CS 413 - Software Engineering Project Management
2018-2019 Spring

“Patient Tracking System”

Software Project Management Plan

Team #13

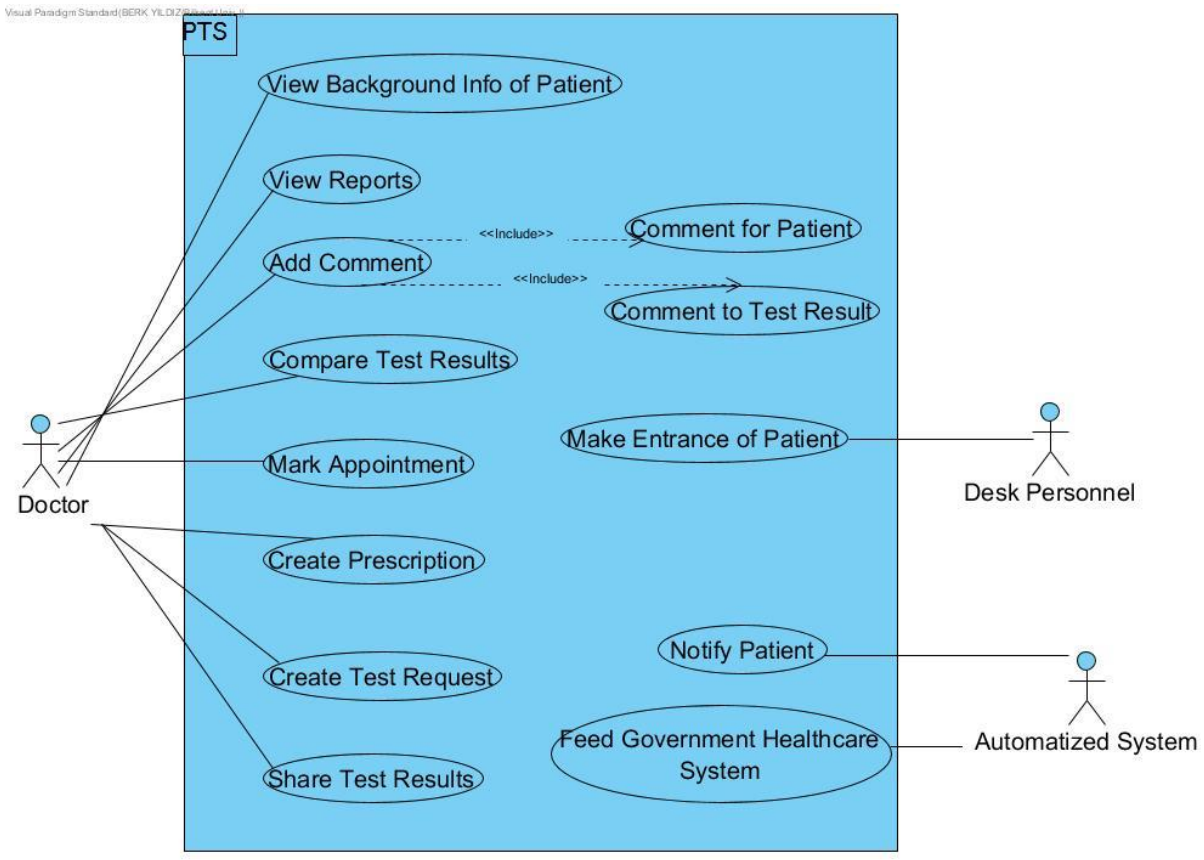
Berk Yıldız	21502040
Mastan Abdulkhaligli	21403007

İçindekiler

1. Estimation Plan

1.1. Use Case Points (UCP)

1.1.1. Use Case Diagram



1.1.2. Unadjusted Use Case Weight (UUCW)

Use Case Classification	No. of Transactions	Weight
Simple	1 to 3 transactions	5
Average	4 to 7 transactions	10
Complex	8 or more transactions	15

Table 1: UUCF Classification [1]

View Background Info of Patient	Simple
View Report	Simple
Add Comment for Patient	Simple
Add Comment to Test Result	Average
Compare Test Results	Average
Mark Appointment	Simple
Create Prescription	Simple
Create Test Request	Simple
Share Test Result	Average
Make Entrance of Patient	Simple
Notify Patient	Simple
Feed Government	Average

UUCW = (Total No. of Simple Use Cases x 5) + (Total No. Average Use Cases x 10) + (Total No. Complex Use Cases x 15)

For PTS, **UUCW = (8 x 5) + (4 x 10) = 80**

1.1.3. Unadjusted Actor Weight (UAW)

Actor Classification	Type of Actor	Weight
Simple	External system that must interact with the system using a well-defined API	1
Average	External system that must interact with the system using standard communication protocols (e.g. TCP/IP, FTP, HTTP, database)	2
Complex	Human actor using a GUI application interface	3

Table 2: UAW Classification [1]

UAW = (Total No. of Simple actors x 1) + (Total No. Average actors x 2) + (Total No. Complex actors x 3)

For PTS there are two complex actors: Doctor and Desk Personnel

One average actor: Automatized System

So, **UAW = (0 x 1) + (1 x 2) + (2 x 3) = 8**

1.1.4. Technical Complexity Factor (TCF)

Factor	**Description**	**Weight**	**Assigned Value**	**Weight x Assigned Value**
T1	Distributed system	2.0	2	4
T2	Response time/performance objectives	1.0	5	5
T3	End-user efficiency	1.0	5	5
T4	Internal processing complexity	1.0	1	1
T5	Code reusability	1.0	2	2
T6	Easy to install	0.5	5	2.5
T7	Easy to use	0.5	5	2.5
T8	Portability to other platforms	2.0	2	4
T9	System maintenance	1.0	2	2
T10	Concurrent/parallel processing	1.0	2	2
T11	Security features	1.0	4	4
T12	Access for third parties	1.0	1	1
T13	End user training	1.0	1	1
TOTAL			**36**	

Table 3: TCF

TCF = 0.6 + (TF/100) = 0.6 + (36/100) = 0.96

1.1.5. Environmental Complexity Factor (ECF)

Factor	Description	Weight	Assigned Value	Weight x Assigned Value
E1	Familiarity with development process used	1.5	2	3
E2	Application experience	0.5	1	0.5
E3	Object-oriented experience of team	1.0	4	4
E4	Lead analyst capability	0.5	3	1.5
E5	Motivation of the team	1.0	4	4
E6	Stability of requirements	2.0	1	2
E7	Part-time staff	-1.0	0	0
E8	Difficult programming language	-1.0	3	-3
			Total (EF):	**11**

Table 4: ECF [1]

ECF = 1.4 + (-0.03 x EF) = 1.4 + (-0.03 x 11) = 1.07
So, ECF of PTS is 1.07

1.1.6. Use Case Points Calculations

UCP = (UUCW + UAW) x TCF x ECF
For PTS, UCP = (80 + 8) x 0.96 x 1.07 = 90.39
UCP = 90.39
Suppose effort required is 10 Staff-Hours per UCP, then
Estimated Effort = 10 x 90.39 = 903.9 hours

1.2. Unadjusted Function Points (UFP)

<u>External Inputs (EIs):</u>

- Username input
- Password input
- Patient name input
- Patient surname input
- Patient birthday input
- Patient address input
- Patient mobile phone number input
- Patient e-mail address input
- Patient comment input
- Patient chronic diseases input
- Patient past diseases input
- Report or test result comment input
- Prescription input
- Patient entrance input

Number of EIs = 14

All external inputs will be considered as <u>simple</u>.

<u>External Outputs (EOs):</u>

- Test result & report display ***(simple)***
- Test comparison display ***(average)***
- Patient personal & background info display ***(simple)***
- Comment display ***(simple)***
- Prescription display ***(simple)***

Number of EOs = 5

<u>External Inquiries (EQs):</u>

- Display patient list
- Display test & result list

Number EQs = 2

All external inquiries will be considered as <u>simple</u>.

