

P-3557



**OPTIMIZED RESOURCE ALLOCATION FOR
SOFTWARE RELEASE PLANNING**



PROJECT REPORT

Submitted by

T.ANITHA

Register No: 0920108001

In partial fulfillment for the award of the degree of

Of

MASTER OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University of Technology, Coimbatore)

COIMBATORE – 641 049

APRIL 2011

BONAFIDE CERTIFICATE

Certified that this project report titled “**OPTIMIZED RESOURCE ALLOCATION FOR SOFTWARE RELEASE PLANNING**” is the bonafide work of **Ms.T.Anitha (0920108001)** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report of dissertation on the basis of which a degree or ward was conferred on an earlier occasion on this or any other candidate.


GUIDE**Mr.G.S.Nandakumar M.E.,**
HEAD OF THE DEPARTMENT**Mrs.P.DEVAKI M.E., (Ph.D.,)**

The candidate with **University Register No. 0920108001** was examined by us in Project Viva-Voce examination held on 25.4.2011


INTERNAL EXAMINER
EXTERNAL EXAMINER

HINDUSTHAN COLLEGE OF ENGINEERING & TECHNOLOGY

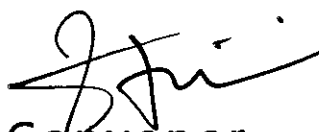
COIMBATORE - 32

DEPARTMENT OF INFORMATION TECHNOLOGY CERTIFICATE

This is to certify that
Mr/Ms.....*T. ANITHA*.....
...*KVMARAVURU*...*COLLEGE OF TECHNOLOGY*.....
has participated and presented a paper
entitled...*Optimized Resource Allocation for*..
Software Release Planning.....
in the National Conference on Recent
trends in Communication, Networking and
Computing, held at Hindusthan college of
Engineering and Technology, Coimbatore,
on March 10th 2011.



Organising
Secretary



Convener

Principal

ACKNOWLEDGEMENT

If the words are considered as symbols of approval and tokens of acknowledge, then the words play the heralding role of expressing our gratitude.

I express my profound gratitude to our Chairman **Padmabhusan Arutselvar Dr.N.Mahalingam, B.Sc., F.I.E.**, for giving this opportunity to pursue this course.

I would like to thank **Dr.S.Ramachandran, Ph.D.**, *Principal* for providing the necessary facilities to complete my thesis.

I express my sincere thanks to **Dr.S.Thangasamy Ph.D.**, *Dean*, Department of Computer Science and Engineering, for being my greatest of inspiration and for embedding the quest for innovative ideas.

I also thank **Mrs.P.Devaki M.E., (Ph.D.)** *HOD*, Department of Computer Science and Engineering, for her support and timely motivation.

I thank all project committee members for their comments and advice during the reviews. Special thanks to **Mrs.V.Vanitha M.E., (Ph.D)**, and **Mr.V.Subramani M.Tech.**, *Associate Professor*, Department of Computer Science and Engineering, for arranging brain storming project review sessions.

I register my sincere thanks to my Guide **Mr. G.S.Nandakumar M.E.**, *Assistant Professor*, Department of Computer Science and Engineering for his valuable suggestions and support right from the beginning to perform my project work extremely well.

I would like to convey my honest thanks to all **Teaching** staff members and **Non Teaching** staffs of the department for their support. I would like to thank all my classmates who gave me a proper light moments and study breaks apart from extending some technical support whenever I needed them most.

I dedicate this project work to my **parents** for no reasons but feeling from bottom of my heart, without their love this work wouldn't be possible.

3	System Analysis	
	3.1 Drawbacks of Existing System	21
	3.2 Proposed System	21
4	System Specification	
	4.1 Hardware Requirements	23
	4.2 Software Requirements	23
	4.3 Software Description	24
5	Project Description	
	5.1 Problem Definition	29
	5.2 Overview of the Project	29
	5.3 Module Description	30
6	Results and Discussion	35
7	Conclusion and Future Enhancement	36
8	Appendices	
	8.1 Source Code	37
	8.2 Screen Shots	56
9	References	65

ABSTRACT

Application delivery plan is one of the most important in every organization. Source allocation for their delivery plan is help to achieve their goal. HR team is responsible for plan, allocation and meeting arrangement. They should arrange client meeting for gather requirements and through it to development team for analysis. The development team will be analysis their requirements and design for the development based on the requirements and pass it to HR team. In the analysis phase the development team have mention their required time period to develop the application.

In this case release planning for incremental software development assigns features to releases such that technical, resource, risk, and budget constraints are met. A feature can be offered as part of a release only if all of its necessary tasks are done before the given release date. It considers a pool of human resources with different degrees of productivity to perform different types of tasks.

This method is to allocate these resources to the tasks of implementing the features such that the value gained from the released features is maximized. Planning of software releases and allocation of resources cannot be handled in isolation. To address the inherent difficulty of this process, a two-phase optimization approach called OPTIMIZERASORP that combines the strength of two solution methods.

ஆய்வுச் சுருக்கம்

ஒரு பயன்பாட்டினை கொடுக்கும் திட்டமானது ஒவ்வொரு அமைப்பின் முக்கிய செயலாகும். இந்த இலக்கினை அடைய ஆதார ஒதுக்கீடு முறை பயன்படுகிறது. ஒதுக்கீடு, கலந்தாய்வு ஏற்பாட்டிற்கு மனித ஆதார அணி காரணியாக உள்ளது. இந்த அணியினர் சேவையுருவங்களை ஆயத்தப்படுத்தி, கலந்தாய்விக்கான தேவைகளை பெற்றுக் கொள்கின்றனர். இதன் வழியாக வளர்ச்சிக் குழுப் பகுப்பாய்வு செய்கிறது. இந்த வளர்ச்சிக் குழு தேவைகளை பகுப்பாய்வு செய்யும் மற்றும் வளர்ச்சிக் குழுவின் வடிவமைப்பானது தேவைகளைச் சார்ந்தது. இந்த வடிவமைப்பு மனித ஆதாரக் குழுவிற்கு அனுப்பப்படுகிறது. பகுப்பாய்வுக் கட்டத்தில், வளர்ச்சிக் குழு ஒரு பயன்பாட்டினை மேம்படுத்துவதற்குத் தேவையான காலக்கட்டம் குறிப்பிடப்படும்.

LIST OF FIGURES

FIGURE NO.	CAPTION	PAGE NO.
1.1	Software Engineering Layers	1
1.2	The Software Process	3
1.2.1	The Phase of Problem Solving Loop	4
1.3.1	Determines the Software Quality	6
1.3.2.3	Software Metrics Collection Process	8
1.7	Software Release Planning	10
1.9	Release Planning Iteration	11
2.1.3	Release Planning with a Changing Environment	19
3.1	Project Architecture	22

LIST OF ABBREVIATIONS

ILP	Integer Linear Programming
SRP	Software Release Planning
UI	User Interface
RASORP	Resource Allocation for Software Release Planning
PMS	Performance Management System
KPAs	Key Process Areas

CHAPTER 1

1 INTRODUCTION

1.1 SOFTWARE ENGINEERING:

Software engineering is an engineering discipline that is concerned with all aspects of software production. Software engineers should adopt a systematic and organized approach to their work and use appropriate tools and techniques depending on the problem to be solved, the development constraints and the resources available.

Software engineering *process* is the glue that holds the technology layers together and enables rational and timely development of computer software. Process defines a framework for a set of *key process areas* (KPAs). The key process areas form the basis for management control of software projects and establish the context in which technical methods are applied, work products (models, documents, data, reports, forms, etc.) are produced, milestones are established, quality is ensured, and change is properly managed.



Figure.1.1: Software Engineering Layers

Software engineering *methods* provide the technical how-to's for building software. Methods encompass a broad array of tasks that include requirements analysis, design, program construction, testing, and support. Software engineering methods rely on a set of basic principles that govern each area of the technology and include modeling activities and other descriptive techniques.

Software engineering *tools* provide automated or semi-automated support for the process and the methods. When tools are integrated so that information created by one tool can be used by another, a system for the support of software development, called *computer-aided software engineering*, is established. CASE combines software, hardware, and a software engineering database. Software engineering encompasses a process, management and technical methods, and tools.

1.1.2 SOFTWARE ENGINEERING VS SYSTEM ENGINEERING:

- System engineering is concerned with all aspects of computer-based systems development including hardware, software and process engineering.
- Software engineering is part of this process concerned with developing the software infrastructure, control, applications and databases in the system.

1.2 SOFTWARE PROCESS:

A software process is a *common process framework* is established by defining a small number of framework activities that are applicable to all software projects, regardless of their size or complexity. A number of *task sets*—each a collection of software engineering work tasks, project milestones, work products, and quality assurance points—enable the framework activities to be adapted to the characteristics of the software project and the requirements of the project team. Finally, umbrella activities—such as software quality assurance, software configuration management, and measurement—overlay the process model. Umbrella activities are independent of any one framework activity and occur throughout the process.

A set of activities whose goal is considered as the development or evolution of software. Generic activities in all software processes are:

- **Specification** - what the system should do and its development constraints
- **Development** - production of the software system
- **Validation** - checking that the software is what the customer wants
- **Evolution** - changing the software in response to changing demands.

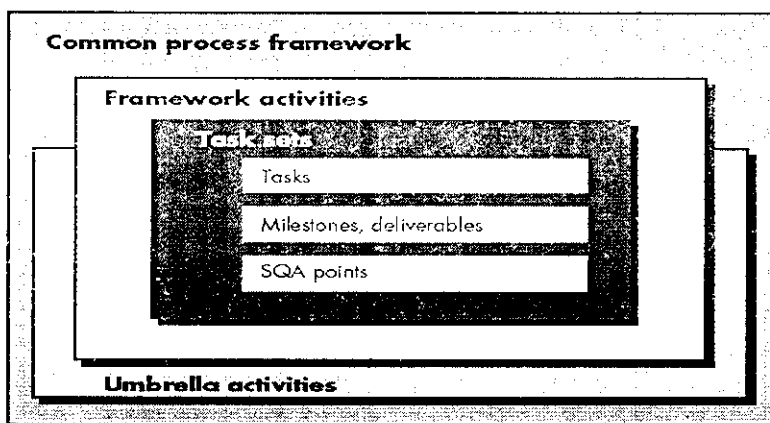


Figure.1.2: The Software Process

1.2.1 SOFTWARE PROCESS MODELS:

A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used, and the controls and deliverables that are required. In an intriguing paper on the nature of the software process, all software development can be characterized as a problem solving loop in which four distinct stages are encountered: status quo, problem definition, technical development, and solution integration.

This problem solving loop applies to software engineering work at many different levels of resolution. It can be used at the macro level when the entire application is considered, at a mid-level when program components are being engineered, and even at the line of code level.

A variety of different process models for software engineering are discussed. Each represents an attempt to bring order to an inherently chaotic activity. It is important to remember that each of the models has been characterized in a way that (ideally) assists in the control and coordination of a real software project.

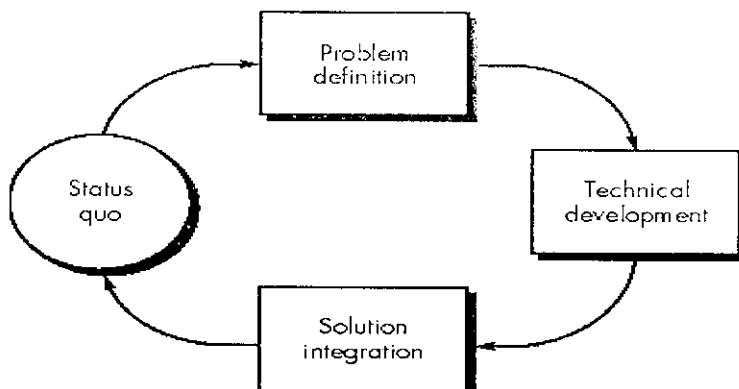


Figure.1.2.1The Phases of a Problem Solving Loop

1.3 SOFTWARE PROJECT METRICS:

Software metrics refers to a broad range of measurements for computer software. Measurement can be applied to the software process with the intent of improving it on a continuous basis. Measurement can be used throughout a software project to assist in estimation, quality control, productivity assessment, and project control.

