THE BASICS OF GOING SCRUM

COUNTERSOFT

Contents

Going Scrum

What we're going to cover

Scrum is an implementation of Agile, a methodology for software development created in 2001 that favours short, iterative cycles of development, requires multi-stakeholder participation and favours encapsulating requirement gathering, design and development in the same cycle. Scrum is Agile but Agile is not Scrum is perhaps the best way to describe the distinction between Scrum and Agile because in many cases they are synonymous. Agile requires that all phases of the software development cycle fall within the iteration cycle and if this happens in a Scrum then that Scrum is fully Agile. For one reason or another, some teams have the Requirements Gathering phase occurring outside the Scrum and in those instances purists would argue that the Scrum was not fully compliant with the principles of Agile.

Scrum defines work as being in one of two key states:

- In a Backlog – this is the pool of work that will be considered by the team for development in the next delivery iteration.
- In a Sprint – Sprint is the term used by Agile teams to define the current iteration. The Sprint contains the work that the team has agreed will be done in the given timeframe.

After this point the status of the work depends very much on the decisions and structure of the team. For example, the Sprint might include a testing phase that does not need to be marked out as a special status of it may not. The team may have to go through formal sign-off and review processes and there may be a need to track statuses for a DevOps team, such as Deployed to xyz. For the purposes of most teams one can assume a level of testing and therefore a Status Test and a point of closure, therefore a Status Closed.

We are going to cover the features of Gemini that a team employing the Scrum methodology could use to go from Requirements Gathering to Code Review and Sign off, from being In Backlog all the way to Closed.

Capturing Requirements

Epics and Digital Assets

Requirements come in many physical forms. They are often a mix of emails, docs, notes, spreadsheets, wireframes etc. For Requirements in Scrum, read Stories, a Story being a distinct, described requirement that can be completed in a Sprint cycle. A Sprint can consist of several Stories but no single Story can take longer to complete than a single cycle. We shall examine Stories in more detail but for now let us focus on the fact that there are many occasions where the requirement is either too vague or too big to fit into a single Story. In this case, Scrum uses the term Epic, and there are various rules about how many man hours of effort a Story can contain before it becomes an Epic. We might therefore say that Epics come in many physical forms and the process of extracting from them the Stories for Scrum is the work of the Agile team.

A team working with Scrum will often need to store the documents etc. that make up the Epic (or Epics!). Something has to feed the endless backlog and it is very rarely the team going out to the business asking what they would like to get done next.

Gemini has a home for all the elements of an Epic both in the ability to define a data structure for an Epic in an Agile Project and in the DocStore app. This is an out-of-the-box, Open Source app that can be effortlessly added to your Project Templates. DocStore understands Gemini User Groups and permissions and provides a secure home for Project documentation and other digital assets.

Epics, Stories and Tasks as Types

All Projects in Gemini are built from Project Templates that define the Types of things the Project will contain. The default Agile Template in Gemini defines a Type for Stories and a Type for Task. It is assumed that Epics will more typically reside as documents in DocStore but it is a simple matter to create an Epic Type, which will consist of a few fields and provides a way for those who want to utilize all Gemini's functionality on the more complex (and larger) requirement Types.

As stated, Stories and Tasks are default Types on Gemini's customizable Agile Template, this means that they have standard fields and workflow for Scrum pre-defined. Scrum teams can modify the Agile Template or create their own Template to have different fields, including Custom fields, and different Workflow from the default. For example, the default Workflow is In Backlog…In Sprint…etc. There is no difficulty in adding an extra Status to the Project Template of Approved and then modifying the Workflow to be In Backlog…Approved…In Sprint etc.

Managing Hierarchical Stories and Tasks

On the default Agile Template, Stories and Tasks both have Dependencies enabled, this means that these items can be hierarchical. Gemini will allow Tasks to be a children of a Story and Tasks to be children of other Tasks, with no logical limit to the depth of nesting. Indeed it is the enabling of the Dependency property that allows the Quick Entry app to work when creating hierarchical items. What is more surprising is that Gemini's dependency functions will allow a Story to be a child of a Task. This is valid; all that is being implied in the Dependency tree is that the Task cannot be closed unless the Story (and all its child Tasks/Stories) are first closed.

