UNIT 4

PLANNING

PART-A

SHORT QUESTIONS WITH SOLUTIONS

Q1. Define project vision.

Answer : Model Paper-I, Q1(g)

The project vision is an idea that is evolved by single person or group of persons. They develops, declare and gets approval of idea to use it. And these people or a person is considered as 'visionary'. In a project team, there may be one or many visionaries and each visionary has unique idea corresponding to the project output. But inorder to execute this, the project requires a person (a single vision) who takes responsibility to combine, communicate and promote the product vision to the team members. This person is called product manager.

Q2. What is release planning?

Answer : Model Paper-II, Q1(g)

Release planning can be defined as the process of planning the software releases. This planning process helps to develop a software effectively in order to achieve good profits. In doing so, the team should work on single project at a time because with this approach the team will be able to release each project as soon as it gets completed.

Q3. What is time boxed plan?

Answer : Model Paper-III, Q1(g)

Timeboxed is a plan in which the release date is specified but the features that are to be developed is not defined properly. It is best to use timeboxed plans because it limit the total work done by people and enables them to make prioritized decisions. Here, the release dates are defined at regular intervals and the team is asked to determine low cost, more valuable alteratives for addressing requests.

Q4. What is the purpose of risk management?

Answer :

Risk management is designed to create and achieve long term commitments. Basically, every team performs adequate amount of work in every iteration inorder to meet expected targets at the time of a project. Every project has a dedicated a velocity of story points. These story points are delivered through team every week but not in one attempt. The team members improves the velocity of work regularly to meet the organizations deadlines. In doing so, they integrate the employee velocity with the release plan.

Q5. What is the role of transition indicator?

Answer :

The role of the transition indicators is to estimate supposed occurrence of the risk. It is inevitable that team ignores the upcoming risk which is ready to happen while they focus on objective indicators instead of subjective indicators. For instance, consider a risk such as "unpredictable publicity leads to lengthen downtime". Therefore, the transition indicator could be as "server utilization displaying an upcoming usage nearly 80%.

Q6. Write a note on commitment ceremony.

Answer : **Model Paper-I, Q1(h)**

The commitment ceremony can be defined as a small activity that takes place at the end of the iteration. In this ceremony, a commitment or a promise is made to the team and stakeholders in terms of execution of user stories in the iteration plan. In this commitment event, all the team members, customers, testers and programmers are assembled and meeting is held where in they are asked to read all the user stories of iteration plan. In addition to this, they also discuss that the team will be delivering or all executed user stories by the end of the iteration. Each and every person should participate in this commitment of delivering the user stories. If there are any issues or disagreements regarding the commitment then it should be discussed among the team and a suitable plan is designed respectively.

Q7. Define estimating.

Answer : **Model Paper-II, Q1(h)**

Estimating can be defined as the process of predicting the time require to accomplish the story or task. Ideally, the process of estimation is considered as the most difficult task. For some programmers, the task of estimation becomes in accurate. To get good estimates, they adapt the concept of padding, however this is also in approach because in padding the estimates are multiplied by numbers.

Q8. Write any two ways to improve the team velocity.

Answer : **Model Paper-III, Q1(h)**

Some of the ways that enhance the team velocity are as follows,

1. Reduce the Technical Debt

In general, technical debt is considered as a major technical issue which effects the team productivity. If there are any productivity issues then there will be no improvement in velocity. In order to overcome this issue, focus on the higher quality code and the velocity will be improved automatically. But, teams facing acute technical debt consumes months or years to cleanup. So, it is better to resolve the technical debt issue incrementally. Iteration slack can be used to solve the issue. It is not possible to notice a change or improvement in just few months.

2. Enhance Customer Involvement

The programmers either make some random guess or wait for the customer feedback when customers are not available it is better to engage the customers in the meetings. So that they are always available to clarify the programmers doubts. If they are not available, it may effect the velocity. So, it is good if the customer is present to clarify or answer the programmer questions.

PART-B
ESSAY QUESTIONS WITH SOLUTIONS
4.1 VISION

Q9. Define product vision. Explain how to evolve, identify document, create and promote the product vision.

Answer : Model Paper-III, Q8

Product Vision

Every product in an organization initiates with a vision. It is one of the essential part of the product's success. Ideally, a product vision can be viewed as an idea, inspiration or purpose. It serves as focus for the project.

Every project starts with a small idea that focuses on output. The idea should be expanded and draw attention among the customers (by introducing preview in the market). It should provide new services. So, achieve this programmers, domain experts etc., should be appointed. The goal of the project should be to deliver an output that satisfy the customer requirements.

Evolution of Vision

The project vision is an idea that is evolved by single person or group of persons. They develops, declare and gets approval of idea to use it. And these people or a person is considered as 'visionary'. In a project team, there may be one or many visionaries and each visionary has unique idea corresponding to the project output. But inorder to execute this, the project requires a person (a single vision) who takes responsibility to combine, communicate and promote the product vision to the team members. This person is called product manager.

Identifying the Vision

The ability of product manager in maintaining and effectively promoting the vision gets effected. So, to address this if the project team has only one visionary it is better to consider visionary as a product manager because it eliminates the possibility of any confusion. However, if the product manager sees that vision is worth giving a try and is achievable then his involvement improves the chances of project to deliver a good product. If in case, the visionary is not able to involve in the product development then the responsibility of product manager should be dedicated to some other person. The Ex. visionary should recommend a person who communicates and understands about the vision. Mostly, projects has multiple visionaries and this situation generally rises in custom software development. In this case, multiple visionaries should combine the ideas into a single common vision.

If a team is assigned with multiple projects, then each project should possess unique vision and each vision is executed in an order. However, if the product manager fails to dedicate separate vision for every project the he can helps team members to interact with multiple visionaries. So, here in particular the role of the product manager can be fulfilled by a person who has business skills, political knowledge and facilitation skills.

Documenting the Vision

After identifying single vision for a team, the vision should be documented in a form of a statement. This task should be performed jointly so as to eliminate any confusion and conflicts with respect to the vision statement. If the statement is not created well, then the entire process results in an unsatisfactory product. The creation of vision statement helps to maintain and promote the vision.

The vision statement should be considered as a living document because product manager should check and make changes on daily basis.

Creating a Vision Statement

The vision statement should consists of three parts,

(i) What the project should achieve?

(ii) Why the project should be considered valuable?

(iii) The success criteria of the project.

The vision statement should be clear, simple and short and should describes the main idea of the project.

Consider an example of Google vision statement.

"to provide access to the world's information in one click".

The above statement has three parts as world's information, access and one click.

Google vision statement describes that worlds information can be accessed by people in just one click.

(i) What the Project Should Achieve

This section encompasses problem description as well as opportunity address by project. The entire process is presented as end result. Google products help to access the information. It enables easy access of information. It maintains the databases for information and also provide services to every one in the world just by one click. It is easy to use.

(ii) Why the Project is Considered as Valuable

Google is considered as valuable to the customers because it provides access to information from anywhere. Some times, it is not possible to find information in books in that case, Google allows the customers to access and find the information in the search engine.

(iii) The Success Criteria of the Project

This section is concern with the success of product, like when it is successful. It also advice the project managers to be keep tough, clear and unambiguous targets. The success criteria of the project can be known if the project is used by multiple customers. Now-a-days, Google is accessed everywhere in the world.

Promoting the Vision

Once the vision statement is created, it should be declared as project vision.

Every visionary should participate in product decisions, iteration demos and interact with real customers. They should provide feedback regarding the progress that helps to improve the project plan. Sometimes, it is not possible to involve visionaries in product decisions. This may lead to confusion with respect to the project output and also decreases the teams understanding of product development. So, it is required for the team to interact with the visionary for better understanding of product's purpose develop more ideas and decreases the cost. If there is no possibility for team to interact with visionaries, then the product manager should be responsible to communicate with the visionary for the project plan, receive feedback and share it among the team members.

4.2 RELEASE PLAN

Q10. Define release planning. Explain briefly about it.

Answer : Model Paper-I, Q8(a)

Release Planning

Release planning can be defined as the process of planning the software releases. This planning process helps to develop a software effectively in order to achieve good profits. In doing so, the team should work on single project at a time because with this approach the team will be able to release each project as soon as it gets completed. Consider an example of team A and team B. These two teams work on multiple projects simultaneously.

Team A

Month	1	2	3	4	5	6	7	
Project 1	Works		Works		Works (release of project 1)	₹ ₹	₹ ₹	₹ ₹ ₹ ₹ ₹
Project 2		Works		Works		Works (release of project 2)	₹ ₹	Total Amount

Team A works on two projects simultaneously. But to avoid the task-witching penalties, the team works on projects alternatively i.e., In first month the team works on project 1 and in second month the team works on project 2 and so on. By implementing this process, project 1 will be completed in five months and project 2 will be completed in six months. After releasing two projects, the team earns some value for the projects.

Team B

Month	1	2	3	4	5	6	7	
Project 1	Works	Works	Works (release of project 1)	₹ ₹	₹ ₹	₹ ₹	₹ ₹	₹ ₹ ₹ ₹ ₹ Total Amount
Project 2				Works	Works	Works	₹ ₹	

Team B also works on one project at a time that is team B works on project 1 and completes it within three months. At the end of the third month, the project 1 is released. This release enables them to make profit out of it. Mean while, same team also works on project 2 and complete it in next 3 months i.e., so it is observed that the two projects got completed in 6 months and the value of work didn't changed.

