How to compose Scrum and other Practices using Rational Method Composer

TABLE OF CONTENTS

1. Introduction

This guideline explains how to combine the Scrum practice with other practices in the IBM Practices library. The reader is expected to be familiar with Rational Method Composer authoring concepts and the structure of the IBM Practices library. More information on the IBM Practices library architecture can be found here.

2. Structure and dependencies

The Scrum practice is two plug-ins:

- practice.mgmt.scrum.base-ibm – contains most of the elements, and all of the text
- practice.scrum.assign-ibm – assigns elements to standard categories, as well as assigning roles to tasks and work products to slots. This allows the Scrum practice to be combined with other practices in the library.

The Scrum "base-ibm" plug-in has no dependencies on other plug-ins, with the exception of the IBM copyright plug-in, which is included just to add the copyright notice. dependencies on the rest of the library. The "assign-ibm" plug-in does have dependencies on several core plug-ins so that it can add category and slot assignments.

There is one configuration that includes the Scrum practice, "agile_alm-ibm". This is the IBM Solution for Agile ALM with Scrum. This configuration also includes a plug-in publish.agile_alm.base-ibm that adds solution-specific information, such as how to get started with the solution, and guidance on how Rational Team Concert is configured in terms of work item types and templates. When combining Scrum and other practices, you generally will create your own configuration and "publish" plug-in.

3. How the Scrum practice differs from other Practices

The Scrum practice is different from other practices in the following ways:
ts

1. It does not use the roles defined in "core" – Scrum defines its own roles.

2. It has artifacts like "Increment" and "Product vision" which overlap with work products defined in Core such as "Implementation" and "Vision".

These overlaps are intentional – to ensure that Scrum when used alone is defined accurately. However, it does mean that there are additional considerations when combining Scrum with other practices.

4. General approach for combining practices

The general approach for combining practices to create a configuration is:

1. Create a configuration. Select "Core" and desired practices.

2. Create a process plug-in that defines any new WBS structures.
3. (Optional) Tailor elements and relationships
4. (Optional) Create Rational Team Concert work item templates and process templates.
5. Create a publish plug-in that defines the tree navigation for the configuration.
6. Publish and deploy to a webserver.

This approach can work fine as long as there are no overlaps between elements. However, because Scrum roles, tasks, and work products can overlap with other practices, some additional tailoring is required.

5. Combining Practices with overlapping elements

The Scrum practice overlaps with the roles, work products, and tasks of other practices. In this section we describe how to deal with these overlaps.

5.1. Roles

The standard role for requirements related practices, such as "User story driven development" is the Analyst role. If you want to combine these practices with the Scrum practice, you may want to combine these roles.

One way to do this is to replace both roles with a new role named "Analyst/Product Owner". RMC does not allow you to replace two roles with one role, but you can approximate this as follows:

1. Rename "Analyst" to "Analyst/Product Owner" using an "extends and replaces" element that changes the presentation name, and modifies the description to clarify the responsibilities of this new role.

This can be done "library wide" by putting the replacing element in a "core.default.role_def.extend" plug-in, or could be done just for selected configurations by placing in a "publish" plug-in.

2. Replace the practice.mgmt.scrum.assign-ibm plug-in with a new plug-in that assigns tasks to the "Analyst/Product Owner" role.

3. Hyperlinks within the Scrum practice will no longer work if you remove the Product Owner role element from the configuration.
You can simply ignore these errors using the publishing option "convert broken hyperlinks to plain text".
Alternatively, you can add a new Product Owner element that has a "replaces" relationship to the original Product Owner element. The new element simply contains a link to direct the reader to the new "Analyst/Product Owner" role.

A similar approach can be taken if you decide to combine Scrum Master with other roles.

5.2. Work products

The Scrum practice introduces a “Product Vision” work product. You may want to instead adopt the practice “Shared Vision”, which produces a more elaborate work product named “Vision”. In this case, you would do the following:

1. Suppress the Scrum “Product Vision” work product and Scrum “Develop product vision” task.
2. This can be done in the Team process perspective.
3. Add the “Shared vision” practice to your configuration, and add “Shared Vision” tasks into your WBS structures.
4. Since “Shared vision” involves the “Analyst” role, review “Combining with Practices with overlapping roles”.

