Infosys®

Building Tomorrow's Enterprise

Infosys Global Agile Developer Certification

Agile Planning and Estimation

Study Material

www.infosys.com

Revision History

Version	Date	Remarks
1.0	Dec’2014	Initial Version

Table of Contents

1 INTRODUCTION 1

2 PROJECT LEVEL 3
2.1 Planning 3
2.1.1 Product Backlog 3
2.1.2 Workshops 3
2.1.3 Application Life Cycle Management 4
2.2 Estimation 5

3 RELEASE LEVEL 7
3.1 Planning 7
3.1.1 Determine Objective of Release 8
3.1.2 Estimating the User Stories (Size) 8
3.1.3 Prioritizing User Stories 8
3.1.4 Selecting stories and defining Release date 8
3.2 Estimation 9
3.2.1 Wideband Delphi 9
3.2.2 Estimation by Analogy: 9
3.2.3 Planning Poker: 10
3.2.4 Relative Sizing 10

4. SPRINT LEVEL 12
4.1 Planning 12
4.2 Estimation 13

5. GOOD PRACTICES 15

6. PRACTICE QUESTIONS 17

7. APPENDIX 18

8. REFERENCES 19

Table of Figures

FIGURE 1: PLANNING AND ESTIMATION AT VARIOUS STAGES OF AN AGILE PROJECT 2

FIGURE 2: SAMPLE ALM TOOL (RALLY) 4

FIGURE 3: PROJECT LEVEL ESTIMATION 6

FIGURE 4: RELEASE PLANNING 7

FIGURE 5: RELEASE LEVEL ESTIMATION 10

FIGURE 6: DEPICTION OF SPRINT PLANNING 12

FIGURE 7: SPRINT LEVEL ESTIMATION 14

FIGURE 8: FACTORS CONTRIBUTING TO STORY POINT ESTIMATION 15

1 Introduction

Planning and Estimation are essential in software projects to achieve predictability, reducing the risks involved and to set a basic expectation to all stakeholders. Planning brings lot of focus on preparation and forecasting whereas Estimation is a process to forecast project related variables i.e. effort, scope, schedule, etc.

Planning

Planning is required irrespective of the project management methodologies that the team follows, whether it is Waterfall or Agile. Planning gives the project team a perspective on how to meet the objective in a systematic way. It helps project stakeholder to keep tab on the project progress and investments done.

As Mike Cohn defines it, "Agile planning balances the effort and investment in planning with the knowledge that we will revise the plan through the course of the project. An agile plan is one that we are not only willing, but also eager to change". This concept exists mainly to avoid the weakness of the planning. These weaknesses include:

- Concentrating on activities rather than delivered features
- Ignoring the prioritization
- Ignoring the existence of uncertainty
- Using estimations as commitments

Such weaknesses in planning make the team not able to cope up with the project dynamic environments. For that, the planning has to evolve to become Agile as well.

Estimation

Software projects are typically controlled by four major variables i.e. Schedule, Scope, Cost, and Effort. Unexpected changes in any of these variables will have an impact on a project. Predictability and De-risking of software projects is essential for the success of IT services organizations. Estimation is a process to forecast these variables to develop or maintain software based on the information specified by the client.

There are three main challenges faced during estimation i.e. Uncertainty, Self-knowledge, and Consistency of Method used for Estimation. Usage of standardized and scientific estimation methods for estimating size, effort and schedule, helps towards maintaining minimal variance between the planned estimates and actual values thereby achieving maximum estimation accuracy. This provides better client experience. No single method fits all estimation needs for a project. It is important to have different methods of estimation for different stages.

Planning and Estimation in Agile projects brings a lot of focus on preparation and forecasting. Both these activities are done keeping business context in mind and measurable value delivery is committed to the client. Therefore, required planning and estimation are recommended in Agile from the start of the project, to ensure better risk coverage and higher predictability.

Planning and Estimation in Agile projects are generally done at three different stages, which are:

1. Project Initiation Level
2. Release Level
3. Sprint Level (i.e. an iteration within each Release)

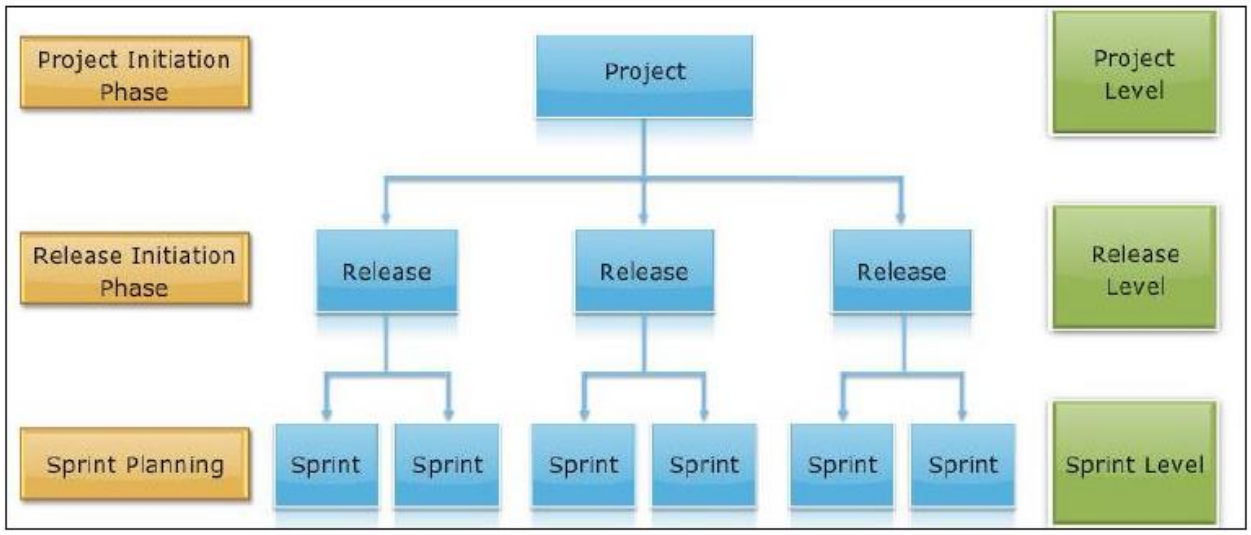


Figure 1: Planning and Estimation at various stages of an Agile Project

The objective of this document is to provide an insight on the importance of planning and estimation at the various stages of the project. This document provides a view on each stage of project for planning and estimation, and how their combination ensures predictability. Multiple levels of planning and estimation stages in Agile project, helps not only in achieving the goal but also in de-risking the projects at various phases.

