SOFTWARE REQUIREMENT CHANGE EFFORT ESTIMATION MODEL FOR SOFTWARE DEVELOPMENT PHASE

JALAL SHAH

UNIVERSITI TEKNOLOGI MALAYSIA

SOFTWARE REQUIREMENT CHANGE EFFORT ESTIMATION MODEL FOR SOFTWARE DEVELOPMENT PHASE

JALAL SHAH

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Doctor of Philosophy

Razak Faculty of Technology and Informatics
Universiti Teknologi Malaysia

DECEMBER 2018

ACKNOWLEDGEMENT

In preparing this thesis, I was in contact with many people, researchers, academicians, and practitioners. They have contributed towards my understanding and thoughts. In particular, I wish to express my sincere appreciation to my main thesis supervisor, Associate Professor Dr. Mohd Nazri bin Kama, for encouragement, guidance, critics and friendship. I am also very thankful to my co-supervisor Dr Nur Azaliah Binti Abu Bakar for her guidance, advices and motivation. Without their continued support and interest, this thesis would not have been the same as presented here.

My fellow postgraduate student should also be recognised for their support. My sincere appreciation also extends to all my colleagues and others who have provided assistance at various occasions. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space. I am grateful to all my family member.

ABSTRACT

Software requirement changes are a typical phenomenon in any software development project. Software requirement changes occur at all stages of software development life cycle. Allowing too many changes might cause delay in project delivery. Therefore, this study aims to improve the accuracy of software requirement change effort estimation (SRCEE) model for software requirement change in software development phase (SDP). It proposed a new algorithmic-based SRCEE model which could be used to improve the accuracy of the change effort estimation in SDP. This study analysed the existing SRCEE models and change impact analysis techniques for software development phase. Then, proposed a new SRCEE model by combining change impact analysis technique with effort estimation model. In addition, a prototype tool was developed to automate the implementation of SRCEE model. Next, the applicability and accuracy of the new SRCEE model was evaluated by selecting four small size software projects as case selections in applying experimental approch. The estimation results produced from the new proposed SRCEE model were compared with the existing effort estimation models. The finding showed that the new proposed SRCEE model has given more accurate results in terms of applicability and accuracy in estimating the amount of effort for software requirement changes implementation as compared to the existing models. Hence, it can be concluded that the proposed SRCEE model has improved the accuracy rate of effort estimation for software requirement changes during software development phase.

ABSTRAK

Perubahan keperluan perisian merupakan satu fenomena biasa dalam mana-mana projek pembangunan perisian. Perubahan keperluan perisian berlaku pada semua peringkat kitaran hayat pembangunan perisian. Membenarkan terlalu banyak perubahan akan menyebabkan kelewatan dalam penghantaran projek. Oleh itu, kajian ini bertujuan untuk meningkatkan ketepatan keperluan perisian ubahsuai (SRCEE) untuk perubahan keperluan perisian dalam fasa pembangunan perisian (SDP). Ia mencadangkan model SRCEE berasaskan algoritma baru yang boleh digunakan untuk meningkatkan ketepatan anggaran usaha perubahan dalam SDP. Kajian ini menganalisis model SRCEE sedia ada dan mengubah teknik analisis impak untuk fasa pembangunan perisian. Kemudian, mencadangkan model SRCEE baru dengan menggabungkan teknik analisa kesan perubahan dengan model anggaran usaha. Di samping itu, alat prototaip telah dibangunkan untuk mengautomasikan pelaksanaan model SRCEE. Seterusnya, kebolehgunaan dan ketepatan model SRCEE yang baru telah dinilai dengan memilih empat projek perisian saiz kecil sebagai pilihan kes dalam mengaplikasikan ujikaji eksperimen. Keputusan anggaran yang dihasilkan dari model SRCEE baru yang dicadangkan dibandingkan dengan model anggaran usaha yang sedia ada. Dapatan menunjukkan bahawa model SRCEE baru yang dicadangkan telah memberikan hasil yang lebih tepat dari segi kebolehgunaan dan ketepatan dalam menganggarkan jumlah usaha untuk melaksanakan perubahan keperluan perisian berbanding dengan model sedia ada. Oleh itu, dapat disimpulkan bahawa model SRCEE yang dicadangkan telah meningkatkan kadar ketepatan penganggaran usaha untuk perubahan keperluan perisian semasa fasa pembangunan perisian.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CIA	-	Change Impact Analysis
CIP	-	Class Interaction Prediction
COCOMO II	-	Construct Cost Model II
EI		External Input
EIF	-	External Interface Files
EO	-	External Output
EQ	-	External Inquiry
FPA	-	Function Point Analysis
ILF	-	Internal Logical Files
KSLOC	-	Kilo Source Lines of Code
LOC	-	Lines of Code
OBA	-	On-Board Automobile
PS	-	Payroll System
SRCEEM	-	Software Requirement Change Effort Estimation Model
SRCEEMPT	-	Software Requirement Change Effort Estimation Model Prototype Tool
SCM	-	Software Change Management
SDD	-	Software Design Document
SDLC	-	Software Development Life Cycle
SDP	-	Software Development Phase
SDP-CIAF	-	Software Development Phase Change Impact Analysis Factor
SEE	-	Software Effort Estimation
SLOC	-	Source Lines of Code
SMP	-	Software Maintenance Phase
SRC	-	Software Requirement Change
SRS	-	Software Requirements Specifications
UCP	-	Use case Point
UFP	-	Unadjusted Function Points

VMCS - Vending Machine Control System

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Overview

Software Development Life Cycle (SDLC) methodology consists of several phases named as requirements gathering phase, analysis phase, design phase, coding phase, testing phase, deployment phase and maintenance phase. The methodology is used by software development team for the development of high quality software that meets the needs of its end users (Ruparelia, 2010). There are commonly two types of SDLC methodologies as highlighted in the literature, namely: (i) Traditional methodology and (ii) Agile methodology. Traditional methodology focuses on comprehensive planning, extensive documentation and detailed design (Awad, 2005). Agile methodology, on the other hand, supports customer collaboration over detailed planning, stresses on the working software over the comprehensive documentation and beliefs individual interactions over detailed processes of design (Beck *et al.*, 2001). Nevertheless, of adopting any SDLC methodology, the more important for software development team is to consider the planning and controlling of Software Effort Estimation (SEE). According to (Lehtinen *et al.*, 2014a), planning and estimation of software effort are very important for the delivering of a successful software.

Software Requirement Change Effort Estimation (SRCEE) is the process of predicting that how much work and how many hours of work are required to develop a software. Normally it describes in person-month unit (Basri *et al.*, 2015; Tanveer *et al.*, 2016). SEE has been started since 1960s (Farr and Zagorski, 1964; Nelson, 1967) and still there are lot of opinions and discussions for achieving an accurate effort estimation results (Bardsiri *et al.*, 2013; Idri *et al.*, 2016; Lehtinen *et al.*, 2014b; Shah and Kama, 2018a). Although, the researchers have proposed various types of SRCEE models and most of the proposed models are estimating the development work for

Software Planning Phase (SPP) and Software Maintenance Phase (SMP) where software artifacts are in consistent states i.e. that all the classes are Fully Developed (means that software is completely developed and can be run) (Basri *et al.*, 2016b). On the other hand, few models are found in the literature for measuring effort for a Software Requirement Change (SRC) during Software Development Phase (SDP) where software artifacts are in inconsistent states such as: some of the classes are Fully Developed, some classes are Partially Developed and some are Not Developed yet and it becomes a challenging task for software project managers to measure accurately the amount of effort for a SRC in SDP (Asl and Kama, 2013b; Basri *et al.*, 2016b; Kchaou *et al.*, 2015). However, (Kama and Halmi, 2013) stated that the combination of Change Impact Analysis (CIA) technique with SEE model can help software project managers in estimating the effort for an SRC in SDP.

Change Impact Analysis (CIA) technique predicts the impact of SRC on software artifacts and it also helps software project managers in knowing the actual status of software artifacts i.e. weather a class is Fully Developed, Partially Developed or Not Developed yet. Whereas, without knowing the actual status of software artifacts it results in over or under estimation and for an accurate estimation it is important to know the actual status of software artifacts. Therefore, the combination of CIA with SRCEE can be useful for software project managers in estimating the effort for SRC during SDP.

This research presents a new Software Requirement Change Effort Estimation Model (SRCEEM) that can be used in measuring the amount of effort for Software Requirement Change(s) during Software Development Phase. The new model identifies and considers the related factors that contribute to the effort estimation for Software Requirement Change(s) in Software Development Phase.

The following sections present research background, problem statement, research questions, research objectives, research scope, research significance and thesis organization.

1.2 Problem Background

Software Project Management (SPM) has existed for years, but it still remains a great challenge for software project team to produce successful software that fulfil its end user requirements within the predicted time and cost (Kerzner, 2018). Several studies have highlighted the importance of the role of software project manager in project's success or failure (Gupta and Kalia, 2017; Liu and Wang, 2014; Medina and Francis, 2015). Moreover, Software project manager plays a vital/main role in a software development team and after all he is responsible/accountable for the success or failure of the project (Gupta and Kalia, 2017). According to, Lehtinen *et al.* (2014b) a software project failure means an identifiable failure in the cost, schedule, scope, or quality of the project. In addition, Kaur and Sengupta (2013a) stated that the most common reasons for project failure are rooted in the project management process itself and also, they have identified some estimation mistakes in their study which are: unclear project goals, objectives, and project requirement changes during the project. Therefore, a software project manager is always responsible in managing the SRCs and justifies the decisions that he has taken while accepting or rejecting a SRC (Bano *et al.*, 2012).

Software Requirement Changes (SRCs) may occur at any phase of SDLC (Basri *et al.*, 2016b). Accommodating a huge amount of SRCs might increase the development time and cost of the software and denying a large number of SRCs possibly increase customer dissatisfaction. However, a good change acceptance decision can help software project managers in managing SRCs (Shah and Kama, 2018a). According to change acceptance decision is a process that help software project managers in accepting or rejecting a SRC or SRCs. In Addition to this they have also stated that there are two most related inputs that help software project managers in an effective change acceptance decision for SRCs during SDP are: (i) Change Impact Analysis (CIA) and (ii) Software Requirement Change Effort Estimation (SRCEE) (Kchaou *et al.*, 2015). CIA is the process of predicting the impact of an SRC on software artifacts and it also identifies the factors that need to be modified to accomplish a SRC. Alternatively, SRCEE is a process that predicts

the amount of work that is required to implement a SRC or SRCs (Kama and Halmi, 2013; Kchaou *et al.*, 2017).

There are two types of SRCEE models which are widely used: (i) algorithmic-based models and (ii) non-algorithmic-based models. Some of the most common algorithmic-based models are: COCOMO II (Hira *et al.*, 2016), Function Point Analysis (Hira and Boehm, 2016) and Use-Case Points (Alves *et al.*, 2013). Whereas, some of the non-algorithmic-based models are: Expert Judgement (Rastogi *et al.*, 2014), Analogy Based Estimation (Usharani *et al.*, 2016) and Delphi (Britto *et al.*, 2014). Although several extensions of these models are developed to estimate effort in SMP. On the other hand, lack of studies have been found to estimate effort in SDP and still it remains an interesting task for software project managers to estimate the amount of effort for SRCs in SDP (Chinthanet *et al.*, 2016; Idri *et al.*, 2016; Junior *et al.*, 2015; Kchaou *et al.*, 2015).

Although, (Kama and Halmi, 2013) stated that the combination of CIA and SEE may improves the estimation accuracy for SDP. At present, few studies (Basri *et al.*, 2016a; Kama and Halmi, 2013; Kchaou *et al.*, 2015) have been identified which are using the combination of CIA and SRCEE as an effort estimation model and provided better estimation results for SDP (Kama and Halmi, 2013; Shahid and Ibrahim, 2016). However, these technique are limited as they are using Source Lines of Code (SLOC) for SRC size estimation where it has high dependency on the source codes existence whereby in the early stage of development the source codes have yet to be developed. Therefore, the proposed model is using Function Point Analysis (FPA) instead of SLOC for SRC size estimation, which is able to estimate the effort in the early stage of software development.

1.3 Problem Statement

From algorithmic-based Software Requirement Change Effort Estimation (SRCEE) perspective this study has reviewed several research works and identified the most common models which are used for SRCEE are: COCOMO II and FPA.

These models are providing good SRCEE results for miantance phase and planning phase where the software artifacts are in consistent states means that all classes are Fully Developed or Not Developed yet. However, these two models i.e. COCOMO II and FPA are limited to provide accurate SRCEE results for SDP where software artifacts are in inconsistent such as: some classes are Fully Developed, some classes are Half Developed, some classes are Major Developed, some classes are Minor Developed and even some classes are Not Developed yet. On the other hand, some of the current studies (Basri *et al.*, 2015; Sufyan *et al.*, 2016a; Sufyan *et al.*, 2016b), stated that it is a significant need to combine SRCEE with the CIA technique to improve estimation accuracy for SRC during SDP. The main reason of this combination is because the capability of CIA to consider the inconsistent states of software artifacts. However, one main challenge on this combination is how SRCEE and CIA technique could examine the inconsistent states of software artifacts. Research done by (Basri *et al.*, 2015) has significantly presented the impact of combination between SRCEE and CIA technique. One of the arguments raised by (Basri *et al.*, 2015) in his study is that the accuracy of the estimation has high dependency on the SRCEE. For instance, (Basri *et al.*, 2015) has used the COCOMO II and Single Lines of Code (SLOC) to estimate the amount of effort that is required to implement a SRC. This technique is limited where it has high dependency on the source codes existence whereby in the early stage of development the source codes have yet to be developed. Moreover, it has considered three states of software artifacts i.e. Fully Developed, Partially Developed and Not Developed. Fully Developed means (100%) of code is developed, Partially Developed means (50%) of code is developed and Not Developed means (0%) of code is developed. Whereas, it is limited to provide accurate estimation results for an SRC when the code is developed that is in between Fully Developed and Partially Developed or Not Developed and Partially Developed. Therefore, it is important to have a SRCEE technique that is able to provide more accurate estimation results for an SRC during SDP. Hence, this research proposed a new SRCEE model for SRC by combining COCOMO II with FPA to support the estimation in early phase of development when the source code has yet to be developed. Furthermore, it considers five states of code development status instead of three states such as: Fully Developed (100% of code is developed), Major Developed (75% of Code is Developed) Half Developed (50% of code is developed), Minor Developed (25% of code is developed) or Not

Developed (0% of code is developed) to provide more accurate software requirement change effort estimation results as compare to previous models.

1.4 Research Questions

This research deals with the main question of "How to improve the accuracy of Software Requirement Change Effort Estimation Model for Software Requirement Change in Software Development Phase?" To provide an effective solution for the main research question, several sub-questions are constructed:

(a) What are the existing algorithmic-based Software Requirement Change Effort Estimation Models and Change Impact Analysis techniques used for Software Development Phase?

(b) How Change Impact Analysis technique for Software Development Phase could be combined with algorithmic-based Software Requirement Change Effort Estimation Model?

(c) How to calculate the estimated effort for Software Requirement Change in Software Development Phase with the new algorithmic-based Software Requirement Change Effort Estimation Model?

(d) How applicable is the new algorithmic-based Software Requirement Change Effort Estimation Model for Software Requirement Changes in Software Development Phase?

(e) How the new algorithmic-based Software Requirement Change Effort Estimation Model improves the effort estimation's accuracy for Software Requirement Changes in Software Development Phase as compared to the existing Software Effort Estimation Models?

1.5 Research Objectives

The aim of this research is to propose a new algorithmic-based SRCEE model which could be used to improve the accuracy of the change effort estimation in SDP. Hence to achieve the aim, several objectives are identified as follow:

(a) To analyse the existing Software Requirement Change Effort Estimation Models and Change Impact Analysis techniques used for Software Development Phase.

(b) To design an algorithmic-based Software Requirement Change Effort Estimation Model using a Change Impact Analysis technique for Software Development Phase.

(c) To develop a Software Requirement Change Effort Estimation Prototype Tool that implements the new algorithmic-based Software Requirement Change Effort Estimation Model.

(d) To evaluate the applicability of the new algorithmic-based Software Requirement Change Effort Estimation Model for Software Requirement Change in Software Development Phase.

(e) To evaluate the accuracy of the new algorithmic-based Software Requirement Change Effort Estimation Model as compared to the existing Software Requirement Change Effort Estimation Models for Software Requirement Changes in Software Development Phase.

1.6 Scope of Research

The main purpose of describing a research scope is to focus the research area and highlight the borders and constraints of the research. The limitation of the research scope are as following:

(a) Small size of software projects which implemented either Traditional or Agile methodology

(b) Software projects which are in the development states (requirement analysis, design, coding, testing or deployment phase)

(c) Software projects that implemented in any programing language

(d) The development phase duration from three (3) to six (6) months

1.6.1 Research Context

The objective of this research is to develop a Software Requirement Change Effort Estimation Model for Software Development Phase using a Change Impact Analysis technique. While, most of the techniques are developed to support Software Requirement Changes during Software Maintenance Phase. However, this research focuses on CIA techniques which are developed for SDP. In overall, SDP differs from SMP due to the presence of inconsistent states of software artifacts.

