Integration guidance

1. Contents

2. Introduction

This Eylean Board integration guidance provides a user a few general cases how to customize existing integration configuration, also, some tips are included for TFS and TeamPulse integrations. Finally fully detailed specification is included with an example configuration for a very basic case.

In case this is not helpful enough, please don't hesitate to contact our helpful support at support@eylean.com and we will guide you to configure a template for your particular needs!

3. Change log

These are the changes introduced to configuration of integrated boards. Deprecated parts of this guidance are in italic and colored gray.

1. Version 1.2

- Removed **StructType** configuration value for rows. With this, the ability to synchronize Areas as rows is disabled and if **RowType** value is **Structure**, then rows will be synchronized with **Iterations**. Areas are now synchronized with **Categories**, the new feature added in 1.2
- Added new feature to only add users to workspace members, who have assigned tasks. To do this, set flag **AddOnlyAssignedUsers** to **true** for config.

4. How To

For this part of the guidance, as a reference, an example template is going to be used which is at the end of this document, in the 7. Configuration xml specifications section.

1. … configure an additional item transition state.

In this case, a Story item has a new **State** added, which is called "Removed".

To configure this custom state these steps needs to be followed:

a. In the board structure part of the template you need to add a new lane, to accommodate the items which are in that state:

```
<Section Name="Output" AllowSubtasks="false" Limit="(null)">
   <Lane Columns="4" Limit="(null)" Queue="0" Name="Resolved" />
   <Lane Columns="2" Limit="(null)" Queue="0" Name="Removed" />
</Section>
```

b. Then in the particular story configuration add a state to lane mapping:

```
<StateToLaneMappings>
   <StateToLaneMapping State="To Do" LaneName="To Do" />
   <StateToLaneMapping State="Active" LaneName="Active" />
   <StateToLaneMapping State="Resolved" LaneName="Resolved" />
   <StateToLaneMapping State="Removed" LaneName="Removed" />
</StateToLaneMappings>
```

c. Define the transitions to and from the state (Note: this is optional if the workflow for the item is not configured. If it is not the case, this must be defined to access the state from the board):

```
<Workflow>
   <WorkflowItem StateFrom="" StateTo="To Do" />
   <WorkflowItem StateFrom="To Do" StateTo="Active" />
   <WorkflowItem StateFrom="Active" StateTo="To Do" />
   <WorkflowItem StateFrom="Active" StateTo="Resolved" />
   <WorkflowItem StateFrom="Resolved" StateTo="Active" />
   <WorkflowItem StateFrom="Active" StateTo="Removed" />
   <WorkflowItem StateFrom="Resolved" StateTo="Removed" />
   <WorkflowItem StateFrom="Removed" StateTo="To Do" />
</Workflow>
```

*Note: optionally you can set a state to enable complete item deletion, to do this, you set **StateTo** to a blank value.*

2. … configure and additional item type (TFS only)

Firstly, requirements for the item must be set by answering these questions:

a. Do states of the item introduce more lanes or it can map to an existing board structure?
b. Does it have subtasks and how is the relationship defined? (Relationship between **Story** and its **Tasks** are, in most cases, defined via association link object)
c. What fields does the item have?
d. Are there any fields that are required to be filled, on creation of the item?