2 Project Level

2.1 Planning

Planning activities at Project level takes 360 degree of project view in terms of understanding the big picture, overall vision, roadmap, value delivery, planning for better risk coverage, creation of Product Backlog, defining the Releases, etc. This is the highest level of planning that focuses on impending activities of the project and its ultimate goal. It creates the macro image of the project and its importance for the stakeholders which helps project teams to create priorities and estimations. Product Owner (or Client/Business manager) is the primary stakeholder involved in this activity.

During this stage, Product Owner (PO) explains the project/delivery team about the strategies which need to be followed by them to deliver the potentially shippable product on time. The team's role here would be to understand the strategic plan clearly and full fill the needs of the Product Owner to the fullest. Based on the operational standards set by the PO, the project team which is cross functional must do the project work with frequent work replenishment for solution delivery.

Once project goal is known, then it is required to define how to achieve that goal. It is about breaking the requirements into various blocks to build, validate, and release the final deliverables. This helps to plan and estimate various releases as per the feature requirements. Product Owner plays an essential role over here. This planning leads to generation of Product Backlog, which then is used in further planning i.e. Release Planning. Ignoring the Project Planning activity would lead to developing an end product which is not useful or not as expected by the stakeholders.

In practice, planning at Project level involves lot of components which provides an overall control in order to achieve predictability in each delivery, mitigating the risks, and setting basic expectation with all stakeholders. These components are related to infrastructure planning, quality management planning, environment setup planning, tools and reuse planning, build automation and continuous integration planning, etc. However, at a high level the important aspects of planning includes creation of Product Backlog, having Requirements Workshop, creation of Release plan by defining the releases and planning of Agile practices through Application Life Cycle Management tools, that needs to be adopted in the project.

2.1.1 Product Backlog

In order to meet the defined goals of the project, it is very important that the business clearly articulates requirements, scope, and how they are prioritized. Also, it is very important that development team understands these critical factors before the start of development. Product Backlog is a prioritized list of simplified requirements which is necessary to accomplish the intended delivery of the product/feature. It is a key input for planning. Product Owners often prepare the backlog with the consultation of business strategists and end users. The prioritization is often based on business value, release date, interdependency and is maintained in a common place such as centralized MS Excel file, ALM tools, etc.

2.1.2 Workshops

The main purpose of conducting workshops is for gathering, understanding and prioritizing requirements, and defining high level plan for the Releases mainly during the Project Initiation phase. Workshop is an event where a group of people collaborate to present different ideas to reach a predetermined objective supported by an impartial facilitator. It is not the same as brainstorming. In a workshop, there should be a clear idea on what is required and what should be the outcome. It is usually conducted to compress timelines (i.e. defining the Releases) and reach a consensus/decision faster. However, workshops are worthwhile to do once in every 6 months, just to review the direction of the project, restate the objectives and realign the teams.

2.1.3 Application Life Cycle Management

Application Lifecycle Management (ALM) aids in managing Agile projects by providing mechanisms to deliver software more predictably and to drive business value. A typical ALM tool supports development process, project management, metrics and dashboards, compliance reports, sharing artifacts within team etc. Agile ALM tools are used for planning and estimating user stories and sharing it within team. It can be used for building a Product Backlog, Sprint Backlog, establishing team commitment and velocity, visualizing team activity and project progress via Burn Down charts and reporting on team progress. Team can prioritize user stories in order just by dragging up/down in backlog. Also, teams can efficiently capture, self-assign and manage their tasks using a good ALM tool.

Application Lifecycle Management (ALM) tool has significant importance when the teams operate from different locations and time zones. It facilitates the following activities of the distributed Agile project:

- Product and Release Planning i.e. for planning and managing Agile requirements, epics, stories, and goals across multiple projects, teams, and portfolios.
- Sprint planning and Tracking i.e. for iteratively planning user stories, defects, tasks, tests, and impediments in a single environment.
- Managing project artifacts (Product and Sprint backlog items, Source Code, Development and Testing artifacts, Test reports)
- Managing reports (Release and Sprint Burn Down charts)
- Tracking process information (who does what and when)
- Managing communication tools (discussion forum, chat, information-sharing, notifications to users etc.

