Pasi Pekkanen, Reko Jokelainen
School of Science and Technology
Aalto University
Helsinki, Finland
Email: {firstname.lastname}@soberit.hut.fi

Introduction

Agilefant is an open source backlog management tool enhanced with hierarchical requirements handling capabilities. Agilefant provides three different planning levels for backlog management: product, project and iteration backlogs. Besides traditional flat-list backlogs, requirements can be expressed in tree form, in order to maintain traceability and transparency.

In addition to backlog and requirement management, Agilefant contains different composition views to the items in different backlogs, which can be useful for small to mid-sized development organizations. These composition views include time tracking features, personal work management and portfolio handling.

In the following chapters we start from an idea and show how it can be represented, refined, implemented, and monitored using the Agilefant.

Refining an Idea

Say an eCommerce online store application has been developed for some time now. Multiple versions have already been released and the development of another release is about to begin. One of the sales team members has had an idea of developing a feature, which displays what other people have bought in the web store. He has communicated his idea to one of the Product Owners (PO) in the development organization. By Product Owner we mean a person who acts as a link between the business people and development organization and is responsible for the product's requirements.

The PO immediately creates a story called "suggestions what others have bought" to Agilefant's product backlog. *The product backlog* represents all ideas, requests, features, user stories and other desires placed on the product, while *a story* is an item representing one of these desires.

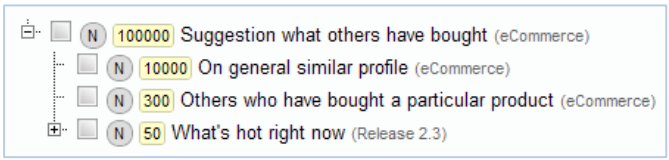

Figure 1 - A view to the story tree

With a technical team, the PO investigates the new story; these sessions are often called 5%-workshops or story times. During the session they decide that the idea is too vague to be implemented as such, and enter a story point estimate of e.g. 100 000 points. *Story points* are used to estimate the relative size of a story, meaning that a story with 4 story points is twice as big as story with 2 story points. In this case, 100 000 story points denotes that the story cannot be currently estimated, or is an epic as some might say.

During the same session the PO and technical team agree that the original idea contains at least 3 different features. They create three child stories to the original story in the product

backlog. The child stories are “suggestions on general from people with a similar profile”, “suggestions from others who have bought a particular product” and “what’s hot right now”. Technical team estimates these stories and the PO places them in relative priority order. This is done by using the Agilefant’s product backlog, which is actually a story tree. In the story tree, child and sibling stories can be created to a given story, and name, description, story point estimate, assignees, state, labels, and backlog of a story can be updated as well. In addition, the stories in same branch can be prioritized.

One of the stories, “what’s hot right now”, is estimated to be 50 story points in size. With the help of metrics from previous releases we can see that this seems to be in the scope of the upcoming release. Thus the PO assigns the story to the upcoming release project.

The release, or project, view also contains a story tree, which is a sub-set of the product backlog, consisting of the stories, which have been assigned to the given release. During other sessions, the PO and the technical team continue to refine the “what’s hot right now” story. From previous iterations in Agilefant, they can see that the team can complete around 15 story points per iteration. Thus, the story is split to 10 child stories, which are estimated to be less than 10 story points each. These 10 stories are small enough to be implemented in iterations, and don’t need to be split to finer pieces. In Agilefant these stories are called leaf stories. *Leaf stories* are stories with no child stories. They are the leaves of a tree, whereas their parent stories are the branches.

In addition to story trees, the leaf stories are shown in the project leaf story list, which is a flat list, or a traditional backlog as some might say. This list shows all leaf stories in a given project and the leaf stories, which reside in the iterations of the given project. The leaf story list allows prioritizing the stories, which will be worked on by the development team. Moreover, the leaf story list provides a convenient way to assign stories to different iterations.

From an Idea to Action

The Product Owner has selected three of previously mentioned leaf stories to an iteration and is currently doing sprint planning with one of the development teams. During the sprint planning the development team members create tasks to elaborate the implementation details for the stories. *Tasks* describe how a particular story will be implemented, whereas a story describes what will be implemented.

Once the team has finished breaking down the stories, they estimate the time each task will take to implement. The estimation provides a rough picture of how long the implementation will take, and the PO can further adjust the iteration backlog’s priorities based on that information. In Agilefant this initial estimate for a task is called *original* estimate.