<u>Internal Logical Files (ILFs):</u>

- Database Configuration File
- Local Network Manager

Number of ILFs = 2

All internal logical files will be considered as <u>average</u>

External Interface Files (EIFs):

- Medical imaging and archiving ***(complex)***

Number of EIF = 1

		Weighting Factor			**Count**
		Simple	**Average**	**Complex**	
EIs	Username input	3			42
	Password input	3			
	Patient name input	3			
	Patient surname input	3			
	Patient birthday input	3			
	Patient address input	3			
	Patient mobile phone number input	3			
	Patient e-mail address input	3			
	Patient comment input	3			
	Patient chronic diseases input	3			
	Patient past diseases input	3			
	Report or test result comment input	3			
	Prescription input	3			
	Patient Entrance Input	3			
EOs	Test result & report display	4			21
	Test comparison display		5		
	Patient personal & background info display	4			
	Comment display	4			
	Prescription display	4			
EQs	Display patient list	3			6
	Display test & result list	3			
ILFs	Database Configuration File		10		20
	Local Network Manager		10		
EIFs	Medical imaging and archiving			10	10
TOTAL UFP					**99**

Table 5: Unadjusted Function Point Calculation

Number	Complexity Weighting Factor	Value
1	Backup and recovery	5
2	Data communications	4
3	Distributed processing	2
4	Performance critical	5
5	Existing operating environment	1
6	On-line data entry	2
7	Input transaction over multiple screens	1
8	Master files updated online	2
9	Information domain values complex	1
10	Internal processing complex	1
11	Code designed for reuse	2
12	Conversion/installation in design	4
13	Multiple installations	4
14	Application designed for change	2
	Total complexity adjustment value	**36**

Table 6: Complexity Adjustment Value Calculation [2]

Total Unadjusted Function Points (UFP) = 110

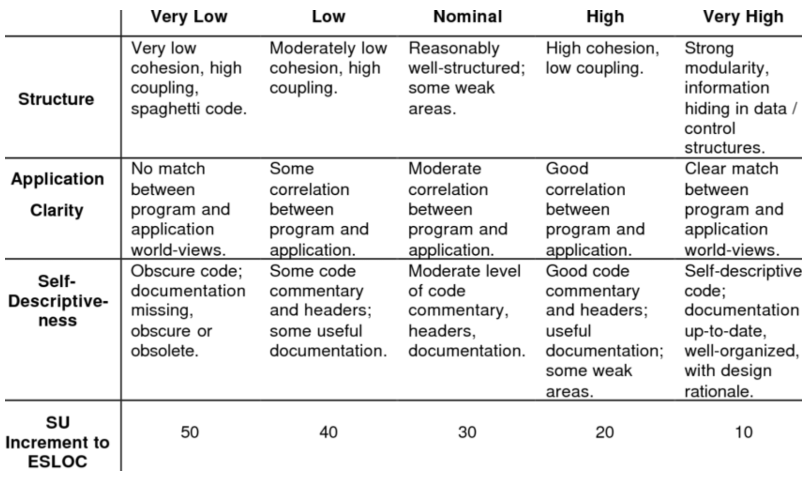
Product Complexity Adjustment (PC) = 0.65 + (0.01 * 36) = 1.01

Total Adjusted Function Points (FP) = UFP * PC = 111.1

Language Factor (LF) for C++ = 50

Source Lines of Code (SLOC) = FP * LF = 5555

1.3. Line of Code (LOC) Based on COCOMO Re-Use Model



	Very Low	Low	Nominal	High	Very High
Structure	Very low cohesion, high coupling, spaghetti code.	Moderately low cohesion, high coupling.	Reasonably well-structured; some weak areas.	High cohesion, low coupling.	Strong modularity, information hiding in data / control structures.
Application Clarity	No match between program and application world-views.	Some correlation between program and application.	Moderate correlation between program and application.	Good correlation between program and application.	Clear match between program and application world-views.
Self-Descriptive-ness	Obscure code; documentation missing, obscure or obsolete.	Some code commentary and headers; some useful documentation.	Moderate level of code commentary, headers, documentation.	Good code commentary and headers; useful documentation; some weak areas.	Self-descriptive code; documentation up-to-date, well-organized, with design rationale.
SU Increment to ESLOC	50	40	30	20	10

Table 7: Rating Scale for Software Understanding Increment [3]

	Rating	**SU Increment to ESLOC**	**Definition**
Structure	Nominal	30	Reasonably well structured; some weak areas
Application Clarity	Low	40	Some correlation between program and application
Self Descriptiveness	Nominal	30	Moderate level of code commentary, headers, documentation.

Table 8: Rating Scale for Software Understanding Increment of PTS

Average SU = 100 / 3 = 33.3

Assesment and Assimilation (AA) Increment = 2 (Basic module search and documentation)

Unfamiliarity (UNFM) Increment = 0.4 (Somewhat familiar)

CM (Code Modified) = 20%

DM (Design Modified) = 55%

IM (Integration Required) = 40%

$$Equivalent\ KSLOC = Adapted\ \ KLOC * (1 - \frac{AT}{100}) * AAM$$

$$AAF\ = 0.4 * DM + 0.3 * CM + 0.3 * IM$$

$$AAM = \begin{cases} \dfrac{AA + AAF(1 + 0.02 * SU * UNFM)}{100}, for\ AAF \leq 50 \\ \dfrac{AA + AAF + SU * UNFM}{100}, for\ AAF > 50 \end{cases}$$

[4]

AAF = (0.4 * 55) + (0.3 * 20) + (0.3 * 40) = 40

AAM = (4 + 40 * (1 + (0.02 * 33.3 * 0.4))) / 100 = 0.54656

KSLOC = 0.54656 * 1000 = 546.56

Product Size = SLOC + KSLOC = 5555 + 547 = 6102

1.4. Effort and Duration Based on COCOMO Post-Architecture Effort Model

$$PM_{NS} = A \times Size^{E} \times \prod_{i=1}^{n} EM_i$$

$$\text{where } E = B + 0.01 \times \sum_{j=1}^{5} SF_j$$

$$TDEV_{NS} = C \times (PM_{NS})^{F}$$

$$\text{where } F = D + 0.2 \times 0.01 \times \sum_{j=1}^{5} SF_j$$

$$= D + 0.2 \times (E - B)$$

Driver	**Sym**	**VL**	**L**	**N**	**H**	**VH**	**XH**
PREC	SF_1	0.0405	0.0324	0.0243	0.0162	0.0081	0.00
FLEX	SF_2	0.0607	0.0486	0.0364	0.0243	0.0121	0.00
RESL	SF_3	0.0422	0.0338	0.0253	0.0169	0.0084	0.00
TEAM	SF_4	0.0494	0.0395	0.0297	0.0198	0.0099	0.00
PMAT	SF_5	0.0454	0.0364	0.0273	0.0182	0.0091	0.00
RELY	EM_1	0.75	0.88	1.00	1.15	1.39	
DATA	EM_2		0.93	1.00	1.09	1.19	
CPLX	EM_3	0.75	0.88	1.00	1.15	1.30	1.66
RUSE	EM_4		0.91	1.00	1.14	1.29	1.49
DOCU	EM_5	0.89	0.95	1.00	1.06	1.13	
TIME	EM_6			1.00	1.11	1.31	1.67
STOR	EM_7			1.00	1.06	1.21	1.57
PVOL	EM_8		0.87	1.00	1.15	1.30	
ACAP	EM_9	1.50	1.22	1.00	0.83	0.67	
PCAP	EM_{10}	1.37	1.16	1.00	0.87	0.74	
PCON	EM_{11}	1.24	1.10	1.00	0.92	0.84	
AEXP	EM_{12}	1.22	1.10	1.00	0.89	0.81	
PEXP	EM_{13}	1.25	1.12	1.00	0.88	0.81	
LTEX	EM_{14}	1.22	1.10	1.00	0.91	0.84	
TOOL	EM_{15}	1.24	1.12	1.00	0.86	0.72	
SITE	EM_{16}	1.25	1.10	1.00	0.92	0.84	0.78
SCED	EM_{17}	1.29	1.10	1.00	1.00	1.00	

Table 9: COCOMO II Cost Driver Values [5]

Driver	**Value**
RELY	0.88
DATA	1.09
CPLX	0.75
RUSE	1
DOCU	1.06
TIME	1.31
STOR	1.06
PVOL	0.87
ACAP	1.5
PCAP	0.74
PCON	1.10
AEXP	1
PEXP	1.25
LTEX	0.91
TOOL	1.12
SITE	0.78
SCED	1

Table 10: Cost Driver Values of PTS

EM = 0.88*1.09*0.75*1 *1.06*1.31*1.06*0.87*1.5*0.74*1.1*1*1.25*0.91*1.12*0.78*1

EM = 1.1177

E = PREC * FLEX * RESL * TEAM * PMAT

B = 0.91

E = 0.91 + 0.01 * (3.72 * 1.01 * 2.83 * 2.19 * 1.59) = 1.2802

A = 2.94   B = 0.91   C = 3.67   D = 0.28

PMns = A * (KSLOC^E) * EM = 2.94 * (6102^1.2802) * 1.1177 = 227 person months.