Release Frequently or Early

If a team is working on a single project then the software release can be done early. At the time of release, the project should consist of important features that makes the product unique and increase in value.

Consider an example of two teams A and B respectively. Team A completes a project and releases, it within six months. The team earns some amount that is equal to project investment.

On the other hand, the Team B combines the important features that makes the product unique and releases it after three months. As a result, the product release earns certain value or amount in each month. After the first release, the team B starts working on the other features that should be released after six months.

In comparison to previous example, Team B has just changed the release plan and not the productivity. They earned triple to that of team A. The increase in income is because of the income obtained in first release month and in second release they earned nearly three times the amount of the project.

Team A

Month	1	2	3	4	5	6	7	
First Release						Release of prject	₹ ₹ ₹	₹ ₹ ₹

Team B

Month	1	2	3	4	5	6	7	
First release	Works	Works	Works (release of project)	₹ ₹	₹ ₹	₹ ₹	₹ ₹	₹ ₹ ₹ ₹ ₹ ₹
Second release				Works	Works	Works (release of project)	₹ ₹	

Q11. Explain how to release a product frequently with an example.

Answer : **Model Paper-II, Q8(a)**

A software product should be released frequently with helpful features. A certain deadline is fixed to release a product. But setting aggressive deadlines may cause harm rather than good. This is because they extend the schedules. So, start frequently releasing by including less in every release. In essence, a software product can be released with less features in each release. To perform this a minimum marketable feature is considered as a best tool. MMF is defined as a feature that provides numerous benefits to the market. This feature is applicable to internal users as well as external customers. It provide benefits such as competitive differentiation, revenue generation, and cost savings.

Release plan should be created on basis of stakeholder profit. Once, if the minimal features are made from the product then these features are combined and set into release.

Consider an example of a team that produce a software to create PDF files. The software should be developed with unique features that has more value. The basic features of PDF creator are creating documents with multiple formats (PDF, PDF/A, PDF/X), including watermarks or stamps in the document, combining multiple documents into single PDF, sending the created files through e-mail.

The unique features can be password protection for PDF documents, saving PDF documents automatically. So, in the first release, the product should consists of features such as creating documents with multiple formats, password protection for PDF documents, saving PDF documents automatically. The first release features can be used as technical preview to promote the product. In later releases, the other basic features should be developed such as including watermarks or stamps, combining multiple documents into single PDF.

Q12. Write short note on,

(a) Adapting plans

(b) Keeping the options open.

Answer :

(a) Adapting Plans

The significant results can be achieved by increasing the value of each release. This process can be executed easily i.e., once it is released to collect the feedback from the stakeholders and focus on the important or valuable features analyzed by stakeholders. Also, do not include the work which is not important. Using XP, the team can change the plan frequently in each release and also it enables to make adjustments in the plan in every iteration. This helps to meet or solve the sudden challenges quickly. A software project can be initiated only if the team members are aware of less amount of risk associated with software which will make it valuable. Eventhough, if the team keeps the information corresponding to the software value, it is important to note that the opinions from stakeholders matters a lot. Thus, representing the demo samples and also performing the actual releases. By implementing this process, the team realizes that some of the initial assumptions mode by them about the software value were incorrect and can be changed. Subsequently, the plan can be changed to represent the new changes.

There are many ways to increase the value of the software such as creating opportunities and considering the plan an learning plan not as implementation plan. It is necessary that the team clarifies the doubts and create good ideas to implement in practise.

Consider an example of collaborative online word processor in which the team members are unaware about the support required to import the Microsoft word.

However, it is important that some type of support to be included and it may take long time to provide support for all the word documents and also prevent including other features. If the necessary support is not provided then it may result in losing the customers. In order to handle this situation, it is better to add a rudimentary import feature to the software and release it. And also it is best to create a report on the capabilities which are required to provide support to import the documents. This information or report helps the team to adapt the plan and increases the product value.

(b) Keeping the Options Open

In order to increase the software value, the opportunities are created. To use these opportunities, it is required to develop a plan with which the team can release any time. If there is any occurrence of case that the product should be released at any cost then the product can be released with the possible opportunities. But if there is sudden project is cancellation then the product can be released with the features it currently possess. It is also required to release a product at any time that posses some value which is equivalent to the investment made.

In order to release a product at any time, it better to develop a plan in such a way that no story is dependent on the another story. It means each story should posses its own value. In some cases, the subsequent stories are developed on the basis of previous stories, but every story should be released separately. Consider an example of a plan in which one feature is to provide login screen and the other is to allow 'login screen' to provide the client-specific brand. Here the second feature improves the first feature but the features are releasable separately.

Consider an example of an application which provides features such as get data from a user, validates the data and writes the data to the database. Initially, story is created for each and every feature respectively and this is often considered as horizontal stripes. It is as an simple and easy way to create stories.

However, it is not possible to release, review effectively till the completion of all three stories. Because these features are combined as whole.

The another way to create the stories for each task is to create them in (narrower) single utility. Consider the other stories such as processing customer data which includes processing shipping address and processing billing information and these are considered as vertical stripes.

In earlier days, it is difficult to make the stories releasable because it needs practise and experience. Some times, releasable stories provides more flexibility in planning but it may consists few story defects.

Q13. Explain how to create a release plan.

Answer : **Model Paper-III, Q9(a)**

Plans are of two types that are as follows,

1. Scopeboxed plans
2. Timeboxed plans.

1. Scopeboxed Plans

It is a plan in which the features that are to be developed are defined well in advance. But the release date is not specified.

2. Timeboxed Plans

It is a plan in which the release date is specified but the features that are to be developed is not defined properly. It is best to use timeboxed plans because it limit the total work done by people and enables them to make prioritized decisions. Here, the release dates are defined at regular intervals and the team is asked to determine low cost, more valuable alteratives for addressing requests. With the help of project vision more information is added to the plan in order to create minimum marketable features. After creating the features, divide them into stories, hire experts or programmers to estimates the value of these stories. With the help of estimation, the values and cost of stories are determine so that they can be prioritized. A list of prioritized stores is formed and it is called as release plan. This plan can be improved with the feedback of stakeholders while iteration planning.

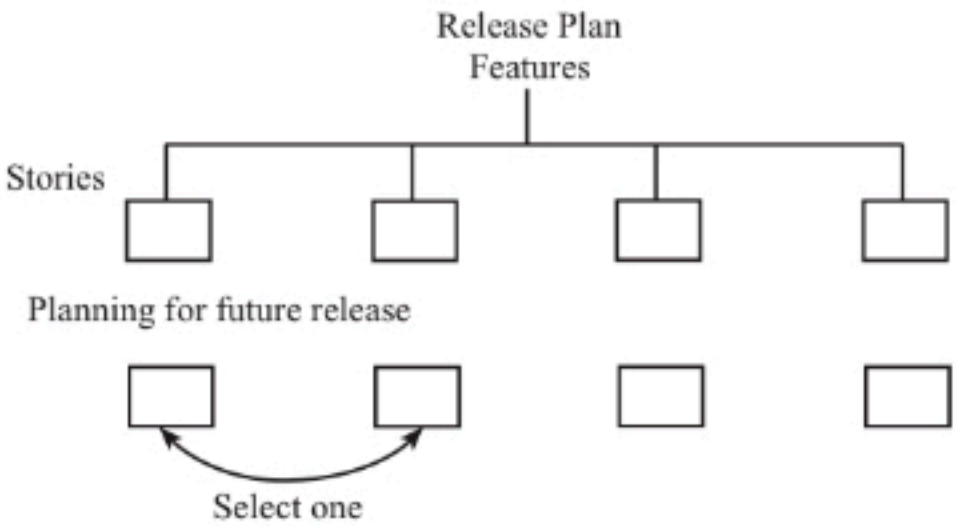

Figure: A Release Planning Chart

Q14. Discuss about,

(a) Planning at the last responsible moment.

(b) Adaptive planning and organizational culture.

Answer :

(a) Planning at the Last Responsible Moment

The last responsible moment an be defined as the moment at which responsible and important decisions can be made. In order to reduce the work effort in the plan, last responsible moment is implemented. By using this, it is easy to make release plan with less features. Therefore, last responsible moment saves the time during brainstorm stories, time estimation and prioritizing.

Furthermore, another way to implement last responsible moment is planning horizons. It is defined as the time taken by an expert to plan for the future in business (i.e., having for sight in business). In general, project requirements are checked in advance but planning horizon extends till the end of the project. There are few planning horizons which are used for specific purpose such as,

1. Tired set of planning horizons are used to plan at the last responsible moment.
2. Long planning horizons are used for general plans.
3. Short planning horizons are used for specific and detailed plans.

The planning horizons are based on business situation. If there are many responsibilities to perform, the planning horizon should be long and in detail. If there are any changes made to the plan then the planning horizon should be short.

The following steps are involved in a planning horizon. If the experts are not sure which planning to adopt,

1. Initially, define a project vision.
2. Define the release dates and corresponding to next two releases.
3. Define all the minimum marketable features for current release and initiate the point to push the unwanted features into the next release.