The terms *measure*, *measurement*, and *metrics* are often used interchangeably, it is important to note the suitable differences between them.

Measure: Measure can be used either as a noun or a verb, definitions of the term can become confusing. Within the software engineering context, a measure provides a quantitative indication of the extent, amount, dimension, capacity, or size of some attribute of a product or process. Measurement is the act of determining a measure.

Metric: A quantitative measure of the degree to which a system, component, or process possesses a given attribute.

Indicator: An *indicator* is a metric or combination of metrics that provide insight into the software process, a software project. An indicator provides insight that enables the project manager or software engineers to adjust the process, the project, or the process to make things better.

1.3.1 SOFTWARE PROCESS IMPROVEMENT:

The process, develop a set of meaningful metrics based on these attributes, and then use the metrics to provide indicators that will lead to a strategy for improvement. The process sits at the center of a triangle connecting three factors that have a profound influence on software quality and organizational performance.

The technology (i.e., the software engineering methods) that populates the process also has an impact. In addition, the process triangle exists within a circle of environmental conditions those include the development environment (e.g., CASE tools), business conditions (e.g., deadlines, business rules), and customer characteristics (e.g., ease of communication).

To derive a set of metrics based on the outcomes that can be derived from the process. Outcomes include measures of errors uncovered before release of the software, defects delivered to and reported by end-users, work products delivered (productivity), human effort expended, calendar time expended, schedule conformance, and other measures. It also derives process metrics by measuring the characteristics of specific software engineering tasks.

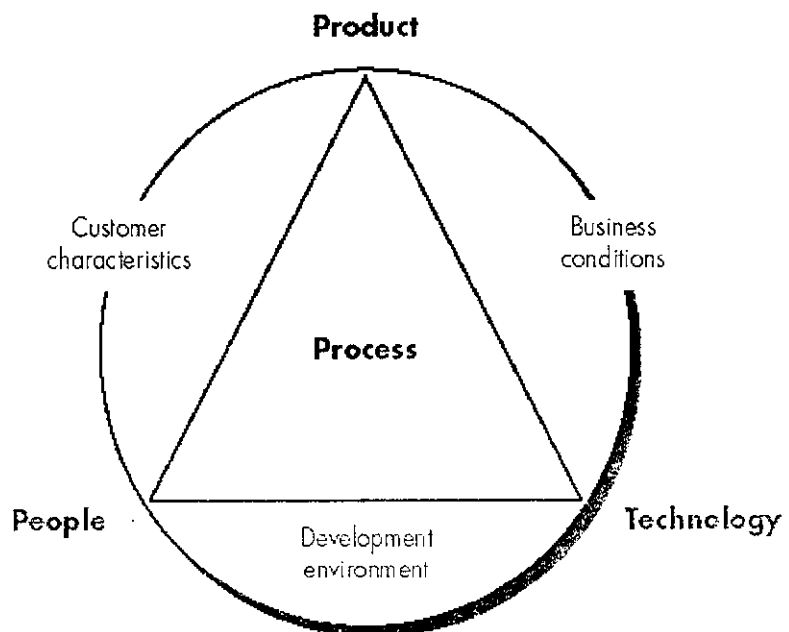


Figure.1.3.1: Determines the software quality

1.3.2 INTEGRATING METRICS WITHIN THE SOFTWARE PROCESS:

The majority of software developers still do not measure, and sadly, most have little desire to begin.

1.3.2.1 ARGUMENTS FOR SOFTWARE METRICS:

It is important to measure the process of software engineering and the product (software) it produces. By requesting and evaluating productivity and quality measures, senior management can establish meaningful goals for improvement of the software engineering process

1.3.2.2 ESTABLISHING A BASELINE:

The process, project, and product (technical) levels. Yet the information that is collected need not be fundamentally different. The same metrics can serve many masters. The metrics baseline consists of data collected from past software development projects and can be as simple as the table

To be an effective aid in process improvement and/or cost and effort estimation, baseline data must have the following attributes: (1) data must be reasonably accurate —“guesstimates” about past projects are to be avoided; (2) data should be collected for as many projects as possible; (3) measures must be consistent, for example, a line of code must be interpreted consistently across all projects for which data are collected; (4) applications should be similar to work that is to be estimated—it makes little sense to use a baseline for batch information systems work to estimate a real-time, embedded application.

1.3.2.3 METRICS COLLECTION, COMPUTATION AND EVALUATION:

The data collection requires a historical investigation of past projects to reconstruct required data. Once measures have been collected (unquestionably the most difficult step), metrics computation is possible. Depending on the breadth of measures collected, metrics can span a broad range of LOC or FP metrics as well as other quality- and project-oriented metrics. Finally, metrics must be evaluated and applied during estimation, technical work, project control, and process improvement. Metrics evaluation focuses on the underlying reasons for the results obtained and produces a set of indicators that guide the project or process.

Baseline metrics data should be collected from a large, representative sampling of past software projects.

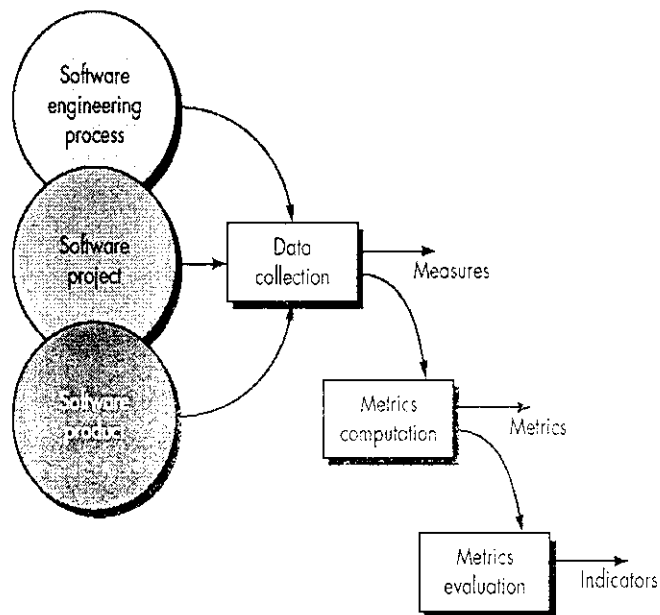


Figure.1.3.2.3: Software Metrics Collection Process

1.4 SOFTWARE ENGINEERING METHODS:

Structured approaches to software development which include system models, notations, rules, design advice and process guidance.

- **Model descriptions**
 - Descriptions of graphical models which should be produced;
- **Rules**
 - Constraints applied to system models;
- **Recommendations**
 - Advice on good design practice;
- **Process guidance**
 - What activities to follow.

1.5 ATTRIBUTES OF GOOD SOFTWARE:

The software should deliver the required functionality and performance to the user and should be maintainable, dependable and acceptable.

- **Maintainability**
 - Software must evolve to meet changing needs;
- **Dependability**
 - Software must be trustworthy;
- **Efficiency**
 - Software should not make wasteful use of system resources;
- **Acceptability**
 - Software must be accepted by the users for which it was designed.

This means it must be understandable, usable and compatible with other systems.

1.6 CHALLENGES OF SOFTWARE ENGINEERING:

- **Heterogeneity**
 - Developing techniques for building software that can cope with heterogeneous platforms and execution environments;
- **Delivery**
 - Developing techniques that lead to faster delivery of software;
- **Trust**
 - Developing techniques that demonstrate that software can be trusted by its users.

1.7 SOFTWARE RELEASE PLAN:

A release plan is an assignment of features to releases. It is formally described by a matrix x of decision variables $x(n, k)$ with

$x(n, k) = 1$ if and only if feature $f(n)$ is offered at release k ($k = 1 \dots K$) (and $x(n, k) = 0$ otherwise).

A release plan is an evolving flowchart that describes which features will be delivered in upcoming releases.

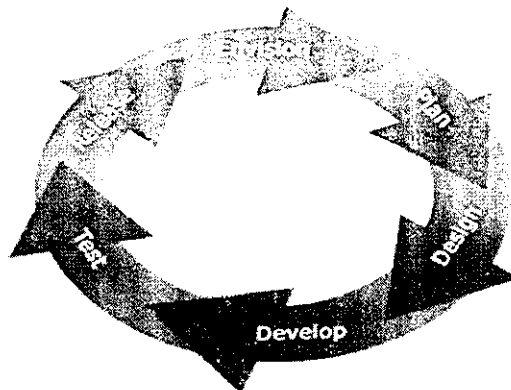


Figure.1.7: Software Planning Steps

1.8 RESOURCE ALLOCATION:

The process of allocating resources among the various projects or business units.

Necessity:

- Resource allocation is mainly for software reliability.
- To maintain proper delivery for software products



1.9 RELEASE PLANNING:

A release planning meeting is used to create a release plan, which lays out the overall project. The release plan is then used to create iteration plans for each individual iteration.

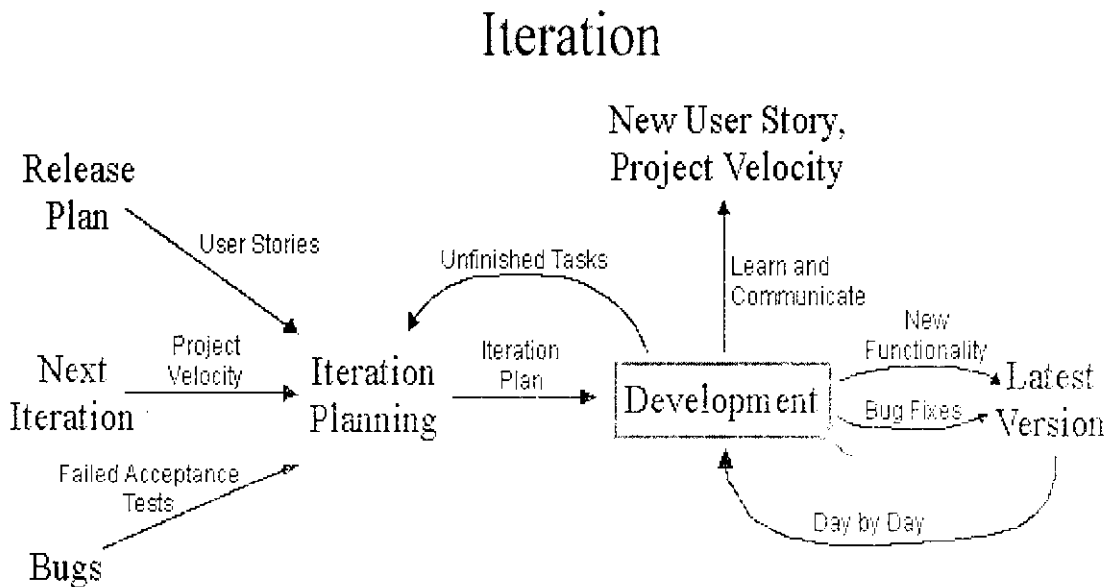


Figure.1.9: Release Planning Iterations

CHAPTER 2

LITERATURE SURVEY

2.1 INTELLIGENT SUPPORT FOR RELEASE PLANNING:

2.1.1 SOFTWARE RELEASE PLANNING FOR EVOLVING SYSTEMS

D. Greer* University of Calgary to achieve higher flexibility and to better satisfy actual customer requirements, there is an increasing tendency to develop and deliver software in an incremental fashion.