F = 0.28 + 0.2 * 0.01 * (3.72 + 1.01 + 2.83 + 2.19 + 1.59) = 0.30268

TDEVns = 3.67 * (227^0.30268) = 19 months

Avg Number of Contributor = PMns / TDEVns = 227/19 = 12

2. Project Monitoring

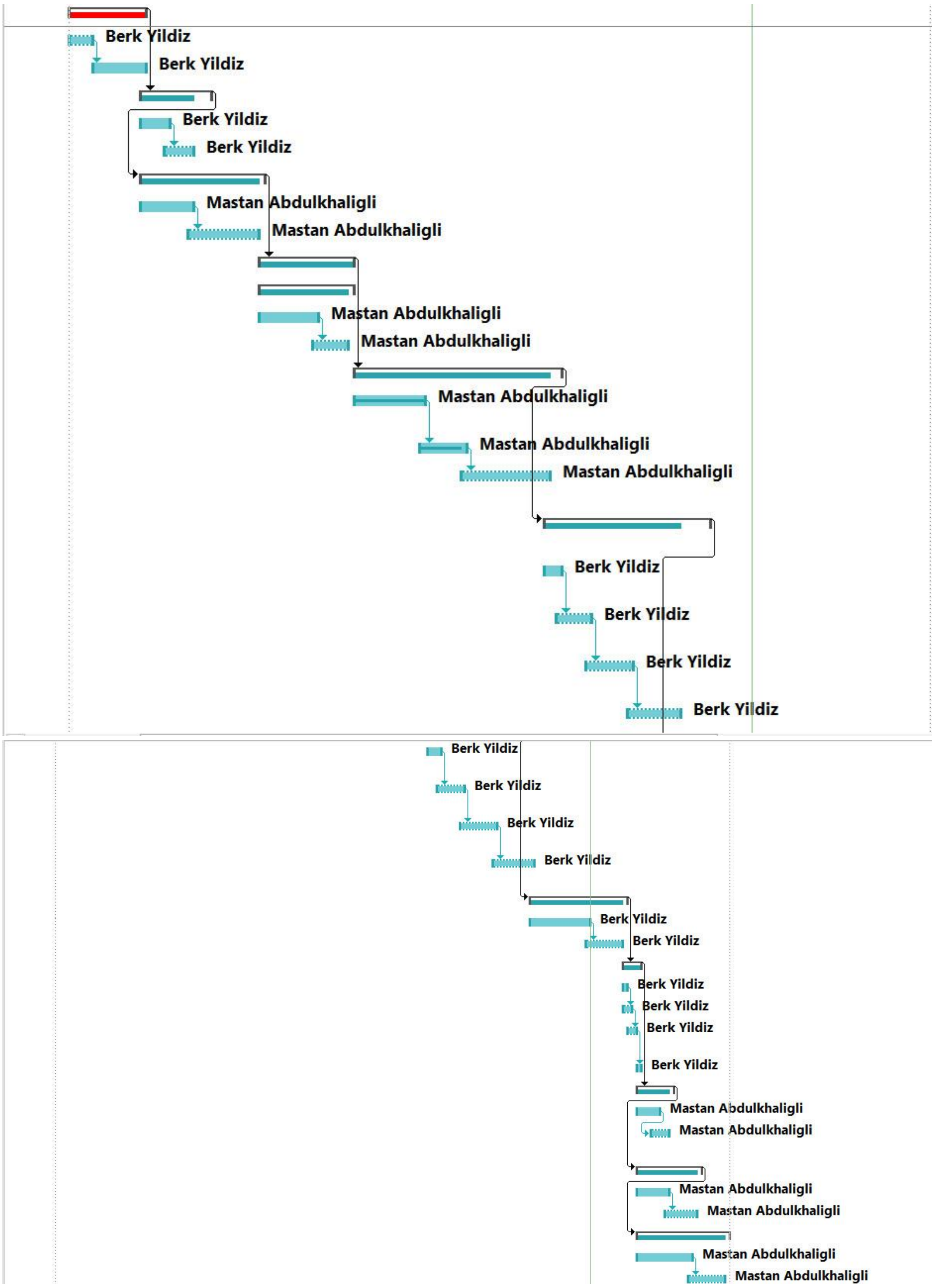
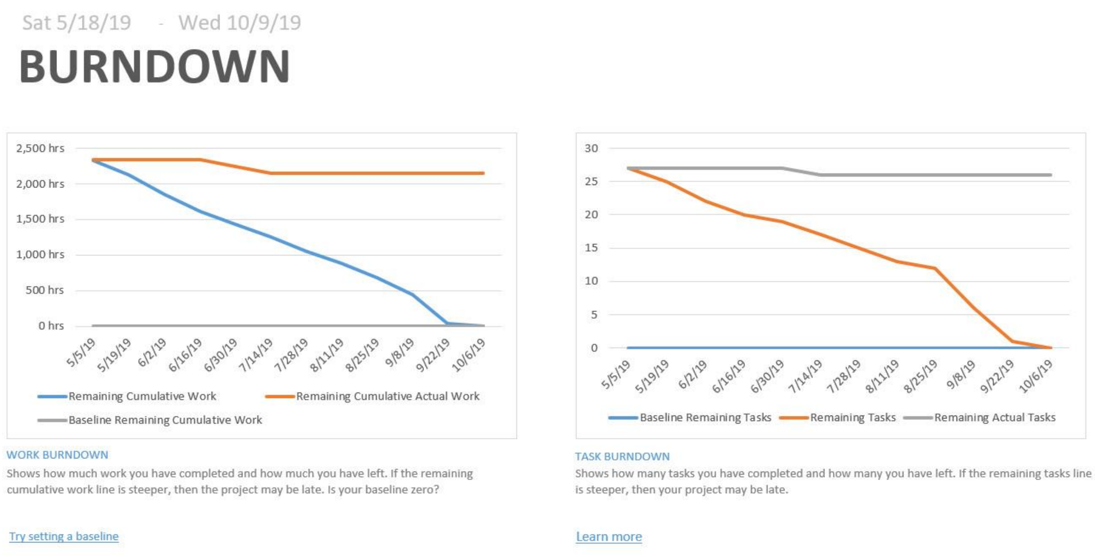


Figure 1: Gantt Chart



Sat 5/18/19 - Wed 10/9/19

BURNDOWN

WORK BURNDOWN

Shows how much work you have completed and how much you have left. If the remaining cumulative work line is steeper, then the project may be late. Is your baseline zero?

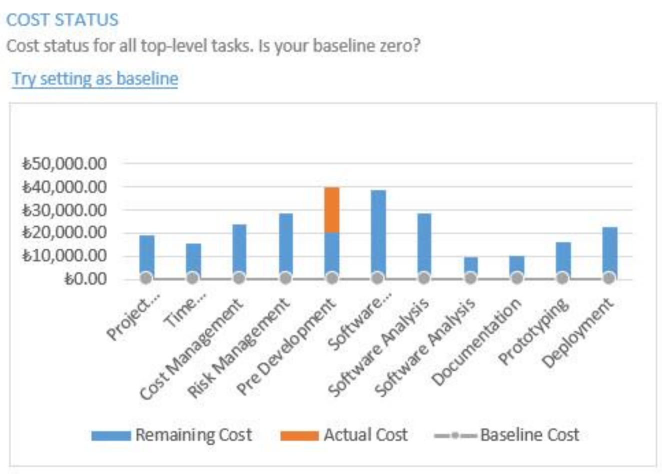
Try setting a baseline

TASK BURNDOWN

Shows how many tasks you have completed and how many you have left. If the remaining tasks line is steeper, then your project may be late.

Learn more

Figure 2: Burndown Chart

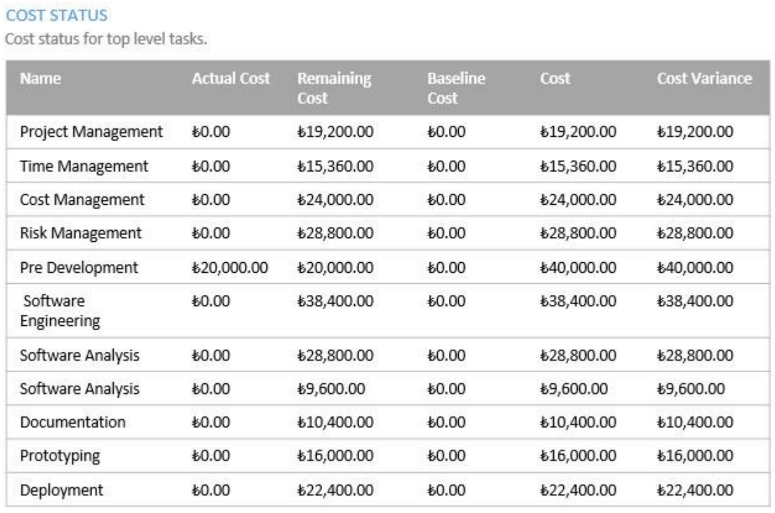


COST STATUS

Cost status for all top-level tasks. Is your baseline zero?