1.7 Significance of Research

Main impact of this research is important in different of perceptions. In one perception, the new Software Requirement Change Effort Estimation Model (SRCEEM) will offer key information in predicting how much work and how many hours of work are needed for a SRC. The effort to implement requirement changes need to be assessed precisely to support the change acceptance decision during SDP. Additionally, it will care for well planning and arranging of the requirements implementation during software project management.

In other perception, most change acceptance decision assessment during SDP is based on CIA techniques. The CIA examines the potential impacts by assessing current state of software artifacts such as requirement specifications and source code

during SDP. By realizing the significance of the CIA, the effectiveness of the development work prediction will be expected to be improved by including the current CIA into the new SEE model for SDP.

1.8 Operational Definition

Traditional Methodology	:	Describe one of the process to develop a software that practices detailed planning, comprehensive documentation and extensive design.
Agile Methodology	:	More recent technique in developing a software that practices customer collaboration over detailed planning, emphasizes on the working software over the comprehensive documentation and values individual interactions over extensive processes and design.
Software Development Phase	:	Identify the stages of the software process in developing a software. The stages start from requirement, analysis, design, implementation, testing until deployment.
Algorithmic Model	:	A formal technique that apply algorithms and formulas to derive a result of the estimation calculation.
Non-Algorithmic Model	:	An informal technique that are not using any algorithms or formal methods and / or formulas in deriving the estimation result.
Software Requirement Change	:	The modification or adjustment that occurs during software development phase, which may involve the requirement being developed.
Change Impact Analysis	:	A process of identifying potential consequences of change or estimating what needs to be modified to

		accomplish a change.
Software Requirement Change Effort Estimation	:	A process of predicting the amount of work or task required in implementing the modification that occurred.
Magnitude of Relative Error	:	An absolute value that was derived from the difference between the estimated values as compared to the actual value.
Applicability	:	The degree of how much the new model is significant for software requirement change in software development phase.
Accuracy	:	The degree of precision of the estimated effort as compared to the actual effort.
Software artifacts	:	An artefact is one of many kinds of tangible by-product produced during the development of software. Some artifacts (e.g., use cases, class diagrams, and other UML models, requirements and design documents) help describe the function, architecture, and design of software.
Fully Developed	:	Means 100% of code is developed for the particular software requirement change.
Partially Developed	:	Means 50% of code is developed for the particular software requirement change.
Not Developed	:	Means 00% of code is developed for the particular software requirement change.
Major Developed	:	Means 75% of code is developed for the particular software requirement change.
Half Developed	:	Means 50% of code is developed for the particular software requirement change.
Minor Developed	:	Means 25% of code is developed for the particular software requirement change.

1.9 Organization of the Thesis

This thesis comprises of seven chapters. Chapter one, includes the research introduction, background, problem statement, research questions, research objectives, scope of the research, significance of this research and the organization of the thesis.

(a) Chapter Two discusses the comprehensive review of the literature.

(b) Chapter Three describes the research methodology used in conducting the research.

(c) Chapter Four presents the new proposed Software Requirement Change Effort Estimation Model.

(d) Chapter Five presents the new proposed Software Requirement Change Effort Estimation Model Prototype Tool.

(e) Chapter Six presents the Results and Discussions of the Experiments performed for the Evaluation of newly developed Software Requirement Change Effort Estimation Model.

(f) Chapter Seven presents the Conclusion and Recommendations of this study.

REFRENCES

Agrawal, P., and Kumar, S. (2016, 18-19 March 2016). *Early phase software effort estimation model.* Paper presented at the 2016 Symposium on Colossal Data Analysis and Networking (CDAN), 1-8.

Albrecht, A. J. (1984). AD/M productivity measurement and estimate validation. *IBM Corporate Information Systems, IBM Corp., Purchase, NY.*

Albuquerque, R., Fernandes, R., Fontana, R. M., Reinehr, S., and Malucelli, A. (2016). *Motivating Factors in Agile and Traditional Software Development Methods: A Comparative Study.* Paper presented at the Brazilian Workshop on Agile Methods, 136-141.

Ali, N., and Lai, R. (2016). A method of requirements change management for global software development. *Information and Software Technology, 70*, 49-67.

Alliance, A. (2016). What is Agile. *Agile Alliance. https://www. agilealliance. org/agile101/what-is-agile.*

Alves, L. M., Oliveira, S., Ribeiro, P., and Machado, R. J. (2014, June 30 2014-July 3 2014). *An Empirical Study on the Estimation of Size and Complexity of Software Applications with Function Points Analysis.* Paper presented at the 2014 14th International Conference on Computational Science and Its Applications, 27-34.

Alves, L. M., Sousa, A., Ribeiro, P., and Machado, R. J. (2013, 23-26 Oct. 2013). *An empirical study on the estimation of software development effort with use case points.* Paper presented at the 2013 IEEE Frontiers in Education Conference (FIE), 101-107.

Ammann, P., and Offutt, J. (2016). *Introduction to software testing*: Cambridge University Press.

Anandhi, V., and Chezian, R. M. (2014). *Regression techniques in software effort estimation using cocomo dataset.* Paper presented at the Intelligent Computing Applications (ICICA), 2014 International Conference on, 353-357.

Arnaut, B. M., Ferrari, D. B., and Souza, M. L. d. O. e. (2016, 3-5 Oct. 2016). *A requirements engineering and management process in concept phase of*

complex systems. Paper presented at the 2016 IEEE International Symposium on Systems Engineering (ISSE), 1-6.

Asl, and Kama. (2013a, 4-7 June 2013). *A Change Impact Size Estimation Approach during the Software Development*. Paper presented at the 2013 22nd Australian Software Engineering Conference, 68-77.

Asl, M. H., and Kama, N. (2013b, 4-7 June 2013). *A Change Impact Size Estimation Approach during the Software Development*. Paper presented at the 2013 22nd Australian Software Engineering Conference, 68-77.

Awad, M. (2005). A comparison between agile and traditional software development methodologies. *University of Western Australia*.

Bano, M., Imtiaz, S., Ikram, N., Niazi, M., and Usman, M. (2012, 14-15 May 2012). *Causes of requirement change - A systematic literature review*. Paper presented at the Evaluation & Assessment in Software Engineering (EASE 2012), 16th International Conference on, 22-31.

Bardsiri, V. K., Jawawi, D. N. A., Bardsiri, A. K., and Khatibi, E. (2013). LMES: A localized multi-estimator model to estimate software development effort. *Engineering Applications of Artificial Intelligence*(0).

Basri, S., Kama, N., Haneem, F., and Ismail, S. A. (2016a). *Predicting effort for requirement changes during software development*. Paper presented at the Proceedings of the Seventh Symposium on Information and Communication Technology.

Basri, S., Kama, N., and Ibrahim, R. (2015). A Novel Effort Estimation Approach for Requirement Changes during Software Development Phase. *International Journal of Software Engineering and Its Applications, 9*(1), 237-252.

Basri, S., Kama, N., and Ibrahim, R. (2016b). COCHCOMO: An extension of COCOMO II for Estimating Effort for Requirement Changes during Software Development Phase.

Beck, K., Beedle, M., Bennekum, A. v., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). Manifesto for Agile Software Development. from http://www.agilemanifesto.org/

Boehm, B., Clark, B., Horowitz, E., Westland, C., Madachy, R., and Selby, R. (1995). Cost models for future software life cycle processes: COCOMO 2.0. *Annals of Software Engineering, 1*(1), 57-94.

Boehm, B. W. (2000). *Software Cost Estimation with Cocomo II*: Prentice Hall.

Boehm, B. W., Madachy, R., and Steece, B. (2000). *Software cost estimation with Cocomo II with Cdrom*: Prentice Hall PTR.

Britto, R., Freitas, V., Mendes, E., and Usman, M. (2014, 18-21 Aug. 2014). *Effort Estimation in Global Software Development: A Systematic Literature Review.* Paper presented at the 2014 IEEE 9th International Conference on Global Software Engineering, 135-144.

Cabinet, D. o. P. a. (2008). Project Management

Fact Sheet. 2018

Cameron, E., and Green, M. (2015). *Making sense of change management: a complete guide to the models, tools and techniques of organizational change*: Kogan Page Publishers.

Chai, T., and Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)? *Geoscientific Model Development Discussions, 7*, 1525-1534.

Chen, C.-Y., and Chen, P.-C. (2009). A holistic approach to managing software change impact. *Journal of Systems and Software, 82*(12), 2051-2067.

Chinthanet, B., Phannachitta, P., Kamei, Y., Leelaprute, P., Rungsawang, A., Ubayashi, N., et al. (2016). *A review and comparison of methods for determining the best analogies in analogy-based software effort estimation.* Paper presented at the Proceedings of the 31st Annual ACM Symposium on Applied Computing. from http://delivery.acm.org.ezproxy.psz.utm.my/10.1145/2860000/2851974/p1554-chinthanet.pdf?ip=161.139.39.211&id=2851974&acc=ACTIVE%20SERVICE&key=69AF3716A20387ED%2EC758BA176ED44BB8%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=691584591&CFTOKEN=82404349&__acm__=1478577844_44d552733b49beab42e034c9a61ceb8b

Clark, B., Devnani-Chulani, S., and Boehm, B. (1998). *Calibrating the COCOMO II post-architecture model.* Paper presented at the Proceedings of the 20th international conference on Software engineering, 477-480.

Coad, P., Yourdon, E., and Coad, P. (1991). *Object-oriented analysis* (Vol. 2): Yourdon press Englewood Cliffs, NJ.

Creswell, J. W. (2011). *Educational Research: Planning, Conducting, and Evaluating Quantitative and Qualitative Research*: Pearson.

Creswell, J. W. (2013). *Research design: Qualitative, quantitative, and mixed methods approaches*: Sage publications.

Cuadrado-Gallego, J. J., Rodriguez-Soria, P., Gonzalez, A., Castelo, D., and Hakimuddin, S. (2010, 18-20 Aug. 2010). *Early Functional Size Estimation with IFPUG Unit Modified.* Paper presented at the Computer and Information Science (ICIS), 2010 IEEE/ACIS 9th International Conference on, 729-733.

De Lemos, R., Giese, H., Müller, H. A., Shaw, M., Andersson, J., Litoiu, M., et al. (2013). Software engineering for self-adaptive systems: A second research roadmap. In *Software Engineering for Self-Adaptive Systems II* (pp. 1-32): Springer.

Desharnais, J.-M., and Abran, A. (2003). *Approximation techniques for measuring function points.* Paper presented at the Proceedings of the 13th international workshop on software measurement (IWSM 2003).

Di Martino, S., Ferrucci, F., Gravino, C., and Sarro, F. (2016). Web Effort Estimation: Function Point Analysis vs. COSMIC. *Information and Software Technology, 72*, 90-109.

Edith Cowan University. (2018). Project Sizing Guidelines. Retrieved 8/14/2018, 2018,, from https://intranet.ecu.edu.au/__data/assets/pdf_file/0017/670400/ITSC_Project_Sizing_Guidelines.pdf

Farr, L., and Zagorski, H. J. (1964). *Factors that Affect the Cost of Computer Programming. Volume Ii. a Quantitative Analysis*: DTIC Documento. Document Number)

Fedotova, O., Teixeira, L., and Alvelos, H. (2013). Software Effort Estimation with Multiple Linear Regression: Review and Practical Application. *J. Inf. Sci. Eng., 29*(5), 925-945.

Ferrucci, F., Gravino, C., and Lavazza, L. (2016). *Simple function points for effort estimation: a further assessment.* Paper presented at the Proceedings of the 31st Annual ACM Symposium on Applied Computing. from http://delivery.acm.org.ezproxy.psz.utm.my/10.1145/2860000/2851779/p1428-ferrucci.pdf?ip=161.139.39.211&id=2851779&acc=ACTIVE%20SERVICE&key=69AF3716A20387ED%2EC758BA176ED44BB8%2E4D4702B0C3E

38B35%2E4D4702B0C3E38B35&CFID=691584591&CFTOKEN=82404349&__acm__=1478602102_95339ebc049f9488ca58fe79623a6bc6

Ghafoor, F., Shah, I. A., and Rashid, N. (2017). Issues in Adopting Agile Methodologies in Global and Local Software Development: A Systematic Literature Review Protocol with Preliminary Results. *International Journal of Computer Applications, 160*(7).

Gupta, M., and Kalia, A. (2017). Empirical Study of Software Metrics. *Research Journal of Science and Technology, 9*(1), 17-24.

Hall, N. (2007). R. A. Fisher and his advocacy of randomization. *Journal of the History of Biology, 40*(2), 295-325.

Hayes, J. (2014). *The theory and practice of change management*: Palgrave Macmillan.

Hira, A., and Boehm, B. (2016). *Function Point Analysis for Software Maintenance.* Paper presented at the Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. Retrieved 11/8/2016, from http://delivery.acm.org.ezproxy.psz.utm.my/10.1145/2970000/2962613/a48-hira.pdf?ip=161.139.39.211&id=2962613&acc=ACTIVE%20SERVICE&key=69AF3716A20387ED%2EC758BA176ED44BB8%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=691584591&CFTOKEN=82404349&__acm__=1478601096_ff217bd29131a65de2a3caed199dfa87

Hira, A., Sharma, S., and Boehm, B. (2016). *Calibrating COCOMO® II for projects with high personnel turnover.* Paper presented at the Proceedings of the International Conference on Software and Systems Process. Retrieved 11/8/2016, from http://delivery.acm.org.ezproxy.psz.utm.my/10.1145/2910000/2904367/p51-hira.pdf?ip=161.139.39.211&id=2904367&acc=ACTIVE%20SERVICE&key=69AF3716A20387ED%2EC758BA176ED44BB8%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=691584591&CFTOKEN=82404349&__acm__=1478602545_b114e3c297da83a197b481ebaf8938e2

Humayun, M., and Gang, C. (2012). Estimating effort in global software development projects using machine learning techniques. *International Journal of Information and Education Technology, 2*(3), 208.

Ibrahim, S., Idris, N. B., Munro, M., and Deraman, A. (2006). *A Software Traceability Validation For Change Impact Analysis of Object Oriented Software.* Paper presented at the Software Engineering Research and Practice, 453-459.

Idri, A., Amazal, F. a., and Abran, A. (2015). Analogy-based software development effort estimation: A systematic mapping and review. *Information and Software Technology, 58*, 206-230.

Idri, A., Hosni, M., and Abran, A. (2016). Systematic literature review of ensemble effort estimation. *Journal of Systems and Software, 118*, 151-175.

IEEE Standard for Software Configuration Management Plans. (2005). *IEEE Std 828-2005 (Revision of IEEE Std 828-1998)*, 1-30.

Jorgensen, M., and Molokken-Ostvold, K. (2004). Reasons for software effort estimation error: impact of respondent role, information collection approach, and data analysis method. *IEEE Transactions on Software engineering, 30*(12), 993-1007.

Junior, M. d. F., Fantinato, M., and Sun, V. (2015). Improvements to the Function Point Analysis Method: A Systematic Literature Review. *IEEE Transactions on Engineering Management, 62*(4), 495-506.

Kama, and Halmi, M. (2013). *Extending Change Impact Analysis Approach for Change Effort Estimation in the Software Development Phase.* Paper presented at the WSEAS International Conference. Proceedings. Recent Advances in Computer Engineering Series.

Kama, N., and Azli, F. (2012a, 4-7 Dec. 2012). *A Change Impact Analysis Approach for the Software Development Phase.* Paper presented at the 2012 19th Asia-Pacific Software Engineering Conference, 583-592.

Kama, N., and Azli, F. (2012b). *A Change Impact Analysis Approach for the Software Development Phase.* Paper presented at the Proceedings of the 2012 19th Asia-Pacific Software Engineering Conference - Volume 01.

Kama, N., Basri, S., and Ibrahim, R. (2014). Considering Partially Developed Artifacts in Change Impact Analysis Implementation. *JSW, 9*(8), 2174-2179.

Kama, N., Ismail, S. A., Kamardin, K., Zainuddin, N. M., Azmi, A., and Zainuddin, W. S. (2015). *A Change Impact Analysis Tool: Integration Between Static and Dynamic Analysis Techniques.* Paper presented at the International

Conference on Intelligent Software Methodologies, Tools, and Techniques, 413-424.

Kaur, M., and Sehra, S. K. (2014, 7-8 Feb. 2014). *Particle swarm optimization based effort estimation using Function Point analysis.* Paper presented at the Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014 International Conference on, 140-145.

Kaur, R., and Sengupta, J. (2013a). Software Process Models and Analysis on Failure of Software Development Projects. *CoRR, abs/1306.1068.*

Kaur, R., and Sengupta, J. (2013b). Software process models and analysis on failure of software development projects. *arXiv preprint arXiv:1306.1068.*

Kchaou, D., Bouassida, N., and Ben-Abdallah, H. (2015, 20-22 July 2015). *Change effort estimation based on UML diagrams application in UCP and COCOMOII.* Paper presented at the 2015 10th International Joint Conference on Software Technologies (ICSOFT), 1-8.

Kchaou, D., Bouassida, N., and Ben-Abdallah, H. (2017). UML models change impact analysis using a text similarity technique. *IET Software, 11*(1), 27-37.

Kerzner, H. (2018). *Project management best practices: Achieving global excellence*: John Wiley & Sons.

Kitchenham, B., Pfleeger, S. L., McColl, B., and Eagan, S. (2002a). An empirical study of maintenance and development estimation accuracy. *Journal of systems and software, 64*(1), 57-77.