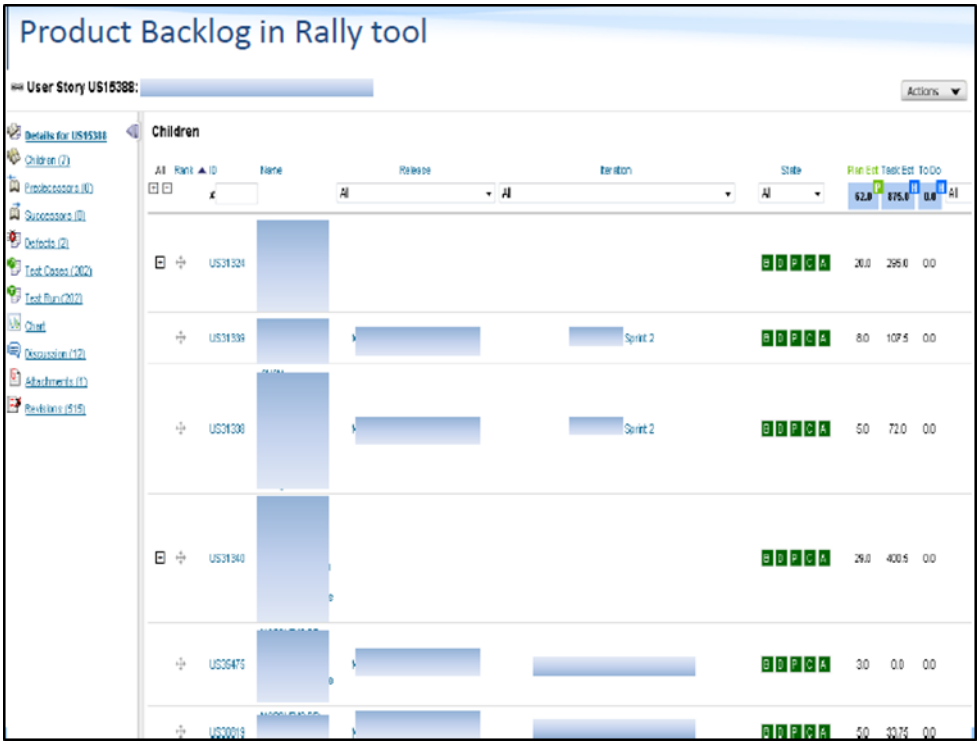


Figure 2: Sample ALM Tool (Rally)

Source: http://teamwiki/pub/InfyAgile/Agiletools/Agile_session_-Rally.pdf

Most of the recent Agile ALM tools are visual, instinctive, and collaborative and would need little time to learn. Please refer Appendix section for the brief description of various ALM tools.

2.2 Estimation

During the Project Initiation stage, functional requirements are at high level, or the details of requirements are not well-formulated and documented. Many requirements are still at Epic or Feature level in the Product Backlog. Estimation is required at this stage, as this will help in prioritization and planning the Releases.

Different estimation techniques can be used at this stage. Some of them are

- QFPA (Quick Function Point Analysis)
- Use Case Points
- Complexity based estimation
- COCOMO (Constructive Cost Model)
- COSMIC FP

Further details on above techniques are available at http://10.185.51.68:9005/Home

Estimation during Project Initiation stage should be either a group activity, results should be compared and differences to be resolved. All the project assumptions and risk should be highlighted clearly.

This process starts with gathering additional details about the Epics (i.e. high level User Stories / requirements) mentioned in the Product Backlog during the project initiation. Interactions with the Product Owner (or Client manager) will be necessary to get more information about it and Epics are sized using “Quick Function Point Estimation” (or any similar technique) and the overall effort is derived based on several inputs.

Typically the team for performing this activity will include:

- Product Owner – The product owner briefs the entire team about the project objectives, business initiatives and the corresponding timelines or deadlines for the completion of the project
- Clients & Business Stakeholders - Ensures that the software developed will meet the requirements and the correct/expected functionality. They are responsible for participating in the estimation meetings, provide clarification to any doubts/queries that arise during this activity
- Project Manager – Responsible for performing estimation along with the developers and designers
- Developers and Designers – Who design and build the application to meet the requirements
- Testers – Who validates the system from functional and non-functional requirements perspective

The general approach for estimation at the project level is illustrated by the diagram below:

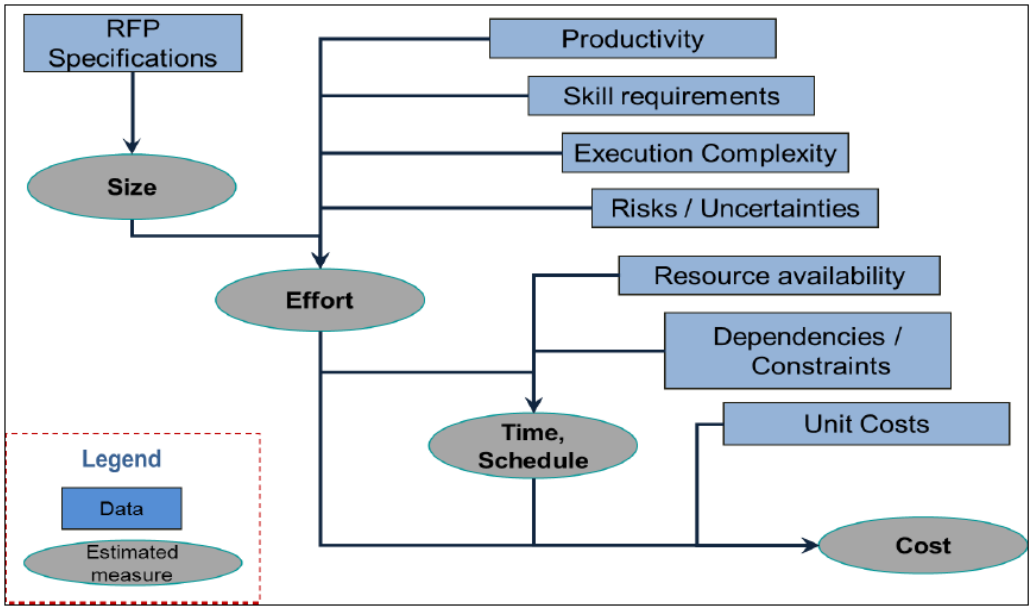


Figure 3: Project Level Estimation

The Project Level estimate will give details about the

- Approximate number of Releases/Sprints required before deployment
- Approximate end date of project
- Approximate staffing plan
- Approximate effort plan etc.

The project level estimates gets refined regularly at planning phase of each Release/Sprint.

3 Release Level

3.1 Planning

Release plan serves as guidepost towards which the project team can progress. Without release plan, project team would end up moving endlessly from one Sprint to another without knowing where to stop (i.e. logically). At the end of every Release, a shippable product is delivered and ready to be deployed. So it is important for a project team to understand and follow the Release plan.

Release planning is not about deciding who does what or assigning team members to tasks. It is about organizing and scheduling user stories (i.e. functional/non-functional requirements) that are to be converted into the working software by the team and delivered for that release. While doing the Release Planning, the team decides number of Sprints, for each of the Release in a time-boxed manner. The duration of Sprints is mostly one to four weeks.