Try setting as baseline

Figure 3: Cost Status

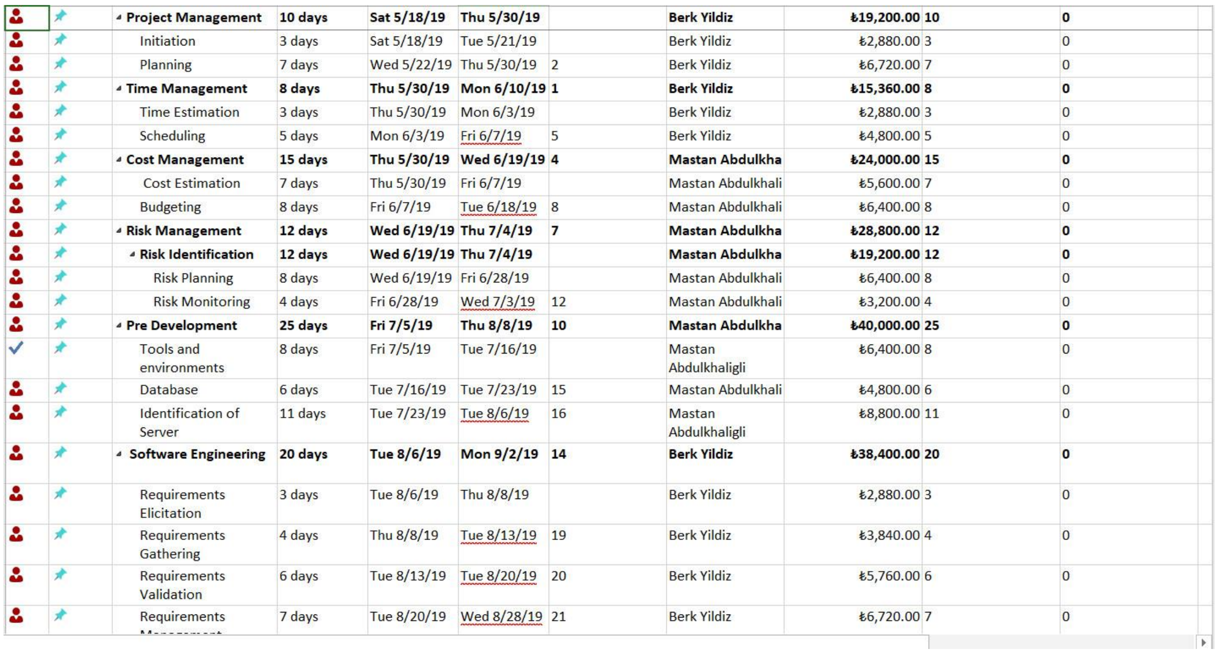


COST STATUS

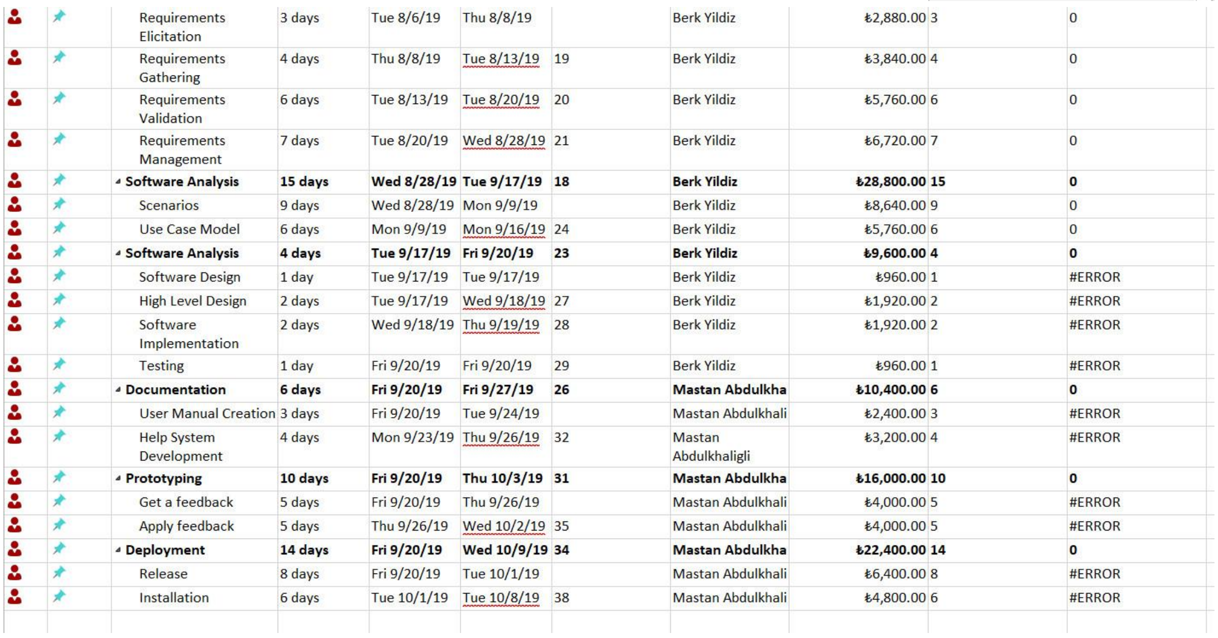
Cost status for top level tasks.

Name	Actual Cost	Remaining Cost	Baseline Cost	Cost	Cost Variance
Project Management	฿0.00	฿19,200.00	฿0.00	฿19,200.00	฿19,200.00
Time Management	฿0.00	฿15,360.00	฿0.00	฿15,360.00	฿15,360.00
Cost Management	฿0.00	฿24,000.00	฿0.00	฿24,000.00	฿24,000.00
Risk Management	฿0.00	฿28,800.00	฿0.00	฿28,800.00	฿28,800.00
Pre Development	฿20,000.00	฿20,000.00	฿0.00	฿40,000.00	฿40,000.00
Software Engineering	฿0.00	฿38,400.00	฿0.00	฿38,400.00	฿38,400.00
Software Analysis	฿0.00	฿28,800.00	฿0.00	฿28,800.00	฿28,800.00
Software Analysis	฿0.00	฿9,600.00	฿0.00	฿9,600.00	฿9,600.00
Documentation	฿0.00	฿10,400.00	฿0.00	฿10,400.00	฿10,400.00
Prototyping	฿0.00	฿16,000.00	฿0.00	฿16,000.00	฿16,000.00
Deployment	฿0.00	฿22,400.00	฿0.00	฿22,400.00	฿22,400.00

Figure 4: Cost status for top level tasks



👤	📌	**Project Management**	**10 days**	**Sat 5/18/19**	**Thu 5/30/19**		**Berk Yildiz**	**₺19,200.00**	**10**	**0**
👤	📌	Initiation	3 days	Sat 5/18/19	Tue 5/21/19		Berk Yildiz	₺2,880.00	3	0
👤	📌	Planning	7 days	Wed 5/22/19	Thu 5/30/19	2	Berk Yildiz	₺6,720.00	7	0
👤	📌	**Time Management**	**8 days**	**Thu 5/30/19**	**Mon 6/10/19**	**1**	**Berk Yildiz**	**₺15,360.00**	**8**	**0**
👤	📌	Time Estimation	3 days	Thu 5/30/19	Mon 6/3/19		Berk Yildiz	₺2,880.00	3	0
👤	📌	Scheduling	5 days	Mon 6/3/19	Fri 6/7/19	5	Berk Yildiz	₺4,800.00	5	0
👤	📌	**Cost Management**	**15 days**	**Thu 5/30/19**	**Wed 6/19/19**	**4**	**Mastan Abdulkha**	**₺24,000.00**	**15**	**0**
👤	📌	Cost Estimation	7 days	Thu 5/30/19	Fri 6/7/19		Mastan Abdulkhali	₺5,600.00	7	0
👤	📌	Budgeting	8 days	Fri 6/7/19	Tue 6/18/19	8	Mastan Abdulkhali	₺6,400.00	8	0
👤	📌	**Risk Management**	**12 days**	**Wed 6/19/19**	**Thu 7/4/19**	**7**	**Mastan Abdulkha**	**₺28,800.00**	**12**	**0**
👤	📌	**Risk Identification**	**12 days**	**Wed 6/19/19**	**Thu 7/4/19**		**Mastan Abdulkha**	**₺19,200.00**	**12**	**0**
👤	📌	Risk Planning	8 days	Wed 6/19/19	Fri 6/28/19		Mastan Abdulkhali	₺6,400.00	8	0
👤	📌	Risk Monitoring	4 days	Fri 6/28/19	Wed 7/3/19	12	Mastan Abdulkhali	₺3,200.00	4	0
👤	📌	**Pre Development**	**25 days**	**Fri 7/5/19**	**Thu 8/8/19**	**10**	**Mastan Abdulkha**	**₺40,000.00**	**25**	**0**
✓	📌	Tools and environments	8 days	Fri 7/5/19	Tue 7/16/19		Mastan Abdulkhaligli	₺6,400.00	8	0
👤	📌	Database	6 days	Tue 7/16/19	Tue 7/23/19	15	Mastan Abdulkhali	₺4,800.00	6	0
👤	📌	Identification of Server	11 days	Tue 7/23/19	Tue 8/6/19	16	Mastan Abdulkhaligli	₺8,800.00	11	0
👤	📌	**Software Engineering**	**20 days**	**Tue 8/6/19**	**Mon 9/2/19**	**14**	**Berk Yildiz**	**₺38,400.00**	**20**	**0**
👤	📌	Requirements Elicitation	3 days	Tue 8/6/19	Thu 8/8/19		Berk Yildiz	₺2,880.00	3	0
👤	📌	Requirements Gathering	4 days	Thu 8/8/19	Tue 8/13/19	19	Berk Yildiz	₺3,840.00	4	0
👤	📌	Requirements Validation	6 days	Tue 8/13/19	Tue 8/20/19	20	Berk Yildiz	₺5,760.00	6	0
👤	📌	Requirements	7 days	Tue 8/20/19	Wed 8/28/19	21	Berk Yildiz	₺6,720.00	7	0



