TEST ENGINE SYSTEM

(TES)

SOFTWARE PROJECT PLAN
PHASE I

April 8, 2003

Version 1.0

Team Members

Test Engine System
Software Project Plan
Table of Contents

1.0 Introduction

The ‘test engine’ program provides test administrators an easy way to create exams, give exams, and monitor exam-taker responses. The ‘test engine’ will consist of an administration ‘view’ and a user ‘view’. The administrator of the system will be the professor administering the exam. The user of the system will be the exam-taker or the student taking the exam. The exam-taker will scroll through each exam question of the exam he is taking and provide a response to the question before moving on to the next question.
Exam questions will be entered by the exam administrator prior to the exam time. Each exam will be ‘activated’ by an exam administrator. Once the exam is ‘activated’, the exam-takers will be able to begin the exam and post answers to exam questions. The ‘test engine’ program will be delivered as an intranet solution. Use of the program is envisioned to take place in one classroom at one given instance of time. Hence, if two exams are going on at the same time by two different professors, each classroom will have its own URL / intranet website to view the exam and enter responses to exam questions.

1.1 Project Scope

The goal of the system is to provide test administrators an easy way to create exams, give exams and monitor exam-taker responses. The system will also provide a uniform interface to exam-takers and an ability to immediately display exam results to the exam taker and compile exam reports to the system administrator.

The presentation layer will be divided into two parts: administration and user. The ‘logic engine’ will consist of a library to process calculations required by the system administrator based on student responses to questions during the exam. The ‘logic engine’ will reside on the same server as the web-server. The data-store will retain all data related to the exam.
Inputs into the system will be by both the administrator and by the exam taker of the system. The administrator will enter an exam into the system. Exam type will be specified. Example exam types include ‘real exam’ and ‘practice exam’. The administrator will enter an exam type. Example exam question types could be ‘multiple choice’ or ‘true / false’. Based on the question type, the administrator of the system is prompted for an exam question and possible answers. The amount of answer choices the administrator enters will be dependent on the exam question type selected.
Another input into the system is answer responses entered by the user taking the exam. The user is prompted for the exam type to take. Example exam types include ‘real exam’ and ‘practice exam’. The exam-taker is displayed an exam question and prompted for an answer to the exam question. Upon answering all questions for the exam, the user comes to the last question in the exam. Upon clicking finish, the exam-taker sends all results of his exam to a persistent data store.
The administrator can also enter usernames / passwords into the system. The username / password will be used to authenticate the user in the logon process.

The administrator will also have the ability to view reports based on the exam responses. Sample reports include: List of current students for an administrators' classroom, exam results per student (group all student exam grades together) and student response report (showing a detailed response per student).

1.2 Major software functions

The system should be simple enough to be used by people with elementary browsing skills.
The people who will use the system are: students and the professor. The system should permit students to take practice or real graded tests and also to view their test results. The professor should be able to create tests, activate them, evaluate questions and also view various reports.

The information entered in the system will be:

- Students
  - Login name and password.
  - Test answers.
- Professor
  - Login name and password.
  - Test questions.
  - Test answers.
  - Student login names and their passwords.

The recipients of the information produced by the system will be:

Students
View and take practice tests.
Take real exam.
View exam results.
View question answers after finishing the test.
Professor
View various reports such as class list, class grades, test questions and their respective answers.

Actors in the "Test engine" system will be:

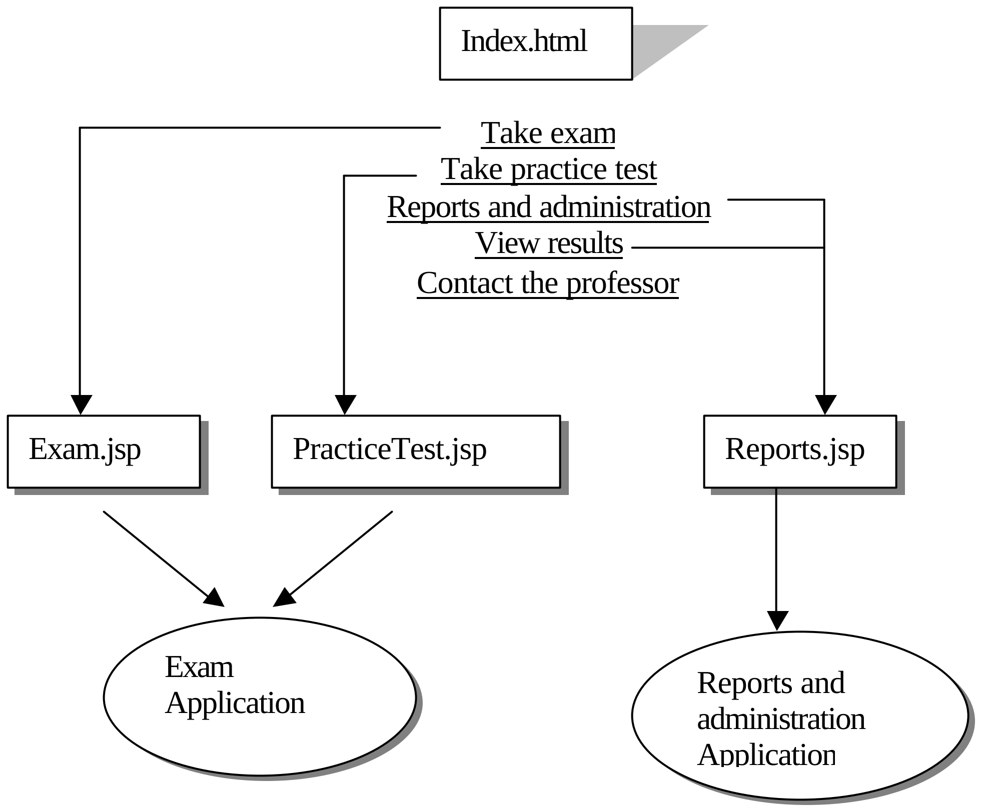
Professor: (System Administrator – privileged user)
Students: (Limited user)

1.3 Performance/Behavior Issues

Test Engine is design to be compatible with the browser Internet Explorer and Netscape Navigator.

1.4 Management and Technical constrains

Test Engine has a drop dead delivery date of 05/01/03.



2.0 Project estimates

2.1 Historical data used for estimates

The company has recorded the following data from previous projects.

Project Id	Size(KLOC)	Effort(PM)
1	50	120
2	80	192
3	40	96
4	10	24
5	20	48

2.2 Estimation techniques

2.2.1

We will use first LOC-Based Estimation . The formula used is:

$$\text{Cost} = a * \text{KLOC}^{ß} + ?$$

Where KLOC stands for “thousand lines of code” and Cost is expressed in “Programmer Month” effort.

From the data given in table in point 2.1 we can derive

$$a=2.4$$
$$\mathbf{ß=1}$$
$$?=0$$
$$\text{Cost} = 2.4 * \text{KLOC}$$

2.2.2

We will also use the following COCOMO formulas derived for the application programs:

$$PM=2.4*(KDSI)^{1.05}$$
$$TDEV=2.5*(PM)^{0.38}$$

Where

PM stands for programmer-month
KDSI (thousand delivered source instructions) is the equivalent of KLOC
TDEV stands for the standard development time.

2.3 Estimates

2.3.1 LOC – based cost estimation

Estimated (LOC) = (Optimistic + 4xMost-Likely + Pestimistic)/6

Module	LOC			
	Optimistic	Most Likely	Pessimistic	Estimated
ExamApp	400	600	800	634
ReportsApp	500	700	900	700
Configurations	70	100	300	129
Database	600	800	1500	1083
Total	**1570**	**2200**	**3500**	**2311**

Standar deviation = 576
Cost= 2.4* 2.311= 5.6 (programmer months)

2.3.2 COCOMO – based cost estimation

PM = 5.78
TDEV = 4.87

2.4 Project resources

People

Test Engine will be developed by:

Faisul Haque
Joseph Schoback
Artan Rukaj

Hardware and Software

Hardware

- PC Pentium 2 GHz , 256MB RAM

Software Packages

- **JDK**(tm) - Java(tm) Development Kit 1.4
- Apache Tomcat 1.4.1
- MySql

Development Tools

- Notepad

3.0 Project risks

This section discusses project risks and the approach to managing them. Risk identification is the process of identifying possible risks. Risks can be classified as affecting the project plan (project risks), affecting the quality (technical risks), or affecting the viability of the product (business risks). Risk estimation involves estimating the risk probability and the risk impact.