Kitchenham, B. A., Pfleeger, S. L., Pickard, L. M., Jones, P. W., Hoaglin, D. C., El Emam, K., et al. (2002b). Preliminary guidelines for empirical research in software engineering. *IEEE Transactions on software engineering, 28*(8), 721-734.

Lehtinen, T. O., Mäntylä, M. V., Vanhanen, J., Itkonen, J., and Lassenius, C. (2014a). Perceived causes of software project failures–An analysis of their relationships. *Information and Software Technology, 56*(6), 623-643.

Lehtinen, T. O. A., Mäntylä, M. V., Vanhanen, J., Itkonen, J., and Lassenius, C. (2014b). Perceived causes of software project failures – An analysis of their relationships. *Information and Software Technology, 56*(6), 623-643.

Liu, S., and Wang, L. (2014). Understanding the impact of risks on performance in internal and outsourced information technology projects: The role of strategic importance. *International Journal of Project Management, 32*(8), 1494-1510.

Longstreet, D. (2002). Fundamentals of function point analysis. *Longstreet Consulting, Inc.*

MacFarland, T. W., and Yates, J. M. (2016). Mann–whitney u test. In *Introduction to nonparametric statistics for the biological sciences using R* (pp. 103-132): Springer.

Management, Q. S. (2018). Function Point Languages Table. 5th version. Retrieved 10/04/2018, 2018, from http://www.qsm.com/resources/function-point-languages-table

Mauro Gasparini, M. P. R. (2008). *7 Design of Experiments. Handbook of Probability: Theory and Applications. SAGE Publications, Inc.* Thousand Oaks, CA: SAGE Publications, Inc.

Medina, A., and Francis, A. J. (2015). What are the characteristics that software development project team members associate with a good project manager? *Project Management Journal, 46*(5), 81-93.

Myers, G. J., Sandler, C., and Badgett, T. (2011). *The art of software testing*: John Wiley & Sons.

Nelson, E. A. (1967). *Management handbook for the estimation of computer programming costs*: DTIC Documento. Document Number)

Nguyen, V., Steece, B., and Boehm, B. (2008). *A constrained regression technique for COCOMO calibration.* Paper presented at the Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, 213-222.

Nurmuliani, N., Zowghi, D., and Williams, S. (2006). *Requirements volatility & its impact on change effort: Evidence based research n software development projects.* Paper presented at the Verified OK.

Rasool, R., and Malik, A. A. (2015, 19-20 Dec. 2015). *Effort estimation of ETL projects using Forward Stepwise Regression.* Paper presented at the 2015 International Conference on Emerging Technologies (ICET), 1-6.

Rastogi, H., Dhankhar, S., and Kakkar, M. (2014, 25-26 Sept. 2014). *A survey on software effort estimation techniques.* Paper presented at the Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference -, 826-830.

Ruparelia, N. B. (2010). Software development lifecycle models. *SIGSOFT Softw. Eng. Notes, 35*(3), 8-13.

Ruxton, G. D. (2006). The unequal variance t-test is an underused alternative to Student's t-test and the Mann–Whitney U test. *Behavioral Ecology, 17*(4), 688-690.

Sabrjoo, S., Khalili, M., and Nazari, M. (2015, 5-6 Nov. 2015). *Comparison of the accuracy of effort estimation methods.* Paper presented at the 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), 724-728.

Shah, J., and Kama, N. (2018a). *Extending Function Point Analysis Effort Estimation Method for Software Development Phase.* Paper presented at the Proceedings of the 2018 7th International Conference on Software and Computer Applications, 77-81.

Shah, J., and Kama, N. (2018b). *Extending Function Point Analysis Effort Estimation Method for Software Development Phase.* Paper presented at the Proceedings of the 2018 7th International Conference on Software and Computer Applications.

Shahid, M., and Ibrahim, S. (2016, 12-16 Jan. 2016). *Change impact analysis with a software traceability approach to support software maintenance.* Paper presented at the 2016 13th International Bhurban Conference on Applied Sciences and Technology (IBCAST), 391-396.

Shepperd, M., and Schofield, C. (1997). Estimating software project effort using analogies. *IEEE Transactions on software engineering, 23*(11), 736-743.

Small, A. W., and Downey, E. A. (2001a, 2001). *Managing change: some important aspects.* Paper presented at the Change Management and the New Industrial Revolution, 2001. IEMC '01 Proceedings., 50-57.

Small, A. W., and Downey, E. A. (2001b). *Managing change: some important aspects.* Paper presented at the Change Management and the New Industrial Revolution, 2001. IEMC'01 Proceedings., 50-57.

Sommerville, I. (2010). *Software Engineering* (9 ed.). Harlow, England: Addison-Wesley.

Stensrud, E., Foss, T., Kitchenham, B., and Myrtveit, I. (2002). *An empirical validation of the relationship between the magnitude of relative error and project size.* Paper presented at the Software Metrics, 2002. Proceedings. Eighth IEEE Symposium on, 3-12.

Sufyan, B., Nazri, K., Faizura, H., and Saiful, A. I. (2016a). *Predicting effort for requirement changes during software development*. Paper presented at the Proceedings of the Seventh Symposium on Information and Communication Technology.

Sufyan, B., Nazri, K., Saiful, A., and Faizura, H. (2016b). Using static and dynamic impact analysis for effort estimation. *IET Software, 10*(4), 89-95.

Tanveer, B., Guzm, L., aacute, and Engel, U. M. (2016). *Understanding and improving effort estimation in Agile software development: an industrial case study*. Paper presented at the Proceedings of the International Conference on Software and Systems Process.

Usharani, K., Ananth, V. V., and Velmurugan, D. (2016, 3-5 March 2016). *A survey on software effort estimation*. Paper presented at the 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), 505-509.

Vickers, P., and Street, C. (2001). An Introduction to Function Point Analysis. *School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, UK.*

Westfall, L. (2014). What, Why, Who, When, and How of Software Requirements. In *Handbook of Research on Emerging Advancements and Technologies in Software Engineering* (pp. 1-18): IGI Global.

Wiegers, K., and Beatty, J. (2013). *Software requirements*: Pearson Education.

Willmott, C. J., and Matsuura, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate research, 30*(1), 79-82.

Wohlin, C., and Aurum, A. (2014). Towards a decision-making structure for selecting a research design in empirical software engineering. *Empirical Software Engineering*, 1-29.

Wohlin, C., and Aurum, A. (2015). Towards a decision-making structure for selecting a research design in empirical software engineering. *Empirical Software Engineering, 20*(6), 1427-1455.

Appendix A Software Requirements Specification

UTM AIS Payroll System Prototype

Prepared for:

Advanced Informatics School

(AIS)

Prepared by:

JALAL SHAH

UNIVERSITI TEKNOLOGI MALAYSIA

The Development of a Payroll System Prototype

VERSION 1.1

ADVANCED INFORMATICS SCHOOL

UNIVERSITI TEKNOLOGI MALAYSIA

Scope

Software Requirement Specification (SRS) is an overview of Payroll System Prototype (PSP). It tries to help the developers of (PS) to understand the requirements of the system.

Identification

System Identification	:	Payroll System-1.0
System Name	:	Payroll System
System Abbreviation	:	PS
Version Number	:	1.0

System Overview

This system is designed for the new Payroll System, on the request of UTM AIS. This system is designed in such a way to solve the different issues of existing Payroll System which was using before in UTM AIS and was hopelessly out of date.

There are six types of actors namely Employee, Payroll Administrator, Project Database, Printer, System Clock and Bank System in this system. Employee can login in the system to create employee report, maintain timecard. Payroll Administrator can login in the system to maintain employee info, create timecard and to create administrative report. This system also deals with Bank System, System Clock, Printer and Project Database for the financial tasks.

Functional Requirement

This chapter of the SRS contains all software requirements to a level detailed sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements.

Software Capability Requirement

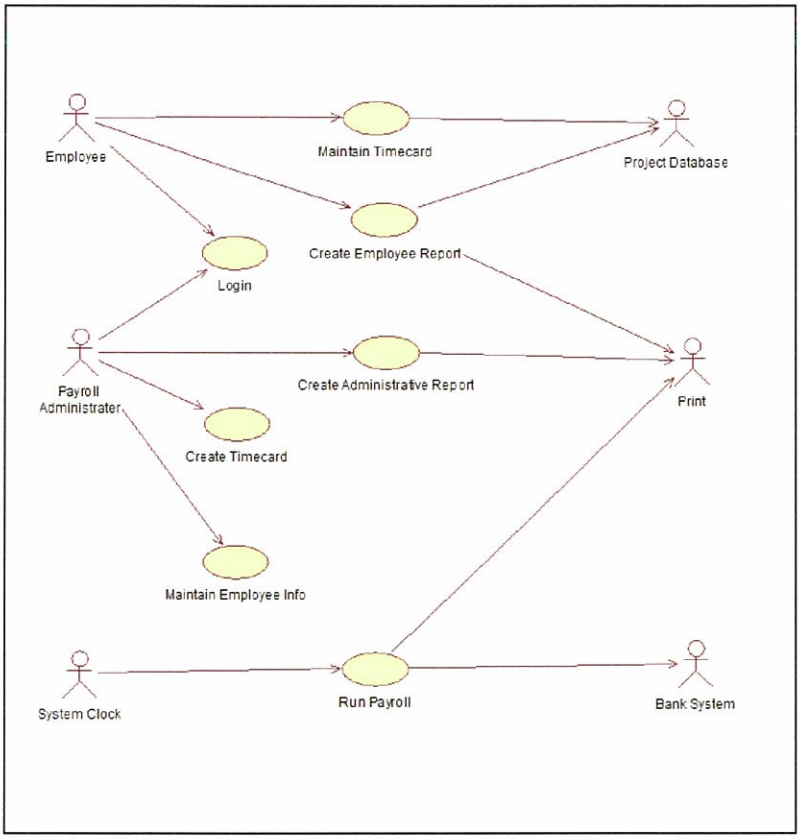

Figure 1: Use Case Diagram of Payroll System

There are six actors and seven use cases for Course Registration System:

List of actors:

a. Employee- Internal User.

b. Payroll Administrator- Internal User.

c. Project Database- External System.

d. Bank System- External System.

e. Printer- External System.

f. System Clock- External System.

List of use cases:

a. Login.

b. Maintain Timecard.

c. Create Employee Report.

d. Maintain Employee Information.

e. Create Timecard.

f. Create Administrative Report.

g. Run payroll.

Brief Description of each Use Cases

Login

This use case describes how a user (Employee or Payroll Administrator) logs into the Payroll System.

Maintain Timecard

This use case allows the Employee to update and submit timecard information. Hourly and salaried employees must submit weekly timecards recording all hours worked that week and which projects the hours are billed to. Project Database is an external actor with this use case.

Create Employee Report

The use case allows the Employee to create a "Total Hours Worked," "Total Hours Worked for a Project", "Vacation/Sick Leave," or "Total Pay Year-to-Date" report. Project Database is an external actor with this use case.

Maintain Employee information

This use case allows the Payroll Administrator to maintain employee information. This includes adding, changing, and deleting employee information from the system.

Create Administrative Report

The use case allows the Payroll Administrator to create either a "Total Hours Worked" or "Pay Year to-Date" report.

Create Timecard

This use case allows the Payroll Administrator to create a time a card for each employee for a certain time period.

Run Payroll

The use case describes how the payroll is run every Friday and the last working day of the month. There are two external actors with this use case which are Printer and Bank System.

Login Use Case [SRS_REQ_01]

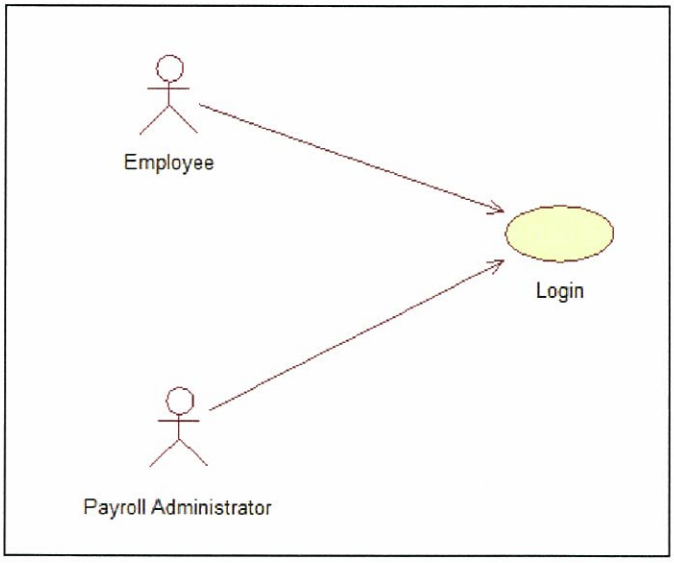

Figure 2 Use Case Diagram of Login

Brief Description

This use case describes how a user logs into the Payroll System.

Flow of Event(s)

Basic Flow

This use case starts when the actor wishes to Login to the Payroll System.

a. The actor enters his/her user name and password **[SRS_REQ_01_1.1].**

b. The system validates the entered user name and password and logs the actor into the system **[SRS_REQ_01_1.2]** **[A-1: Invalid Name/Password].**
c. This use case ends **[SRS_REQ_01_1.3].**

Alternative Flows

[A-1: Invalid Name/Password]

If, in the Basic Flow, the actor enters an invalid user name and/or password, the system displays an error message. The actor can choose to either return to the beginning of the Basic Flow or cancel the login, at which point the use case ends **[SRS_REQ_01_1.4]**.

Special Requirements

None.

Pre-Conditions

The system is in the login state and has the login screen displayed.

Post-Conditions

If the use case was successful, the actor is now logged into the system. If not, the system state is unchanged.

Extension Points

None.

Graphical User interface (GUI)

Figure 3 User Login Interface Diagram

Maintain Employee Information Use Case [SRS_REQ_02]

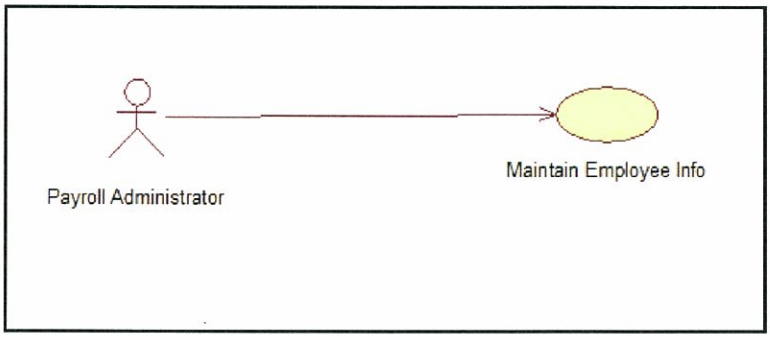

Figure 4 Use Case Diagram of Maintain Employee Information

Brief Description

This use case allows the Payroll Administrator to maintain employee information. This includes adding, editing, and deleting employee information from the system.

Flow of Events

Basic Flow

This use case starts when the Payroll Administrator wishes to add, edit, or delete employee information from the system.

a. The system requests that the Payroll Administrator specify the function he/she would like to Perform (either Add an Employee, Update an Employee, or Delete an Employee) **[SRS_REQ_02_2.1]**.
b. Once the Payroll Administrator provides the requested information, the sub flow "Add an Employee" is executed **[SRS_REQ_02_2.2]** **[A-1: Update an Employee]** **[A-2: Delete an Employee]** **[A-1: Employee Not Found]**.

Add an Employee

c. The system requests that the Payroll Administrator enter the employee information. This includes:
    - Name
    - Employee type (hour, salaried, commissioned)
    - Mailing address
    - Social security number
    - Standard tax deductions
    - Other deductions
    - Phone number
    - Hourly rate (for hourly employees)
    - Salary (for salaried) - hour limit (some Employees may not be able to work overtime) **[SRS_REQ_02_2.3]**.
d. Once the Payroll Administrator provides the requested information, the system generates and assigns a unique employee id number to the employee and the Employee is added to the system **[SRS_REQ_02_2.4]**.
e. The system provides the Payroll Administrator with the new employee id **[SRS_REQ_02_2.5]**.
f. This use case ends **[SRS_REQ_02_2.6]**.

Alternative Flows

[A-1: Update an Employee]

a. The system requests that the Payroll Administrator enter the employee id **[SRS_REQ_02_2.7]**.
b. The Payroll Administrator enters the employee id. The system retrieves and displays the employee information **[SRS_REQ_02_2.8]** **[E-1: Employee not found]**.
c. The Payroll Administrator makes the desired changes to the employee information. This includes any of the information specified in the Add an Employee sub-flow **[SRS_REQ_02_2.9]**.
d. Once the Payroll Administrator updates the necessary information, the system updates the employee record with the updated information **[SRS_REQ_02_2.10]**.

[A-1: Delete an Employee]

a. The system requests that the Payroll Administrator specify the employee id **[SRS_REQ_02_2.11]**.
b. The Payroll Administrator enters the employee id. The systems retrieve and display the employee information **[SRS_REQ_02_2.12]**.
c. The system prompts the Payroll Administrator to confirm the deletion of the employee **[SRS_REQ_02_2.13] [E-2: Delete Cancelled]**.
d. The Payroll Administrator verifies the deletion **[SRS_REQ_02_2.14]** The system marks the employee record for deletion. The next time the payroll is run; the system will generate a final pay check for the deleted employee and remove the Employee from the system **[SRS_REQ_02_2.15]**.