This example will explain how to add a new **Story** type of item named “Bug” to the configuration with its subtask item. This story item has states which already map to the board (in case it doesn’t, follow previous example on how to add and map states) and it has a field named “Reproduce Steps”, which is a required value and is a string containing description of how to reproduce a particular bug.
The configuration for this example should look like this:

```xml
<Story Key="story2" ItemType="Bug" Id="ID" Name="Title" State="State"
   AssignedTo="Assigned To" StoryPoints="Original Estimate" Description="Reproduce Steps"
   ParentAssociationType="Property" ParentAssociationProperty="IterationID" Color="2">
   <StateToLaneMappings>
      <StateToLaneMapping State="To Do" LaneName="To Do" />
      <StateToLaneMapping State="Active" LaneName="Active" />
      <StateToLaneMapping State="Resolved" LaneName="Resolved" />
   </StateToLaneMappings>
   <Workflow>
      <WorkflowItem StateFrom="" StateTo="To Do" />
      <WorkflowItem StateFrom="To Do" StateTo="Active" />
      <WorkflowItem StateFrom="Active" StateTo="To Do" />
      <WorkflowItem StateFrom="Active" StateTo="Resolved" />
      <WorkflowItem StateFrom="Resolved" StateTo="Active" />
   </Workflow>
   <PresetValues>
      <PresetValue FieldName="Reproduce Steps" Value="Steps to reproduce the bug:" />
   </PresetValues>
   <Tasks>
      <Task Key="task1" ItemType="Task" Id="ID" Name="Title" State="State"
         AssignedTo="Assigned To" Remaining="Remaining Work" Completed="Completed Work"
         Description="Description" ParentAssociationType="AssociationLink" Color="4">
         <StateToLaneMappings>
            <StateToLaneMapping State="To Do" LaneName="To Do" />
            <StateToLaneMapping State="Active" LaneName="Active" />
            <StateToLaneMapping State="Resolved" LaneName="Resolved" />
         </StateToLaneMappings>
         <Workflow>
            <WorkflowItem StateFrom="" StateTo="To Do" />
            <WorkflowItem StateFrom="To Do" StateTo="Active" />
            <WorkflowItem StateFrom="Active" StateTo="To Do" />
            <WorkflowItem StateFrom="Active" StateTo="Resolved" />
            <WorkflowItem StateFrom="Resolved" StateTo="To Do" />
         </Workflow>
      </Task>
   </Tasks>
</Story>
```

3. Configure additional fields to synchronize

You can add additional field integrations to work item configuration. Each additional field mapping should contain Eylean field name and a TFS field name to map to. **Type** currently is not available and will be enabled in later Eylean versions. Example of such configuration:

```
<Story Key="story1" ItemType="User Story" Id="ID" Name="Title" State="State" AssignedTo="Assigned To" StoryPoints="Story
ParentAssociationType="Property" ParentAssociationProperty="Iteration Path" Color="4">
       <AdditionalFieldMappings>
                <AdditionalFieldMapping EyleanField="How to demo" TFSField="How to demo" Type="string"/>
                <AdditionalFieldMapping EyleanField="Acceptance criteria" TFSField="Acceptance criteria" Type="string"/>
       </AdditionalFieldMappings>
</Story>
```

5. Important notes for TeamPulse integration configuration

TeamPulse has a strongly typed service and doesn't have as much flexibility data wise as TFS, therefore configuration is much simpler. To set configuration for any TeamPulse project you need to take into account these notes:

1. **Row**, **Story** and **Task** nodes need to have unique **Key** attributes through the whole integration.
2. *Only **StructType** attribute for **Row** is required (deprecated since 1.2)*. You can still set Color attribute for **Story** and **Task**.
3. **Row** must have an array of **Story** nodes, and each **Story** node must have an array of **Task** nodes as usual.
4. **Story** and **Task** must have an array of **StateToLaneMapping** nodes set, so that integration would know which states goes to which lanes.

6. Important notes for TFS integration configuration

To configure TFS integration properly, follow the configuration specification. Since TFS is a very flexible system, precise configuration and validation is required, so if you are not sure if the TFS configuration works and you are building a board template, try running validation for configuration first and then copying the configuration to a template. It is under *Configuration* -> *Workspace* -> Select an integrated board and click *Edit integration*. Make sure that you are validating against a real project.