👤	📌	Requirements Elicitation	3 days	Tue 8/6/19	Thu 8/8/19		Berk Yildiz	₺2,880.00	3	0
👤	📌	Requirements Gathering	4 days	Thu 8/8/19	Tue 8/13/19	19	Berk Yildiz	₺3,840.00	4	0
👤	📌	Requirements Validation	6 days	Tue 8/13/19	Tue 8/20/19	20	Berk Yildiz	₺5,760.00	6	0
👤	📌	Requirements Management	7 days	Tue 8/20/19	Wed 8/28/19	21	Berk Yildiz	₺6,720.00	7	0
👤	📌	**Software Analysis**	**15 days**	**Wed 8/28/19**	**Tue 9/17/19**	**18**	**Berk Yildiz**	**₺28,800.00**	**15**	**0**
👤	📌	Scenarios	9 days	Wed 8/28/19	Mon 9/9/19		Berk Yildiz	₺8,640.00	9	0
👤	📌	Use Case Model	6 days	Mon 9/9/19	Mon 9/16/19	24	Berk Yildiz	₺5,760.00	6	0
👤	📌	**Software Analysis**	**4 days**	**Tue 9/17/19**	**Fri 9/20/19**	**23**	**Berk Yildiz**	**₺9,600.00**	**4**	**0**
👤	📌	Software Design	1 day	Tue 9/17/19	Tue 9/17/19		Berk Yildiz	₺960.00	1	#ERROR
👤	📌	High Level Design	2 days	Tue 9/17/19	Wed 9/18/19	27	Berk Yildiz	₺1,920.00	2	#ERROR
👤	📌	Software Implementation	2 days	Wed 9/18/19	Thu 9/19/19	28	Berk Yildiz	₺1,920.00	2	#ERROR
👤	📌	Testing	1 day	Fri 9/20/19	Fri 9/20/19	29	Berk Yildiz	₺960.00	1	#ERROR
👤	📌	**Documentation**	**6 days**	**Fri 9/20/19**	**Fri 9/27/19**	**26**	**Mastan Abdulkha**	**₺10,400.00**	**6**	**0**
👤	📌	User Manual Creation	3 days	Fri 9/20/19	Tue 9/24/19		Mastan Abdulkhali	₺2,400.00	3	#ERROR
👤	📌	Help System Development	4 days	Mon 9/23/19	Thu 9/26/19	32	Mastan Abdulkhaligli	₺3,200.00	4	#ERROR
👤	📌	**Prototyping**	**10 days**	**Fri 9/20/19**	**Thu 10/3/19**	**31**	**Mastan Abdulkha**	**₺16,000.00**	**10**	**0**
👤	📌	Get a feedback	5 days	Fri 9/20/19	Thu 9/26/19		Mastan Abdulkhali	₺4,000.00	5	#ERROR
👤	📌	Apply feedback	5 days	Thu 9/26/19	Wed 10/2/19	35	Mastan Abdulkhali	₺4,000.00	5	#ERROR
👤	📌	**Deployment**	**14 days**	**Fri 9/20/19**	**Wed 10/9/19**	**34**	**Mastan Abdulkha**	**₺22,400.00**	**14**	**0**
👤	📌	Release	8 days	Fri 9/20/19	Tue 10/1/19		Mastan Abdulkhali	₺6,400.00	8	#ERROR
👤	📌	Installation	6 days	Tue 10/1/19	Tue 10/8/19	38	Mastan Abdulkhali	₺4,800.00	6	#ERROR

Figure 5: WBS

PROGRESS VERSUS COST

Progress made versus the cost spent over time. If % Complete line below the cumulative cost line,your project may be over budget.

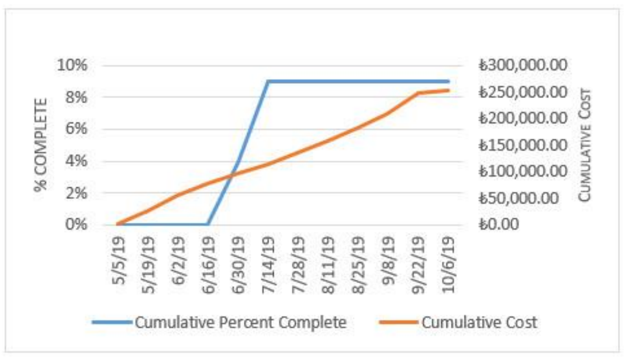
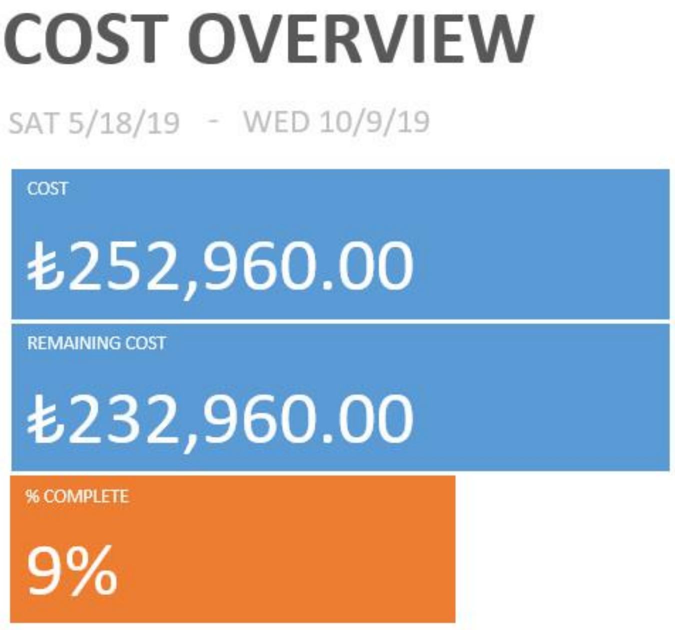


Figure 6: Progress Versus Cost



COST OVERVIEW

SAT 5/18/19 - WED 10/9/19

COST

₺252,960.00

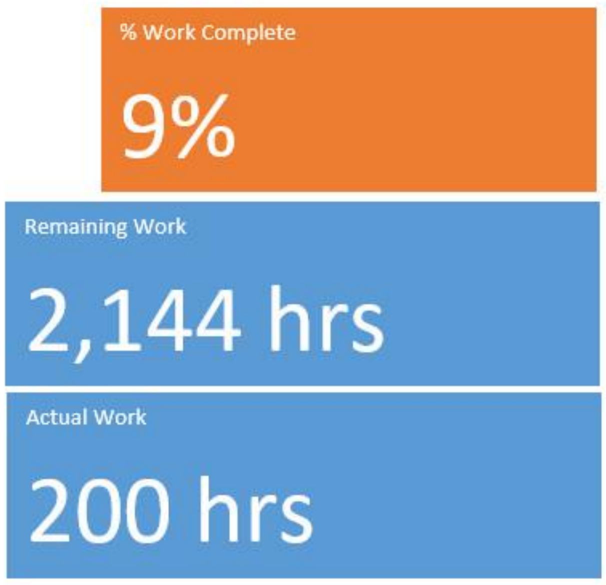
REMAINING COST

₺232,960.00

% COMPLETE

9%

Figure 7: Cost Overview



% Work Complete

9%

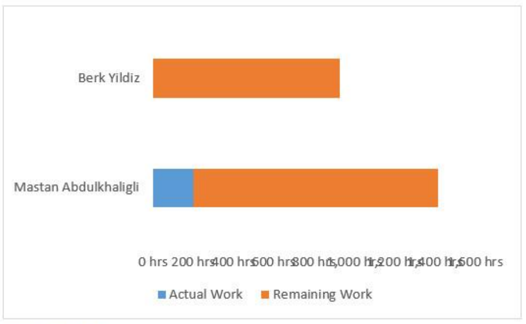
Remaining Work

2,144 hrs

Actual Work

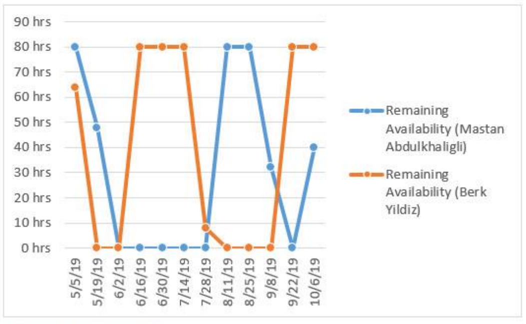
200 hrs

Figure 8: Work Process



RESOURCE STATS

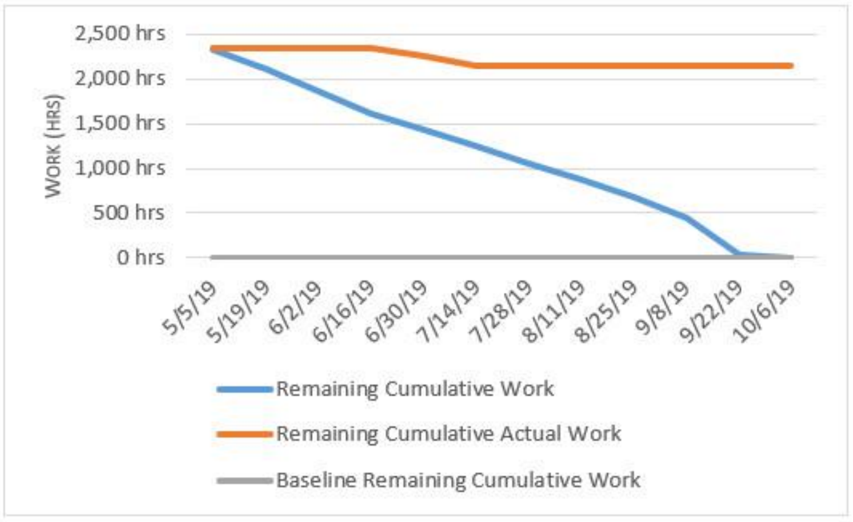
Shows work stats for all your resources.

