Rational. software

Rational Method Composer

Agile delivery of software with Rational Team Concert and Rational Method Composer

How to set up a project based on the process template in Rational Team Concert

By
Ricardo Balduino, Senior Software Engineer
Bruce MacIsaac, Manager, Method Content
Peter Haumer, Software Architect
IBM Corporation

December 2009

Contents

Introduction

This article describes how to use IBM® Rational Team Concert™ (see References) during agile development projects. It is based on the assumption that the reader will be using the *IBM Practices for Agile Delivery* process, which is available as part of the IBM® Measured Capability Improvement Framework (MCIF) for Agile Delivery 1.1 and later service offering.

The Practices for Agile Delivery are documented by using IBM® Rational® Method Composer (see link in References). The breakdown structure (tasks and activities) that are documented in this process are mirrored as work items in a matching Rational Team Concert process template. This saves time in creating work items, ensures consistency with the process, and helps team members find the guidance that they need to do their work.

This article is based on the assumption that the reader knows how to use Rational Team Concert work items and process templates (see References). The authors also assume that the reader has the IBM Practices for Agile Delivery **process template** archive file, which is available in Rational Team Concert: `practices-agile-delivery_process_template.zip`

Overview of the process configuration

The IBM Practices for Agile Delivery includes core agile development practices, such as iterative development, test-driven development, continuous integration, and so on, as well as additional practices to address scaling factors, such as team size, governance, and team distribution.

Included practices

The practices referenced are a subset of those available with Rational Method Composer. Table 1 lists the IBM Practices for Agile Delivery.

For more information on Rational Method Composer and the complete set of available IBM practices, see the Rational Method Composer page on ibm.com (listed in References).

Table 1. Practices from IBM Practices for Agile Delivery in Rational Method Composer

Group	Practice	Description
Agile core	Iterative development	This practice helps the team create a solution in increments. Each increment is completed in a fixed period of time called an *iteration*. The team delivers something of value to ensure that the increment is evaluated and feedback is provided.
	Release planning	This practice embodies the concept of high-level planning for the complete project scope (macro) as separate from the low-level (micro) planning for the immediate and next increments or iterations.
	Whole-Team	This practice helps members of the development team organize the team to work effectively.
	Continuous Integration	This practice helps the team members integrate their work frequently (at least daily).
Requirements management	Shared Vision	This practice focuses on defining and communicating an overall vision for the project.
	User Story-Driven	This practice helps the team capture requirements as

Group	Practice	Description
	Development	user stories, which are then used to guide planning, development, and testing.
Architecture management	Evolutionary Design	This practice describes an approach to design that assumes that the design will evolve over time, thus minimizing documentation while still providing guidance for making design decisions and communicating those decisions.
	Evolutionary Architecture	This practice helps the team analyze the major technical concerns that affect the solution, The team captures those architectural decisions to ensure that those decisions are assessed and communicated.
Quality management	Test-Driven Development	This practice describes an approach to development in which test cases are defined first, and then code is developed to pass the tests.
	Concurrent Testing	This practice describes how to fold testing into agile development.
	Test Management	This practice describes the process of managing the software testing effort.
Change and release management	Team Change Management	This practice describes capturing change requests that are managed as part of work item management.
Governance and compliance	Risk-Value Lifecycle	This practice supplements the iterative development and release planning practices with the unified process lifecycle. This lifecycle identifies four phases, each of which attempts to balance value provided against risk mitigation appropriate to the phase.

Work breakdown structure

IBM Practices for Agile Delivery include a process example that groups project work according to the four lifecycle phases:

- Inception
- Elaboration
- Construction
- Transition

The *Inception* phase determines a high-level scope, and it estimates high-level budget and schedule. The *Elaboration* phase implements the fundamental architecture to confirm feasibility; whereas, *Construction* and *Transition* phases develop increments of the solution and deploy them for customer use. The phases are optional, and you can substitute your own. For more detail, see the Risk-Value Lifecycle practice.

Each phase contains one or more iterations (sometimes called *time boxes*), during which increments of work are created.

Within each iteration, there are three sub-phases: Plan, Execute, and Stabilize.

- During the initial Plan sub-phase (which may be a few hours to a couple of days, depending on iteration length), the team focuses on the iteration objectives and plan.
- During the Execute sub-phase, the team performs the planned work.
- In the Stabilize sub-phase, the team focuses on the quality of the final results.

