Planning Larger Projects

Agile differs from Waterfall in both how much planning is done and when it is done. Agile methodologies focus on frequent delivery in order to get frequent customer feedback and reduce the risks of late integration. Scrum teams should be formed independent of projects and could be working on multiple projects at a time through a single prioritized Product Backlog. As a project finishes up, the Product Owner is responsible for making sure enough is known about the next project that the Delivery Team continues to have work. There are a variety of tools and techniques a Product Owner can use to prepare for writing detailed stories.

Personas

A Persona is a profile of a typical user of a product or feature. A Persona should be created for each distinct type of user in the system. A "distinct persona" is one that would require separate features, for example an 18 year old and an 80 year old would only be considered different Personas if the available features for each are different.

Name: Sally
Type: SysAdmin

Responsible For: Making sure systems stay up and useful
Skills: Bash scripting, fire fighting

Challenges: Constantly interrupted, juggling many tasks at once

Feature Impact: Needs mobile app as not at desk a lot; Integrate tool with the To-Do list to make navigation easier

Give each of your Personas a real name, this gives the team a handy short-cut when discussing features: "Sally creates a new account for Bob". Name Personas so that each starts with a different letter — this makes them phonetically distinct and also makes it easy to have multiple examples of the same Persona in a feature description. For example, imagine a Persona named "Bob" and the description "Bob chats with Barb". Both Personas in this description start with "B" indicating two different "Bob" Personas are interacting. This makes it easy to describe interactions of the same Persona without having to specify each individually. Avoid using names of people in your organization or your customers, it can cause confusion.

User Journeys

A User Journey outlines the steps that a user takes within a process in your software. For small products, the journey might encompass the life cycle of the user, for larger systems a journey might be only part of a bigger picture. Different Personas will have one or more journeys, each describing ways the Persona uses the software. One way of mapping out a journey is to use a series of sticky-notes. Each step in a process is put on a single sticky-note and a horizontal line of notes is created for each journey. Alternative paths, or choices in the system, can be denoted using additional sticky-notes in a column underneath the corresponding part of the journey line.

A "Journey Map" is a User Story Map (see below) outlining the current state of a system. Creating a Journey Map helps understand the features each Persona uses and gives an excellent starting point to discuss new changes to the software.

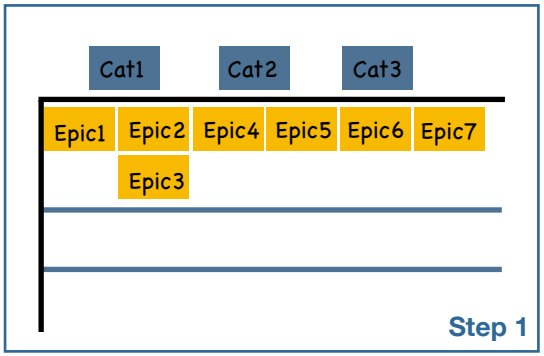
User Story Maps

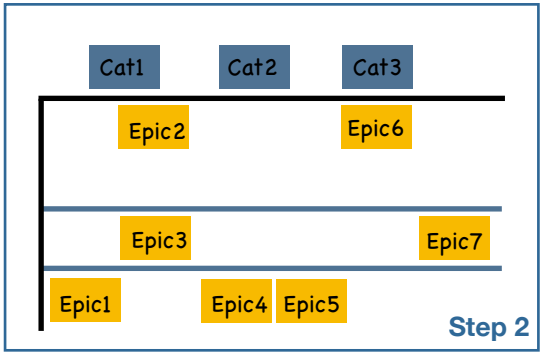
A User Story Map is a powerful tool for helping to build out a backlog or do release planning. A Map usually consists of three rows, each representing a fixed time period. The length of time periods depends on the planning horizon, a row might be a sprint, a month, or even a quarter. To plan a backlog, start with a User Journey. Step 1 shows a series of epics that have not yet been prioritized. Above the top black line are categories, these represent groups of features. Categories are normally consistent within a feature domain. For example, categories in a trading

system will be the same in any organization that has trading systems. The features being built in the trading system — in this map, the epics — will be specific to your implementation. Depending on the size of the system being planned, multiple maps may be required from the perspective of different feature sets or Personas.

Once a User Journey is complete, the next step is to prioritize. Move the epics up or down the rows of the Map to indicate higher or lower priority, shown in Step 2. This grouping of priority will inform the order of the Product Backlog. This 2-Dimensional view gives a better overall picture than the list format of a Product Backlog. The categories help indicate if a variety of feature areas are covered and the rows relate to forecast release time frames.

If a prioritized and estimated Product Backlog exists, the User Story Map can also be used for release planning. Start with an empty map and add items, in priority order, from the backlog. Add to the top row first and as each item is added examine the total size of all work allotted to that row. If a row is full, subsequent items go in the next row. This grouping gives a rough idea of what each of the first three releases would contain. If the set of features in the Minimum Viable Product is known, the Map reveals when it will be delivered and can drive conversation about changing priorities to minimize time to market.





Epic Budgeting

When nothing is known about a team's ability to deliver, all estimation exercises are guesswork. Once a team has been working together for some time, collected performance data can guide the process. Epic budgeting is the process of taking known data about teams and using it to help forecast future work more accurately. A common mechanism for sizing epics is T-Shirt sizing, where each epic is put into buckets of Small, Medium, or Large. Epics are broken down into stories by the team and stories are sized using Story Points. Using these two different sizing mechanisms is key to Epic Budgeting.

After a team has delivered a number of epics, analysis can be done on what the average size of an epic was in total story points. For example, if a team completed three Medium sized epics that turned into stories totalling 80, 90 and 100 points, the average Medium sized epic is 90 points. If a team's velocity is 45 points, the average time to deliver a Medium epic is two sprints. Understanding these numbers allows roadmaps to be adjusted. The new information can be used to adjust the scope, re-estimate the epics or change delivery forecasts. Adjusting scope is always the preferred option in Agile delivery.

Budgeting epics can also be used to help control gold-plating: the over delivery of features that were not wanted. Knowing the average size of an epic for a team gives the Product Owner a guide to control the stories inside of an epic. Stories should always be built in priority order so those at the end of an epic may not really be needed. Instead of gold-plating, move on to the next epic.