3.1 Project Risks

Identification

Staff cannot contribute enough time to the project.

Staff does not have enough experience in the software tool.

Product cannot be released within the deadline.

Do not have enough documents for the users.

Customers keep changing the requirements.

Staff cannot provide enough support to the customers.

Product has not been tested enough before it is released.

Running the product causes unexpected effect to running environment.

The performance is down because too many users use the product at the same time.

Users hack the system.

(For a more detailed list of project risks, see the Risk Mitigation, Monitoring, and Management – RMMM document)

3.2 Risk Table

Category	Risks	Probability	Impact
Employee Risk	Staff cannot contribute enough time to the project	35%	1
Employee Risk	Staff does not have enough experience in the software tool.	70%	1
Business Risk	Product cannot be released within the deadline.	40%	1

Business Risk	Do not have enough documents for the users	30%	2
Customer Risk	Customers keep changing the requirements.	70%	2
Employee Risk	Staff cannot provide enough support to the customers	50%	2
Process Risk	Product has not been tested enough before it is released	40%	4
Technology Risk	Running the product causes unexpected effect to running environment	1%	1
Technology Risk	The performance is down because too many users use the product at the same time.	30%	1

Impact Values	Description
1	Catastrophic
2	Critical
3	Marginal
4	Negligible

3.3 Overview of Risk Mitigation, Monitoring, Management

3.3 .1 Risk aversion options and Risk monitoring procedures

Staff cannot contribute enough time to the project.

Strategy:

- Manage the timetable.

Monitoring activities -- The following factors can be monitored:

- Project schedule

Staff does not have enough experience in the software tool.

Strategy:

- Train the staff.
- Encourage the staff to study more.

Monitoring activities -- The following factors can be monitored:

- Time that the staff spends on development
- Number of errors.
- Number of defects

Product cannot be released within the deadline.

Strategy:

- Keep in touch with the customers and find the appropriate deadline.
- Find more staff.

Monitoring activities -- The following factors can be monitored:

- Project schedule
- Other related schedule

Do not have enough documents for the users (quality and quantity).

Strategy:

- Update the documents often.
- Collect the feedback about the documents from the users.
- Provide alternatives of support such as electronic message board.

Monitoring activities -- The following factors can be monitored:

- Understanding of the users to the product.
- Number of messages in electronic message board.

Customers keep changing the requirements.

Strategy:

- Keep in touch with the customers.
- Have the formal requirements gathering meetings with the customers.

Monitoring activities -- The following factors can be monitored:

- Usability of the product

Staff cannot provide enough support to the customers.

Strategy:

- Provide on-line help.

- Provide on-line manual.
- Provide training sessions.
- Provide alternatives of support such as electronic message board.

Monitoring activities -- The following factors can be monitored:

- Amount of e-mail from the users about the problem of using the product.
- Number of messages in electronic message board.

Product has not been tested enough before it is released.

Strategy:

- Make sure to have enough tests.
- Negotiate the deadline not to be too early.

Monitoring activities -- The following factors can be monitored:

- Amount of test before the product is released
- Number of defects

Running the product causes unexpected effect to the system.

Strategy:

- Provide enough tests before the product is released.
- Provide backup database.
- Provide logging and recovery strategy.

Monitoring activities -- The following factors can be monitored:

- Performance and load of the running environment

The performance is down because too many users use the product at the same time.

Strategy:

- Limit number of users in a session.
- Provide the statistic about when is the most popular time of the users to use the product so that some of them maybe able to avoid the traffic.

Monitoring activities -- The following factors can be monitored:

- Users' behavior in using the product
- Performance and load of the running environment

Users hack the system.

Strategy:
Provide secured access.