Figure 1 and Figure 2 show two of the work breakdown structures (WBS) that are available in the IBM practices for iterations performed during the Inception and Elaboration phases, respectively.

Figure 1. Inception iteration WBS

Breakdown Element	Planned
Inception Iteration [1..n]	✔
Plan	✔
Plan Iteration	✔
Execute	✔
Develop Technical Vision	✔
Plan Project	✔
Plan Test Effort	
Identify User Roles and User Stories	✔
Outline the Architecture	✔
Stabilize	✔
Assess Results	✔
Refine Project Plan	
Ongoing Tasks	
Manage Iteration	
Lifecycle Objectives Milestone	✔

Figure 2. Elaboration iteration WBS

Breakdown Element	Planned
Elaboration Iteration [1..n]	✔
Plan	✔
Plan Iteration	✔
Plan Test Cycle	✔
Execute	✔
Identify and Refine Requirements	✔
Outline the Architecture	✔
Develop User Story	✔
Perform Integration Test	✔
Stabilize	✔
Fix Defect	✔
Perform Acceptance Test	✔
Refine Project-Scope Work Product	✔
Assess Results	✔
Ongoing Tasks	
Manage Iteration	
Request Change	
Monitor and Control Test	
Lifecycle Architecture Milestone	✔

Scenario: Setting up a project

This scenario describes how to set up a project in Rational Team Concert by using the IBM Practices for Agile Delivery process template. It explains how to perform these actions:

1. Install the process template
2. Initiate a project
3. Create a project plan
4. Create an iteration plan
5. Create work items
6. Assign work items to team members

Install the process template

You need the Rational Team Concert administrator privilege to install a process template. After it is installed, you can use the template to create any number of projects.

If you have the necessary privilege, you can install the process template following either of these Rational Software Information Center help topics, depending on which client you use for access to Rational Team Concert:

- Importing process templates in Rational Team Concert Eclipse-based interface
- Import process template in the Rational Team Concert Web interface

Publish process content

As part of installing the process template, you also need to install the IBM Practices for Agile Delivery process guidance so that team members can consult it easily.

The content published from Rational Method Composer should be available as a WAR (Web archive) file named `process_for_agile_delivery.war`.

In the Jazz server (Tomcat), deploy that WAR file. This will create a folder called `process_for_agile_delivery` on the server, under the */jazz/server/tomcat/webapps* folder.

Set up the process description for your project

The process description for a project created by using the IBM for Agile Delivery process template points to the published process that was deployed as a WAR file in the previous step. If you are using the default local installation of the Jazz server, the .war file will be deployed to http://localhost:9080/process_for_agile_delivery/index.htm. If that is not the location of your server, you must change your link to the process description that is in the `index.htm` file that is designated in your project area so that it points to the correct URL:

1. Change the `localhost:9080` portion of the URL to the correct server name and port number.
2. Keep the rest of the URL intact (`process_for_agile_delivery/index.htm`).

The content of this file should look like the code in Listing 1.