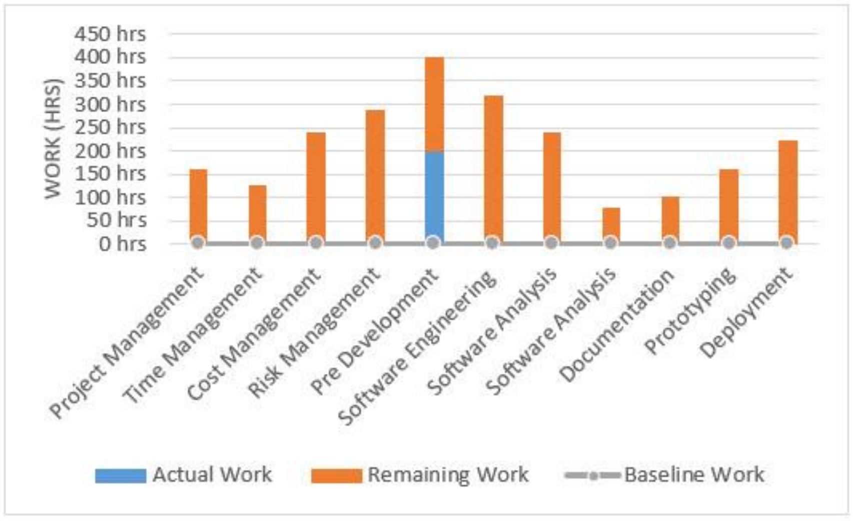


REMAINING AVAILABILITY

Shows remaining availability for all work resources.

Figure 9: Human Resource





WORK STATS

Shows work stats for all top level tasks.

Figure 10: Work Stats

3. Risk Management Plan

Even the most carefully prepared and planned projects can face problems. It does not matter how you prepared your project, projects always can encounter unexpected problems. For example, team member can be sick or quit, project manager can not catch deadline, weather condition is to harsh and team member cannot come to meeting, financial problems and so on. All of these does not mean project manager is helpless. Project manager can use risk planning to identify potential problems that can cause trouble for our project. Project manager analyze them, identify how likely they occur and take action to prevent these risks.

A risk is unexpected and uncertain events. Most of the time all risks are negative but some time they can be seen as opportunity. When Project manager is planning project, risks are still uncertain. Because they have not happened yet. But when risks happen, there are 4 basic ways to handle with them.

1. ***Avoid*** : The best thing for PM is avoid from risks. If PM can not prevent it the best way is avoid from these risks. But of course, this is not case for all projects.
2. ***Mitigate:*** If PM cannot avoid risk, second best way is mitigating this risk. It means that, PM should take action which decrease the damage of the risk.
3. ***Transfer:*** The another effective way to deal with risk is to pay someone which will take care of these risk for you. It known as insurance.
4. ***Accept:*** If PM cannot avoid, mitigate or transfer risk the best method is accepting the risk. Because PM already search for alternatives and knows about problem. PM also know what will happen when it occurs. If these tree basic ways is not a solution the best way is accepting the risk.

3.1. Risk Management Process

Risk assessments and mitigation strategy are part of managing risks. Risk assessments consist of identification of problem and evaluation of the potential impact of risk on project. Mitigation plan is designed for decrease impacts of risk or eliminate this risk absolutely. Team should prepare list of everything than can cause problem.

3.1.1. Risk Identification

For PM, more disciplined way is to crate checklist. Checklist which involve all potential risks and when these risks can be happen. Some old companies and industries develop risk checklist based on their last experience. These checklists are very helpful for team to identify risks and thinking very carefully. Identifying source of risks and then categorizing them can be helpful.

1. Technical
2. Cost
3. Schedule
4. Client
5. Contractual

6. Weather
7. Financial
8. Political
9. Environmental
10. People

PM can use same framework as the work breakdown structure(WBS) for developing **risk breakdown structure (RBS).**

3.1.2. Risk Evaluation

After the potential risks have been identified, project team evaluate each risk based on their probability that can risk happen. Of course, not all the risks are equal, some of them are more dangerous some of them is tiny. Some of the risks are more likely to happen some of them not. Evaluating the risk for probability of occurrence and the severity or the potential loss to the project is next step in the risk management process.

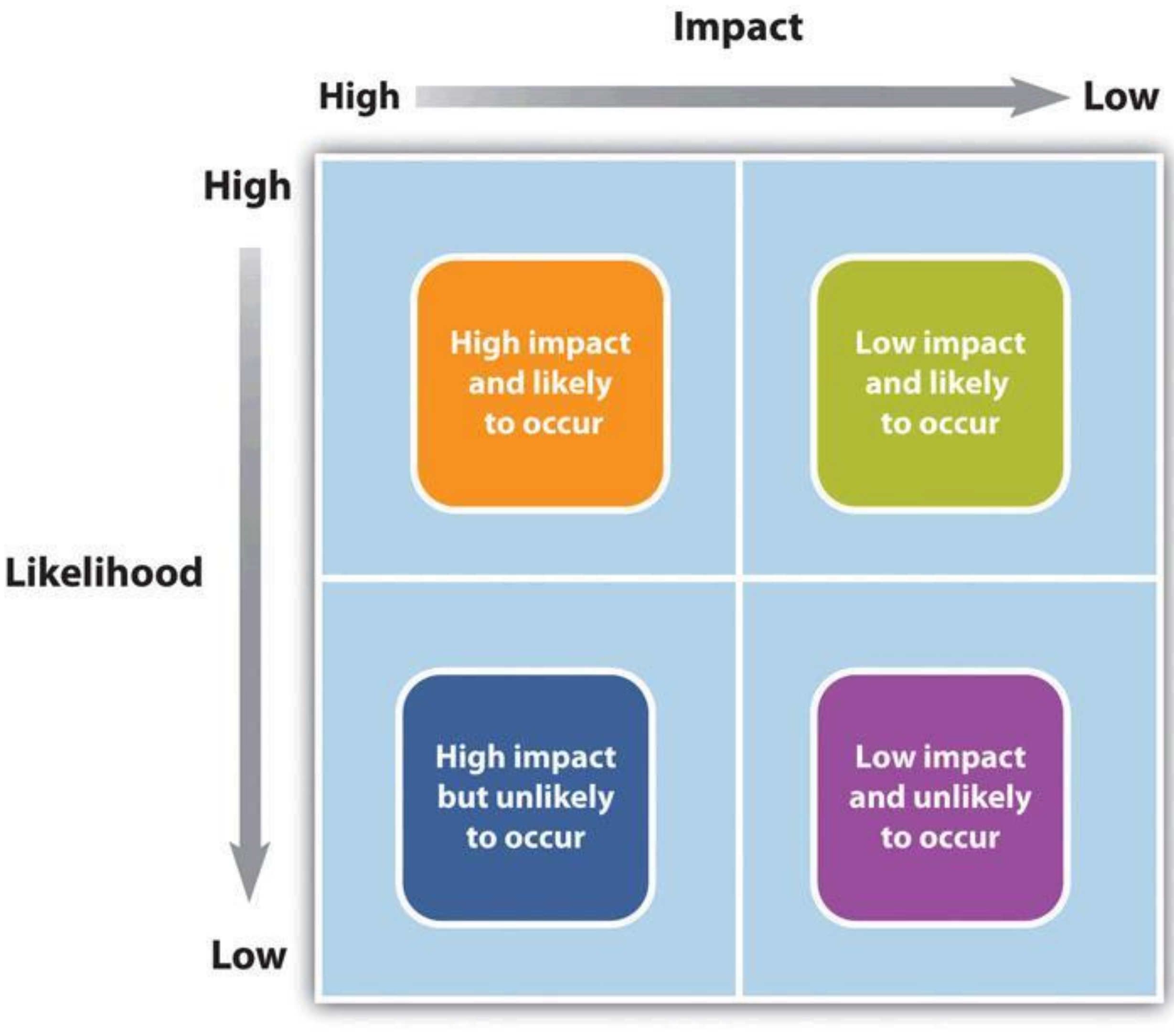

Impact
High
Low
High
Likelihood
Low
High impact and likely to occur
Low impact and likely to occur
High impact but unlikely to occur
Low impact and unlikely to occur

Table 11: Risk Evaluation

Having criteria to determine high-impact risks can help narrow the focus on a few critical risks that require mitigation. For example, suppose high-impact risks are those that could increase the project costs by 5% of the conceptual budget or 2% of the detailed budget. Only a few potential risk events meet these criteria.

3.1.3. Risk Mitigation

After risks are identified and categorized, team develops a risk mitigation plan. This is a plan that which mitigates or eliminate unexpected events. The project teams can mitigate risks in a various ways. For example:

1. Risk Avoidance
2. Risk sharing
3. Risk reduction
4. Risk transfer

All of these mitigation techniques can be an effective tool in reducing and eliminating individual risks and risk profile of the objects.

3.1.4. Contingency Plan

Project team often develops several alternative methods to get a goal and finish project. When problems and risks occurs it can affect goal of the team. These pans are called contingency plan.

4. Metrics Collection Plan

Without measure we can not talk or give any data about objects length or speed. We can not determine how far we have come or how many miles we got. In project management it is impossible to know project success or not without measuring it.

Today growth of data gives us a chance to search in deep and has prompted organizations to increasingly rely on metrics for different purpose. Such as business operations including project management. Metrics help to management in understanding capabilities and facilitate more advanced planning for production, service delivery and product development. When time comes to control cost, improve quality and identification of precious industry trend metrics help us.