   Define the stories for current feature and also for many of the current release. Additionally, put those stories forward that doesn't fit in upcoming releases.
4. Define the list of prioritized stories for the current iteration and also for next three iterations.
5. Check the requirements, customer tests of the stories in the current iteration.

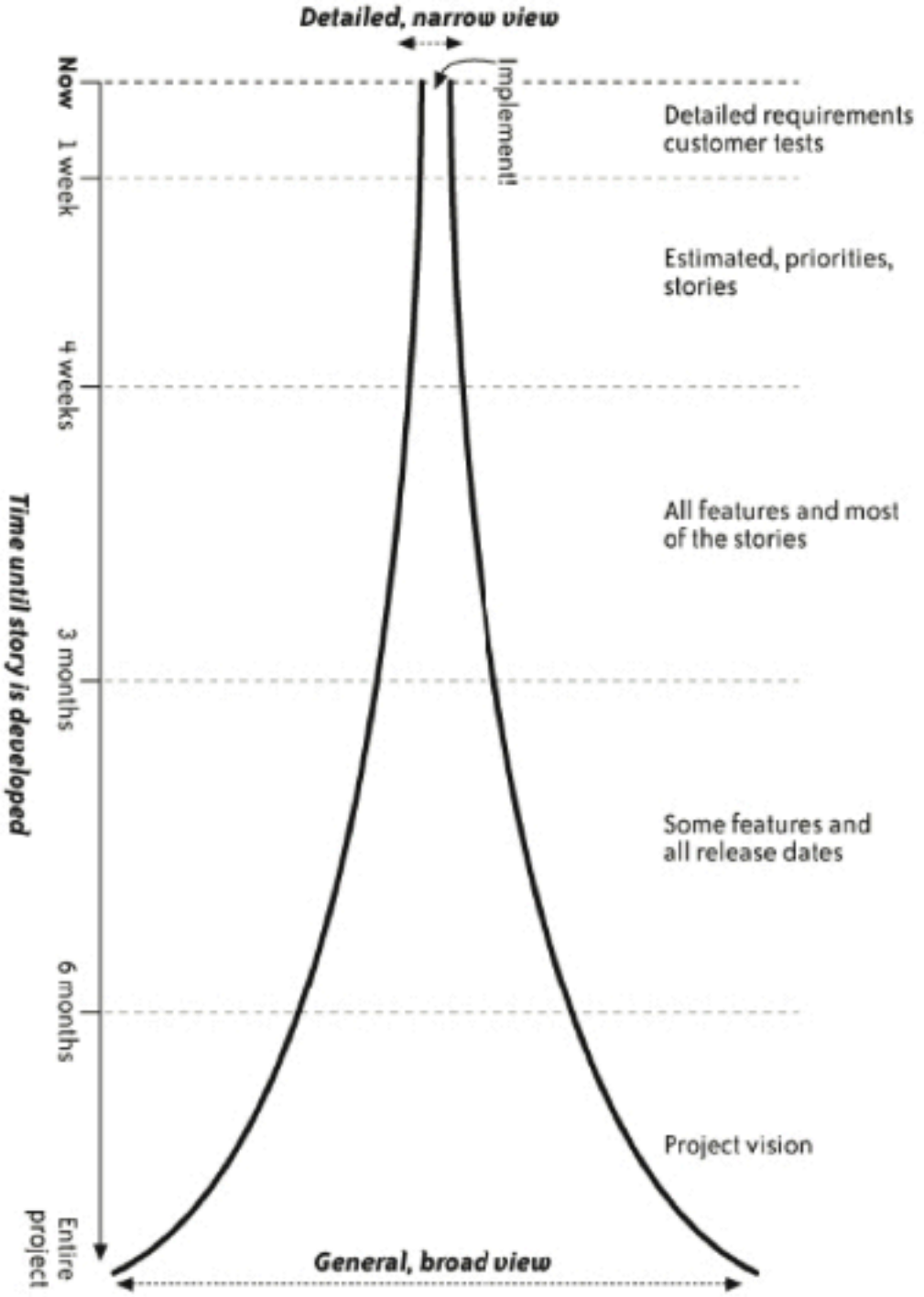

Figure: Planning Horizons

(b) Adaptive Planning and Organizational Culture

Adaptive planning can be defined as the simple and unplanned process or method that helps to achieve a vision. In this process, the team is unaware of the result but it achieves the vision.

In agile development, features does not play any role in disturbing organizational culture but the culture certainly gets effected when there is a change in adaptive planning. So, changes in adaptive planning confirms to change in the development team, reporting evaluation team and in executive management adaptive planning is preferred in multiple areas of project community. In general, changes cannot be made to adaptive planning. However, with the help of executive support, changes can be made slowly.

With in the supervision control, adaptive planning is done in the organization. In addition to this goals should be set and planned so as to meet the organization standards and expectations. In small and frequent releases, estimation and prioritization of the stories is required for the current release. Subsequently, once the trust is gained from stakeholders and executives, the planning for setting and achieving the goal can be minimize and the time saved can be useful in moving towards the adaptive plan.

4.3 RISK MANAGEMENT

Q15. Explain in detail about Risk Management and its plan.

Answer : Model Paper-I, Q8(b)

Risk Management

Risk management is designed to create and achieve long term commitments. Basically, every team performs adequate amount of work in every iteration inorder to meet expected targets at the time of a project. Every project has a dedicated a velocity of story points. These story points are delivered through team every week but not in one attempt. The team members improves the velocity of work regularly to meet the organizations deadlines. In doing so, they integrate the employee velocity with the release plan.

Furthermore, efficient XP teams meets a stable velocity. But, velocity often addresses the problems that a team undergoes at the time of a project development. So, in dealing with these problems the team often get frustrated with issues like hard drives failure. Eventhough backup is possible it may not allow data restoration. In the mean while stakeholders demands for necessary improvements in the software provided, before it gets launched.

Apart from all this, stakeholders depends on few periodic targets which are made possible only through risk management.

A Generic Risk Management Plan

Each and every individual project encounters group of risks which includes turnover, new requirements, work disruption etc. These risks factors gives negative impact on development by increasing the time taken to meet the schedule commitments. The range of risks encountered by the team members is entirely based on an organization. Ideally, the database of organization is depicted in the figure. This represents the possibility of achieving the schedule commitments in prior to the risk multipliers.

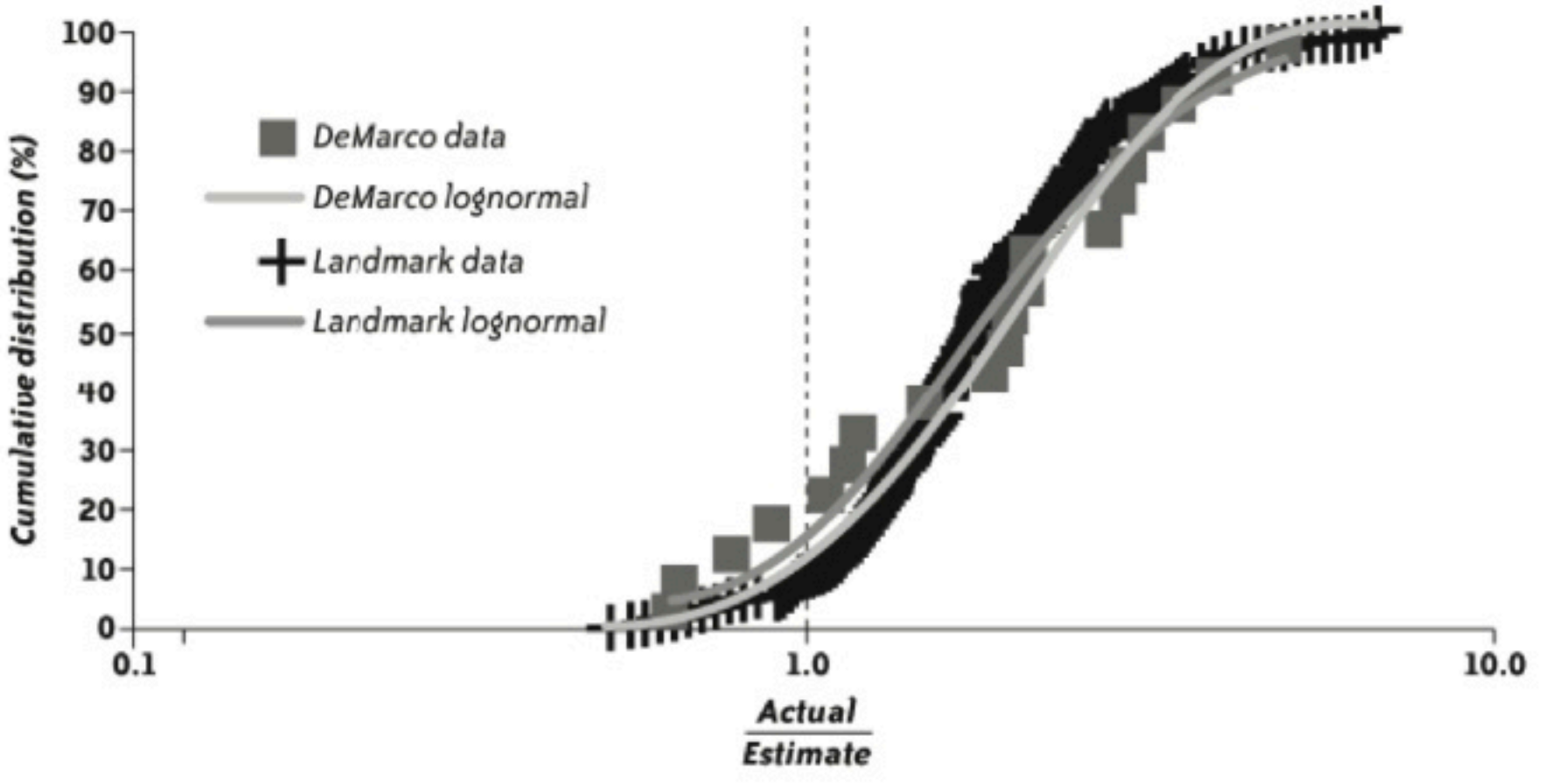

Figure (i): Example of Historical Project Data

Most of the organisations does not contain the information corresponding to risk multipliers which illustrates the probability of achieving several schedule commitments. This can be shown in the figure (ii). For instance, if team is adopting a risky approach where there is a 10% possibility of achieving the schedule commitments. But, if the team increases the estimate time, then there exists 50% chance of finishing the estimated schedule, virtually, it can also be achieved by quadrupling the estimated schedule.