Listing 1. Revised content of the index.htm file

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" ""
title="http://www.w3.org/TR/html4/loose.dtd">"
class="link">http://www.w3.org/TR/html4/loose.dtd">
<html>
      <head>
          <meta content="text/html; charset=utf-8" http-equiv="Content-Type">
                <meta http-equiv="REFRESH"
            content="0;url=http://localhost:9080/process_for_agile_delivery/i
            ndex.htm">
          <title>Go to IBM Practices for Agile Delivery Web site...</title>
      </head>
      <body>
      </body>
</html>
```

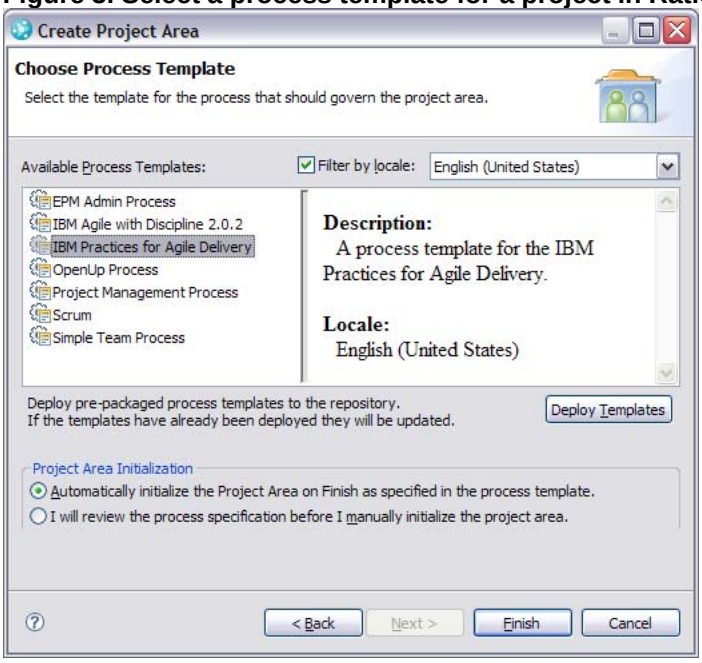
For more information on how to customize the process description for your project, see the Rational Team Concert Information Center: Collaborating > Collaborating using Rational Team Concert > Working with projects, teams, and process > Working in the Eclipse client > Customizing the process in projects, teams, and templates > Editing the process description.

Create a new project and add team members

After you have installed the IBM Practices for Agile Delivery process template, you can use that template to create a new project, as Figure 3 shows.

1. From the **Create a Project Area** window, go to the **Choose Process Template** view and select the template from the Available Process Templates list.

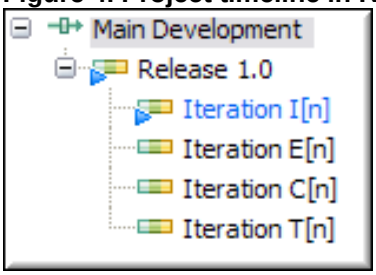
Figure 3. Select a process template for a project in Rational Team Concert



When a project is created in Rational Team Concert by using the *IBM Practices for Agile Delivery* process template, the project timeline is initially populated with one release time box (iteration) and four placeholder iterations that correspond to each of the phases of the Risk-Value Lifecycle practice (Inception, Elaboration, Construction, or Transition).

