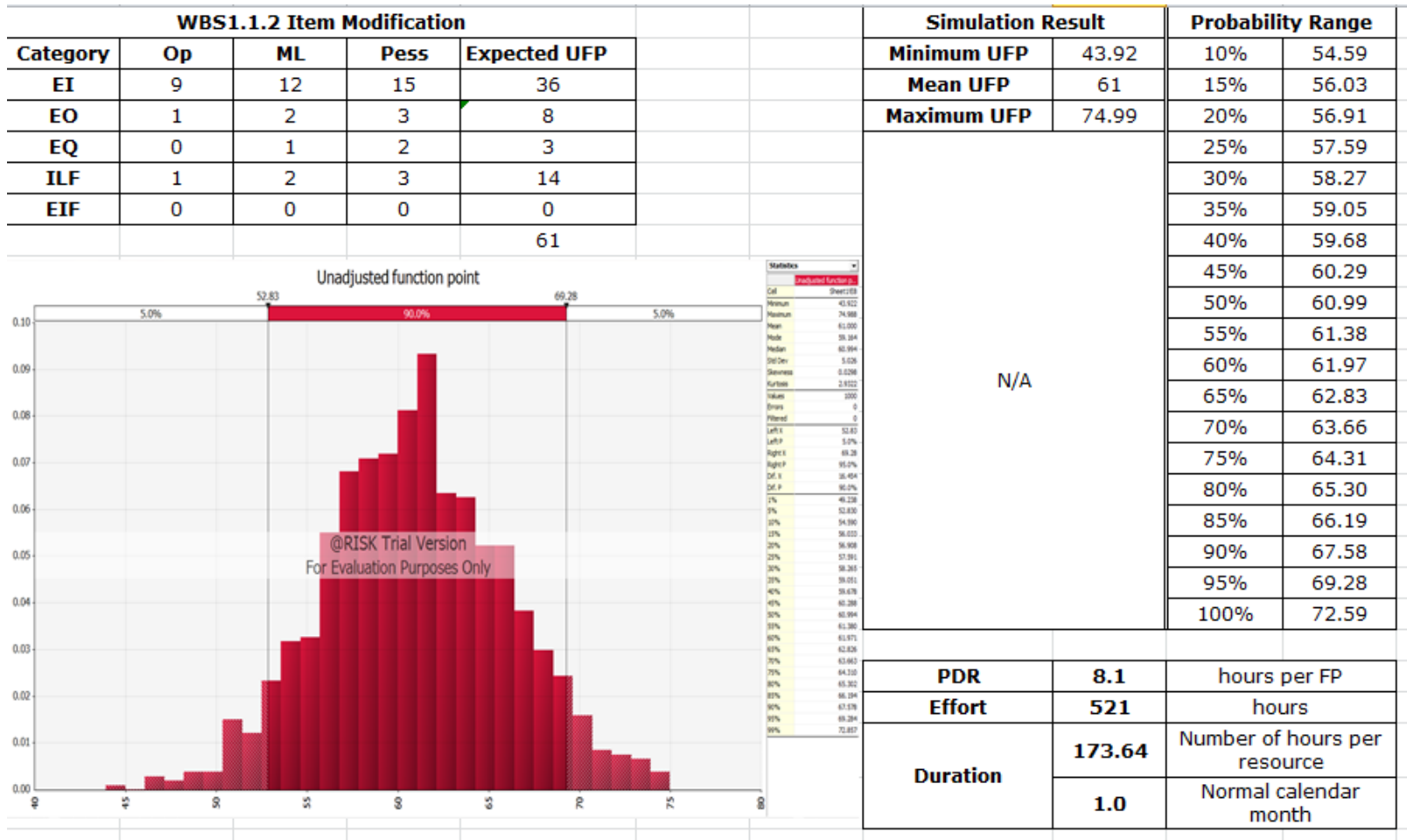


Figure 13 WBS 1.1.2 Item Modification UFP



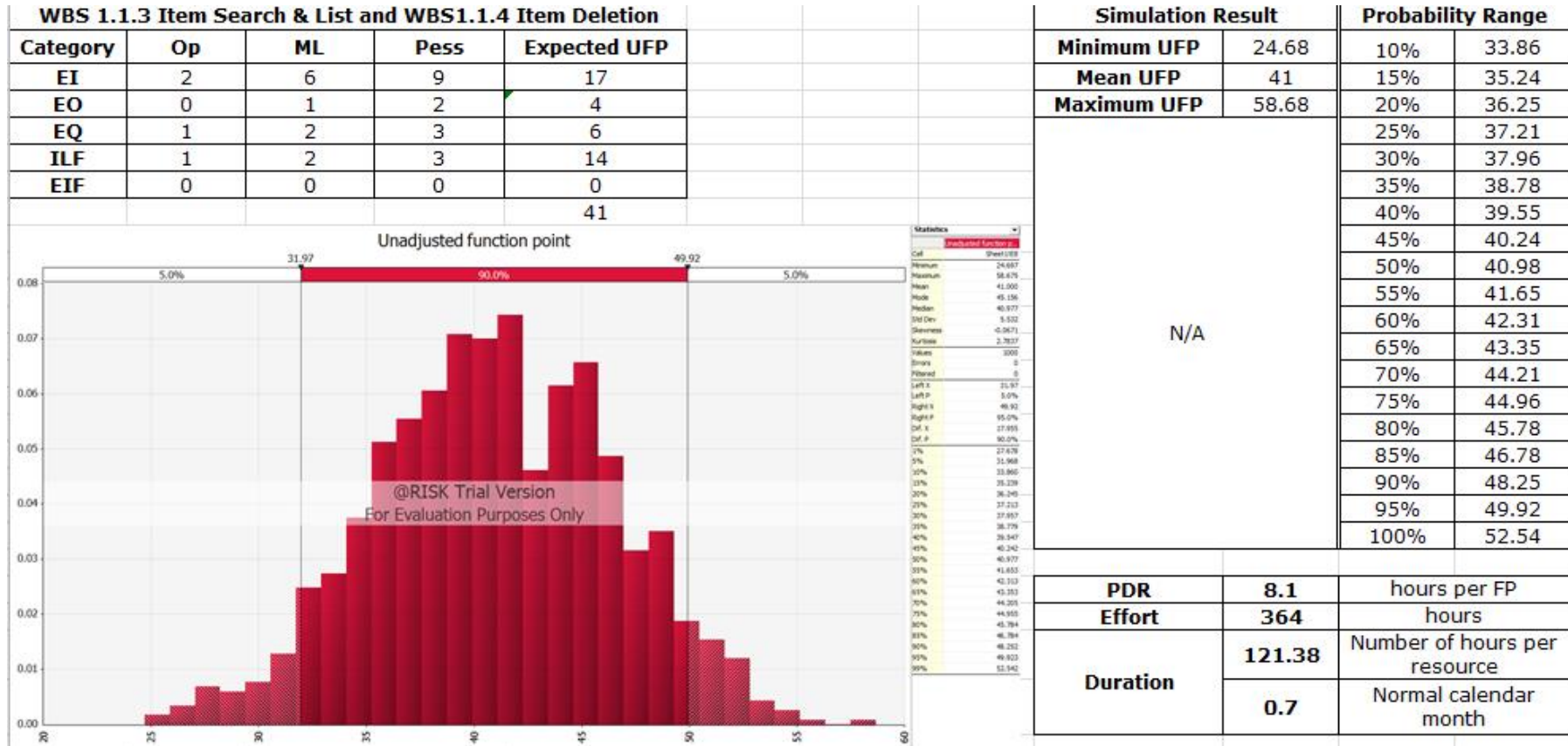


Figure 14 WBS 1.1.3 Item Search & List UFP and WBS 1.1.4 Item Deletion UFP

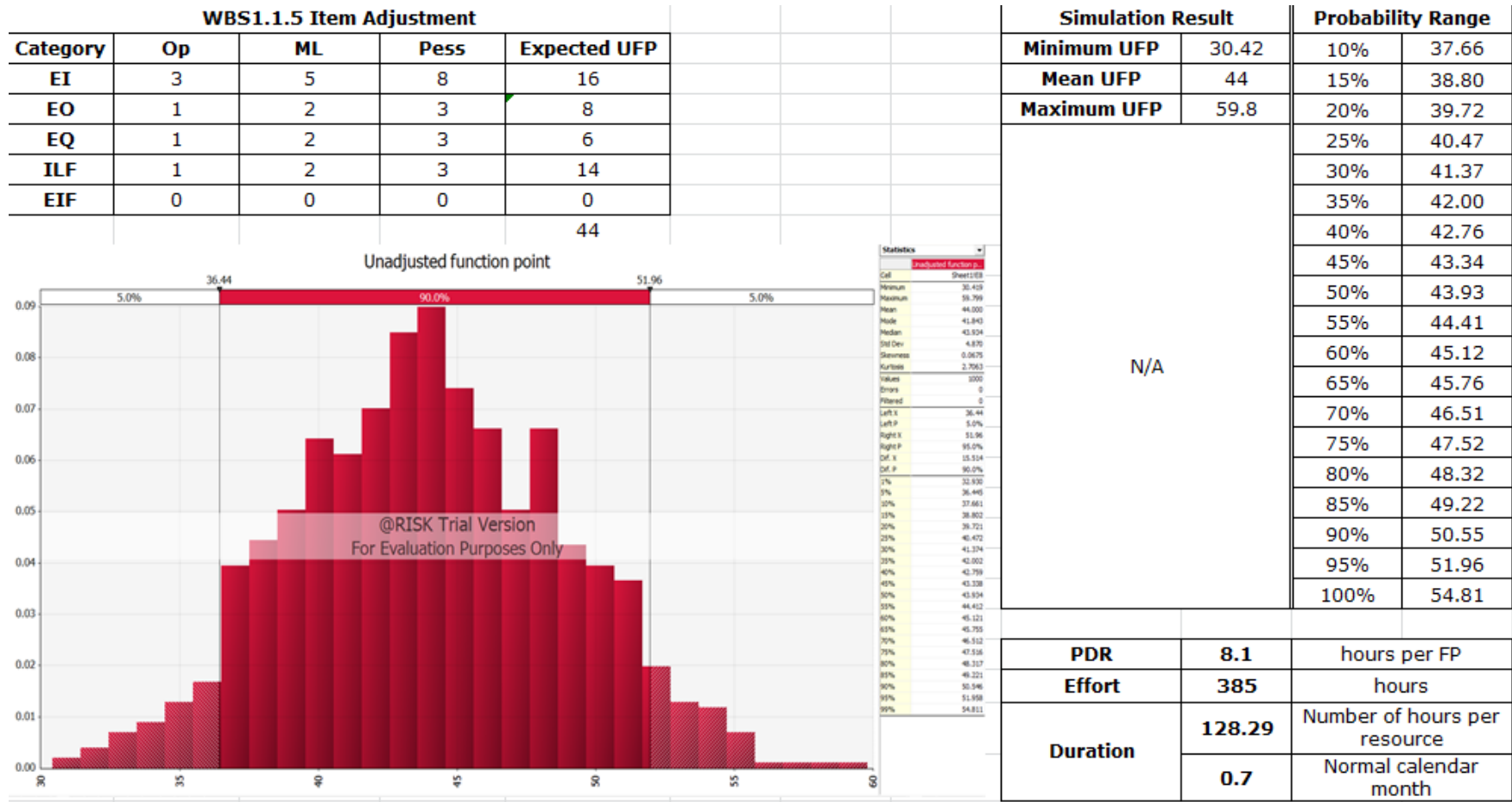


Figure 15 WBS 1.1.5 Item Adjustment UFP

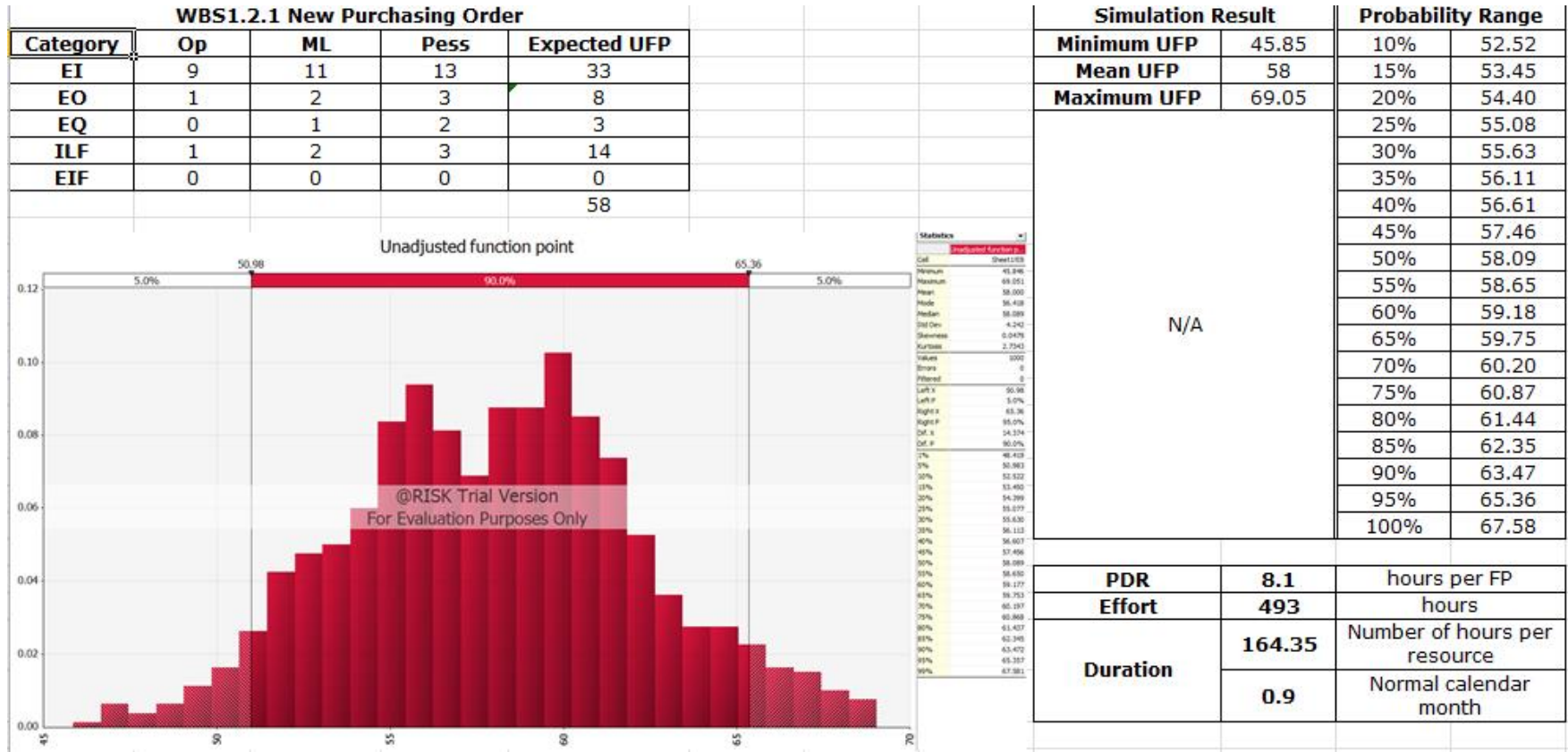


Figure 16 WBS 1.2.1 New Purchasing Order UFP

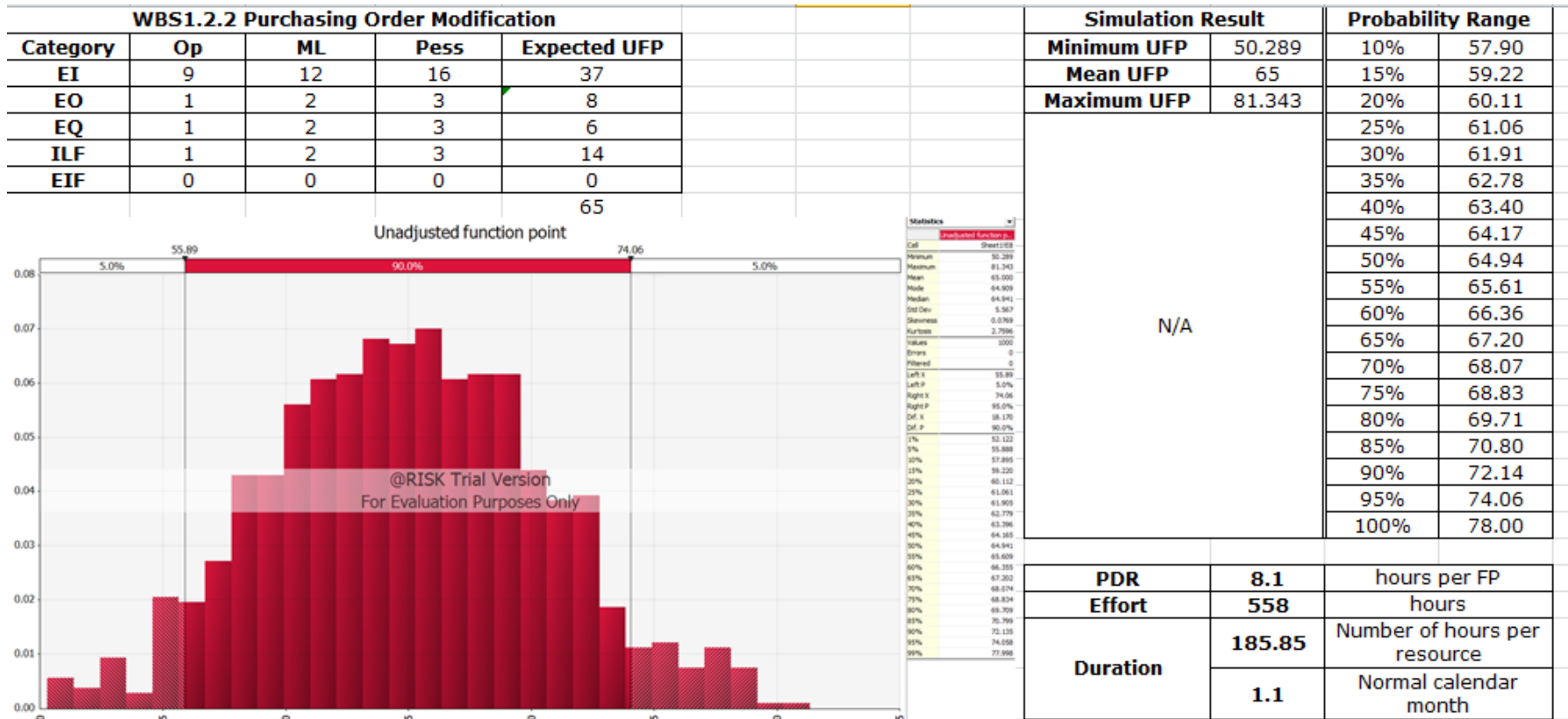
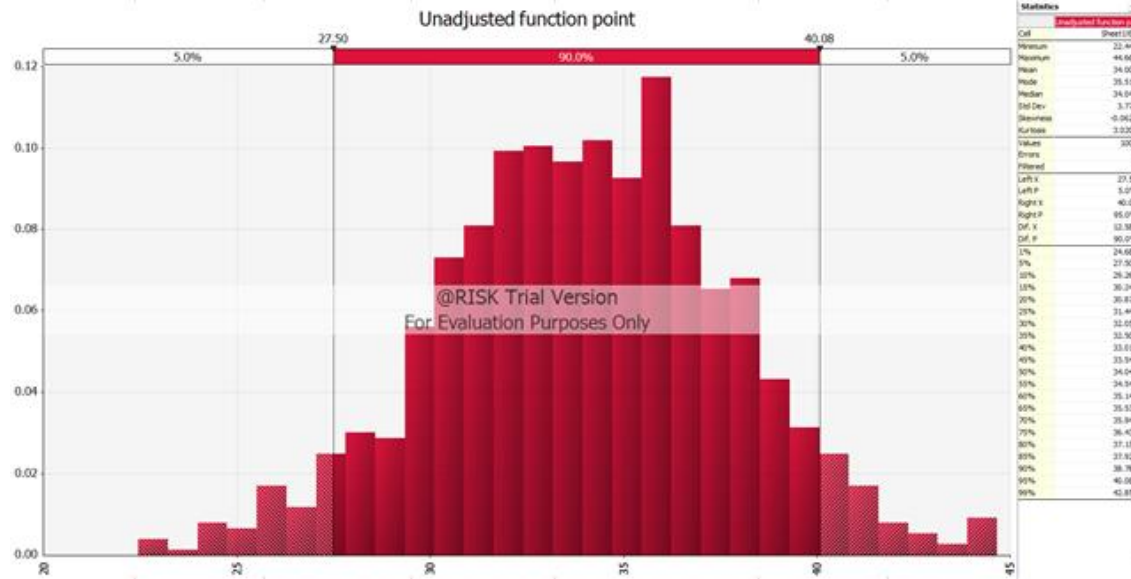


Figure 17 WBS 1.2.2 Purchasing Order Modification UFP

WBS1.2.3 Purchasing Order Cancellation				
Category	Op	ML	Pess	Expected UFP
EI	1	2	3	6
EO	1	2	3	8
EQ	1	2	3	6
ILF	1	2	3	14
EIF	0	0	0	0
				34

Simulation Result		Probability range	
Minimum UFP	22.45	10%	29.27
Mean UFP	34	15%	30.24
Maximum UFP	35.51	20%	30.87
N/A		25%	31.44
		30%	32.05
		35%	32.50
		40%	33.02
		45%	33.55
		50%	34.04
		55%	34.55
		60%	35.14
		65%	35.53
		70%	35.94
75%	36.43		
80%	37.16		
85%	37.93		
90%	38.79		
95%	40.08		
100%	42.85		

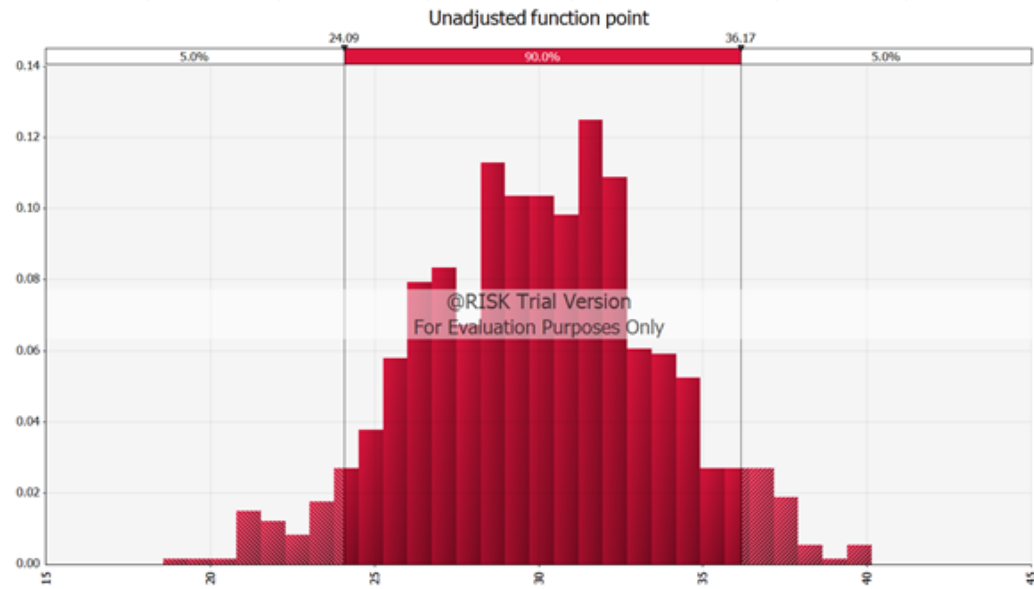


<b>PDR</b>	<b>8.1</b>	hours per FP
<b>Effort</b>	<b>295</b>	hours
<b>Duration</b>	<b>98.37</b>	Number of hours per resource
	<b>0.6</b>	Normal calendar month

Figure 18 WBS 1.2.3 Purchasing Order Cancellation UFP

WBS1.2.4 Purchasing Order Search & List, WBS1.2.5 Order Acceptance, and WBS1.2.6 Accepted Order Search & List				
Category	Op	ML	Pess	Expected UFP
EI	1	2	3	6
EO	0	1	2	4
EQ	1	2	3	6
ILF	1	2	3	14
EIF	0	0	0	0

30



Statistics	
Stat	Value
Cell	Sheet1!C12
Minimum	18.57
Maximum	40.14
Mean	30.00
Mode	28.688
Median	30.004
Std Dev	3.600
Skewness	-0.9620
Kurtosis	2.8872
Values	1000
Errors	0
Filtered	0
Left X	24.09
Left P	5.0%
Right X	36.17
Right P	95.0%
Diff. X	12.081
Diff. P	90.0%
1%	21.193
5%	24.089
10%	25.437
15%	26.289
20%	26.868
25%	27.474
30%	28.169
35%	28.694
40%	29.170
45%	29.574
50%	30.004
55%	30.574
60%	31.065
65%	31.536
70%	31.945
75%	32.318
80%	32.620
85%	32.729
90%	32.864
95%	33.170
99%	37.894

Simulation Result		Probability range	
Minimum UFP	18.57	10%	25.44
Mean UFP	30	15%	26.17
Maximum UFP	40.14	20%	26.87
N/A		25%	27.47
		30%	28.17
		35%	28.69
		40%	29.14
		45%	29.57
		50%	30.00
		55%	30.57
		60%	31.07
		65%	31.54
		70%	31.95
75%	32.32		
80%	32.92		
85%	33.73		
90%	34.56		
95%	36.17		
100%	37.89		
PDR	8.1	hours per FP	
Effort	262	hours	
Duration	87.26	Number of hours per resource	
	0.5	Normal calendar month	

Figure 19 WBS 1.2.4 Purchasing Order Search & List, WBS 1.2.5 Order Acceptance, and WBS 1.2.6 Accepted Order Search & List UFP

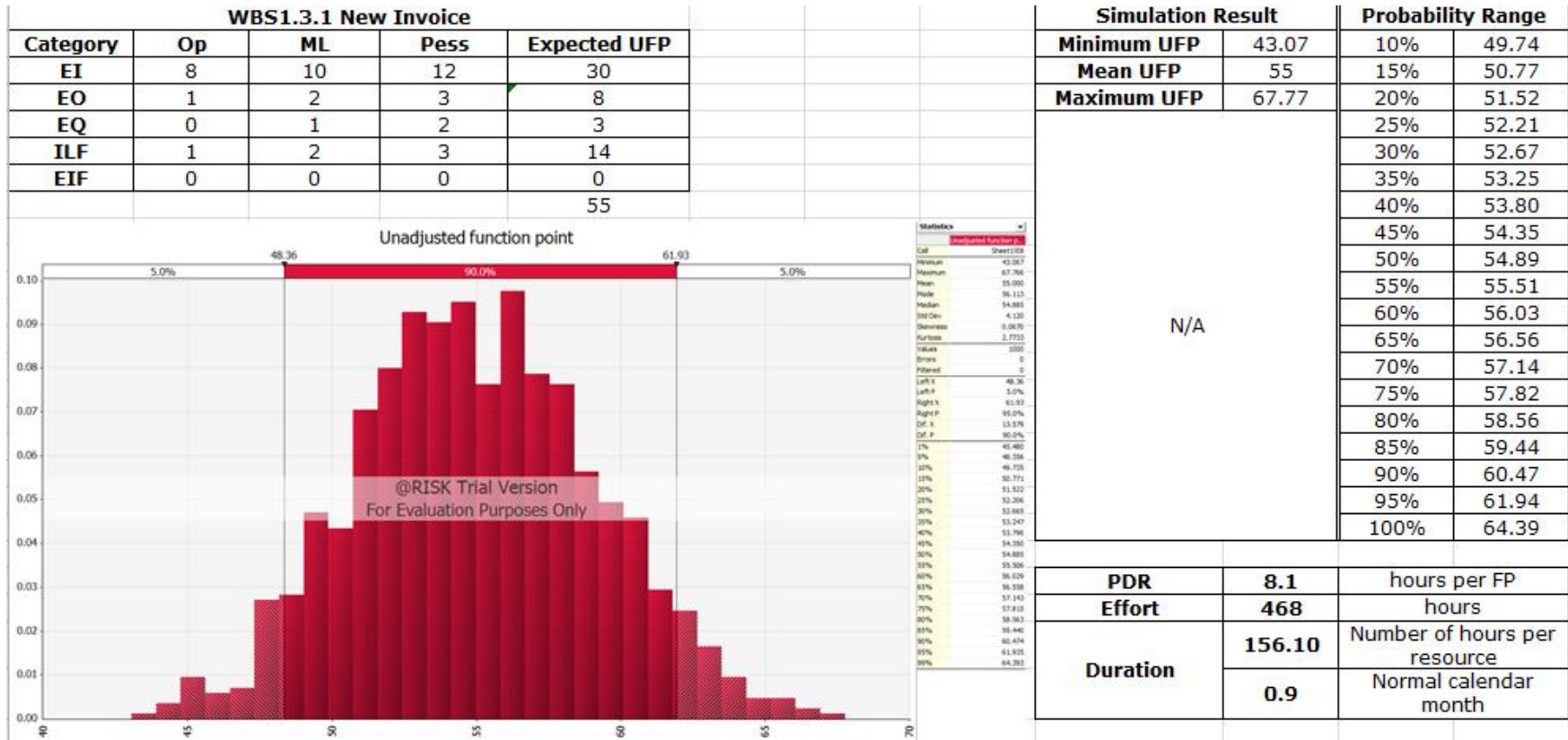


Figure 20 WBS 1.3.1 New Invoice UFP

<b>PDR</b>	<b>8.1</b>	hours per FP
<b>Effort</b>	<b>468</b>	hours
<b>Duration</b>	<b>156.10</b>	Number of hours per resource
	<b>0.9</b>	Normal calendar month

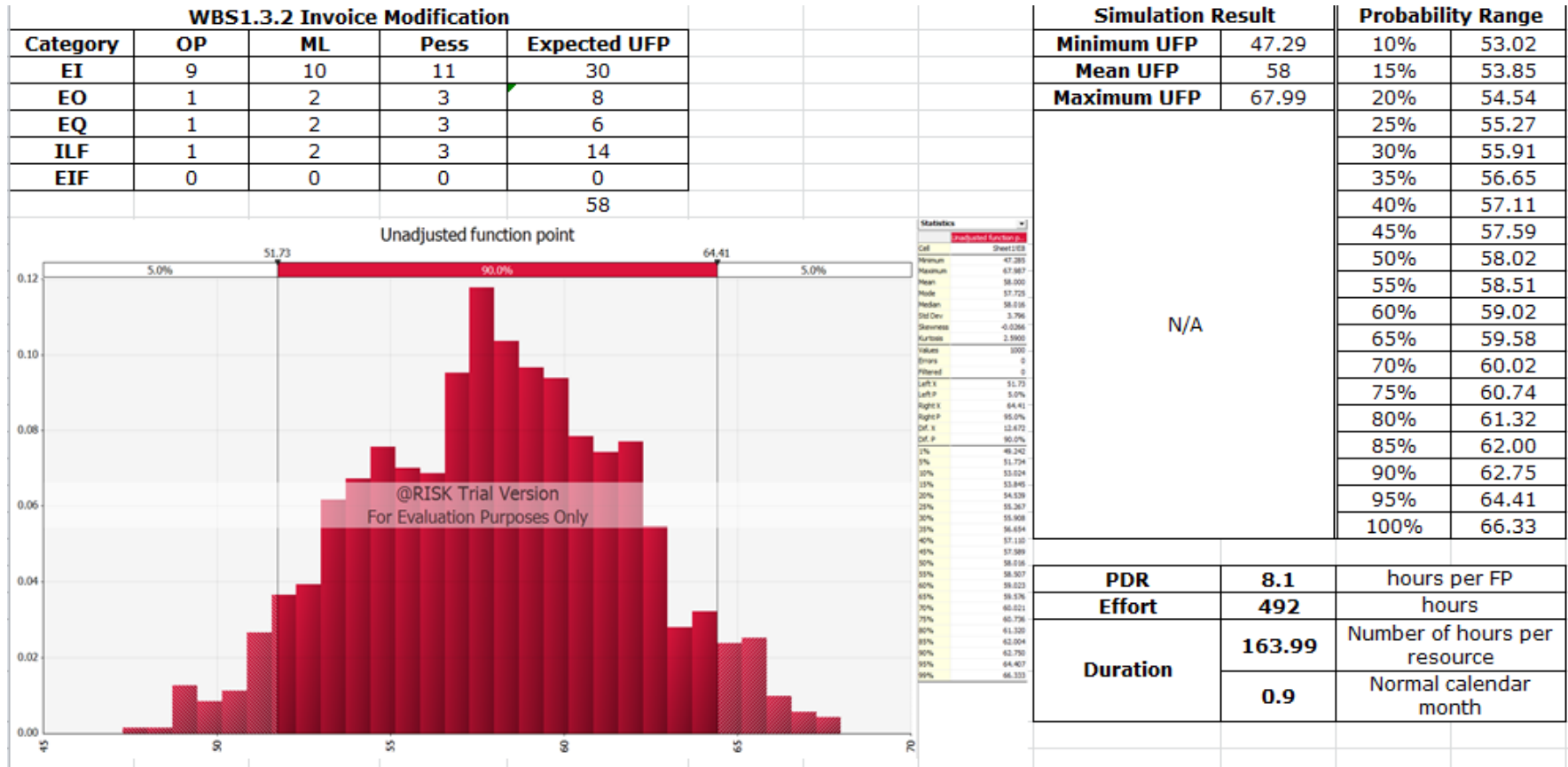
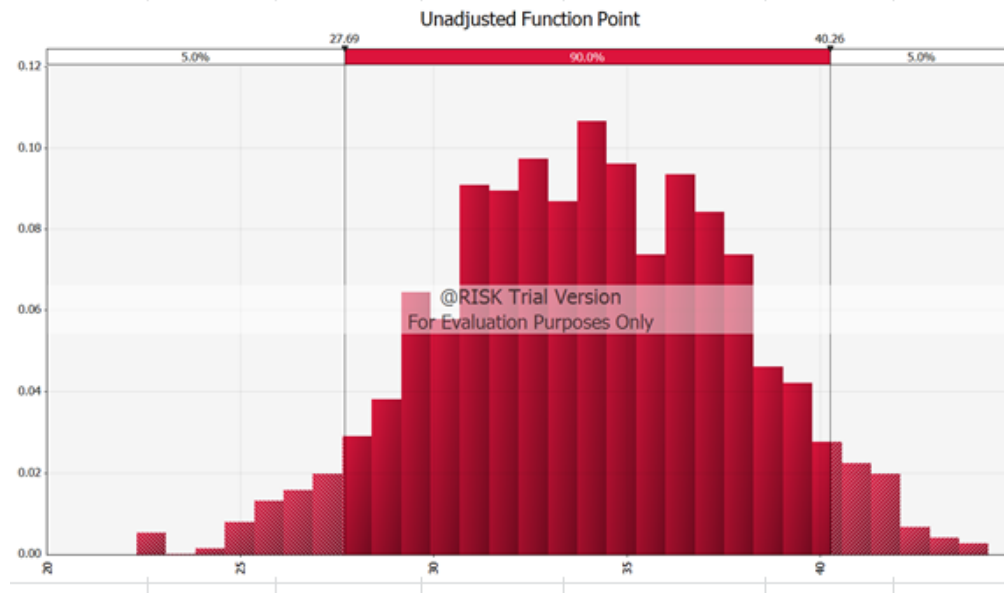