Release plan can also mention some key assumptions like how long a Sprint will be, how big the team will be, who will be working in the team, when the first Sprint will start, when the last Sprint will finish etc. Releases are recommended to be planned anywhere between three to six months based on business context for the annuity (long duration i.e. more than one year) projects and one to three months for the non-annuity projects.

Release planning is based upon the historical data (if available) to do the initial sizing estimation of the release in terms of story points. To achieve a highly efficient Release plan, it should be established based on:

- Satisfying all the stakeholders needs/expectations
- Readiness to provide maximum business value and ensuring right sequence of releases
- Capability of completion of the scope within the available capacity

It is important to plan for the Releases, as it provides clear picture to all the stakeholders, on what is expected to get into production, in due course of project execution. Release planning is an integral part of Agile software development and is done at the beginning of each Release. This helps the team to deliver the expected project output in an incremental basis to the client which helps them to get an early feedback so that future planning gets more aligned to the need of the client. While doing the Release Planning, the team decides number of Sprints, for each of the Release. Sprints are time-boxed where a set of user stories get converted into the working software by the team. The duration of Sprints is mostly two to four weeks.

The duration, goal and content for each Sprint and Release is defined and agreed by all stakeholders. However this is a continuous process till implementation, because as the project progresses there could be addition and deletion of requirements. Hence in the beginning of each Release and Sprint, the planning for the same is revisited and scope is revised. The project team can decide to have additional Sprints or Releases during the course of the project depending on the dynamics at that point of time.

Following figure depicts various activities involved in Release planning.

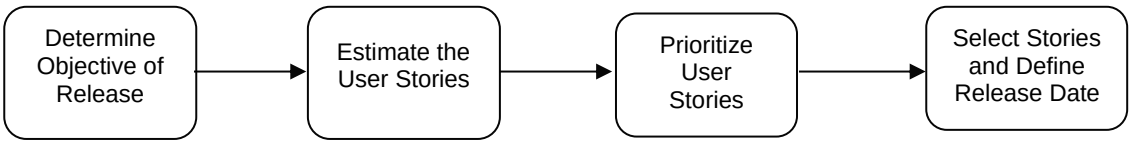


Figure 4: Release Planning

3.1.1 Determine Objective of Release

While the Release is in progress, it is important for the team to know the evaluation criteria to check if the project is going in the right direction or not. There are various factors on which the project success can be evaluated like time of completion, ROI (Return on Investment), etc. Product Owners definition of success will be generally driven by combination of these factors. Some of the leading indicators to asses these factors are schedule, effort or cost. Product Owner will generally discuss the desired objective in Release planning meeting. Mostly the objective will be either time\date driven output or functionality driven output. When the target is time\date driven output, Product Owner will expect the release to be finished by said date. If the target is driven by functionality, Product Owner will expect 'x' functionalities to be completed by the end of the Release.

3.1.2 Estimating the User Stories (Size)

Product owners will always have a wish list of items which they would like to be targeted during every upcoming Releases of the project. It is important to identify and estimate the size (complexity) of the Product Backlog items (i.e. User Stories) based on the business value associated with them. These estimates provides key inputs that needs to be considered for the Release planning to define the time lines for the Release as well as for the duration of the Sprint length.

3.1.3 Prioritizing User Stories

Projects always faces a situation where there is either too much of work to deliver or too less time to deliver. So it is always advisable that product owner prioritizes the user stories available, so that Agile team can work on it accordingly based on the business necessity. One of the best practice for Product Owner is to consult the team while prioritizing, sequencing user stories for the Release. MoSCoW is one of the technique that can be used while conducting the prioritization of requirements. MoSCoW stands for 'Must have', 'Should have', 'Could have', and 'Won't have')

3.1.4 Selecting stories and defining Release date

By this step, the agile team would have information about User Story estimates (in size/complexity), the prioritized user stories (based on business necessity), and required duration for each Sprint. This inputs would help in Release planning which meets the objective as defined by the Product Owner (or Client) for a Release. Entire project team along with Product Owner and any other stakeholders (i.e. end users, etc.) are involved in this activity and the Release end date is defined collaboratively with common consensus.

If it is a functionality or feature driven project, the estimates for all the user stories identified by Product Owner and team are summed up and divided by average velocity of the team from the past releases (if available, else taken from similar projects). This will give number of Sprints required to complete the desired set of functionalities based on which the Release end date can be defined. If it is a date-time driven project, number of Sprints are defined based on the working days. In this case, the Release end date would be defined based on logical end of the various modules/functionalities in the software or the user stories from prioritized list.

3.2 Estimation

During Release planning, prioritized Requirements which are in the form of User Stories, are taken from the Product Backlog. These user stories are estimated so that Release can be sized and planned. Estimation is done in Story Points for these user stories. *[Note: Story Points indicates absolute number assigned to each User Story based on the complexity in Relative (or comparison) to other User Stories in the Product Backlog]*

Estimation during Release Planning is done by the entire team using one of the following techniques –

- Wideband Delphi (or Consensus approach)
- Estimation by Analogy (or Extrapolation)
- Planning Poker

3.2.1 Wideband Delphi

Following are the steps followed while performing estimation of user story points using Wideband Delphi method:

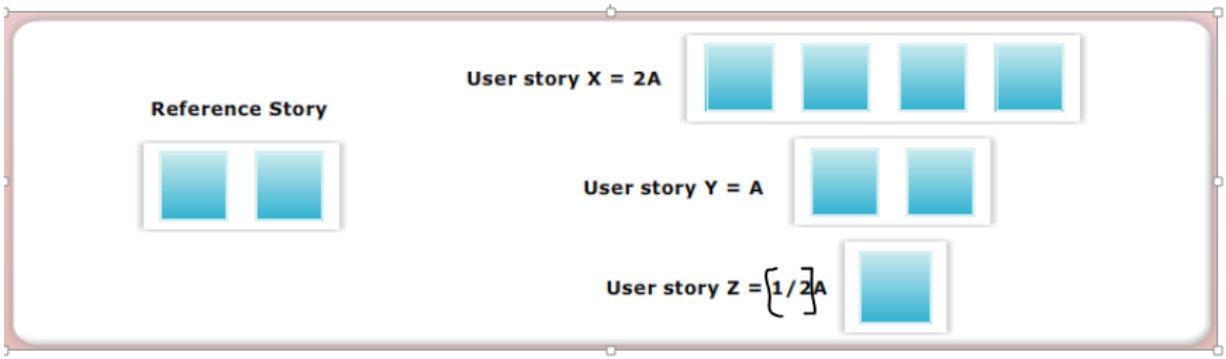
- The coordinator speaks to the team and the Product Owner about the User Stories.
- Individual members in the team estimate the number of person days required to develop each functionality.
- The coordinator compares the individual estimates and discusses it with the team.
- Process is repeated until there is no significant difference in the estimation by the group.
- Typically this exercise can be completed within 3 rounds.

3.2.2 Estimation by Analogy:

Following are the steps followed while performing estimation of user story points using Estimation by Analogy method:

- Estimation is done for a user story based on its relationship with one or more reference
- Similar sized user stories are grouped and estimation is done for them together
- For example, a user story “Ability to add payee” is estimated for 40 hrs. If the complexity of user story “Ability to edit payee” is half of add payee, then the effort required for this story would be 20 hrs.

Following is another depiction of an example:



3.2.3 Planning Poker:

Planning Poker technique uses estimation cards, which is based on Fibonacci series i.e. 0, 1, 2, 3, 5, 8, 13, 20, 40, 100, and ∞ while estimating the story points for the User Stories. Planning poker includes all the team members, i.e. developers, testers, analysts etc. It is a relative sizing technique and based on consensus of the team. Following are the steps followed while performing estimation using this method:

- Each member / estimator has a deck of cards, with each card having a valid estimate.
- Moderator (who generally does not play) reads a story and it is then discussed briefly.
- Product Owner answers any questions that are raised.
- Each estimator selects the card representing their estimate and keeps it facing down (hidden).
- Cards are simultaneously turned over so all can see them.
- If estimates differ, then the highest and lowest estimator provides justification.
- Re-estimate until consensus is reached on the story points.

Release level estimation is done at the time of Release planning. The prioritized requirements are taken from Product Backlog which is in the form of User Stories. The User Stories are estimated in terms of Story Points during Release planning which focuses on estimating the size of the software to be delivered for that particular Release. The other inputs considered are project-level size, effort, cycle time and available skills. Based on this estimation, number of releases and total number of story points in each release is planned for overall project.

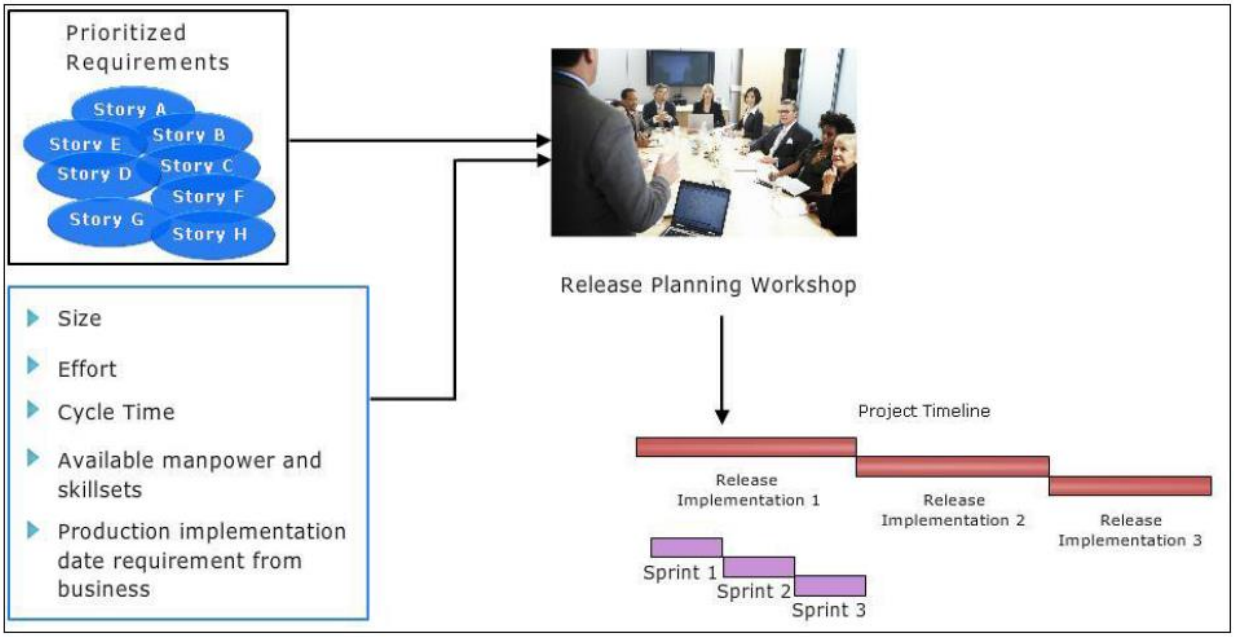


Figure 5: Release Level Estimation

3.2.4 Relative Sizing

The estimate for the user story (as detailed in Product Backlog) is developed by entire team utilizing experience of all members. Relative sizing is used at this stage. Following are some of the User Story Estimation methods (following 'Wideband Delphi or 'Estimation by Analogy':

- T-shirt sizing – Each User Story is classified based on t-shirt sizes as XL (eXtra Large), L (Large), M (Medium), S (Small), XS (eXtra Small) etc.
- Fibonacci series – Relative sizing is done based on Fibonacci series 0, 1, 2, 3, 5, 8, 13, 20, 40, 100, ∞.