Process Approach			
Percent Chance	**Rigorous**	**Risky**	**Description**
10%	X1	X1	Almost impossible ("ignore")
50%	X1.4	X2	50-50 chance ("stretch goal")

Figure (ii): Generic Risk Multipliers

Process Approach			
Percent Chance	**Rigorous**	**Risky**	**Description**
90%	X1.8	X4	Virtually Certain ("Commit")

If the team is adopting XP practices, in doing so if the team is particular in achieving "done done" for each iteration. Then, the velocity becomes stable and team will be able to fix the bugs at every single iteration, thus minimizing the risk factor. The use of risk multiplier should be within the Rigorous column. In certain cases, user may not be particular in performing "done done" at each iteration, this cause the velocity to fluctuate and also delay in fixing the bugs subsequently, the iterations leads to the usage of risk multipler within "Risky" column.

Q16. Describe in detail about project specific risks.

Answer :

The implication of XP practices and using risk multipliers on the stories leads to the exposure of risks in every project. The basic risk multipliers involves the risks corresponding to flawed release plan, ordinary requirements growth, employee turnover and some other risks could be in relevance with the project. These can be managed by generating risk census which consists of the list of risks that come across during the project development. So, that they can resolve unique risks within a project.

Further more, the programmers DeMarco and Lister initiated the task of collecting census through brainstorming the potential risks or catastrophes. Subsequently, conduct the team meeting on a project at a place and distribute the index cards. During this meeting employees should come up with potential risk and factors associated with the project failure. It involves various questionnaires which are as follows,

1. There could be possibility that project manager expect the team to work at the night also.
2. There could be possibility that the team may have to answer the higher officials corresponding to the failure of project. The questions could be in relevance with the difficulties faced at the time of project like where exactly things went wrong.
3. There could be possibility that the team should set better goals regarding a project but mention exactly in reverse.
4. There could be possibility that the team assumes those conditions in which project may fail eventhough there does not exist any failure with team members.
5. There could be possibility that team assumes the factors that has maximum chances for a project to get failed whether it can be stakeholders faults? Customer's faults? Testers? Programmers? Management? Team members faults? and so on.
6. There could be possibility that the team may anticipate the possible ways to make a project succeeded by leaving particular stakeholders unhappy.

The above mentioned questions stimulate the team members with some new ideas. After collection of fall potential risk factor the team member discuss the series of catastrophes that are occurred due to some brainstorm scenarios. With these scenarios, team members can identify the fundamental reason for the occurrence of risks which results into catastrophes.

For instance, consider an application generation which contains a catastrophe of "extended downtime". Here, the scenario which entails above catastrophe can be "excessively high demand". The reasons for this include "denial of service attack".

Immediately after completing the brainstorming risks, all the team members are supposed to get back to the task of iteration mean while rest of the member should work on this with in the smaller group. Identify the following possibility for every risk estimated,

1. Estimated Probability: The team should estimate the extend of probability like whether it is high, medium, low.
2. Estimating the loss in terms of money, delayed days and project cancellation.

Employees can ignore the risks that are insignificant i.e., which does not effect the project. This can be done by risk multiplier. Team members need to determine whether to avoid the risk for maintain it through spending additional time or money using risk multiplier. Else, mitigate the risk inorder to minimize effect of risk.

The risks for which the team decided to deal which are to be needs to identify transition indicators, mitigation, contingency, risk exposure activities.

(i) Transition Indicators

The role of the transition indicators is to estimate supposed occurrence of the risk. It is inevitable that team ignores the upcoming risk which is ready to happen while they focus on objective indicators instead of subjective indicators. For instance, consider a risk such as "unpredictable publicity leads to lengthened downtime". Therefore, the transition indicator could be as "server utilization displaying an upcoming usage nearly 80%.

(ii) Mitigation Activities

It minimizes the effect of risk. This can occur in prior of the risks that are supposed to be encountered execute. Here, the team should generate stories for those risks and add in the respective release plan. These stories involve support horizontal scalability, preparing load balancer.

(iii) Contingency Activities

It also minimizes the effect of risk. This is applied only when it is needed i.e., at the time of risk occurrence. These activities are based upon mitigation activities which are implemented in prior. Some of the examples corresponding to these activities are "purchase more bandwidth from ISP", install load balancer, "purchase and prepare additional frontend servers".

(iv) Risk Exposure

It emphasizes on time/money taken to maintain the risk. Inorder to compute it, initially calculate the numerical probability corresponding to risk, then multiply it with the generated impact. While performing this, team members should be aware that they had paid for mitigation activities while the contingency activities are caused due to impact. For instance, team members assume the downtime with the popularity as 35%, then as a consequence the delay in time includes three days and $20,000 incase of bandwidth, collocation fees, new requirements. Therefore, the complete risk exposure comes upto nearly $7000 for one day.

There exists 100% possibility for a risk to occur which eventually becomes reality. So, inorder to deal with it the teams should improve respective release plan for maintaining it.

The remaining risks leads to failure of a project. For instance, consider a corporate reorganization which may disrupt the project team. So, forward such risks to respective executive sponsor.

It is important that the team members maintain time, money, contingency activities as these can be used whenever required. Else, combine risk exposure, dollar exposure with budget, day exposure of a schedule, where its effect is similar to the risk exposure.

Q17. Explain the procedure to make a release commitment.

Answer :

The team members working on a project may identify story points, that they decide to complete before release plan. This can be obtained by calculating the number of iterations remaining until the release date and subtracting it with the risk exposure. Again, multiply this with team velocity for identifying the total points within a schedule. Then, divide it with every simple risk multiplier inorder to predict the team member choices to meet the release date using necessary story points.

risk_adjusted_points_remaining = (iteration_remaining – risk_exposure) * velocity / risk_multiplier.

For instance, if a team member follows rigorous approach having 12 iterations, velocity of 14 points , risk exposure as one iteration. Then, compute the number of chances are given as,

Points remaining = (12 – 1) * 14

= 154 points

10% chance = 154/1

= 154 points

50% chance = 154/1.4

= 110 points

90% chance = 154/1.8

= 86 points

Therefore, in case of release date when team members have 90% chance then they need to complete within 86 points. Similarly, if its given 50% chance then they require 110 points as well as 10% means it include 154 points.

This can be illustrated pictorially using a burn-up chart shown in the below figure, which consists of total number of story points that team members finished and those points which are available for further release, total number of risk-adjusted points leftover. Map all these over a burn-up chart given below,

It resembles the effort of team members in meeting the release date using characteristics so as to deploy a project. They follow 90% chance to complete the project. Where as, the features lying in between 50 to 90% are described as stretch goals. It is not necessary to denote the 50% chance of finishing project.

