Feature flow Documentation

Release 0.1

Feature-flow contributors

Aug 21, 2017

Contents:

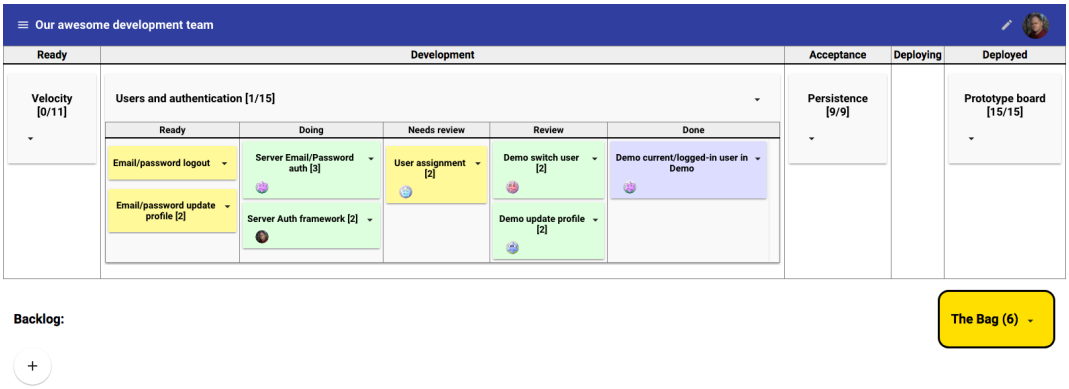
CHAPTER 1

Feature flow

Feature flow is an agile practice that seeks to provide value as quickly as possible by focusing teams on individual units of value, called "features". Ideally, a team works on one feature at a time[1].

A feature remains a team focus until it's providing value. It's not enough, for example, for software to be checked in with tests passing. It must be in front of customers or otherwise providing stakeholder value.

How it works



Feature flow uses a two-level board. At the top level, is a feature board, showing features in progress or in the product backlog. The feature board provides a high-level view so stakeholders can see what's in progress and prioritize the backlog.

When a feature enters development, a task board is used for the feature. This is the low-level view.

At any point in time, there's feature board and 0 or more task boards, however, there should usually be only one or two task boards.

[1] In practice, when a feature is nearing completion, there may not be enough work left to occupy the whole team, so the team may start another, however, the top priority of the team is getting the first task finished.

Core concepts

Board A board provides a place for a team to track their work.

Feature Features are units of value.

We use the term *feature* rather than value to emphasize visibility.

A feature will often encompass multiple stories, depending on the granularity of stories. A feature should be as small as possible and still provide value[2].

Task Features are broken into tasks, so that work may be subdivided and that progress may be tracked. Features *can* be small enough to have a single task. That is a good thing, because it means that value can be recognized sooner, but typically features require multiple tasks.

When a feature enters a development state, a task board for the feature is used, allowing the team to coordinate effort to drive the feature to completion.

How it's implemented

Feature flow can be implemented in a number of ways:

- You can implement feature flow with a feature board and a collection of task boards, either using a software tool or sticky-notes on physical boards.
- You can use a single board with big cards for features and sticky notes for tasks. When a feature enters a development state, you can move the stickies between development task states.
- Possibly using a tool that supports complex workflows.
- The *Valuenator* application.

  The Valuenator open source application is an attempt to automate the practice in a simple and opinionated way. There's a *demo version* you can try without installing or signing up for anything to get a feel for the mechanics of the practice.

However it's implemented, it's important that the implementation makes it easy to see everything relevant to a team at once. This is one reason why we think that a more specialized and opinionated tool has value.

How it fits in with other agile practices

Like any other agile practice, feature flow is a part of a larger agile process that teams should tailor to their needs and experience through a process of "inspect and adapt". Just as software should be built incrementally, so should you evolve your agile processes incrementally. Feature flow is one part.

Feature flow is an alternative to Scrum sprints. Rather than organizing work into fixed time increments, feature flow organizes around units of value. Features play a similar role to sprints, focusing a team on a shared goal and features are often similar in size to sprints.

Organizing around value rather than time has a number of advantages:

- It focuses the team on what's important to stakeholders.

[2] In a continuous-deployment environment, you might deploy subsets of features, with subsets not user-visible. This can help to avoid large software changes, to mitigate the risk of breakage. It can be argued that this provides value, but it's value that's not really visible to stake holders. Which isn't to say that feature flow and continuous deployment can't be used together, but they represent different kinds of flow.