Special Requirements

None.

Pre-Conditions

The Payroll Administrator must be logged onto the system before this use case begins.

Post-Conditions

If the use case was successful, the employee information is added, updated, or deleted from the system. Otherwise, the system state is unchanged.

Extension Points

[E-1: Employee Not Found]

If in the Update an Employee or Delete an Employee sub-flows, an employee with the specified id number does not exist; the system displays an error message. The Payroll Administrator can then enter a different id number or cancel the operation, at which point the use case ends **[SRS_REQ_02_2.16]**.

[E-2: Delete Cancelled]

If in the Delete an Employee sub-flow, the Payroll Administrator decides not to delete the Employee, the delete is cancelled and the Basic Flow is re-started at the beginning **[SRS_REQ_02_2.17]**.

Graphical User Interface (GUI)

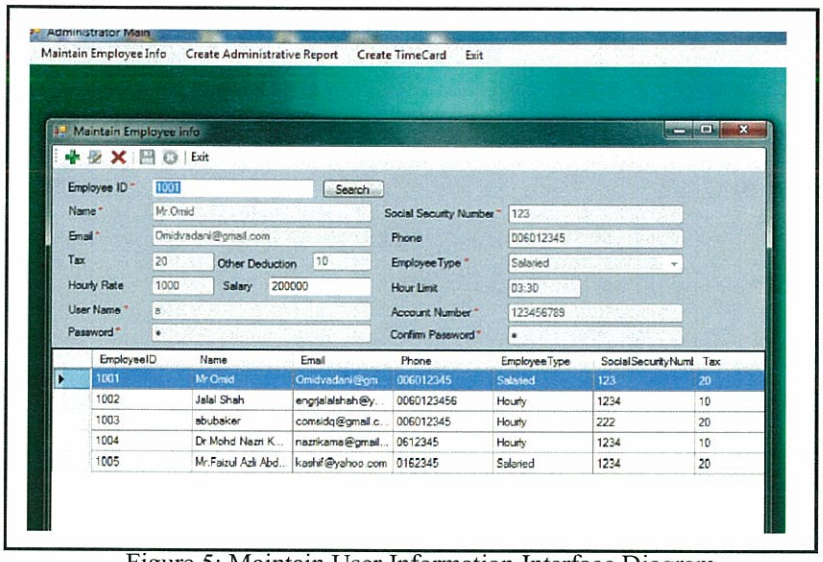

Figure 5: Maintain User Information Interface Diagram

Create Timecard Use Case [SRS_REQ_03]

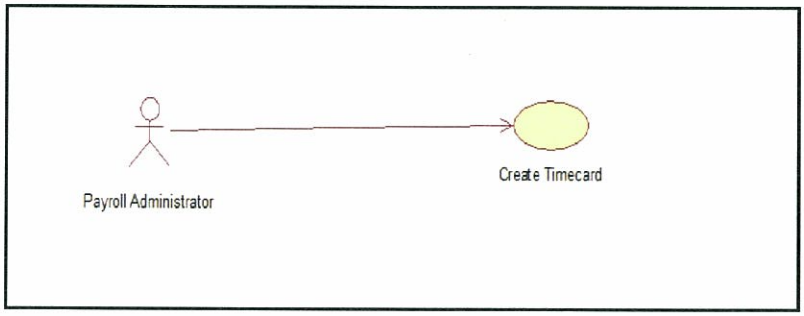

Figure6 Use Case diagram of Create Timecard

Brief Description

This use case allows the Administrator to create, edit or delete timecard for an employee(s) for any project (s) with entering the start and end dates for the related project.

Flow of Events

Basic Flow

This use case starts when the Payroll Administrator wishes to create a timecard for an employee(s).

a. The system requests that the Payroll Administrator specify the function he/she would like to Perform "Add Timecard" or Update Timecard, or Delete Timecard) **[SRS_REQ_03_3.1].**
b. Once the Payroll Administrator provides the requested information, the sub flow "Add Timecard" is executed **[SRS_REQ_03_3.2] [A-1: Update Timecard] [A-2: Delete Timecard].**

Add Timecard

a. The system requests that the Payroll Administrator enter the employee information.

b. This includes:
    - Name
    - Start and End dates
    - Name of project
    - Description if any **[SRS_REQ_03_3.3].**
c. Once the Payroll Administrator provides the requested information, the system
d. Generates and assigns a charge number to the employee and the Timecard is created **[SRS_REQ_03_3.4].**
e. The system provides the Payroll Administrator with the new charge number. **[SRS_REQ_03_3.5].**
f. Once the Payroll Administrator has entered the information, the system saves the timecard **[SRS_REQ_05_3.6].**
g. This use case ends **[SRS_REQ_03_3.7].**

Alternative Flows

[A-1: Update Timecard]

a. The system requests that the Payroll Administrator enter the charge number **[SRS_REQ_03_3.8]**
b. The Payroll Administrator enters the charge number. The system retrieves and displays the employee information **[SRS_REQ_03_3.9]**.
c. The Payroll Administrator makes the desired changes to the employee information. This includes any of the information specified in the Add Timecard sub-flow **[SRS_REQ_03_3.10]** **[E-1: Timecard already submitted]**.
d. Once the Payroll Administrator updates the necessary information, the system updates the Timecard record with the updated information **[SRS_REQ_03_3.11]**.

[A-2: Delete Timecard

a. The system requests that the Payroll Administrator specify the charge number. **[SRS_REQ_03_3.12]**.
b. The Payroll Administrator enters the charge number. The systems retrieve and display the employee information **[SRS_REQ_03_3.13]**.
c. The system prompts the Payroll Administrator to confirm the deletion of the employee **[SRS_REQ_03_3.14]**.
d. The Payroll Administrator verifies the deletion **[SRS_REQ_03_3.15]** **[E-2: Delete Cancelled]**.

Special Requirements

None.

Pre-Conditions

The Payroll Administrator must be logged onto the system before this use case begins.

Post-Conditions

If the use case was successful, the Timecard information is added, updated, or deleted from the system. Otherwise, the system state is unchanged.

Extension Points

[E-1: Timecard Already Submitted]

If, in the Basic Flow, the Employee's current timecard has already been submitted, the system displays a read-only copy of the timecard, so no changes can be made to it by Payroll Administrator. **[SRS_REQ_03_3.16]**.

[E-2: Delete Cancelled]

If in the Delete Timecard sub-flow, the Payroll Administrator decides not to delete the Timecard, the delete is cancelled and the Basic Flow is re-started at the beginning **[SRS_REQ_03_3.17].**

Graphical User Interface (GUI)

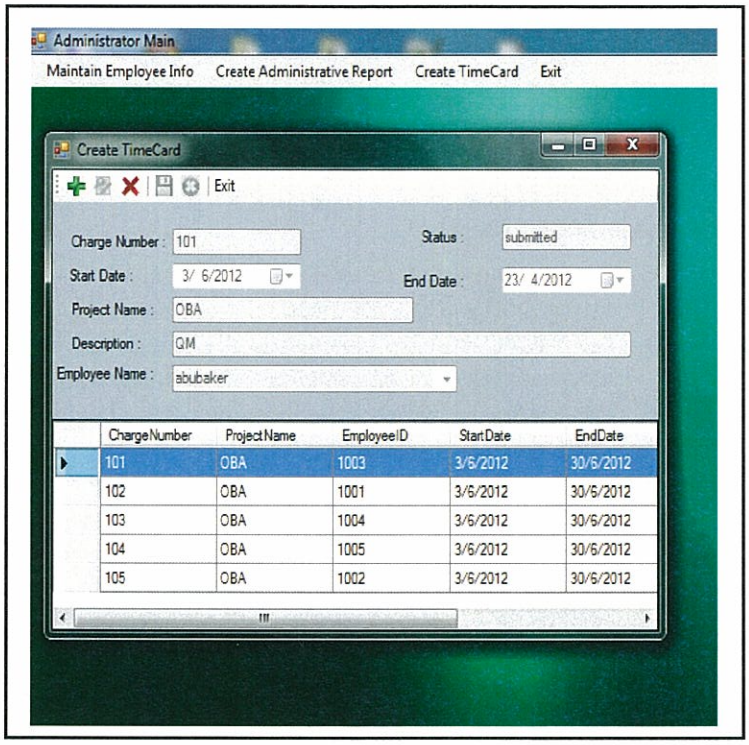

Figure 7 Create Timecard User Interface Diagram

Maintain Timecard Use Case [SRS_REQ_04]

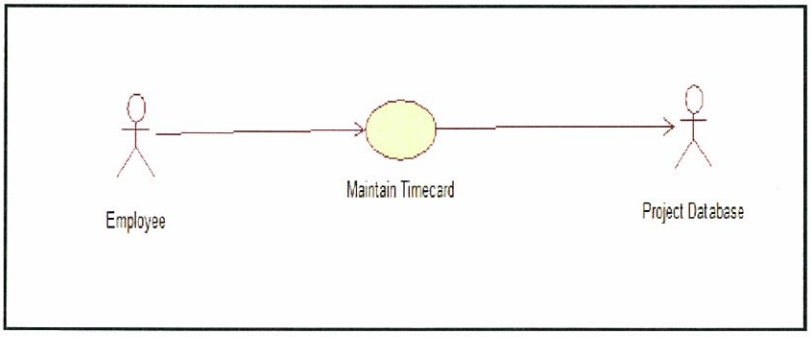

Figure 8 Use Case Diagram of Maintain Timecard

Brief Description

This use case allows the Employee to update and submit timecard information. Hourly and salaried employees must submit weekly timecards recording all hours worked that week and which projects the hours are billed to. An Employee can only make changes to the timecard for the current pay period, before the timecard has been submitted.

Flow of Events

Basic Flow

This use case starts when the Employee wishes to enter hours worked into his current timecard.

a. The system retrieves and displays the current timecard for the Employee. If a timecard does not exist for the Employee for the current pay period, the Payroll Administrator creates a new one. The start and end dates of the timecard are set by the payroll Administrator and cannot be changed by the Employee **[SRS_REQ_04_4.1].**
b. The system retrieves and displays the list of available charge numbers from the Project Management Database **[SRS_REQ_04_4.2].**
c. The Employee selects the appropriate charge numbers and enters the hours worked for any desired date (within the date range of the timecard) **[SRS_REQ_04_4.3] [A-1: Invalid number of Hours].**
d. Once the Employee has entered the information, the system saves the timecard **[SRS_REQ_04_4.4].**

Submit Timecard

a. At any time, the Employee may request that the system submit the timecard **[SRS_REQ_04_4.5].**
b. At that time, the system assigns the current date to the timecard as the submitted date and changes the status of the timecard to "submitted." No changes are permitted to the timecard once it has been submitted **[SRS_REQ_04_4.6].**
c. The system validates the timecard by checking the number of hours worked against each charge number. The total number of hours worked against all charge numbers must not exceed any limit established for the Employee (for example, the Employee may not be allowed to work overtime) **[SRS_REQ_04_4.7] [A-2: Timecard already submitted].**

d. The system retains the number of hours worked for each charge number in the
h. Timecard **[SRS_REQ_04_4.8].**

e. The system saves the timecard **[SRS_REQ_04_4.9].**
f. The system makes the timecard read-only and no further changes are allowed once the timecard is submitted, and this use case ends **[SRS_REQ_04_4.10].**
g. This use case ends **[SRS_REQ_04_4.11].**

Alternative Flows

[A-1: Invalid Number of Hours]

If, in the Basic Flow, an invalid number of hours is entered for a single day (>24), or the number entered exceeds the maximum allowable for the Employee, the system will display an error message and prompt for a valid number of hours. The Employee must enter a valid number, or cancel the operation, at which case the use case ends **[SRS_REQ_04_4.12].**

[A-2: Timecard Already Submitted]

If, in the Basic Flow, the Employee's current timecard has already been submitted, the system displays a read-only copy of the timecard, so no changes can be made to it. The Employee acknowledges the message and the use case ends **[SRS_REQ_04_4.13].**

Special Requirements

None.

Pre-Conditions

The Employee must be logged onto the system before this use case begins.

Post-Conditions

If the use case was successful, the Employee timecard information is saved to the system. Otherwise, the system state is unchanged.

Extension Points

None.

Graphical User Interface (GUI)

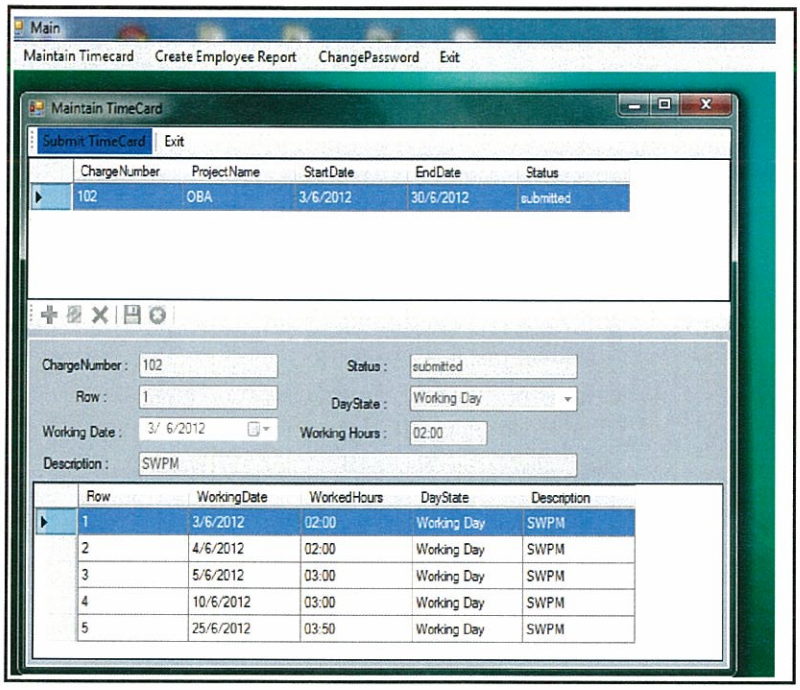

Figure9 Maintain Timecard User Interface Diagram

Create Administrative Report Use Case [SRS_REQ_05]

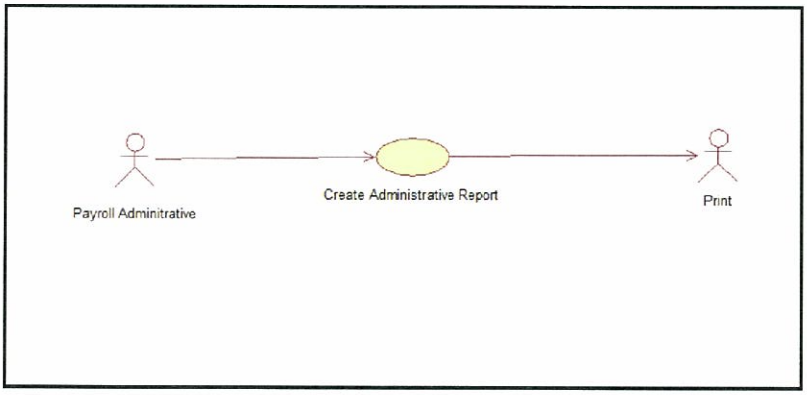

Figure 10 Use Case Diagram of Create Administrative Report

Brief Description

The use case allows the Payroll Administrator to create either a "Total Hours Worked" or "Pay Year to-Date" report.

Flow of Events

Basic Flow

The use case begins when the Payroll Administrator requests that the system create an administrative report.

a. The system requests that the Payroll Administrator specify the following report criteria:
   Report Type (either total hours worked or pay year-to-date),
   Begin and end dates for the report,
   Employee name(s) **[SRS_REQ_05_5.1].**
b. Once the Payroll Administrator provides the requested information, the system provides the Payroll Administrator with a report satisfying the report criteria **[SRS_REQ_05_5.2].**
c. The Payroll Administrator may then request that the system save the report. At which time, the system requests the Payroll Administrator to provide the name and location for saving the report **[SRS_REQ_05_5.3].**
d. Once the Payroll Administrator provides the requested information and confirms the decision to save the report, the system saves the report to the specified name and location **[SRS_REQ_05_5.4].**
e. If the Payroll Administrator did not elect to save the report, the report is discarded **[SRS_REQ_05_5.5].**
f. This use case ends **[SRS_REQ_05_5.6].**

Alternative Flows

None.

Special Requirements

None.

Pre-Conditions

The Payroll Administrator must be logged onto the system in order for this use case to begin.

Post-Conditions

The system state is unchanged by this use case.

Extension Points

None.