Note: There is no need to change the “input work products” for any tasks.
This is because of the magic of work product slots. “Vision” is assigned to the “technical specification slot”, and tasks (including Scrum tasks) that need a “Vision” or “Product Vision” use the “technical specification” slot. If “Vision” is in the configuration, then it will be an input to Scrum tasks.

The other Scrum work products are less likely to require replacement, but if so, similar techniques can be applied.

5.3. Tasks

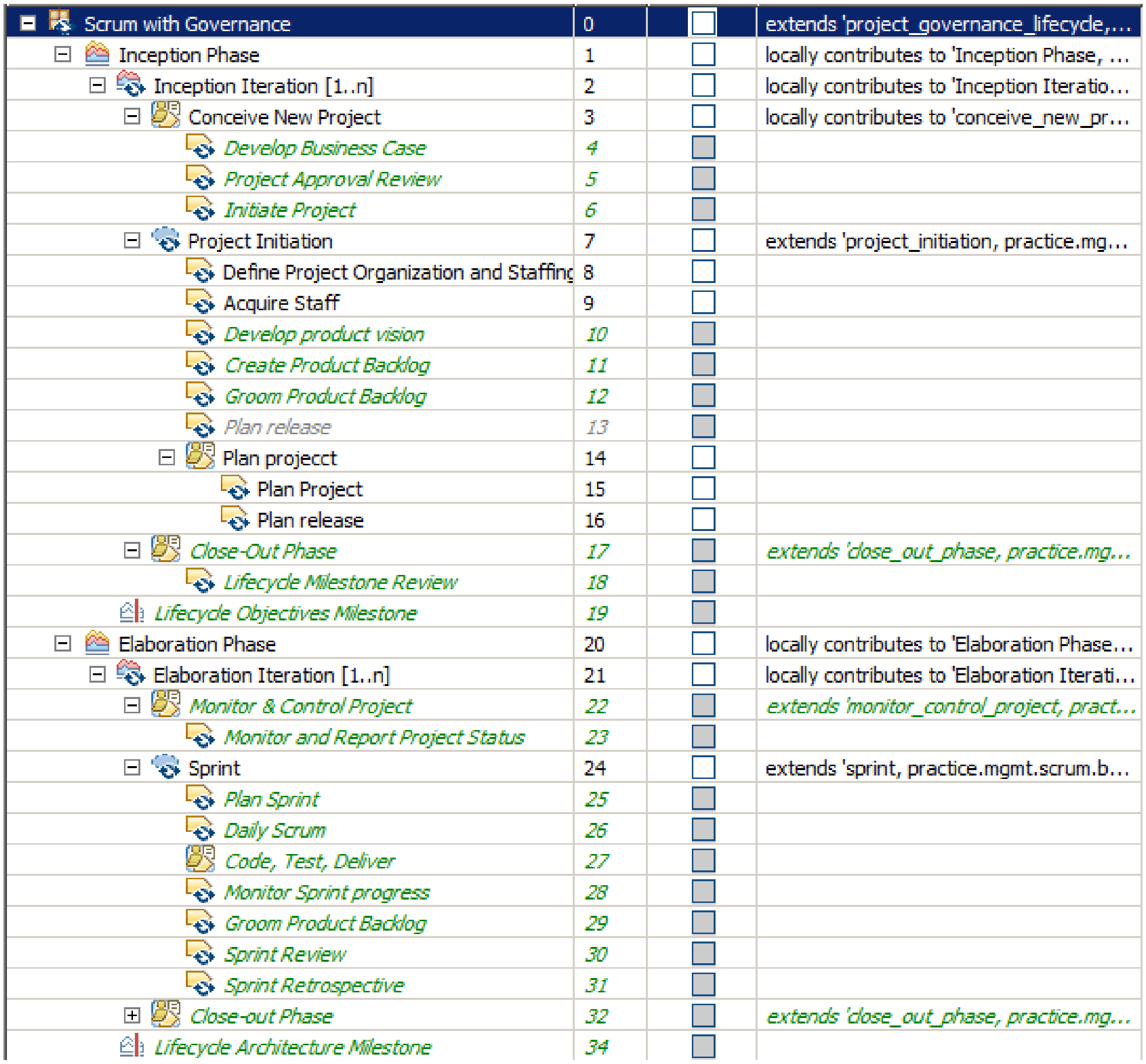
The Scrum practice includes a task for “Plan release”. If you want to combine with the “Release planning” practice, then this task overlaps that practice’s task “Plan project”.