To achieve quick estimation, most of the user story estimations are done based on Relative sizing. One small story is picked and considered as a base story, and all other stories are sized in relation to this Base story. Following is an example of Relative Sizing:

Ex1: Changing a static text on an HTML page is 1 Story Point. Now all other story points are relative to this.
So, if the User Story says
"Provide images in the HTML page in line with the static text and the format should be consistent across all the browsers".
This has to be relative to the base story and would probably team can estimate of 5 story points considering the complexity in testing multiple systems.

Ex2: Person-A asks Person-B a question *"How tall is Building-1"*? Person-B might give an absolute answer as *"50 ft."* or Relative answer as *"Taller than Building-XYZ"*.

4. Sprint Level

4.1 Planning

Release plan provides a high level view of what team intends to build, and what team intends to deploy at the end of the Release. It does not provide detailed view of how the team would plan to drive the work within Sprints. Once Releases and its Sprints (i.e. number of iterations within each Release) are known, team starts planning for each Sprint. This happens during the start of each Sprint i.e. first day of the Sprint and allows the team to be more specific on what they are going to deliver at the end of the Sprint. Team takes inputs from the Scrum Master and prioritized list from Product Owner. Team decides on what they can take from the prioritized backlog for this Sprint, detail out the tasks for each user story and assign it to themselves.

In Sprint Planning meeting, team will be focused more on what they need to do, to complete user stories selected for that Sprint. This meeting is attended by Product Owner, Scrum Master, all the team members like analysts, programmers, testers, database engineers, user interaction designers and so on. Output of the Sprint planning can be recorded in simple spread sheet or any ALM tool (Application Lifecycle Management), where team will have all the user stories of that Sprint and its tasks mentioned sequentially (i.e. Sprint Backlog).

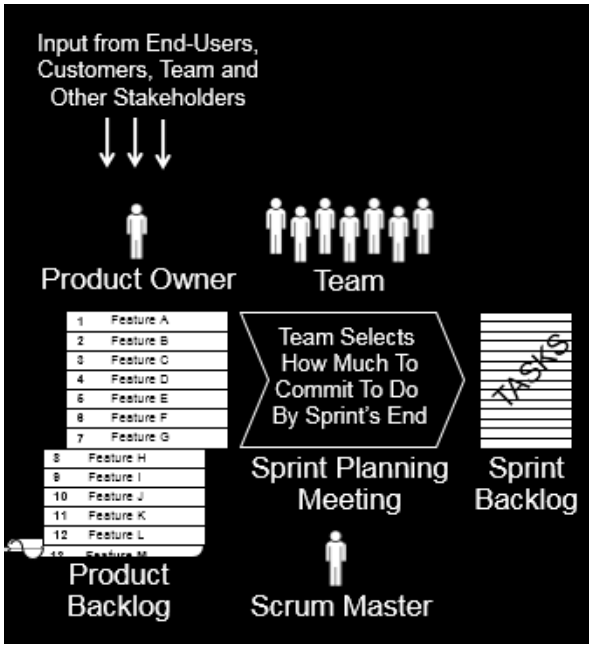


Figure 6: Depiction of Sprint Planning

Source: Pete Deemer, www.Goodagile.com

Team Capacity planning and user story commitment for a Sprint are always great challenges for a Scrum team. Before committing to a Sprint goal (ex. Velocity which indicates how many user story points to be committed to), it is important to know the current team capacity. Team capacity is calculated as per team size i.e. number of team members available for that Sprint and the availability of each member (i.e. number of hours/days they are available/could-spend for that sprint). Based on the Capacity of the Scrum team, commitment would be done for the completion of certain number of user stories (or User Story points) by the end of that Sprint. Based on this inputs and a consent agreement between the Product Owner, Scrum team and the Scrum Master, the tasks can be estimated exactly. In this way, the team capacity can be used for effective planning and estimation.

Sprint Planning is carried out at the outset of each Sprint. This helps the Team to bring focus on the Sprint goal. In the Sprint Planning, the team decides what it will build in the upcoming Sprint and how they will build it. User stories are broken down to tasks, task level estimation is done and the team commits to the Sprint goal. Product Owner, Scrum Master and the Team participate in the Sprint Planning. During this meeting, team velocity, previous velocity trends (if available) and available team capacity is to be considered for effective planning.

The Sprint Planning meeting consists of two parts, each one being a time-box, and assigned with half of the duration of Sprint Planning meeting.

Part 1: In the first part,

- Team collaborates with Product Owner and decides what will be delivered as part of the Sprint.
- Product Owner describes User Stories to team
- Details the interaction
- Reviews infrastructure and interface
- Explains the acceptance criteria

Part 2: In the second part,

- Team figures out how it is going to build this functionality into a product increment during the Sprint.
- Team selects the User Story
- User Story is broken into tasks
- Each of the tasks is estimated in hours
- Team commits on User Stories to be completed in the Sprint

Ideally, for a Sprint with 4-weeks duration, the Sprint Planning meeting happens for one day i.e. 8 hours out of which first four hours are for the Part-1 and second four hours for the Part-2. The time needed for the Sprint Planning meeting will be in similar proportion based on the duration of the Sprint.

4.2 Estimation

During Sprint Planning, prioritized stories from the product backlog are broken into tasks that can be estimated. Bottom-up estimation is used in Sprint Planning and tasks are estimated in days / hours, total of which gives the actual effort for the User Story.

Tasks can be defined related to design, coding, test scenarios, unit testing, test case execution etc. The set of tasks will also depend on the “Definition of Done” where Automation, Regression Testing, Style Guide updates, Code review completion etc. could be part of it. The team has to understand that Sprint success means all the parameters are set as part of Definition of Done.

In Agile projects, ‘Definition of Done’ determines conditions to be considered for a User Story and its associated tasks are completed, and ready for user acceptance. ‘Definition of Done’ provides a shared understanding among the project team members along with Product Owner. When someone in the team says that a task is finished, then everybody in the team has a common understanding of what it means. Through ‘Definition of Done’, the team adheres to quality standards.