Graphical User Interface (GUI)

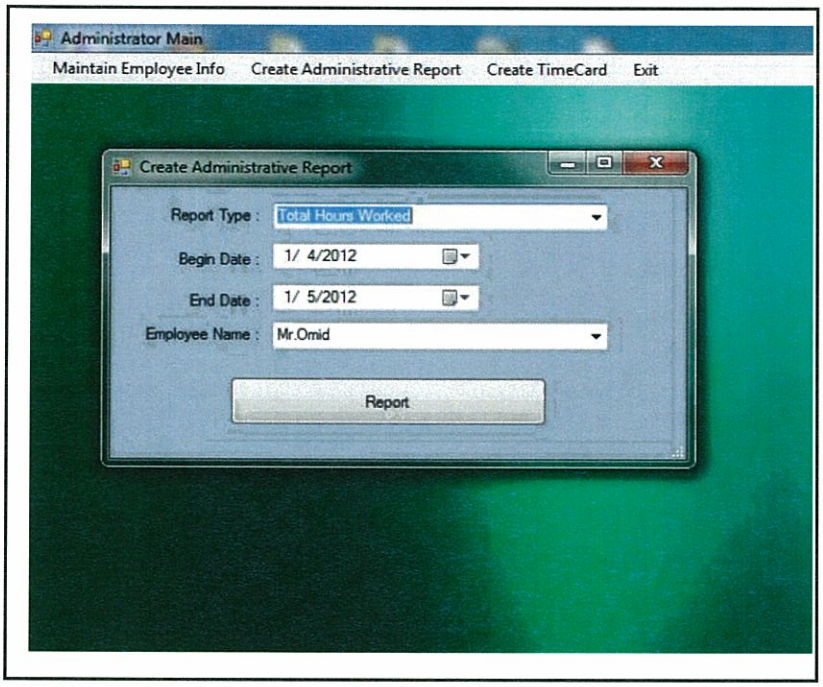

Figure 11 Create Administrative Report User Interface Diagram

Create Employee Report Use Case [SRS_REQ_06]

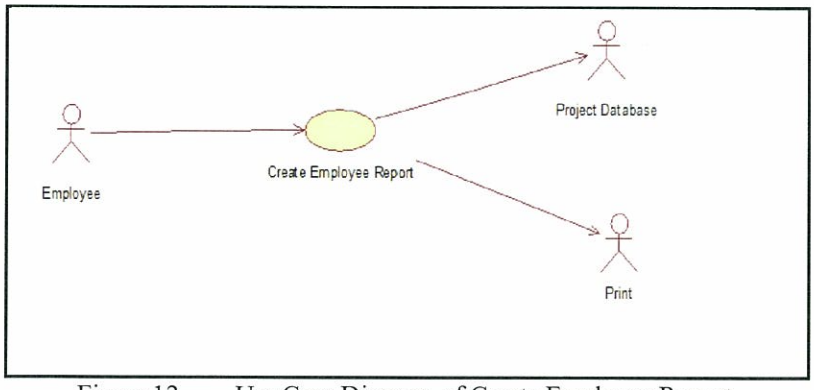

Figure 12 Use Case Diagram of Create Employee Report

Brief Description

The use case allows the Employee to create a "Total Hours Worked," "Total Hours Worked for a Project", "Vacation/Sick Leave," or "Total Pay Year-to-Date" report.

Flow of Events

Basic Flow

This use case starts when the Employee wishes to create a "Total Hours Worked," or "Total Hours Worked for a Project", or "Vacation/Sick Leave," or "Total Pay Year-to- Date" report.

a. The system requests that the Employee specify the following report criteria:
   - Report Type (either "Total Hours Worked," "Total Hours Worked for a Project", "Vacation/Sick Leave," or "Total Pay Year-to-Date")
   - Begin and end dates for the report **[SRS_REQ_06_6.1].**
b. If the Employee selected the "Total Hours Worked for a Project" report, the system retrieves and displays a list of the available charge numbers from the Project Management Database. The system then requests that the Employee select a charge number **[SRS_REQ_06_6.2]**.
c. Once the Employee provides the requested information, the system provides the Employee with a report satisfying the report criteria **[SRS_REQ_06_6.3]**
d. The Employee may then request that the system save the report. At which time, the system requests the Employee to provide the name and location for saving the report **[SRS_REQ_06_6.4]**.
e. Once the Employee provides the requested information and confirms the decision to save the Report, the system saves the report to the specified name and location **[SRS_REQ_06_6.5]**.
f. If the Employee did not elect to save the report, the report is discarded **[SRS_REQ_06_6.6].**
g. This use case ends **[SRS_REQ_06_6.7].**

Alternative Flows

None.

Special Requirements

None.

Pre-Conditions

The Employee must be logged onto the system before this use case begins.

Post-Conditions

The system state is unchanged by this use case.

Extension Points

None.

Graphical User Interface (GUI)

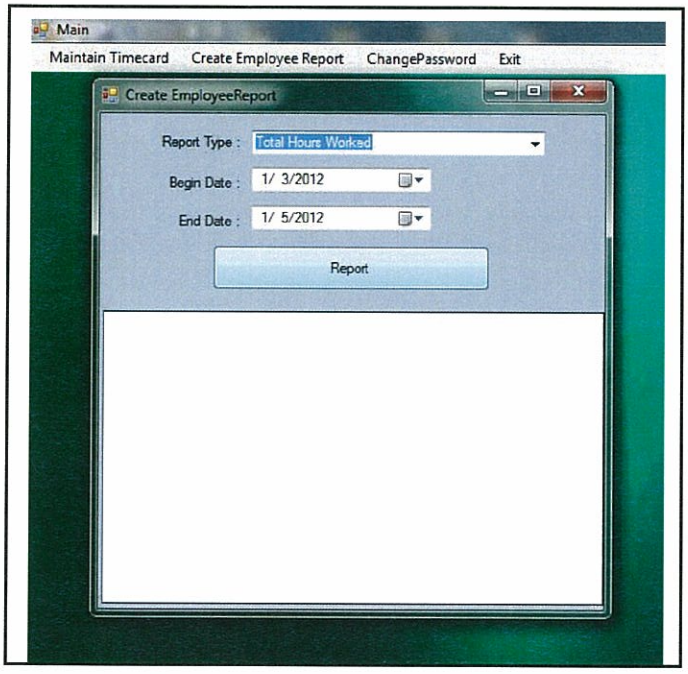

Figure 13 Create Employee Report User Interface Diagram

Run Payroll [SRS_REQ_07]

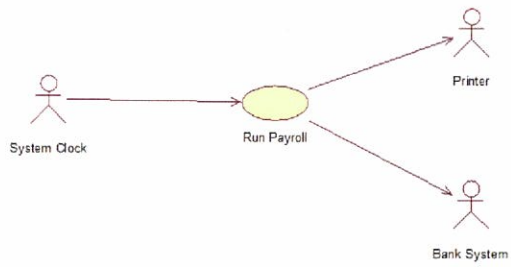

Figure14 Use Case Diagram of Run Payroll

Brief Description

The use case describes how the payroll is run on the every First day of the month.

Flow of Events

Basic Flow

a. The use case begins when it's time to run the payroll. The payroll is run automatically every First day of the month **[SRS_REQ_07_7.1]**
b. The system retrieves all employees who should be paid on the current date **[SRS_REQ_07_7.2].**
c. The system calculates the pay using entered Timecards, employee information and all legal deductions **[SRS_REQ_07_7.3].**
d. If the payment delivery method is mail, the system sends the pay details of the current timecard to the concerned employee by mail **[SRS_REQ_07_7.4].**
e. If the payment delivery method is direct deposit, the system creates a bank transaction and sends it to the Bank System for processing **[SRS_REQ_07_7.5].**
f. The use case ends when all employees receiving pay for the desired date have been processed **[SRS_REQ_07_7.6]**.

Alternative Flows

None.

Special Requirements

None.

Pre-Conditions

None.

Post-Conditions

Payments for each employee eligible to be paid on the current date have been processed.

Extension Points

None.

Graphical User Interface (GUI)

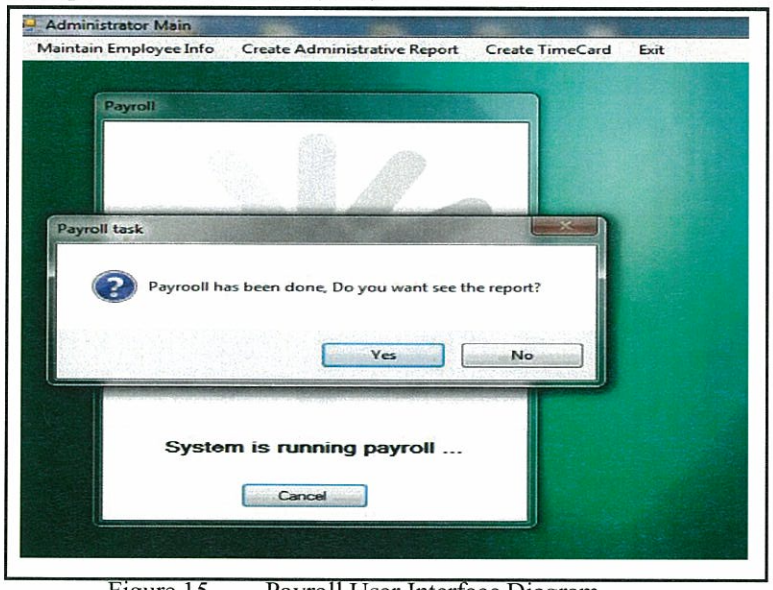

Figure 15 Payroll User Interface Diagram

Non-Functional Requirements

In this chapter we are covering the requirement that was not captured in use case specification.

Functionality

System Error Logging

This system provides an environment in which all the errors captured in a text file with information like date and reason of the error that happened so the developer will be able to check those errors from time to time.

Usability

Windows Compliant

Users shall be able to use windows XP and windows 7 for this system.

Reliability

This section describes all requirements that relate to reliability of the system.

Mean Time between Failures

The mean time between failures must be under 3 ours.

Mean time to repair

Shouldn't be more that 30 minutes.

Performance

The performance characterize of the system describes here.

Simultaneous users

We expect that 1000 users shall be able to use this system at the same time.

Supportability

All requirements that deal with supporting system in the future are described here.

Coding Standards

The main coding standard that used in this system is object oriented programming

Requirement Traceability

No	Requirements		
	Allocated	Reference	Description
1	SRS_REQ_01	2.1.1	Login Use Case

2	SRS_REQ_01 _1.1	2.1.1.2	The actor enters his/her name and password.
3	SRS_REQ_01 _1.2	2.1.1.2	The system validates the entered name and password and logs the actor into the system.
4	SRS_REQ_01 _1.3	2.1.1.2	This use case ends.
5	SRS_REQ_01 _1.4	2.1.1.3	The actor enters an invalid name and/or password, the system displays an error message. The actor can choose to either return to the beginning of the Basic Flow or cancel the login, at which point the use case ends.
6	SRS_REQ_02	2.1.2	Maintain Employee Information Use Case
7	SRS_REQ_02 _2.1	2.1.2.2	The system requests that the Payroll Administrator specify the function he/she would like to Perform (Add an Employee, Update an Employee, or Delete an Employee).
8	SRS_REQ_02 _2.2	2.1.2.2	Once the Payroll Administrator provides the requested information, the sub flow "Add an Employee" is executed.
9	SRS_REQ_02 _2.3	2.1.2.2	The system requests that the Payroll Administrator enter the employee information. This includes: Name Employee type (hour, salaried, commissioned) Mailing address Social security number Standard tax deductions Other deductions Phone number Hourly rate (for hourly employees)

			Salary (for salaried) - hour limit (some Employees may not be able to work overtime)
10	SRS_REQ_02_2.4	2.1.2.2	Once the Payroll Administrator provides the requested information, the system generates and assigns a unique employee id number to the employee and the employee is added to the system.
11	SRS_REQ_02_2.5	2.1.2.2	This use case ends.
12	SRS_REQ_02_2.6	2.1.2.3	The system requests that the Payroll Administrator enter the employee id (for update employee).
13	SRS_REQ_02_2.7	2.1.2.3	The Payroll Administrator enters the employee id. The system retrieves and displays the employee information.
14	SRS_REQ_02_2.8	2.1.2.3	The Payroll Administrator makes the desired changes to the employee information. This includes any of the information specified in the Add an Employee sub-flow.
15	SRS_REQ_02_2.9	2.1.2.3	Once the Payroll Administrator updates the necessary information, the system updates the employee record with the updated information.
16	SRS_REQ_02_2.10	2.1.2.3	The system requests that the Payroll Administrator specify the employee id (for delete an employee).
17	SRS_REQ_02_2.11	2.1.2.3	The Payroll Administrator enters the employee id. The system retrieves and displays the employee information.

18	SRS_REQ_02 _2.12	2.1.2.3	The system prompts the Payroll Administrator to confirm the deletion of the Employee.
19	SRS_REQ_02 _2.13	2.1.2.3	The Payroll Administrator verifies the deletion.
20	SRS_REQ_02 _2.14	2.1.2.3	The system marks the employee record for deletion. The next time the payroll is run, the system will generate a final pay check for the deleted employee and remove the Employee from the system.
21	SRS_REQ_02 _2.15	2.1.2.7	The Update an Employee or Delete an Employee sub-flows, an employee with the specified id number does not exist; the system displays an error message. The Payroll Administrator can then enter a different id number or cancel the operation, at which point the use case ends.
22	SRS_REQ_02 _2.16	2.1.2.7	If in the Delete an Employee sub-flow, the Payroll Administrator decides not to delete the Employee, the delete is cancelled and the Basic Flow is re-started at the beginning
23	SRS_REQ_03	2.1.3	Create Timecard Use Case
24	SRS_REQ_03 _3.1	2.21.3.2	The system requests that the Payroll Administrator specify the function he/she would like to Perform (Add Timecard or Update Timecard, or Delete Timecard).
25	SRS_REQ_03 _3.2	2.1.3.2	Once the Payroll Administrator provides the requested information, one of the sub flows is executed. If the Payroll Administrator selected

			"Add Timecard", the Add Timecard sub flow is executed. If the Payroll Administrator selected "Update Timecard", the Update Timecard sub flow is executed. If the Payroll Administrator selected "Delete Timecard", the Delete Timecard sub flow is executed.
26	SRS_REQ_03_3.3	2.1.3.2	The system requests that the Payroll Administrator enter the employee information. This includes: Name Start and End dates Name of project Description if any
27	SRS_REQ_03_3.4	2.1.3.2	Once the Payroll Administrator provides the requested information, the system generates and assigns a charge number to the employee and the Timecard is created.
28	SRS_REQ_03_3.5	2.1.3.2	The system provides the Payroll Administrator with the new charge number.
29	SRS_REQ_03_3.6	2.1.3.2	Once the Employee has entered the information, the system saves the create timecard.
30	SRS_REQ_03_3.7	2.1.3.2	This use case ends.
31	SRS_REQ_03_3.8	2.1.3.3	The system requests that the Payroll Administrator enter the charge number (update create timecard).
32	SRS_REQ_03_3.9	2.1.3.3	The Payroll Administrator enters the charge number. The system retrieves and displays the

			employee information.
33	SRS_REQ_03 _3.10	2.1.3.3	The Payroll Administrator makes the desired changes to the employee information. This includes any of the information specified in the Add Timecard sub-flow.
34	SRS_REQ_03 _3.11	2.1.3.3	Once the Payroll Administrator updates the necessary information, the system updates the timecard record with the updated information.
35	SRS_REQ_03 _3.12	2.1.3.3	The system requests that the Payroll Administrator specify the charge number (delete create timecard).
36	SRS_REQ_03 _3.13	2.1.3.3	The Payroll Administrator enters the charge number. The systems retrieve and display the employee information.
37	SRS_REQ_03 _3.14	2.1.3.3	The system prompts the Payroll Administrator to confirm the deletion of the employee.
38	SRS_REQ_03 _3.15	2.1.3.3	The Payroll Administrator verifies the deletion.
39	SRS_REQ_03 _3.16	2.1.3.7	If, in the Basic Flow, the Employee's current timecard has already been submitted, the system displays a read-only copy of the timecard, so no changes can be made to it by Payroll Administrator.
40	SRS_REQ_03 _3.17	2.1.3.7	If in the Delete Timecard sub-flow, the Payroll Administrator decides not to delete the Timecard, the delete is cancelled and the Basic

			Flow is re-started at the beginning
41	SRS_REQ_04	2.1.4	Maintain Timecard Use Case
42	SRS_REQ_04 _4.1	2.1.4.2	The system retrieves and displays the current timecard for the Employee. If a timecard does not exist for the Employee for the current pay period, the system creates a new one. The start and end dates of the timecard are set by the system and cannot be changed by the Employee.
43	SRS_REQ_04 _4.2	2.1.4.2	The system retrieves and displays the list of available charge numbers from the Project Management Database.
44	SRS_REQ_04 _4.3	2.1.4.2	The Employee selects the appropriate charge numbers and enters the hours worked for any . desired date (within the date range of the timecard).
45	SRS_REQ_04 _4.4	2.1.4.2	Once the Employee has entered the information, the system saves the timecard.
46	SRS_REQ_04 _4.5	2.1.4.2	At any time, the Employee may request that the system submit the timecard.
47	SRS_REQ_04 _4.6	2.1.4.2	At that time, the system assigns the current date to the timecard as the submitted date and changes the status of the timecard to "submitted." No changes are permitted to the timecard once it has been submitted
48	SRS_REQ_04 _4.7	2.1.4.2	The system validates the timecard by checking the number of hours worked against each charge number. The total number of hours

			worked against all charge numbers must not exceed any limit established for the Employee (for example, the Employee may not be allowed to work overtime).
49	SRS_REQ_04 _4.8	2.1.4.2	The system retains the number of hours worked for each charge number in the Timecard.
50	SRS_REQ_04 _4.9	2.1.4.2	The system saves the timecard.
51	SRS_REQ_04 _4.10	2.1.4.2	The system makes the timecard read-only, and no further changes are allowed once the timecard is submitted, and this use case ends.
52	SRS_REQ_04 _4.11	2.1.4.3	The Project Management Database is not available; the system will display an error message stating that the list of available charge numbers is not available. The Employee acknowledges the error and may either choose to continue (without selectable charge numbers), or to cancel (any timecard changes are discarded and the use case ends).
53	SRS_REQ_04 _4.12	2.1.4.3	The Employee's current timecard has already been submitted; the system displays a read-only copy of the timecard and informs the Employee that the timecard has already been submitted, so no changes can be made to it. The Employee acknowledges the message and the use case ends.
54	SRS_REQ_05	2.1.5	Create Administrative Report