Problem Overview:

Release planning can be approached from different perspectives. We have identified two fundamental approaches which we will call: (1) the art and (2) the science-based planning of releases. The art of release planning involves mainly relying on human intuition, communication, and capabilities to negotiate between conflicting objectives and constraints. The science of release planning involves emphasizing formalization of the problem and applying computational algorithms to generate best solutions. Both art and science are important to achieve meaningful release planning results.

Stakeholders:

Stakeholders are extremely important for performing realistic and customer-oriented release planning. We assume a set of stakeholders $S = \{S(1), \dots, S(q)\}$. Each stakeholder $S(p)$ can be assigned a relative importance $\lambda(p)$, $\{1, \dots, 9\}$. The underlying meaning is

- $\lambda(p) = 1$ means extremely low importance,
- $\lambda(p) = 3$ means low importance,
- $\lambda(p) = 5$ means average importance,
- $\lambda(p) = 7$ means high importance,

- $\lambda(p) = 9$ means extremely high importance.

The interpretation of the even values not shown is that they are refinements of the values above and below them. We have chosen a nine-point ordinal scale for expressing the extent of importance as this gives sufficient detail to differentiate between the importance of stakeholders. However, the whole approach is applicable in the same way for other scales.

Prioritization of features by stakeholders:

To select and schedule features, there must be an agreed upon statement of priorities for features. In our proposed model, prioritization by each stakeholder $S(p)$ is done with respect to three different types of criteria, each defined on a nine point ordinal scale:

- **Value** (denoted value $(i,p) \in \{1, \dots, 9\}$) assigned by stakeholder p to feature $f(i)$. The value-based priority measure is used to express the expected value that the implementation of this feature will add to the stakeholder.
- **Satisfaction** (denoted $\text{sat}(i,p) \in \{1, \dots, 9\}$) assigned by stakeholder p to feature $f(i)$. This priority measure is used to express the extent of satisfaction with the situation that $f(i)$ is assigned to the next release. A measure of satisfaction is different from that of value because it expresses the urgency with which this stakeholder desires the feature. An increasing order of satisfaction corresponds to an increasing satisfaction-based priority.
- **Dissatisfaction** (denoted $\text{dissat}(i,p) \in \{1, \dots, 9\}$) assigned by stakeholder p to feature $f(i)$. This priority measure is used to express the extent of dissatisfaction with the situation that $f(i)$ is not assigned to the next release. An increasing order of dissatisfaction corresponds to an increasing dissatisfaction-based priority.

2.1.2 Software product release planning through optimization and what-if analysis

Problem Overview:

The tool is based on integer linear programming and assumes that an optimal set of requirements is the set with maximal projected revenue against available resources. The input for the optimization is twofold. The first type of input data concerns the list of candidate requirements, estimated revenues, and resources needed. Secondly, managerial steering mechanisms enable what-if analysis in the optimization environment. Experiments based on real-life data made a sound case for the applicability of our approach.

Formalization of release planning:

In this section we formulate requirements management in product software companies as a combinatorial optimization problem. In such a problem we have to find the best from a finite but very large number of solutions. These type of problems occur in many areas within production planning, logistics, transportation, personnel planning and telecommunication. The most famous example is the traveling salesman problem. An example related to requirements management is the problem of a person or company being assigned a number of tasks each with a specific duration and deadline. It receives more tasks than it can handle in time. However, the customer only pays if the work is completed in time. This means that a selection of tasks to be executed has to be made such that the selected tasks are completed on time and the revenue is maximized.

Integer linear programming is a well-known technique for solving combinatorial optimization problems. In general, integer linear programming problems are NP-hard. However, using advanced algorithms and specialized software, an (near-) optimal solution of the integer linear program can often be found within a reasonable amount of time.

We will discuss different variants of the problem of release definition, i.e., selecting requirements for the next release of the software product. We are given a set of n requirements $\{R_1, R_2, \dots, R_n\}$. Suppose that for each requirement R_j we can estimate its revenue v_j . The implementation of each requirement needs a

given amount of resources in the form of labor hours from the development teams. We assume that the date of the next release is given; hence we have to deal with a fixed planning period. Clearly, the available amount of resources is limited. Therefore, we have to make a selection of the requirements to be included in the next release, preferably, such that the revenue is maximal. This can be viewed as the following optimization problem: find the subset of requirements for the next release such that the revenue is maximal and the available capacity is not exceeded.

We present different models taking into account different aspects and managerial steering mechanisms within a product software company. We consider:

- One pool of developers (i.e. no different development teams)
- Different teams without team transfers, each with its own capacity constraint
- Different teams with team transfers allowed
- Hiring external team capacity
- Extension of the development project deadline
- Requirement dependency (functional, revenue and cost related).

Development by one pool of developers:

In the first variant we only deal with the total amount of man days available in the company. We denote our planning period by T and define $d(T)$ as the number of working days in the planning period. Moreover, let Q be the number of persons working in the development teams of the company. The available capacity then equals $d(T)Q$ man days.

Moreover, we have an estimate a_j of the amount of man days needed for the implementation of requirement R_j . Such estimates could come from project managers (top-down) as well as developers (bottom-up). We model the requirements selection problem in terms of binary variables x_j ($j = 1, \dots, n$),

where

$x_j =$ (1 if requirement R_j is selected;
0 otherwise.)

Like with determining the revenue values, also the determination of values for costs can be difficult to calculate. At the software vendor two of the authors worked for in the past, the process was as follows. The product manager inserted the requirement in the requirements database, and determined the software modules for which the requirement had consequences (which modules needed to be updated for this requirement). The product architect(s) determined the workload per module using input from product/module consultants and software engineers. In many cases, to make the resource need more clear, conceptual solutions were written, detailing the requirement in a business context, and showing relations to other modules. In general top down resource calculation and bottom up resource calculation may provide a useful estimation for the costs in many software companies.

Development teams:

In the previous model, there was an only one pool software developer. There are different development teams within a software company, each having their own specialization. So the above model may be too optimistic, since it did not take the individual team capacities into account. Let m be the number of teams and suppose team G_i ($i = 1, \dots, m$) consists of Q_i persons. We assume that the implementation of requirement R_j needs a given amount a_{ij} of man days from team G_i ($i = 1, \dots, m$). Now the team capacities can be included by replacing constraint

$$\sum_{j=1}^R a_{ij} x_j \leq d(T) Q_i \quad \text{for } i = 1, \dots, m.$$

Note that for $m = 1$, this coincides with the model for one pool of developers. This problem is known as the binary m -dimensional knapsack problem. Clearly, the model can be adapted to the situation with different amounts of man hours per person.

Solving integer linear programming problems:

All the problems formulated in this paper are NP-hard. In general, integer linear programming problems are NP-hard. This implies that it is very unlikely that there exists an algorithm that is guaranteed to find the optimal solution in a time that is polynomial in the input size. Finding the optimal solution requires an amount of time which in the worst case grows exponentially with the problem size.

The first step to solve an ILP is always to solve the LP-relaxation. If the solution of the LP-relaxation is integral, we are done. Otherwise, we start with a branch-and-bound tree. The ILP is split into two or more sub-problems corresponding to two or more nodes of a tree, for example by fixing a variable x_j to 0 in one node, i.e. omit requirement R_j in the release, and to 1 in the other node, i.e. select requirement R_j in the new release. (This is the branching part).

The algorithm starts evaluating one of the nodes. First the LP-relaxation in the node is solved. If the solution is integral, the node is finished and the best-known integral solution is updated, if necessary. If the LP-relaxation is infeasible, clearly the node is finished as well. If the values of the LP-relaxation are lower than the best known integral solution, the node can be skipped from further consideration since we have no hope of finding the optimal solution there. (This is the bounding part). Otherwise, new nodes are generated by branching, i.e. by selecting or omitting another requirement. Since we maintain the best known integral solution and we have an upper bound from the LP-relaxation, we have a solution with a quality guarantee from the moment at which an integral solution is found. This allows us to stop if the solution is guaranteed to be within a certain margin from the optimum. This can be beneficial, because it occurs quite often that the optimal solution is found quickly and it takes a lot of time to prove that the solution is indeed optimal.

2.2 Release Planning: An Evolutionary and Iterative Approach

Problem Overview:

Incremental software development addresses the time-to-delivery of software products. Instead of delivering a monolithic system after a long development time, smaller releases are implemented sequentially. If applicable, this approach has many advantages over the traditional waterfall approach. Firstly, requirements can be prioritized so that the most important ones are delivered first and benefits of the new system gained earlier. Consequently, less important requirements are left until later and so if the schedule or budget is not sufficient the least important requirements are the ones more likely to be omitted. Secondly, it means that customers receive part of the system early on and so are more likely to support the system and to provide feedback on it. Thirdly, being smaller, the schedule/cost for each delivery stage is easier to estimate. Fourthly, user feedback can be obtained at each stage and plans adjusted accordingly. Fifthly, perhaps most importantly, an incremental approach allows for a much better reaction to changes or additions to requirements

Incremental Software Delivery:

In the incremental software process model, requirements are gathered in the initial stages and, taking technical dependencies and user priorities into account and the effort required for each requirement, the system is divided into increments. These increments are then successively delivered to customers. In the simplest case requirements are fully known and complete at the beginning of the project of incremental software. It is more likely however that, even during the initial increments, the product will evolve so that some requirements will change and new ones will be introduced. In addition to that, priorities and constraints might have changed as well.

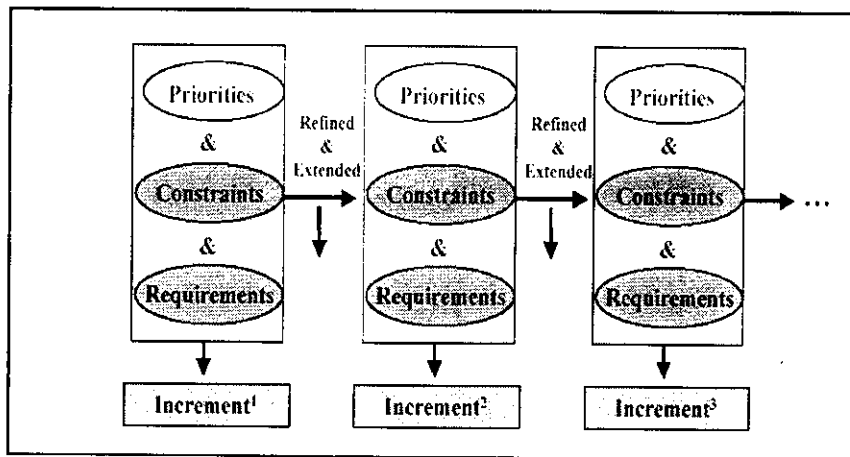


Figure.2.1.3 Release Planning in a Changing Environment with release

Stakeholder Evaluation and Effort Estimates:

One of the challenges to the software engineering research community is to involve stakeholders in the requirements engineering process. In any project several stakeholders may be identified. These are likely to have differing priorities. We consider two kinds of evaluation. Both evaluations are on ordinal scale. One input to the process should be a ranking or scoring of requirements according to the perceived value (expected relative impact on business value of the final product). The other one is prioritization according to the degree of urgency (time-criticality) of each requirement to each stakeholder.

It is assumed that a software system is initially specified by a set R_1 of requirements, i.e., $R_1 = \{r_1, r_2, \dots, r_m\}$. At this stage ($k=1$), we wish to allocate these requirements to the next and future releases. In a later phase $k (>1)$, an extended and/or modified set of requirements R_k will be considered as a starting point to plan for increment k (abbreviated by Inc_k). The requirements are competing with each other (to become implemented).

Evolution of Increments:

As result of the planning process, different increments will be composed out of the given set of requirements. These increments are planned up-front but the possibility of re-planning after any increment is allowed. This re-planning may involve changing some requirements, priorities and constraints and/or introducing new ones. It necessitates a reassignment of requirements (not already implemented in former releases) to increments. Throughout the paper, we assume that the number of releases is not fixed upfront. The complete modeling and solution approach remains valid with only minor modifications for the case of fixed number of releases.