Team members have to own the tasks and estimate for them separately. On a daily basis, task owner has to report the remaining (to-do) hours for the task, which is used to draw the Burn-down chart. One of the good practice during Sprint Planning is to identify the tasks for the user story which would not span more than half day. It implies, whatever tasks are identified by the team members, it should not take more than half day to accomplish it.

Every User Story including non-functional requirements should have well-defined Acceptance Criteria. A story is considered as “Complete” if it is completely developed and tested satisfying the Acceptance Criteria. All

these activities are listed under “Definition of Done” for the story. Sample activities include design, coding, unit testing, unit test cases, test scenarios, test cases, deployment, test data, testing, reviews, meetings, documentation, reporting, etc. They can be categorized as programming time and non-programming time. While doing estimation, effort required for all these activities should be considered.

It is asked many times if Tester should estimate for development effort also, and vice versa. As a team, all the activities for the stories should be estimated together (i.e. testers and developers). During Release Planning, estimation is done based on relative sizing and gets automatically considered. During Sprint planning, each task should be estimated by respective owners.

As part of Sprint Planning, Scrum Master, Product Owner and Project Team members start with prioritized Product Backlog to define the scope of the Sprint and create the Sprint Backlog. The Sprint Backlog consists of User Stories that would be accomplished during the current Sprint. The team together takes the User Stories from the Sprint Backlog and decomposes it into individual tasks that could be estimated.

Tasks are estimated in ideal time. Estimation of ideal time for Tasks translates the size (measured in story points) to a detailed estimate of effort. This effort is typically measured in terms of actual days or actual hours. Task estimation is meant for Sprint Backlog and its existence is within the Sprint.

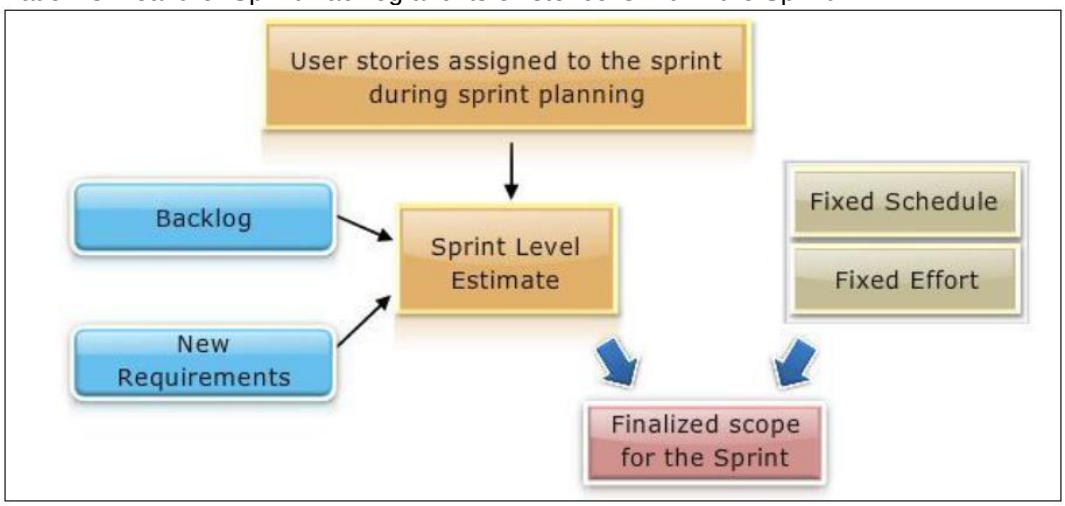


Figure 7: Sprint Level Estimation

For example, as the team does the Sprint Planning, they will pick a User Story which may be sized to 5 Story Points. Then they break it down into a number of smaller working tasks, such as separate work items for detailing, designing, implementing and testing. The purpose of this is to know how many User Stories the development team can commit in a Sprint. The development team must be comfortable with the commitment and the Product Owners must be confident that the team will deliver on that commitment.

Ideal Time Estimation is based on Bottom-Up Approach where the business requirements are broken down into low-level activity by the team members and each activity is estimated separately. Team members are asked to sign up for the tasks and estimate the actual effort, measured in hours or days, for their tasks.

When a team uses ideal time for estimating, they refer exclusively time required to get a feature or task done by the programmer compared to other features or tasks. The effort remaining is displayed in a common place (i.e. Burndown Charts) so that the team is aligned to meet the estimated goal.

For ideal time estimation of tasks, individual team members pick up an activity and provide estimates. If there is a disagreement in these estimates among the team members, then they discuss it and come to consensus.

5. Good Practices

1. **Estimation for User Story Points:** Story Point Estimations are reflection of 3 factors i.e. Volume of Work, Complexity and Doubt (i.e. uncertainty/ambiguity in solution). Agile team needs to collaborate and ensure that an estimate reflects the collective understanding of team. Therefore, it is important that project team prescribe a set of guidelines which should be followed during the process of estimation.

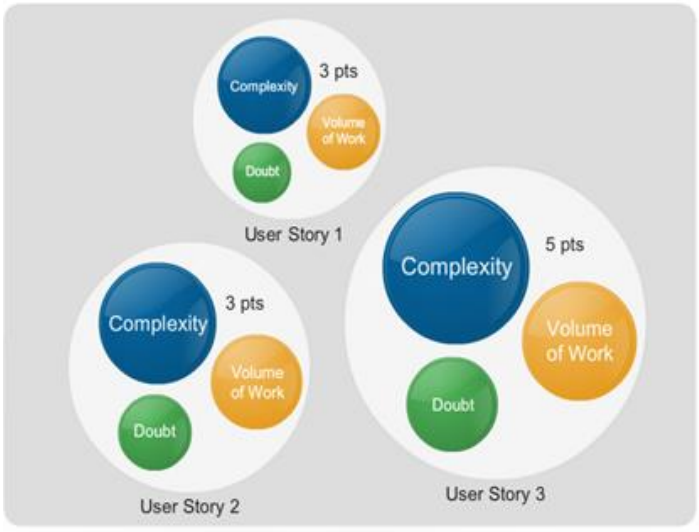


Figure 8: Factors contributing to Story Point Estimation

2. **Team Planning:** Involve the whole team for planning. There can be certain planning activities in a project for which prime responsibility can be with one person or a group. However, the whole team should get involved in project planning. With higher involvement, commitment is more from team members and they are inclined for higher value delivery