To determining success of a project and support project manager to evaluate a project’s status, foresee risks and asses team productivity and quality of work project management metrics comes to help. There are a lot of reasons for implementing metrics as a tool.

4.1. Creating Good Metrics

Every project manager has different objectives, that is why a metrics which works for one business is not suitable for another business. It is important that designing good metrics and determining unique needs for business. There are some additional tips for being a successful.

- Keep definitions of KPIs and metrics simple and useful
- Plan for the long term, including setting things up for comparison to competitors
- Get senior management's buy-in
- Ensure that underlying data is both available and reachable
- Communicate the need to gather metrics, along with your goals and purpose
- Be sure you have the enough budget and stuff

4.2. Useful Metrics For Project Management

Every Project managements have different have different objectives and complexity and their metrics defined based on these. There are 5 most important metrics which covers most important measurements.

1. Productivity- This type of metric allows project managers to use the utilization of resources.
2. Scope of Work- To keep track of change request is necessary and it is important control and keep project on time, on budget.
3. Quality Satisfaction- This type of metrics is directly customer focused metric. Identification of defects early can prevent project from falling and focus.
4. Cost- Cost of the project have to managed well and it is critical to projects success. Cost management is related to other variable such as quality, scope and productivity. These varies below projections, the project will suffer.
5. Gross Margin-The main purpose of project is to increase or contribute the organization profits. What is gross margin? Gross margin is the Total Income Achieved – Total Cost Spent. A project should have a target gross margin in the planning stages.

4.3. Types of Measures

- Outcome Measures-The high level quality outcomes you are aiming at improving
- Process Measures- Specific, frequent, steps in a process that positively or negatively effect an outcome
- Balance Measures-Metrics to ensure improvement in one area is not negatively impacting another.

In our project we will use Dynamic Product Metrics, Static Product Metrics, Project Metrics

1. **In Dynamic Product Metrics,** we will measure *time of system which response* to user. Performance is precious for customer. We will measure *number of bugs* with feedback of user. Buck tracking tool will be helpful for doing this.
2. **Static Product Metrics,** it is important to efficient program. Short and efficient algorithms, fast and less memory spaced data structure is important. We will analyze number of class in OOP. How many data structure we will use.
3. **In Project Metrics,** we measure planned time and actual time of completion. It will calculate after project finished. Effort is main thing in project, difference between planned effort and actual effort also will calculated. Every project has planned budget, in this metrics we will calculate difference between planned and actual budget. For a

good team and successful project training hours is important. In this metrics we measure each team's training time.

5. Verification & Validation Plan

5.1. Verification

Verification is related with ensuring that the product is being built accordign to the requirements and design specifications. The main goal is verifying that the product meets the specified requirements. So the main concern is "are we building the product right?". For the verification of the PTS static verification will be used.

In terms of static verification, software inspections will be held. Software inspections will be a review for identifying the defects without executing the program. Inspections will be applied to requirements specification, software architecture and implementation one by one. Software inspections will help to create more efficient and beneficial testing scenarios for the validation. Also to have a complete software inspection process, InspectAll [6] application will be used as outsource tool.

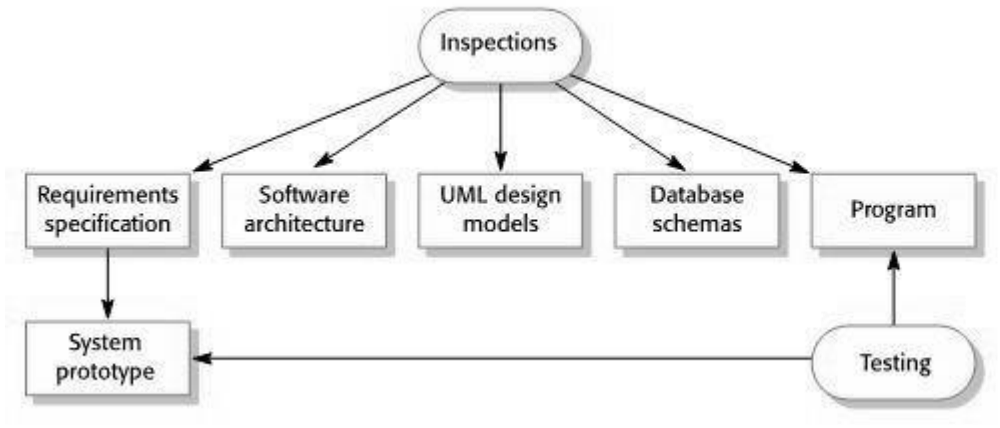


Table 12: Verification Process

5.2. Validation

For the validation of the product dynamic testing will be used. Dynamic testing will be supplementary to static verification in terms of ensuring that product meets customer needs. It is an experimental phase which will check the functional behaviour of the product. In terms of dynamic testing, tests will done in two types: Functional testing and non-functional testing.

5.2.1. Functional Testing

Functional testing will be done in four main levels: Unit testing, integration testing, system testing, acceptance testing.

Unit Testing

Unit testing will be held by developers after the implementation of a unit has finished. Unit testing will be done by automated approach and Jtest [4] will be used as the unit testing tool. White-box testing principles will be used for the unit testing. Unit testing will cover particular object class tests, component tests and interface tests.

Integration Testing

Integration testing will be held by developers after the implementation of several units has finished. Integration testing will be done by considering top down approach, which is an incremental approach so that critical modules will be tested first. For the integration tests, testers will prepare integration test plans and create test scenarios, then developers will execute these prepared test scenarios.

System Testing

After the completion of implementation and product become fully integrated, system testing will be held by the testers. Test scenarios and input data selection will be totally under the responsible of testers. Black box testing will be used for system testing. Usability testing, regression testing, recovery testing, functional testing and hardware/software testing will be held under the system testing processes.

Acceptance Testing

Acceptance testing will be held as user testing in terms of beta testing by the end users. After the system testing, the product will be released as beta version and will be available for the usage of end users.

5.2.2 Non-Functional Testing

Non-functional testing will be held by testers to ensure that product meets the quality aspects of the customer requirements. In terms of non-functional testing performance testing, recovery testing, security testing and usability testing will be done. Performance test will be performed for checking the response time, recovery testing for understanding the success after crashings and failures, security testing for robustness of the product and usability testing for checking friendly user interface aspects of the product.

6. Quality Assurance Plan

Software quality assurace is an important process for guaranteeing that the product will be delivered to customer with the required level of quality. In terms of quality assurance plan, to be standardized in international manners, we are going to follow CMMI standards[5] for all processes. We divide quality assurance plan in to two sub-topics as defect management approach and quality attributes approach. For the defect managment approach as indicated in the CMMI, core sets of software engineering processes are requirements development, requirements management, technical solution, product integration, verification and validation. With the condition of being adhered to CMMI standards, quality assurance plan will cover all of these five main stages.

6.1. Defect Management Approach

Requirements Development

This phase will be a check for understanding that documented standards, processes and procedures are followed according to predetermined paths. It will be an auditing and software metrics will be created. The main metric will be the number of errors.

Requirements Management

In this phase, metrics will be created which measure the effectiveness of this process. The measures will be focused on detecting the defects on last version of the product which is ready to be delivered. The goal will be finding the defects which can not detected because of the lack of test coverage or the requirements which exists in requirements document bot not referenced in traceability matrix.

Technical Solution

Technical solution is the phase which design and implementation solutions of the product is presented. The role of quality assurance for this stage is observing that predetermined processes are followed for the implementation. Software metric will be created according the number of defects. The metric will be quantified as defects per 300 lines of code.

Product Integration

Product integration is the phase which ensures that product components integrated succesfully with full functionality. In terms of quality assurance, metrics will be establised to measure the effectiveness of the integration. Metrics will focus on efects found that resulted from the interface specifications. Also some improvements for specifying interfaces in less ambigious way will be done.

Verification

Verification is the phase which we check that product meets the specified requirements. Role of quality assurance is limited for this stage. Bu quality assurance we will make sure that documentation legislations for verification are followed correctly.

Validation

Validation is the phase which we understand that we created the right product. Validation is also mostly in the domain of quality control instead of quality assurance like verification. So quality assurance is limited for validation. In terms of quality assurance, we will ensure that validation processes meet the standars and documented procedures.