Figure 4 shows how the timeline looks right after you create a project in Rational Team Concert by using that process template.

Figure 4. Project timeline in Rational Team Concert



Note:
You can rename the iterations to reflect the number of each iteration relative to the phase when it is being performed (for example, Iteration I1, Iteration E1, Iteration E2, and so on). If you need more than one iteration per phase, you can add more iterations to the timeline (for example, by duplicating one of the existing iterations and changing the copied iteration internal ID and name). Some teams prefer to name each iteration sequentially (for example: Iteration 1, Iteration 2, and so on), rather than using letters to represent the four lifecycle phases.

For more information on Rational Team Concert process templates, see the Rational Team Concert Information Center: Collaborating > Collaborating using Rational Team Concert > Working with projects, teams, and process > Project and process concepts.

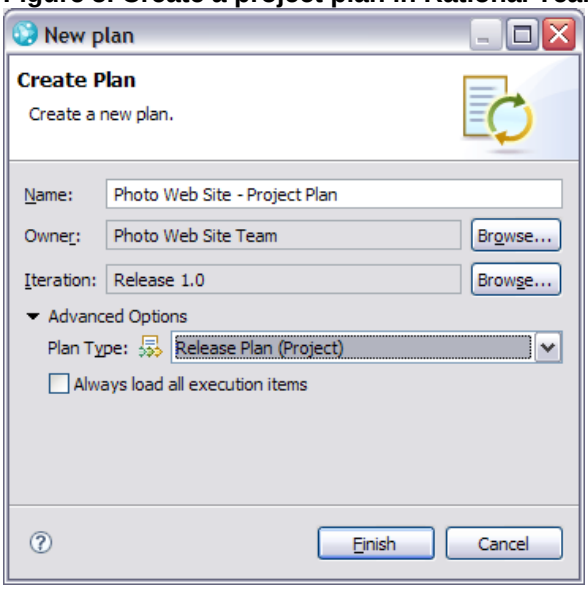
2. Add new or existing team members to the new project, and define their roles in the team (examples: Team Lead, Analyst, Developer, Tester, Architect, or Stakeholder).

Create a coarse-grained plan for the release

Agile teams typically use a *project backlog*, which is a list of prioritized user stories to be assigned to each iteration and developed by the team. To complement the use of a backlog, we suggest that you create a *project plan* (also know as *release plan*) that will contain other relevant project (or release) information, such as project organization, project milestones, objectives for each iteration, deployment information, and so on.