Monitoring activities -- The following factors can be monitored:
Users' behavior in using the product

4.0 Project Schedule

4.1 List of deliverables

Documentation and Analysis

Software system specification
Software requirements specification
Preliminary software design specification
User interface design
Test plan

Implementation

Architectural design representations
Database design
Component level design and representations
User interface design
Complete product and project demonstration

4.2 Functional Decomposition

Database task breakdown

Create database table statements
Stored procedure / embedded query construction

- Insert new user
- Select user with username / password as parameters
- Insert new exam
- Insert new exam question
- Update exam record for 'activating exam'
- Select exam questions with examID as parameter
- Update studentResponse table with student response to question
- Select studentResponses passing examID and studentID as parameters returning a grade for the given exam for the student
- Select studentResponses passing examID as parameter summing grades for the entire exam

Code component / UI task breakdown

Database utility component for database access
Create 'new user' JSP page / component
Create 'log into system' JSP page / component
Create 'new exam' JSP page / component
Create 'enter new question' JSP page / component
Create 'activate exam' JSP page / component
Create 'display exam question' JSP page / component

Create ‘exam results’ JSP page / component

Help task breakdown

User guide on-line html manual
Frequently asked questions on-line html manual

Testing task breakdown

Unit testing
Integration testing
Beta testing

Configuration task breakdown

Web Server Configuration
DataBase Configuration
Client Configuration

4.3 Deliverables and Milestones

Allocated resources and resource costs

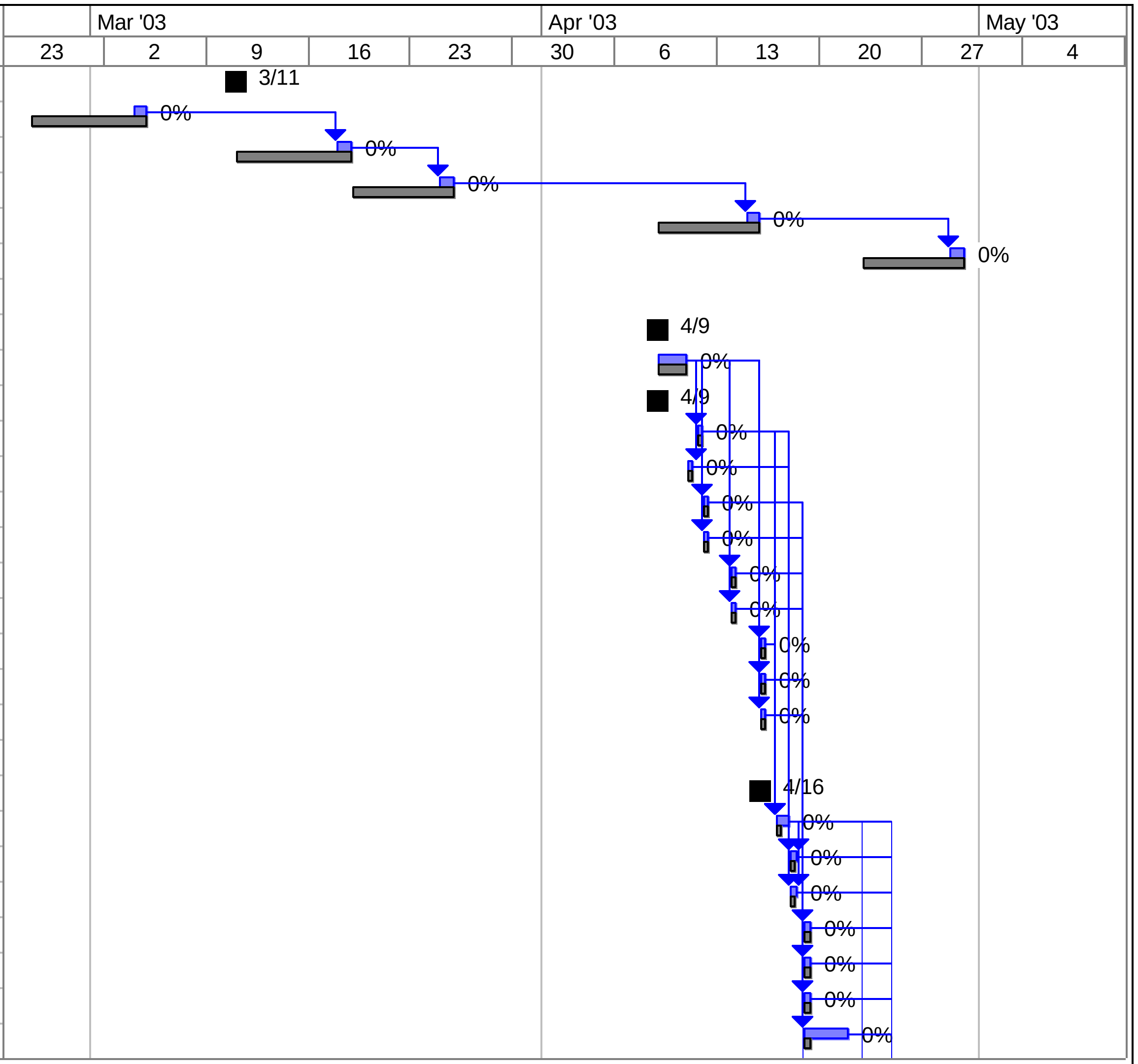
Joseph Shoback	JS	$100.00/h
Artan Rukaj	AR	$90.00/h
Faizul Haque	FH	$75.00/h