CHAPTER 3

SYSTEM ANALYSIS

3.1 DRAWBACKS OF THE EXISTING SYSTEMS

A major problem faced by companies developing or maintaining large and complex systems is determining which elements of a typically large set of candidate features should be assigned to which releases of the software. In addition, there is the question of how to assign resources accordingly. The solution of this complex problem is of pivotal importance for project success. Without good release planning, critical features are not provided at the right time. This might result in dissatisfied customers, time and budget overruns, and decreased competitiveness in the marketplace.

Disadvantages are

- There is no single application for all process.
- Developers didn't interact with client until their project development

3.2 PROPOSED SYSTEM

- We enlarge the scope of release planning by examining details of the resource allocation necessary to actually develop the features. For that, we assume each feature is decomposed into a sequence of tasks such as design, implementation, and testing.
- These development tasks can be defined to an even more fine-grained level. In addition, managerial support and/or other tasks can be considered here as well. For these tasks, a pool of human resources is available.

- This will meet the following achievements. All the three phases will interact and get share their updated process with them. Schedule process will help to develop their process as per commitment.
- Source allocation and pre planned activities will reduce the workload of development process. Client gets their application as per schedule date.

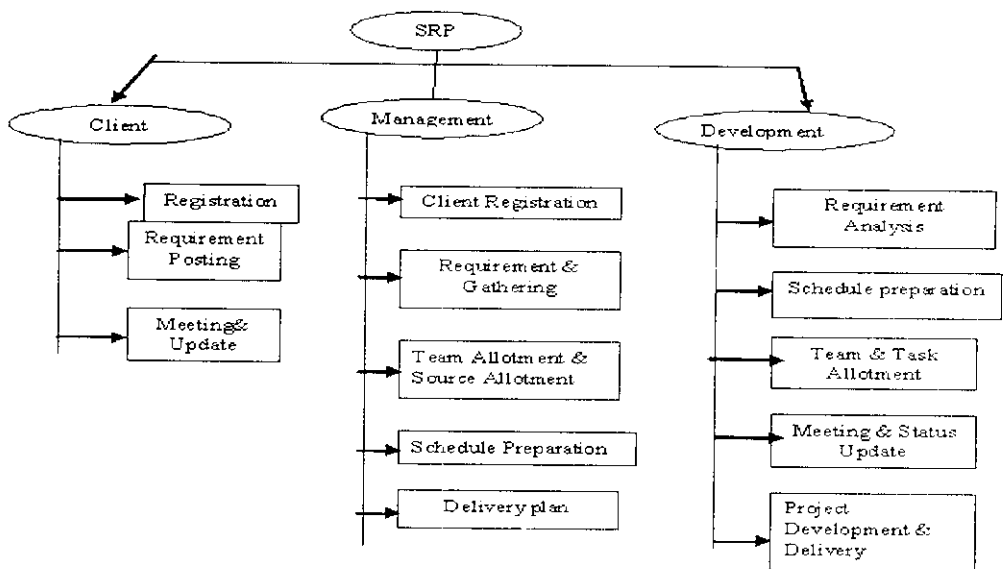


Figure.3.1 Proposed System Architecture

Advantages of the system:

- All the three phases will interact and get share their updated process with them.
- Schedule process will help to develop their process as per commitment.
- Source allocation and pre planned activities will reduce the workload of development process.

CHAPTER 4

SYSTEM SPECIFICATION

4.1 HARDWARE REQUIREMENTS

The hardware used for the development of the project is:

PROCESSOR	:	PENTIUM III 766 MHz
RAM	:	128 MD SD RAM
MONITOR	:	15" COLOR
HARD DISK	:	20 GB
KEYBOARD	:	STANDARD 102 KEYS
MOUSE	:	3 BUTTONS

4.2 SOFTWARE REQUIREMENTS

The software used for the development of the project is:

FRAMEWORK	:	Visual Studio .Net 2005
FRONT END	:	ASP.Net 2.0
DATABASE	:	SQL Server 2000
LANGUAGE	:	C#.Net
OPERATING SYSTEM:		Windows XP

4.3 SOFTWARE DESCRIPTION

FEATURES OF ASP.NET

ASP.NET is the next version of Active Server Pages (ASP); it is a unified Web development platform that provides the services necessary for developers to build enterprise-class Web applications. While ASP.NET is largely syntax compatible, it also provides a new programming model and infrastructure for more secure, scalable, and stable applications.

ASP.NET is a compiled, .NET-based environment, we can author applications in any .NET compatible language, including Visual Basic .NET, C#, and JScript .NET. Additionally, the entire .NET Framework is available to any ASP.NET application. Developers can easily access the benefits of these technologies, which include the managed common language runtime environment (CLR), type safety, inheritance, and so on.

ASP.NET has been designed to work seamlessly with WYSIWYG HTML editors and other programming tools, including Microsoft Visual Studio .NET. Not only does this make Web development easier, but it also provides all the benefits that these tools have to offer, including a GUI that developers can use to drop server controls onto a Web page and fully integrated debugging support.

Developers can choose from the following two features when creating an ASP.NET application. Web Forms and Web services, or combine these in any way they see fit. Each is supported by the same infrastructure that allows you to use authentication schemes, cache frequently used data, or customize your application's configuration, to name only a few possibilities.

Web Forms allows us to build powerful forms-based Web pages. When building these pages, we can use ASP.NET server controls to create common UI

elements, and program them for common tasks. These controls allow we to rapidly build a Web Form out of reusable built-in or custom components, simplifying the code of a page.

THE .NET FRAMEWORK

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet.

OBJECTIVES OF .NET FRAMEWORK:

- To provide a consistent object-oriented programming environment whether object codes is stored and executed locally on Internet-distributed, or executed remotely.
- To provide a code-execution environment to minimizes software deployment and guarantees safe execution of code.
- Eliminates the performance problems.

There are different types of application, such as Windows-based applications and Web-based applications. To make communication on distributed environment to ensure that code be accessed by the .NET Framework can integrate with any other code.

FEATURES OF SQL SERVER 2000

The OLAP Services feature available in SQL Server version 7.0 is now called SQL Server 2000 Analysis Services. The term OLAP Services has been replaced with the term Analysis Services. Analysis Services also includes a new data mining component. The Repository component available in SQL Server version 7.0 is now called Microsoft SQL Server 2000 Meta Data Services. References to the component now use the term Meta Data Services. The term repository is used only in reference to the repository engine within Meta Data Services

SQL-SERVER database consist of six type of objects,

They are,

- **TABLE**
- **QUERY**
- **FORM**
- **REPORT**
- **MACRO**

TABLE:

A database is a collection of data about a specific topic.

VIEWS OF TABLE:

We can work with a table in two types,

- ✓ Design View
- ✓ Datasheet View

Design View

To build or modify the structure of a table we work in the table design view. We can specify what kind of data will be hold.

Datasheet View

To add, edit or analyses the data itself we work in tables datasheet view mode.

QUERY:

A query is a question that has to be asked the data. Access gathers data that answers the question from one or more table. The data that make up the answer is either dynaset (if you edit it) or a snapshot (it cannot be edited).Each time we run query, we get latest information in the dynaset. Access either displays the dynaset or snapshot for us to view or perform an action on it, such as deleting or updating.

FORMS:

A form is used to view and edit information in the database record by record .A form displays only the information we want to see in the way we want to see it. Forms use the familiar controls such as textboxes and checkboxes. This makes viewing and entering data easy.

Views of Form:

We can work with forms in several primarily there are two views,

They are,

- **Design View**
- **Form View**

Design View

To build or modify the structure of a form, we work in forms design view. We can add control to the form that are bound to fields in a table or query, includes textboxes, option buttons, graphs and pictures.

Form View

The form view which display the whole design of the form.

REPORT:

A report is used to view and print information from the database. The report can group records into many levels and compute totals and average by checking values from many records at once. Also the report is attractive and distinctive because we have control over the size and appearance of it.

MACRO:

A macro is a set of actions. Each action in macros does something. Such as opening a form or printing a report .We write macros to automate the common tasks the work easy and save the time.

CHAPTER 5

PROJECT DESCRIPTION

5.1 PROBLEM DEFINITION

A major problem faced by companies developing or maintaining large and complex systems is determining which elements of a typically large set of candidate features should be assigned to which releases of the software. In addition, there is the question of how to assign resources accordingly. The solution of this complex problem is of pivotal importance for project success. Without good release planning, critical features are not provided at the right time. This might result in dissatisfied customers, time and budget overruns, and decreased competitiveness in the marketplace. These tasks correspond to the fundamental technical, managerial, and support contributions necessary to develop software. Minimally, these tasks cover design, implementation, and testing. Assigning properly skilled developers to appropriate tasks is a complex process, typically including a large number of variables related to schedule times, availability, staffing, and skill profile.

5.2 PROJECT OVERVIEW

In this project, the managerial support and/or other tasks can be considered here as well. For these tasks, a pool of human resources is available. For the sake of simplicity, we will refer to this software release planning.

5.3 MODULES:

Our proposed system is implemented using three modules. They are follows.

- Management module
- Development module
- Client module

Management module:

Management team is responsible for all process. They should interact with client as well as development. Initially they could arrange a meeting for gathering requirements from clients. Based on the requirement they will set a strategy and set a commitment with the client to deliver the project.

Management team will provide the commitment report to the clients. In that Commitment report they will mention the payment details, queries and also assign the date for project demo.

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the requirements is to be carried out. This is to ensure that the system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

To develop the project they will allocate the required source for the development team. Management team will provide the commitment report to the

clients. In that Commitment report they will mention the payment details, queries and also assign the date for project demo. The commitment report can also be rescheduled if necessary. They also register the employee's profile allot task/project to corresponding team. They schedule the projects and the man power required to complete the task. The team leader will take care of assigning the sub tasks to the respective team members.

The implementation of this module considers different stages of the specification along with the every domain details of the project. The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

Development module:

The development team works hard to achieve their goal within the target date. With the help of management the development team will work and develop their application based on the client requirement. Every updated process will reported to management team and client every month. The development status would be updated day by day and month. Genetic Algorithm is used to search all the projects which are relevant to each department.

Genetic programming is the evolutionary algorithm this based methodology inspired to find computer programs that perform a user-defined task. It is a specialization of genetic algorithms where each individual is a computer program. It is a machine learning technique used to optimize a population of computer programs according to a fitness landscape determined by a program's ability to perform a given computational task.

The development team works hard to achieve their goal within the target date. With the help of management the development team will work and develop their application based on the client requirement. Every updated process will

reported to management team and client every month. The development status would be updated day by day and month.

The runtime also accelerates developer productivity. For example, programmers can write applications in their development language of choice, yet take full advantage of the runtime, and components written in other languages by other developers.

For example, .Net developers could search and view only the projects which are posted by the clients in .Net domain. Similarly Java Developers can view only the projects which are relevant to their domain. It is applicable to all domains. Thus it prevents the unauthorized people to view their committed project details and also the developer's current status. The team leaders can only view the status of their respective team members and allot the task the team leader is responsible to delegate the task to his team members. He will schedule the task to the employees whose current status is 'free'. By this process he can delegate the task equally to all the developers without any bias.

Each and every developer will give the details of their resume which makes the specification and it also performs search method with this technology. The genetic algorithm use here is mainly to search which all domains are available where they can modify their specification with system. Only with consideration of these developers profile we can allocate the project for certain domain.

The maintenance of this module is considered in such way. The source description is given to users and also with availability of domain. The objectives of this maintenance work are to make sure that the system gets into work all time without any bug. Provision must be for environmental changes which may affect the computer or software system. This is called the maintenance of the system. Nowadays there is the rapid change in the software world. Due to this rapid

change, the system should be capable of adapting these changes. In our project the process can be added without affecting other parts of the system.