Figure 21 WBS 1.3.2 Invoice Modification UFP



WBS1.3.3 Invoice Deletion				
Category	Op	ML	Pess	Expected UFP
EI	1	2	3	6
EO	1	2	3	8
EQ	1	2	3	6
ILF	1	2	3	14
EIF	0	0	0	0
				34



Statistics	
Cell	Sheet1!\$B\$2
Count	22,338
Minimum	44,366
Maximum	34,000
Mean	33,242
Mode	33,955
Median	33,955
Std Dev	3,808
Skewness	-0,9603
Kurtosis	2,7122
Values	1000
Errors	0
Filtered	0
Left T	27,69
Left P	5,0%
Right T	40,26
Right P	95,0%
Diff. T	12,568
Diff. P	90,0%
1%	28,190
5%	27,691
10%	28,190
15%	30,022
20%	30,754
25%	31,273
30%	31,863
35%	32,422
40%	32,925
45%	33,366
50%	33,955
55%	34,472
60%	35,000
65%	35,646
70%	36,184
75%	36,745
80%	37,414
85%	37,972
90%	38,881
95%	40,260
99%	42,048

Simulation Result		Probability Range	
Minimum UFP	22.32	10%	29.16
Mean UFP	34	15%	30.02
Maximum UFP	44.37	20%	30.75
N/A		25%	31.27
		30%	31.86
		35%	32.43
		40%	32.93
		45%	33.37
		50%	33.96
		55%	34.47
		60%	35.00
		65%	35.65
		70%	36.18
75%	36.74		
80%	37.41		
85%	37.97		
90%	38.88		
95%	40.26		
100%	42.05		
<b>PDR</b>	<b>8.1</b>	hours per FP	
<b>Effort</b>	<b>298</b>	hours	
<b>Duration</b>	<b>99.21</b>	Number of hours per resource	
	<b>0.6</b>	Normal calendar month	