1. Create a project plan for the release (see the Rational Team Concert Information Center: Collaborating > Collaborating using Rational Team Concert > Working with Plans > Working in the Eclipse client > Creating a plan).
2. Edit fields as Figure 5 shows:
   - Type a name for your plan in the Name field (example: `Photo Web Site – Project Plan`).
   - For Plan Type, select **Project Release Plan** (Project).
3. Keep the default selections for the other fields.

Figure 5. Create a project plan in Rational Team Concert

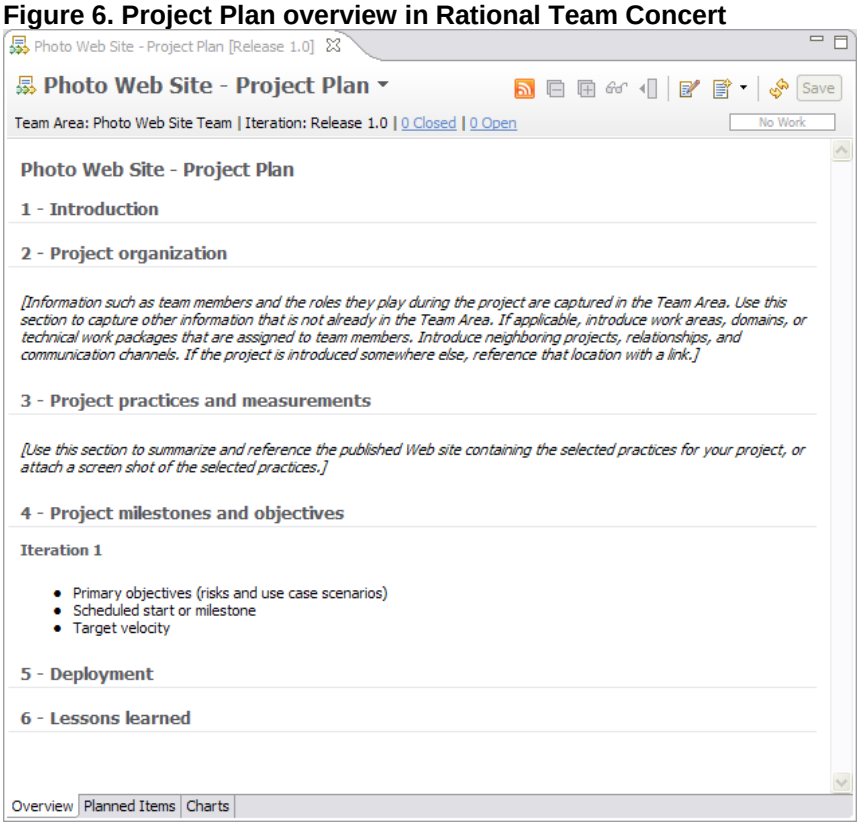


Edit the project plan to insert the standard template

1. Browse the published configuration, and copy the project plan template sections into the project plan in Rational Team Concert, or manually enter the information.
2. Add relevant project-specific content to each section of the project plan.

The result could look like the project plan shown in Figure 6.

Figure 6. Project Plan overview in Rational Team Concert

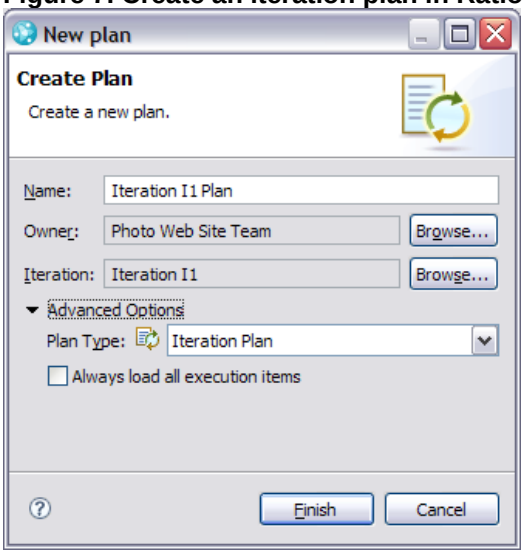


For more information on the Rational Team Concert plan editor, see the **Agile Plans** lesson in the Rational Team Concert Information Center: Tutorials > Do and Learn > Get Started with Rational Team Concert > Lesson 4: Agile Plans.

Create an iteration plan for the first iteration (I1)

1. Create an iteration plan by entering the details, as shown in the example in Figure 7.
   - **Name:** Your iteration plan (example: `Iteration I1 Plan`)
   - **Owner:** Your team area (default)
   - **Iteration:** The iteration that you are creating the plan for (example: `Iteration I1`)