55	SRS_REQ_05 _5.1	2.1.5.2	The system requests that the Payroll Administrator specify the following report criteria: - Report Type (either total hours worked or pay year-to-date) - Begin and end dates for the report, - Employee name(s).
56	SRS_REQ_05 _5.2	2.1.5.2	Once the Payroll Administrator provides the requested information, the system provides the Payroll Administrator with a report satisfying the report criteria.
57	SRS_REQ_05 _5.3	2.1.5.2	The Payroll Administrator may then request that the system save the report. At which time, the system requests the Payroll Administrator to provide the name and location for saving the report.
58	SRS_REQ_05 _5.4	2.1.5.2	Once the Payroll Administrator provides the requested information and confirms the decision to save the report, the system saves the report to the specified name and location
59	SRS_REQ_05 _5.5	2.1.5.2	If the Payroll Administrator did not elect to save the report, the report is discarded.
60	SRS_REQ_05 _5.6	2.1.5.2	This use case ends.
61	SRS_REQ_03	2.1.6	Create Employee Report Use Case
62	SRS_REQ_06 _6.1	2.1.6.2	The system requests that the Employee specify the following report criteria: - Report Type (either "Total Hours Worked," "Total Hours Worked for a Project". "Vacation/Sick Leave," or "Total Pay Year-to-

			Date") - Begin and end dates for the report.
63	SRS_REQ_06_6.2	2.1.6.2	If the Employee selected the "Total Hours Worked for a Project" report, the system retrieves and displays a list of the available charge numbers from the Project Management Database. The system then requests that the Employee select a charge number.
64	SRS_REQ_06_6.3	2.1.6.2	Once the Employee provides the requested information, the system provides the Employee with a report satisfying the report criteria.
65	SRS_REQ_06_6.4	2.1.6.2	The Employee may then request that the system save the report. At which time, the system requests the Employee to provide the name and location for saving the report.
66	SRS_REQ_06_6.5	2.1.6.2	Once the Employee provides the requested information and confirms the decision to save the Report, the system saves the report to the specified name and location.
67	SRS_REQ_06_6.6	2.1.6.2	If the Employee did not elect to save the report, the report is discarded.
68	SRS_REQ_06_6.7	2.1.6.2	This use case ends.
69	SRS_REQ_07	2.1.7	Run Payroll Use Case
70	SRS_REQ_07_7.1	2.1.7.2	The use case begins when it's time to run the payroll. The payroll is run automatically every Friday and the last working day of the month.
71	SRS_REQ_07_7.2	2.1.7.2	The system retrieves all employees who should be paid on the current date.
72	SRS_REQ_07_7.3	2.1.7.2	The system calculates the pay using entered timecards, purchase orders,

			employee information (e.g., salary, benefits, etc.) and all legal deductions.
73	SRS_REQ_07_7.4	2.1.7.2	If the payment delivery method is mail or pick-up, the system prints a pay check.
74	SRS_REQ_07_7.5	2.1.7.2	If the payment delivery method is direct deposit, the system creates a bank transaction and sends it to the Bank System for processing.
75	SRS_REQ_07_7.6	2.1.7.2	The use case ends when all employees receiving pay .for the.

Appendix B Software Design Document

Payroll System

Prepared for:

Advance Informatics School

(AIS)

Prepared by:

JALAL SHAH

UNIVERSITI TEKNOLOGI MALAYSIA

Payroll System

Software Design Document

(SDD)

Introduction

Purpose

Software design is a process by which the software requirements are translated into a representation of software components, interfaces, and data necessary for the implementation phase. The SDD shows how software system will be structured to satisfy requirements. It is primary reference for code development and, therefore, it must contain all the information required by a programmer to write code. The SDD is performed in two stages. The first is a preliminary design in which the overall system architecture and data architecture is defined. In the second stage, the detailed design stage, more detailed data structures are defined and algorithms are developed for the defined architecture.

This SDD adapted from the IEEE Recommended Practice for Software Design Descriptions. The IEEE Recommended Practice for Software Design Descriptions has been reduced in order to simplify this assignment while still retaining the main components and providing a general idea of a project definition report.

Scope

This Software Design Description provides an architectural overview of the Payroll System. This system is designed in such a way to solve the different issues of existing Payroll System which was using before in Acme and was hopelessly out of date.
There are six types of actors namely Employee, Payroll Administrator, Project Database, Printer, System Clock and Bank System in this system. Employee can login in the system to create employee report, maintain timecard. Payroll Administrator can login in the system to maintain employee info, create timecard and to create administrative report. This system also deals with Bank System, System Clock, Printer and Project Database for the financial tasks.

Overview

The Software Design Document is divided into 5 sections with various subsections. The sections of the Software Design Document are:

1) Introduction
2) System Overview
3) System Architecture
4) Data Design
5) Human Interface Design

Reference Material

Software Design Document (SDD) Template exists on website with this address:

http://macs.citadel.edu/verdicchiom/CSCI602-FA2011/docs/Design.pdf

Definitions and Acronyms

Bank System

Any bank(s) to which direct deposit transactions are sent.

Employee

A person that works for the company that owns and operates the payroll system (Acme, Inc.)

Payroll Administrator

The person responsible for maintaining employees and employee information in the system.

Project Database

The legacy database that contains all information regarding projects and charge numbers.

System Clock

The internal system clock that keeps track of time. The internal clock will automatically run the payroll at the appropriate times.

Payment Method

How the employee is paid, either pick-up, mail, or direct deposit.

Timecard

A record of hours worked by the employee during a specified pay period.

Salaried Employee

An employee that receives a salary.

Hourly Employee

An employee that works in limited number of hours.

Document Overview

This specification defines the software requirements for the Payroll System Software. It has been prepared using MIL-STD-498 for guidance. This SDD is organized as follows:

Chapter 2 [Deployment Diagram] – Provides the deployment diagram for payroll system.

Chapter 3 [Architectural Design] – provides the architectural Design of payroll system with defining the classes in more detail.

Chapter 4 [Data_ Design] -Specifies the all data used in payroll system with the data type.

Chapter 5 [Use case realization] – Provides the sequence diagram for all flows.

Chapter 6 [Graphical User Interface] – Provides all interfaces used inside the payroll system.

Deployment Diagram

The Payroll System is responsible for high level operations, it allow the employee to matins their timecard and create employee report using external Desktop PCs which is connected to Server via LAN.
Payroll Administrator can use local Desktop PC that is connected to the Server via LAN to maintain employee information, create timecard and create administrative report.

The system clock run payroll on the 1st day of every month and the administrator send the payment report to the bank.

Another view of the Payroll System and the subsystems that deal with it and how these subsystems collaborate with each other is shown in Payroll System deployment diagram which is illustrate in Figure 1.

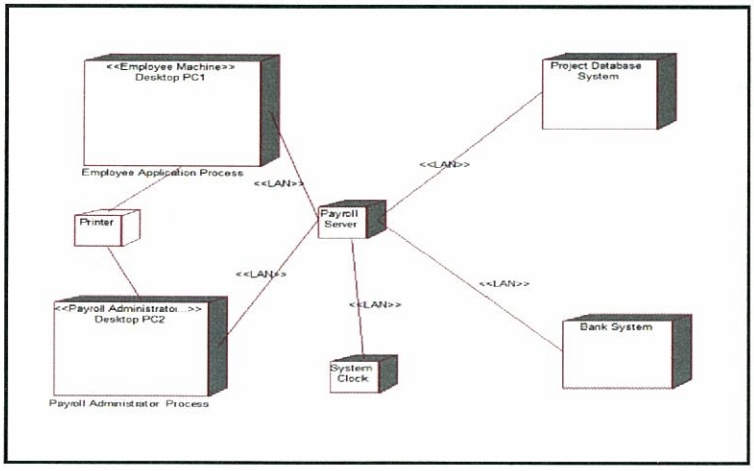

Figure 1 Deployment Diagram of Payroll System

Desktop PC1

The Employee login to Payroll System, continued with employee application process, by using external PCs, which are connected to the Payroll Server via LAN network.

Desktop PC2

The Payroll Administrator login to Payroll System, continued with payroll administrator process, by using external PCs, which are connected to the Payroll Server via LAN network.

Project Database System

The Project Database System, that contains all information regarding projects and charge numbers, which is connected to Payroll Server via LAN network.

Bank System

The Bank System, to which direct deposit transactions are sent, which is connected to Payroll Server via LAN network.

System Clock

The internal system clock that keeps track of time. This is connected to Payroll Server via LAN network.

Payroll Server

The Payroll Server is the main server; all users have access to the main server via LAN network to access to the payroll System.

Printer

The printer is used for printing purpose, which is connected to Desktop PC1 and Desktop PC2 via a LAN network.

Architectural Design

The diagram below shows the architectural design of Payroll system. The payroll architecture follows the layered architecture to achieve modularity, high-cohesion, low- coupling and maintainability. Low coupling is achieved by separating business- logic from general technical services such as the data-access and application from business system.

The payroll architecture consists of three layers to communicate between each other and satisfy functionalities namely, "Application" layer, "Business" layer and "Data" layer.

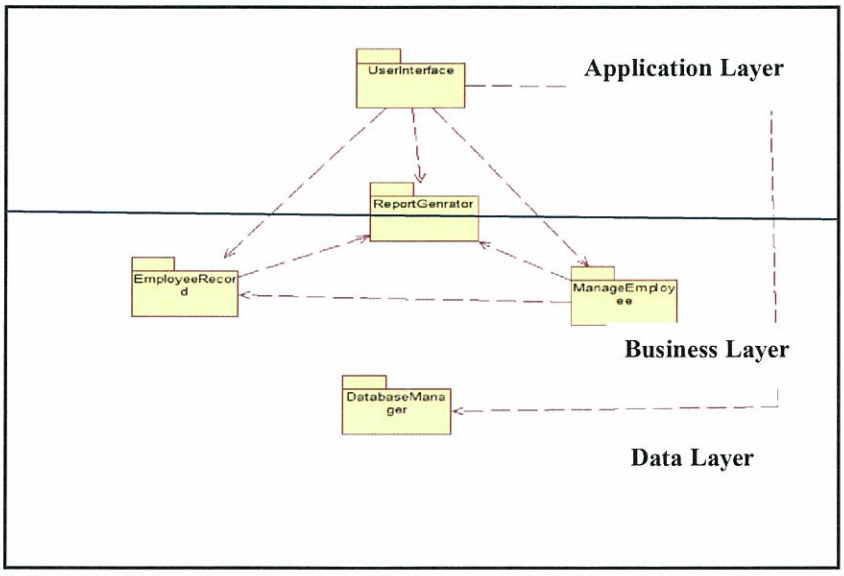

Figure 2 Architectural Design Diagram

User Login Page

Name of Class: FrmLogin

Description

Operations

Name: Form1_Load

a. Arguments: None
b. Returns: No return Value
c. Pre-condition: Connected to the page
d. Post-condition: Show the login page
e. Exceptions: None

Flow of Events

User wants to login into the system.

Name: btncancel_Click

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Connected to the login page
d. Post-condition: Close the login page

Flow of Events

a. User wants to login into the system.
b. User clicks the Cancel button to close the login page.

Name: btnlogin_Click

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Connected to the login page
d. Post-condition:showthenextpageFrmMainEmployee / FrmMainAdministrative
e. Exceptions: None

Flow of Events

a. User wants to login into the system.
b. User enters the user name and password.
c. User clicks the login button to login into the next page.

Main Employee page

. Name of class: FrmMainEmployee

Description

This is the main page for employee, whereas employee can only access after login into the system. There are three buttons. One for maintain timecard, one for Create employee report, one for change password and one for exit from the system.

Operations

Name: maintainTimecardToolStripMenuIten_Click

a. Arguments: None
b. Return: No return value
c. Pre-condition: Employee login to the system
d. Post-condition: Display Maintain Timecard page
e. Exceptions: None

Flow of Events

a. Employee successfully login into the Payroll System.
b. Employee clicks the Maintain Timecard button.
c. System displays the maintain timecard page.

Name: createEmployeeToolStripMenuItem( )

a. Arguments: None
b. Return: No return value
c. Pre-condition: Employee login to the system
d. Post-condition: Display create employee report page

Flow of Events

a. Employee successfully login into the Payroll System.
b. Employee clicks the create employee report button.
c. System displays the main create employee report page. Name: FrmChangePassword ()
a. Arguments: None
b. Return: No return value
c. Pre-condition: Employee login to the system
d. Post-condition: Display create employee report page

Flow of Events

a. Employee successfully login into the Payroll System.
b. Employee clicks the change password button.
c. System displays the main change password page.

Name: existToolStripMenuItem_Click ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Employee login to the system
d. Post-condition: Terminated from the main employee page

Flow of Events

d. Employee successfully login into the Payroll System.
e. Employee clicks the exit button.
f. System closed the main employee page.

Main Administrative page

Name of Class: FrmMainAdministrative

Description

This is the main page for Payroll Administrator, whereas Payroll Administrator can only access after login into the system. There are three buttons. One for

maintain employee information, one for create timecard, one for Create administrative report, and one for exit from the system.

Operations

Name: maintainEmployeeInfoToolStripMenuItem_Click ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Payroll Administrator login to the system
d. Post-condition: Display Maintain Employee information page

Exceptions: None

- **Flow of Events**

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks on maintain employee information button.
c. System displays maintain employee information page.

Name: FrmCreateTimecard ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Payroll Administrator login to the system
d. Post-condition: Display Create Timecard page
e. Exceptions: None

- **Flow of Events**

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks on create timecard button.
c. System displays create timecard page.

Name: FrmCreateAdministrativereport ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Payroll Administrator login to the system
d. Post-condition: Display create Administrative repo page
e. Exceptions: None

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks on create Administrative report button.
c. System displays create Administrative report page.

Name: existToolStripMenuItem_Click ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Payroll Administrator login to the system
d. Post-condition: Terminated from the main Administrator page

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks the exit button.
c. System closed the main Administrator page.

Maintain Employee information page

Name of Class: FrmMaintainEmployeeInfo

Description

This is the maintain employee information page where Payroll Administrator can add, delete or edit an employee's information. There are six buttons one for add an employee, one for delete an employee, one for edit an employee, one for save the changes, one for cancel and one for exit from the maintain employee information.

Operations:

Name: btnNew_click()

a. Arguments: State
b. Type: String
c. Returns: No return Value
d. Pre-condition: Login to the system
e. Post-condition: Save the changes if any made
f. Exceptions: None

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks add button to add new employee information.
c. Payroll Administrator clicks delete button to delete employee information.
d. Payroll Administrator clicks edit button to edit in employee's information.
e. Payroll Administrator clicks save button to save the changes in employee's information.
f. Payroll Administrator clicks cancel button to cancel the edit in employee's information.

Name: ToolStripLabel1_Click ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Payroll Administrator login to the system
d. Post-condition: Terminated from the maintain employee information page

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks the exit button.
c. System closed the maintain employee information page.

Created Timecard page

Name of Class: FrmCreateTimecard ()

Description

This is the create timecard page where Payroll Administrator can add, delete or edit an employee's information in the timecard. There are six buttons one for add timecard, one for delete timecard, one for edit in timecard, one for save the changes, one for cancel the changes and one for exit from the maintain employee information.

Operations

Name: btnNew_click()

a. Arguments: State
b. Type: String
c. Returns: No return value
d. Pre-condition: Login to the system
e. Post-condition: Save the changes if any made
f. Exceptions: None

Flow of Events

Payroll Administrator successfully login into the Payroll System.

a. Payroll Administrator clicks add button to create timecard for employee.
b. Payroll Administrator clicks delete button to delete the timecard.
c. Payroll Administrator clicks edit button to edit in create timecard.
d. Payroll Administrator clicks save button to save the changes in create timecard.
e. Payroll Administrator clicks cancel button to cancel the edit in create timecard.

Name: ToolStripLabel1_Click ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Payroll Administrator login to the system
d. Post-condition: Terminated from the create timecard page

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks the exit button.
c. System closed the create timecard page.

Create Administrative Report page

Name of Class: FrmCreateAdministrativeReport ()

Description

This is the create Administrative report page where Payroll Administrator can see the report by selecting the report type and the start and end dates. There is one button for displaying the selected report.

Operations

Name: FrmCreateAdministrativeReport_Load()

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Login to the system
d. Post-condition: Display the report
e. Exceptions: None

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator selects the report type.
c. Payroll Administrator enters the start and end date for the report.

Name: btnReprot_Click()

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Login to the system
d. Post-condition: Display the report
e. Exceptions: None

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator click the button report.
c. System will display the report.