This may, for example, include activities outside of traditional development, such as deployment or training, because the team is focused on achieving value, not just finishing promised work.

- It provides value as soon as possible, not just at sprint boundaries.
- Much less time is spent in sprint planning, because there aren't sprints.
- Team improvement can be considered at any time, rather than at sprint boundaries, because there's less emphasis on deadlines.

Feature flow isn't new. Feature flow can be seen as an instance of continuous flow, in that there's team focus on individual backlog items.

Feature flow is based on two-tiered Kanban boards as described in the book Kanban, by David Anderson (and elsewhere).

Feature flow can and should be used with other agile practices, as part of a larger process.

CHAPTER 2

Valuenator

Valuenator is a somewhat opinionated implementation of *feature flow* that seeks to automate what's core and proscribing as little else as practical.

Some important guiding principals:

The core ideas should be automated The hierarchical relationship between features and tasks is explicit. Task boards are hierarchically displayed within the feature board when features enter a development state.

A team's activity should be readily visible It should be possible to see important information on who and what on a computer monitor without scrolling.

The goal of this is to support stand-ups where people don't talk about what they're working on, because it's apparent on the board.

Instead, the stand-up is a chance to address issues, like blocked work or pile-ups in the process, which should also be apparent on the board.

Teams should be self organizing Tasks aren't assigned up front by default.

Rather as users pull tasks into working states, like doing or review, they automatically self assign.

Running Valuenator

The easiest way to try Valuenator is with the *demo version*. This lets you try out the application and get familiar with concepts very quickly without making any commitments or signing up for anything.

You can participate in the *on-line beta*.

There's a Docker image that makes deploying Valuenator easy.

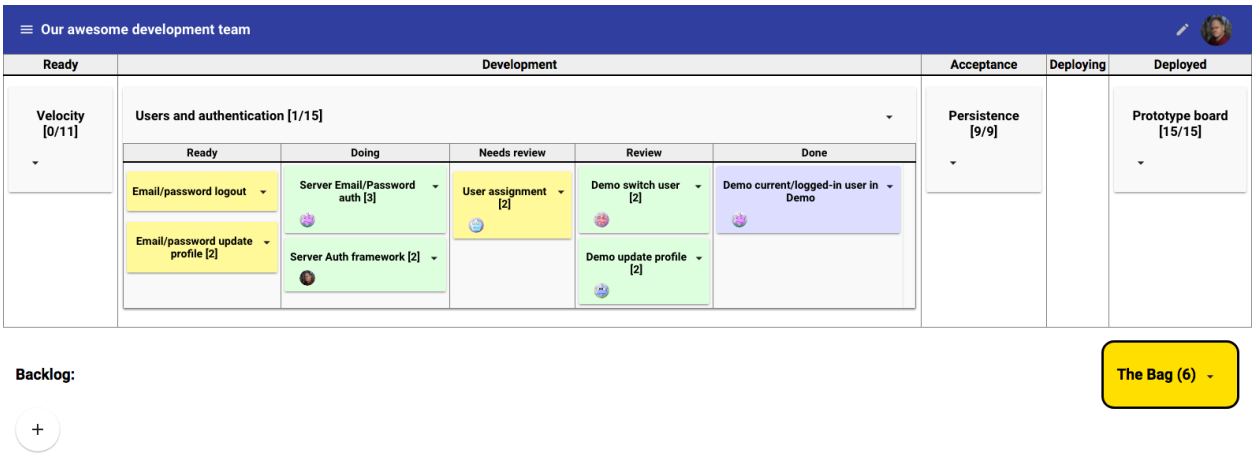
Valuenator is an open-source project, which you can modify and run yourself.

Using Valuenator

Using Valuenator amounts to creating and editing features and tasks and moving them through states by dragging cards on boards.

Features start in the backlog, move through several states. When done, features are dragged to "the bag", where they can be found later, if needed.

Let's walk through the user interface, as shown in the screen shot below:



Top bar

At the top of the window is a bar that shows a menu icon on the left. Clicking on this icon opens the navigation drawer, which lets you navigate between boards.

To the right of the menu coon is the title of the current board. On the right of the bar is the avatar of the current user. Clicking on the avatar displays a menu of user options, like logging out or updating profile information.

The avatar comes from Gravatar. Anyone can use Gravatar to associate an avatar image with their email. If a user's email hasn't been registered with Gravatar, a random silly image is used. The demo version of Valuenator has sample users with `example.com` email addresses so their avatars use silly images. If you click on the avatar and select "Profile" from the menu, you can change the email address to use an address registered with Gravatar, and get a real user image.