7. Configuration specifications

Example configuration xml file:

```xml
<Board ShowGoals="true">
   <Section Name="Input" AllowSubtasks="false" Limit="(null)">
      <Lane Columns="2" Limit="(null)" Queue="0" Name="To Do" />
   </Section>
   <Section Name="Work in progress" AllowSubtasks="true" Limit="(null)">
      <Lane Columns="1" Limit="(null)" Queue="0" Name="Active" />
      <Lane Columns="2" Limit="(null)" Queue="0" Name="To Do" />
      <Lane Columns="1" Limit="(null)" Queue="0" Name="Active" />
      <Lane Columns="1" Limit="(null)" Queue="0" Name="Resolved" />
   </Section>
   <Section Name="Output" AllowSubtasks="false" Limit="(null)">
      <Lane Columns="4" Limit="(null)" Queue="0" Name="Resolved" />
   </Section>
   <Integration IntegrationType="tfs">
      <Config>
         <Rows>
            <Row Key="row1" Type="Structure" StructType="Iterations">
               <Stories>
                  <Story Key="story1" ItemType="Requirement" Id="ID" Name="Title" State="State"
                     AssignedTo="Assigned To" StoryPoints="Original Estimate" Description="Description"
                     ParentAssociationType="Property" ParentAssociationProperty="IterationID" Color="1">
                     <StateToLaneMappings>
                        <StateToLaneMapping State="To Do" LaneName="To Do" />
                        <StateToLaneMapping State="Active" LaneName="Active" />
                        <StateToLaneMapping State="Resolved" LaneName="Resolved" />
                     </StateToLaneMappings>
                     <Workflow>
                        <WorkflowItem StateFrom="" StateTo="To Do" />
                        <WorkflowItem StateFrom="To Do" StateTo="Active" />
                        <WorkflowItem StateFrom="Active" StateTo="To Do" />
                        <WorkflowItem StateFrom="Active" StateTo="Resolved" />
                        <WorkflowItem StateFrom="Resolved" StateTo="Active" />
                     </Workflow>
                     <PresetValues>
                        <PresetValue FieldName="Requirement Type" Value="Business Objective" />
                     </PresetValues>
                     <Tasks>
                        <Task Key="task1" ItemType="Task" Id="ID" Name="Title" State="State"
                           AssignedTo="Assigned To" Remaining="Remaining Work" Completed="Completed Work"
                           Description="Description" ParentAssociationType="AssociationLink" Color="4">
                           <StateToLaneMappings>
                              <StateToLaneMapping State="To Do" LaneName="To Do" />
                              <StateToLaneMapping State="Active" LaneName="Active" />
                              <StateToLaneMapping State="Resolved" LaneName="Resolved" />
                           </StateToLaneMappings>
                           <Workflow>
                              <WorkflowItem StateFrom="" StateTo="To Do" />
                              <WorkflowItem StateFrom="To Do" StateTo="Active" />
                              <WorkflowItem StateFrom="Active" StateTo="To Do" />
                              <WorkflowItem StateFrom="Active" StateTo="Resolved" />
                              <WorkflowItem StateFrom="Resolved" StateTo="To Do" />
                           </Workflow>
                        </Task>
                     </Tasks>
                  </Story>
               </Stories>
            </Row>
         </Rows>
      </Config>
   </Integration>
</Board>
```

This is a striped version of a board template. The templates can be found in …/*My Documents/Eylean/Templates* folder.

In order to integrate the board, these steps need to be followed:

1. Integration node has an attribute IntegrationType, which must be set to ***tfs*** or ***teampulse*** values so Eylean Board could pick up on these and load the needed integration implementation.

2. The main integration configuration starts with **Config** node, which has these attributes:
    a. **CanAddRows** (True/False) flag to enable row adding for a particular integration.
    b. **CanRenameRows** (True/False) flag to enable row renaming
    c. **CanDeleteRows** (True/False) flag to enable row deleting
    d. **CanUpdateRowDates** (True/False) flag to enable row date editing. Row (sprint) start and end dates are enabled in board configuration.
    e. **AddOnlyAssignedUsers** (True/False) when set to True, only users which have assigned tasks in the board will be added as workspace members. The default behavior (when set to false or not specified) adds all users as workspace members.

3. **Config** node also has a child nodes:
    a. **Rows** – contains array of **Row** nodes.

4. **Row** node has these attributes:
    a. **Type** (Item/Structure) – TFS only value to set weather Eylean Row is a WorkItem (eg. Sprint) or a structure item (*Iterations only since 1.2*).
    b. ***StructType*** *(Areas/Iterations) – value to set which type of structure item is used (deprecated since 1.2).*
    c. **Key** – configuration name. This value has to be unique between all levels of integration parts (Row, Story and Task).

    Note: The following attributes are only for TFS integration and only if the ***Type*** *is set to* ***Item.***
    d. **Id** – field name to map Id.
    e. **ItemType** – name of WorkItemType to map rows to.
    f. **Name** – field name to map row name value.
    g. *(Optional)* **StartDate** – field name to map (eg. Sprint) starting date.
    h. *(Optional)* **EndDate** – field name to map (eg. Sprint) ending date.
    i. **ChildAssociationProperty** – name of the field which holds the value to map **Row** and **Story**. This attribute is depended on **ParentAssociationType** and **ParentAssociationProperty** attributes of **Story** node.