The hierarchical nature of items in Gemini is further refined by the ability to have cross-project dependencies. This means that a Story in one Project could have a dependency on a Task in another Project that has a dependency on a Campaign (just another Type) in a Marketing Project and so on. Gemini's flexibility lets you implement the truth of real-world complexity. Take the following scenario:

A Marketing Campaign requires the Software Development team to make changes on the Website but the team cannot do their job unless the PR agency deliver the assets and that is the responsibility of the Marketing Team but this cannot happen unless Finance issue a Purchase Order etc. In Gemini, this type of complex inter-departmental dependency is no problem at all.

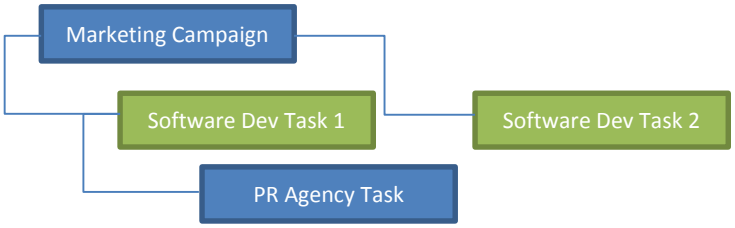

Fig 1.0 the kind of real world scenario Gemini's hierarchical Dependencies can handle

It does not matter what the nature of the Types are in the Dependency hierarchy but while complex scenarios are possible, the most likely is that Tasks will belong to Stories, both will belong to an Epic but the Epic will be represented in Project documentation and not in a Gemini Type.

Scrum Planning

Now that we've covered the structure of Scrum elements in Gemini, and some of the hierarchical capability that links them, we can drill into more of the functionality, starting with work estimation and progress updates.

Estimates

Gemini supports two metrics for estimating: man hours and points. You can use either or both simply by enabling these fields on your Agile Project Template. The field in which hourly estimates are captured is simply called Estimate, as this is the default. To capture Estimates or Points you simply click on either field and enter the appropriate value. Estimates are held in hours and minutes as shown in the image below.

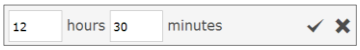

Fig 2.0 Estimate entry

Estimates in hours or by Points are shown on the View Items page, can appear on the Grid and appear on the cards that represent each Story/Task on the Planning Board when they are at Zoom Level 3 or 4.

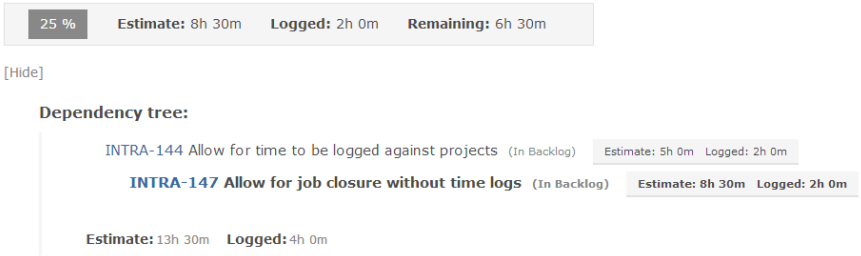

Fig 2.1 Estimate and Points in Items View, the page that shows all of the detail of a Gemini item.

***Note**: The image above shows not just a hierarchical relationship but also that Gemini hold time at an individual Story/Task level and also rolls up Estimated and Logged time to the parent level.*

Story	Type	Priority	Severity	Title	Estimate	Points
INTRA-50384 ▼				Allow recurring tasks to inherit dependencies	4h 30m	200

Fig 2.2 Estimate and Points as shown on the Items Grid.