To the left of the avatar are various buttons. Move your mouse over a button to get a brief description of what it does.

Feature and task boards

Below the top bar is the feature board. Each column in the board represents a feature state[1]. Each feature is shown as a card on the board. You can drag a card with your mouse to move it to a new state.

When a feature is in the development state, its card contains a task board, showing the tasks needed to implement the feature. Tasks are shown as cards on the task board and can be dragged between columns to change their state.

[1] States will be editable eventually. This is a planned but so far unimplemented feature. If this is important to you, send an email to feedback@valuenator.com, or comment on the github issue.

A feature walk through

Let's see how this works by walking through the feature-flow process.

1. Create a feature.

   Click on the "+", add button, below the backlog title.

   A dialog is displayed where you can enter a feature title and description. Enter a title like: "Support multiple browsers". Enter a description, if you wish and click the "Add" button.

   A card for the feature is added to the backlog. By default, the title is shown and some basic metrics are shown. The metrics, shown as 2 numbers in parentheses, are the total number of completed points and the total number of points in the feature.

   A reveal button to the the right can be clicked to show more information, including the feature description, tasks, and buttons for editing the feature and for adding tasks.

2. Add tasks

   Click on the reveal button of the feature you just added. You'll see the description you entered, if any, and buttons for adding tasks (+) and for editing the feature (a pencil).

   Click on the add/+ button. An add-task dialog is displayed. In addition to a title and description, you can specify:

   Size The estimated size of the task in points. This is usually just a rough estimate.

   Assigned You can use this to assign a task to a member of the team, but it's usually best to let team members self assign by dragging tasks into working columns.

   Blocked This is used to indicate a reason preventing the task from making progress. This isn't usually set when adding a task. Don't use this to indicate dependence on other tasks, at least not initially. Blocked tasks get special attention in stand-ups so unblocking them should be actionable.

   Enter a title, like "Safari". Press enter. This immediately adds the task. The title is cleared so you can add another task. A message is displayed briefly at the bottom of the window indicating that the task was added. This allows a number of tasks to be added quickly.

   Now enter "Firefox [2]" and press enter. This second task is added. The "[2]" in the title indicates that we estimate the task will be roughly twice as hard as the smallest task. When a title ends in a number in square braces, the number is taken as the size. This allows for sizes to be supplied in quick-entry mode.

   Enter "Edge [2]". Also enter "Need a computer with the Edge browser" in the blocked field. Click "Add and add another".

   Enter "Write a test script" and press enter.

   Press the escape key to cancel the form and stop entering tasks. Alternatively, you could have clicked the "Add" button to add the previous task.

   A common workflow might be to have story times[2], after which someone does a work-break down to identify the tasks. They're likely to add a batch of initial tasks.

3. Now that tasks have been defined, we indicate that the feature is ready to be worked on by dragging it to the "Ready" column on the feature board. The "Ready" column is orderable, so if there are multiple features, we can indicate the priority by dragging to different positions in the column.

   Normally we wouldn't do this if there was already a feature in the ready column. In fact, we might not bother with taking the time to elaborate features with story times or do work break-downs until the ready column is empty. Requirements and priorities can change very quickly and work can end up being wasted if it's done too soon.

[2] It's common to iterate on stories and multiple meetings are often needed.

In this walk-through though, we'll take our example feature through the various states even though there are other features in them.

4. When we're ready to start working on the feature, we'll drag it to the development column on the feature board.

   When we do, we see all of the tasks are in the "Ready" task state. Imagine we have 2 developers. Drag the "Write a test script" and "Safari" tasks to the "Doing" column. The tasks are automatically assigned to you and your avatar is shown on the tasks. The task colors switch to green as well to visually emphasize that they're in a working state.

   Note that when you dragged to "Doing", the whole column was highlighted. Non-waiting states, like "Doing", "Review" and "Done" aren't orderable. Dragging tasks to those states simply adds them to the top of the list of tasks in the state.

   If you'd wanted to make the exercise more more realistic, you could have switched users by clicking the avatar in the action bar, selecting "Switch user" and then selecting a different user to act as. Of course, you could also just reassign one of the tasks. For example, if you click on the expand button for the Safari task and click the edit/pencil button, you can assign the task to someone else.