4.4 Timeline chart / Project Plan

ID	Resource Name	Work	Details	M	A	M	J	J	A
	Unassigned	0 hrs	Work						
	Documentation and Analysis	*0 hrs*	Work						
	Database task breakdown	*0 hrs*	Work						
	Stored procedure / embedded query construction	*0 hrs*	Work						
	Code component / UI task breakdown	*0 hrs*	Work						
	Testing task breakdown	*0 hrs*	Work						
	Help task breakdown	*0 hrs*	Work						
	Configuration task breakdown	*0 hrs*	Work						
1	**J S**	**128 hrs**	Work	24h	96h	8h			
	Software system specification	*8 hrs*	Work	8h					
	Software requirements specification	*8 hrs*	Work	8h					
	Preliminary software design specification	*8 hrs*	Work	8h					
	User interface design	*8 hrs*	Work		8h				
	Test plan	*8 hrs*	Work		8h				
	Create 'new user' JSP page / component	*4 hrs*	Work		4h				
	Create 'log into system' JSP page / component	*4 hrs*	Work		4h				
	Create 'exam results' JSP page / component	*16 hrs*	Work		16h				
	Unit testing	*16 hrs*	Work		16h				
	Integration testing	*16 hrs*	Work		16h				
	Beta testing	*16 hrs*	Work		16h				
	User guide on-line html manual	*8 hrs*	Work		8h				
	Web Server Configuration	*8 hrs*	Work			8h			
2	**A R**	**132 hrs**	Work	24h	92h	16h			
	Software system specification	*8 hrs*	Work	8h					
	Software requirements specification	*8 hrs*	Work	8h					
	Preliminary software design specification	*8 hrs*	Work	8h					
	User interface design	*8 hrs*	Work		8h				
	Test plan	*8 hrs*	Work		8h				
	Create 'new exam' JSP page / component	*4 hrs*	Work		4h				
	Create 'enter new question' JSP page / component	*4 hrs*	Work		4h				
	Create 'activate exam' JSP page / component	*4 hrs*	Work		4h				
	Create 'display exam question' JSP page / component	*16 hrs*	Work		16h				
	Unit testing	*16 hrs*	Work		16h				
	Integration testing	*16 hrs*	Work		16h				
	Beta testing	*16 hrs*	Work		16h				
	Frequently asked questions on-line html manual	*8 hrs*	Work			8h			
	Client Configuration	*8 hrs*	Work			8h			
3	**F H**	**138 hrs**	Work	24h	106h	8h			
	Software system specification	*8 hrs*	Work	8h					
	Software requirements specification	*8 hrs*	Work	8h					
	Preliminary software design specification	*8 hrs*	Work	8h					
	User interface design	*8 hrs*	Work		8h				
	Test plan	*8 hrs*	Work		8h				
	Create database table statements	*16 hrs*	Work		16h				
	Insert new user	*2 hrs*	Work		2h				