Figure 22 WBS 1.3.3 Invoice Deletion UFP

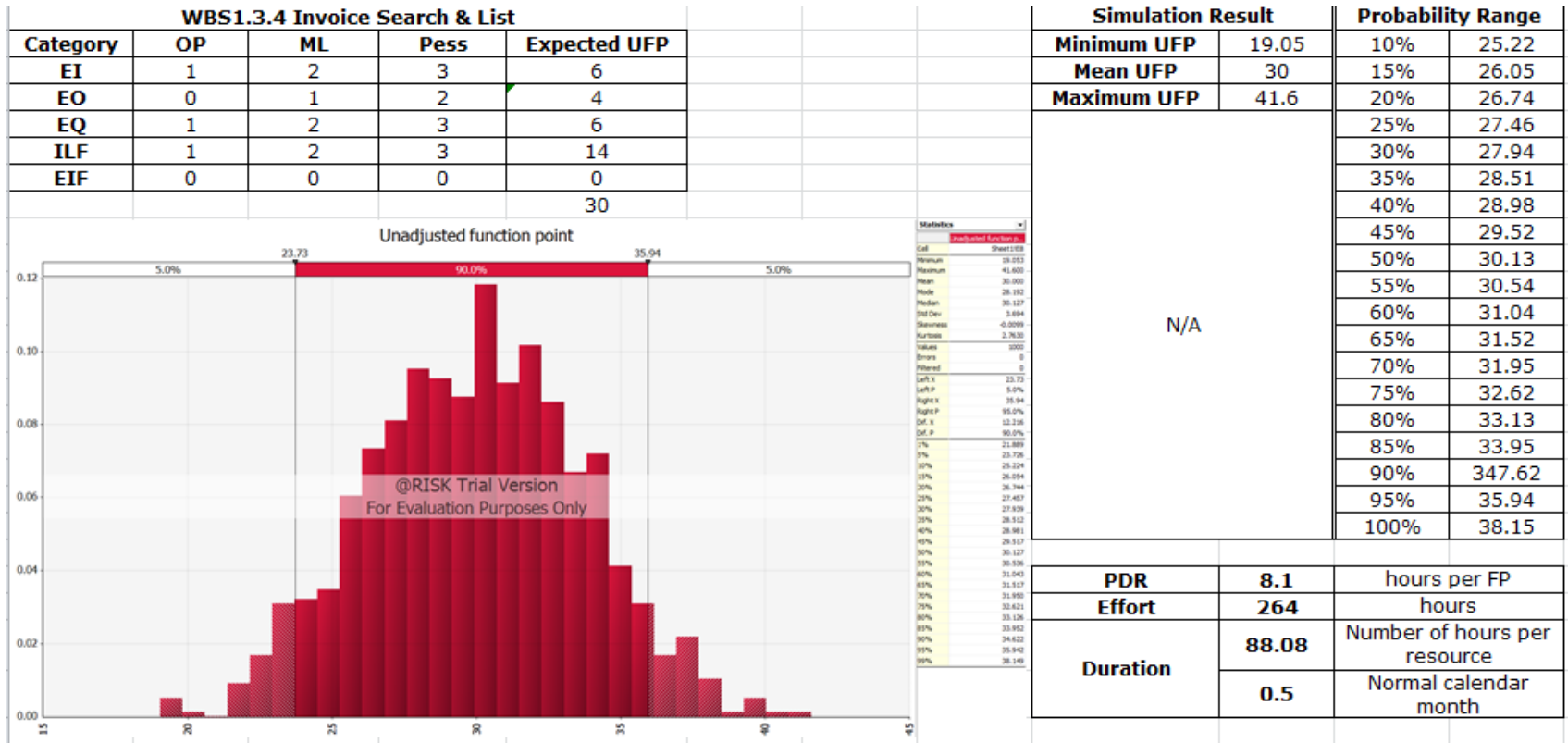


Figure 23 WBS 1.3.4 Invoice Search & List UFP



Figure 24 WBS 1.4.1 New Vendor UFP

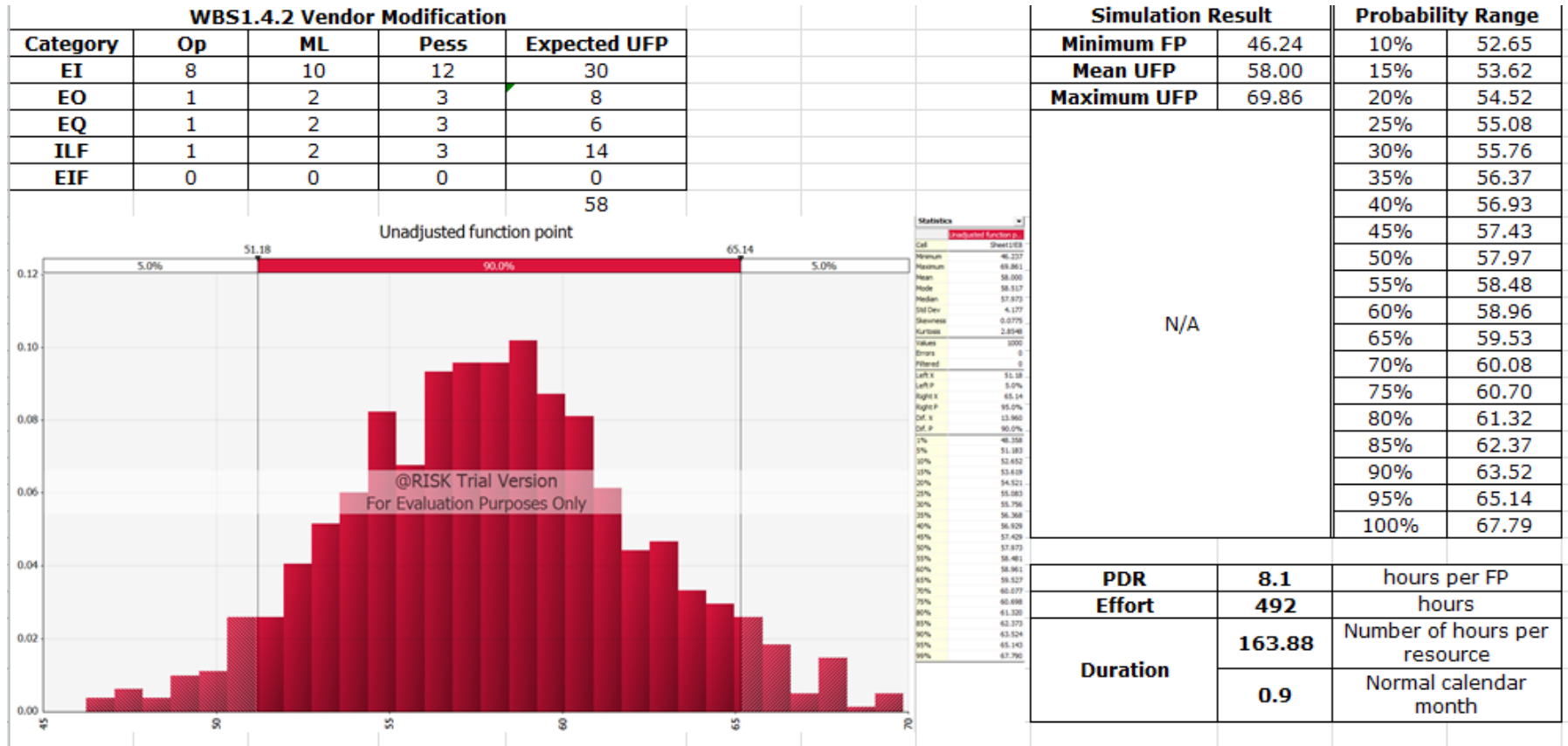


Figure 25 WBS 1.4.2 Vendor Modification UFP

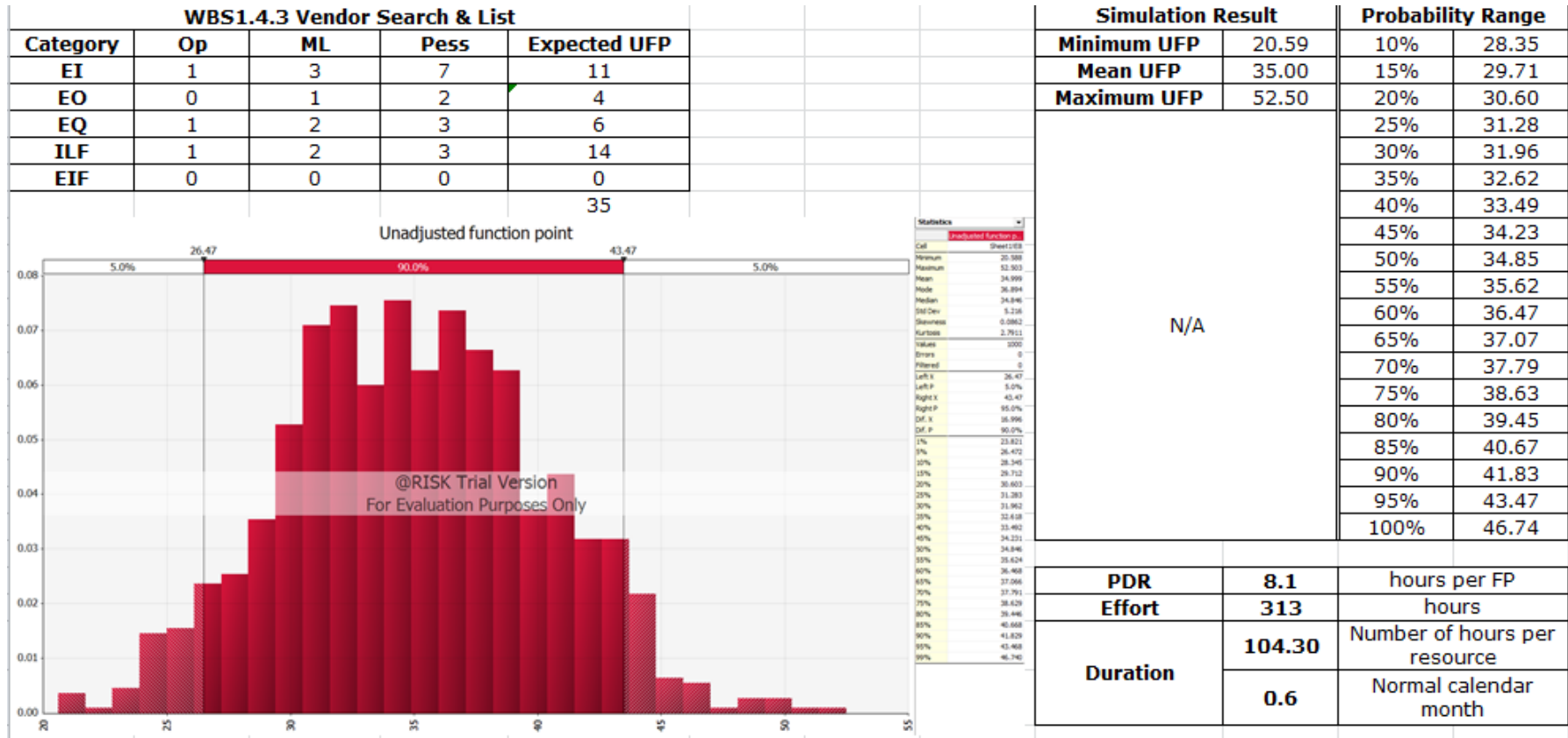


Figure 26 WBS 1.4.3 Vendor Search & List UFP

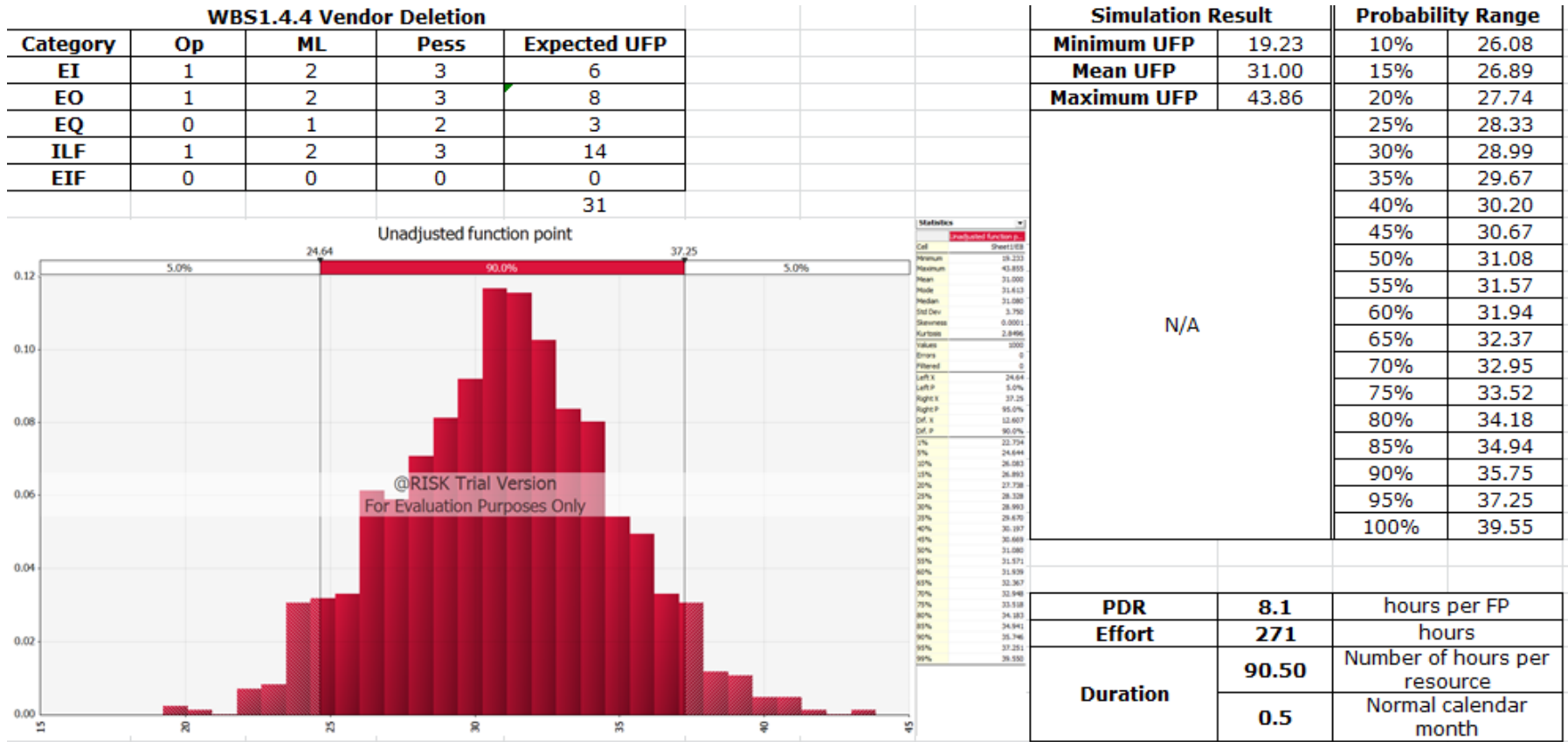


Figure 27 WBS 1.4.4 Vendor Deletion UFP

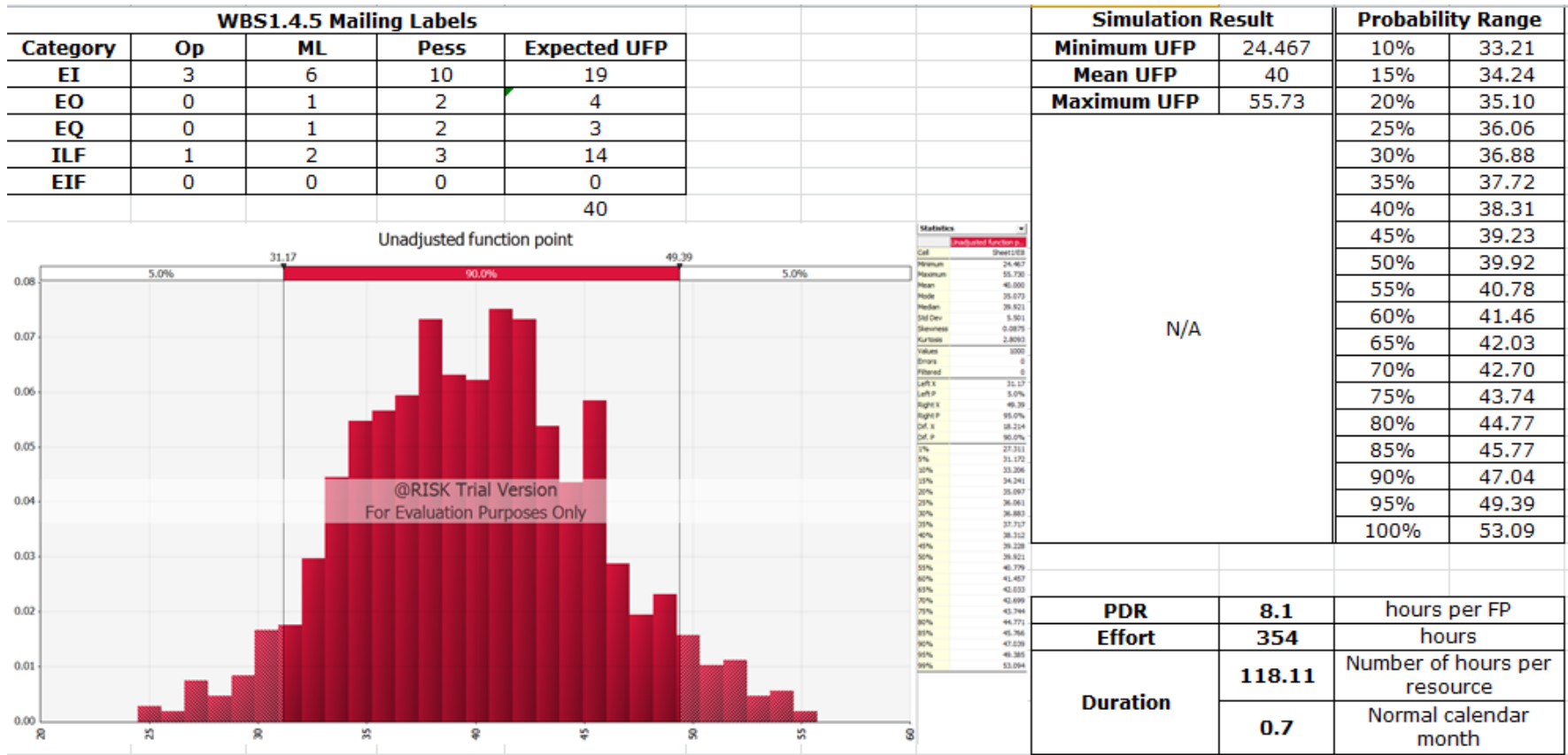


Figure 28 WBS 1.4.5 Mailing Labels UFP

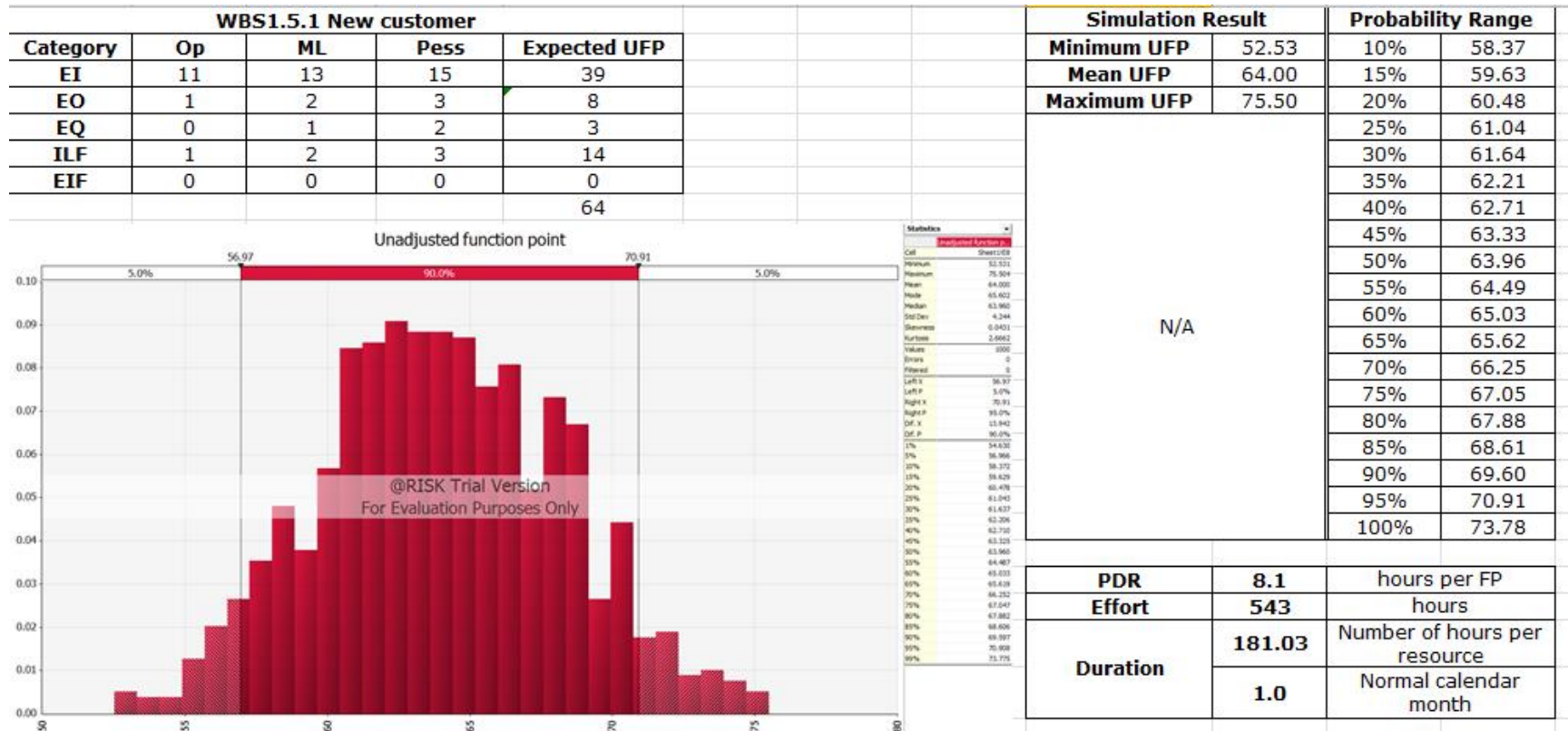


Figure 29 WBS 1.5.1 New Customer UFP



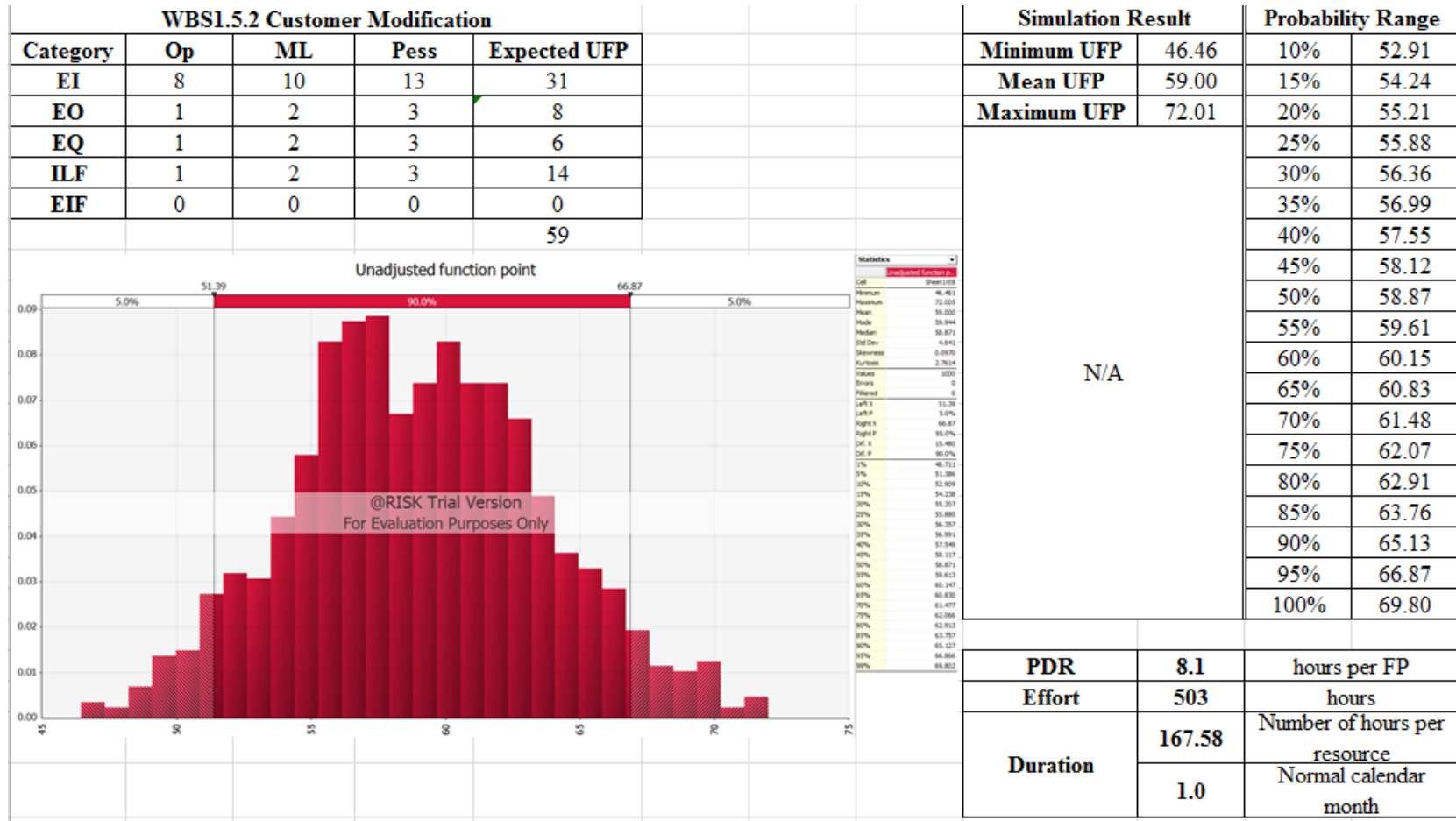


Figure 30 WBS 1.5.2 Customer Modification UFP

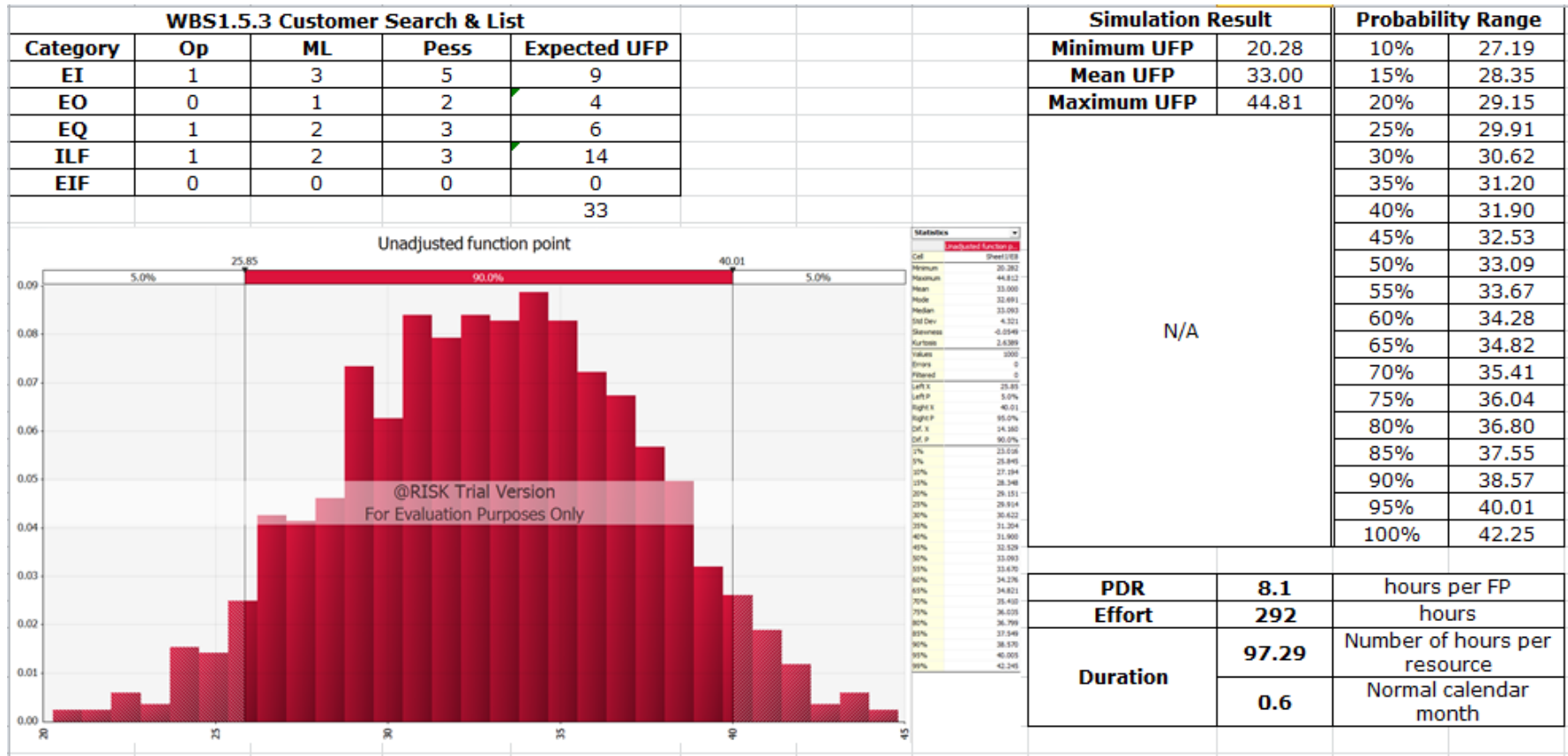


Figure 31 WBS 1.5.3 Customer Search and List UFP

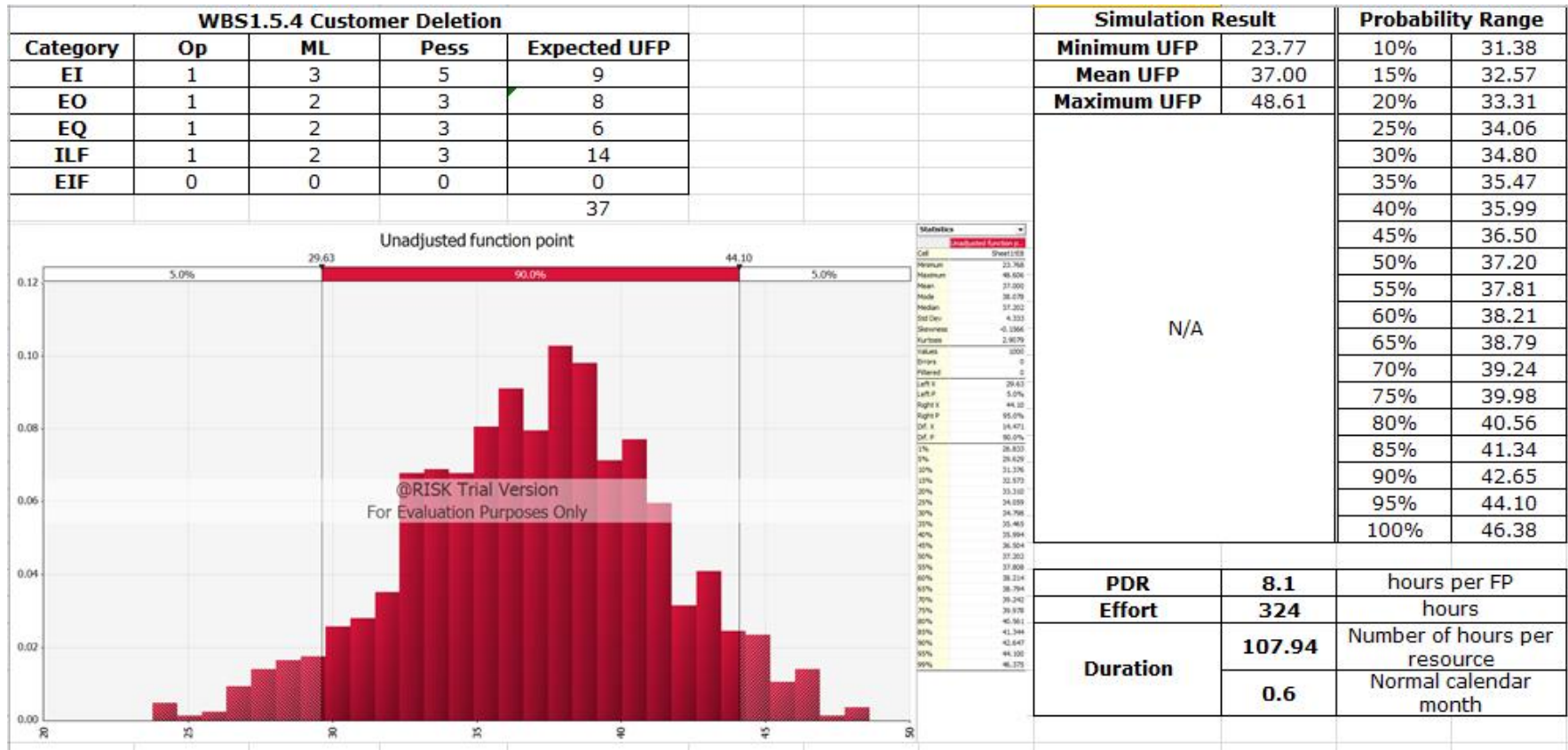
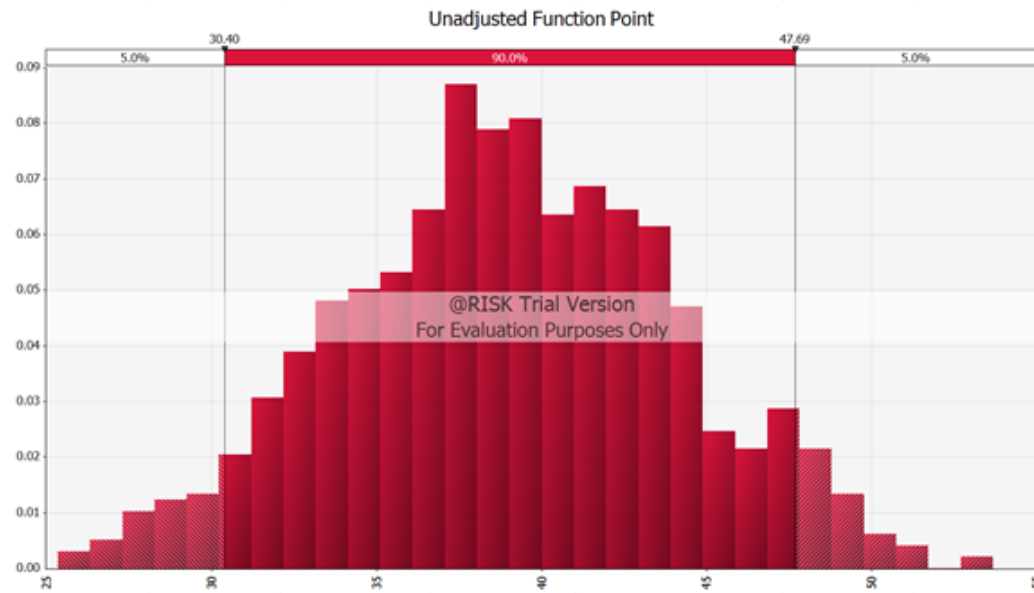


Figure 32 WBS 1.5.4 Customer Deletion UFP

WBS1.6.1 Inventory Report				
Category	Op	ML	Pess	Expected UFP
EI	2	5	8	15
EO	0	1	2	4
EQ	1	2	3	6
ILF	1	2	3	14
EIF	0	0	0	0
				39



Statistics	
Property	Value
Call	Sheet1!\$E\$7
Minimum	25.35
Maximum	53.68
Mean	39.00
Mode	38.400
Median	38.893
Std Dev	5.070
Skewness	0.0023
Kurtosis	2.8998
Values	1000
Errors	0
Filtered	0
Left X	30.40
Left P	5.0%
Right X	47.69
Right P	95.0%
Dist. X	17.288
Dist. P	90.0%
1%	27.757
5%	30.400
10%	32.348
15%	33.952
20%	34.930
25%	35.587
30%	36.111
35%	36.580
40%	37.064
45%	37.529
50%	38.000
55%	38.557
60%	40.268
65%	41.073
70%	41.679
75%	42.524
80%	43.281
85%	44.153
90%	45.679
95%	47.688
99%	50.118

Simulation Result		Probability Range	
Minimum UFP	25.35	10%	32.35
Mean UFP	39.00	15%	33.55
Maximum UFP	53.68	20%	34.43
N/A		25%	35.59
		30%	36.41
		35%	37.18
		40%	37.76
		45%	38.34
		50%	38.89
		55%	39.56
		60%	40.27
		65%	41.07
		70%	41.68
75%	42.52		
80%	43.28		
85%	44.15		
90%	45.68		
95%	47.69		
100%	50.12		
<b>PDR</b>	<b>8.1</b>	hours per FP	
<b>Effort</b>	<b>344</b>	hours	
<b>Duration</b>	<b>114.81</b>	Number of hours per resource	
	<b>0.7</b>	Normal calendar month	

Figure 33 WBS 1.6.1 Inventory Report UFP

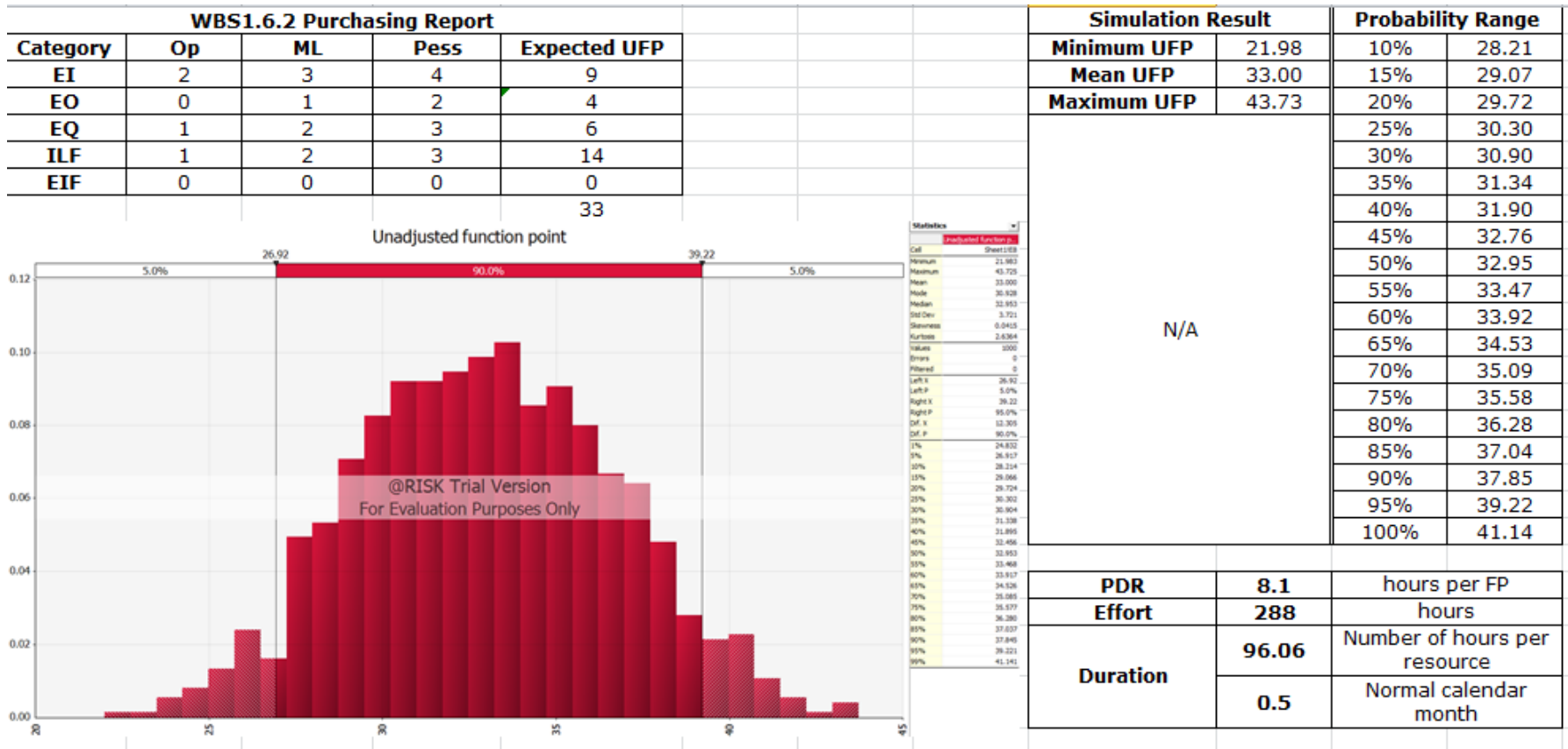


Figure 34 WBS 1.6.2 Purchasing Report UFP

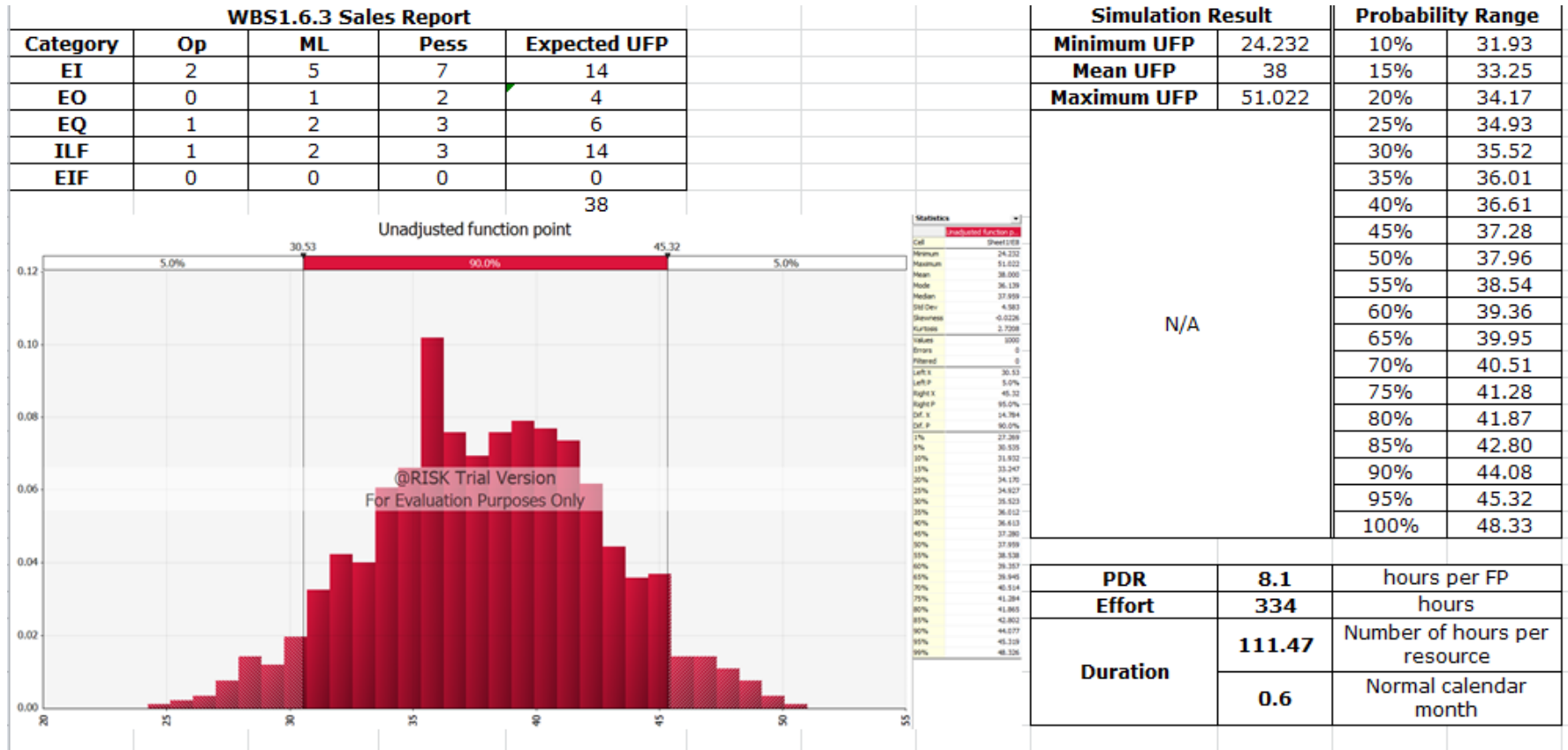


Figure 35 WBS 1.6.3 Sales Report UFP

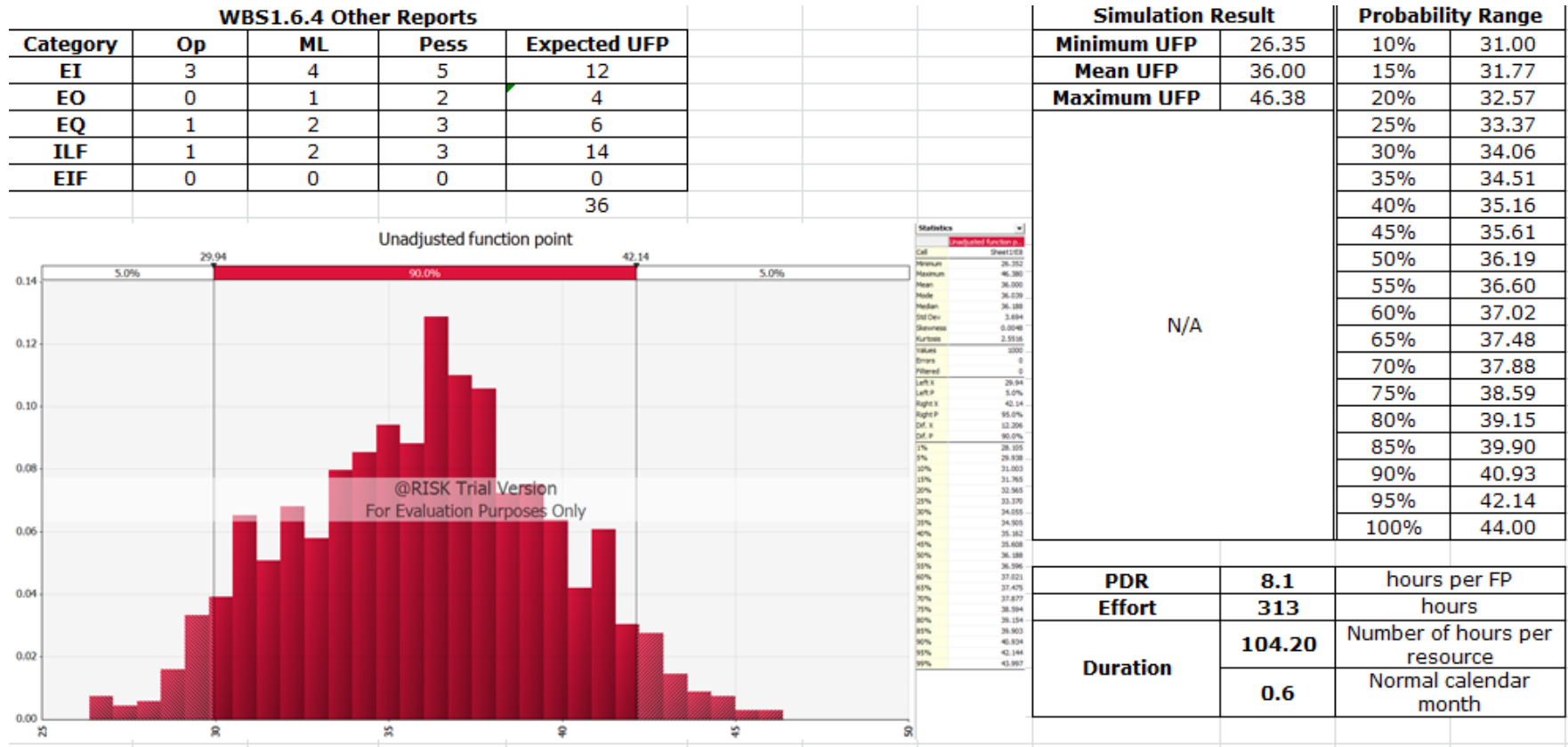


Figure 36 WBS 1.6.4 Other Reports UFP

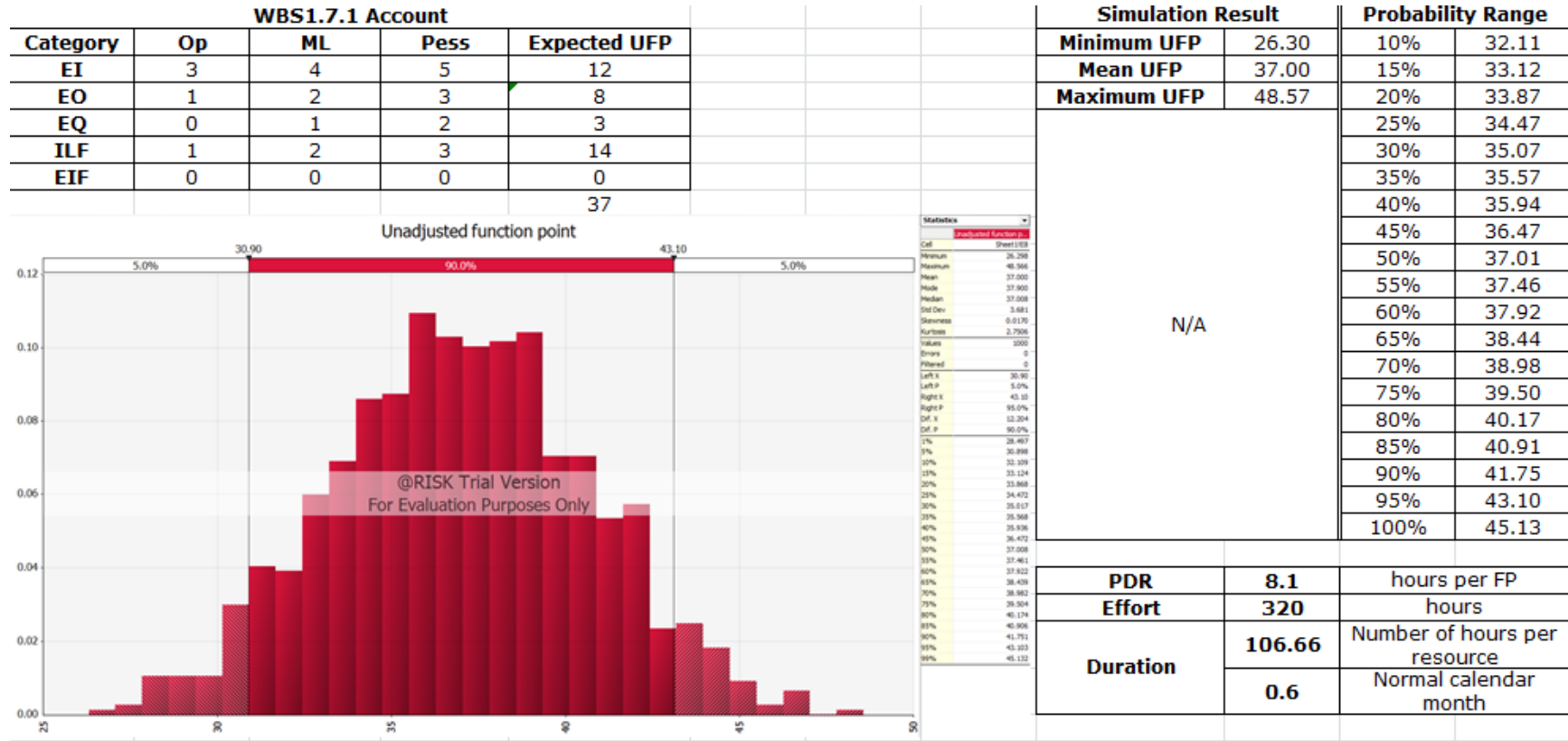


Figure 37 WBS 1.7.1 Account UFP



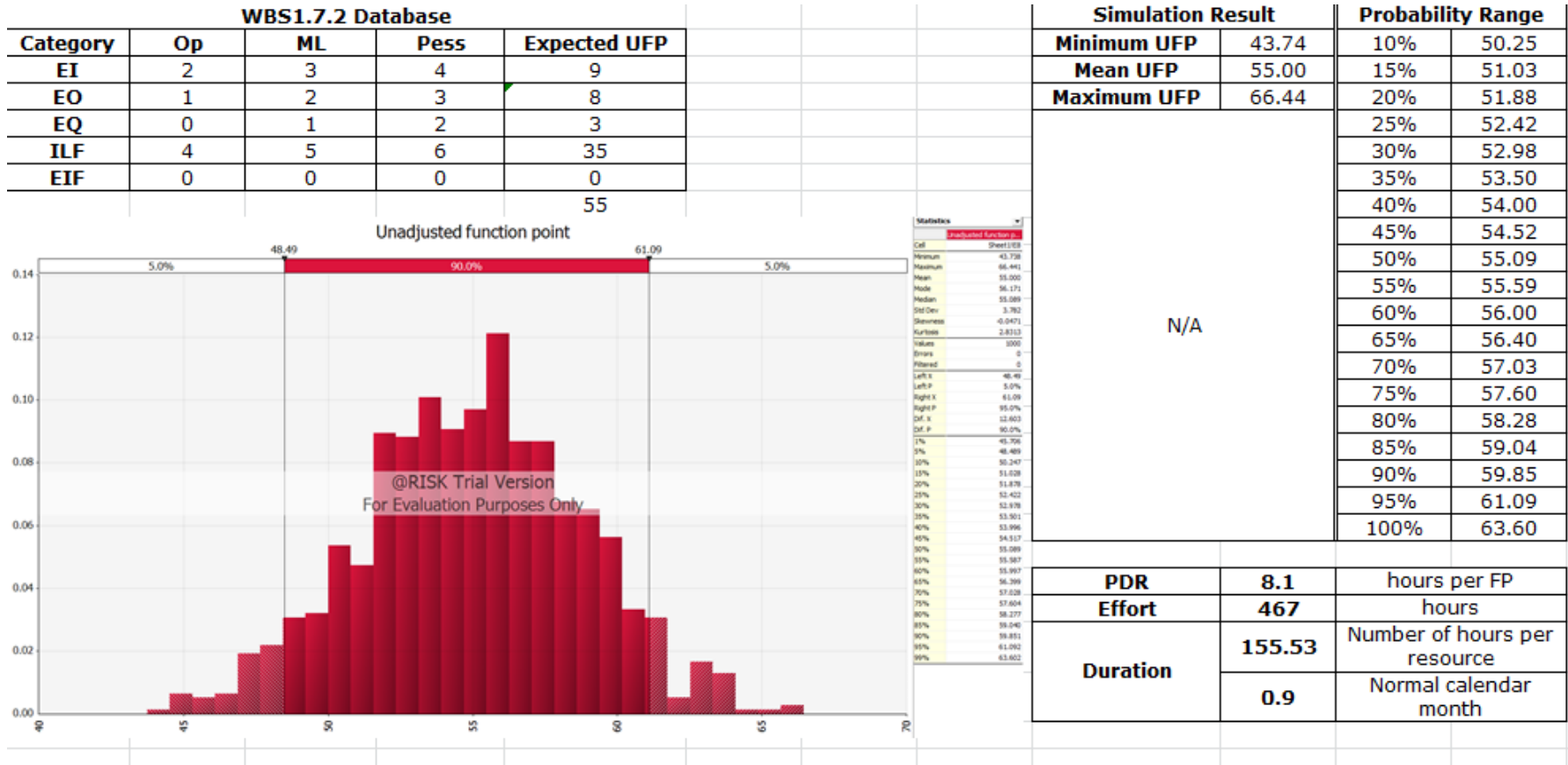


Figure 38 WBS 1.7.2 Database UFP

WP	Min UFP	25% UFP	Mean UFP	75% UFP	Max UFP
1.1.1	41.27	51.55	55	58.52	69.45
1.1.2	43.92	57.59	61	64.31	74.99
1.1.3	24.68	37.21	41	44.96	58.68
1.1.4	24.68	37.21	41	44.96	58.68
1.1.5	30.42	40.47	44	47.52	59.8
1.2.1	45.85	55.08	58	60.87	69.05
1.2.2	50.29	61.06	65	68.83	81.34
1.2.3	22.45	31.44	34	36.43	58.11
1.2.4	18.57	27.47	30	32.32	40.14
1.2.5	18.57	27.47	30	32.32	40.14
1.2.6	18.57	27.47	30	32.32	40.14
1.3.1	43.07	52.21	55	57.82	67.77
1.3.2	47.29	55.27	58	60.74	67.99
1.3.3	22.32	31.27	34	36.74	44.37
1.3.4	19.05	27.46	30	32.62	41.60
1.4.1	51.65	61.15	64	66.70	77.46
1.4.2	46.24	55.08	58	60.70	69.86
1.4.3	20.59	31.28	35	38.63	52.50
1.4.4	19.23	28.33	31	33.52	43.86
1.4.5	24.47	36.06	40	43.74	55.73
1.5.1	52.53	61.04	64	67.05	75.50
1.5.2	46.46	55.88	59	62.07	72.01
1.5.3	20.28	29.91	33	36.04	44.81
1.5.4	23.77	34.06	37	39.98	48.61
1.6.1	25.35	35.59	39	42.52	53.68
1.6.2	21.98	30.30	33	35.58	43.73
1.6.3	24.23	34.93	38	41.28	51.02
1.6.4	26.35	33.37	36	38.59	46.38
1.7.1	26.30	34.47	37	39.50	48.57
1.7.2	43.74	52.42	55	57.60	66.44

**Table 19 The Summarized UFP for Size Estimation of Each Work Package**

#### **4.1.3 Resource Effort Estimation**

The third step is to calculate the resource effort based on the ISBSG regression equations shown in Table 15. Furthermore, according to Table 14, the fixed value of 8.1 is used as the PDR based on a Java platform and the 75% column following the fifth and seventh assumptions. 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 15 (the team size is 3 per the second assumption). The resulting durations for each work package are shown in Table 20. Note that these are the expected durations based on the pessimistic (75% value) UFP.

WP	PDR	75% UFP	Effort	Number of hours per resource	Duration (Month)
1.1.1	8.1	58.52	474	157.99	0.9
1.1.2		64.31	521	173.64	1.0
1.1.3		44.96	364	121.38	0.7
1.1.4		44.96	364	121.38	0.7
1.1.5		47.52	385	128.29	0.7
1.2.1		60.87	493	164.35	0.9
1.2.2		68.83	558	185.85	1.1
1.2.3		36.43	295	98.37	0.6
1.2.4		32.32	262	87.26	0.5
1.2.5		32.32	262	87.26	0.5
1.2.6		32.32	262	87.26	0.5
1.3.1		57.82	468	156.10	0.9
1.3.2		60.74	492	163.99	0.9
1.3.3		36.74	298	99.21	0.6
1.3.4		32.62	264	88.08	0.5
1.4.1		66.70	540	180.09	1.0
1.4.2		60.70	492	163.88	0.9
1.4.3		38.63	313	104.30	0.6
1.4.4		33.52	271	90.50	0.5
1.4.5		43.74	354	118.11	0.7
1.5.1		67.05	543	181.03	1.0
1.5.2		62.07	503	167.58	1.0
1.5.3		36.04	292	97.29	0.6
1.5.4		39.98	324	107.94	0.6
1.6.1		42.52	344	114.81	0.7
1.6.2		35.58	288	96.06	0.5
1.6.3		41.28	334	111.47	0.6
1.6.4		38.59	313	104.20	0.6
1.7.1	39.50	320	106.66	0.6	
1.7.2	57.60	467	155.53	0.9	

**Table 20 The Summarized Pessimistic Duration of WBS Level 3**

#### 4.1.4 Export the WBS to Microsoft Project 2010

The fourth and fifth steps are to export the WBS to Project 2010 and to form the predecessor relationships between each work package. Note that the all work packages are critical path activities according the sixth assumption. The calculated durations based on the three programmers shown in Table 20 are simply put in the WBS level 3 duration column. This is shown in Figure 39.

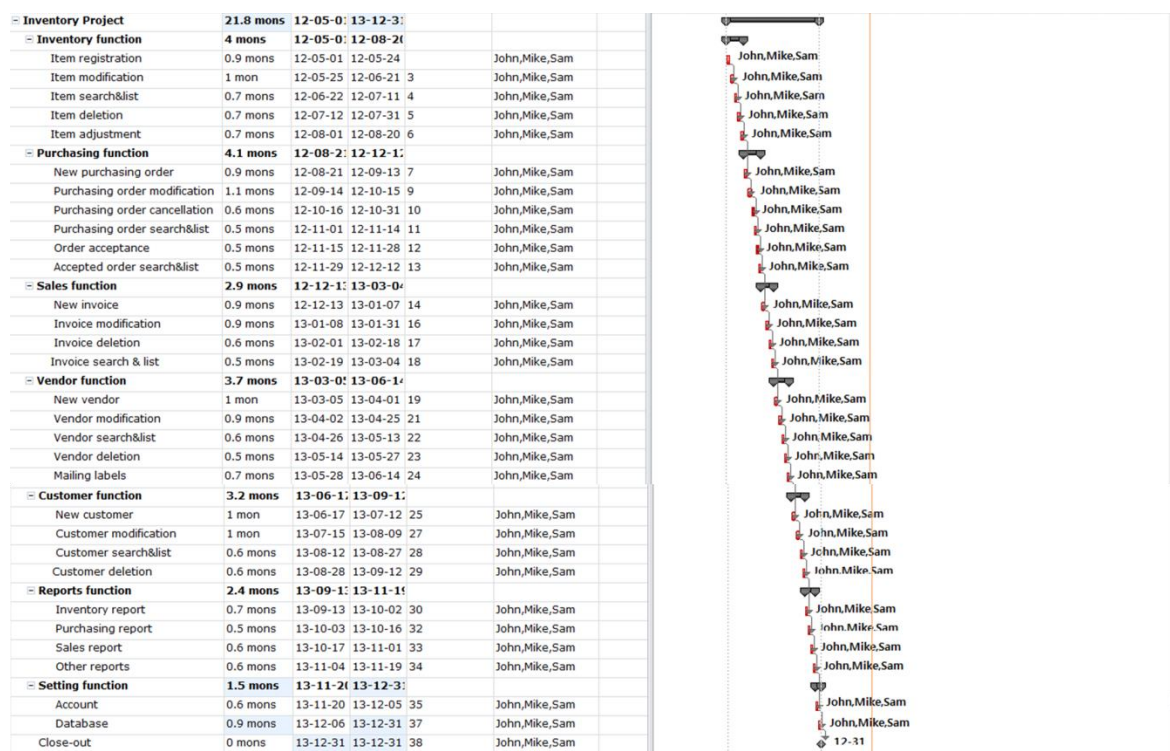


Figure 39 Critical Activities and Duration of WBS Level 2 & Entire Project

#### 4.1.5 Final Schedule Estimation in the Early Stage

As shown in Figure 39, 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. Therefore, the calculated duration of the entire project is 21.8 months.

Yet, 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 (75% UFP). To complete this 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 estimation in step 2).

To accomplish this, the 25%, mean, and 75% values for UFP from Table 19 are used to calculate optimistic, most likely, and pessimistic duration values (as previously shown in Table 20), 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 at WBS level 2 (a sample size of 7 work packages) is shown in Table 21; the final estimate at WBS level 3 (using all 30 work packages) is shown in Table 22. Note that the expected total duration reflected in both tables is less than the 21.8 months that was calculated based on the single-point estimate.