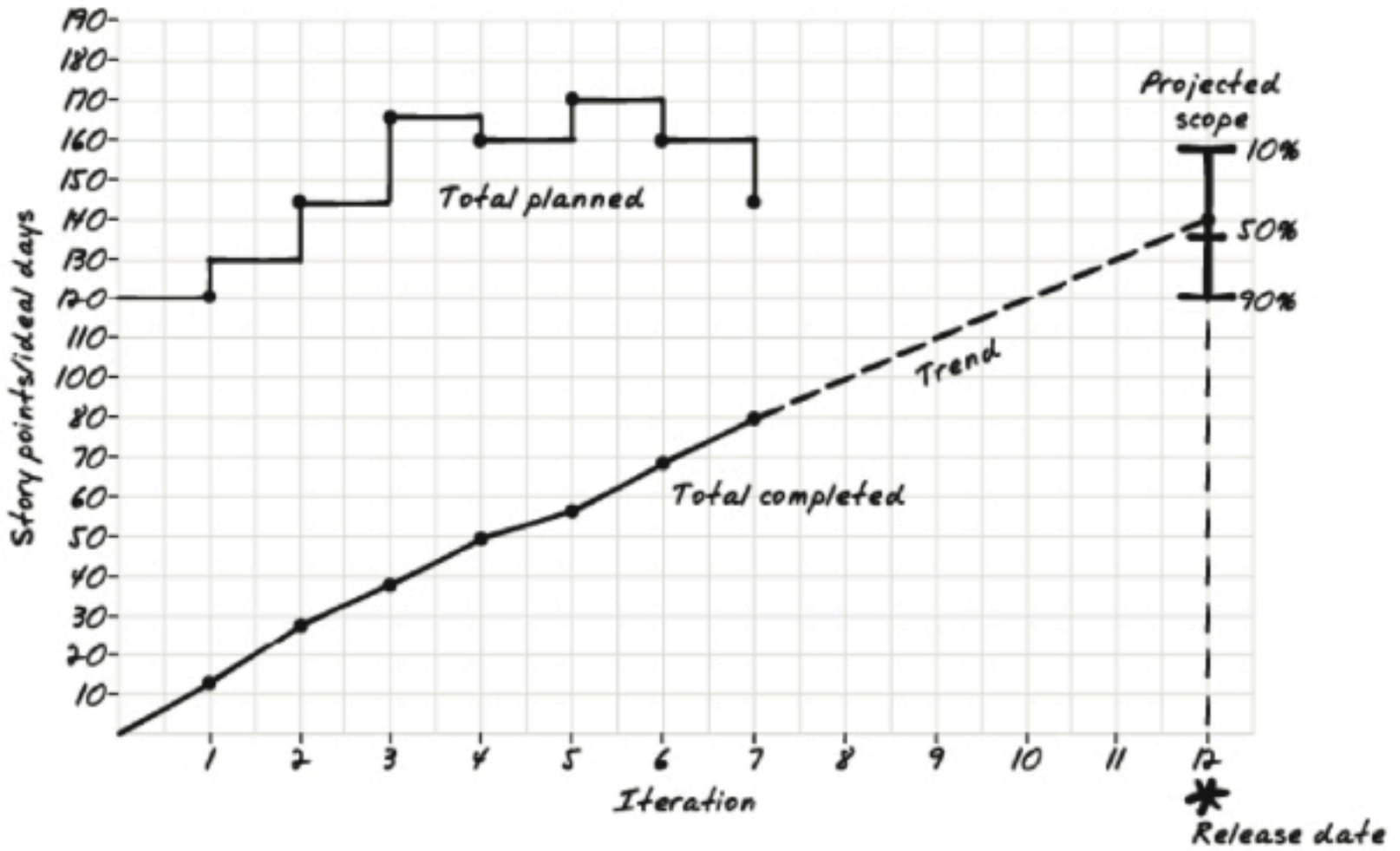

Figure: Burn-up Chart

Success Over Schedule

The main criteria of risk management is to organize the risks inorder to meet the schedule commitment. Moreover, it also includes gaining success to the organization along with meeting the release plan. Sometimes the delay deploys the project which keeps the success at risk. Therefore, achieving the release date is also quite necessary. But the actual goal of an organisation is to gain the success though it takes time to finish the project.

Q18. Explain how to handle the situation in which the commitment is not good enough.

Answer :

It is important to develop a good plan in order to meet the iteration commitments. The best way to develop a plan is to reduce the scope or extend the release date. If required, assistance of a product manager can help to make a good plan. By implementing an uncertain plan, there is a chance of occurrence of negative estimates. It must be ensured that every story is completed in the iteration and whether enough slack is used to maintain the team velocity or not. This may result in decrease in team velocity but it helps to reduce the risk and also allows to use risk multipliers.

Although it is not possible to change the scope or release date at that moment of time, small modifications can be done in the course of time. Over the period of time, the team builds trust with the stakeholders, they will be aware of benefits that are provided by reducing scope or extending the release date. At the same time, it is better not to commit for more work because the increase in the amount of work may result in the technical debt and ruin the plan.

Sometimes, the organizations may set difficult deadlines to increase the team productivity but this may result in low quality product service. In this type of organizations, it is not possible to neglect the project deadlines so it is better to introduce "over time" policy to perform work.

It the organization increases the pressure on the team members then the decision is left to the team members whether to value the current job or to look for other jobs immediately. There is also another alternative i.e., to shift to the another team or department within the same organization. But if the team member is responsible for particular plans and schedules, it is required to respect the demands and complete the respective work.

4.4 ITERATION PLANNING

Q19. What is iteration planning? Explain briefly about iteration timebox and iteration schedule.

Answer : Model Paper-II, Q8(b)

Iteration Planning

The concept of iteration planning arises for defining the intermediate results sequence. Ideally, iterations are considered as one of the important phase in a project. In an iteration phase, the valuable or helpful stories are chosen from the release plan. After this, during the termination the team analyses, build and evaluate each story. Once, this is done, they initiate a new cycle.

Iterations are considered as safe and secure procedure in software project. It operates by checking the weekly or monthly progress of the project and sharing it among the stakeholders. By doing this, the team can be able to communicate with the organization members regarding the activities and progress of the project.

Iteration Timebox

In order to avoid the delays in the progress of the project, an iteration timebox is used i.e., a timebox is fixed. By using this, work completes in the specific time irrespective of the amount of work done. The iteration timebox exposes the problems that arises in the project and allows to fix it.

In XP project, once the iteration is completed, next iteration starts immediately. Most of the teams plans the demo first, so that minor issues can be solved leisurely.

Iteration Schedule

The iteration schedule is as follows,

1. Schedule describes the previous iteration for half an hour.
2. Evaluates and handle the previous iteration for one hour.
3. Plans for the next iteration for half an hour to four hours.
4. Analyses and develop the stories.
5. Constructs the release plan in last step.

In general, most of the team begin their iterations at the starting of the week and ends it before weekend. It is better to select an iteration start time that is suitable to all the team members to work on a project.

Q20. Explain how to plan an iteration in detail.

Answer :

The following steps are involved in planning an iteration.

1. Iteration planning begins right after the completion of iteration demo. In this planning, initially calculate the velocities of the past iterations. Consider all the stories that are completed and calculate their respective estimates. This value estimates the completion time of the next iteration.
2. Based on the velocity value, the stories that are to be executed in iteration are selected. Based on the customer's choice, the valuable user stories are selected from the release plan and also check whether or not it matches with team velocity. If not, arrange the stories that match with the team velocity.
3. After selecting the user stories, divide the stories into engineering tasks. These task are difficult and only programmers can execute them as they are considered as programmer-centric. Some of the engineering tasks are such as,
   (a) Updating build script
   (b) Implementing domain logic
   (c) Include database tables and respective ORM objects.
   (d) Developing a new UI form.
4. Analyze the tasks that are used to complete the iteration user stories. In this process, some tasks are related to only a single story while other tasks are related to multiple stories. So, expert make emphasis on those tasks which are important for completing the iteration stories. Ideally, the concept of analyzing task is considered as design activity. It is easy to develop the software if all the team members has the same idea regarding the software project. Or else, the discussion should be made before they begin the process.
5. Each team member is assigned a task and is held responsible for that respective task. It is also necessary to guage the time needed to complete the task.
6. After analyzing the task, arrange them and check whether these tasks are sufficient to complete all the stories. And also check whether there are any duplicate or overlap tasks.
7. Estimating and analyzing these tasks that can be done simultaneously. Team members estimate the time to complete the each task.
8. The estimation process should be carried out in optimal hours. It also estimates the alternatives if there are any delays. But if there are any larger tasks, divide them into smaller tasks. Again these small tasks are combined that makes for one hour or two hours.
9. The final check of the plan is done. If there are still any differences, discuss and fix it.
10. Now, calculate all the task estimates of the current iteration and compare it with the total tasks estimates of the previous iteration.
11. In some cases, a team member may realize that it is not easy to execute all the stories in the plan. So, in this case, they discuss with on-site customers and make changes or replace the difficult stories with easy stories that are equally valuable or important. If this is not possible then story is divided or removed completely. In the same way, if a team member has ability to deliver more stories then new stores can added to the plan.

Q21. Discuss briefly about,
(a) Commitment ceremony
(b) After the planning session
(c) Handling the long planning sessions.

Answer :

(a) Commitment Ceremony

The commitment ceremony can be defined as a small activity that takes place at the end of the iteration. In this ceremony, a commitment or a promise is made to the team and stakeholders in terms of execution of user stories in the iteration plan. In this commitment event, all the team members, customers, testers and programmers are assembled and meeting is held where in they are asked to read all the user stories of iteration plan. In addition to this, they also discuss that the team will be delivering or all executed user stories by the end of the iteration. Each and every person should participate in this commitment of delivering the user stories. If there are any issues or disagreements regarding the commitment then it should be discussed among the team and a suitable plan is designed respectively. In particular, commitment is considered as important statement for the following reasons,

(i) It consistently assist the team in delivering the iterations as per the plan.

(ii) It provides an opportunity to deal and fix any disagreements and problems.

(iii) It also provides motivation to all the team members to work together and handle the issues in the iteration plan.

(b) After the Planning Session

Once the planning session for iteration is completed, the process of developing software initiates inorder to fulfill the commitment. The programmers in general provide suggestions for performing tasks and also advice other team members to join their work on the respective task. Once the task is completed, they divide and join with other individuals by forming new pairs.

Moreover, team members have their own respective tasks to perform. Programmers plan and write about the task on the task planning boards while the team members may not have a planning boards. So in this case, the customers and testers monitor the progress of the programmer and adjust the work based upon the programmer requirements and make it available each time they require it. This process helps to increase the productivity of the entire team.