In general, if you’ve managed to address overlapping roles and work products, you can simply suppress any tasks you don’t want, and include the tasks you do want in the WBS structure.

However, you may want to accomplish the goals of both tasks. That is, you may need to produce a project plan using a standard template that must be approved as part of an overall governance process, while still creating a release backlog and using it to drive your Scrum-based process.

You can create a new task that includes steps from both practices. However, it may be simpler to group these tasks in a single activity in your WBS structure so that both tasks are performed at the same time.

Here is an example of Scrum and “Release planning” practice tasks incorporated into the “Risk value lifecycle” practice governance WBS structure.



⊟ Scrum with Governance	0	☐	extends 'project_governance_lifecycle,...
⊟ Inception Phase	1	☐	locally contributes to 'Inception Phase, ...
⊟ Inception Iteration [1..n]	2	☐	locally contributes to 'Inception Iteratio...
⊟ Conceive New Project	3	☐	locally contributes to 'conceive_new_pr...
Develop Business Case	*4*	▨	
Project Approval Review	*5*	▨	
Initiate Project	*6*	▨	
⊟ Project Initiation	7	☐	extends 'project_initiation, practice.mg...
Define Project Organization and Staffing	8	☐	
Acquire Staff	9	☐	
Develop product vision	*10*	▨	
Create Product Backlog	*11*	▨	
Groom Product Backlog	*12*	▨	
Plan release	*13*	▨	
⊟ Plan projecct	14	☐	
Plan Project	15	☐	
Plan release	16	☐	
⊟ *Close-Out Phase*	*17*	▨	*extends 'close_out_phase, practice.mg...*
Lifecycle Milestone Review	*18*	▨	
Lifecycle Objectives Milestone	*19*	▨	
⊟ Elaboration Phase	20	☐	locally contributes to 'Elaboration Phase...
⊟ Elaboration Iteration [1..n]	21	☐	locally contributes to 'Elaboration Iterati...
⊟ *Monitor & Control Project*	*22*	▨	*extends 'monitor_control_project, pract...*
Monitor and Report Project Status	*23*	▨	
⊟ Sprint	24	☐	extends 'sprint, practice.mgmt.scrum.b...
Plan Sprint	*25*	▨	
Daily Scrum	*26*	▨	
Code, Test, Deliver	*27*	▨	
Monitor Sprint progress	*28*	▨	
Groom Product Backlog	*29*	▨	
Sprint Review	*30*	▨	
Sprint Retrospective	*31*	▨	
⊞ *Close-out Phase*	*32*	▨	*extends 'close_out_phase, practice.mg...*
Lifecycle Architecture Milestone	*34*	▨	

This example was created as follows:

1. Added a new plug-in process.scrum_with_governance.
2. Created new capability pattern "Scrum with Governance"that extends the "project_governance_lifecycle" capability pattern from "Risk value lifecycle".
3. Added the Scrum "project initiation" capability pattern to the Inception iteration, and the "Scrum" capability pattern to the other phases.
4. Adding staffing-related tasks from the "Release planning" practice.
5. Grouped "Plan project" task from the "Release planning" practice with "Plan release" from the Scrum practice in an activity named "Plan project".
   These were grouped because they are overlapping in terms of goals and objectives.

Note that a Scrum Master is not a project manager, so having both roles may be ok. However, if you have a Project Manager, they will likely need to participate in some of the Scrum practice tasks. For example, you can add the Project Manager as an additional performing role in “Plan release”, “Plan sprint”, “Monitor sprint progress”, “Sprint retrospective” and “Sprint review”.

Note that combining WBS structures, as shown in this example, is strictly optional. You can keep the governance aspect of a process independent, and create work item templates in Rational Team Concert for the governance tasks separately from the Scrum ones.

6. Conclusion

Combining practices with overlapping roles, work products, and tasks can be challenging. Some general strategies are:

1. Replace one and add relationships from the other (as was done with the overlapping roles example).
2. Suppress one and use the other instead (as was done with the overlapping work products example).
3. Group overlapping elements (as was done with the overlapping tasks example).
4. Keep both and cross-link between them.
5. Keep both, and apply them in separate contexts so there is no confusion.