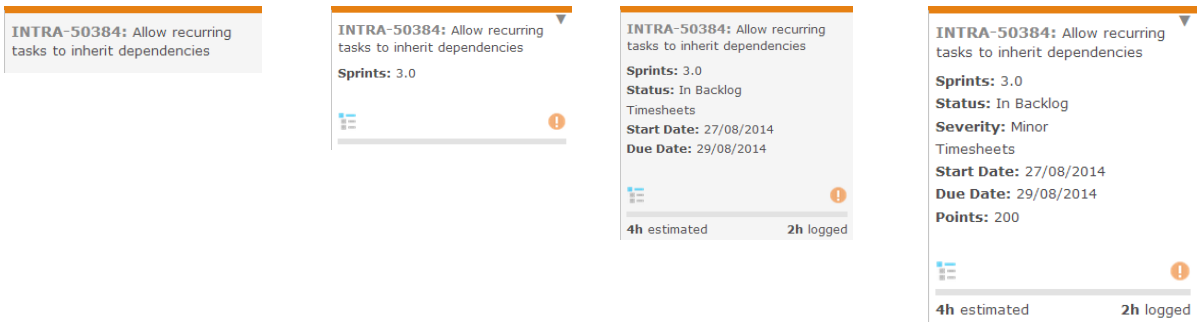

Fig 2.3 Estimate and Points on the Scrum Planning Board. Estimates appear at zoom level 3 and both appear at zoom level

Defining a Sprint cycle

Sprints are defined on a per-project basis and are therefore reached by navigating to the project in question from the Dashboard menu, clicking its link, and selecting the Settings menu. Here you will find Areas, Defaults and Sprints.

***Note**: Areas and Sprints are so called because the Agile Template overrides Gemini base terminology for these, which are Components and Versions. If you have not used the Agile Template for your project do not be confused, they are the same thing by different names. For more information on overriding and using your own taxonomy see the documentation here.*

Sprints can be hierarchical, although the hierarchy is for organization and visual representation, there is no workflow validation such as occurs with items, where the parent cannot be closed until all the children are closed. Each Sprint has Start and Release dates and when the work in a Sprint is over the Sprint can be marked as Released. The Start and Release dates are used by Gemini's Burn-down and Burn-up charts to track progress over time.

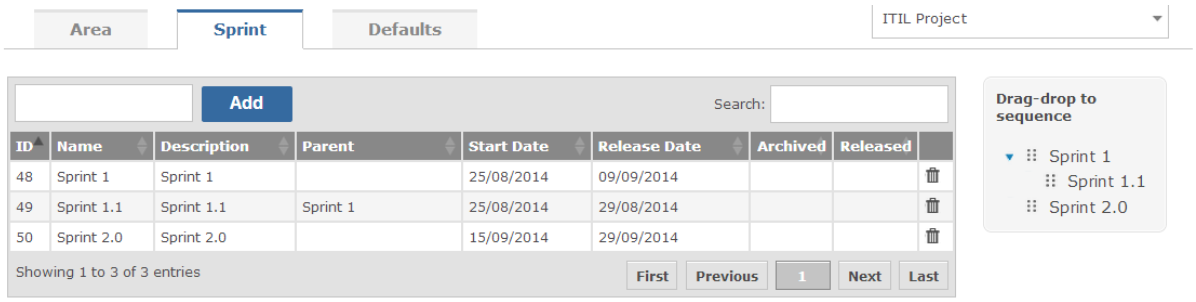

Fig 2.3 The Sprint tab in the Settings menu. On the right you can see that Sprints can be hierarchical

Although you create individual sprints in Projects, Gemini is able to perform all Sprint Planning processes across project. This is true of all Gemini apps because they work on the data that the Workspace is capable of amalgamating and Workspaces can pull together data from all the projects that the Workspace creator has access to. Moreover the Workspace can be shared with other participants and Gemini will apply their individual security permissions to determine what they can see so nit all users of the same Workspace will necessarily see the same information.

Roadmap and Changelog apps

Gemini has 2 apps that many Scrum teams deploy, Roadmap and Changelog. Roadmap shows all the work in Sprints that have yet to be released. Marking the Sprint as Released moves the work items out of the scope of Roadmap and into the scope of Changelog. We can therefore say that the Roadmap contains current and future work while the Changelog relates to the past.

Tracking Daily Progress

Scrum requires daily meetings with the project team in which each member gives feedback on the state of the work they are assigned to. This work spent can be tracked through Time Logs but time does not always equate to the key piece of information that the team should try to define – how much time is left to complete each task in the Sprint. Gemini provides a field for this, which is %Complete, and progress is always represented as a percentage. In figure 2.1 above, the team have decided that the task is 25% complete. The Progress percentage is always entered manually and represented visually on the Items Grid as shown below:

Fig 3.0 Progress as shown on the Items Grid

Scrum masters can attend Daily Standups with Gemini's Items Grid, filtered by the current Sprint and as each Story/Task is discussed update the %Complete field, so al team members can assess at a glance where they are and what is left to focus on.