Simulation Results		Probability Range	
Min. Duration	19.57	10%	19.96
Mean Duration	20.30	15%	20.03
Max Duration	21.04	20%	20.08
N/A		25%	20.13
		30%	20.17
		35%	20.21
		40%	20.24
		45%	20.27
		50%	20.30
		55%	20.33
		60%	20.37
		65%	20.40
		70%	20.43
		75%	20.47
		80%	20.52
		85%	20.56
		90%	20.62
	95%	20.72	
	100%	20.89	

**Table 21 Final Schedule Estimate of WBS Level 2**

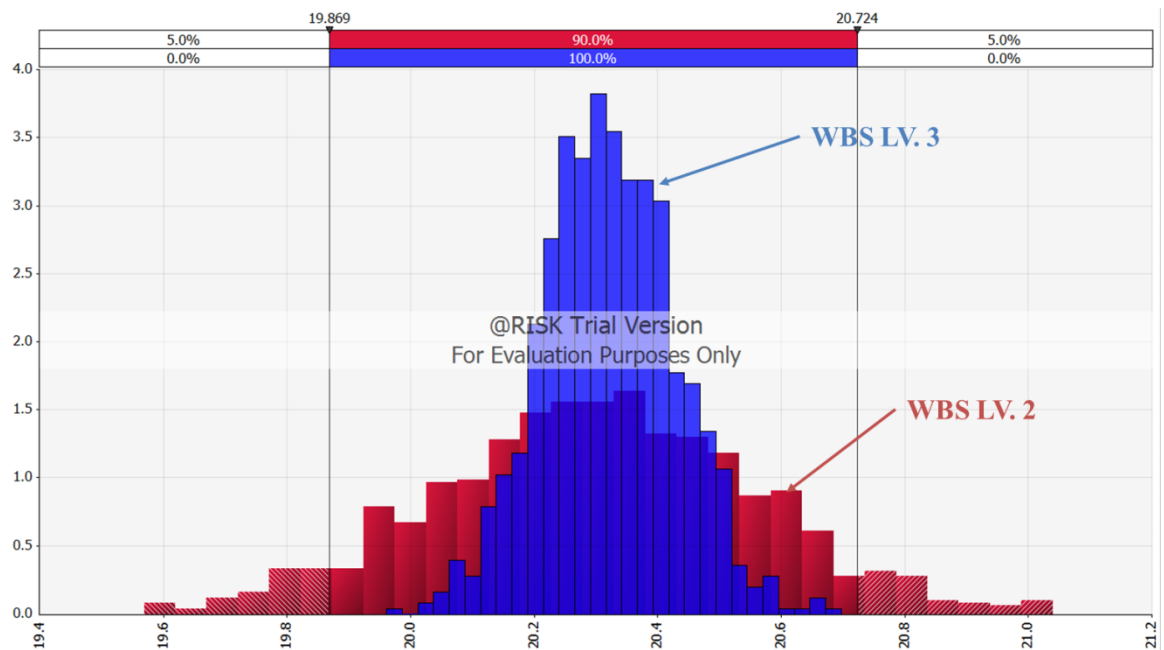
Simulation Results		Probability Range	
Min. Duration	19.91	10%	20.19
Mean Duration	20.32	15%	20.21
Max Duration	20.63	20%	20.24
N/A		25%	20.25
		30%	20.27
		35%	20.28
		40%	20.29
		45%	20.31
		50%	20.32
		55%	20.34
		60%	20.35
		65%	20.37
		70%	20.38
		75%	20.39
		80%	20.41
		85%	20.43
		90%	20.45
	95%	20.49	
	100%	20.56	

**Table 22 Final Schedule Estimate of WBS Level 3**

#### 4.1.6 Result and Analysis of Schedule Estimation in the Early Stage

As an example of how to use the information in Tables 21 and 22, suppose a customer wants to know the chance of completing this project within 20.40 months. The project manager can see that the answer is either 65% (Table 21) or between 75%~80% (Table 22). What to address is that the estimate information in Table 22 (based on more work packages) is more accurate. Indeed, the values in Table 21 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 40.



**Figure 40 Comparison Analysis of the Range of Schedule Estimation between WBS Level 2 and Level 3**

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 seven work packages reflected at WBS level 2.

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 the approach in this section focuses on developing a more accurate schedule, the chance for budget or scope changes caused by schedule problems is significantly reduced. The results in this section have been published in the *Journal of Information Systems Applied Research (JISAR)* (Kwon & Hammell, 2013).

#### **4.2 Schedule Refinement or Verification in the Planning Stage**

For the simulation in this section, the same example and assumptions are used as stated in Section 3, but there is one more assumption to calculate Adjusted Function Point (AFP); that is, the value of  $\Sigma(F_i)$  is 63. Remember that  $\Sigma(F_i)$  is the equation component to calculate AFP as described in Section 3.1.3, and the value 63 is determined by our own discretion as a convenient, somewhat random pick that will serve to illustrate our point.

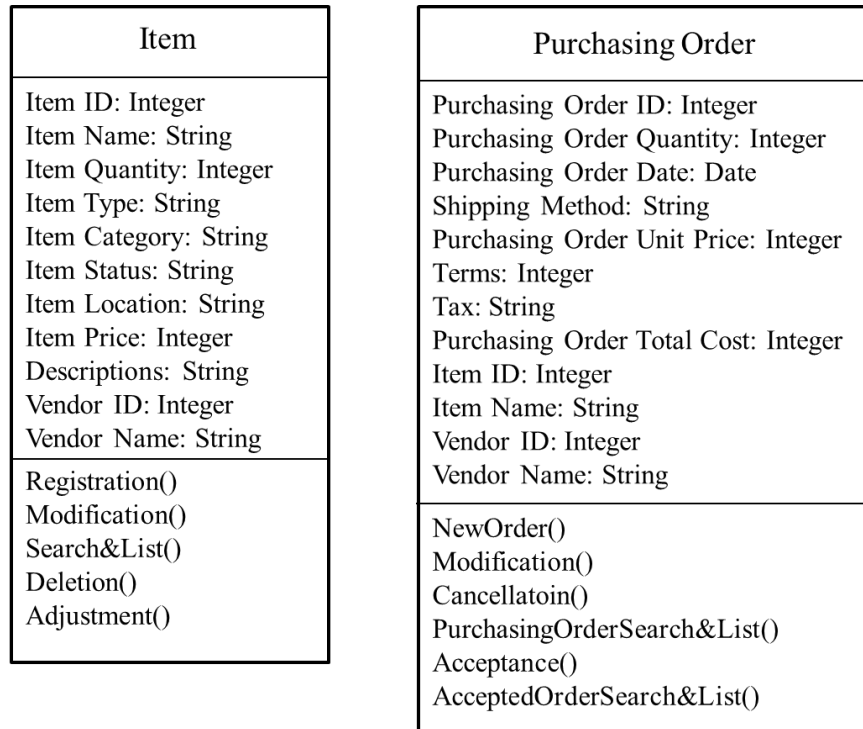
Section 4.1 proposed a framework for the objective and accurate estimation of software project schedules in the proposal preparation stage of large-sized software development projects. A probabilistic approach was used to take into account the



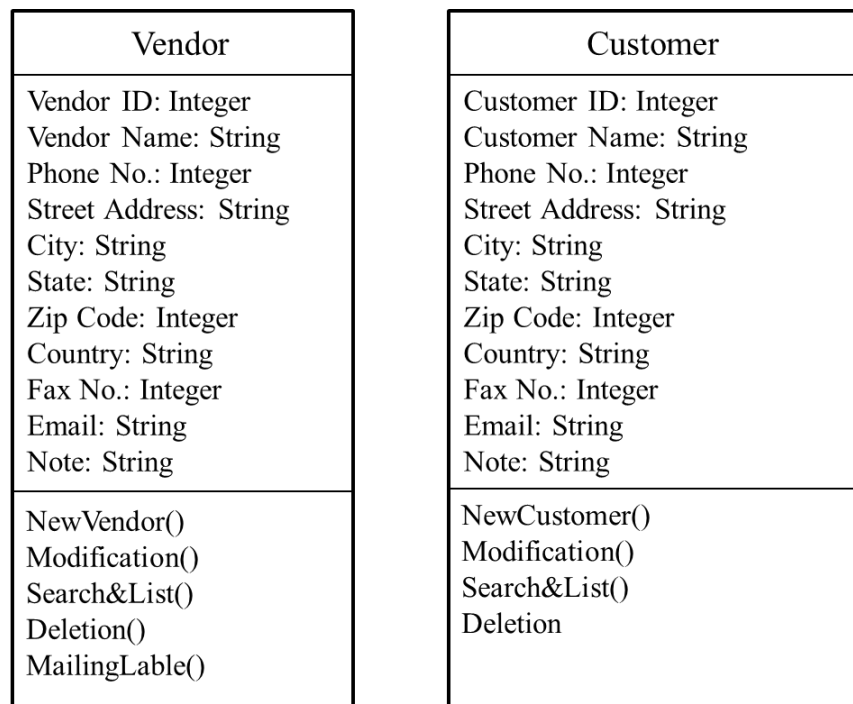
uncertainty inherent in the early stage of such projects. This section illustrates how to use the framework in the later planning stage when additional, more detailed information is available via the design documentation. The process continues to use the International Function Point User Group (IFPUG) function point analysis and the International Software Benchmarking Standards Group (ISBSG) regression equations to calculate project size and resource effort, but substitutes the more accurate AFP information for the previously used UFP data. The use of the proposed framework in the planning stage provides for the reassessment, verification and/or refinement of the estimated schedule produced in the early stage.

Once the planning stage is reached, it is useful (and perhaps necessary) to revalidate the initial project schedule estimate that was done in the proposal preparation stage. This is made possible by being able to obtain more accurate counts of the functional components using design documentation available at this stage. This new functional component information, obtained from class diagrams and sequence diagrams, is used to refine / verify the previous initial estimate.

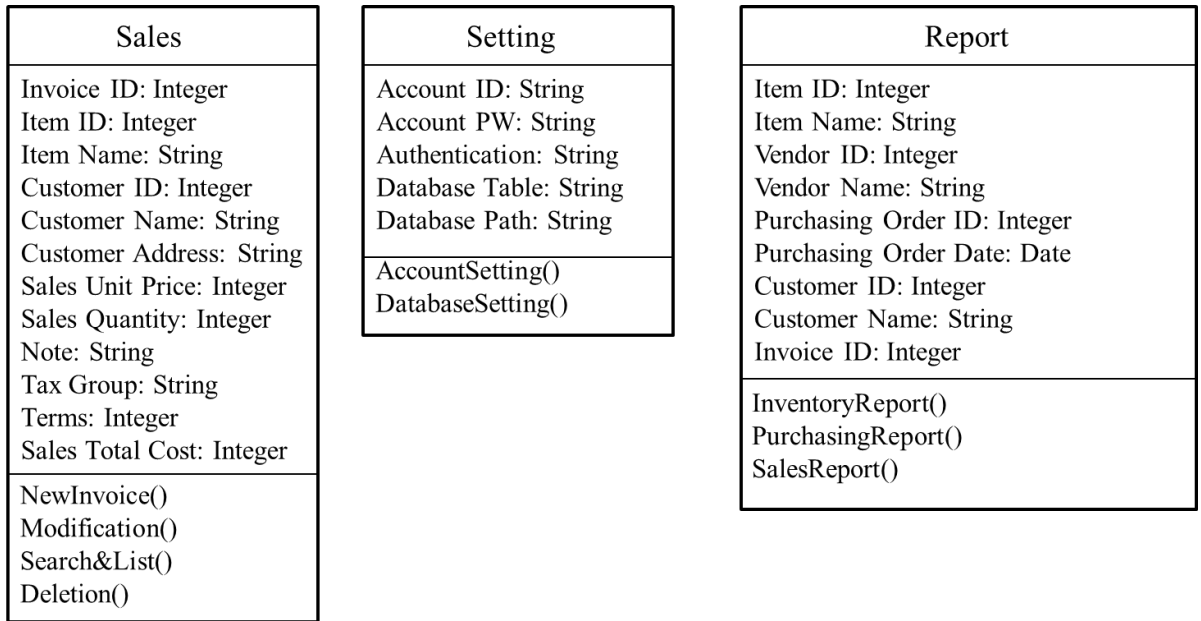
The following Figure 41 through Figure 43 shows the entire class diagram based on class operations. Note that this is an illustrative example. Through these class diagrams, it is possible to count the exact number of External Input (EI) with consideration of the database relationship which is shown in Figure 44. The counted EI number of each work package is also shown in Table 23. To identify the exact number of each functional component except EI, sequence diagrams are required for each work package, and this is shown in Figure 45 through Figure 74.



**Figure 41 Class Diagram for Item and Purchasing Order Functionality**



**Figure 42 Class Diagram for Vendor and Customer Functionality**



**Figure 43 Class Diagram for Sales, Setting, and Report Functionality**

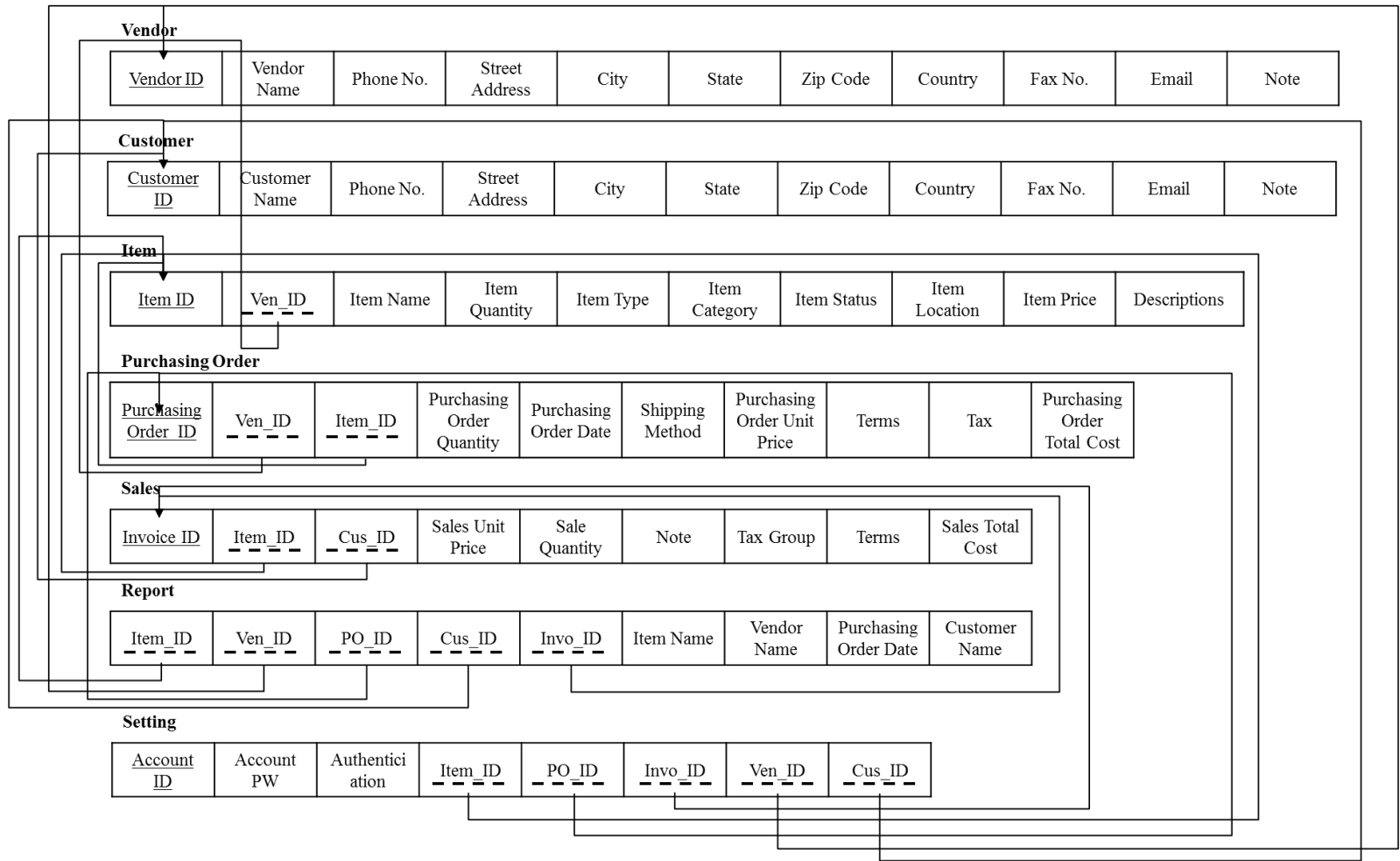
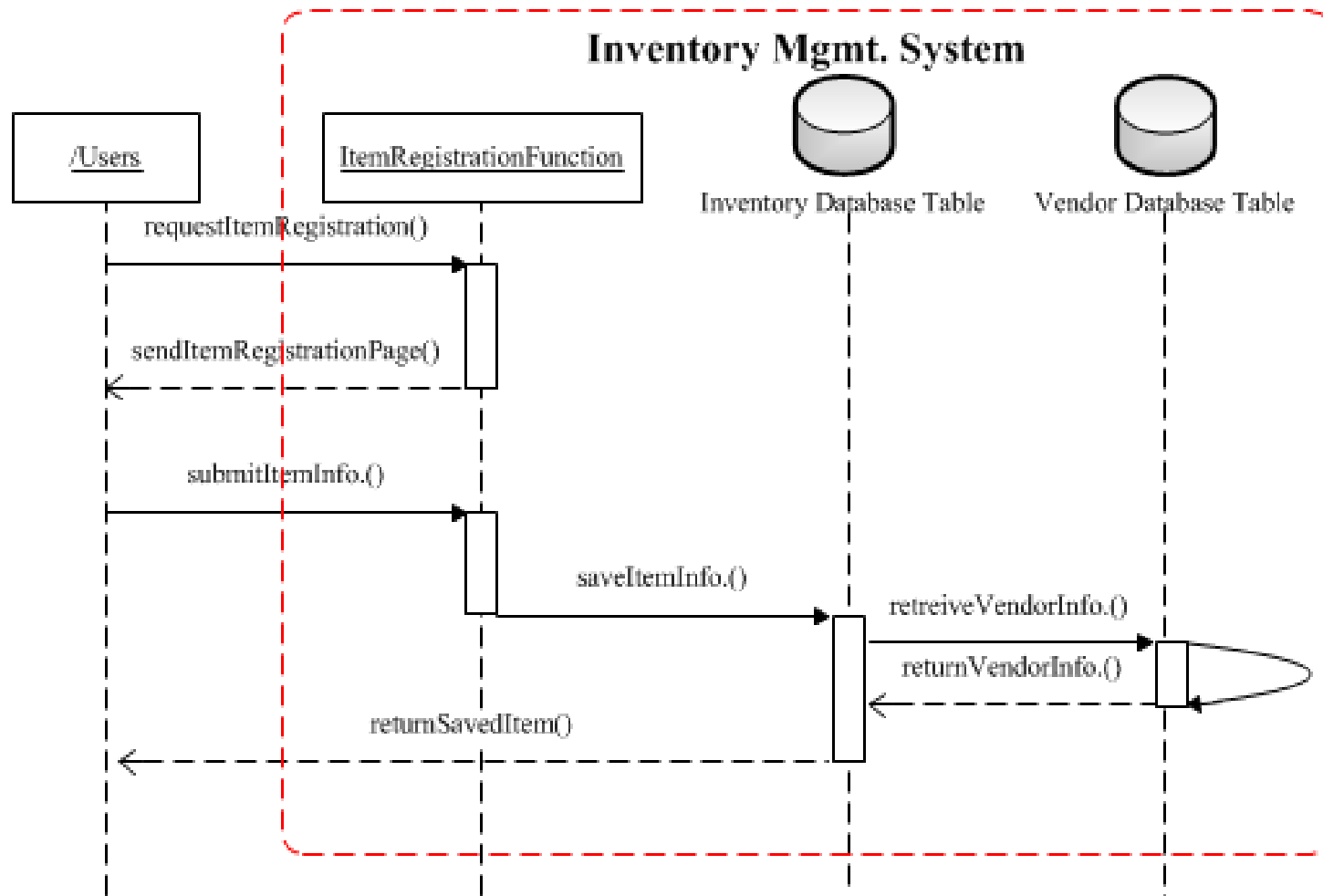


Figure 44 Database Relationship between Each Work Package

Work Package	Input Attributes	EI Number
1.1.1 Item Registration	Item ID, Item Name, Item Quantity, Item Type, Item Category, Item Status, Item Location, Item Price, Descriptions, Vendor Name	10
1.1.2 Item Modification	Item Name, Item Quantity, Item Type, Item Category, Item Status, Item Location, Item Price, Descriptions, Vendor Name	9
1.1.3 Item Search & List	Item Name, Item Category, Item Location	3
1.1.4 Item Deletion	Item ID, Item Name	2
1.1.5 Item Adjustment	Item Name, Item Location, Item Quantity	3
1.2.1 New Purchasing Order	Purchasing Order ID, Purchasing Order Quantity, Purchasing Order Date, Shipping Method, Purchasing Order Unit Price, Terms, Tax, Purchasing Order Total Cost, Item Name, Vendor Name	10
1.2.2 Purchasing Order Modification	Purchasing Order Quantity, Purchasing Order Date, Shipping Method, Purchasing Order Unit Price, Terms, Tax, Purchasing Order Total Cost, Item Name, Vendor Name	9
1.2.3 Purchasing Order Cancellation	Purchasing Order ID, Purchasing Order Name	2
1.2.4 Purchasing Order Search & List	Purchasing Order ID, Purchasing Order Name, Vendor Name	3
1.2.5 Order Acceptance	Purchasing Order ID, Purchasing Order Name, Vendor Name	3
1.2.6 Accepted Order List	Purchasing Order ID, Purchasing Order Name, Vendor Name	3
1.3.1 New Invoice	Invoice ID, Item Name, Customer Name, Customer Address, Sales Unit Price, Sales Quantity, Note, Tax Group, Terms, Sales Total Cost	10
1.3.2 Invoice Modification	Item Name, Customer Name, Customer Address, Sales Unit Price, Sales Quantity, Note, Tax Group, Terms, Sales Total Cost	9
1.3.3 Invoice Deletion	Invoice ID, Item Name, Customer Name	3
1.3.4 Invoice Search & List	Invoice ID, Item Name, Customer Name	3
1.4.1 New Vendor	Vendor ID, Vendor Name, Phone No., Street Address, City, State, Zip Code, Country, Fax No., Email, Note	11
1.4.2 Vendor Modification	Vendor Name, Phone No., Street Address, City, State, Zip Code, Country, Fax No., Email, Note	10
1.4.3 Vendor Search & List	Vendor Name, Phone No., Email	3
1.4.4 Vendor Deletion	Vendor ID, Vendor Name	2

1.4.5 Mailing Label	Vendor Name, Phone No., Street Address, City, State, Zip Code, Country	7
1.5.1 New Customer	Customer ID, Customer Name, Phone No., Street Address, City, State, Zip Code, Country, Fax No., Email, Note	11
1.5.2 Customer Modification	Customer Name, Phone No., Street Address, City, State, Zip Code, Country, Fax No., Email, Note	10
1.5.3 Customer Search & List	Customer Name, Phone No., Email	3
1.5.4 Customer Deletion	Customer ID, Customer Name	2
1.6.1 Inventory Report	Item ID, Item Name	2
1.6.2 Purchasing Order Report	Purchasing Order ID, Purchasing Order Date, Vendor Name	3
1.6.3 Sales Report	Invoice ID, Customer Name	2
1.6.4 Other Reports	Customer ID, Customer Name, Vendor ID, Vendor Name	4
1.7.1 Account Setting	Account ID, Account PW, Authentication	3
1.7.2 Database Setting	Database Table, Database Path	2

**Table 23 The Counted EI Number of Each Work Package**



**Figure 45 WBS 1.1.1 Item Registration Sequence Diagram**

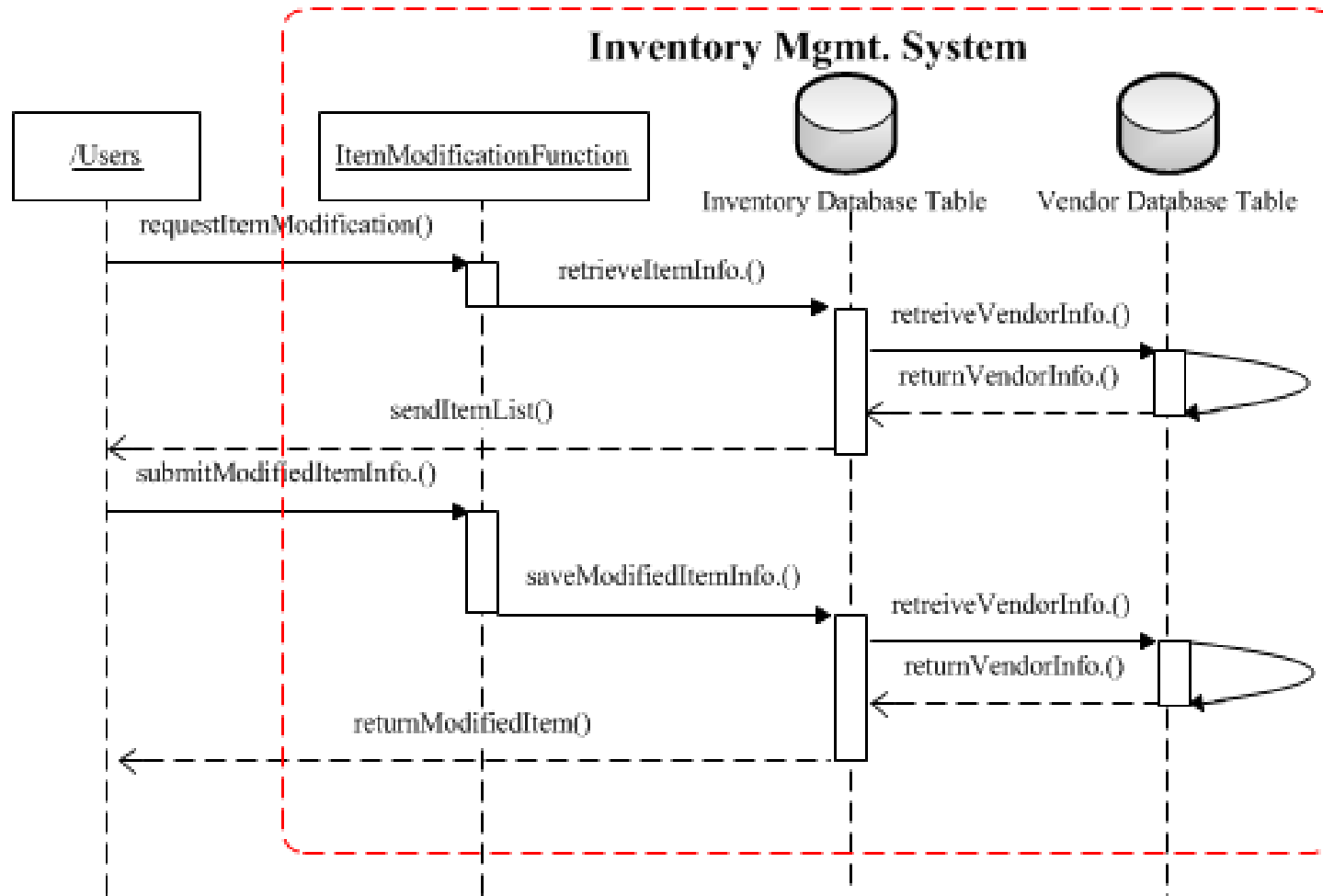
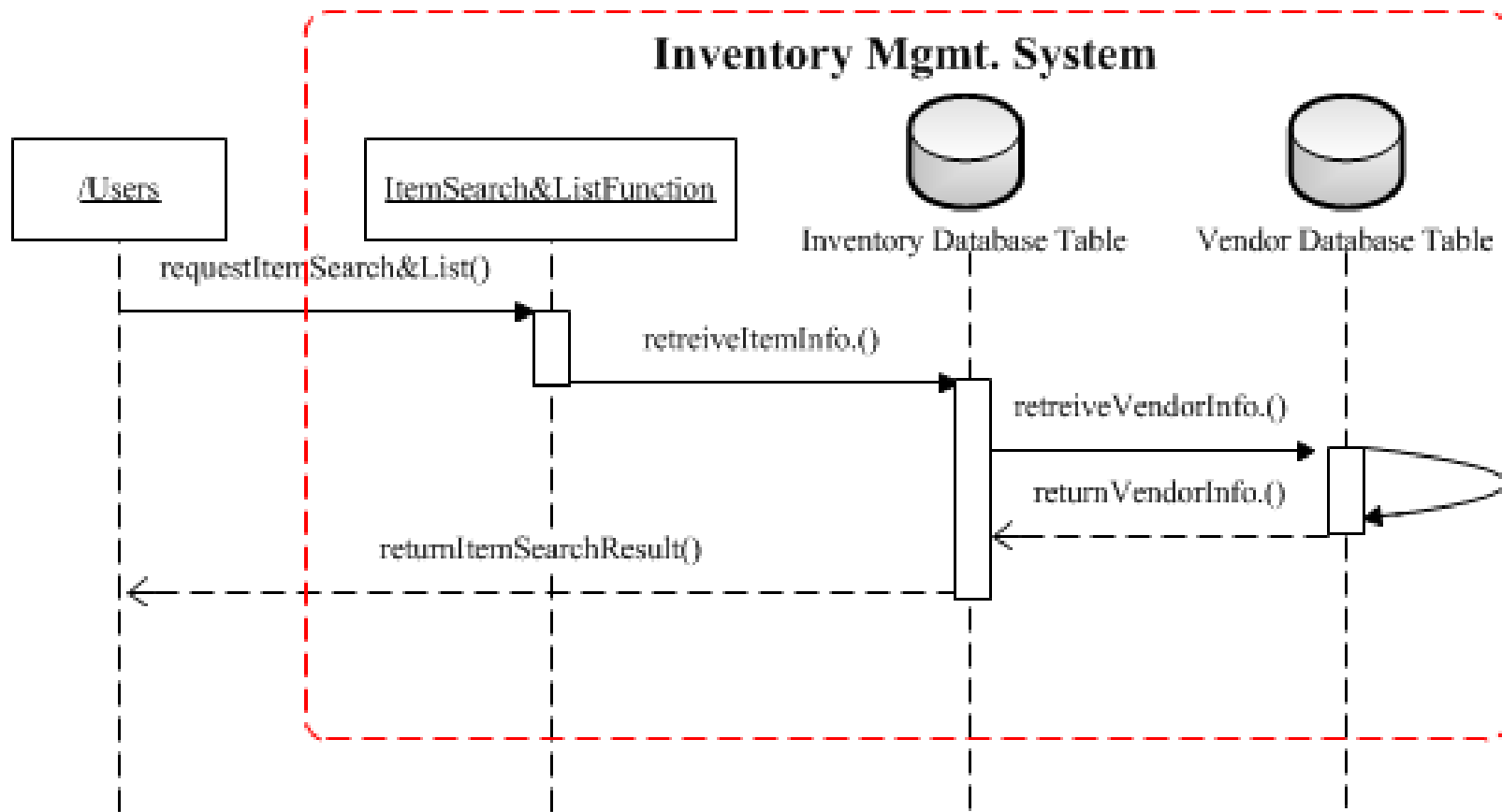