3. **Frequency of Estimation:** If estimation is difficult, Agile team increases the feedback from the Product Owner to get more clarifications in the following ways:
   - Shorten the time for estimation and increase it to take feedback about accuracy of estimate
   - Increase the frequency of estimating
   - Sketch out options and get feedback from the client before doing detailed estimation

4. **Re-Plan:** Re-plan often, at start of Sprint or Release. It is always better to re-plan at the start of each Release or Sprint. The relevance of the plan has to be evaluated at start of each Release or Sprint and plan is updated. Keeping the relevance of the current plan helps achieve the target.

5. **Develop Multiple Estimates:** If requirements and assumptions are unclear, the team develops multiple estimates.
   - Communicate the assumptions or constraints of the estimates to the relevant stakeholders rather than just the numbers
   - Discuss the constraints/assumptions with the relevant stakeholders (i.e. Client or Product Owner) and the client can provide feedback to better align the team's understanding with the business drivers

6. **Consider Learnings while Planning:** Acknowledge the importance of learning. At each Sprint retrospective, there is learning about the project which is discussed by the team. The team must acknowledge learning and consider it while planning.

7. **Validate Estimates:** It is always recommended for the Agile team to validate estimates by comparing them with other estimates/experiences, use simple rules, and intuitive decision making. Once the team has agreed

on an estimating unit, they should ensure to implement it consistently and stick to its original definition. Especially in the projects initial Sprints, everyone should resist the urge to try mapping these units to time units with any exact precision

8. **Leave some slack while planning:** When planning for a Sprint, It is recommended not to plan 100% of every team member's time on development activities. This is essential because the team spends time in non-development activities as well. For example, detailing out user stories whenever required, collaborating with other team members, attending daily stand up meetings, participating in backlog grooming sessions, clarifying requirements, updating ALM tool etc. While planning for capacity, time needed for events like Sprint Retrospective and Sprint Review also should be covered. It is recommended for each team member, to plan for approximately 6 hours of effort for development activities and the rest for non-development activities.

6. Practice Questions

Question 1

Planning and Estimation in Agile projects is NOT done at which of the following stages of the project?

a) Project Initiation
b) Release Initiation
c) Project Closure
d) Sprint Planning

Question 2

An absolute number assigned to each User Story based on the complexity in Relative to other User Stories is known as:

a) Quick Function Points
b) Use Case Points
c) User Story Points
d) Cosmic Function Points

Question 3

In the Sprint Planning, the team decides on how many user stories they will be able to commit for completion in the upcoming Sprint.

a) True
b) False

Question 4

The Project level estimate will give details about:

a) Approximate staffing plan
b) Approximate end date of project
c) Approximate number of Releases/Sprints required before deployment
d) All of the above

Question 5

Which of the following is/are INCORRECT about the activities involved in Release level planning?

a) Prioritization of the User Stories is done
b) Retrospective of previous Sprint is conducted
c) Determines the objective of the Release
d) Estimation for each individual Task is performed

7. Appendix

Below is the description of various Agile ALM (Application Lifecycle Management) Tools available:

Product Name	Product Vendor	Distinct Features
Version One	Version One http://www.versionone.com/Product/Agile-Project-Management-Tool-Overview/	Centralized planning and tracking, simplified team collaboration, improved project visibility
Rally	RallyDev http://www.rallydev.com/	Supports shared product backlogs, hierarchical Projects, Roll-up reporting, advanced analytics, Integrated requirements management and defects management
JIRA (GreenHopper)	Atlassian http://www.atlassian.com/software/greenhopper/overview/the-highlight-reel	Focus on Issue Tracking, Collaboration, Flexible Workflow, Scrum Board, Kanban Board, Velocity, Burndown charts etc.
Mingle	Thought Works http://www.thoughtworks-studios.com/mingle-AgileAgile-project-management	Captures and visualizes all team activity, shared workspace, customizable reports and dashboards, Burn down and Burn up charts, team velocity
Team Foundation Server (TFS)	Microsoft http://msdn2.microsoft.com/en-us/library/ms364061.aspx	Source control, Data collection, Reporting and Project tracking
IBM Rational Team Concert (RTC)	IBM http://www-01.ibm.com/software/rational/products/rtc	Agile planning, Source code management, Work item management, Build management, and project health, Integrated reporting and process support

8. References

- Mike Cohn, “Agile Estimation and Planning”, Pearson Education

- PRidE (Sparsh > WebApps):
  - ‘Guidelines for Early Lifecycle Sizing of Development Projects using Function Points’
  - ‘InfyAgile Global Estimation Guidelines’
  - ‘InfyAgile Global Guidelines for Writing User Stories’

- Websites
  - http://my.safaribooksonline.com/book/software-engineering-and-development/agile-development/9780137126347/why-agile-planning-works/ch22lev1sec9
  - http://sparsh/v1/myUnit/QD/QD_docs/IGAC_Estimation_in_Infosys_Global_Agile.pdf
  - http://www.infoq.com/articles/many-levels-planning-agile-project
  - http://www.allaboutagile.com/how-to-implement-scrum-in-10-easy-steps-step-3-sprint-planning-requirements/
  - http://www.scrumalliance.org/community/articles/2009/december/scrum-in-a-nutshell
  - http://www.scrumalliance.org/why-scrum/core-scrum-values-roles
  - http://www.dummies.com/how-to/content/team-roles-within-an-agile-management-framework.html
  - http://www.scrum-institute.org/Scrum_Roles_The_Scrum_Team.php

www.infosys.com