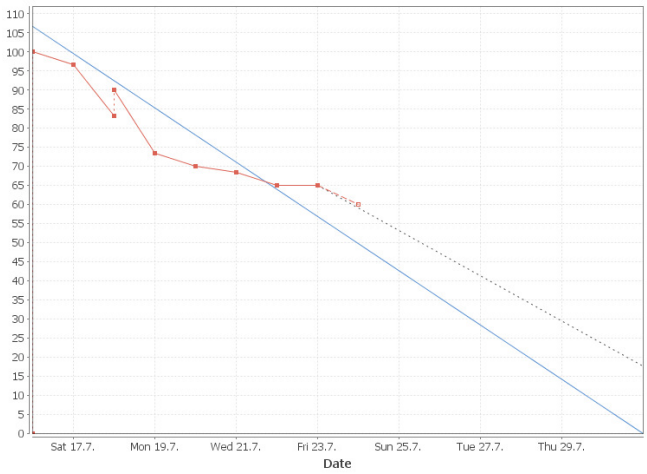

Figure 2 - The iteration burndown

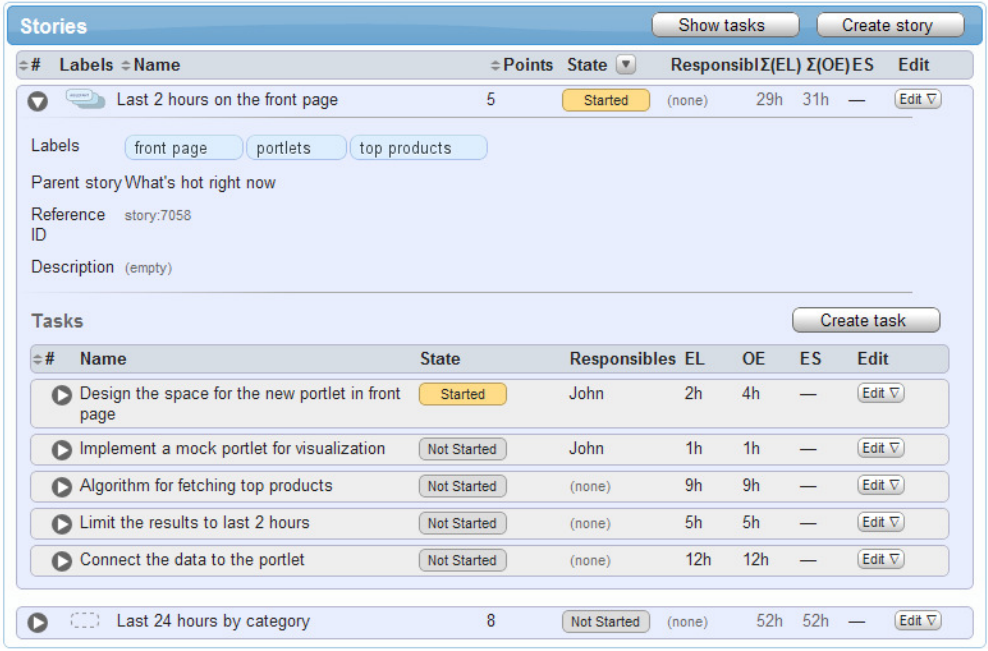

Figure 3 - The iteration backlog

Once the iteration begins, the development team members assign tasks and stories to themselves to indicate who's working on what. To further communicate the progress of the work, a state can be selected for tasks and stories. Available states are: not started, started, pending, blocked, ready, and done. While working on the tasks, the team members update the tasks' effort left estimates according to their best knowledge. *Effort left* marks the amount of time that the team members think they need to accomplish the task. This estimate may very well be higher than the original estimate as the developers become more informed of the situation once they have actually started working on the tasks.

An important element of the iteration view is the iteration burndown. It is a graphical representation of task progress in the iteration over time. It shows a linear reference based on the original estimates. The burndown provides a convenient tool for tracking iteration progress both for the development team and for external stakeholders.

Tracking the Progress

Once the iteration has ended and iteration demo has been held, the PO reviews the state of higher level stories. Even if all child stories of a story are marked done, the parent story might not be complete. Often stories are split only partially and refining the story is continued when needed. Thus, manual inspection of the stories is required.

Project progress can be tracked from project burnup chart. *Project burnup* is a time series displaying planned work versus completed work. Planned work is the story point sum of all leaf stories that reside in the project backlog. Similarly, the work done is the sum of all leaf stories marked as done. As stated earlier, stories assigned to iterations are also shown in the project backlog and thus, when a story is completed in an iteration, the progress is shown immediately in the project burnup.