Name: ToolStripLabel1_Click ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Payroll Administrator login to the system
d. Post-condition: Terminated from the create administrative page

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks the exit button.
c. System closed the create timecard page.

Maintain Timecard page

Name of Class: FrmMaintainTimecard ()

Description

This is the maintain timecard page where employee can add, delete or edit an in the timecard. There are seven buttons one for add timecard, one for delete timecard, one for edit in timecard, one for save the changes, one for cancel the changes, one for submit the timecard and one for exit from the maintain timecard.

Operations:

Name: btnNew_click()

a. Arguments: State
b. Type: String
c. Returns: No return value
d. Pre-condition: Login to the system
e. Post-condition: Save the changes if any made
f. Exceptions: None

Flow of Events

a. Employee successfully login into the Payroll System.
b. Employee clicks add button to create timecard.
c. Employee clicks delete button to delete the timecard.
d. Employee clicks edit button to edit in timecard.

e. Employee clicks save button to save the changes in timecard.
f. Employee clicks cancel button to cancel the edit in timecard.
g. Employee clicks submit button to submit the timecard.

Name: ToolStripLabel1_Click ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Employee login to the system
d. Post-condition: Terminated from the maintain timecard page

Flow of Events

a. Employee successfully login into the Payroll System.
b. Employee clicks the exit button.
c. System closed the maintain timecard page.

Create Employee Report page

Name of Class: FrmCreateEmployeeReport ()

Description

This is the create employee report page where Payroll Administrator can see the report by selecting the report type and the start and end dates. There is one button for displaying the selected report.

Operations:

Name: FrmCreateAdministrativeReport_Load()

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Login to the system

d. Post-condition: Display the report
e. Exceptions: None

Flow of Events

a. Employee successfully login into the Payroll System.
b. Employee selects the report type.
c. Employee enters the start and end date for the report.

Name: btnReprot_Click()

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Login to the system
d. Post-condition: Display the report
e. Exceptions: None

Flow of Events

a. Employee successfully login into the Payroll System.
b. Employee click the button report.
c. System will display the report.

Name: ToolStripLabel1_Click ()

a. Arguments: None
b. Return: No return value
c. Pre-condition: Employee login to the system
d. Post-condition: Terminated from the create employee report page

Flow of Events

a. Employee successfully login into the Payroll System.
b. Employee clicks the exit button.
c. System closed the create timecard page.

Database Manager

Name of Class: DBManager

Description

It provides the connection with the database, with the help of this class any data can be add, edit or deleted from the database of the payroll system.

Operations:

Name: initial ()

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Login to the system
d. Post-condition: save changes into the database
e. Exceptions: None

Flow of Events

Make a connection string to open a connection to the database.

Name: fillDS ()

a. Arguments: query
b. Type: String
c. Returns: DataSet
d. Pre-condition: Login to the system
e. Post-condition: Save changes into the database
f. Exceptions: None

Flow of Events

The connection that are used to fill the system database.

Run Payroll

Name of Class: FrmPayRoll

Description

This is the Payroll page where Payroll Administrator can see the report. The Payroll system will run automatically by the system clock. There are two options either the Payroll Administrator run the Payroll are cancel the operation.

Operations

Name: backgroundWorker1_DoWork()

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Login to the system
d. Post-condition: Display the report
e. Exceptions: None

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll run automatically run on first day of every month.
c. Payroll Administrator can run the payroll by clicking yes.

Name: btnCancel_Click()

a. Arguments: None
b. Returns: No return value
c. Pre-condition: Login to the system
d. Post-condition: exist from the run payroll
e. Exceptions: None

Flow of Events

a. Payroll Administrator successfully login into the Payroll System.
b. Payroll Administrator clicks the exit button.
c. System closed the run payroll.

Data Structure Design

The data is stored in a relational database of Payroll Database using SQL expression. The relations are described by the database administrator. The fields fro transmitting to and from the database are given in fallowing table.

Data field types and sizes

Table 1 Data Field Types and Sizes

Attribute Name	Attribute Type	Attribute Size
EmployeeID	int	4
Name	varchar	30
Email	varchar	25
Phone	nvarchar	15
EmployeeType	varchar	15
SocialSecurityNumber	int	4
Tax	float	8
OtherDeduction	float	8
HourlyRate	Int	8
Salary	Float	8
HourLimit	varchar	5
AccountNumber	String	50
UserName	nvarchar	20

UPassword	nvarchar	length
User Type	Boolean	1
Password*#	String	10

Use Case: Login

See detail of login in Software Requirements Specifications.

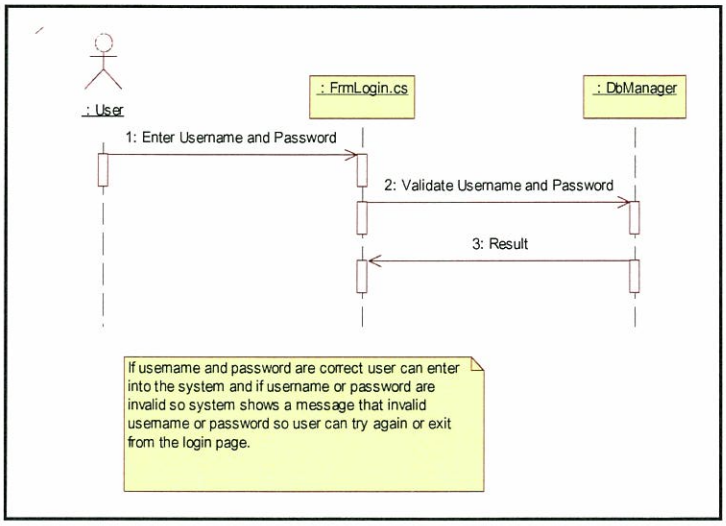

Figure 4 Sequence diagram for login use case

Use Case: Create Timecard

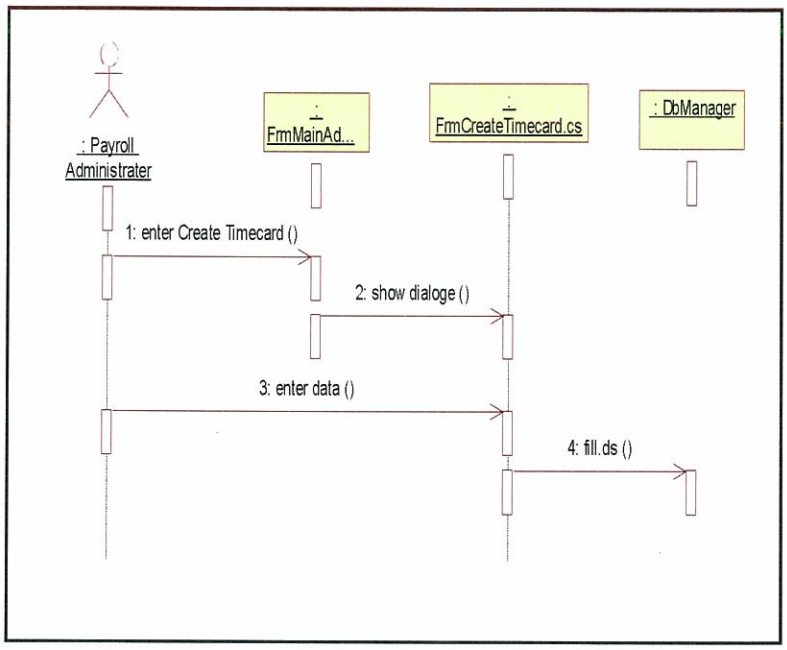

Figure 5 Sequence diagram of Create Timecard

Use Case: Create Administrative Report

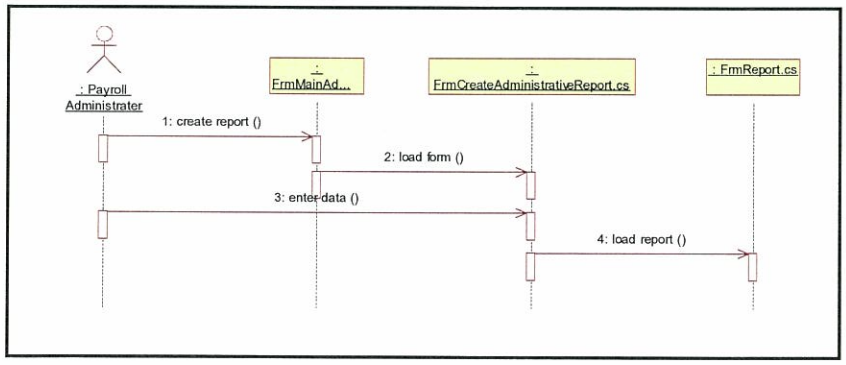

Figure 6 sequence diagram for Create Administrative Report use case

Use Case: Create Employee Report

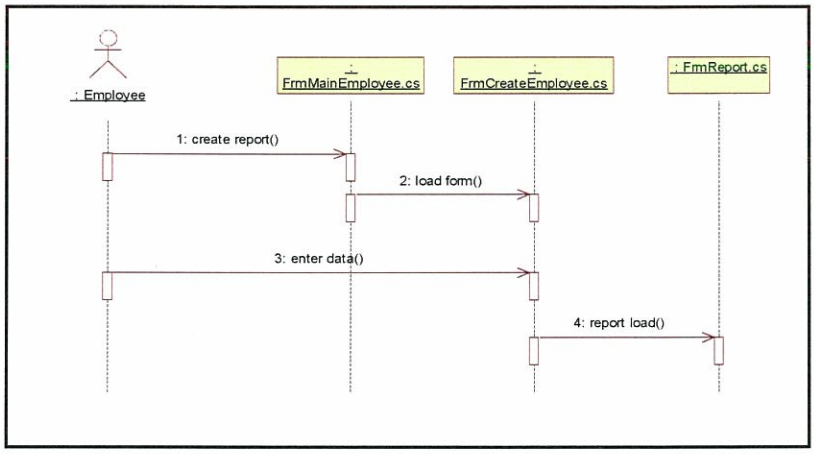

Figure7 sequence diagram for Create employee Report use case

Use Case: Maintain Employee Information

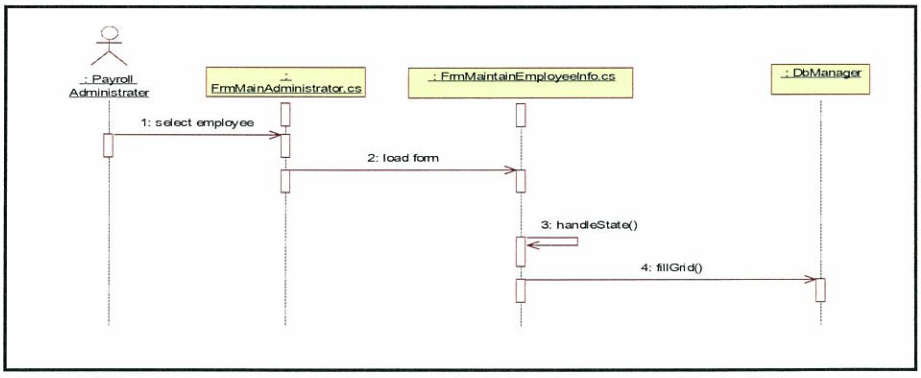

Figure 8 Sequence diagram for Maintain Employee Information

Use Case: Maintain Timecard

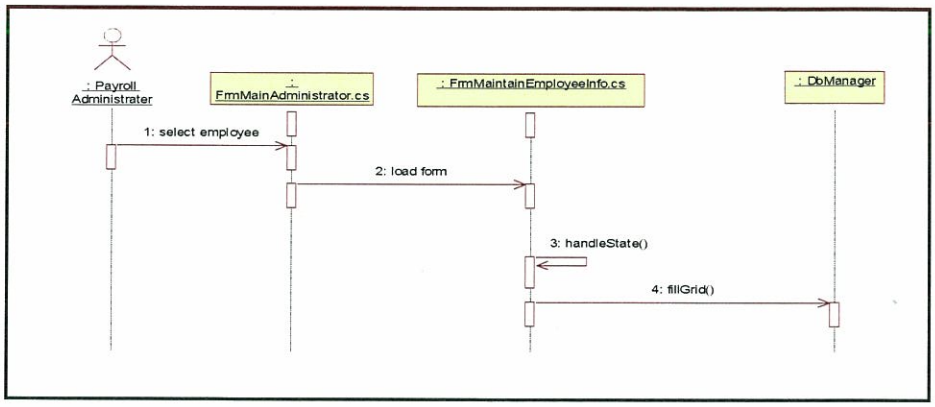

Figure 9 Sequence Diagram for Maintain Employee information Use case

Interface Design

The user interfaces are shown below.