Figure 46 WBS 1.1.2 Item Modification Sequence Diagram





**Figure 47 WBS 1.1.3 Item Search & List Sequence Diagram**

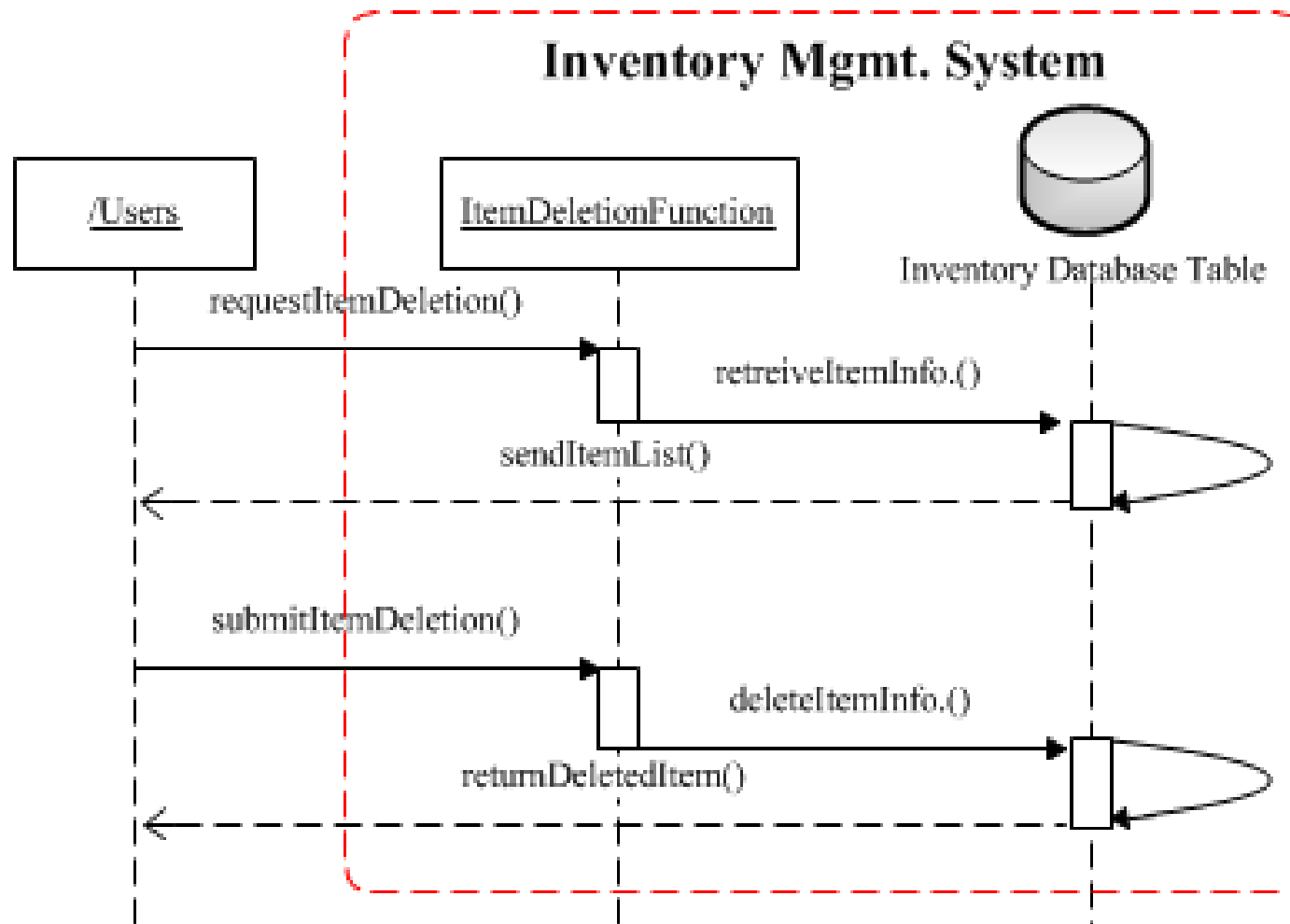


Figure 48 WBS 1.1.4 Item Deletion Sequence Diagram

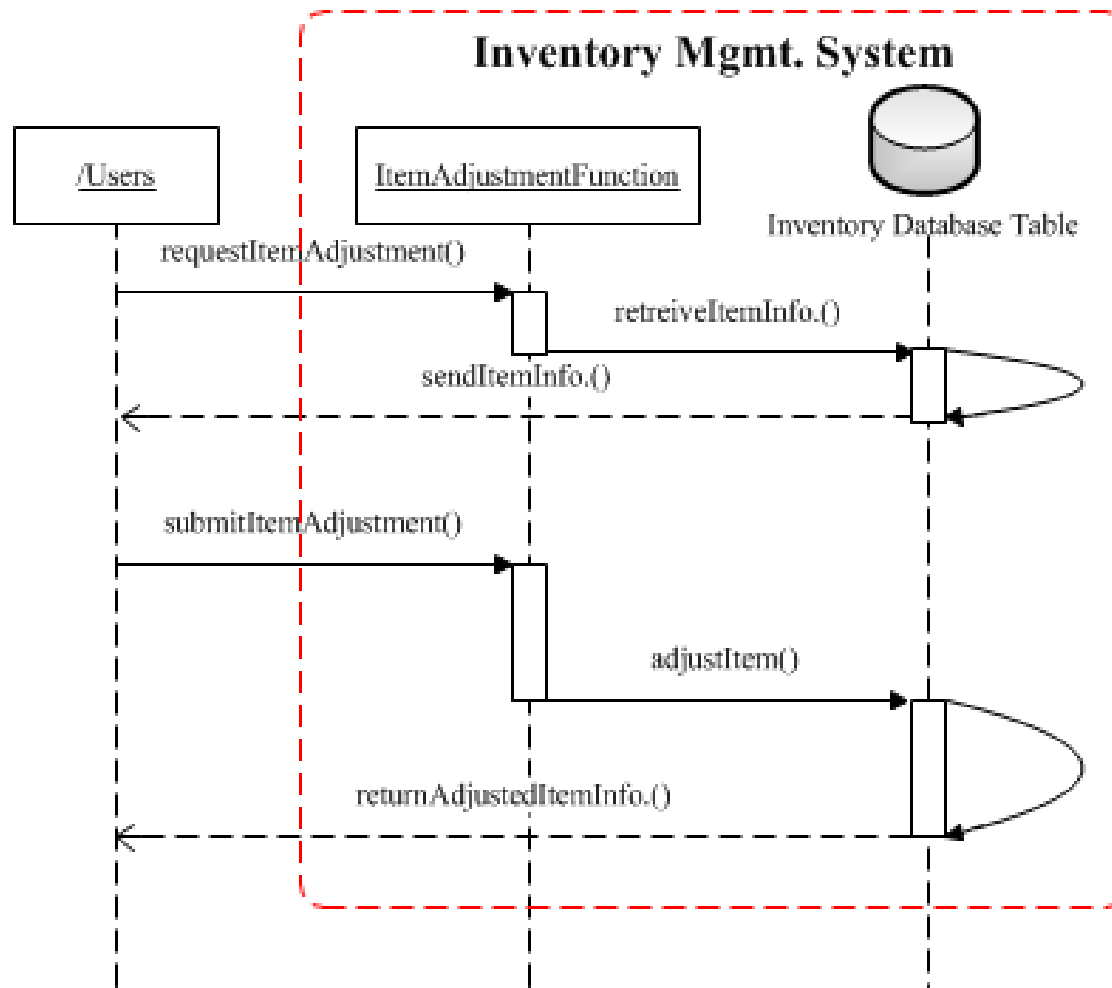


Figure 49 WBS 1.1.5 Item Adjustment Sequence Diagram

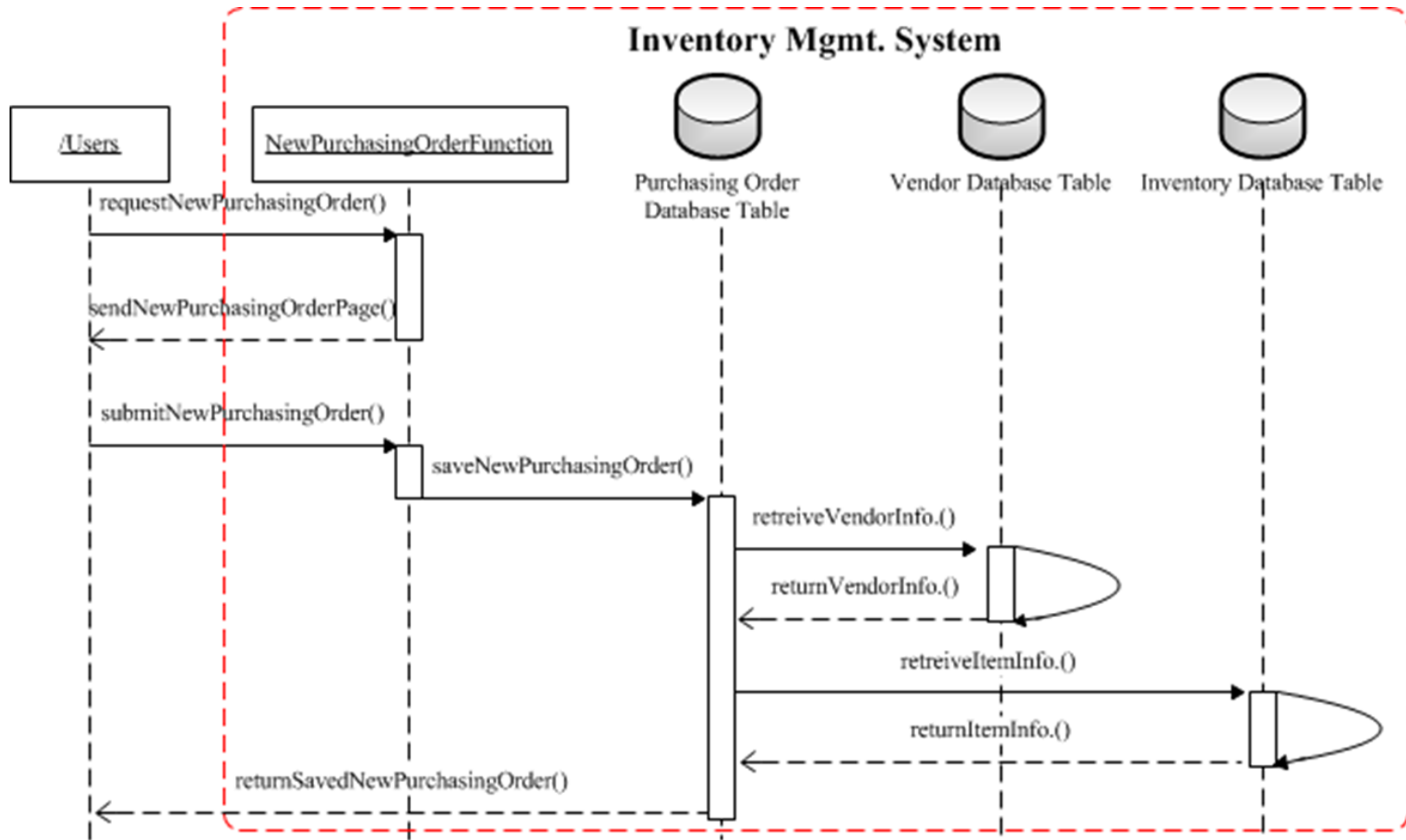
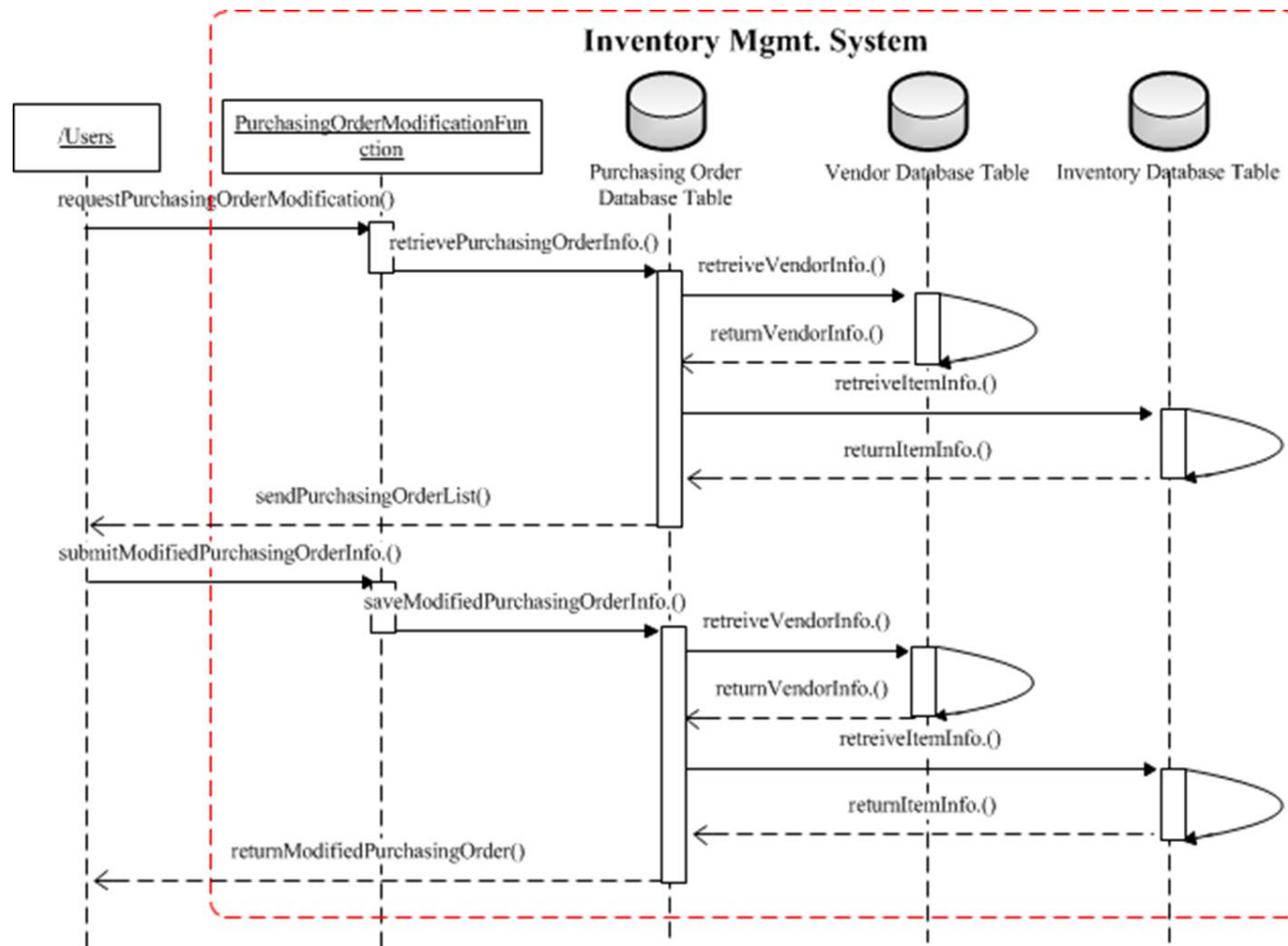