Client module:

At first, new clients should register with the management. Then they can login using their username and password. Then he can register his profile. After profile registration, he could add projects in different domains and submit it to the management. The management analyzes it and sends them the commitment report. In turn client can view that Commitment Report for acceptance.

It is applicable to all domains. Thus it prevents the unauthorized people to view their committed project details and also the developer's current status. The team leaders can only view the status of their respective team members and allot the task the team leader is responsible to delegate the task to his team members.

They could interact with development team with the help of management team in meeting. They can view their project status also. They will be provided with the demo in each quarter and can also update their requirements. The client meeting may be conducted through VoIP, Tele conference, video conference or seminars as per the client's wish. Finally they can get the project as per committed date.

In the arrangement of meeting we are considering the method of Integer Linear Programming. The Purpose of this method is given a description of the technology

Linear programming can be applied to various fields of study. It is used most extensively in business and economics, but can also be utilized for some engineering problems. Industries that use linear programming models include

telecommunications and manufacturing. It has proved useful in modeling diverse types of problems in planning, scheduling, assignment, and design.

The client can also search for the domain they can schedule their project. The allotment of the description will be on their describing methods.

CHAPTER 6

RESULTS AND DISCUSSION

Implementation is the most crucial stage in achieving a successful system and giving the user's confidence that the new system is workable and effective. Implementation of the modified application to replace an existing one. This type of conversation is relatively easy to handle, provide there are no major changes in the system.

Each program is tested individually at the time of development using the data and has verified that this program linked together in the way specified in the programs specification, the computer system and its environment is tested to the satisfaction of the user. The system that has been developed is accepted and proved to be satisfactory for the user. And so the system is going to be implemented very soon. A simple operating procedure is included so that the user can understand the different functions clearly and quickly.

Initially as a first step the executable form of the application is to be created and loaded in the common server machine which is accessible to the entire user and the server is to be connected to a network. The final stage is to document the entire system which provides components and the operating procedures of the system.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

Thus a release planning for software is proposed which have been presented with different requirements. It also prepares procedures and forms related to Management Resources to ensure that Developers are made aware of company practices and processes. Here it also makes the consideration about the preparation of job description for each and every task which is included in the Performance Management System (PMS). In future appending the new modules with additional features were the websites may be very effective and useful for the satisfaction of customer.

APPENDICES

SOURCE CODE

Coding for Client registration

a) ProfileRegistration.aspx

```

using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Data.SqlClient;
using System.IO;

public partial class client_ClientProfile : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        lblCIUserName.Text = Session["CIUserName"].ToString();

        Calendar2.Visible = false;

        lblTime.Text = DateTime.Now.ToShortTimeString();

        txtCommitDate.Text = Convert.ToString(DateTime.Now.ToShortDateString());

        btnAdd.Enabled = false;
    }

    public void clear()
    {
        txtCIContact.Text = "";
        txtCIWebsite.Text = "";
        txtDeliverDate.Text = "";
        txtPreferTime.Text = "";
        txtProjDecription.Text = "";
    }
}

```

```
        txtProjName.Text = "";
    }
protected void btnSubmit_Click(object sender, EventArgs e)
{
    projCode();

    btnAdd.Enabled = true;

    btnSubmit.Enabled = false;
}

public void projCode()
{
    int i;

    int pcode=0;

    string pjcode = "select max(code) from prjcode";

    SqlConnection con1 = new SqlConnection("server=.;uid=sa;pwd=;database=SRPlanning");
    SqlCommand com1 = new SqlCommand(pjcode,con1);

    SqlDataReader dr;

    con1.Open();

    dr=com1.ExecuteReader();

    while (dr.Read())
    {
        pcode=int.Parse(dr[0].ToString());
    }

    i = pcode+1;

    con1.Close();

    con1.Open();

    string pjcode1 = "insert into prjcode values(" + i + ")";

    SqlCommand com2 = new SqlCommand(pjcode1, con1);

    com2.ExecuteNonQuery();
}
```

```

con1.Close();

string profile = "insert into ClientProfile values('" + lblCIUserName.Text + "','" +
txtCompName.Text + "','" + txtProjName.Text + "','" + i + "','" + txtProjDecription.Text + "','" +
txtCIWebsite.Text + "','" + txtCIContact.Text + "','" + txtCommitDate.Text + "','" +
txtDeliverDate.Text + "','" + ddlCIApplication.SelectedItem.ToString() + "','" +
ddlFrontEnd.SelectedItem.ToString() + "','" + ddlBackEnd.SelectedItem.ToString() + "','" +
ddlPlatform.SelectedItem.ToString() + "','" + ddlPreferDay.SelectedItem.ToString()
+ "','" + txtPreferTime.Text + "','" + rbtnConference.SelectedItem.ToString() + "','" + 'NotSet')";

con1.Open();

SqlCommand com4 = new SqlCommand(profile, con1);

com4.ExecuteNonQuery();

DirectoryInfo info = Directory.CreateDirectory("D:/Project-
2009/D9ESE01/softwarerelease/ClientRequirement/" + lblCIUserName.Text + "/" + i);

info.Create();

string s = FileUpload1.PostedFile.FileName;

string p = "D:/Project-2009/D9ESE01/softwarerelease/ClientRequirement/" +
lblCIUserName.Text + "/" + i.ToString() + "/";

FileInfo f = new FileInfo(s);

f.CopyTo(@p+f.Name);

con1.Close();
}

protected void btnDeliveryDate_Click(object sender, EventArgs e)
{
    Calendar2.Visible = true;
}

protected void Calendar2_SelectionChanged(object sender, EventArgs e)
{
    txtDeliverDate.Text = Calendar2.SelectedDate.ToShortDateString();

    if (FileUpload1.HasFile && FileUpload1.FileName != null)

```



```

    {
        FileUpload1.Attributes["filename"] = FileUpload1.FileName;
    }
}

protected void btnAdd_Click(object sender, EventArgs e)
{
    addProjClear();

    btnAdd.Enabled = false;

    btnSubmit.Enabled = true;
}

public void addProjClear()
{
    txtCompName.Enabled=false;
    txtCIWebsite.Enabled = false;
    txtCommitDate.Enabled = false;

    txtProjName.Text = "";
    txtProjDecription.Text = "";
    txtCIContact.Text = "";
    txtDeliverDate.Text = "";

    txtPreferTime.Text = "";

    rbtnConference.SelectedItem.Attributes.Clear();
}
protected void btnBack_Click(object sender, EventArgs e)
{
    Response.Redirect("ClientDetails.aspx");
}
protected void lnkbtnSignOut_Click(object sender, EventArgs e)
{
    Response.Redirect("~/webusercontrol/HomePage.aspx");
}
}

```

Coding for Management Profile

a) TaskAllocation.aspx

```

using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Data.SqlClient;

public partial class Development_TaskAllocation : System.Web.UI.Page
{
    SqlConnection con = new SqlConnection("server=.;uid=sa;pwd=;database=SRPlanning");
    int siz;

    protected void Page_Load(object sender, EventArgs e)
    {
        ddlPCode.AutoPostBack = true;

        lblName.Text=Session["Developer"].ToString();

        lblDomain.Text=Session["Domain"].ToString();

        Disable();

        ProjCode();

        lblGridDomain1.Text = lblDomain.Text;
    }

    public void Disable()
    {
        lblECommitDt.Visible = false;
        lblEDomain.Visible = false;
        lblEDuration.Visible = false;
    }
}

```

```

lblEID.Visible = false;
lblEPCode.Visible = false;
lblEPhase.Visible = false;
lblETargetDt.Visible = false;
lblEEmpName.Visible = false;

```

```

txtECommitDt.Visible = false;
txtEDomain.Visible = false;
txtEEmpName.Visible = false;
txtEPhase.Visible = false;
txtEPhaseDuration.Visible = false;
txtETargetDt.Visible = false;
txtEProjCode.Visible = false;

```

```

ddlEEmpID.Visible = false;

```

```

btnEAdd.Visible = false;
btnTarget.Visible = false;
btnScheduleTask.Visible = false;

```

```

lblGrid.Visible = false;
lblGridDomain1.Visible = false;

```

```

lblSchedule.Visible = false;

```

```

}

```

```

public void ProjCode()

```

```

{
    string pcod = "select ProjCode from ProjSchedule where Domain=" + lblDomain.Text +
    """;

```

```

SqlCommand com = new SqlCommand(pcod, con);

```

```

SqlDataReader dr;

```

```

con.Open();

```

```

dr = com.ExecuteReader();

```

```

while (dr.Read())

```

```

{
    ddlPCode.Items.Add(dr[0].ToString());
}

```

```

    con.Close();
}

```

```

protected void btnFetch_Click(object sender, EventArgs e)
{
    lblGridDomain1.Visible = true;
    lblGrid.Visible = true;
    btnScheduleTask.Visible = true;

    Session["Size"] = txtTeamSize.Text;

    GridFetch();
}

```

```

public void GridFetch()
{
    if (lblDomain.Text == ".NET")
    {
        string retriev = "select EmpId,EmpName,Status from DotNetTeam";

        SqlDataAdapter adt = new SqlDataAdapter(retriev, con);

        DataSet ds=new DataSet();

        con.Open();

        adt.Fill(ds);

        gdvwDeveloper.DataSource = ds;

        DataBind();

        con.Close();

    }
}

```

```

protected void ddlPCode_SelectedIndexChanged(object sender, EventArgs e)
{

```

```

    string retriev = "select DeliveryDt,TotalDuration,TeamSize from ProjSchedule where
ProjCode='" + ddlPCode.SelectedItem.ToString() + "'";

    SqlCommand com = new SqlCommand(retriev, con);

```

```

SqlDataReader dr;

con.Open();

dr = com.ExecuteReader();

while (dr.Read())
{
    txtDelDt.Text = dr[0].ToString();
    txtDuration.Text = dr[1].ToString();
    txtTeamSize.Text = dr[2].ToString();
}
con.Close();

}

protected void btnEAdd_Click(object sender, EventArgs e)
{

    int size1 =int.Parse(Session["Size"].ToString());

    int pcod;
    string add;

    if (size1 == 0)
    {
        btnEAdd.Enabled= false;
    }
    else
    {
        size1--;
        Session["Size"] = size1;
    }

    if (lblDomain.Text == ".NET")
    {

        pcod = Convert.ToInt32(txtEProjCode.Text);

        add = "update DotNetTeam set EmpName=" + txtEEmpName.Text + ",Domain=" +
txtEDomain.Text + ",ProjCode=" + pcod + ",Phase=" + txtEPhase.Text + ",CommitDt=" +
txtECommitDt.Text + ",Duration=" + txtEPhaseDuration.Text + ",TargetDt=" +
txtETargetDt.Text + ",Status='Yet To Finish' where EmpID=" +
+ddlEEmpID.SelectedItem.ToString() + """;

```

```

        SqlCommand com = new SqlCommand(add, con);

        con.Open();

        com.ExecuteNonQuery();

        con.Close();
    }
    else if (lblDomain.Text == "J2EE")
    {
        pcod = Convert.ToInt32(txtEProjCode.Text);

        add = "update J2EETeam set EmpName='" + txtEEmpName.Text + "',
Domain='" + txtEDomain.Text + "',ProjCode=" + pcod + ",Phase='" + txtEPhase.Text +
"',CommitDt='" + txtECommitDt.Text + "',Duration=" + txtEPhaseDuration.Text + ",TargetDt='"
+ txtETargetDt.Text + "',Status='Yet To Finish' where EmpID='" + txtEProjCode.Text + "'";

        SqlCommand com = new SqlCommand(add, con);

        con.Open();

        com.ExecuteNonQuery();

        con.Close();
    }
    else if (lblDomain.Text == "EMBEDDED")
    {
        pcod = Convert.ToInt32(txtEProjCode.Text);

        add = "update EmbeddedTeam set EmpName='" + txtEEmpName.Text + "',
Domain='" + txtEDomain.Text + "',ProjCode=" + pcod + ",Phase='" + txtEPhase.Text +
"',CommitDt='" + txtECommitDt.Text + "',Duration=" + txtEPhaseDuration.Text + ",TargetDt='"
+ txtETargetDt.Text + "',Status='Yet To Finish' where EmpID='" + txtEProjCode.Text + "'";

        SqlCommand com = new SqlCommand(add, con);

        con.Open();

        com.ExecuteNonQuery();

        con.Close();
    }
    else if (lblDomain.Text == "MAINFRAME")
    {
        pcod = Convert.ToInt32(txtEProjCode.Text);