In course of time, while the work is in progress, the iteration plan should be revised. After revising reflect those changes in the task. It is important is important to note that, all the original tasks and story estimates should be monitored so that they will be useful in planning next iteration.

The main idea of the commitment is to execute all the stories in the plan and check whether the current iteration plan delivers all the stories. After the completion of current iteration, release the completed software and allow the stakeholders to see it.

(c) Handling the Long Planning Sessions

The duration of iteration planning process can range between half an hour to four hours. In this time, the engineering tasks that are to be included in the plan are discussed. In a typical XP project, the team starts and complete the iteration demo within 4 hours. On the other hand, the newly appointed team during the first or initial iterations may face difficulties to plan them. They may take time to understand and learn the problem space, ideal methods to design problems. Even after the first month, if the iteration planning consumes more time then apply certain approaches to increase the speed of planning.

In iteration planning, the team may spend more time in designing the release plans which may cause problems. So, the planning should take place with previous iteration within the customers and at the same time the programmers can work on stories. In particular, selecting the stories corresponding to iteration plan is a simple process because these are selected from the front of release planning.

The long planning sessions are caused due to is the time dedicated in disintegrating the stories into engineering tasks. So, the occurrence of this can result in a lot of design work, which could be very complex. In iteration planning, most of the actual design work is done when the programmers and individuals combine to work on specific tasks. They collaborately work on problems to accomplish the engineering tasks.

Apply collective code ownership and pair programming techniques each time the team members fail to understand the existing design. Their incapability could be due to the lack to shared design understanding.

Subsequently, the team members should concentrate on the requirements to get the simple design. Else problems may exist. Inorder avoid this problem, discuss with on-site customer about the requirements and develop the design properly.

Q22. Explain briefly about,

(a) Tracking an iteration

(b) How to handle when the things go wrong

(c) What if a story is partially done.

Answer :

(a) Tracking an Iteration

Iteration planning is one of the important part of informative work space and it is necessary to track the progress of every iteration. In general, an iteration planning board is used to enter the tasks and stories of the plan before starting the iteration. So, if a team member starts or handles a particular task, then his/her name or initial should be mentioned on board for easy understanding to other team members. And after the completion of task, the status should be specified with a marker.

In the planning process, it is difficult to identify the tasks that gives problems. So to avoid these problems, track the progress of iteration everyday. Once, the iteration is completed, then the team removes all the stories and tasks from the board and set them aside or store them for future use.

(b) How to Handle when Things go Wrong

Commitment can ideally be viewed as a process of solving the problems and executing the stories of the plan. In some cases, if commitment is not fulfilled, it means that certain problem has occurred in the iteration commitment. If such situation arises, first check if there is any possible way to change the plan which still meets the commitment. This decision process should take place within the team and then revise the plan. In case if the problem is complex then the revising iteration plan involves reduction of iteration scope and dividing or removing of the difficult or big stories from the plan.

After revising the plan, a meeting should be held with customers and stakeholders to rebuild the trust. In this meeting, the revised plan is discussed and explained, why it is changed and how the team can avoid these type of problems in future.

(c) What if a Story is Partially Done

In an iteration, every story should be completed according to the respective plan. In some cases, if a story is partially completed then it is clear that there is a problem in planning. To handle these type of situations some of the teams may eliminate all the partially done stories and only deliver only those that are fully completed. In software development, the timeboxing concept is applied i.e., story should be executed within specific time. By implementing this, the software becomes atomic in nature i.e., either it is done completely within the timebox deadline else it is eliminated if it is not completed. Timeboxing concept enables to meet the project deadlines by providing limited product scope but does not compromise the quality of software.

With the help of timeboxing method, it is easy to develop the code and it does not keep the code unused. However, initiating from the start may require to rewrite the code and this pattern retains the code written first time and allows to develop a better code in second attempt.

If the story kept pending and the user is taking time to execute it, then it is better to delete the story. This is because unused stories may produce potential problems and serves as technical debt. However, if the user needs it, then they can retrieve it from version control.

Q23. Write a short notes on,

(a) Emergency requests

(b) The batman.

Answer : Model Paper-III, Q9(b)

(a) Emergency Requests

Emergency requests can be defined as the request which demand changes in the implementation in the middle of an iteration. These changes are requested by customers. And implementing these changes may cause disruption in the current iteration and delays in work. But understanding the business requirement is considered as core agile value. So, it is important to accept the emergency request change and provide the customers with benefits of the request changes.

In an iteration, plan can be changed on the basis of the condition. It works by taking out maximum work as it is added. In essence, the newly added stories should be placed out of the plan. In doing so, the stories that are yet to be started are only replaced. Consider an example that there is a two-point story which is done partially. Now it is not considered as a one-point story. So, replacing this partially done story into the plan may lead to technical error. Therefore, this point is difficult to guage how much work is pending.

Instead of leading confusion in the team, first understand individually the gravity of the situation and than plan it until the next iteration meeting starts.

(b) The Batman

In an iteration emergency requests can occur once and it helps the team to become responsive. But if it occurs frequently then revise or check the progress of the iteration. On-site customers should monitor the release planning after every second and third iteration.

In order to overcome these type of situations the concept of batman is introduced. Batman can be defined as a role which can be performed by the programmer who handles only the emergency requests while the other programmers can handle the task of programming. The batman doesn't work on the stories of the iteration plan. It there are more emergency requests, then based on the situation two or more batmans can be introduced for each iteration. It is also better to use daily iterations instead of a batman. This is because a daily iteration enables the team members to postpone all the emergency requests. So, that team members can focus or handle the plan. The daily iteration is only used when there are small adhoc issues such as bug problems or issues and minor changes and also it does not contain long-term release plan.

4.5 SLACK

Q24. Define slack. Explain how to introduce slack.

Answer : Model Paper-II, Q9(a)

Slack

Slack can be defined as amount of medium required for the stories for getting delivered within the iteration deadline.

Consider an example of a power cable and the workstation system. Refer it as a story because it is essential that the work done by the system is to be delivered. If the cable does not have sufficient length for connecting it to the work station system and the power socket then the small disturbance in the cable can result in sudden system shut down resulting in the loss of data. In this, case, the system can be moved closer to the socket so that it is resolved to some extent or a power cord can be sticked to the floor to avoid slight disturbances. Additionally, back up server can be setup to restore the lost work. In a similar way, the project plans also require slack to avoid small disturbances.

Inorder to introduce slack in iteration, plan in such a way that no work should be done on the last two days of the iteration. Although this process may provide the slack but it becomes waste. A much better approach is to schedule useful, important work that is independent of time. In essence, the team can set work differently in situation of emergency.

Consider an example of paying off the technical debt. Some times, the best teams might face the problem of technical debts. Eventhough, if the code is developed error free, and is perfect, there is a possibility that technical debt may occur.

Refactoring method can be applied to solve this problem and make the code free of technical debt. In the existing code, if there is any difficulty to understand, it is better to change and some part of code is not used any longer then delete it.

In iteration, paying down the technical debt helps to increase the team productivity. The best way to solve the technical debt problem is to allocate 10% of the respective iteration. In refactoring process, it is required to allocate the time for each specific and relatively small problem.

Q25. Discuss in detail about research time.

Answer :

Programmers should improve their skills inorder to maintain the consistency in developing project. This pattern will help to improve by making them learn new things which in turn enhances the work in the project. Research time can be defined as the dedicated time assigned to perform research on a particular topic or task. It provides support to learn new things and to include extra slack within the iterations. In order to implement this idea or concept every programmer should involve and perform their own research on a particular topic. This research time should not be utilizing in making project stories and changes in the existing project code.

By implementing this technique, the organization received many benefits. Subsequently, if the programmers are self directed and motivated then they can make use of research time to receive high returns on the project.

Furthermore, the time given for research plays a major role and is very important for XP teams. Some times, consistent and strict iteration deadlines may help to minimize the risk and keeps motivating the team to improve. However, sometime this can introduce a tunnel vision into the iteration. So, therefore though dedicated research time, the programmer can expand their search ranges. This can given them deep information about the development.

In order to make use of research time effectively, it is required to concentrate on the specific task, avoid interruptions, and take help of project manager with respect to task. It is also important task of programmer to select a specific topic on which the research is to performed and generate the spike solutions for the respective topic. Also, better to create the standalone programs which represents the topic that has been researched.

Iteration Commitment is at Risk

Research time and addressing the technical debts are considered as important tasks which assists the programmers not just to develop the skills and the deliver the user stories more efficiently. But if iteration plan commitment is at risk, then it is better to temporarily tact these tasks so as to meet the iteration commitments.

Refactoring

In iteration plan, refactoring can be used to minimize or address the technical debt. But before initiated the refactoring process, once check the iteration plan whether it is proceeding with out any interruptions or difficulties. It it has no interruption then continue the process of refactoring. But if there are any, than it is better to focus on iteration commitments. If the period of addressing the technical debt is varying constantly then the iteration commitments can be made easily and on time.

Incur Voluntary Overtime

Every iteration brings there own challenges along with them. As it was said, changing the period of addressing the technical debt is helpful to iteration process. If needed the programmers can work for extra hours to compensate other works and meet the commitments. Also, the team can take help of other team members.

