Jira definition of done template

In Agile Methodologies, Scrum Lists, Definition of Fact (DoD) and Acceptance Criteria are very important concepts. They link what the product owner wants to what the development team will deliver. In fact, it can be considered a contract between two parties. Putting them on paper is simple. The difficulty often lies in the Development Team actually respecting contracts; even when the team is well-intentioned. Teams with a dedicated Scrum Masters may have no problem with respect for the DoD, as the Scrum Master will push them to ensure that the Acceptance Criteria for each User Story are met and that they respect the DoD. Unfortunately, the reality is that many teams do not have a dedicated Scrum Master and that master Scrum's role is just another hat that a member has to wear. So what can a team do to ensure that the DoD and Acceptance Criteria are respected? From our experience, the most effective way is to try to embed them in your natural workflow. Since the development team is using JIRA and probably JIRA Agile, embed the DoD Criteria and Acceptance list directly in JIRA! This article will present good tips and practices for managing DoD and Acceptance Criteria and how to apply them within JIRA. Creating the DoD The DoD typically contains elements applicable to different stages of development. In our DoD, we have elements applicable to Technical Tasks (subtasks), User Stories (Problems), Sprints and even Versions. The Checklist snap-in can help you with the Technical Tasks and User Stories items. First, install Checklist and create a new custom field of type Checklist for Technical Tasks-Related State Department: On the Custom Fields administration page, click the + Add Custom Field button on the Step 1 of 2 screen: Select the Custom Field Type Checklist. Click the Next >> button. On the Step 2 of 2: Enter Fact Definition screen for the Field Name. Enter Fact definition for technical tasks for field description. For applicable problem types, select Technical Task or/and any type of problem that is typically used as subtasks. For the applicable context, leave it as is or select the projects for which you want the DoD to apply. Click the Finish button. You should now be presented with a page asking you to associate the new custom field with the JIRA screens. Associate the field with the Default Screen so that the DoD appears in the create and edit views. You can also associate the field with the Resolution Issues Screen so that the you can mark the DoD elements as facts when they resolve the issue. This will also be useful if you decide to enforce the DoD later. When you are finished, click the Update button. Now, create another custom checklist field for the DoD related to User Stories: Follow the same steps as that for the DoD of Technical Tasks, but this time, in Step 2: Enter Fact Definition for User Stories for the Field Description. Select History for the problem type. If your computer also uses other types of issues that are treated in the same way as User Stories, you can also select them. In fact, you can select any type of problem that shares the same DoD as the user's story. Setting up dod elements Now that our DoDs are created, they must be configured if we want them to be of any use. What to remember with a DoD is that it should be applicable to any problem, but surely we don't want to manually recreate the DoD every time a new problem is created. Because Checklist is a custom multi-selection field, you can use Options to configure your items. On the Custom Field page, find the DoD applicable to technical tasks, and from its actions menu (Cog icon), select Configure. This will take you to the custom field settings page. From there, click the Edit Options link. On the Options page, specify all DoD items that are applicable to Technical Tasks. Typical elements are: Code is compiled and introduced into CM code builds without warnings Tested Code Unit - All Green Reviewed by peer a DoD element is entered by adding an Option Value field. Options can be sorted alphabetically or in any order you consider important. Once the options are entered in the custom field, they become available for any new or previously created issues. This means that you can open and edit any Technical Task that is not already closed and you will see the Technical Task DoD appear. Even better, if you add other items to the DoD later and re-edit the same technical task, they will also be displayed. This is a powerful feature for managing a DoD over time. We'll cover this in the next section. Now, you can repeat the same steps to create the dod elements of the user story. Typical elements of a user story can be: All acceptance criteria meet the updated Build documentation published on the demo server By having a DoD for different problem types, we can customize the criteria based on the workflow. When the user story is created, it will have its own particular DoD elements. When the story will be divided into Technical Tasks during Sprint Planning, each technical task will have doD elements related to software development practice. DoD A DoD administration is a live document that should be reviewed regularly. As rooted in daily activities, you may want to replace some DoD elements with other less-rooted practices. For example, when your team naturally creates unit tests for all of its classes, you might want to move to Test-Driven Development (TDD). Therefore, you want to replace the DoD Tested Code Unit - All Green element with Code developed with TDD. Although you can delete/rename items and replace them with new ones, a is to disable the elements. When editing DoD elements through the Option page, each option entered has a Disable hyperlink. Disabling an option will keep the option on the system, but will prevent it from appearing in the problem. Therefore, when you create a new problem, disabled DoD items will not appear in the problem. By disabling items instead of deleting them, you keep a record of your DoD evolving over time. Here's what practices are now