   - **Plan Type:** Iteration Plan

Figure 7. Create an iteration plan in Rational Team Concert

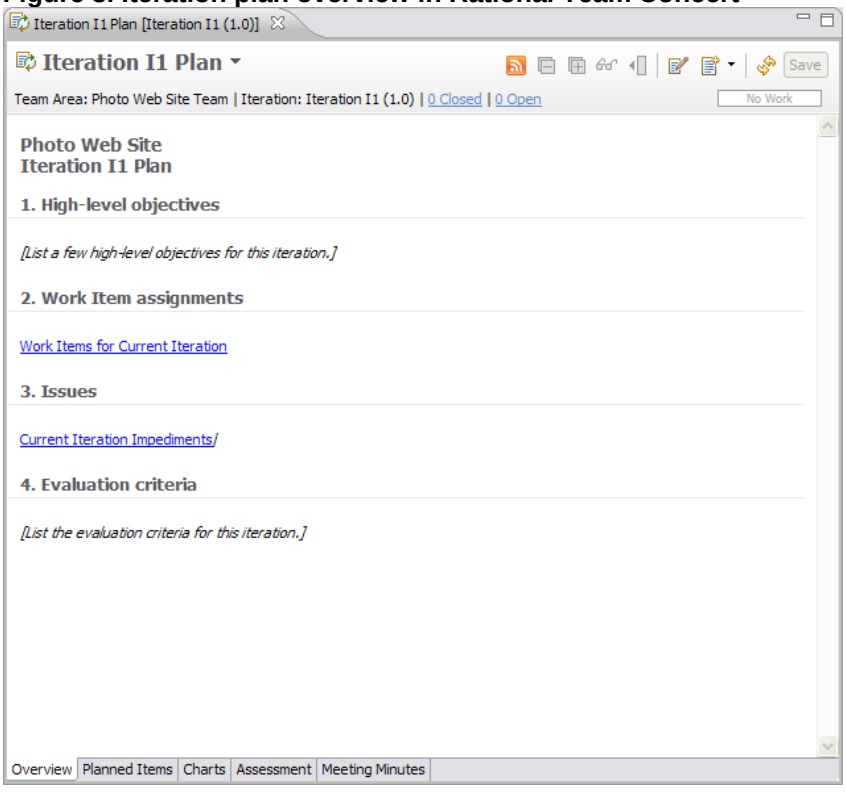


Edit the iteration plan to insert the standard template

1. Browse the published configuration and copy the iteration plan template sections into the iteration plan in Rational Team Concert, or manually enter the information.

The resulting page could look like the iteration plan in Figure 8.

Figure 8. Iteration plan overview in Rational Team Concert



2. Add relevant project-specific content to the Iteration Plan section.

3. Add two new tabs to the iteration plan, one for `Assessment` and another for `Meeting Minutes`. Both tabs have text suggested by the Iteration Plan template found in the Iterative Development practice.

The resulting pages could look like Figure 9 and Figure 10.

Tip:
Fill out the Assessment tab on the last day of the iteration. Fill out the Minutes tab as daily meetings or other meetings happen, adding the most recent meeting information at the top of the page.

Figure 9. Iteration assessment in Rational Team Concert

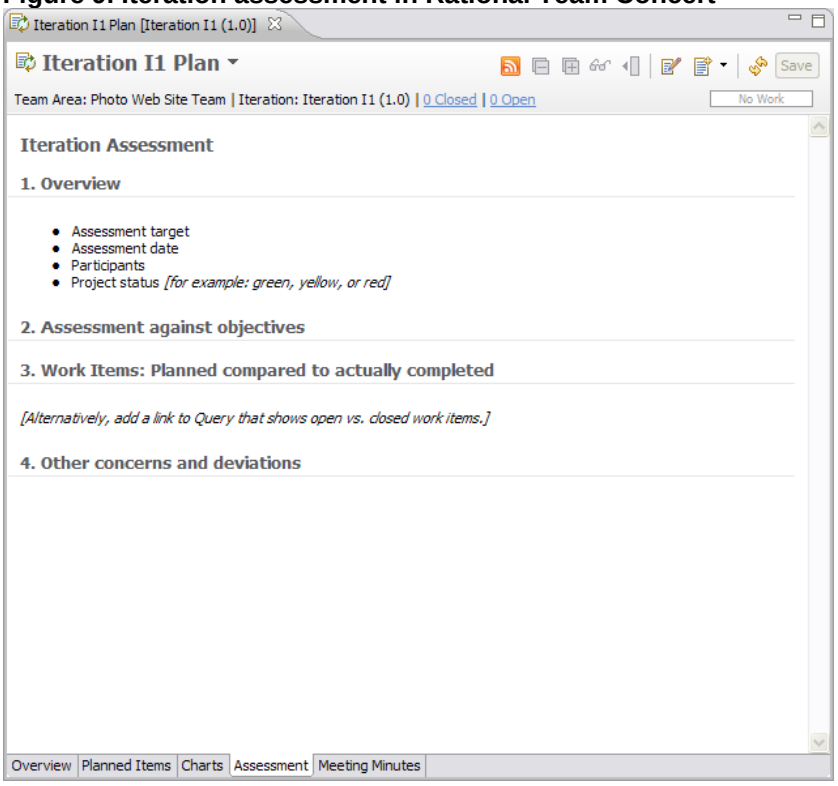
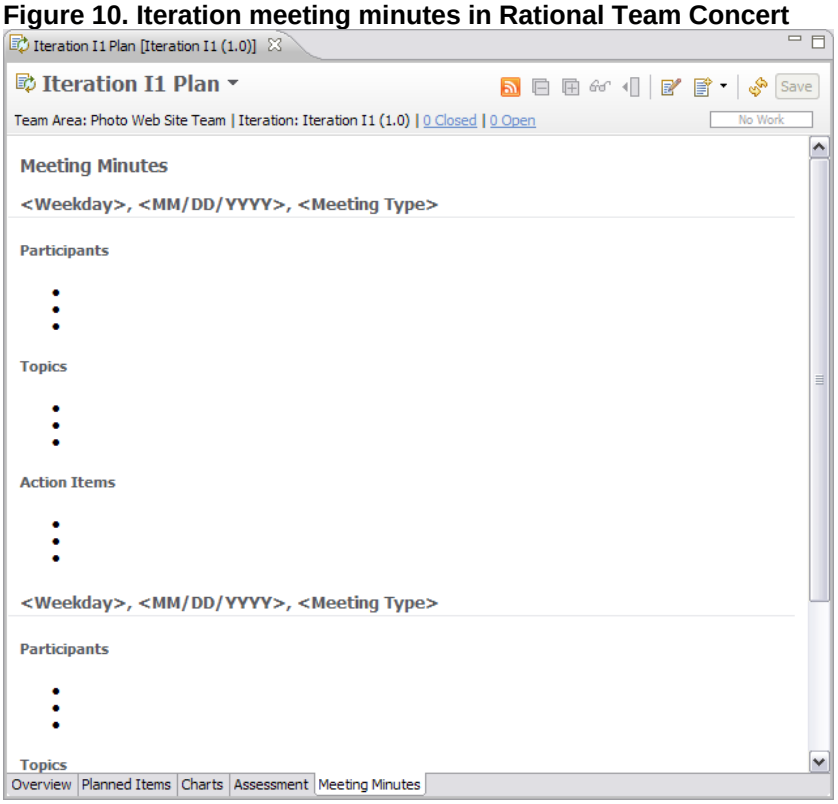


Figure 10. Iteration meeting minutes in Rational Team Concert



Create initial work items

The next step is to create default work items and populate the first iteration with the tasks that the team is supposed to perform, so they can start work.

Note:
The list of suggested tasks comes from the WBS found in the IBM practices, and the tasks are represented by the planned activities and tasks in the WBS from the previous section.