   In your stand-up meeting you'd note the blocked task because the task is shown with a pink color and shows the blocked reason. You find a suitable computer for testing, and thus unblock the task. Click on the expand button for the task and then the edit button. Delete the text in the blocked field and save. Now the task is shown in yellow, and is ready to be worked on.

   Drag tasks through the phases until all of the tasks are in the Done column. Notice that as tasks are dragged to the "Done" column, the count of completed tasks increases.

   Finally, drag the feature to the "Acceptance" column. It's shrinks back down to a single card.

5. Drag the feature to the "Deploying" column and finally to the "Deployed" column. After appreciating your accomplishment, drag to feature to "The Bag". The feature is now "in the bag".

You can change your mind and break rules

While you'll usually drag tasks across boards state by state in one direction, Valuenator doesn't enforce this. You can skip states. You can drag tasks from one feature to another. You can drag tasks to feature columns, turning them into features. You drag empty features to task columns making them tasks.

Features don't have to be done to be moved to the Bag. The Bag is a place to put features you don't want to think about any more. You may for example decide that a feature is too hard and not worth the effort and drag it to the bag. (Bag it.) If you change your mind later, you can pull it back out of the bag.

Empty features

Sometimes features are very small and don't need to be broken into multiple tasks. If you drag an empty feature into development, a task with an empty title will be created automatically for you. This is useful for tracking progress through development states and seeing assignments. Of course, you can edit this task if you wish.

Working with the Bag

The bag has a reveal button. If you click on it, the bag will expand to show the most recently bagged features. Over time, as you bag more and more value, you'll have more than can be shown at once. Arrow-buttons at the bottom of the bag let you scroll through features. You can also search for features based on feature and task title and description text.

Each feature is shown with title and metrics and has a reveal button. Clicking on the reveal button for a feature shows its description and tasks. Also shown, at the bottom, is a button to restore the feature from the bag. Clicking on this button restores the feature to the state it was in before it was bagged.

Users

User types

There are currently two types of users: regular and administrative users.

Administrative users can:

- create and rename boards
- adjust user types
- export board data
- In the future, adjust board states

User-management in the on-line version

The online version currently supports user login using email addresses and passwords[3]. When you set up Valuenator, you need to create a site and bootstrap administrative user.

Once a site and administrative user have been set up, other users can request access. The administrative user can then review and approve access requests and users are send links to set their passwords.

How do I used Valuenator with Jira or github or (substitute your issue tracker)?

We believe that issue trackers should be used to capture problems and opportunities. Tools like Valuenator should be used to create features that address issues. While it might be a good idea to be able to trace from features to issues, we don't think it's a good idea to reuse issues as features directly.

We'll likely integrate with tools like Jira and github in the future. Let is now how important this is to you and what sort of integration you'd like to see by creating or commenting on an issue in the Valuenator issue tracker.

Can I get my data out of Valuenator if I decide not to use it any more?

You'll be able to export all of your data in a simple JSON format.

[3] In the future, we plan to also provide login using single-signon services.

CHAPTER 3

Trying Valuenator

Valuenator demo

The Valuenator demo is a version of Valuenator that stores its data locally in your browser. Data are stored persistently, so you can reload the demo page or restart your browser without losing data.

The demo has all of the core features of the online version.

The demo version has some limitations:

- Because data are stored in your browser, you can't collaborate with other people. You also can't view data from multiple browsers.
- Searching in the demo is a little different than in the online version. The demo searches for features and tasks using simple substring searches, while the on-line version uses a language-aware full-text search engine.
- It requires a modern web browser. It is known to work with current versions of Chrome, Edge, Firefox, and Safari. Desktop browsers are required to drag tasks between states.

To run the demo, open the following URL:

> http://valuenator.com/demo#/board/sample

This will take you to a board with some sample features.

See the *Valuenator documentation* for information of using Valuenator.

Valuenator beta

You can try the online version of Valuenator for free during the beta period.

The goals of the beta are:

- Get feedback
- Find out what resources are needed to run Valuenator.

To request beta access, fill out the beta request form:

https://goo.gl/forms/nxECJrBPCYB6WC6x2

When we're ready, we'll get back to you with instructions for getting started.

As with the demo version, the on-line beta requires a modern browser, such as Chrome, Edge, Firefox, or Safari.

Docker image

There's a Valuenator docker image that you can use to easily deploy Valuenator yourself using Docker.

Valuenator is open-source

Valuenator is open source. You can check it out from github and run it yourself, although the easiest way to run Valuenator yourself is with the docker image.