5. **Row** node also has child nodes:
    a. **Stories** – contains array of **Story** nodes.
    b. **PresetValues** – contains array of **PresetValue** nodes (set this only for TFS and Item type of maping).

6. **Story** and **Task** nodes has these attributes:
    a. **Id** – field name to map Id.
    b. **ItemType** – name of WorkItemType which the configuration node maps to.
    c. **Name** – field name to map items name value.
    d. **Key** – configuration name. This value has to be unique between all levels of integration parts (Row, Story and Task).
    e. **State** – field name to map items' state to.

f. **AssignedTo** – field name to map items assigned user to.
g. **Description** – field name to map description.
h. **ParentAssociationProperty** – name of the field which value is the same as **ChildAssociationProperty** attribute in **parent** node to map **parent** with this **child**. This attribute is depended on **ParentAssociationType** attribute in **this** node and **ChildAssociationProperty** attribute in **parent** node.
i. **ParentAssociationType** (Property/AssociationLink) – this attribute defines how **Story** or **Task** should be associated with its parent. There are two things to keep in mind when setting this attribute:
    i. When the value is set to **Property**, both **ChildAssociationProperty** in parent node and **ParentAssociationProperty** in this node has to be set. The field values of these properties must match to make an association.
    ii. When value is set to **AssociationLink**, the association with parent is done automatically using TFS WorkItemLinkInfo bridge object and Id field. Note that this is possible only if parent and child are WorkItems.
j. *(Optional)* **LinkType** – name of the link between two levels of tasks. The default value is **System.LinkTypes.Hierarchy**, which is a Parent – Child link. Available types of links:
    i. **Scrum.ImpededBy**
    ii. **Scrum.ImplementedBy**
    iii. **Scrum.FailedBy**
    iv. **Microsoft.VSTS.Common.Affects**
    v. **Microsoft.VSTS.Common.TestedBy**
    vi. **Microsoft.VSTS.TestCase.SharedStepReferencedBy**
    vii. **System.LinkTypes.Dependency**
    viii. **System.LinkTypes.Hierarchy**
    ix. **System.LinkTypes.Related**
k. *(Optional)* **StoryPoints** – field name to map story points (or equivalent property).
l. *(Optional)* **Remaining** – field name to map remaining time left or estimation in hours.
m. *(Optional)* **Completed** – field name to map tracked time.
n. *(Optional)* **Color** – this is an integer value from 1 to 8 to set different colors for different types of items. This has no real integration meaning and will be set to default if the value is missing.

7. **Story** has these child nodes:
    a. **Tasks** – contains an array of **Task** nodes. Usually there is a single task node, but for completeness you can configure multiple Task items.

8. In addition **Story** and **Task** has these child nodes:
    a. **StateToLaneMappings** – contains an array of **StateToLaneMapping** nodes which defines a map between items state to a lane in the board as the name suggests. It has these attributes:
        i. **State** – state of the item.
        ii. **LaneName** – name of the lane to map the state.

    Note: when mapping items to lanes, make sure that stories map to lanes which are not details and tasks map to lanes which are only for subtasks in the board.

b. *(Optional)* **Workflow** – contains an array of **WorkflowItem** nodes which defines a transition between two states (*Not lane names!*). **WorkflowItem** has these attributes:
   i. **StateFrom** – state which the item is in. Note, if this state is set to blank, means that item can be created in the state specified in the **StateTo** attribute.
   ii. **StateTo** – state which the item can be transitioned to. Note, if this state is set to blank, means that item can be deleted in the state specified in the **StateFrom** attribute.

c. *(Optional)* **PresetValues** – contains an array of **PresetValue** nodes. These nodes are used for items which require some field to be set, but Eylean Board has no physical way to set the value or some value has to be set during the creation of the item, since item in the board is created empty, except with name set to “New Task”. **PresetValue** has these attributes:
   i. **FieldName** – name of the field.
   ii. **Value** – value to set the field.

8. Final remarks

You can take a look at the different examples already configured for the out of the box templates in TFS and TeamPulse at the templates folder and if that doesn't help, please don't hesitate to email support@eylean.com and helpful support will help you configure the template for your needs!