ID	Resource Name	Work	Details	M	A	M	J	J	A
	Select user with username / password as parameters	*2 hrs*	Work		2h				
	Insert new exam	*2 hrs*	Work		2h				
	Insert new exam question	*2 hrs*	Work		2h				
	Update exam record for ‘activating exam’	*2 hrs*	Work		2h				
	Select exam questions with examID as parameter	*2 hrs*	Work		2h				
	Update studentResponse table with student response to question	*2 hrs*	Work		2h				
	Select studentResponses passing examID and studentID as parameters returning a grade for th	*2 hrs*	Work		2h				
	Select studentResponses passing examID as parameter summing grades for the entire exam	*2 hrs*	Work		2h				
	Database utility component for database access	*8 hrs*	Work		8h				
	Unit testing	*16 hrs*	Work		16h				
	Integration testing	*16 hrs*	Work		16h				
	Beta testing	*16 hrs*	Work		16h				
	DataBase Configuration	*8 hrs*	Work			8h			

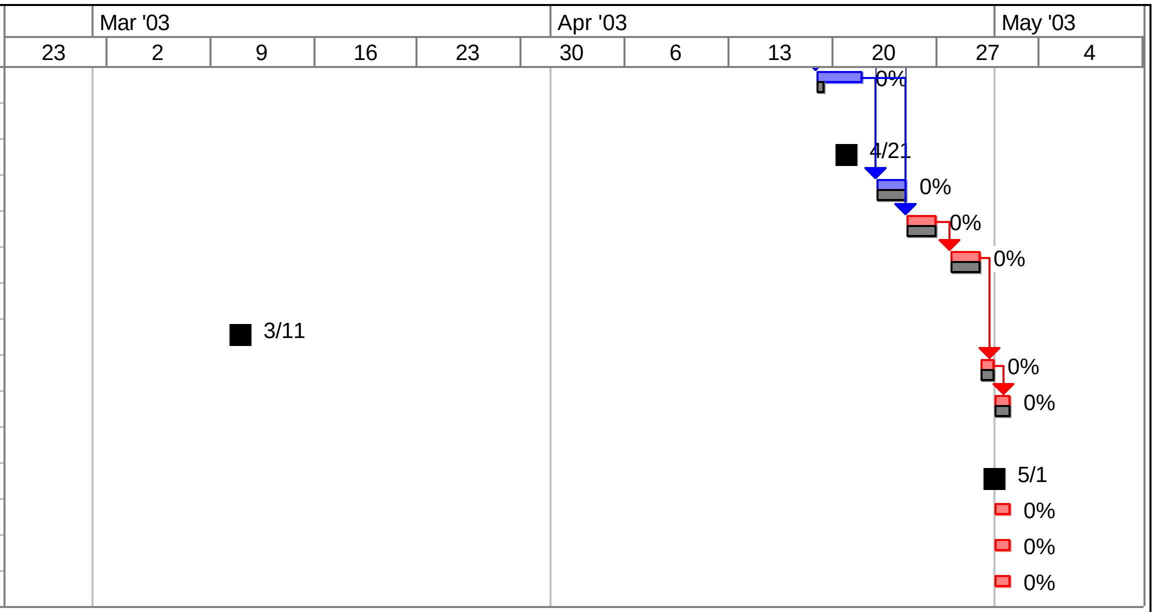
ID	Task Name	Duration
1	Documentation and Analysis	0 days
2	Software system specification	1 day
3	Software requirements specification	1 day
4	Preliminary software design specification	1 day
5	User interface design	1 day
6	Test plan	1 day
7		
8	Database task breakdown	0 days
9	Create database table statements	2 days
10	Stored procedure / embedded query construction	0 days
11	Insert new user	2 hrs
12	Select user with username / password as parameters	2 hrs
13	Insert new exam	2 hrs
14	Insert new exam question	2 hrs
15	Update exam record for ‘activating exam’	2 hrs
16	Select exam questions with examID as parameter	2 hrs
17	Update studentResponse table with student response to	2 hrs
18	Select studentResponses passing examID and studentID	2 hrs
19	Select studentResponses passing examID as parameter s	2 hrs
20		
21	Code component / UI task breakdown	0 days
22	Database utility component for database access	1 day
23	Create ‘new user’ JSP page / component	4 hrs
24	Create ‘log into system’ JSP page / component	4 hrs
25	Create ‘new exam’ JSP page / component	4 hrs
26	Create ‘enter new question’ JSP page / component	4 hrs
27	Create ‘activate exam’ JSP page / component	4 hrs
28	Create ‘display exam question’ JSP page / component	2 days