***Note**: Gemini tracks Time Remaining, the difference between Estimated and Logged time as an absolute calculated value.*

Scrum Team Metrics

If you have not previously used Gemini Workspaces and have not read the document *Getting Productive With Workspaces <hyperlink to this>*, watched a video where Workspaces are discussed or read the documentation on Workspaces then it is strongly recommended that you take the opportunity now to do so. Gemini is a Workspace-driven product, that is its unique difference from most if not all products that offer similar functionality and it is inconceivable that your Scrums and all other work packages will occur outside of Workspaces.

One of the features of Workspaces is that they have an Instant Metrics tab that allow you to define such visually represented data as Open vs Closed, Burn-down and Burn-up charts and various other summary data. In addition, Gemini has the Progress menu, which contains larger versions of the Burn-down, Burn-up and Velocity charts with more functionality. For those not familiar with what these charts represent here is a brief description:

Burn-Down: Among Agile teams this is the most commonly used visual measure of progress. It shows work remaining against time. Gemini will show you a Burn-down by number of work items, hours remaining or Points.

Burn-Up: Shows work done against time. Where a Burn-down chart shows how much work is left to do and consequently goes down from left to right, a Burn up shows how much is done and consequently goes up from left to right. Gemini will show you a Burn-up by number of work items, hours remaining or Points.

Velocity: Shows the average value in terms of work delivered by the team in recent Sprints. It is a good measure of whether the estimates for what the current Sprint can/will achieve are realistic.

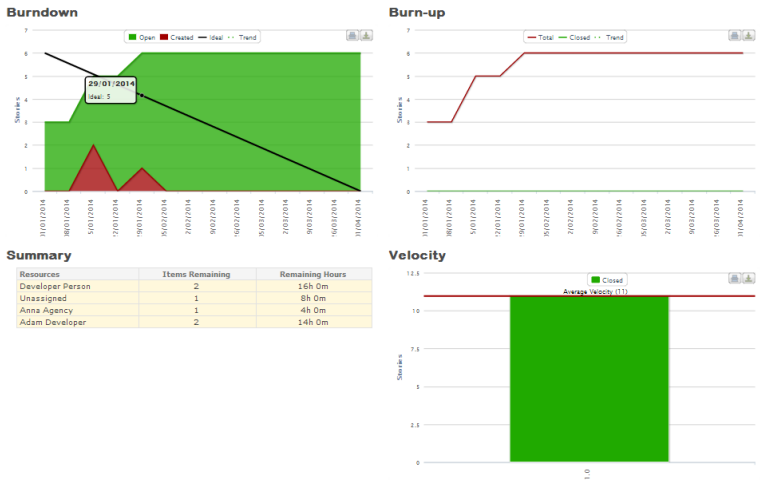

Fig 4.0 Snapshot of the Gemini Progress page, showing Burn-down, Burn-up and Velocity charts

Reporting

Reporting is a bone of contention that many Agile teams are forced to chew on. It is a necessary evil because people outside of the team need to know what is going on. Luckily Gemini takes the pain from reporting and makes it so simple a team can regularly report progress without even knowing it is going on. It is all down to the magic of Workspaces.

Workspaces filter data, from any and all projects, and Gemini's reporting mechanism passes whatever data has been filtered to Excel, to be pivoted, charted, listed or any combination thereof. Since Gemini supports the creation of ad hoc Excel reports and Gemini's workspaces let you define schedules and distribution lists for report production, the combination of the two results in effortless reporting on the status and progress of Sprints, Tasks and people in the process.