Eliminate Research Time

In iteration plan, sometimes the problems are important and valuable that cannot be eliminated from the plan. So it is better to delete the research time. When the problems are too big then deleting research time may not bring any change in the plan.

Slack is considered as a best tool that assists in achieving the team commitments. It also provides time to complete the tasks which inturn increases the productivity. In such cases, it is important to be vigilant because, though it handles the temporary problems, but it may make the systematic problems get unnoticed by the team.

If the team members work for overtime, then delete the research time and do not address the technical debt continuously for two or three iterations then it is said to be as overcommitted. It means they have used most of the slack so as to meet iteration commitments. It is difficult to allocate time for non urgent tasks because they are not so important to execute to meet the commitments. These type of situations can be seen in new XP teams. So, there it is better to use slack that helps to meet commitments and develop a successful XP project.

4.6 STORIES

Q26. Discuss briefly about stories and story cards.

Answer : Model Paper-I, Q9(a)

Stories

Stories can be defined as the simple and short descriptions of the designs or frameworks that are supported by the system. Essentially, the developers plan their work in the form of small, customer-centric pieces. In XP project, the stories are misinterpreted as requirements, usecases and narratives. But, in actual these stories are simpler and used for planning. Each and every requirement of stakeholder should possess a story. For instance, it should contain corresponding warehouse inventory report, complete screen demo option related to job fair, customized corporate branding on user login screen. However, according to the team these stories are not sufficient to implement and release working software because it is a small description of the feature (or) requirement. Ideally, a story should be a place holder which gives a detailed description with respect to requirements.

Even though, the stories are short but it consists of two important characteristics that are as follows,

1. It represents the customer value and is written in a terminology the understandable by customers. This is because customers are considered most appropriate person to narrate and write stories. So, the stories provides description about the end-result of the product that the customer requires rather than any implementation details.
2. It provides clear and complete criteria. In essence, the customers clearly define objective test. This test helps the programmers indicate the team the successful implementation of story.

Some of the examples of user stories are as follows,

(i) As a content writer, I want to create good content so that it can be helpful to the customers or students. The above story represents the value to the customers.

(ii) As a manager, I need to check a candidate profile so that I can handle the application process throughout the recruitment.

The above story represents clear and complete criteria. Some of the examples which are not considered as stories are as follows,

(i) An example "Automating the integration build" is not a story characteristic i.e., because it does not add any customer value.

(ii) Another example is setting up the staging server out side the firewall. It gives implementation details instead of end result also it is not described or written in customer terminology.

Story Cards

User stories are written on the index cards. Customer and stakeholders at the time of release planning held up a meeting for selecting the stories for the next release. In this process, it is difficult to select the good stories. To overcome this situation, index cards are used. These cards provides the story priorities that helps to work clearly towards the plan without any confusion.

Moreover, story cards are considered as one the core component of an information work space. Once the meeting related to planning is completed, the cards i.e., release planning session, place all the selected user stories are placed on the release planning board so that every team member can be aware of it and read it. In every iteration plan, place the respective story cards that are to be delivered at the end of iteration on another iteration planning board. These two planning boards should be visible to all the team members for easy understanding about the progress of the development.

Essentially, these index cards helps the developers to be responsible toward their work as they have to answer the stakeholders to provide quick reaction to stakeholders. This approach provides an easy way to write the information or suggestion given by stakeholder. The user stories can be altered if there is any changes in the customer requirements. This decision is made by the product manager and stakeholder. If they decide to add up a new story to the release plan, first they approach the programmers to estimate the story. Programmers provides the estimation and as small description of the story on the card. Once this process is completed the stakeholders and product manager include the story card into the release plan.

The index cards are consider as simpler and effective compared to computerized tools. These cards enables us to set up, handle and change the process without any configuration or programming easily.

Q27. Write short notes on,

(a) Customer-centricity

(b) Splitting and combining stories.

Answer :

(a) Customer-centricity

In general, stories should be written in the form of customer-centric. In essence, they are written on the basis of on-site customers opinion and also it should provide benefit to the customer in some or other way. The on-site customers are responsible for prioritizing the stories accordingly. In the planning game, if a story does not represent any value, then it should not be included in the plan. In customer-centric stories, there is no such story which is related to technical issues such as 'creating a build script' etc., because customers are not aware how to prioritize it. But, programmers helps customers by informing where the technical stories should be placed in the plan. This results in the interruption of the project scope. In order to include any technical considerations for the story add it at the time of estimation for the respective story.

Customer-centric stories are more helpful to the on-site customers compared to end-users.

(b) Splitting and Combining Stories

Stories can be of any size but the estimation becomes complex when the stories get too short or too long. In order to overcome this situation divide the large stories and combine the small stories. The size of each story should be is based on the iteration velocity. In each iteration, atleast 4 to 10 stories should be completed. According to this velocity number divided and combine the stories respectively. For instance, if a team is operating at a velocity of 10 days on every single iteration then estimation of splitted stories is more than 2 days. Subsequently, integrate those stories which have velocity less than half day.

The process of combining the stories is simple and easy. It involves combining multiple story cards having similar estimate and writing new estimate in front. On contrary, the process of splitting the stories is complex. This is because, it drives the developer from vertical stripes and releasable stories. This can be avoided by creating new stories at every step in the previous story. Although, it is functional but this approach entails story clumps. The remedy for this is that, the team should consider only the essence of entire story. Here, all other issues should be ignored and then write them in the form of new stories.

Some of the various ways to divide the stories are given below,

1. Divide the large stories from one end towards the other end boundaries that are sustained by the story.
2. Divide the large stories depending upon the operations, involved in the story.
3. Divide the large stories into small stories based on CRUD (Create, Read, Update, Delete) operations.
4. Divided the large stories by keeping apart the functional and nonfunctional elements.
5. Divide a large story into a small one, if the small stories posses different priorities.

Q28. Discuss briefly about special stories.

Answer :

Special Stories

The stories provide many capabilities to the software. Each and every task that requires the team's effort and does not comes under their routine work demands a story.

Various types of stories are as follows,

1. Documentation Stories

In XP project, team requires very less documentation to carryout the work, but essentially the task of documentation can not be ignored completely as documentation is required for other reasons. The stories that are related to documentation are referred as documentation stories and they are customer-centric and determine a specific completion criteria.

2. Non-functional Stories

The terms performance, scalability and stability are nonfunctional requirements. They are mostly scheduled along with stories. Subsequently, these stories should have clear and completion criteria.

3. Bug Stories

It is necessary that the team should resolves the bug issues prior to declaring it as "done done" i.e., completion. But, if there are any few bugs left to resolve, then schedule these bugs with the story cards in essence fix one or many user editing bug. These bug related stories are referred to as Bug stories. These stories should be resolved quickly to maintain bug free code. It is difficult to estimate the bug stories. Generally, it takes more time to evaluate the software and estimating the correct time becomes difficult. So, in this case, use timebox concept i.e., which fixes a specific time to work on a particular story.

4. Spike Stories

In some cases, programmers fails to estimate a story because they may not aware of the technology that is required to implement the story. So, in order to deal with this situation, create a story to 10 researches on that technology. They make use of a spike solution to perform research on the respective technology. So, the stories working on those technology is referred to as spike stories.

Consider these stories to achieve the goals. In essence, the team should work on the research story to estimate for the real story but not to find all the details and to fix the problem. If the programmers take more time to estimate a story then timebox concept can be used similar to the bug stories.

Estimating

Inorder to get estimate of spike stories, it is necessary to get time scheduled by the programmers. However, it is not mandatory for programmers to schedule time for normal stories as they can get scheduled any time.

This is considered as an overhead corresponding to iteration because of support requests and interruptions. If they think estimation is consuming much time in the form of interruption then create a new story card to estimate whenever it is comfortable. If there are multiple number of new stories to estimate, then create a new story card to estimate the large stories.

Meetings, Architecture, Design, Refactoring and Technical Infrastructure

Meeting are considered as a part of the iteration. If there are any time commitments such as training, off-site day etc., then create a new story for these things. Stories should not be created for technical information. Because technical tasks are considered as a part of implementing stories and estimates.

4.7 ESTIMATING

Q29. Define Estimating. Explain what works (and doesn't) in estimating.

Answer :

Estimating

Estimating can be defined as the process of predicting the time require to accomplish the story or task. Ideally, the process of estimation is considered as the most difficult task. For some programmers, the task of estimation becomes inaccurate. To get good estimates, they adapt the concept of padding, however this is also in approach because in padding the estimates are multiplied by numbers.

The reason why the estimating process is so difficult is because the programmers hardly predict the time duration for a particular task. Subsequently, if a programmer can estimate that a particular task can consume eight hours to get completed but or it can take 2 or 3 days as the task are programmer has to handle any constant interruptions.. However, things can go worst if the programmers works on another task simultaneously as both consumes lot of time. In order to get good estimates, it is better to predict the effort required by the task instead of estimating time. And also prepare the estimates on the basis of ideal engineering days. These days are nothing but the number of days required to complete a task provided if there are no interruptions. These days are often referred to as story points.

Sometimes within the ideal time, the estimates will be inaccurate but they were constant. This problem can be addressed by the concept of velocity.

Q30. Write about velocity and iteration timebox.