Figure 50 WBS 1.2.1 New Purchasing Order Sequence Diagram



**Figure 51 WBS 1.2.2 Purchasing Order Modification Sequence Diagram**

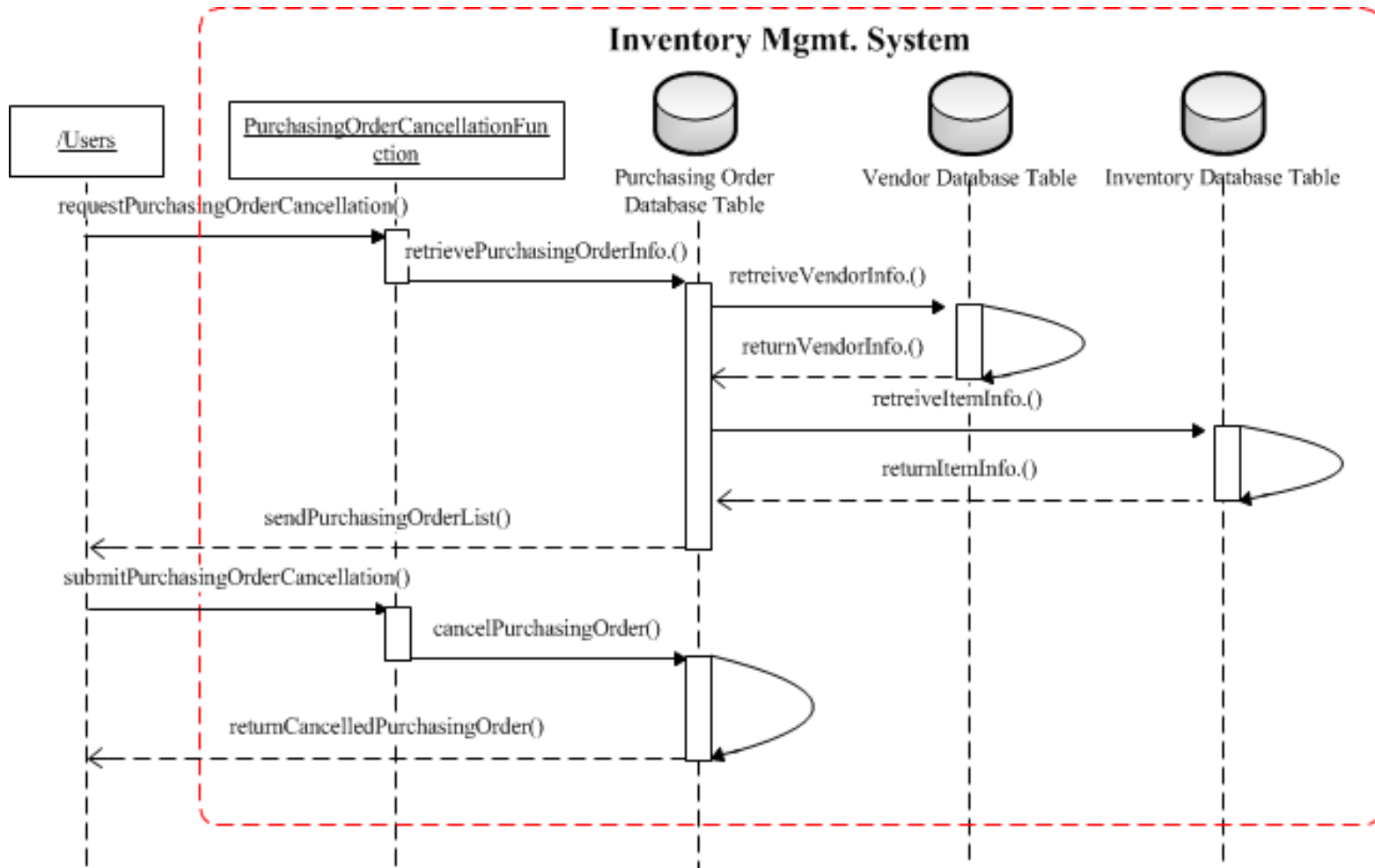


Figure 52 WBS 1.2.3 Purchasing Order Cancellation Sequence Diagram

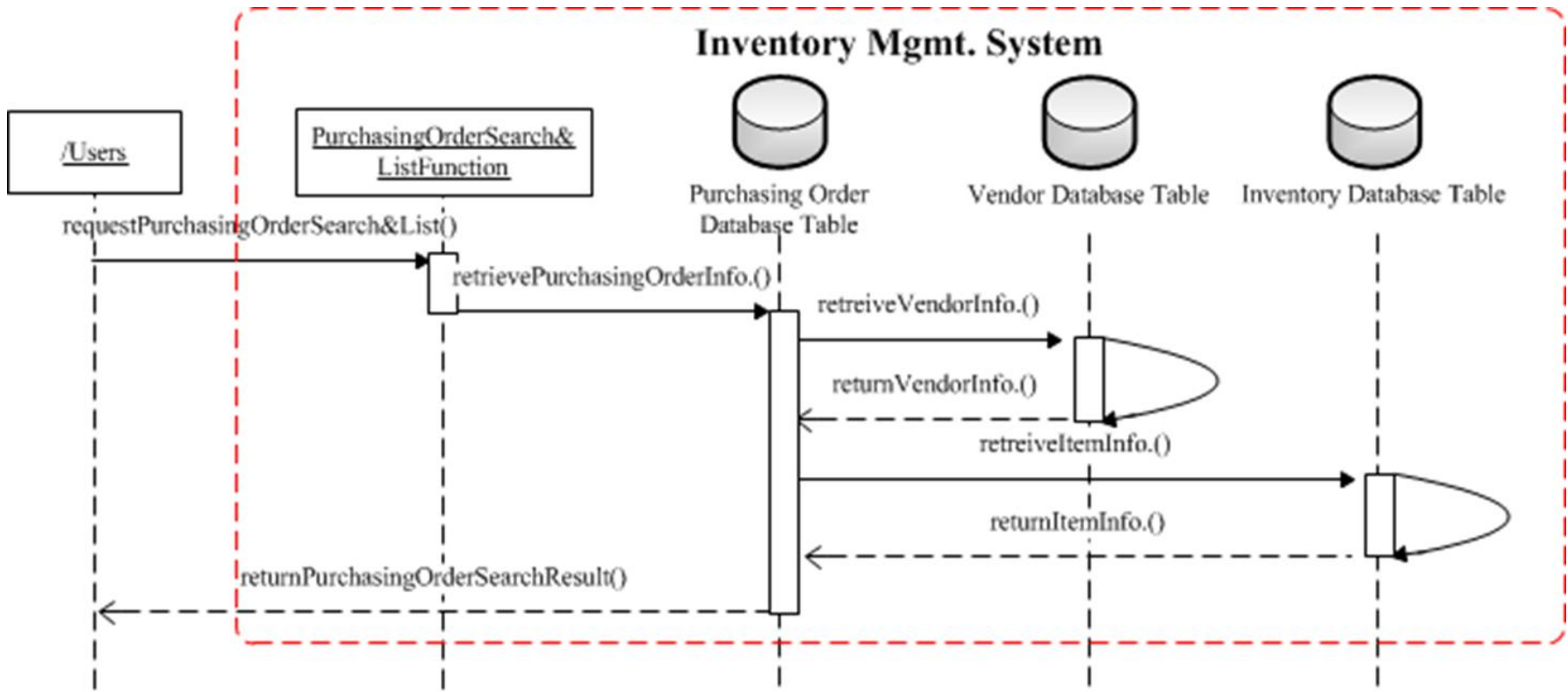


Figure 53 WBS 1.2.4 Purchasing Order Search & List Sequence Diagram

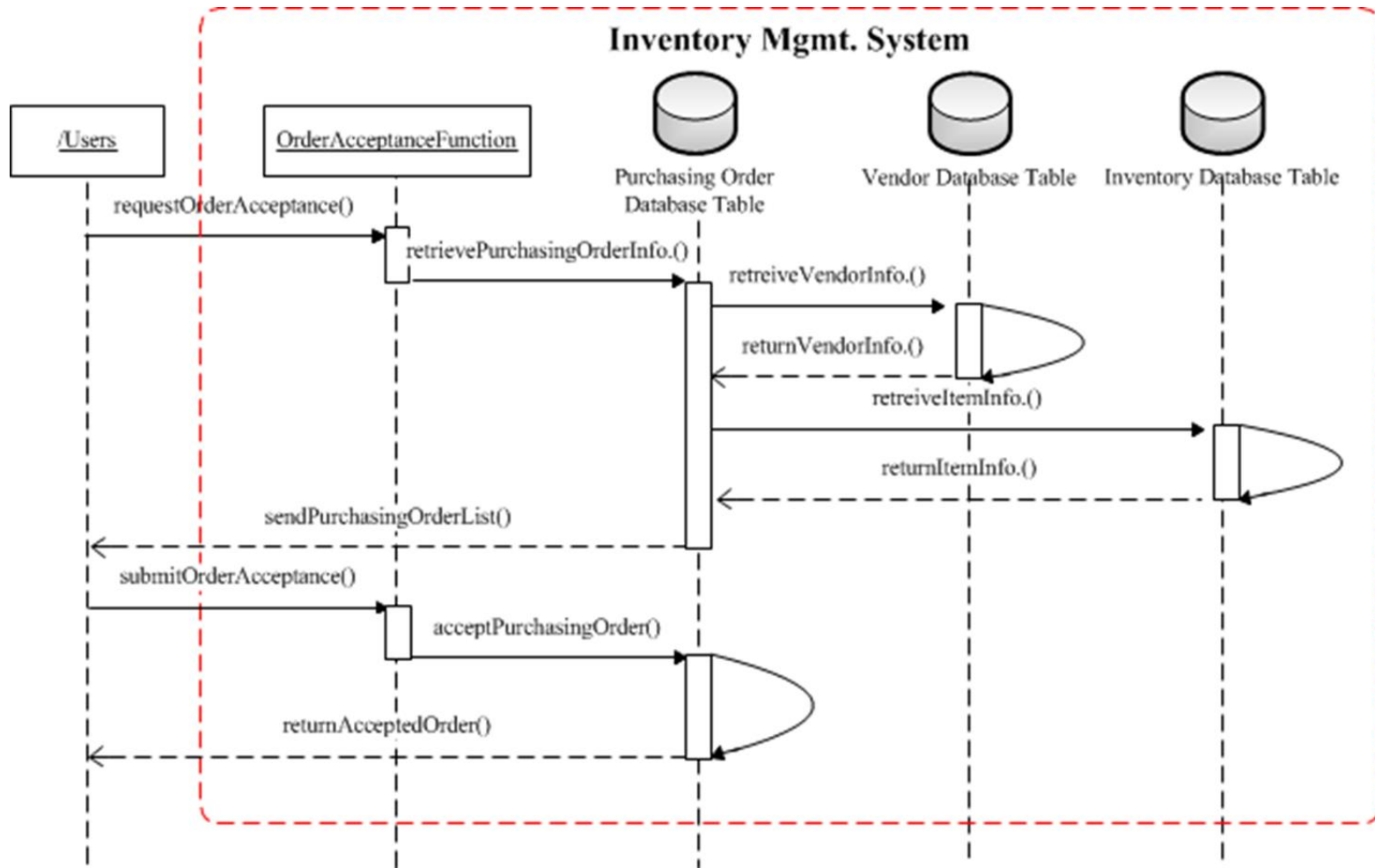


Figure 54 WBS 1.2.5 Order Acceptance Sequence Diagram



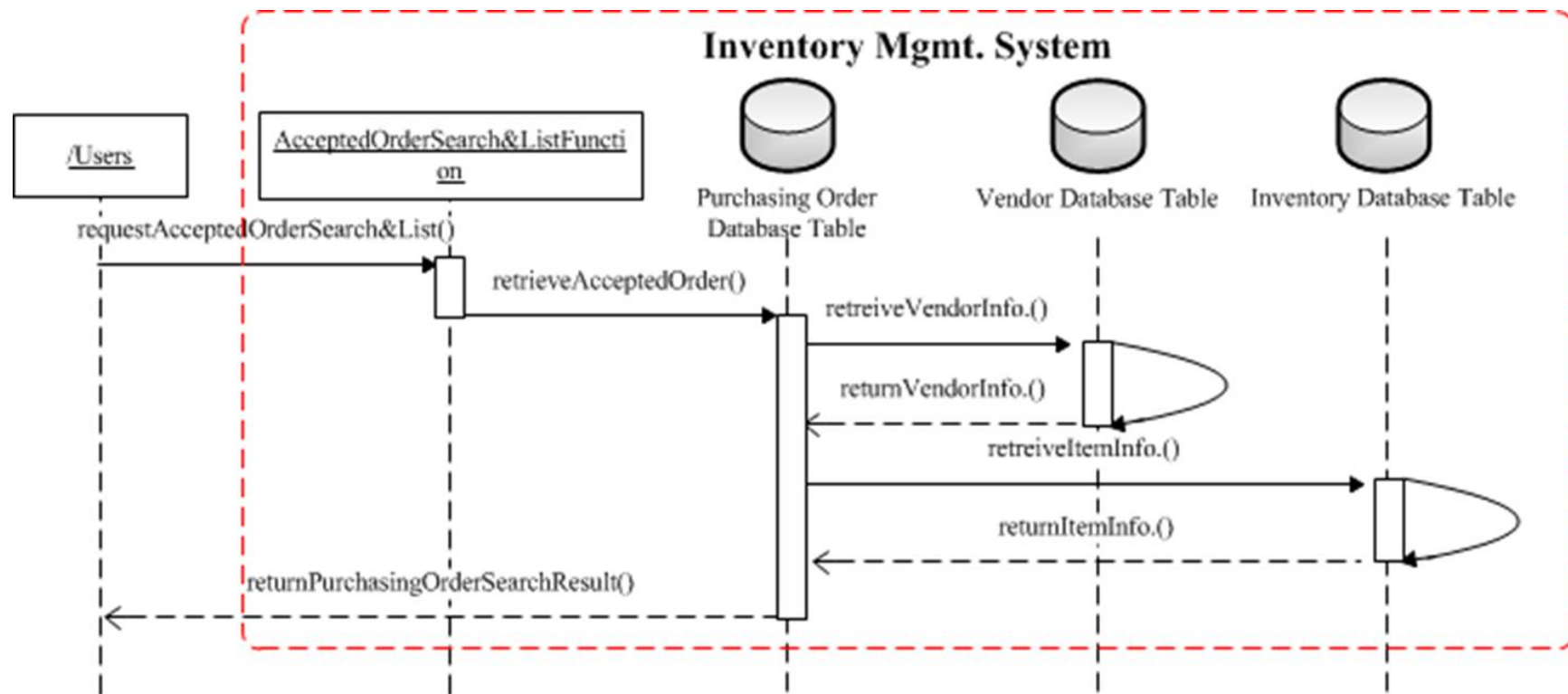


Figure 55 WBS 1.2.6 Accepted Order Sequence Diagram

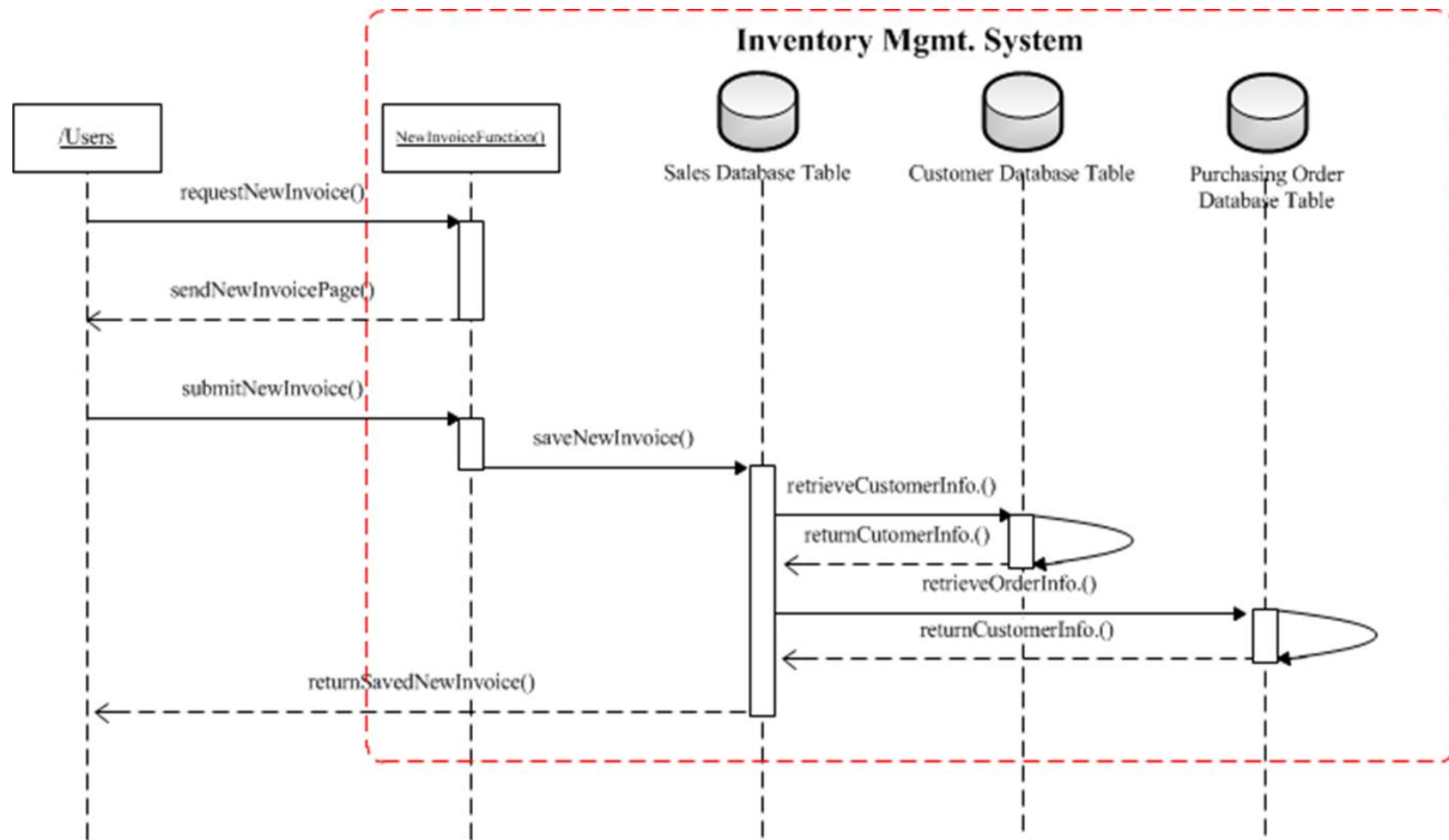


Figure 56 WBS 1.3.1 New Invoice Sequence Diagram

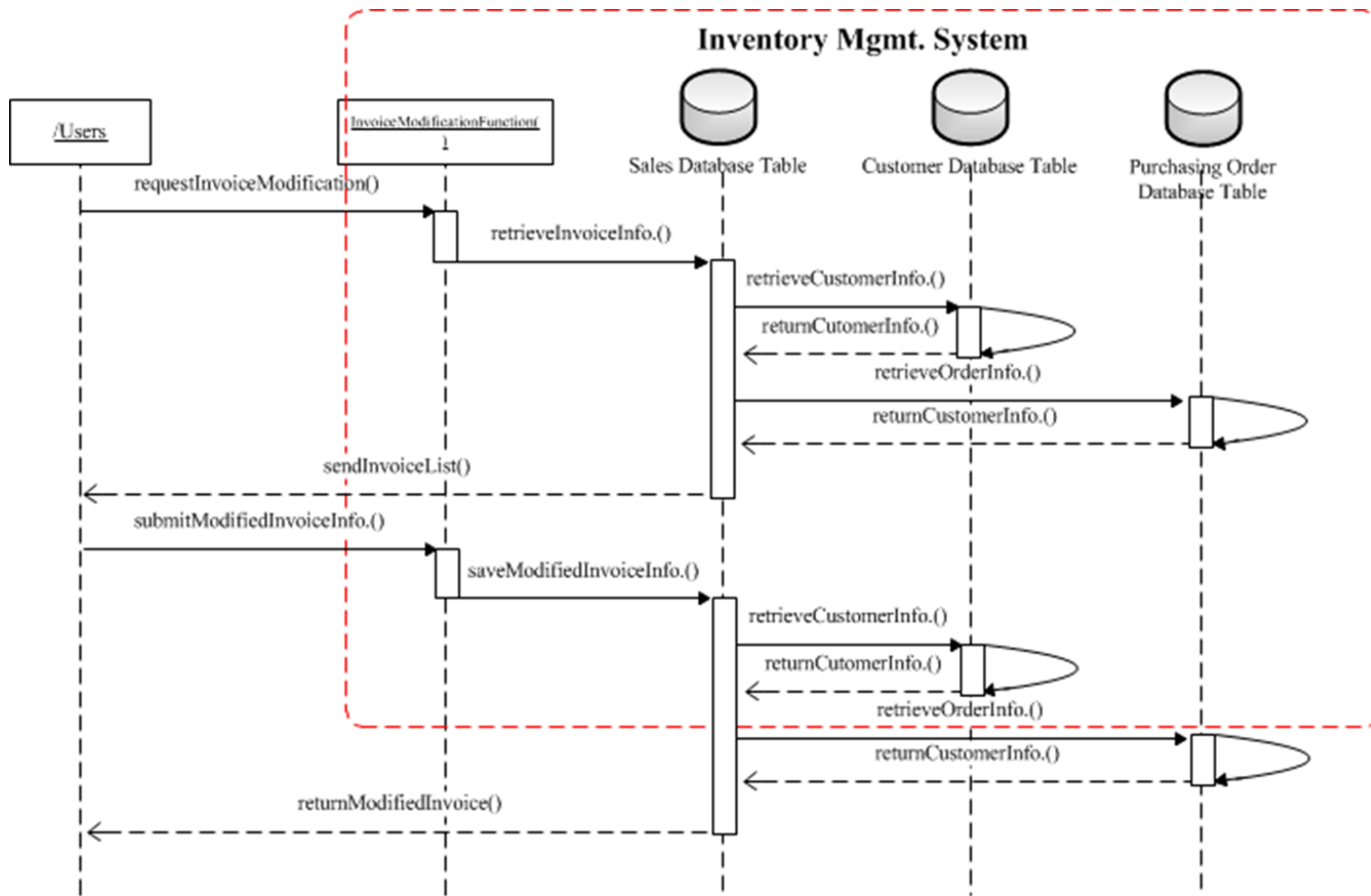


Figure 57 WBS 1.3.2 Invoice Modification Sequence Diagram

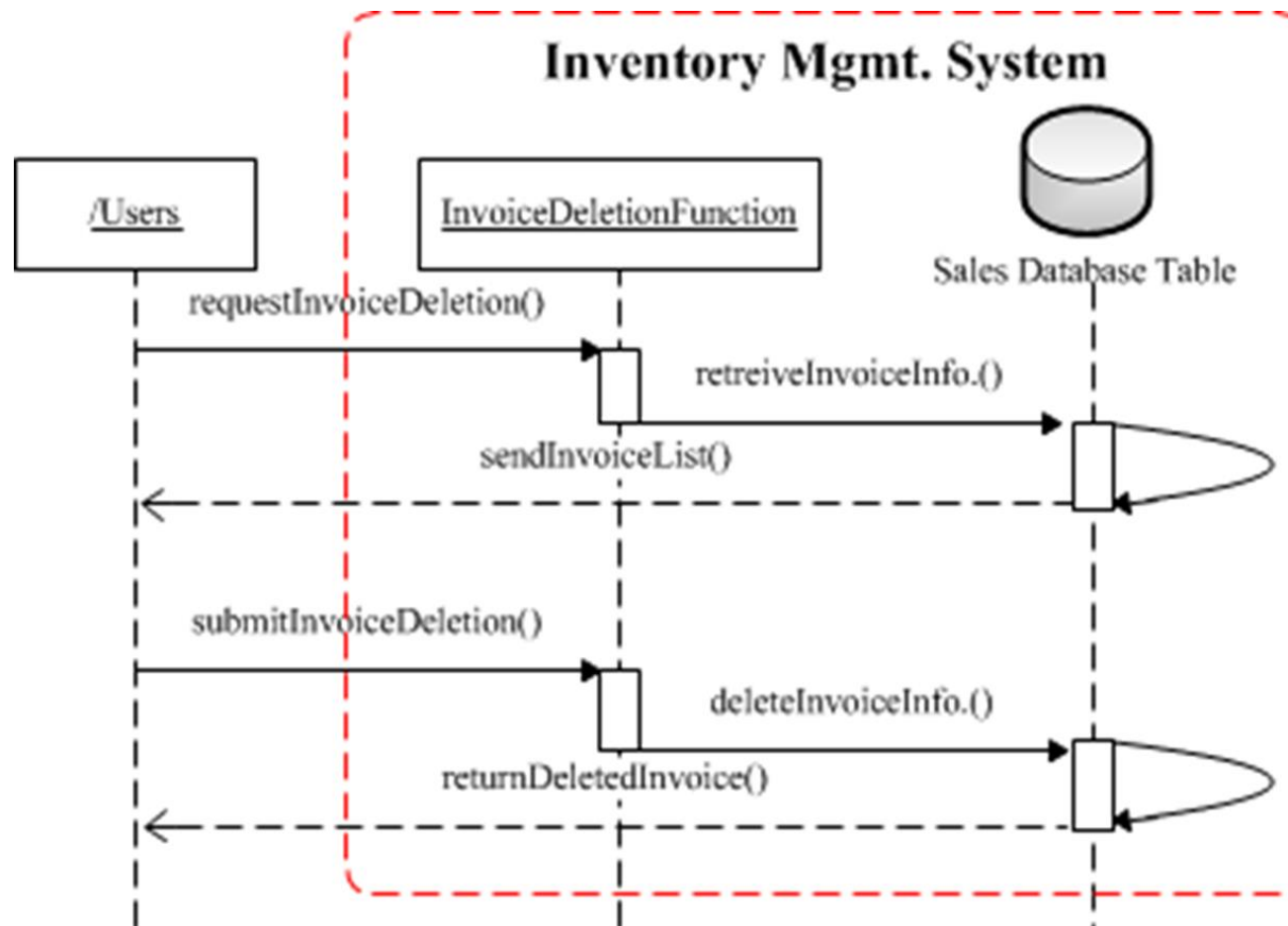


Figure 58 WBS 1.3.3 Invoice Deletion Sequence Diagram

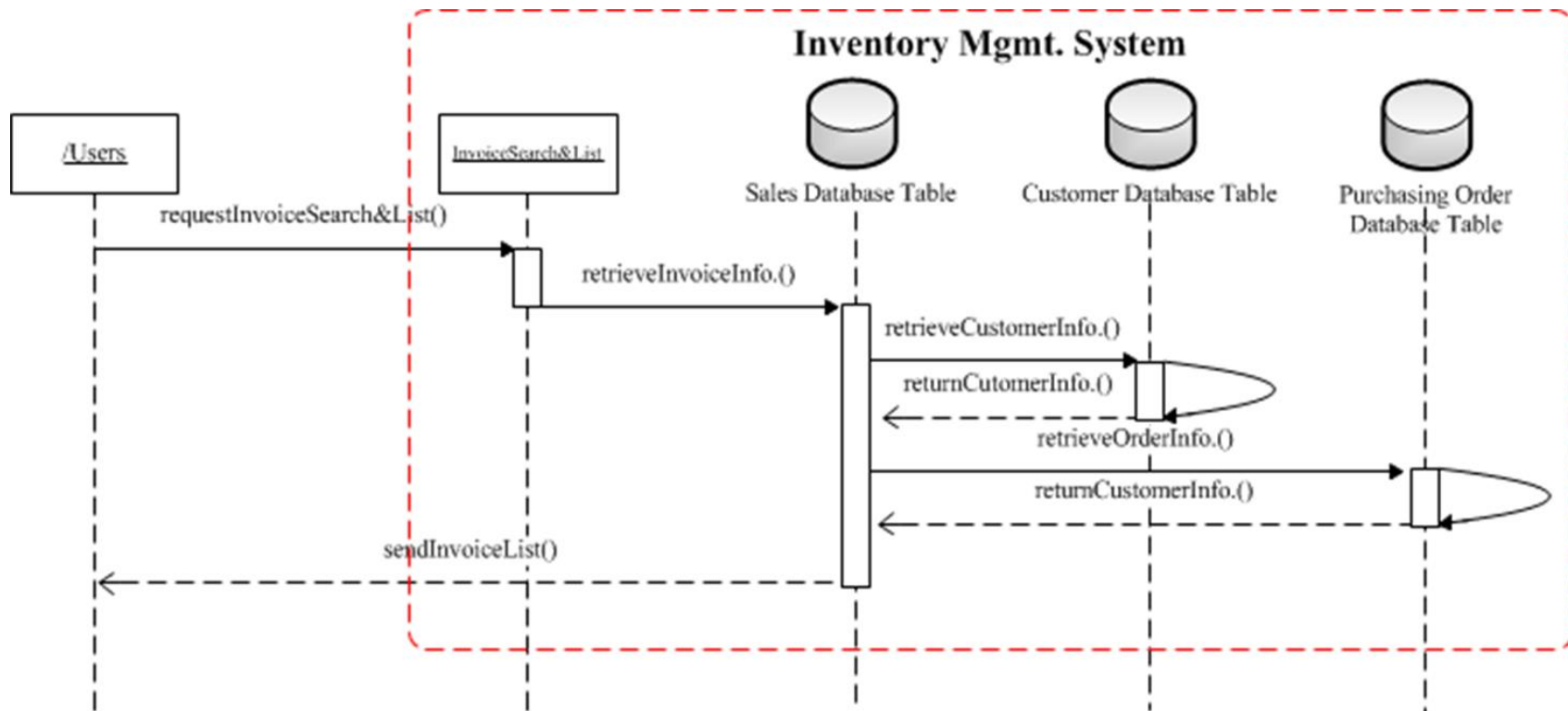


Figure 59 WBS 1.3.4 Invoice Search & List Sequence Diagram

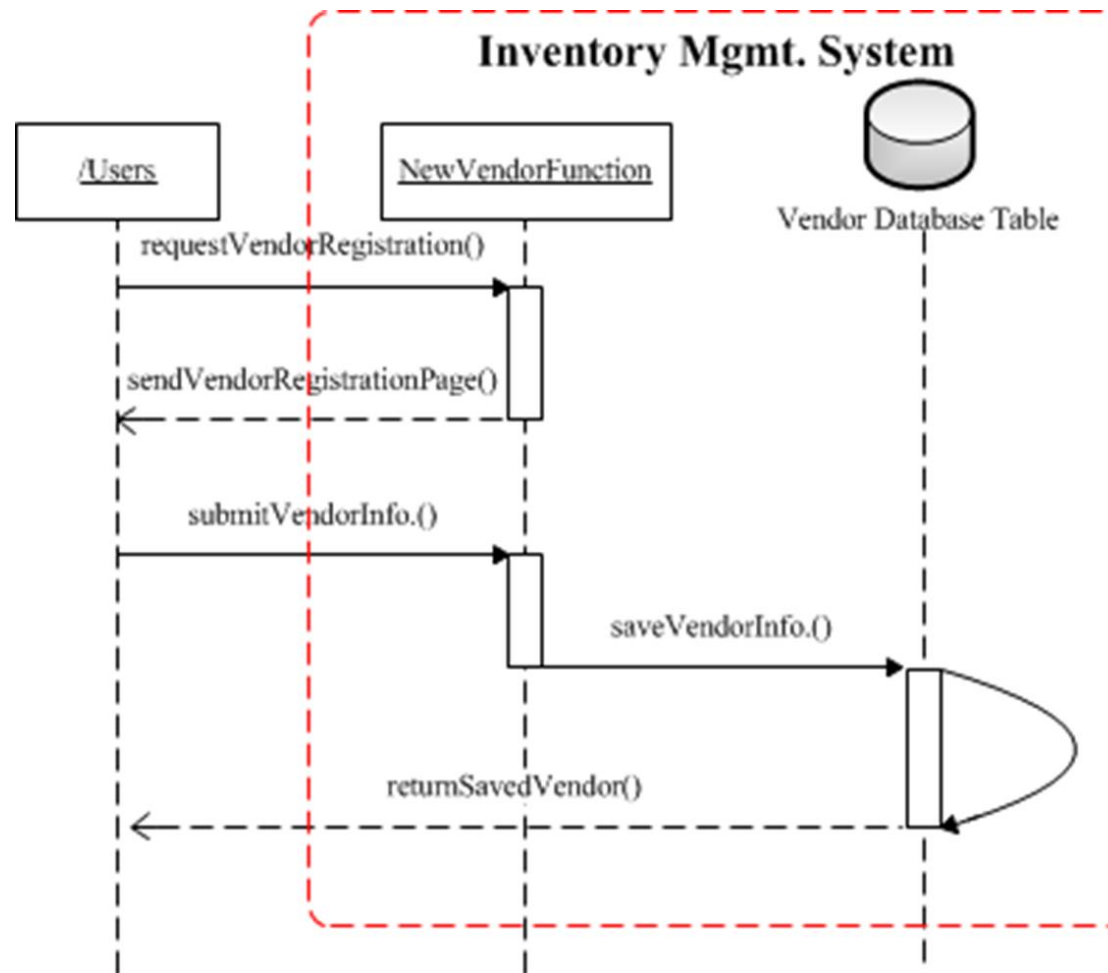


Figure 60 WBS 1.4.1 New Vendor Sequence Diagram

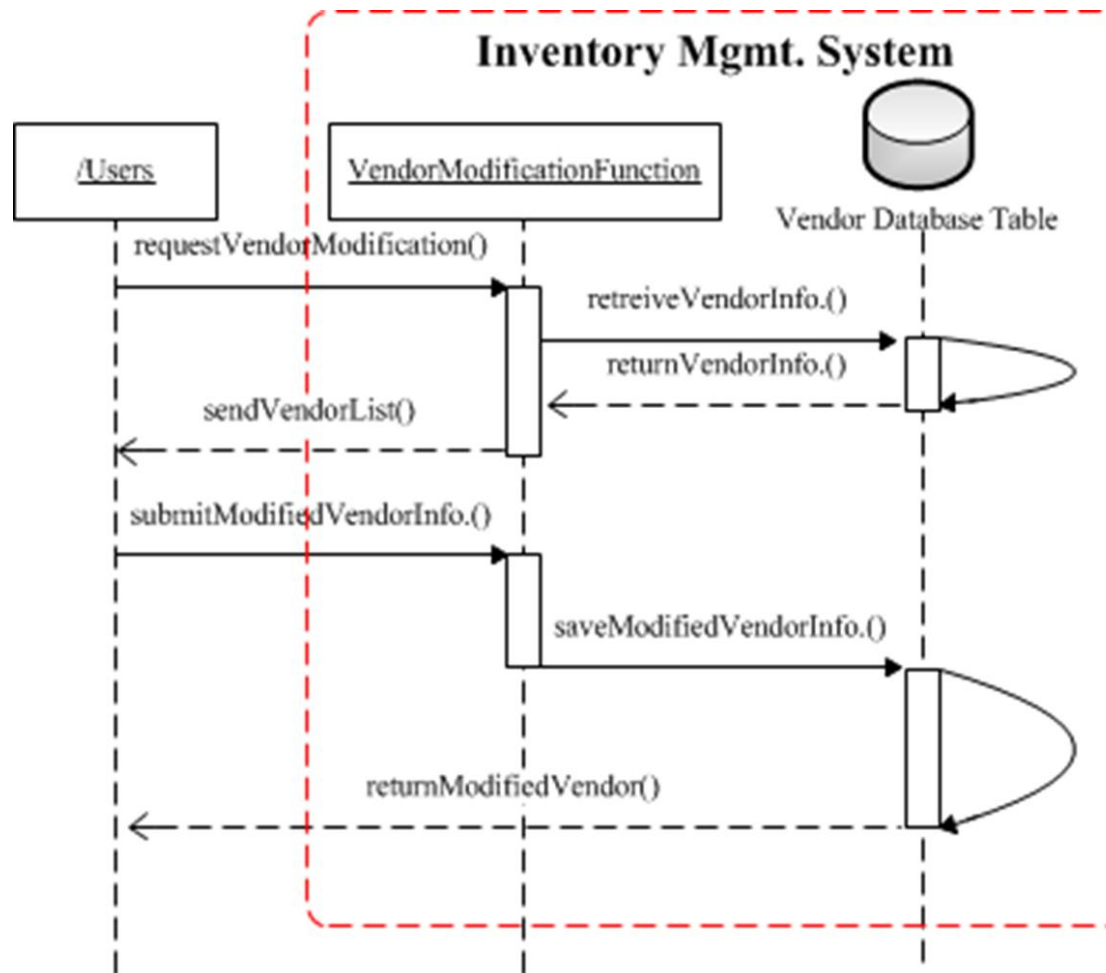


Figure 61 WBS 1.4.2 Vendor Modification Sequence Diagram

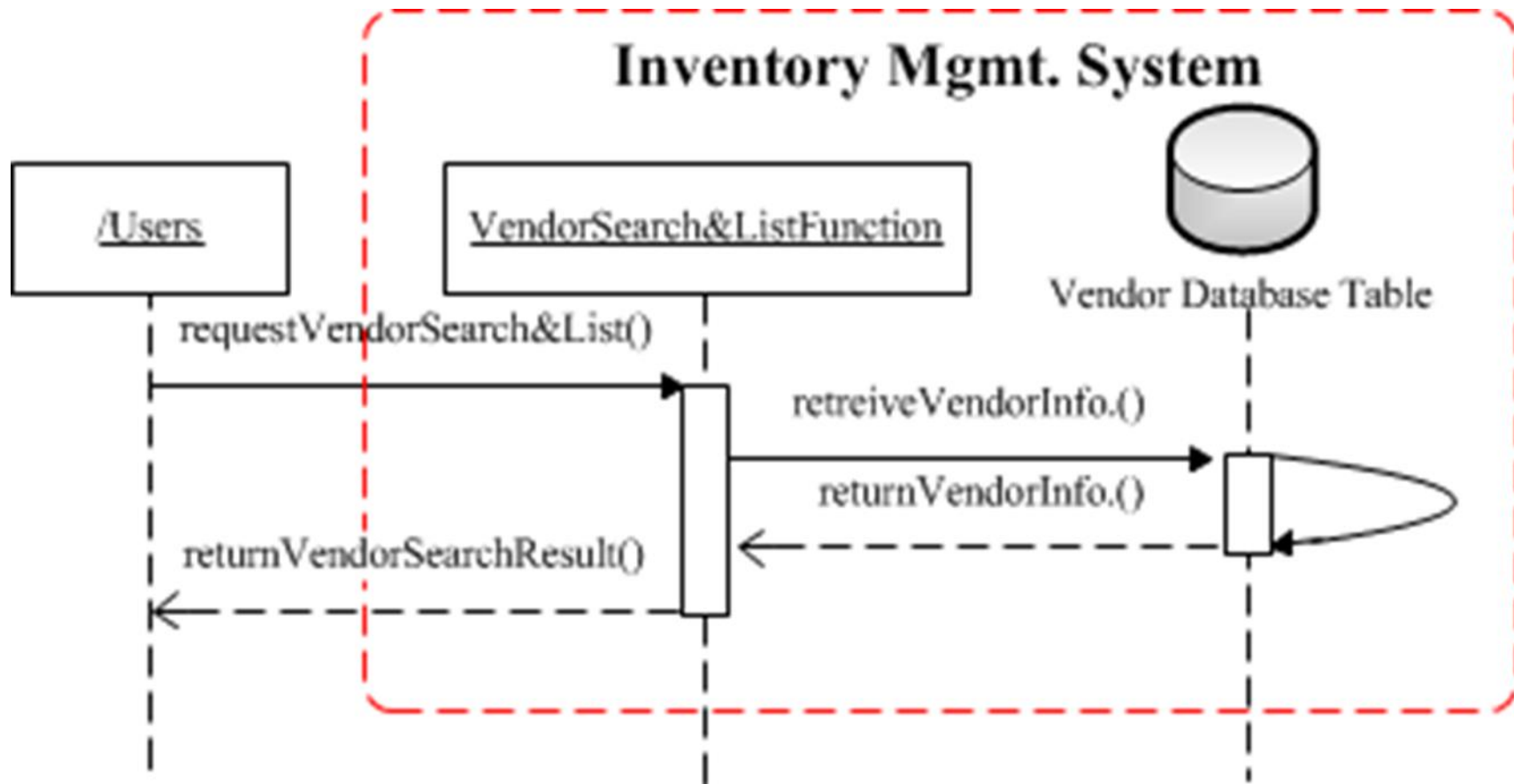


Figure 62 WBS 1.4.3 Vendor Search & List Sequence Diagram



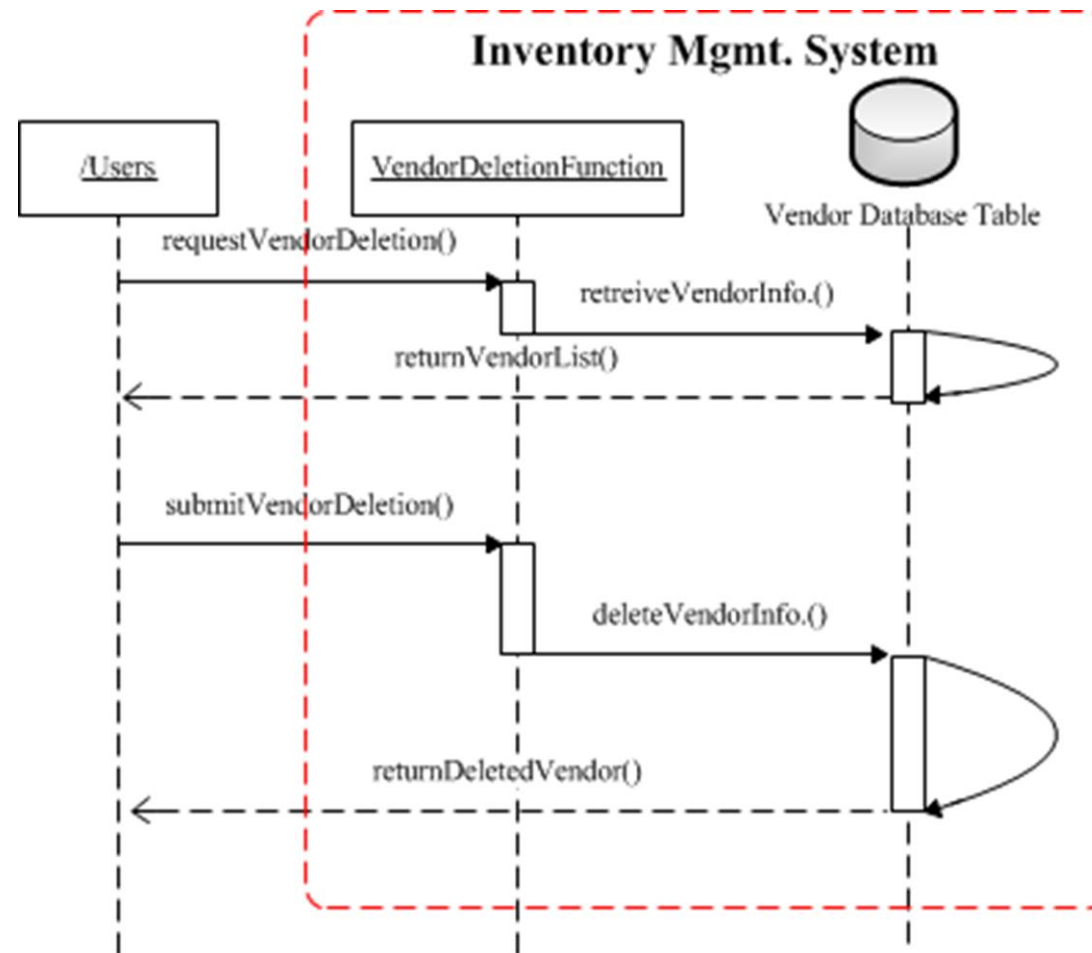


Figure 63 WBS 1.4.4 Vendor Deletion Sequence Diagram

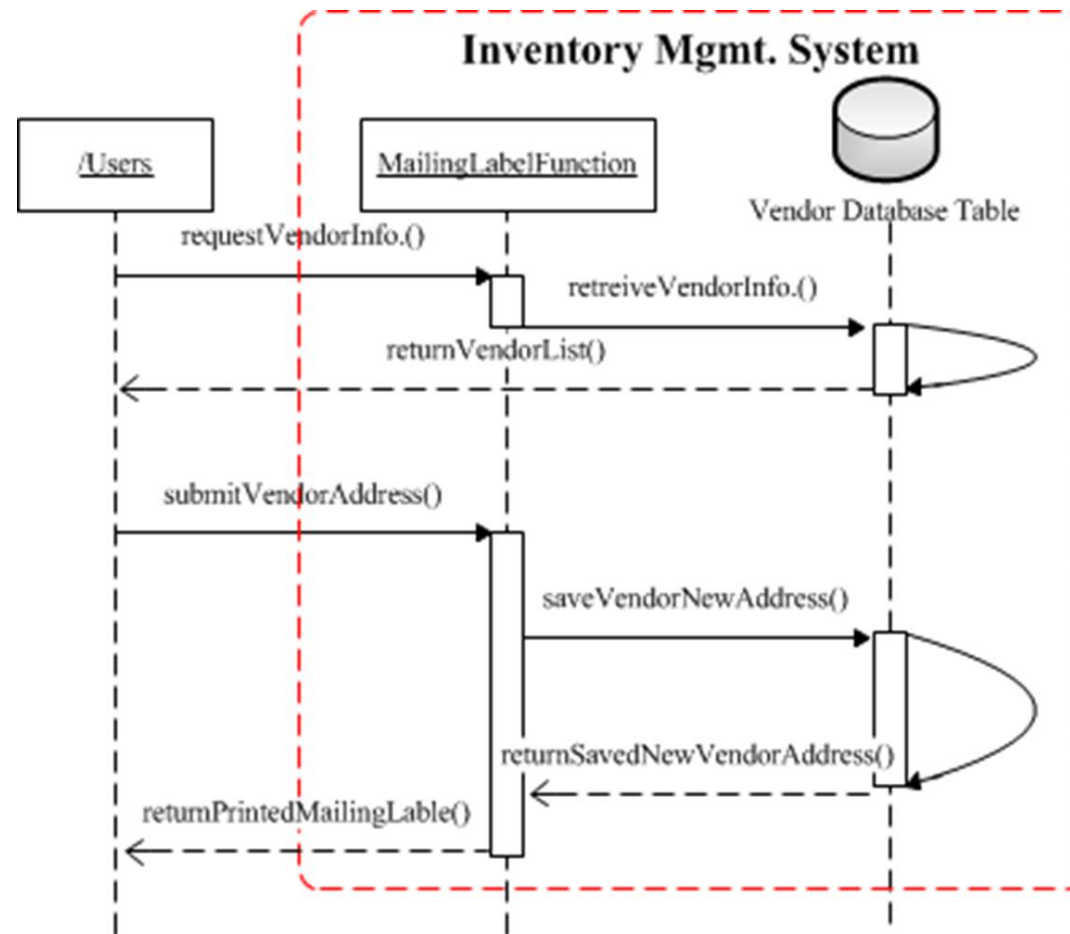


Figure 64 WBS 1.4.5 Mailing Label Sequence Diagram

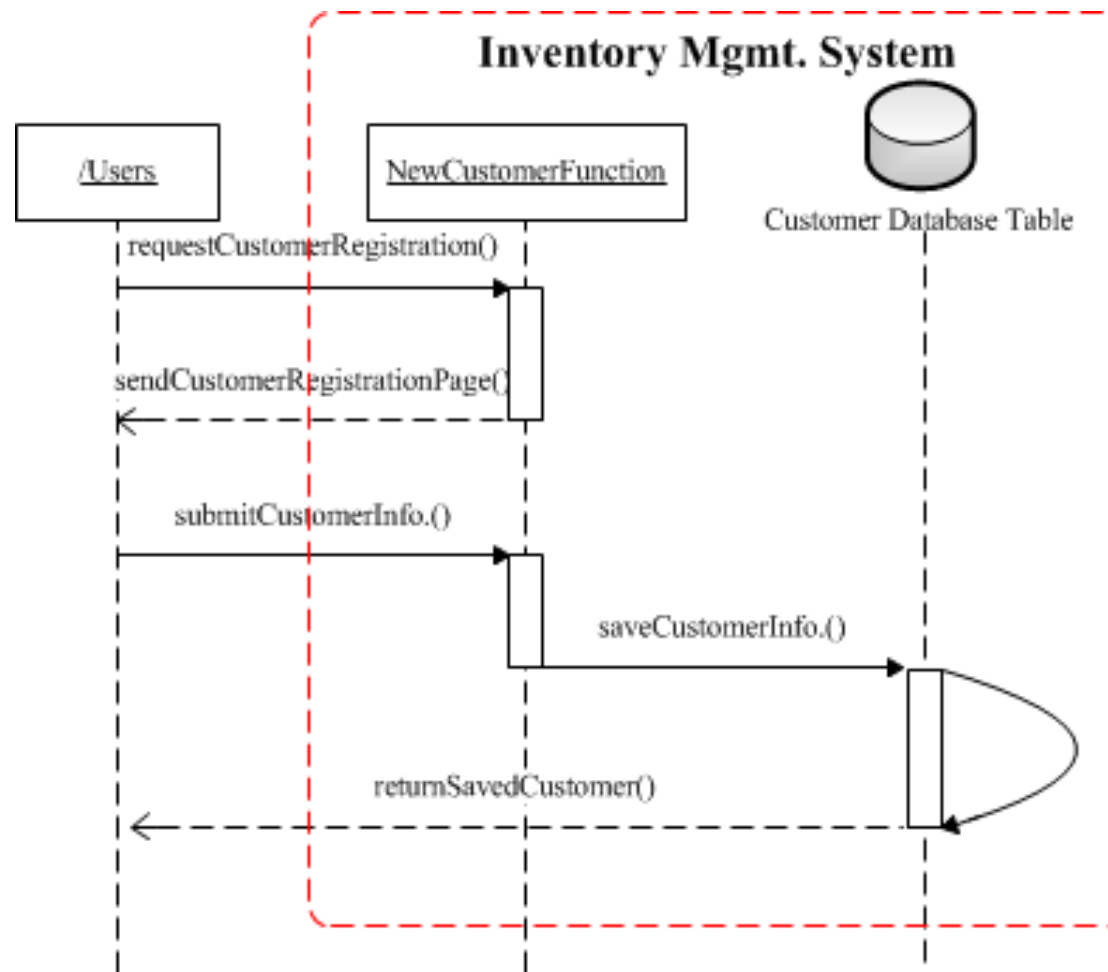


Figure 65 WBS 1.5.1 New Customer Sequence Diagram

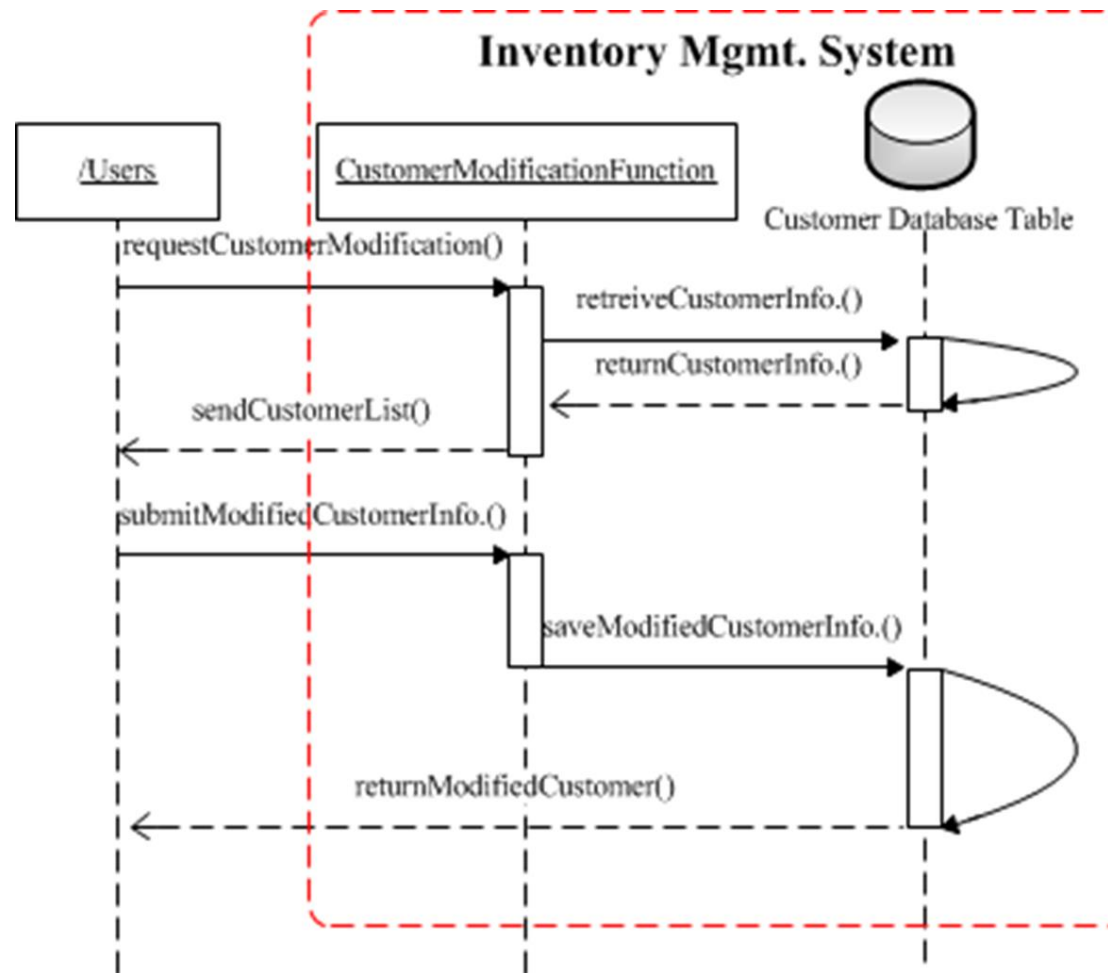


Figure 66 WBS 1.5.2 Customer Modification Sequence Diagram

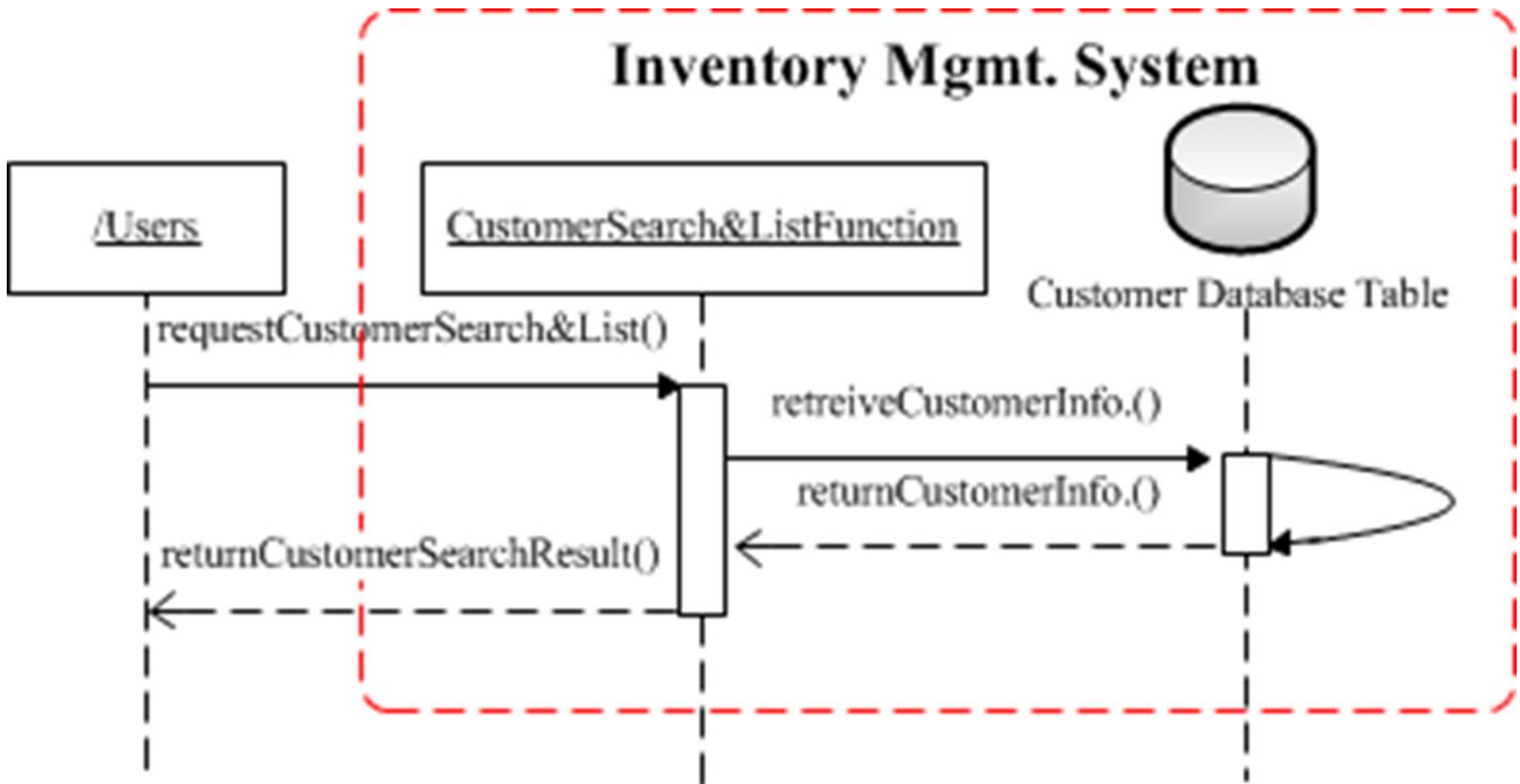


Figure 67 WBS 1.5.3 Customer Search & List Sequence Diagram

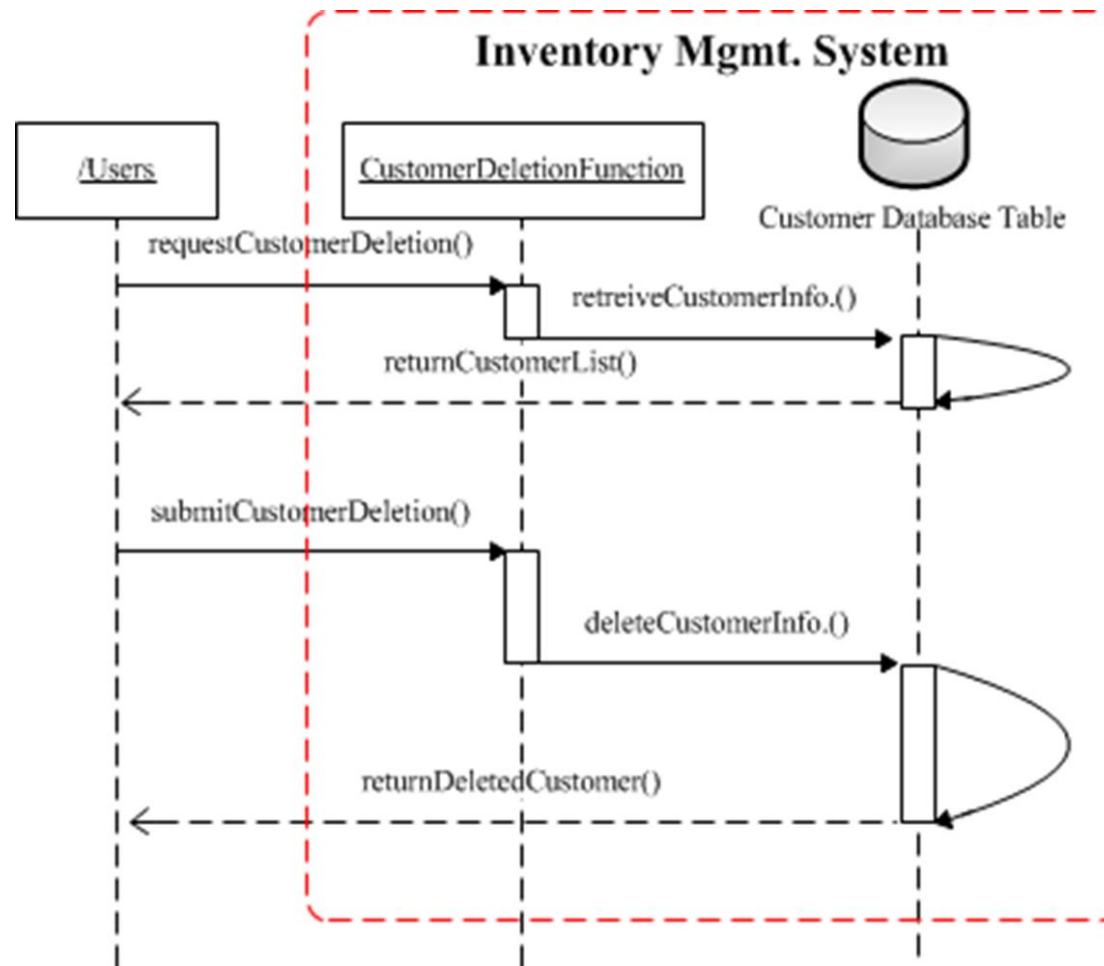


Figure 68 WBS 1.5.4 Customer Deletion Sequence Diagram

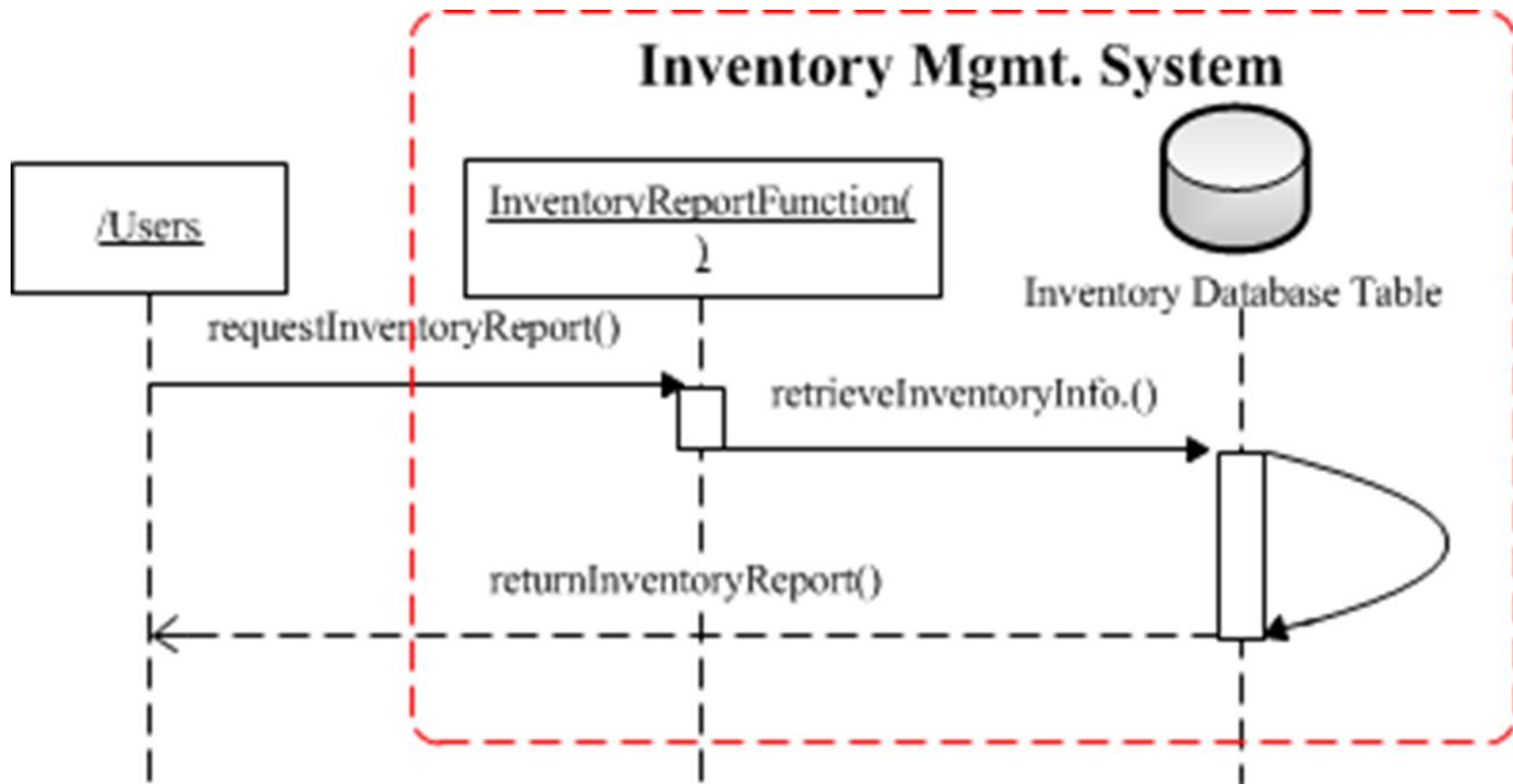


Figure 69 WBS 1.6.1 Inventory Report Sequence Diagram

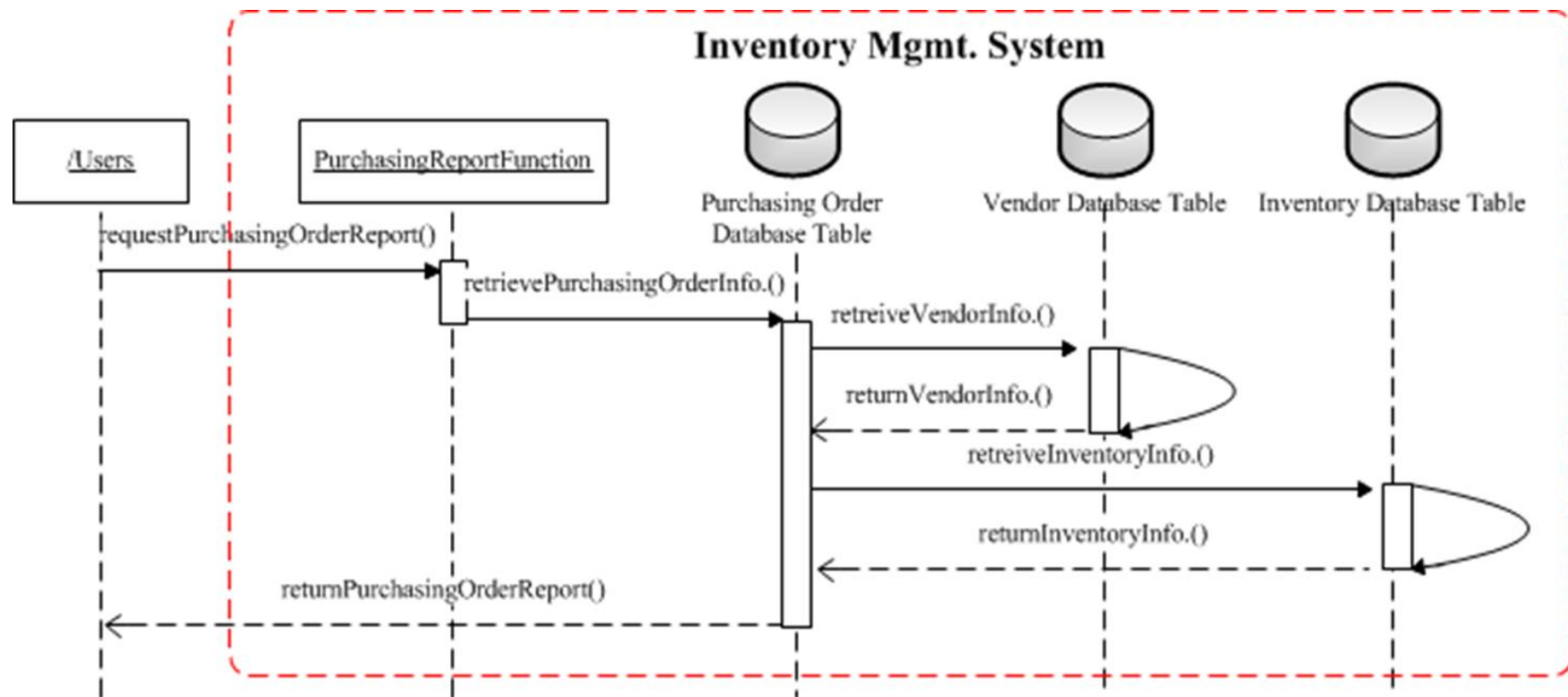


Figure 70 WBS 1.6.2 Purchasing Order Report Sequence Diagram



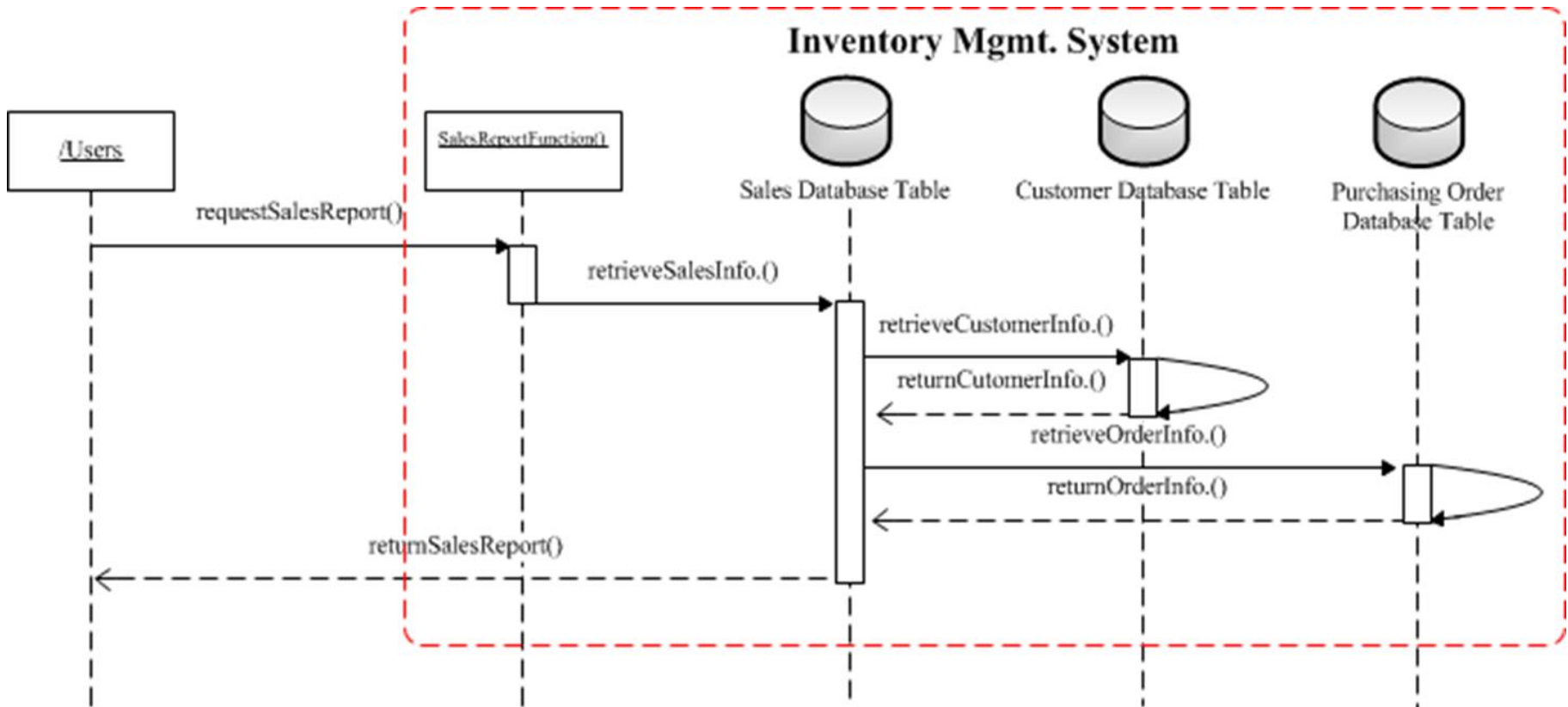


Figure 71 WBS 1.6.3 Sales Report Sequence Diagram

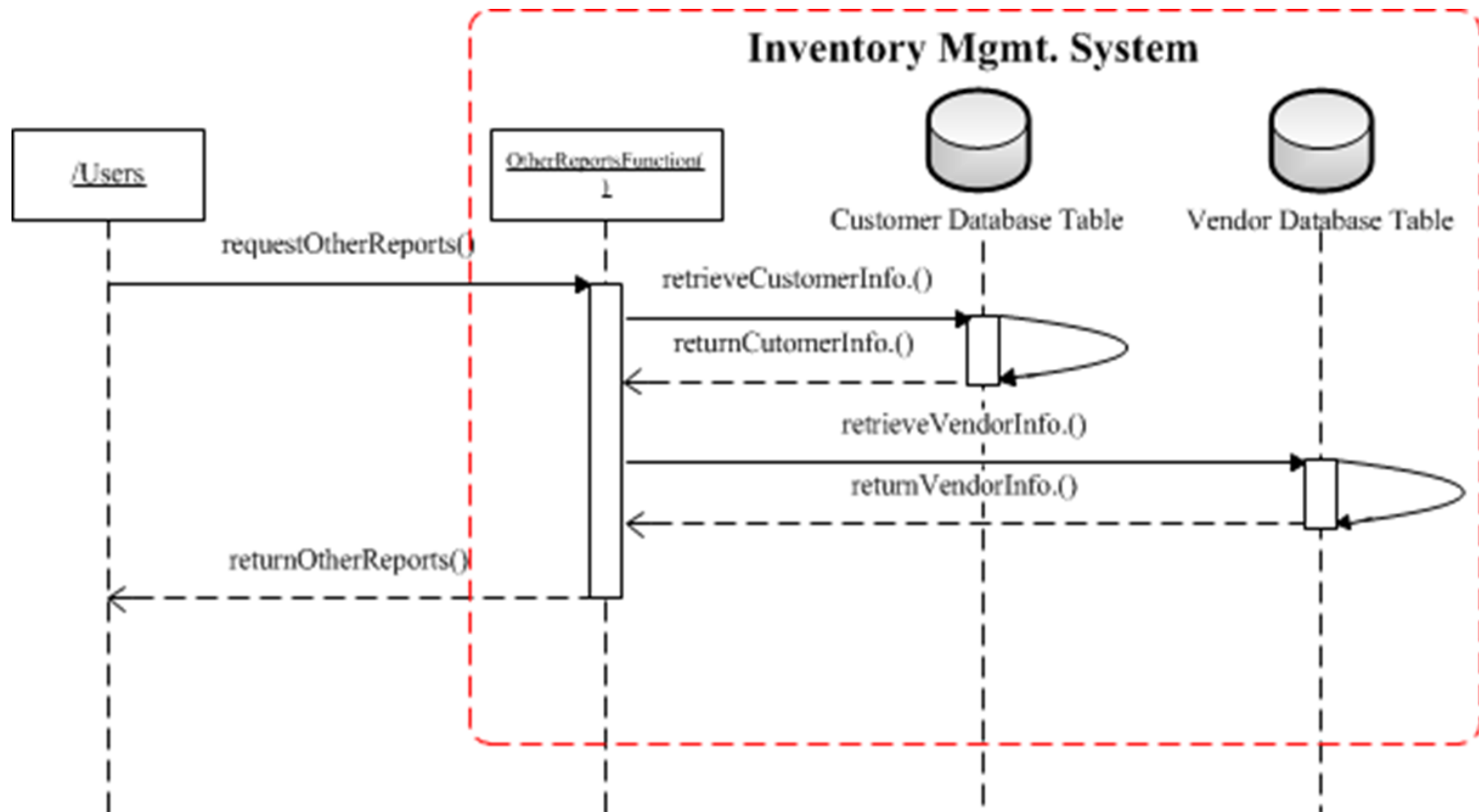


Figure 72 WBS 1.6.4 Other Reports Sequence Diagram

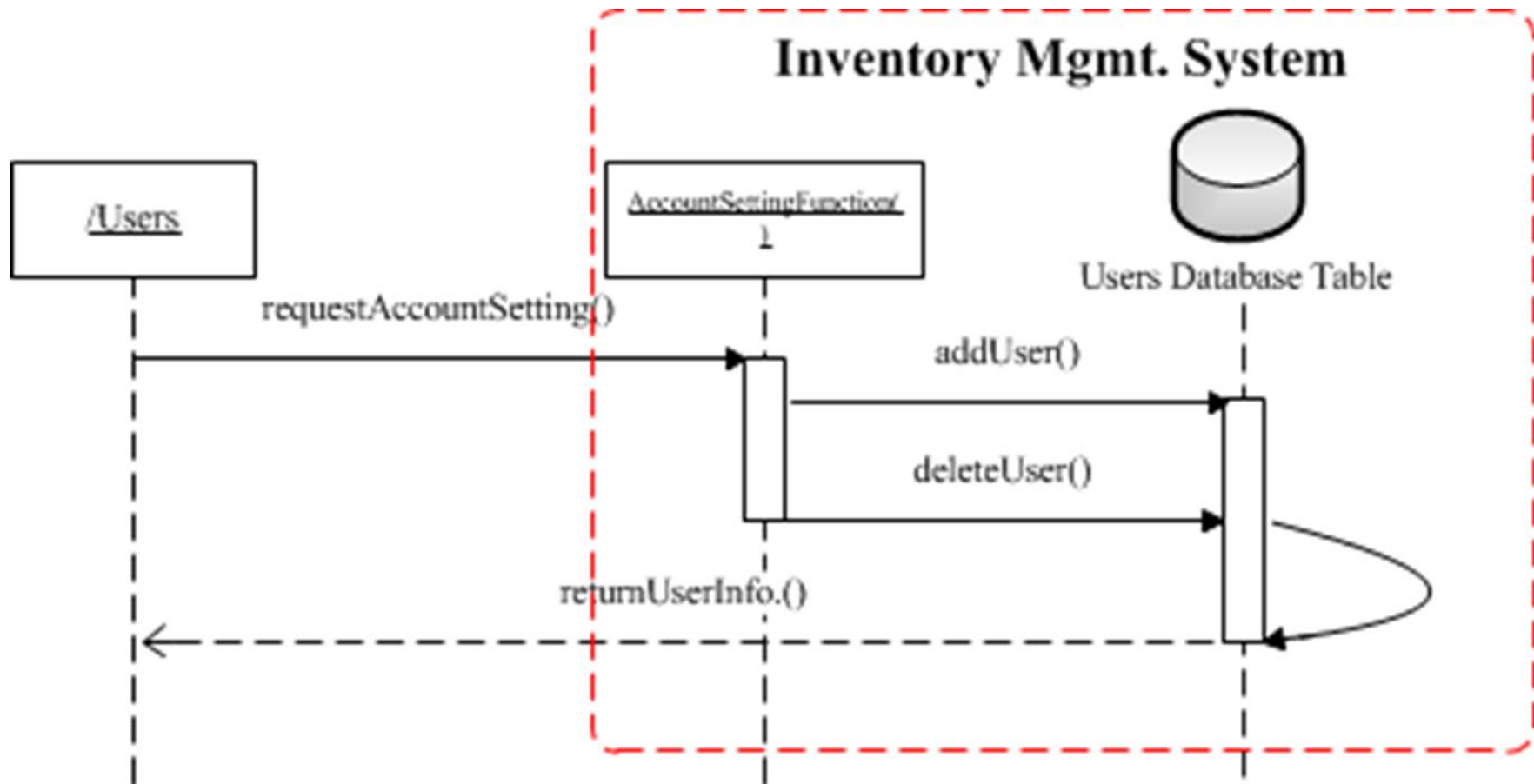


Figure 73 WBS 1.7.1 Account Setting Sequence Diagram

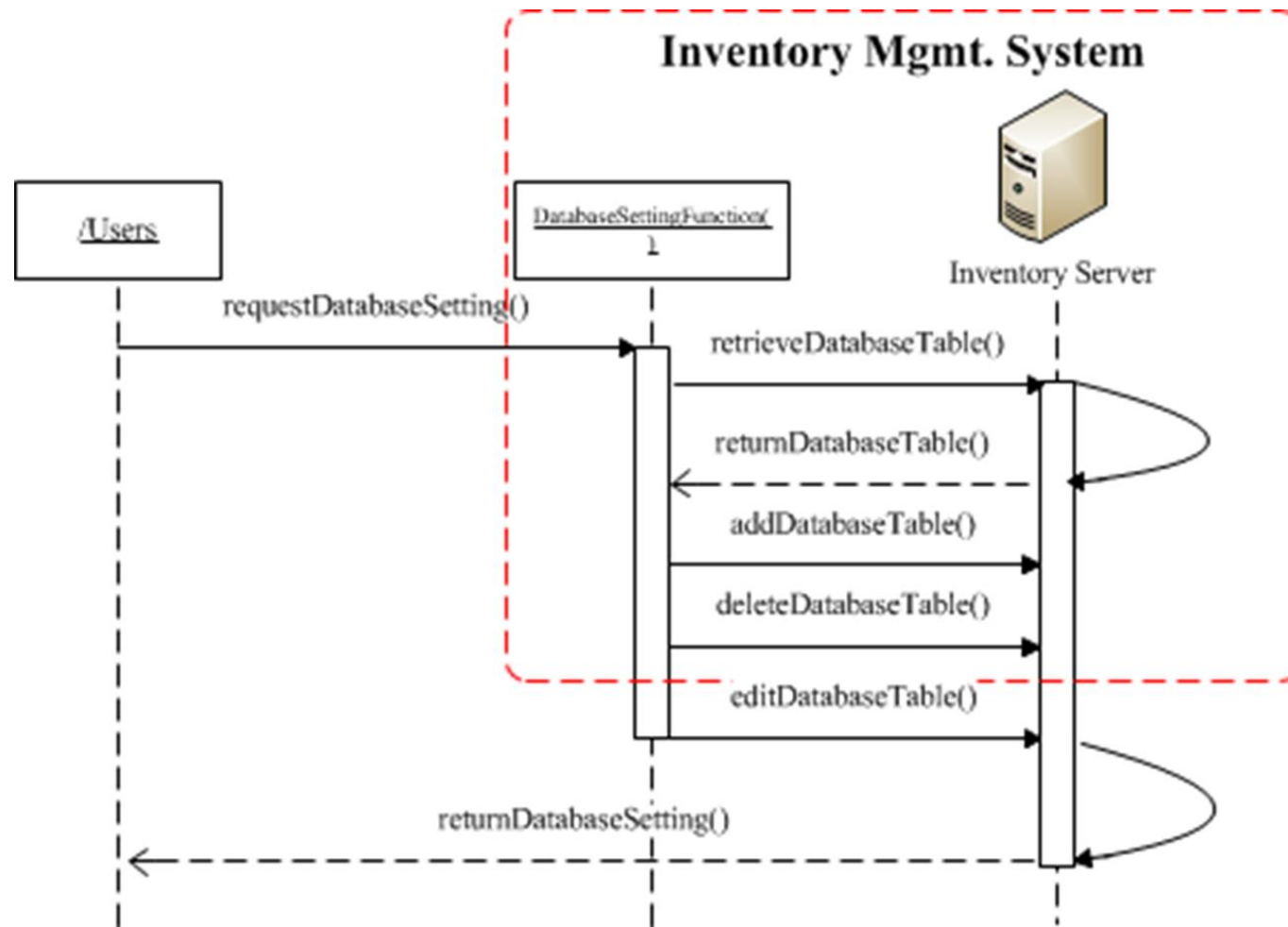


Figure 74 WBS 1.7.2 Database Setting Sequence Diagram

Once this is done, the AFP value is calculated using the equation shown in Section 3.1.3,

$$\text{Adjusted Function Point} = \text{Count total (UFP)} \times [0.65 + 0.01 \times \Sigma(\text{Fi})]$$

where the value of  $\Sigma(\text{Fi})$  is 63 according to eighth assumption. Once getting the Adjusted Function Point (AFP) value, it replaces the Unadjusted Function Point (UFP) value which was used to estimate project schedule in the proposal stage, and it is also possible to get normal calendar months. Table 24 below shows the results of the identified number of the functional components and the recalculated project schedule.

WP	EI	EO	EQ	ILF	ELF	UFP	AFP	Effort (Hours)	Duration (Month)
1.1.1	10	1	0	2	0	48	61.4	497.66	0.94
1.1.2	9	1	1	2	0	48	61.4	497.66	0.94
1.1.3	3	0	1	2	0	26	33.3	269.57	0.51