natural for the team and which ones they're still fighting in. Since the DoD is a contract between the Product Owner and the Equipment, it is natural to want to fit as many items into the DoD as possible in order to ensure the quality of the product. As good as it sounds in theory, in practice, having too many elements of the DoD can be counterproductive to you. When some teams face too many DoD elements, whether they work on a subset or on them or worse, they do a very bad job trying to do them all and eventually stop looking at the DoD all together. Therefore, the safest approach is to be reasonable with the number of items listed in the DoD. Choose the ones that are really important and that the team should focus on. When they're good at it, replace the items with new ones. As the adage says: Rome was not built in a day. But what if you want to challenge your team a little bit? Simple, make some DoD elements mandatory and others optional. You cannot fail in the sprint if one or more mandatory doD elements have been omitted, but it leaves it a success when only the optional items were omitted. Make it clear that optional items will one day be mandatory! This will challenge the team to take on more while still having the impression of being in control. Checklist gives you the ability to configure optional items. On the custom field settings page, select the Edit discretionary options hyperlink. You will be presented with the list of active options entered for the field. Simply select the DoD Elements you want to make optional. When you make DoD elements optional, it's good for developers to know which elements are required and which are optional. Return to the custom field settings page and select the Edit Parameters link. When you are on the parameters page, select the Emphasize required items option and make sure it is checked. Mandatory and optional DoD elements must now be displayed differently. Enforcing the DoD As mentioned in the introduction, the best way for a team to follow the DoD is to embed it in their workflow. Checklist comes included a workflow validator that can be used to apply the DoD. This validator will prevent the technical task or user history from being resolved or closed until all items in the Department of Labor have been completed. Therefore, when a user tries to resolve the issue they are working on, the validator will check whether all items in the DoD have been checked. If that's not the case, case, will avoid the transition of the problem workflow. The developer will then be reminded that some of the DoD elements have yet to be implemented. To add the Checklist validator to the workflow, edit the project workflow and: Select the transition to which you want to attach the Checklist validator. This is usually the Resolve Problem transition or the Close Problem transition or both. On the transition page, click the Validators hyperlink, and then click the Add hyperlink. On the Add Validator page, select the Checklist Validator and click the Add button. Select the custom field from the checklist, in our case, the Fact Definition field. Since we have two custom fields with the same name, it can be difficult to determine which one you are selecting. They are typically displayed in the same order as the one on the Custom Fields page. It might also be a good idea to have subtly different names. Now, decide the type of validation you want. You can require that all items be marked or simply required. If you have optional items in your DoD, you must choose to validate only the required one. Be sure to publish your workflow changes if you want them to take effect. Et voila, has now applied its DoD and made sure that the development team is aware of it. Be careful, if you simply attach a validator to the Resolve Problem transition, you can uncheck DoD Items after the issue has been resolved. Therefore, a best practice is to put the validator in the Resolve and Close transitions or at least on the close. Acceptance Criteria List While a DoD is global for all issues of a given type, the Acceptance Criteria list is specific to each user story. The checklist is perfect for acceptance criteria lists because it allows you to add items directly at the problem level. Follow the procedure described above to create a DoD, but this time, create a custom field named Acceptance Criteria and assign it to the History problem type. Because it doesn't really make sense to have global acceptance criteria, there's no need to create options for the field. However, if necessary, Checklist supports both modes simultaneously, that is, global and local items. Now, when you create a new user story, the product owner can add acceptance criteria directly from the problem creation page to make them immediately available to the developer. The developers then only have to check them when they meet them. Like the DoD, you can a workflow validator to the Acceptance Criteria list to apply when the story is resolved or closed. Managing who can add acceptance criteria It might be a good practice to limit who can add/change or remove acceptance criteria. In theory, the Product Owner is responsible for specifying the criteria so that it only makes sense that be the only one to manage them. Go to the custom field settings page for the Acceptance criteria field and click the Edit Parameters link. On the parameters page, select the role that must have the rights to manage the list. By default, the Administrators, Developers, and Users role appears in the Edit Roles section. While we can limit access to administrators, it makes much more sense to create a new role called Product Owners. After the new role is created, it must be available in the Edit Roles section. When you select the Product Owners role, only users who have that role can edit the Acceptance Criteria. The project administrator adds users to the Product Owners role. In conclusion, managing the Definition of Fact (DoD) and accepting Citeria lists directly in JIRA is not too complicated and can lead to teams that eventually act on them. Them.