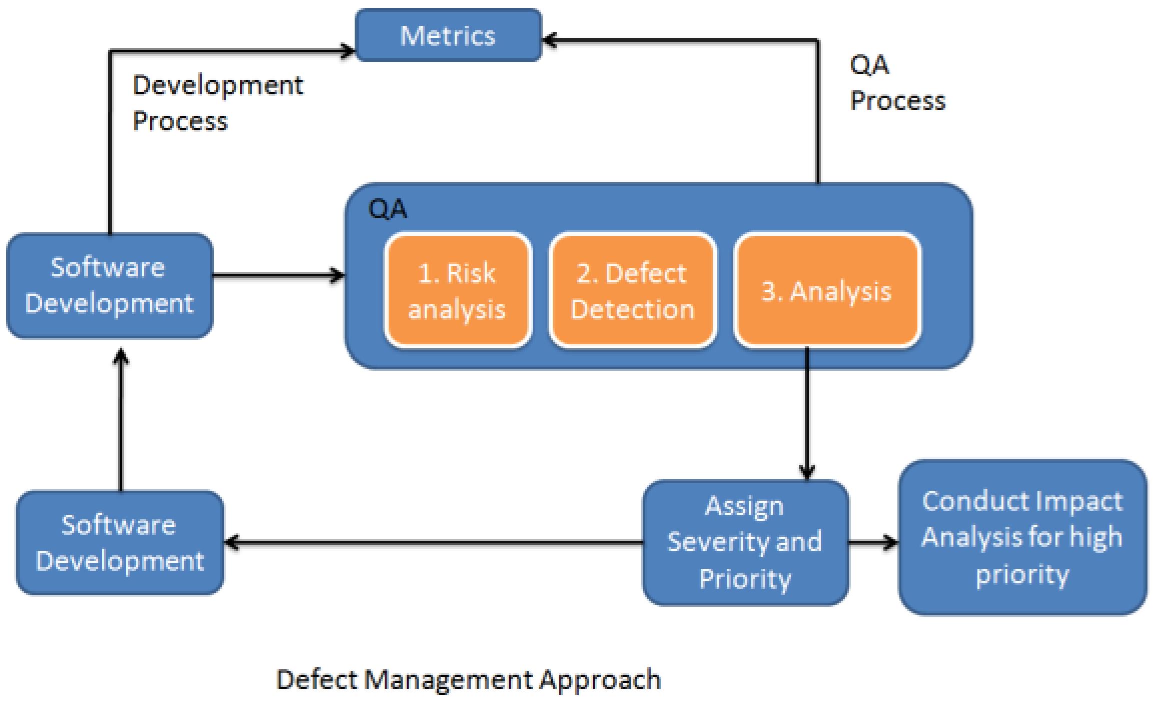


Table 14: DMA Process [9]

6.2. Quality Attributes Approach

The quality (non-functional) requirements of PTS are listed in detail in Project Scope Time Management Document (1.1.2). The priotorized quality requirements of PTS are security, accessibility, availability, confidentiality, efficiency, usability and interoperability. In parallel to defect managament approach, all of these quality requirements will be audited in terms of documented processes.

Security

The role of quality assurance in security will be following the rightness of implementation for the security aspects of the program. The main security aspect is login system. Also software’s ability to execute data related transactions securely will be followed.

Accesibility

PTS will be accessible locally by just clinic/hospital personnel and mobile or tablet access will not be allowed. In terms of quality assurance, the suitability to these aspects will be followed in all stages of defect management approach.

Availability

Availability is related with operation time of the system and PTS will be available locally on all clinic/hospital computers for 7/24. In terms of quality assurance, procedures for sustainability of the system will be followed and checked.

Confidentiality

Confidentiality is an important aspect of PTS because the product will store private information of the patient and all these information are secured by the laws. So, the role of quality assurance will be following the documentation and suitability to legal obligations.

Efficiency

Efficiency covers the system response times of PTS for notifications and result sending. For the quality assurance, infrastructure of the PTS will be followed for meeting the speed and performance needs of the requirements.

Usability

Usability is the most important quality requirement of PTS because featured aspect of PTS is user-friendly interface and simplicity. In terms of quality assurance, documentation and design process of development will be followed.

Interoperability

Medical imaging modules will be supplied by outsource services. For a healthy integration, documentation of outsource service will be followed and integration process will be checked regularly.

7. Software Process Improvement (SPI) Plan

Software process improvement is an important aspect for ensuring the quality of the software or speeding up the development processes [10]. In terms of software process improvement, we should totally understand the existing processes and update these processes according to needs and upgrade product quality. The goal of the SPI is reducing the development time and releasing PTS more early than the planned schedule. To construct a qualified SPI, we will follow the process improvement stages: Process measurement, process analysis and process change.

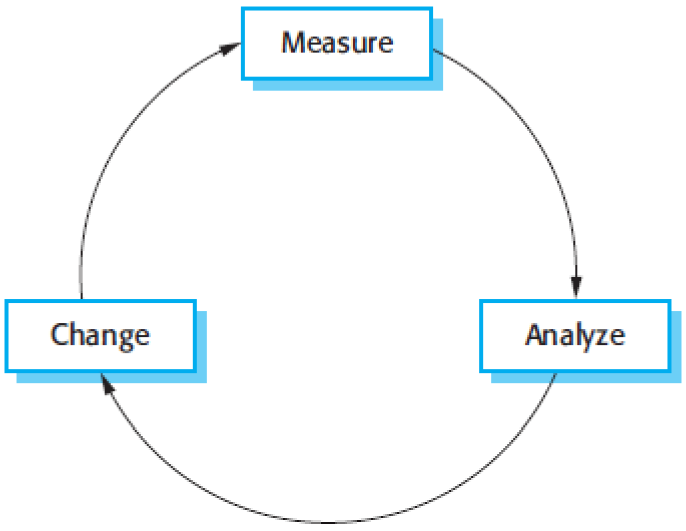


Table 15: Process improvement cycle [10]

Process Measurement

Already we have a plan-driven software process model which the process standards are clearly defined. Quantitative process data will be collected for reducing the development time of the project. Furthermore, we will build process metric which will include calendar time to complete an process for measuring taken time for a task. For finding out the parameters to be measured, we will use GQM (Goal-Question-Metric) paradigm.

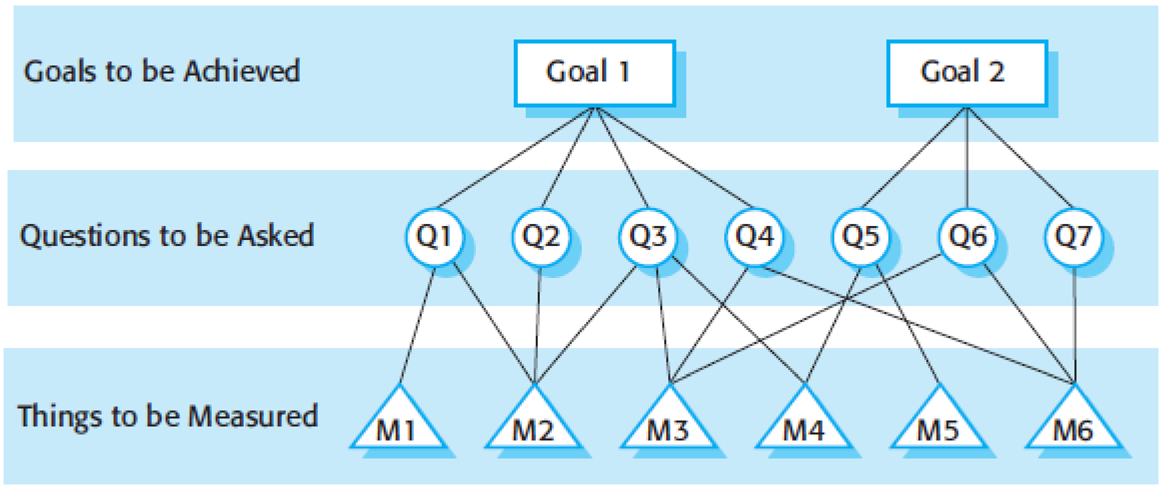


Table 16: The GQM Paradigm [10]

Process Analysis

Process analysis and process measurement are related sub-topics. Process analysis includes the understanding of activities be used in the process [10] . In terms of process analysis, relationship between measurements and activities will be revealed analyzed. Technically, questionnaries & interviews and ethnographic studies will be conducted for the process analysis.

Process Change

Improvement identification will be done after the process analysis. There can come up more than one change, so improvement prioritization will be done after the identification. Then process change introduction will be done by replacing new procedures with the old ones and integrating them with the existing procedures. To make the engineers understand about the changes process change training will be done and finally tuning phase will be started for the adaptation period.

8. References

[1] Projects, C. T. (2019, January 30). Use Case Points. Retrieved from http://www.wikizero.biz/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvVXNlX0Nhc2VfUG9pbnRz

[2] Function Points Analysis. (n.d.). Retrieved from http://people.cs.ksu.edu/~padmaja/Project/CostEstimate.htm

[3] Rating Scale For Software Understanding. (n.d.). Retrieved from https://www.researchgate.net/figure/Rating-Scale-for-Software-Understanding-Increment-SU_tbl4_243482239

[4] COCOMO Reuse. (n.d.). Retrieved from http://csse.usc.edu/csse/event/2009/ARR/presentationByDay/Tue_Mar17/14_Software%20Reuse%20and%20Maintenance%20Estimation.ppt

[5] COCOMO Post Architect. (n.d.). Retrieved from https://csse.usc.edu/TECHRPTS/1998/usccse98-502/CalPostArch.Pdf

[6] InspectAll. (n.d.). Retrieved from http://www.inspectall.com/

[7] Parasoft Jtest - Java Development Testing for the Enterprise. (n.d.). Retrieved from https://www.parasoft.com/products/jtest

[8] CMMı Standards.(n.d.). Retrieved from http://www.sqa.net/cmmi.html

[9] Software Testing Certifications. (n.d.). Retrieved from https://www.test-institute.org/What_is_Software_Quality_Assurance.php

[10] Sommerville, I. (2011). Software engineering. Boston: Pearson.