1.1.4	2	1	1	1	0	20	25.6	207.36	0.39
1.1.5	3	1	1	1	0	23	29.4	238.46	0.45
1.2.1	10	1	0	3	0	55	70.4	570.24	1.08
1.2.2	9	1	1	3	0	55	70.4	570.24	1.08
1.2.3	2	1	1	3	0	34	43.5	352.51	0.67
1.2.4	3	0	1	3	0	33	42.2	342.14	0.65
1.2.5	3	1	1	3	0	37	47.4	383.62	0.73
1.2.6	3	0	1	3	0	33	42.2	342.14	0.65
1.3.1	10	1	0	3	0	55	70.4	570.24	1.08
1.3.2	9	1	1	3	0	55	70.4	570.24	1.08
1.3.3	3	1	1	1	0	23	29.4	238.46	0.45
1.3.4	3	0	1	3	0	33	42.2	342.14	0.65
1.4.1	11	1	0	1	0	44	56.3	456.19	0.86
1.4.2	10	1	1	1	0	44	56.3	456.19	0.86
1.4.3	3	0	1	1	0	19	24.3	196.99	0.37
1.4.4	2	1	1	1	0	20	25.6	207.36	0.39
1.4.5	7	1	1	1	0	35	44.8	362.88	0.69
1.5.1	11	1	0	1	0	44	56.3	456.19	0.86
1.5.2	10	1	1	1	0	44	56.3	456.19	0.86
1.5.3	3	0	1	1	0	19	24.3	196.99	0.37
1.5.4	2	1	1	1	0	20	25.6	207.36	0.39
1.6.1	2	0	1	1	0	16	20.5	165.89	0.31
1.6.2	3	0	1	3	0	33	42.2	342.14	0.65
1.6.3	2	0	1	3	0	30	38.4	311.04	0.59
1.6.4	4	0	1	2	0	29	37.1	300.67	0.57
1.7.1	3	1	0	1	0	20	25.6	207.36	0.39
1.7.2	2	1	0	5	0	45	57.6	466.56	0.88
<b>Final Project Schedule</b>									20.42

**Table 24 Final Simulation Results**

As it is shown in Table 24, the calculated final project schedule is 20.42 months so that the probability of completing a project within this timeframe is a little more than 80% and less than 85% according to Table 22.

The overall contribution of this section is the presentation of a process for refining and/or validating schedule estimations made in the very early stage of a software development project, thus completing a “closing of the loop” of the methodology offered in Section 4.1. The use of UML diagrams and accurate AFP information should provide software project managers with greater confidence in their schedule estimates and an enhanced potential for project success. The results in this section have been published in the 15<sup>th</sup> IEEE / ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel / Distributed Computing (SNPD) (Kwon & Hammell, 2014).

### **4.3 Early-Stage Comparison of Software Development Project Types**

This section proposes to establish a framework for use in the earliest stages of a software project for comparing the various types of software development, thus allowing this information to be part of the project selection process. The possible types of software development projects are classified into three categories: new development, reuse-based without modification, and reuse-based with modification. This section proposes to utilize a decision tree framework that includes quantitative risk management related to project cost and possible project success. The quantitative risk analysis consists of cost estimation, possibility (success rate), Net Present Value (NPV), and Expected Monetary Value (EMV). Probabilistic schedule estimation based on the International Function Point User Group (IFPUG) function point

analysis and the International Software Benchmarking Standards Group (ISBSG) regression equations (on which we previously conducted research) is also fundamental to the project cost estimation of each decision node. The use of the proposed framework provides more accurate and objective comparisons among the types of software development in the early stages of the software project life cycle.