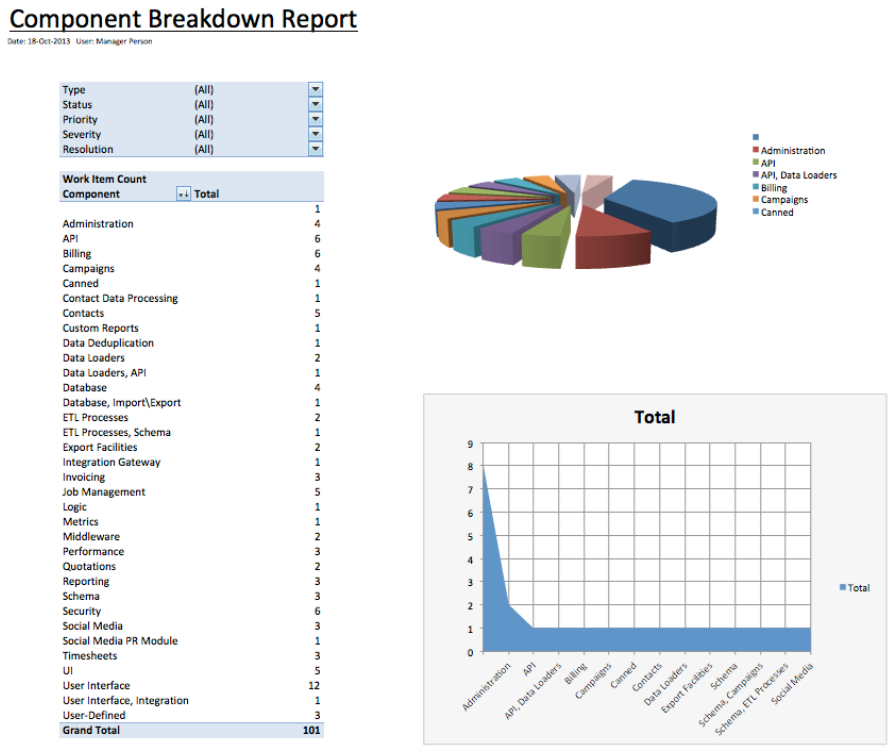

Fig 5.0 A typical Gemini report

Data for the Retrospective

The Scrum has been planned, tracked, executed and reported on. At the end the Scrum Master sets the Sprint to Released and if you have added the Changelog app to your Scrum Workspace(s) then all the data that you need for the Retrospective is there, in one place.

Gemini's standard views will always exclude closed item by default but that is just a display option, simply check the Include Closed Items option and all the history of your completed Sprints and Tasks is available to you. You can even create Workspaces that specifically filter closed items, since the display options are a property stored with every Workspace.

It remains only to mention one last Scrum-related app, but let us start by explaining why you might want to deploy it. *Burn-down and Burn-up charts are driven in part by the Estimate. Sometimes estimates are wrong and because of the impact on these tracking charts they need to be changed. This however can create a dilemma if estimates serve a dual purpose, for example if they have been used in pricing a piece of work.* To handle this situation there is a Gemini Event app called Original Estimate. If deployed the Estimate is stored in an Original Estimate field as well as the Estimate field so that this original value is never lost and can always be viewed and reported on.

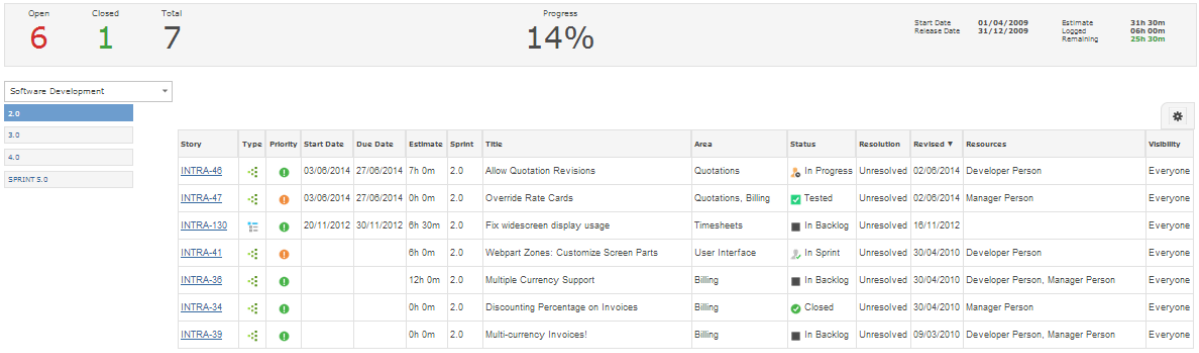

Fig 6.0 Snapshot of Gemini's Roadmap and Changelog app layouts