In addition to project burnup, Agilefant provides the possibility to track story level progress. For example, the sales person, who originated the idea “suggestions what others have bought”, might wish to follow its progress. Agilefant calculates two different progress metrics for each story. First, the *simple metric* shows total leaf story points and total *done* leaf story points in that story. Second, the *advanced metric* takes into account the branches under the story in question.

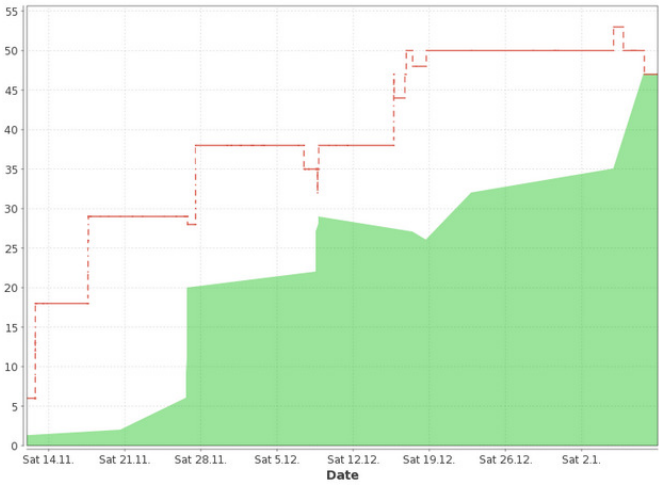

Figure 4 - The project burnup

Summary of the Concepts and Basic Views

As described earlier, Agilefant has a product oriented approach. Projects reside in products, and iterations in projects. Projects and iterations have a fixed time span and users can be assigned to them. Stories describe the work to be done, or on the other hand the features to be implemented. Their size can vary from simple needs to a multi-feature bulk. Stories that are selected to be implemented are moved to iteration level, where they can be elaborated as tasks, which depict the actual work that needs to be done (e.g. design the test). Tasks are estimated in hours, which are easy to grasp for developers. Stories are given ballpark estimates in story points, which describe the relative size of the stories.

Product backlog is actually a tree consisting of the product’s requirements. Projects have both, a story tree and a flat list of the project’s leaf stories. Iteration level contains only a list of leaf stories.

Using Agilefant in a Project Organization

The previous narrative was written from a product organization’s point of view. Although the concepts and backlogs’ names better suit the needs of a product organization, Agilefant is not restricted to those. Project organizations can benefit from Agilefant, too, by using it in a slightly different manner. Products can be used to represent a specific customer, whereas projects can be used as releases or milestones. The iterations may or may not suit project organizations, depending on the amount of projects and teams. Since there is no “team iteration” in Agilefant currently, a new one must be created for every project.

Other Functionalities in Agilefant

In addition to the aforementioned functionalities, Agilefant has also a couple of other useful features. For organizations that need time tracking, there is a *timesheets* feature allowing you to log hours to products, projects, iterations, stories and tasks, and generate reports on the effort spent. You can also export the reports to an Excel sheet.

The *Daily Work* page is an aggregate page, where everything that a single user is currently working on is gathered to one place. It shows the assigned stories and tasks from the currently

ongoing iterations. Users can easily keep their own work queue of tasks to keep in mind, what's currently important.

Portfolio view is also an aggregate page. *Project portfolio* is a special page gathering the ongoing and future projects to a list, where you can track them. You can create other portfolios with customizable widgets to suit your needs. Portfolios can be public or private.

Conclusion

Different stakeholders in a software development organization have various information desires. Some wish to view the high level picture, while some are only interested in individual iterations or stories. Agilefant satisfies these various information interests by offering different planning perspectives, requirement abstraction levels and composition views.

Iteration backlogs represent actual work done by the development teams. Projects backlogs are a composition of work assigned to different iterations, and work waiting for further refinery, or to be assigned to an iteration. Product backlog contains all needs, ideas, goals, and requirements that have been expressed by different stakeholders towards the product in question. Together, these three types of backlogs form nested planning horizons.

The hierarchical stories provide a traceable path through the different planning horizons. With the story tree a developer can connect his work to a larger scope and an external client can see what is currently happening in his high-level feature.

Different backlogs and possibility for hierarchical stories form Agilefant's core. This core is supported by several radiator views. The Daily Work view displays what everyone is currently doing and how occupied they are with those activities. The Timesheets view provides possibility to log and track time spent. The Portfolio view combines information from different products and thus offers an organization-wide perspective.