Answer :

Velocity

It is known fact that the estimates are mostly inaccurate. Individuals perform and provide different estimates for tasks but their aggregate will be constant in value. In essence, the estimate might come up to half the actual time or could take 20% more than actual time. Each iteration holds different kinds of interruptions but the amount of time required for interruptions is same for all the iterations. Essentially, estimates can be converted into calender time by combining all the stories of an iteration. This is possible through scaling factor which uses the concept of velocity. It can be defined as the number of story points that can be completed in one iteration. It is considered as an easy but advanced tool and makes use of a feedback loop concept. It implies that every velocity of iteration relates achievement of team in terms of past iterations. It helps to handle the tasks effectively.

In an iteration, if a team is unable to complete all the stories within the estimated deadline then it results in the downfall or decrease of velocity. Therefore, the team's workload is also decreased. So, this helps the team to complete all the user stories in time without causing any delay in the succeeding week. In the same way, if a team is capable of completing more stories within the specified iteration deadline. It leads to increase in their velocity which inturn increases the team's workload to meet the team capacity.

Velocity is considered as an effective way to balance the team's workload. In a XP team, velocity is used to predict the plans with higher accuracy.

The Iteration Timebox

The team velocity operates upon rigid iteration timebox. In order to maintain adequate team velocity it is important that the team should not consider the unfinished stories at the end of the iteration. In addition to this, the team should not exceed deadline. Some times, iteration deadline is changed to complete the stories in time and also to increase the velocity. But, this increase of velocity results in disruption of the stability feedback cycle and thus reducing the team's ability to work for commitments.

In the initial iterations velocity might be unstable so it can consume three to four iterations to reach stabilized state. The stable velocity should be maintained consistently in every iteration. Subsequently, iteration slack can be used to check whether the team is executing the commitments in every iteration. If the value of team velocity changes frequently then evaluate or check the iteration for any issues.

Q31. Explain how to make consistent estimates.

Answer : Model Paper-I, Q9(b)

Generally, experts always make automatic, good and consistent estimates. They do this by implementing a technique or method to make consistent estimates. In the process of estimation, select any positive and single value in which the story gets completed without any interruptions. Also the team should estimate that is there any chance to combine with other members with in the team to perform a task. This process rules out the requirement for adding extra time for the estimates and also for providing probabilistic range. This is because, velocity provides padding (adding extra time) for short-term estimates while the risk management provides padding for long-term estimates. The two consequences may occur by following this method:

The first one is that, in the process, if a person is an expert and he can automatically make good and consistent estimates but do not possess confidence about his ability to make good estimates. Then in this case, it is sufficient to consider any random numbers that is consistent with other estimates.

In the second one, if a person is not an expert then it is better to become an expert gradually inorder to make good estimates. An expert is also considered as an beginner but with good knowledge and experience. In order to become an expert, focus in making multiple estimates with the help of short timescales and also consider the respective results.

In the estimation process, all the programmers should get involved and ensure that atleast one customer is made available to clarify the programmer questions. In this process, once the required information is collected the programmer who has good experience and knowledge will start estimation. Subsequently, if there is any doubt or confusion regarding the provided estimate then discuss, analyze and resolve it in the upcoming discussion. After finalizing the estimate, it is better to note it on the card.

Initially, the team members may have multiple ideas regarding this estimates which may lead to inconsistency. So, if a discussion is made jointly then the programmers can adjust the estimates and clear the confusion among team members in first several iterations only. There is also another way to make consistent estimates which is to compare the estimates with the actual time that is essential time to complete each story or task. Thus, providing a feedback can draw more consistent estimates.

Q32. Explain briefly how to estimate stories and iteration tasks.

Answer :

The process of estimating stories should be performed in terms of story points. Initially, assume that story point is an ideal day. It means estimating the effort to implement a user story. During estimation, focus on the engineering tasks that are required to implement the stories. It is better to interact with the on-site customers to know about the outcomes and emphasize on those points that may cause problems to estimate. If there are any over-priced features then suggest or provide the other better alternatives to the customers. The estimation process can be performed naturally if the person has knowledge of the project. This avoids the step of performing engineering tasks. It is normal that the team usually focus on similar stories instead of ideal days.

In order to make an accurate estimates, additional information is required from the customers. If the customers are not available then mark it on the card to get information later. Inorder to do technology research, spike can be written instead of estimate and spike stories.

In the iteration planning process, programmers develop engineering tasks which helps them to execute the user stories in the respective plan. Each and every engineering task is considered as rigid and technical task.

Also, perform engineering tasks in ideal hours instead of ideal days and include all the necessary things that are required to complete a task such as testing, customer review.

Q33. Explain the circumstances when the estimation process is considered as difficult. Also how to explain estimates to customers.

Answer :

Circumstances of Estimation Process

The following are the cases where the estimation process is considered as difficult.

1. In some cases, programmers completely understand the requirements and become experts in intended technology. This makes them capable of estimating stories within a minute. However, if they do not understand properly, they need to discuss with customers and get clarified regarding the questions. So here in this case, the estimation might take more time than the expected.
2. Ideally, the cause of slow estimation is due to the customer preparation. Inorder to carry out estimation programmers ask questions regarding a particular estimates to which the customers are not prepared. So, there are chances that the customers may not agree to the response and demand more time to operate an it. There is another way to handle this type of situations i.e., conducting a customers meeting where the customers discuss, analyze and work on the issue to provide solution or mark the estimate to work on it later. Mostly all the customers are unaware of the questions in first few iterations. And later as the iterations proceeds, the customers become aware of the programmers questions.

3. The lack of knowledge and experience in programmers may result in slow estimation process. In essence if the programmer do not understand the problem domain then they start asking many questions corresponding to it inorder to make an estimate. Additionally, if the problem of inadequate customer preparation occurs combinely then it takes more time to make an estimate. Programmers analyze the requirements in order to make an estimate. Anyhow the issues which make small changes in the estimate are considered as important. It requires lot of effort to analyze which details are important and which details can be ignored.

4. In some cases, programmers pay over attention to minute details of requirements. This scenario occurs when they are hesitant to make estimates. This type of behaviour in programmers can be seen when they face difficulties and feel pressure from the company side. With this, the estimation process become more difficult.

Explaining the Estimates

It is important to explain the estimates to customers and stakeholders. There are chances that after explanation, they provide the feedback to the team members regarding the estimates. Some times the customers may feel disappointed and provide negative feedback in offensive tone. Such situations can be handled by ignoring the tone and considering it as requests. In some cases, the feedback helps the team to emphasize and develop the features with high value and low cost elements. Corresponding to customer ideas, the team should be prepared to explain the strict questions raised by customers related to the cost. The response should be should be such that they should safeguard their company's honor and also list the points in reasonably.

Also, the programmers can explain the issues encountered while developing the respective feature. Then provide the other alternatives that may reduce the cost which inturn reduces the scope of the story.

Moreover, there are chances that customers may not be convinced with the explanation and might ask for additional information or questions. So, it is duty of the team to consider these questions as simple requests. In cases, where the team is not aware of the response then it is better to accept and change the estimate. Subsequently, these questions raised by the customers might create doubt or confusion about the estimate, so it is better to change only if it is really worth. This can also be refused.

Q34. What are the ways to improve the team velocity?

Answer : **Model Paper-II, Q9(b)**

Some of the ways that enhance the team velocity are as follows,

1. Reduce the Technical Debt

In general, technical debt is considered as a major technical issue which effects the team productivity. If there are any productivity issues then there will be no improvement in velocity. In order to overcome this issue, focus on the higher quality code and the velocity will be improved automatically. But, teams facing acute technical debt consumes months or years to cleanup. So, it is better to resolve the technical debt issue incrementally. Iteration slack can be used to solve the issue. It is not possible to notice a change or improvement in just few months.

2. Enhance Customer Involvement

The programmers either make some random guess or wait for the customer feedback when customers are not available it is better to engage the customers in the meetings. So that they are always available to clarify the programmers doubts. If they are not available, it may effect the velocity. So, it is good if the customer is present to clarify or answer the programmer questions.

3. Support Energized Work

The exhausted, stressed out programmers are prone to commit errors in the programming. And such mistakes fetches high recovery cost. This is because, they don't put full effort to work. It may result in decrease in velocity. So, it is better to avoid the company's pressure and initiate a no-overtime policy.

4. Offload Programmer Duties

As the programmers are considered as the constraint for the team, assign only project related tasks to them. Also, the team should make it a point that they should stop wasting time on unnecessary things such as meetings, interruptions and non-project related tasks. It is also a good idea to hire or appoint someone as an administrative assistant to handle or manage all the non-project related tasks.

5. Provide Required Resources

In order to improve the team velocity, provide the required resources to all the programmers in the team. If there are insufficient resources, there will be unnecessary charges and may not deliver proper features. It is better to spend on equipment cost rather than paying for programmer as double charges.

6. Add Programmers

The term 'velocity' signifies the number of programmers present in the team. If a project team has well and experienced staff to operate effectively then adding people or programmers doesn't bring any difference. If it is required then only carefully add the programmers. Some times, by adding or hiring the junior programmers into the team may decrease the team productivity. And also if there are large teams then the communication can becomes a challenging factor which inturn decreases the team productivity. So it is better to maintain a good team size to achieve productivity.