To understand the framework of this section, which is shown in Figure 10, the fundamental concept of a decision tree must be examined. Basically, a decision tree refers to an influence diagram where decisions, chance nodes, and results are linked from left to right by lines (Olivas, 2007). Decisions, chance nodes, and end results are represented by nodes; squares for decisions, circles for chance nodes, and triangles for end results.

#### **4.3.1 Cost Estimation to Design a Decision Tree**

A decision tree begins with a square on the left side as the root node or parent node that represents the first set of decision alternatives (see Figure 75 for an example decision tree) (Olivas, 2007). One thing to keep in mind is that there must be at least two or more decision alternatives so that chance nodes, which represent a possibility, can be identified. Additionally, each decision alternative with the investment cost should be extended to the right from the root node and connected by a line or branch.

Developing the decision tree is the first step in building the research model for this portion of the research. Cost estimation of each decision alternative must now be considered (the second step). As shown in Table 16, it needs to focus on the factors in each equation for the three decision alternatives. For the first category



(development cost with no reuse), labor costs refer to human resource efforts; such effort is one of the important criteria for a cost estimation technique which was identified by various different research models (Kwon & Hammell, 2013). For this reason, if appropriate information such as project size, Project Delivery Rate (PDR), and so on are given, it is possible to calculate labor costs which will be based on Person Month (PM) as discussed in Section 3.1.6 (Kwon & Hammell, 2013). Then, PM should be converted into normal calendar months or hours, and if the individual human resource's hour rate is given then labor costs can be estimated. The hardware purchasing expense can be simply based on market prices. It should also be noted that additional software could be needed for the new development effort and this cost must be included.

The factors in the equation for the reuse-based without modification alternative (the second alternative) are  $Cost_{search}$  and probability. There are two considerations with respect to the  $Cost_{search}$  factor. Firstly, the search operation could be manually conducted by a programmer with the goal of finding similar code from a database owned/controlled by the organization. However, a second approach could be to use Commercial-Off-the-Shelf (COTS) software. The COTS software would not require modification, which matches the reuse-based without modification category. For this reason, project costs should be estimated in terms of human resource efforts and COTS purchasing expense.

For the third alternative (reuse-based with modification), the  $Cost_{adapt}$  factor must be examined. It is related to revising programming codes that are also performed by a programmer, so human resource efforts with consideration of an hourly rate and

the time required to revise code should be estimated in terms of project costs. Table 25 shows the revised cost estimation equations.

Category	Equation	Note
New Development	Labor costs + (Hardware purchasing expense) + (Software purchasing expense)	Labor costs: Calculate Person Month and convert it into normal calendar month. Then, estimate labor costs using the given hourly rate  (): Optional. This factor can be applied or not.
Reuse-based without modification	Labor Costs + (COTS purchasing expense)	1. Labor Costs: Costs for own database inquiry and deployment based on hourly rate  2. COTS purchasing expense: optional
Reuse-based with modification	$Cost_{search} + (COTS \text{ purchasing expense}) + Cost_{adapt}$	1. $Cost_{search}$ : Costs for own database inquiry based on hourly rate  2. COTS purchasing expense: optional  3. $Cost_{adapt}$ : the total amount costs of revising components or module

**Table 25 Revised Cost Estimation Equations**

#### 4.3.2 Calculate the Success Rate

The third step as shown in Figure 10 is to calculate the success rate of each decision alternative. As described in Section 3.3.1, a decision tree is driven by a probability and Expected Monetary Value (EMV). The problem is that probability is measured by subjective expert judgment, which this work is trying to replace with objective methods whenever possible. This research proposes substituting a possibility measure for the probability term, which will be provided by the success rate calculations. The success rate of each decision alternative will be calculated based on the equation and criteria shown in Table 17 (Maglyas, 2009).

### 4.3.3 Profitability Estimation

The last step is to determine the EMV of each end result. The first part of this step involves profitability estimation. While there are both numeric and non-numeric benefits involved (Wagner, Xie, Rubel-Otterbach, & Sell, 2007) for the reasons provided in Section 3.3, this work only considers numeric benefits. Further, a decision tree requires visible and quantifiable future monetary benefits only and the Net Present Value (NPV) can be used for estimating quantifiable benefits (Wagner et al., 2007). The fundamental concept of the NPV is shown below.

$$\text{NPV} = \sum_{t=0}^n \frac{\text{Cash flow}}{(1+d)^t} + I_0 \quad (\text{Wagner et al., 2007; Erdogmus, 1999})$$

where  $t$  means the number of years,  $d$  means the discount rate, *cash flow* means the sum of cash inflow and outflow, and  $I_0$  means initial investment.

Initial investment is basically same as project cost estimation and it is calculated according to Table 25, but the problem is how to calculate cash inflow and outflow. Cash inflow represents the total expected future revenues from sales, licenses, royalties, and direct cost savings from the end product, and cash outflow is total expected expenditure from operating the end product (Wagner et al., 2007). Additionally, the appropriate discount rate should be determined with consideration of a project risk, but discount rates for software investment vary widely in the range of 10 to 25 percent (Huizingh & Vrolijk, 1997). Since how to determine a discount rate is beyond the scope of this research, it is assumed that an appropriate discount rate can be provided.

Once NPV and the success rate are calculated, the last step is to finish the Expected Monetary Value (EMV) calculation. EMV is a statistical concept which calculates average outcome when a scenario in the future may or may not occur (PMI, 2013). If EMV is considered as an opportunity, it has a positive value; but when considered as a threat, it has a negative value (PMI, 2013). The equation for calculating EMV is the value of each possible outcome times the possibility of its occurrence (PMI, 2013). For this calculation this research uses NPV as the value of each outcome and use the success rate as the possibility of occurrence. Thus, the equation of EMV for this framework is  $EMV = NPV * \text{success rate}$ .

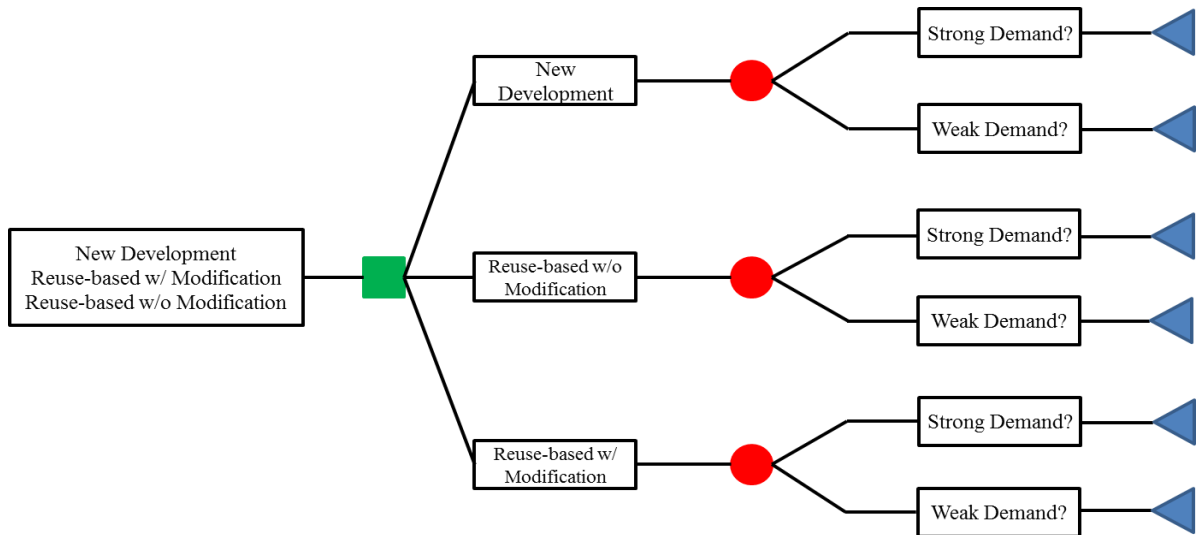
#### **4.3.4 Demonstration**

For the simulation in this section, the same example is used as stated in Section 4.1. From this example, project duration based on 30 work packages is estimated, and this schedule estimation results in project cost estimation that can be used for the first decision alternative (new development) as well as for initial investment in the EMV determination. Yet, there are additional assumptions that must be made in this scenario so that cost estimation for all decision alternatives can be made. These are as follows:

- The first assumption is that 3 programmers who perform every work package participate as a single team in a project and the hourly rate of each programmer is \$20/hr, \$18/hr, and \$16/hr; all decision alternatives have the same hourly rate.

- The second assumption is that the duration of the second decision alternative (reuse-based without modification), and the third decision alternative (reuse-based with modification) is  $1/3$  and  $1/2$ , respectively of the first decision (new development) alternative.
- The third assumption is that there is no hardware and software purchasing expense for new development, but there is COTS purchasing expense of \$30,000 for the second and third decision alternatives.
- The fourth assumption is that the discount rate of NPV is 15% and cash inflow and outflow are calculated based on our own discretion.
- The fifth assumption is that the criteria for the success rate of each decision alternative are also measured by our own discretion.

Using these assumptions and methodology, a decision tree with consideration of 3 decisions and chance events needs to be drawn as depicted in Figure 75. In this diagram, each chance node consists of two different demands, strong and weak, which are uncertain. Furthermore, this diagram is depicted based on a decision tree model provided in (PMI, 2013).



**Figure 75 Decision Tree for Each Decision Alternative**

The next step is to estimate initial project cost of each decision alternative as initial investment. According to the first and third assumption, initial investment for the first decision alternative can be calculated; in other words, since it is required to consider the labor cost only as the main input in the equation, all programmers are simply assigned to each work package and entered hourly rate into Microsoft Project 2010. The result is shown in Table 26.

WP	Duration (Month)	Cost	WP	Duration (Month)	Cost
1.1.1	0.9	\$7,776	1.4.1	1.0	\$8,640
1.1.2	1.0	\$8,640	1.4.2	0.9	\$7,776
1.1.3	0.7	\$6,048	1.4.3	0.6	\$5,184
1.1.4	0.7	\$6,048	1.4.4	0.5	\$4,320
1.1.5	0.7	\$6,048	1.4.5	0.7	\$6,048
1.2.1	0.9	\$7,776	1.5.1	1.0	\$8,640
1.2.2	1.1	\$9,504	1.5.2	1.0	\$8,640
1.2.3	0.6	\$5,184	1.5.3	0.6	\$5,184
1.2.4	0.5	\$4,320	1.5.4	0.6	\$5,184
1.2.5	0.5	\$4,320	1.6.1	0.7	\$6,048
1.2.6	0.5	\$4,320	1.6.2	0.5	\$4,320
1.3.1	0.9	\$7,776	1.6.3	0.6	\$5,184
1.3.2	0.9	\$7,776	1.6.4	0.6	\$5,184
1.3.3	0.6	\$5,184	1.7.1	0.6	\$5,184
1.3.4	0.5	\$4,320	1.7.2	0.9	\$7,776
Grand Total			\$188,352		

**Table 26 Cost Estimation of New Development**

As it is shown in Table 26, initial investment of the first decision alternative is \$188,352. Thus, according to the second assumption, initial investment for the second and third decision alternative is \$62,787 and \$94,176.

The third step is to measure the success rate of each decision alternative. These measurements are based on the fifth assumption and the result of measuring the success rate is shown in Table 27.

Criteria	New Development	Reuse-based w/o Modification	Reuse-based w/ Modification
User involvement	0 (19*0)	0 (19*0)	19 (19*1)
Clear statement of requirements	16 (16*1)	0 (16*0)	16 (16*1)
Realistic expectation	15 (15*1)	15 (15*1)	15 (15*1)
Competent staff	0 (11*0)	11 (11*1)	0 (11*0)
Clear vision and objectives	10 (10*1)	10 (10*1)	10 (10*1)
Executive Management support	9 (9*1)	9 (9*1)	9 (9*1)
Proper planning	8 (8*1)	0 (8*0)	8 (8*1)
Smaller project milestones	0 (6*0)	0 (6*0)	0 (6*0)
Ownership	3 (3*1)	0 (3*1)	0 (3*1)
Hard-working, focused staff	3 (3*1)	3 (3*1)	3 (3*1)
<b>Total</b>	64	48	80

**Table 27 Success Rate of Each Decision Alternative**

The resulting single number shown for each alternative can be considered as a percentage. Additionally, the value of weak demand can be simply calculated by 100 minus the value of success rate because the value of success rate corresponds to the value of strong demand, which is considered as an event probability (framed as possibility in this paper).



The fourth step is to calculate NPV based on the fourth assumption, and cash inflow, outflow, the discount rate, and Net Cash Flow (NCF) are discretionally determined. Before conducting this step, the source of cash inflow and outflow need to be considered. The source of cash inflow for new development can be sales, licenses, and cost savings from the end product, but the source of cash outflow can be operation and maintenance cost. The source of cash inflow for reuse-based without modification is direct cost savings, with the source of cash outflow being operation cost. The source of cash inflow for reuse-based with modification is direct cost savings, and the source of cash outflow is operation and maintenance cost. Based on this, the fourth assumption, and the NPV equation, it is possible to make 5 years NPV data as shown in Table 28 through Table 33.

<b>Year</b>	<b>Investment</b>	<b>Cash Inflow</b>	<b>Cash Outflow</b>	<b>NCF</b>	<b>NPV</b>
0	-188,352			-188,352	-188,352
1		100,500	-14,000	86,500	75,217
2		103,700	-15,000	88,700	67,070
3		125,200	-19,000	106,200	69,828
4		137,900	-21,000	116,900	66,838
5		157,650	-25,000	132,650	65,950
<b>TOTAL</b>	-188,352	624,950	-94,000	342,598	<b>156,552</b>

**Table 28 NPV for New Development Strong Demand**

Year	Investment	Cash Inflow	Cash Outflow	NCF	NPV
0	-188,352			-188,352	-188,352
1		55,760	-7,500	48,260	41,965
2		58,500	-8,200	50,300	38,034
3		60,550	-9,500	51,050	33,566
4		63,500	-11,000	52,500	30,017
5		78,500	-12,500	66,000	32,814
<b>TOTAL</b>	-188,352	316,810	-48,700	79,758	<b>-11,956</b>

**Table 29 NPV for New Development Weak Demand**

Year	Investment	Cash Inflow	Cash Outflow	NCF	NPV
0	-62,787			-62,787	-62,787
1		26,800	-3,400	23,400	20,348
2		29,800	-4,650	25,150	19,017
3		34,000	-6,700	27,300	17,950
4		35,700	-8,900	26,800	15,323
5		38,000	-11,600	26,400	13,125
<b>TOTAL</b>	-62,787	164,300	-35,250	66,263	<b>22,976</b>

**Table 30 NPV for Reuse-based w/o Modification Strong Demand**

Year	Investment	Cash Inflow	Cash Outflow	NCF	NPV
0	-62,787			-62,787	-62,787
1		13,200	-3,400	9,800	8,522
2		15,000	-4,650	10,350	7,826
3		17,400	-6,700	10,700	7,035
4		19,200	-8,900	10,300	5,889
5		21,300	-11,600	9,700	4,823
<b>TOTAL</b>	-62,787	86,100	-35,250	-11,937	<b>-28,692</b>

**Table 31 NPV for Reuse-based w/o Modification Weak Demand**

Year	Investment	Cash Inflow	Cash Outflow	NCF	NPV
0	-94,176			-94,176	-94,176
1		73,800	-9,200	64,600	56,174
2		76,200	-9,950	66,250	50,095
3		81,100	-11,200	69,900	45,960
4		83,400	-13,600	69,800	39,908
5		87,700	-15,450	72,250	35,921
<b>TOTAL</b>	-94,176	402,200	-59,400	248,624	<b>133,882</b>

**Table 32 NPV for Reuse-based w/ Modification Strong Demand**

Year	Investment	Cash Inflow	Cash Outflow	NCF	NPV
0	-94,176			-94,176	-94,176
1		36,400	-5,760	30,640	26,643
2		38,900	-6,500	32,400	24,499
3		42,600	-7,750	34,850	22,914
4		47,900	-9,080	38,820	22,195
5		50,500	-13,160	37,340	18,565
<b>TOTAL</b>	-94,176	216,300	-42,250	79,874	<b>20,641</b>

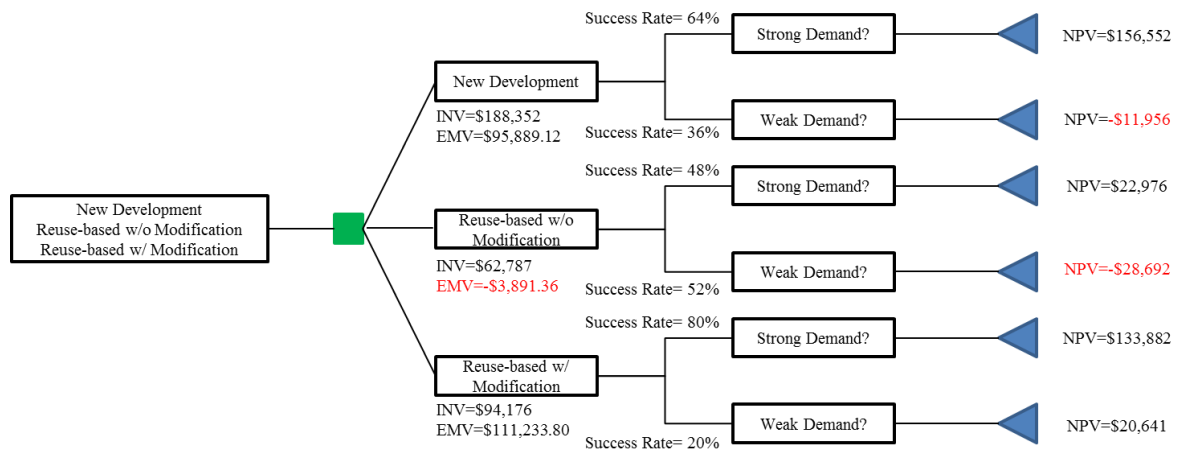
**Table 33 NPV for Reuse-based w/ Modification Weak Demand**

With the calculation of NPV for each decision alternative completed, the last step is to calculate EMV using the equation (NPV \* the success rate) of each chance node.

The results are:

1. New development EMV: **\$95,889.12** = (\$156,552 \* 64%) + (-\$11,956 \* 36%)
2. Reuse-based w/o modification EMV: **-\$3,891.36** = (\$22,976\*48%) + (-\$28,692\*52%)
3. Reuse-based w/ modification EMV: **\$111,233.8** = (\$133,882\*80%) + (\$20,641\*20%)

Therefore, the completed decision tree is shown in Figure 76.



**Figure 76 Final Decision Tree**

A noteworthy result here is that typical comparison methodologies using a business management point of view would have selected new development as the preferred project type based on Net Present Value (NPV). However, from a project management viewpoint, reuse-based with modification is recommended using our framework since it has the highest overall Expected Monetary Value (EMV). The EMV decision criterion focuses not only on financial management but also quantitative risk analysis using the success rate. The results in this section will be published as part of the 14th IEEE / ACIS International Conference on Computer and Information Science (ICIS 2015).

## **Chapter 5 Conclusion, Limitations, and Improvements**

The goal of this dissertation was to develop an objective methodology and framework in terms of 3 research areas: schedule estimation in the early stage (perhaps in the proposal preparation stage), schedule refinement / verification in the planning stage, and early-stage comparison of software development project types. This chapter describes the contributions, the limitations of this research, and possible future work. The contributions are connected back to the research questions which were identified in Chapter 1.

According to the identified first research questions (RQ1), “Can an objective technique / framework be developed to help estimate the project schedule in the proposal preparation stage?”, this research proposed a framework for estimating an objective project schedule in the proposal preparation stage and demonstrated its effectiveness. Based on a scenario-based case study, it was possible to generate greater probability accuracy by making the range of schedule estimation narrower using the CLT concept. Probabilistic schedule estimation based on inferential statistics was also 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 associated with the first research question is that the framework 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.

This research answered the second research question (RQ2), “Can the estimated project schedule in the proposal preparation stage be refined / verified in the planning stage using design documentation?”. This research proposed the framework which was based on probabilistic schedule estimation in the early stage related to the first research question. The idea of using UML diagrams for software size estimation is not new. Foundational differences between this work and other oft cited similar work were discussed. Further, the methodology regarding the second research questions did not stop with size estimation but continued the process to predict the probabilistic project schedule. Thus, the framework combined Function Point Analysis (FPA), probability, and project management techniques to assist project managers with risk management early in the project life cycle phase.

The Decision Tree-based framework also clearly answered the third research question (RQ3), “Can an objective technique / framework be developed for early-stage comparison of software development project types?” A major contribution was a continuing work to provide objective (quantifiable) ways to assist project management decision making in the very early stages of the project life cycle. The reason for the focus on the concept of quantitative risk analysis from the beginning of this work was that its key benefit was to reduce project uncertainty for supporting decision making. There was no doubt that not enough information existed in the early stage before initiating a project; as a result, there were many difficulties with determining the right type of software development. Therefore, the decision tree-based framework which used the fundamental concept of Expected Monetary Value (EMV), possibility (referred as success rate), and Net Present Value (NPV) proposed

objective comparison of the software development project types such as new development, reuse-based without modification, and reuse-based with modification.

The common limitation of all those 3 research areas is that the proposed methodology and framework was demonstrated on an example project. To more accurately examine the efficacy of the methodology and framework and to verify that they can be used to produce more accurate estimations, they 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. For the RQ1, redundancy in the work packages or modules must be taken into account while creating the project WBS. For the RQ2, while deliverables-based WBS was used for answering this research question, how to create class operations-based WBS with consideration of the class relationship should be examined. Furthermore, database normalization to identify a more accurate number of the functional components should be considered. For the third research area, since the success rate of each decision alternative was scored subjectively, the objective evaluation technique of each criterion must be taken into account. Further, actual financial data for calculating NPV should be used when possible.

## Bibliography

- Alexander. A. J. (2004). How to Determine Your Application Size Using Function Points. *Borland Conference 2004*. Retrieved from <http://www.mif.vu.lt/~adamonis/psp/1213p/How%20to%20Determine%20Your%20Application%20Size%20Using%20Function%20Points.pdf>. pp. 1-17
- Asaka. C. N., Aila. O., Odera. O, and Abongo. B. E. (2012). Project selection and management implications in Kenyan local authorities. *Asian Journal of Business and Management Sciences*. Vol. 1 No. 10. pp. 65-75.
- Azath. H. and Wahidabanu. R. S. D. (2008). Function Point: A Quality Loom for the Effort Assessment of Software Systems. *International Journal of Computer Science and Network Security*. Vol. 8 No. 12. pp. 321-328.
- B. B. Pour, S. Noori, and R. H. Atashgah (2013). Project selection problem under uncertainty: An application of utility theory and chance constrained programming to a real case. *International Journal of Industrial Engineering Computations*, Vol. 4 Issue 3, pp. 373-386.
- Basha, S., & P, D. (2010). Analysis of Empirical Software Effort Estimation Models. *International Journal of Computer Science and Information Security*. Vol. 7 No. 3, pp. 68-77.
- Basili. V. R., Bailey. J., Rombach. H.D., and Joo. B.G. (1987). Software Reuse: A Framework for Research. Retrieved from <https://www.cs.umd.edu/~basili/publications/technical/T60.pdf>. pp. 1-21
- Boehm. B. & Abts. C. (2000). Software Development Cost Estimation Approaches – A Survey. *IBM Research*. pp. 2.



- Brooks, F. P. "No Silver Bullet – Essence and Accident in Software Engineering",  
*Journal Computer Archive*, Volume 20 Issue 4, 1987, pp. 10-19
- 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, pp. 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, pp. 628-640.
- 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.
- Erdogmus, H. (1999). Comparative evaluation of software development strategies  
based on Net Present Value. Retrieved from  
[http://hakanerdogmus.net/weblog/uploads/papers/EDSER1999-  
ErdogmusNPV.pdf](http://hakanerdogmus.net/weblog/uploads/papers/EDSER1999-ErdogmusNPV.pdf)
- Forselius, P. (2012). Software development program characteristics. Retrieved from  
<http://www.compaid.com/caiinternet/ezine/forselius-characteristics.pdf>. pp. 2-  
4
- Garmus, D. (2006). The Principles of Sizing and Estimating Projects Using  
International Function Point User Group (IFPUG) Function Points. Retrieved

from <http://www.compaid.com/caiinternet/ezine/garmus-principlesofsizing.pdf>. pp. 2

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

Haimes, Y. Y, Li, D., and Tulsiani, V. (1990), Multiobjective Decision-Tree Analysis, *Risk Analysis*, Vol. 10 No. 1, pp. 111-127.

Hoffer. J. A., George. J., and Valacich. J. (2010), *Modern Systems Analysis and Design 6th Edition*. Pearson  
[http://davidmlane.com/hyperstat/sampling\\_dist.html](http://davidmlane.com/hyperstat/sampling_dist.html)

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

Huizingh, E. K. R. E. and Vrolijk, H. C. J. (1997). Evaluation of Eight Project Selection Methods: the Case of Information Systems. *Estudios de Administracion*, Vol. 4, No. 2. pp. 1-19

Ibrar. M. (2013). UML Diagrams: an aid to Database Design Specification: A Review. *International Journal of Advanced Research in Computer Science and Software Engineering*, Vol. 3 No. 3. pp. 598-602.

Iorio. T. (2004). IFPUG Function Point analysis in a UML framework. *In conference proceedings of SMEF04*, Rome, Italy. pp. 1-10

Irawati. A. R. & Mustofa. K. (2012). Measuring Software Functionality Using Function Point Method Based On Design Documentation. *International Journal of Computer Science Issues*, Vol. 9, Issue 3 No. 1, pp. 124-130.

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

- K.S. J. & Vansantha. R. (2008). Cost Estimation Model For Reuse Based Software Products. *Proceeding of the International MultiConference of Engineers and Computer Scientists (IMECS 2008)*. Vol. 1. pp. 1-3.
- 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. IEEE. pp. 425-438.
- Kwon, D. & Hammell, R. J. (2013). Early Stage Probabilistic Software Project Schedule Estimation. *Journal of Information Systems Applied Research*. 6(4), pp. 31-48.
- Kwon, D. & Hammell, R. J. (2014). Refinement/Verification of Early Stage Probabilistic Software Project Schedules in the Planning Stage. *Proceeding of 15th IEEE / ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel / Distributed Computing (SNPD 2014)*, Las Vegas, NV, pp. 239-244.
- Kwon, D. & Hammell, R. J. (2014). Objective Framework for Early-Stage Comparison of Software Development Project Types. *Proceeding of 14th IEEE / ACIS International Conference on Computer and Information Science (ICIS 2015)*, June 28<sup>th</sup>~July 1<sup>st</sup>, Las Vegas, NV, Accepted.
- Lane, D. M. (2007). HyperStat Online Statistics Textbook. Retrieved from
- Lavazza. L. and Robiolo. G. (2010). Introducing the Evaluation of Complexity in Functional Size Measurement: a UML-based Approach. *Proceedings of the*

- 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*. No. 25. pp. 1-10.
- Levesque, G., Bevo, V., and Cao, D. T. (2008). Estimating Software Size with UML Models. *Proceedings of the 2008 C3S2E conference*. pp. 81-87.
- Lind, K., & Haldal, R. (2010). On the Relationship between Functional Size and Software Code Size. *Proceeding of the 2010 ICSE Workshop on Emerging Trends in Software Metrics*. ACM. pp. 47-52.
- 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.
- Longstreet, D. (2005). Fundamentals of Function Point Analysis. Retrieved from *Software Development Magazine*:  
<http://www.softwaremetrics.com/files/Fundamentals%20of%20Function%20Point%20Analysis.pdf>. pp. 2-4
- Maglyas, A. (2009). EVALUATING THE SUCCESS OF SOFTWARE DEVELOPMENT PROJECTS IN RUSSIA, UKRAINE, AND BELARUS. *Master's thesis paper*, 2009, pp.22-23
- Malik, A. A. (2010). *Quantitative and Qualitative Analyses of Requirements Elaboration for Early Software Size Estimation*. Retrieved from A Dissertation Presented to the FACULTY OF THE USC GRADUATE SCHOOL UNIVERSITY OF SOUTHERN CALIFORNIA:  
[http://csse.usc.edu/csse/TECHRPTS/PhD\\_Dissertations/files/Malik\\_Dissertation.pdf](http://csse.usc.edu/csse/TECHRPTS/PhD_Dissertations/files/Malik_Dissertation.pdf) pp. 7-13.

- Nasir, M. (2006). A Survey of Software Estimation Techniques and Project Planning Practices. *Proceeding of the Seventh ACIS International Conference. SNPD'06*. pp. 305-310.
- 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, pp. 12-21.
- Nickel, J., Ross, A. M., and Rhodes, D. H. (2009). Comparison of Project Evaluation Using Cost-Benefit Analysis and Multi-Attribute Tradespace Exploration in the Transportation Domain. *Second International Symposium on Engineering Systems*. pp. 2.
- 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, pp. 599-605
- Olivas, R. (2007). *Decision Tree: A Primer for Decision-Making Professional*. Retrieved from [http://www.lumenaut.com/download/decision\\_tree\\_primer\\_v5.pdf](http://www.lumenaut.com/download/decision_tree_primer_v5.pdf)
- Palisade Corporation (2010). *Guide to Using @ Risk: Risk analysis and Simulation Add-In for Microsoft Excel Version 5.7*. Palisade Corporation, New York.
- Phillips, J. (2009). *PMP Project Management Professional Study Guide Third Edition*. McGraw-Hill.
- 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 5<sup>th</sup> Edition)*. PMI, Newtown Square, PA.

- Project Management Solutions, Inc. (2012). *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/rempp/Papers/Smith&Wells\\_NERA06.pdf](http://www.umass.edu/rempp/Papers/Smith&Wells_NERA06.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*, pp. 61-69.
- THE NETHERLANDS SOFTWARE METRICS USERS ASSOCIATION (2008). *FPA applied to UML/Use cases Version 1.0*. NESMA. pp. 6-8
- Thomas, D. B., & Luk, W. (2008). Estimation of sample mean and variance for Monte-Carlo simulations. *ICECE Technology 2008*. Taipei: IEEE. pp. 89-96
- Total Metrics (2007). Methods for Software Sizing: How to Decide which Method to Use. Retrieved from [http://www.totalmetrics.com/function-point-resources/downloads/R185\\_Why-use-Function-Points.pdf](http://www.totalmetrics.com/function-point-resources/downloads/R185_Why-use-Function-Points.pdf). pp. 5
- Uemura, T., Kusumoty, S. & Inoue, K. (1999). Function Point Measurement Tool for UML Design Specification. *Proceeding METRICS '99 Proceedings of the 6th International Symposium on Software Metrics*, IEEE, pp. 62-69.
- Wagner, S., Xie, S., Rubel-Otterbach, M., and Sell, B. (2007). Profitability Estimation of Software Projects: A Combined Framework. *The First International Workshop on Software Productivity Analysis and Cost Estimation (SPACE'07)*.

- Zhou. Y., Yang. Y., Xu. B., Leung. H., and Zhou. X. (2014). Source code size estimation approaches for objected-oriented systems from UML class diagrams: A comparative study. *Information and Software Technology*. Vol. 56. Issue 2. pp. 220-237.
- Zivkovic. A., Rozman. I., and Hericko. M. (2005). Automated software size estimation based on function points using UML models. *Information and Software Technology*. Vol. 47. Issue 13. pp. 881-890.

## Curriculum Vita

NAME: Donghwoon Kwon



PROGRAM OF STUDY: Applied Information Technology

DEGREE AND DATE TO BE CONFERRED: Doctor of Science, 2015

Collegiate institutions attended	Dates	Degree	Date of Degree
TOWSON UNIVERSITY	September 2010	Doctor of Science	May 2015
UNIVERSITY OF ALASKA ANCHORAGE	September 2008	Master of Science	May 2010
DAEGU UNIVERSITY, SOUTH KOREA	March 2001	Bachelor of Engineering	February 2008

Professional publications and conference proceedings:

D. Kwon and R. Hammell, "Objective Framework for Early-Stage Comparison of Software Development Project Types," Proceeding of 14th IEEE / ACIS International Conference on Computer and Information Science (ICIS 2015), June 28th~July 1st, Las Vegas, NV, Accepted.

D. Kwon and R. Hammell, "Refinement/verification of early stage probabilistic software project schedules in the planning stage", 15th IEEE / ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Las Vegas, June 2014

D. Kwon, Y. Kwon, Y. Song, and R. Lee, "An Empirical Study on the Relationship between User Characteristics and Quality Factors for Effective Shopping Mall Websites Implementation," Software Engineering Research, Management and Applications, Springer International Publishing, ISSN: 1860-949X, 2014

D. Kwon and R. Hammell, "Early Stage Probabilistic Software Project Schedule Estimation," Journal of Information Systems Applied Research (JISAR) Vol. 6 No. 4, November 2013



Y. Kwon, J. Cui, and D. Kwon, "A Study on Comparison Analysis of the System Quality Factors between Korea and China Shopping Mall Websites," *Journal of Korea Multimedia Society* Vol. 16 No. 11, November 2013

D. Kwon, J. Cui, Y. Kwon, and C. Jin, "A Study on Comparison on the Technical Factors of Korea and China Shopping Mall Website," 14th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Hawaii, July 2013

Y. Kwon, J. Cui, and D. Kwon, "A Study on Effective Web Site Implementation using Reliability Measurement," *Journal of Korea Multimedia Society* Vol. 14 No. 6, June 2011

Y. Kwon, C. Yang, and D. Kwon, "A Study on E-Commerce Recommender Systems Based on LSI and Fuzzy Model," *Journal of Computer & Communication* Vol. 6 No. 1, January 2009

Y. Kwon, C. Yang, and D. Kwon, "A Fuzzy-SVD Model for Product Recommendation," *The International Conference on Multimedia, Information Technology, and its Applications (MITA 2008)*, Chiang Mai, Thailand, July 2008