```

```

        add = "update MainframeTeam set EmpName='" + txtEEmpName.Text + "',
Domain='" + txtEDomain.Text + "',ProjCode=" + pcod + ",Phase='" + txtEPhase.Text +
"',CommitDt='" + txtECommitDt.Text + "',Duration=" + txtEPhaseDuration.Text + ",TargetDt='"
+ txtETargetDt.Text + "',Status='Yet To Finish' where EmpID='" + txtEProjCode.Text + "'";

```

```

        SqlCommand com = new SqlCommand(add, con);

```

```

            con.Open();

```

```

            com.ExecuteNonQuery();

```

```

            con.Close();

```

```

        }

```

```

        Response.Write("<script>alert('Successfully Task Alocated... Continue!!')</script>");

```

```

        Enable();

```

```

        txtETargetDt.Visible = true;

```

```

        lblETargetDt.Visible = true;

```

```

        EEmpCode();

```

```

    }

```

```

    public void clear()

```

```

    {

```

```

        txtECommitDt.Text = "";

```

```

        txtEDomain.Text = "";

```

```

        txtEEmpName.Text = "";

```

```

        txtEPhase.Text = "";

```

```

        txtEPhaseDuration.Text = "";

```

```

        txtEProjCode.Text = "";

```

```

        txtETargetDt.Text = "";

```

```

    }

```

```

    protected void lnkbtnSignOut_Click(object sender, EventArgs e)

```

```

    {

```

```

        Response.Redirect("~/webusercontrol/HomePage.aspx");

```

```

    }

```

```

    protected void btnScheduleTask_Click1(object sender, EventArgs e)

```

```

    {

```

```

        Enable();

```

```
DataLoad();

EEmpCode();

}

public void Enable()
{
    lblEEmpName.Visible = true;
    lblECommitDt.Visible = true;
    lblEDomain.Visible = true;
    lblEDuration.Visible = true;
    lblEID.Visible = true;
    lblEPCode.Visible = true;
    lblEPhase.Visible = true;

    txtEEmpName.Visible = true;
    txtECommitDt.Visible = true;
    txtEDomain.Visible = true;
    txtEEmpName.Visible = true;
    txtEPhase.Visible = true;
    txtEPhaseDuration.Visible = true;

    txtTeamSize.Visible = true;
    txtEProjCode.Visible = true;

    ddlEEmpID.Visible = true;

    btnTarget.Visible = true;

    lblGridDomain1.Visible = true;
    lblGrid.Visible = true;
    btnScheduleTask.Visible = true;
    btnScheduleTask.Enabled = false;
}

public void DataLoad()
{
    txtEDomain.Text = lblDomain.Text;
```



```
txtECommitDt.Text = Convert.ToString(DateTime.Now.ToShortDateString());

txtEProjCode.Text = ddlPCode.SelectedItem.ToString();
}

public void EEmpCode()
{
    if (lblDomain.Text == ".NET")
    {
        string ID = "select EmpID from DotNetTeam where Status='Completed'";

        SqlCommand com = new SqlCommand(ID, con);

        SqlDataReader dr;

        con.Open();

        dr = com.ExecuteReader();

        while (dr.Read())
        {
            ddlEEmpID.Items.Add(dr[0].ToString());
        }

        con.Close();
    }
    else if (lblDomain.Text == "J2EE")
    {
        string ID = "select EmpID from J2EETeam where Status='Completed'";

        SqlCommand com = new SqlCommand(ID, con);

        SqlDataReader dr;

        con.Open();

        dr = com.ExecuteReader();

        while (dr.Read())
        {
            ddlEEmpID.Items.Add(dr[0].ToString());
        }
    }
}
```

```
con.Close();
}
else if (lblDomain.Text == "EMBEDDED")
{
    string ID = "select EmpID from EmbeddedTeam where Status='Completed'";
        SqlCommand com = new SqlCommand(ID, con);

    SqlDataReader dr;

    con.Open();

    dr = com.ExecuteReader();

    while (dr.Read())
    {
        ddlEEmpID.Items.Add(dr[0].ToString());
    }

    con.Close();
}
else if (lblDomain.Text == "MAINFRAME")
{
    string ID = "select EmpID from MainframeTeam where Status='Completed'";

    SqlCommand com = new SqlCommand(ID, con);

    SqlDataReader dr;

    con.Open();

    dr = com.ExecuteReader();

    while (dr.Read())
    {
        ddlEEmpID.Items.Add(dr[0].ToString());
    }

    con.Close();
}
```

```

    }
}
protected void ddlEEmpID_SelectedIndexChanged(object sender, EventArgs e)
{
    con.Open();

    if (lblDomain.Text == ".NET")
    {
        string ename = "select EmpName from DotNetTeam where
EmpID='"+ddlEEmpID.SelectedItem.ToString()+"'";

        SqlCommand com = new SqlCommand(ename, con);

        SqlDataReader dr;

        dr = com.ExecuteReader();

        while (dr.Read())
        {
            txtEEmpName.Text=dr[0].ToString();
        }
    }

    con.Close();

    Enable();
}

protected void btnTarget_Click(object sender, EventArgs e)
{
    Enable();

    lblETargetDt.Visible = true;
    txtETargetDt.Visible = true;

    btnEAdd.Visible = true;
    string s;

    s = Convert.ToString(txtECommitDt.Text);

    System.DateTime ed = DateTime.Parse(s);

    ed = ed.AddDays(int.Parse(txtEPhaseDuration.Text));
}

```

```

txtETargetDt.Text = ed.Date.ToString();

}

protected void lnkbtnBack_Click(object sender, EventArgs e)
{
    Response.Redirect("SRP.aspx");
}
}

```

b) ProjectSchedule.aspx

```

using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Data.SqlClient;

public partial class managment_Commitment : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        txtBookDate.Text = Convert.ToString(DateTime.Now.ToShortDateString());

        lblDelDtToClient.Visible = false;

        Projcode();
    }
    public void Projcode()
    {
        SqlConnection con = new SqlConnection("server=.;uid=sa;pwd=;database=SRPlanning");

        string pcode = "select ProjCode from ClientProfile";
    }
}

```

```

SqlCommand com = new SqlCommand(pcode,con);

SqlDataReader dr;

con.Open();

dr = com.ExecuteReader();

while (dr.Read())
{
    ddlpcode.Items.Add(dr[0].ToString());
}

con.Close();
}

protected void ddlpcode_SelectedIndexChanged(object sender, EventArgs e)
{
    DelivDtToClient();

    lblDelDtToClient.Visible = true;

    ProjDomain();

    TeamSize();
}

public void TeamSize()
{
    SqlConnection con = new SqlConnection("server=.;uid=sa;pwd=;database=SRPlanning");

    string team = "select TeamSize from ProjCommitReport where ProjCode="" +
    ddlpcode.SelectedItem.ToString() + """;

    SqlCommand com = new SqlCommand(team, con);

    SqlDataReader dr;

    con.Open();

    dr = com.ExecuteReader();

    while (dr.Read())
    {
        txtTmMembers.Text = dr[0].ToString();
    }
}

```

```

    }

    con.Close();
}

protected void btnschedule_Click(object sender, EventArgs e)
{
    Add();

    string s = txtBookDate.Text;

    System.DateTime ed = DateTime.Parse(s);

    ed = ed.AddDays(int.Parse(txtduration.Text));

    txtDelivDt.Text = ed.Date.ToString();

    Response.Write("<script>alert('Successfully The Project Is Scheduled')</script>");
}

public void Add()
{
    int t;

    t = Convert.ToInt32(txtanalysis.Text) + Convert.ToInt32(txtdesign.Text) +
    Convert.ToInt32(txtdevelop.Text) + Convert.ToInt32(txtintegrate.Text) +
    Convert.ToInt32(txttest.Text) + Convert.ToInt32(txtimplement.Text);

    txtduration.Text = t.ToString();
}

protected void btnNext_Click(object sender, EventArgs e)
{
    Save();

    Response.Redirect("HomeDetails.aspx");
}

public void Save()
{
    SqlConnection con = new SqlConnection("server=.;uid=sa;pwd=;database=SRPlanning");

    string Schedule = "insert into ProjSchedule
values('"+ddlpcode.SelectedItem.ToString()+"', '"+txtDomain.Text+"', '"+txtTLID.Text+"', '"+txtT