Figure 10 Login page

Figure11 Main Administrator Page

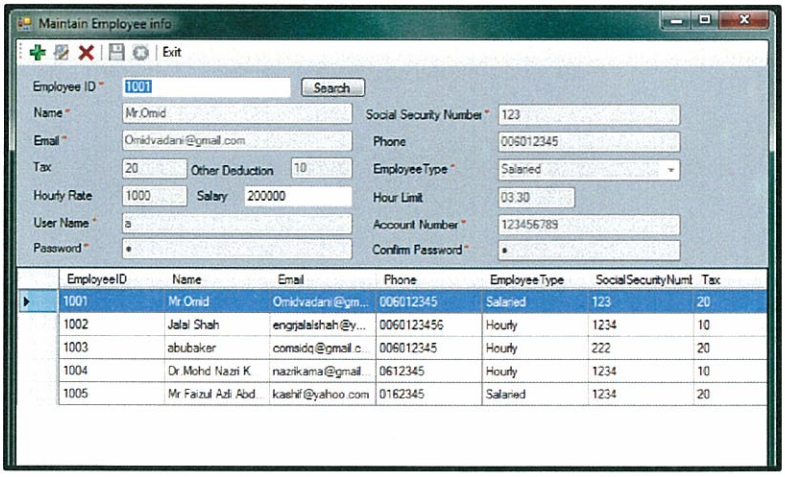

Figure 12 Maintain Employee Information page

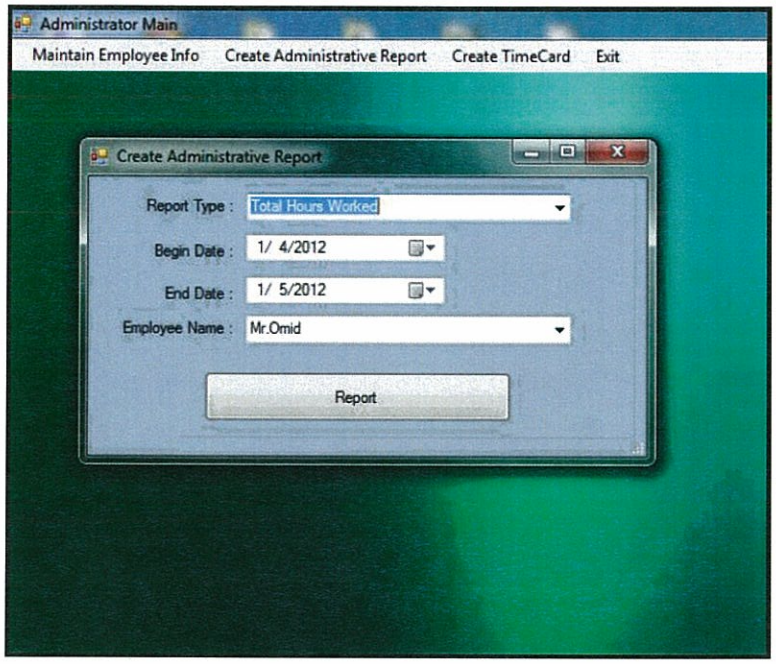

Figure 13 Create Administrative Report Page

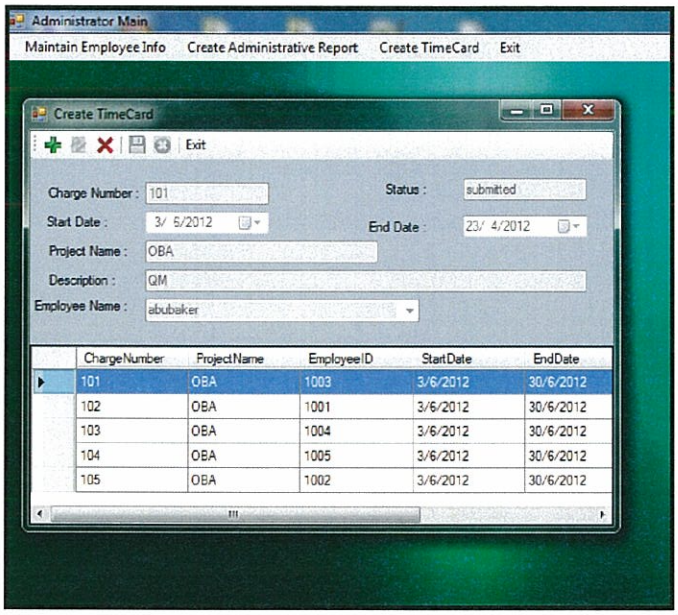

Figure 14 Create Timecard page

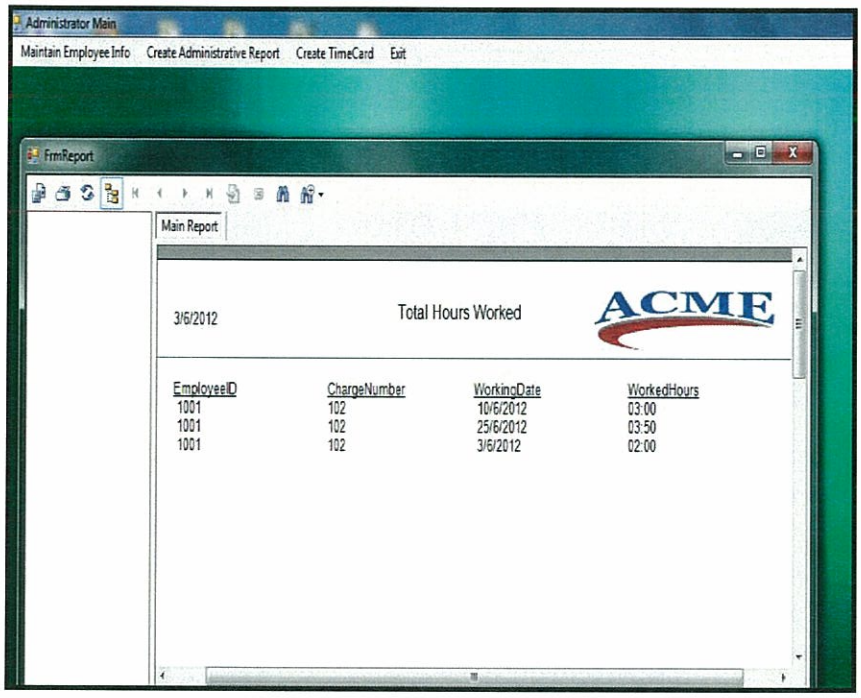

Figure 15 Total Hours Worked Report

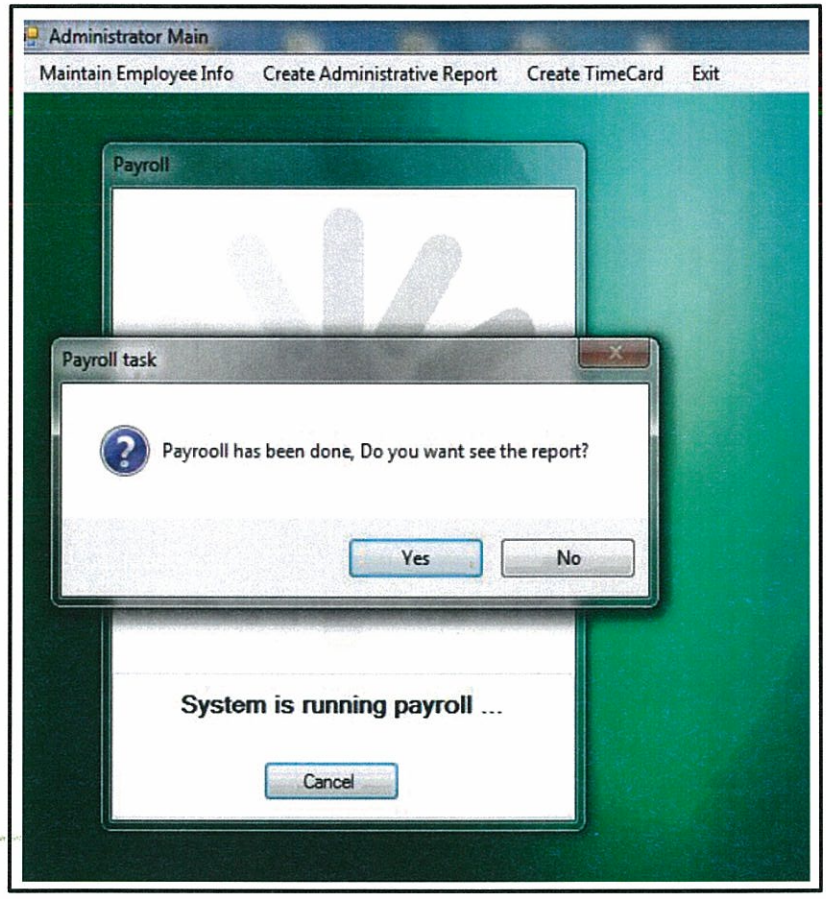

Figure16 Run Payroll Page

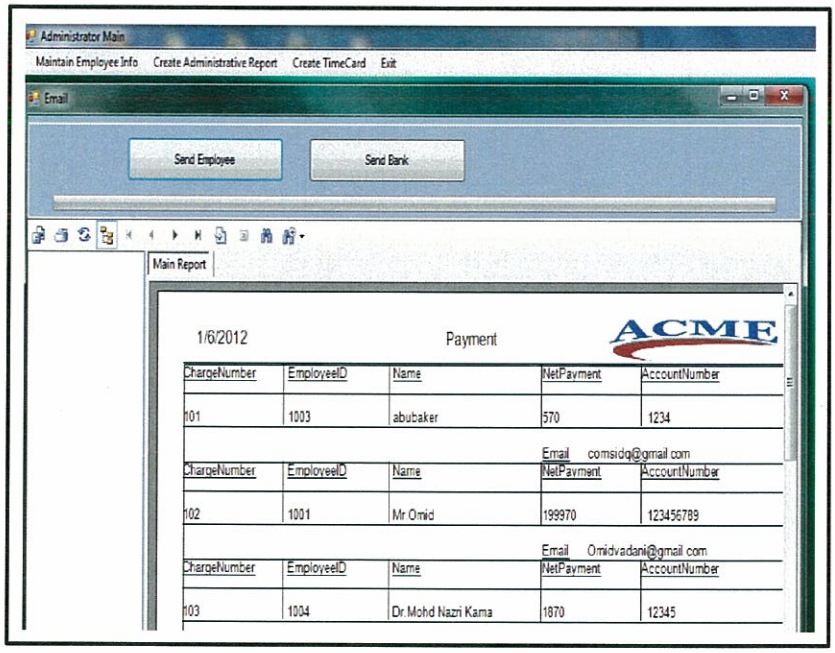

Figure 17 Main Payroll Payment Report

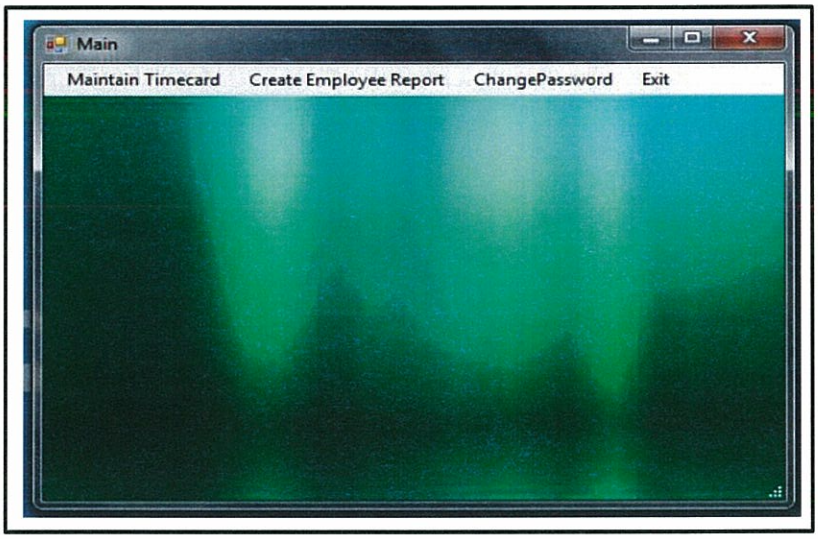

Figure 18 Main Employee page

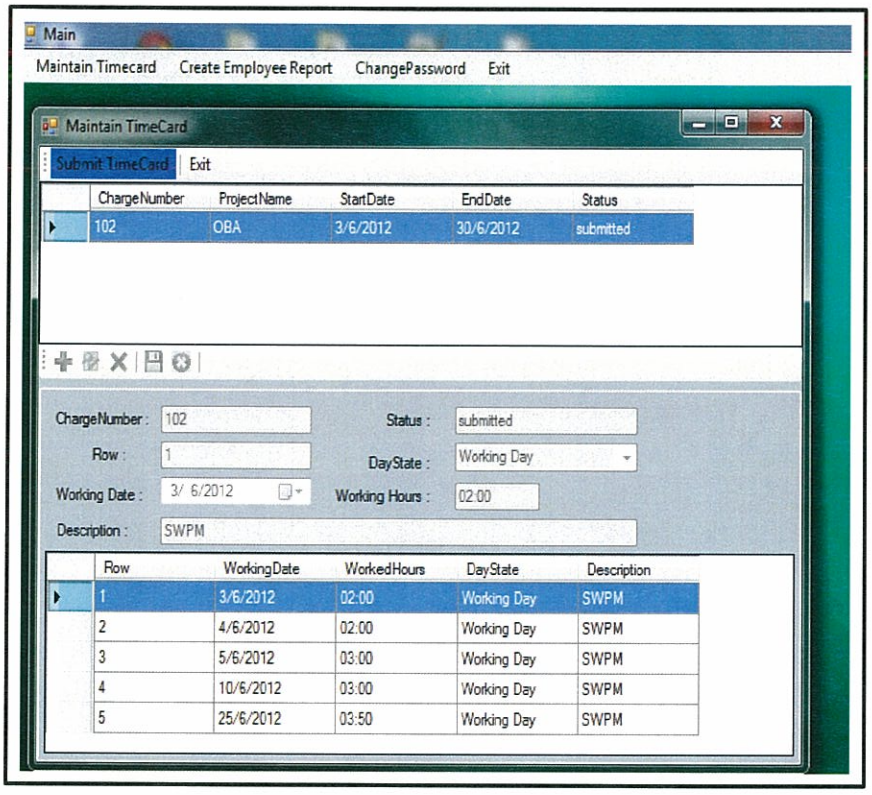

Figure 19 Maintain Timecard Page

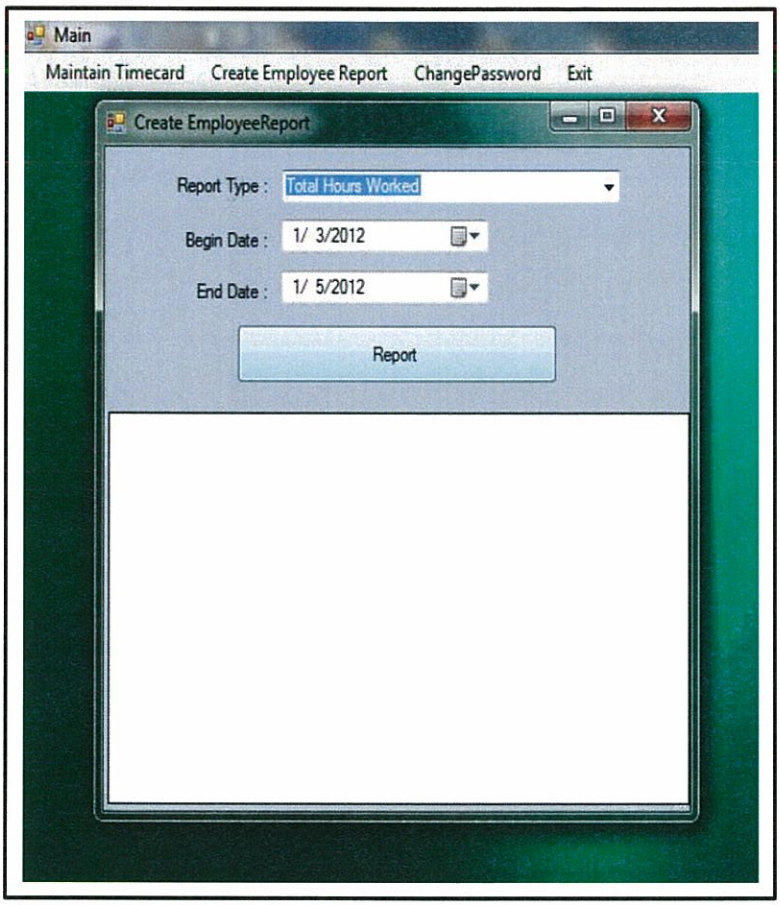

Figure 20 Create Employee Report Page

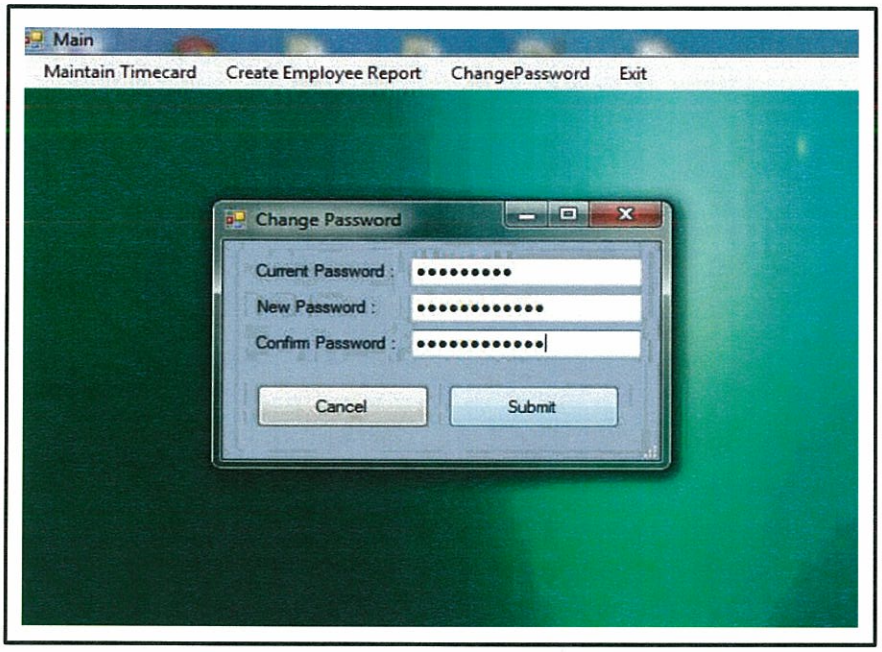

Figure 21 Change Password Page

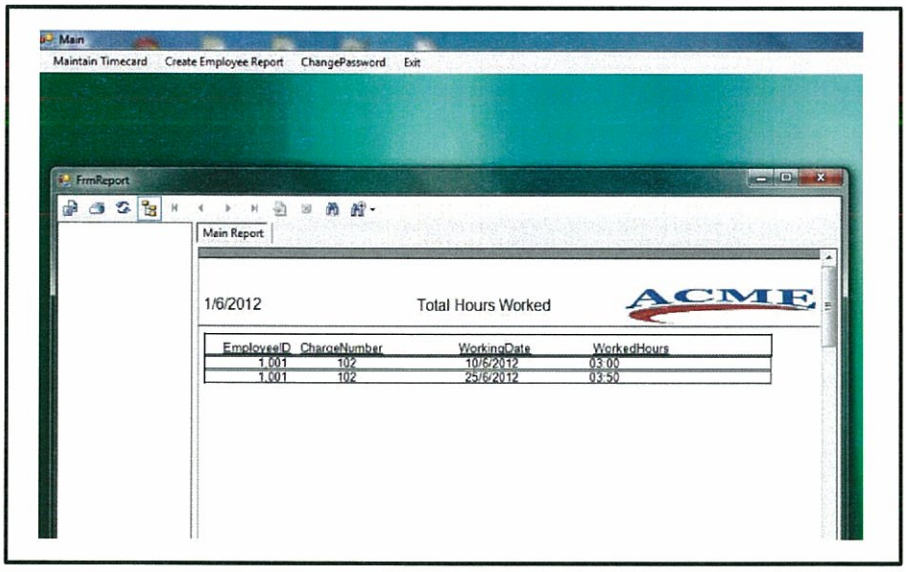

Figure 22 Total Hours Worked by Employee

APPENDIX C LIST OF PUBLICATIONS

Indexed Journal

1. **Shah, J.**, Kama, N., & Bakar, N. A. A. (2018). A Novel Effort Estimation Model For Software Requirement Changes During Software Development Phase(**indexed in era**)

2. **Jalal Shah,** N. K., Saiful Adli Ismail. (2018). An Empirical Study with Function Point Analysis for Software Development Phase Method. Paper presented at the 2018 7th International Conference on Software and Information Engineering (ICSIE 2018), Cairo, Egypt. **(Indexed in Scopus).**

3. **Shah, J.**, & Kama, N. (2018a). Extending Function Point Analysis Effort Estimation Method for Software Development Phase. Paper presented at the Proceedings of the 2018 7th International Conference on Software and Computer Applications, Kuantan, Malaysia **(Indexed in Scopus).**

4. **Shah, J.**, & Kama, N. (2018b). Issues of Using Function Point Analysis Method for Requirement Changes During Software Development Phase. Paper presented at the Asia Pacific Requirements Engeneering Conference, Melaka Malaysia. https://link-springer-com.ezproxy.utm.my/chapter/10.1007/978-981-10-7796-8_12#citeas **(Indexed in Scopus)**

Non Indexed Conference

1. **Jalal Shah***a, N. K. b., Amelia Zahari. (2017). An Empirical Study With Function Point Analysis For Requirement Changes During Software Development Phase. Paper Presented at the Asia International Multidisciplinary Conference 2017, Johor Bharu.