Project: TestEngineProjectPlan
Date: Mon 4/7/03

Critical	🟥	Baseline	▬	Project Summary	▬
Critical Split	▭	Baseline Split	▭	External Tasks	▭
Critical Progress	🟥	Baseline Milestone	◆	External Milestone	◆
Task	🟦	Milestone	■	Deadline	⇧
Split	▭	Summary Progress	▭		
Task Progress	🟦	Summary	▬		

ID	Task Name	Duration
29	Create ‘exam results’ JSP page / component	2 days
30		
31	Testing task breakdown	0 days
32	Unit testing	2 days
33	Integration testing	2 days
34	Beta testing	2 days
35		
36	Help task breakdown	0 days
37	User guide on-line html manual	1 day
38	Frequently asked questions on-line html manual	1 day
39		
40	Configuration task breakdown	0 days
41	Web Server Configuration	1 day
42	DataBase Configuration	1 day
43	Client Configuration	1 day



Project: TestEngineProjectPlan
Date: Mon 4/7/03

Critical	🟥	Baseline	▬	Project Summary	▬
Critical Split	▬	Baseline Split	▬	External Tasks	▬
Critical Progress	▬	Baseline Milestone	◆	External Milestone	◆
Task	🟦	Milestone	■	Deadline	⇩
Split	▬	Summary Progress	▬		
Task Progress	▬	Summary	▬		

5.0 Staff Organization

5.1 Team structure

AR

- Engine Designer
- Interface Designer
- Documentation
- Overall Testing and Report

F H

- Data Modeling
- Database Design
- Documentation
- Overall Testing and Report

JS

- Engine Designer
- Interface Designer
- Documentation
- Overall Testing and Report

5.2 Management Reporting and Communication

Project progress is communicated via e-mail. Once a week we meets together to update progress. All files sent to other members are done via email.

6.0 Tracking and Control Mechanisms

6.1 Quality Assurance and Control

Software Quality Assurance is any activity aimed at evaluating an attribute or capability of a program or a system and determining that it meets its required results. Delivering high quality software for use in today's world is significant challenge. As user demand increased functionality and guaranteed compatibility with the latest software, hardware, and operating systems, it becomes increasingly difficult to deliver software on time. Software quality assurance is the key to success.
The SQA team's to ensure that the product does works like original design specifications. If they found any errors, the SQA team will notify the development team to correct the errors. Also, the SQA team will perform a walkthrough to analyze the product's quality at any particular stage of development. Error detection and possible enhancements are also expressed to the development team. The SQA organizational role is to review the product at specific times during implementation. The SQA team will directly interact with the software developer team in group discussions, error discussions and possible enhancements that have been identified.

6.2 Change Management and Control

The main goal of SCM is to recognize substantial changes within software. It makes sure the modifications are managed efficiently and effectively. The SCM are also held responsible for reporting the changes to those employees who are affected by the amendment.

The focus of the SCM is to make crucial changes without creating a big impact on the entire system. It minimal changes, it is easier to do backtracking which saves time and result in better and higher-capacity result.

The SCM team will carefully look into these matters and any changes to the existing code or architectural design must be inspected before it is implemented. To do that, they closely work with the SQA department. The role of the SQA is to test many documents using the software and report any complications they face and suggest changes to the Software Engineers. The software engineers then reports SQA's results to the SCM team. The SCM will further review the SQA test results and their finding and make suggestions to their team leader. The team leader will carefully review each idea and make a decision that he/she feels are crucial. The SCM leader keeps all suggestions on file whether they are approved or denied