```

```
eamLeader.Text+"", "+txtTmMembers.Text+", "+txtBookDate.Text+", "+txtanalysis.Text+", "+txt
design.Text+", "+txtdevelop.Text+", "+txtintegrate.Text+", "+txttest.Text+", "+txtimplement.Text+
", "+txtduration.Text+", "+txtDelivDt.Text+"");
```

```
SqlCommand com1 = new SqlCommand(Schedule, con);
```

```
con.Open();
```

```
com1.ExecuteNonQuery();
```

```
con.Close();
```

```
}
```

```
public void DelivDtToClient()
```

```
{
```

```
SqlConnection con = new SqlConnection("server=.;uid=sa;pwd=;database=SRPlanning");
```

```
string delDt = "select DeliveryDt from ClientProfile where ProjCode=" +
ddlpcode.SelectedItem.ToString() + """;
```

```
SqlCommand com = new SqlCommand(delDt, con);
```

```
SqlDataReader dr;
```

```
con.Open();
```

```
dr = com.ExecuteReader();
```

```
while (dr.Read())
```

```
{
```

```
lblDelDtToClient.Text = dr[0].ToString();
```

```
}
```

```
con.Close();
```

```
}
```

```
public void ProjDomain()
```

```
{
```

```
SqlConnection con = new SqlConnection("server=.;uid=sa;pwd=;database=SRPlanning");
```

```
string ProjCode = "select Platform from ClientProfile where ProjCode=" +
ddlpcode.SelectedItem.ToString() + """;
```

```
SqlCommand com = new SqlCommand(ProjCode, con);
```

```
SqlDataReader dr;
```

```

con.Open();

dr = com.ExecuteReader();

while (dr.Read())
{
    txtDomain.Text=dr[0].ToString();

}
if ((txtDomain.Text == ".NET") || (txtDomain.Text == "J2EE") || (txtDomain.Text ==
"EMBEDDED") || (txtDomain.Text == "MAINFRAME"))
{
    TeamLead();
}
con.Close();
}
public void TeamLead()
{
    SqlConnection con = new SqlConnection("server=.;uid=sa;pwd=;database=SRPlanning");

    string TL = "select EmpID,TLName from TeamLeader where Domain=" + txtDomain.Text
+ """;

    SqlCommand com = new SqlCommand(TL, con);

    SqlDataReader sdr;

    con.Open();

    sdr = com.ExecuteReader();

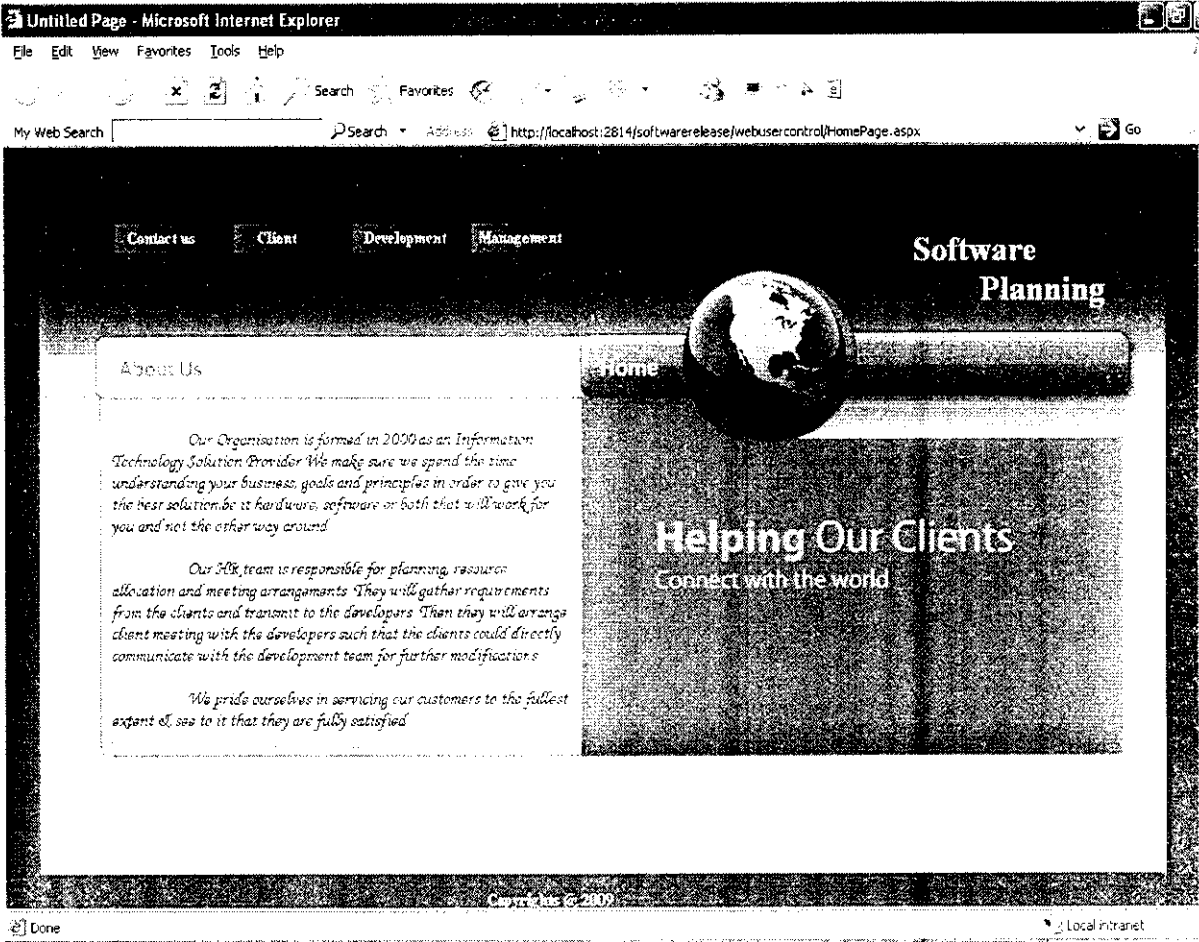
    while (sdr.Read())
    {
        txtTLID.Text = sdr[0].ToString();

        txtTeamLeader.Text = sdr[1].ToString();
    }
    con.Close();
}
protected void InkbtnSignOut_Click(object sender, EventArgs e)
{
    Response.Redirect("~/webusercontrol/HomePage.aspx");
}

```


SNAP SHOTS

HOME PAGE:



CLIENT LOG IN:

Untitled Page - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites Go

My Web Search Search http://localhost:2014/softwarerelease/client/clienthome.aspx Go

SoftWare Release Planning

Log In

Register

Sign Out

Sign In

User Name

Password

Submit

javascript: __doPostBack('ctl00\$ContentPlaceHolder1\$lnkbtnClientSignOut','') Local Intranet

CLIENT REGISTRATION:

Untitled Page - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites Go

My Web Search Search http://localhost:2014/softwarerelease/client/LoginRegistration.aspx Go

SoftWare Release Planning

CLIENT REGISTRATION

First Name

Last Name

Password

Re-Enter Password

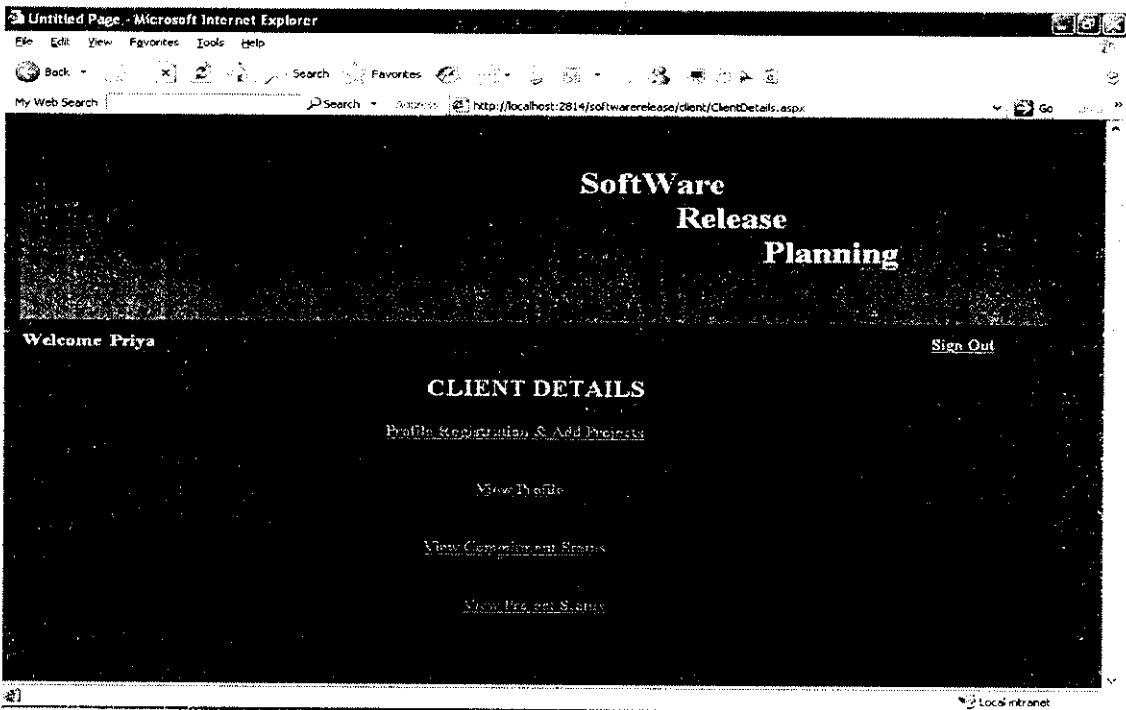
Mail ID

Date

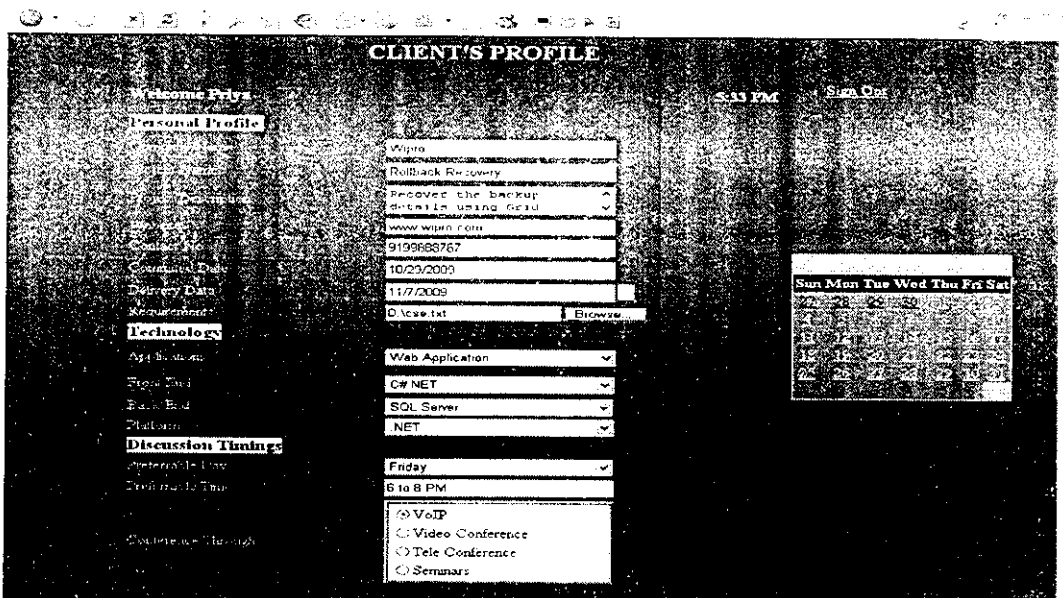
Submit Back

Done Local Intranet

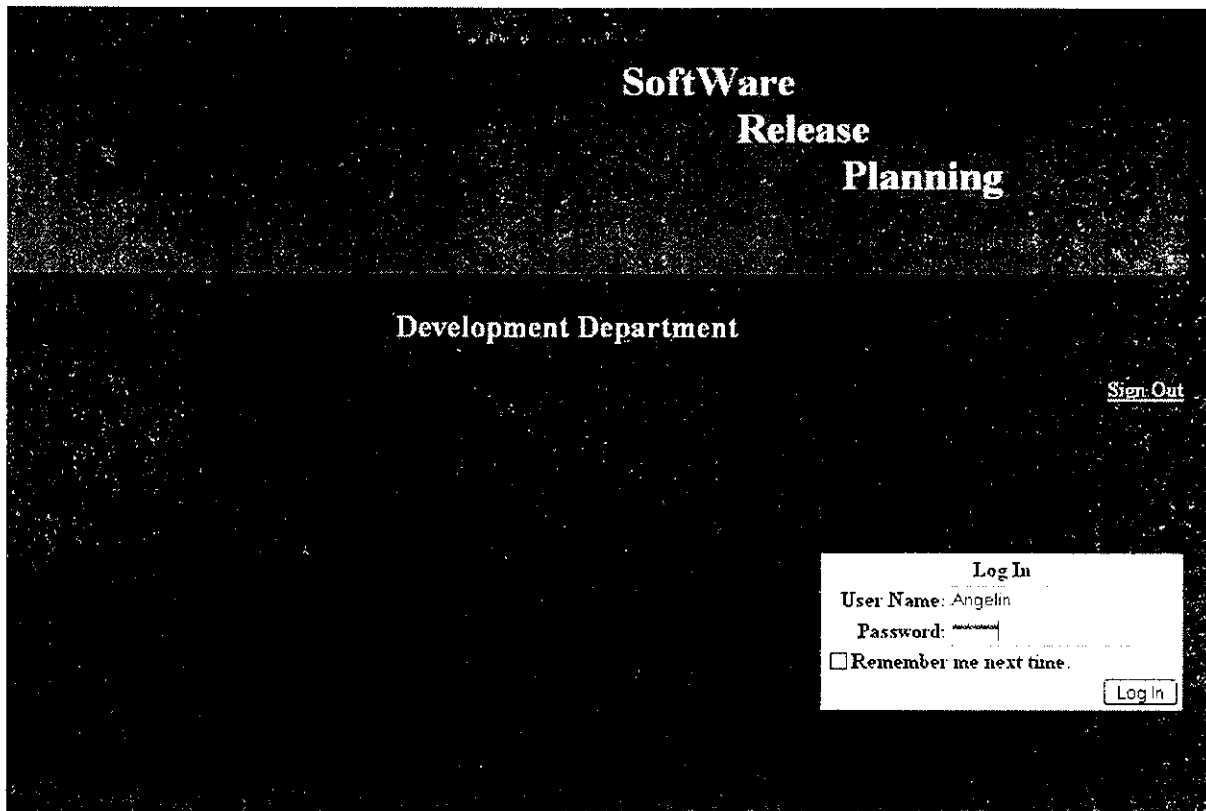
CLIENT DETAILS:



CLIENT PROFILE REGISTRATION:



DEVELOPMENT LOG IN:



DEVELOPMENT DETAILS:

Domain : NKT

Client Name	Comp Name	ID of Name	ID of Code	Contact	Delivery Date
parrot	swiz	aa	209	7208	10/17/2009 12:00:00 AM
parrot	dfasdd	multic act	212	980876765	11/17/2009 12:00:00 AM
Angel	Wipro	nascom	212	98754987	12/5/2009 12:00:00 AM
Angel	Wipro	viscom	214	9876598768	1/9/2010 12:00:00 AM
					10/29/2009

DEVELOPMENT TEAM STATUS:

Soft Ware Release Planning

Status About The Team Members

Welcome Angelin
Domain: NET

Sign Out
IP: D111

Team Status

EmpName	EmpID	ProjCode	Phase	TargetDt	Status1
Pragathi	J1212	209	Analysis	11/21/2009 12:00:00 AM	Yet to finish
Angelin	D1111	213	Testing	11/25/2009 12:00:00 AM	Yet to finish
Geetha	D1224	213	Testing	10/30/2009 12:00:00 AM	Completed
Geetha	D1225	213	Integrate	11/9/2009 12:00:00 AM	Completed

Task To Be Assigned To...

EmpName	EmpID
Geetha	D1224
Geetha	D1225

DEVELOPMENT PROJECT STATUS:

Project Status

Sign Out

PHASE

Project Code: 213

Domain: NET

Client Name: Angelin

Client Date: 10/12/2009 12:00:00 AM

Analysis Phase: 12

Development Phase: 3

Integration Phase: 45

Testing Phase: 12

Implementation Phase: 3

Total Duration: 86

DEMO DATES

First Demo On: 10/29/2009 12:00:00 AM

Second Demo On: 12/13/2009 12:00:00 AM

Final Demo On: 1/3/2010 12:00:00 AM

Delivered On: 1/6/2010 12:00:00 AM

MANAGEMENT LOG IN:

**SoftWare
Release
Planning**

SRP Management

[Sign Out](#)

Log In

User Name: admin

Password:

Remember me next time.

SCHEDULE OF PROJECT:

Project Schedule

[Sign Out](#)

Project Allotment		Delivered To Client On:
Project code	227	
Allocated Domain	.NET	
Team Leader ID	D111	
Team Leader	Angelin	
Team Members	8	
Allocated		
Project Start	11/14/2009	
Schedule		
Analysis Phase	12	
Design Phase	5	
Development Phase	45	
Integration Phase	12	
Testing Phase	9	
Implementation Phase	3	
<input type="button" value="Schedule"/>		
Total Duration	86	
Delivery Date	2/9/2010 12:00:00 AM	

DEVELOPERS PROFILE DETAILS:

Untitled Page - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost:3404/software/release/management/DeveloperDetails.aspx

Developer's Profile

Sign Out

NET DEVELOPERS

EmpID	EmpName	ProjCode	Phase	Status
J1212	Pragathi	209	Analysis	Yet to finish
D111	Angelin	213	Testing	Yet to finish
D1224	Geetha	213	Analysis	Yet to finish
D1225	Geetha	213	Integrate	Completed

J2EE DEVELOPERS

Sorry!! There Are No Developers In JAVA Domain

EMBEDDED DEVELOPERS

EmpID	EmpName	ProjCode	Phase	Status
E21213	Deepa	201	Analvse	Completed

MAINFRAME DEVELOPERS

EmpID	EmpName	ProjCode	Phase	Status
M21213	Rekaa	202	Testing	Yet To Finish

Done

ALLOCATION OF TASK TO DEVELOPER:

Task Allocation

Sign Out/Back

Welcome Angelin

Domain: NET

213

1/6/2010 12:00:00 AM

86 Days

7

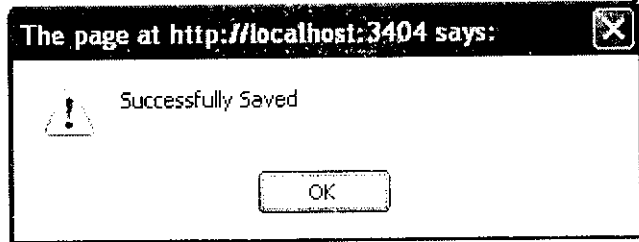
View Developers

NET - Employee's Status

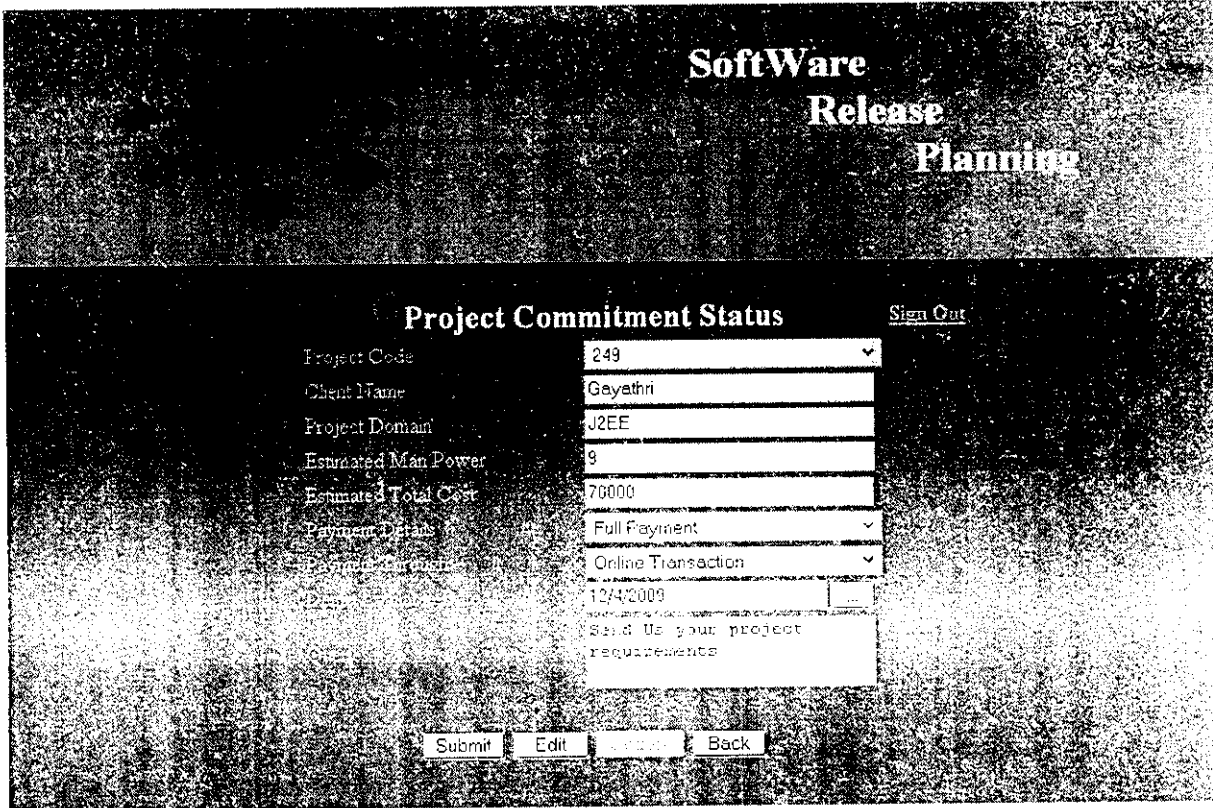
EmpId	EmpName	Status
J1212	Pragathi	Yet to finish
D111	Angelin	Yet to finish
D1224	Geetha	Completed
D1225	Geetha	Completed

Schedule

TASK ALLOTTED:



PROJECT COMMITMENT STATUS:



REFERENCES

- [1] S. Acuña, N. Juristo, and A.M. Moreno, "Emphasizing Human Capabilities in Software Development," *IEEE Software*, vol. 23, no. 2, pp. 94-101, Mar./Apr. 2006.
- [2] M. Crissis, M. Konrad, and S. Shrum, *CMMI—Guidelines for Process Integration and Product Improvement*. Addison-Wesley, 2006.
- [3] A. Amandeep, G. Ruhe, and M. Stanford, "Intelligent Support for Software Release Planning," *Proc. Fifth Int'l Conf. Product Focused Software Process Improvement*, pp. 248-262, 2004.
- [4] J. Blazewicz, W. Domschke, and E. Pesch, "The Job Shop Scheduling Problem: Conventional and New Solution Techniques," *European J. Operational Research*, vol. 93, no. 1, pp. 1-33, 1996.
- [5] L. Briand, J. Feng, and Y. Labiche, "Experimenting with Genetic Algorithms and Coupling Measures to Devise Optimal Integration Test Orders," *Software Eng. with Computational Intelligence*, pp. 204- 234, Kluwer Academic Publishers, 2003.
- [6] TRAVASSOS, G.H., GUROV, D., AMARAL, E.A.G.G., "Introduction to Experimental Software Engineering," In: Technical Report ES-590/02-April. PESC. COPPE/UFRJ, available at <http://www.cos.ufrj.br/publicacoes>, 2001.
- [7] Ruhe G, Ngo-The A. 2004. "Hybrid intelligence in software release planning." *International Journal of Hybrid Intelligent Systems* 1(2): 99–110.
- [8] Saliu O, Ruhe G. 2005. "Software release planning for evolving systems." *Innovations in Systems and Software Engineering – A NASA Journal* 1(2): 189–204.
- [9] Greer, D., Bustard, D. and Sunazuka, T., "Effecting and Measuring Risk Reduction in Software Development", *NEC Journal of Research and Development*, Vol.40, No.3, pp.378-38, July, 1999.

- [10] Ruhe, G., "Software Engineering Decision Support: Methodology and Applications". In: Innovations in Decision Support Systems (Ed. by Tonfoni and Jain). International Series on Advanced Intelligence Volume 3, 2003, pp 143-174.
- [11] Karlsson, J., "Software requirements prioritizing", Proceedings of the Second International Conference on Requirements Engineering, pp 110 –116, 1996.
- [12] O.Saliu and G.Ruhe, "Supporting Software Release Planning Decisions for Evolving Systems," Proc. 29th IEEE/NASA Software Eng. Workshop, Apr. 2005.
- [13] P. Kapur, A. Ngo-The, G. Ruhe, and A. Smith, "Optimized Staffing for Product Releases and Its Application at Chartwell Technology," J. Software Maintenance and Evolution, vol. 20, pp. 365-386, 2008.
- [14] Wang, Q. and Lai, X.: Proc. Requirements Management for the Incremental Development Model. 2nd Asia-Pacific Conference on Quality Software, pp 295-301, 2001.
- [15] Acuña, N. Juristo, A.M. Moreno, "Emphasizing Human Capabilities in Software Development." IEEE Software 23(2): pp. 94-101 (2006).
- [16] ZELKOWITZ, M.V., WALLACE, D.R., BINKLEY, D.W., "Experimental Validation of New Software Technology," Lecture Notes On Empirical Software Engineering, Chapter 6, pp 229-263, World Scientific, 2003.
- [17] Larman, C.: "Agile & Iterative Development, A Manager's Guide." Addison-Wesley, 2003.
- [18] S.T. Acuña and N. Juristo, A.M. Moreno (2006): "Emphasizing Human Capabilities in Software Development." IEEE Software, Volume 23, Issue 2, Page, 94-101.
- [19] Yomi Kastro, Ayse Basar Bener (2008): "A defect prediction method for software versioning." Software Quality Journal, Volume 16, Number 4, Pages 543-562.
- [20] Magne Jorgensen, M. Shepperd (2007): "A systematic review of software development cost estimation studies." IEEE Transactions on Software Engineering, Volume 33, Issue 1, Pages 33–53.