When a project is created based on the IBM Practices for Agile Delivery process template, the typical tasks for the first iteration in the Inception phase of the project are already created as work items in Rational Team Concert. To create tasks for additional iterations, follow the instructions in this section.

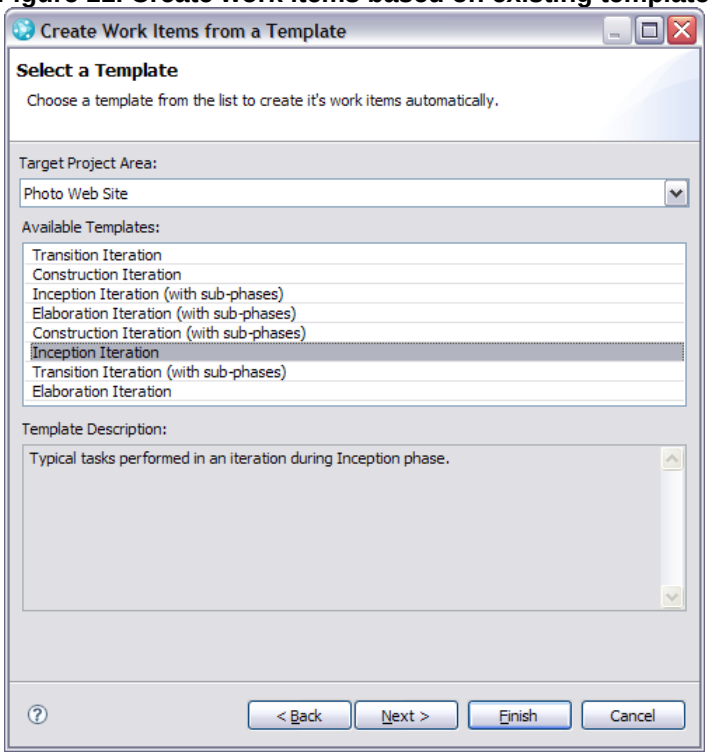
Tip:
If you need only one iteration in the Inception phase, skip to the next section.

Create the default work items (tasks) for an iteration. In this case, instantiate tasks from work item templates available in the IBM Practices for Agile Delivery process template. For example, in the Rational Team Concert Eclipse client, you can create new work items from work item templates by selecting **File > New > Other > Work Items > Create Work Items from a Template**, and following the steps.

You will find a list of work item templates that are typical of each of the four phases (Inception, Elaboration, Construction, and Transition) as Figure 12 shows. There are two types of breakdown structure available for each iteration type: one showing three sub-phases in the iteration work breakdown (resembling the WBS in Section 2 above), and one showing a flat list of tasks (no sub-phases) in the

iteration work breakdown. The options are seen in Figure 11, and this distinction is illustrated in the iteration plans in the next section.

Figure 11. Create work items based on existing templates in Rational Team Concert



Add work items to the iteration plan

In this step, you add the work items of type *task* to the corresponding iteration plan.

There are three different approaches to choose from, according to your preference:

- **Option 1:** Use the default tasks created by using the work item templates available in the IBM Practices for Agile Delivery process template. If you use the work item template that does no contain sub-phases in the WBS, those work items are created as a flat list of tasks in Rational Team Concert. All you have to do after creating the work items is to assign them to a particular iteration, so they can be visualized in an iteration plan, as Figure 12 shows.

- **Option 2:** Organize the iteration tasks into folders that resembles the sub-phases structure found in the Rational Method Composer WBS. You can create your work items based on the work item template that does not contain sub-phases in it structure, and then create the folders in the iteration plan manually:
  1. In the Planned Items tab of the iteration plan, select to **View As Team Folders**.
  2. Rename the existing folders or create new ones to represent the three sub-phases known as Plan, Execute, and Stabilize (the Ongoing Tasks folder is optional).
  3. Allocate the created tasks under the corresponding folder. The resulting iteration plan could look like the plan in Figure 13.
- **Option 3:** Create work items based on the work item templates that already provide the WBS hierarchy, including sub-phases. In this way, there is no need for the manual step of adding

folders to the plan, as illustrated in previous option. The sub-phases are represented as tasks in the plan (see Figure 14).

Figure 12. Planned items for an Inception iteration in Rational Team Concert (Option 1)

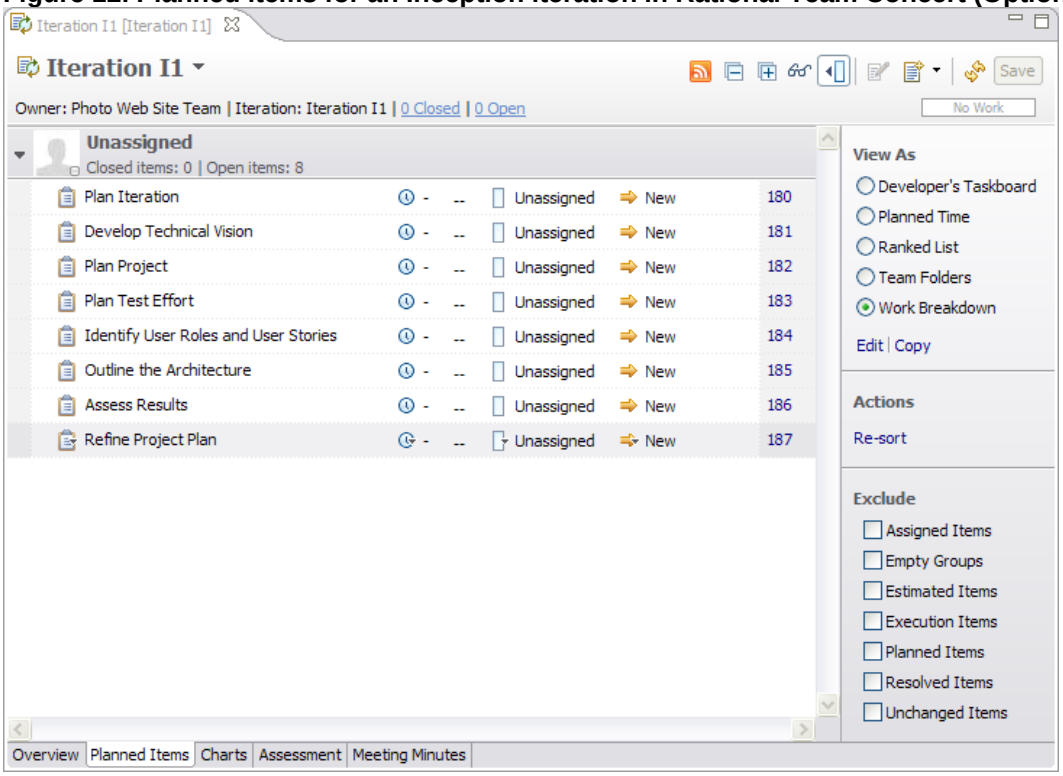


Figure 13. Planned items for an Inception iteration in Rational Team Concert (Option 2)

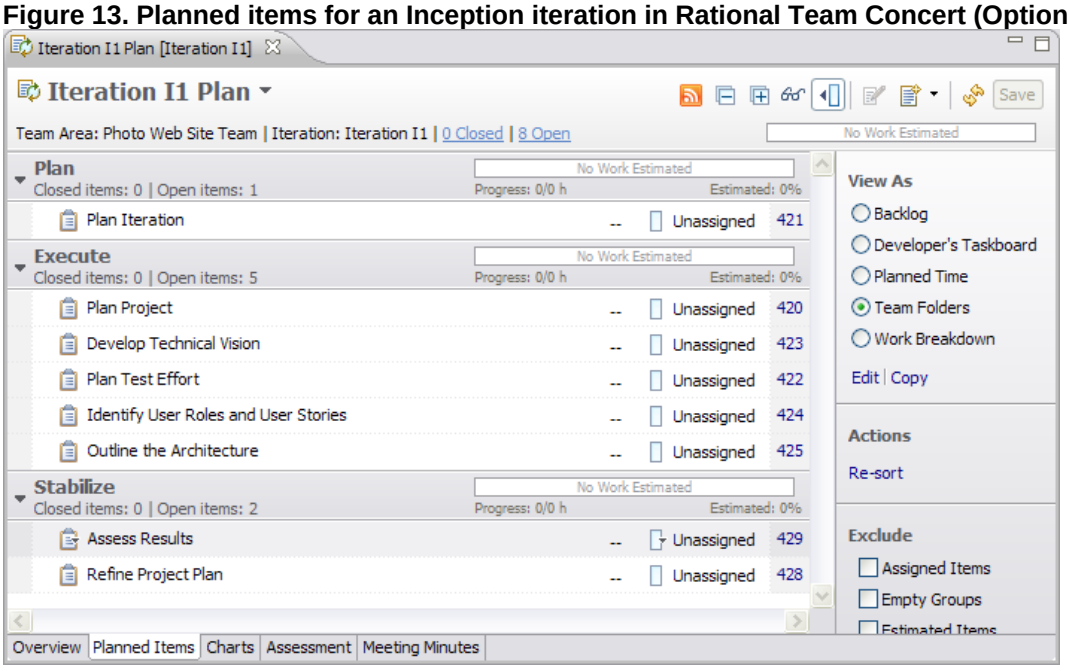
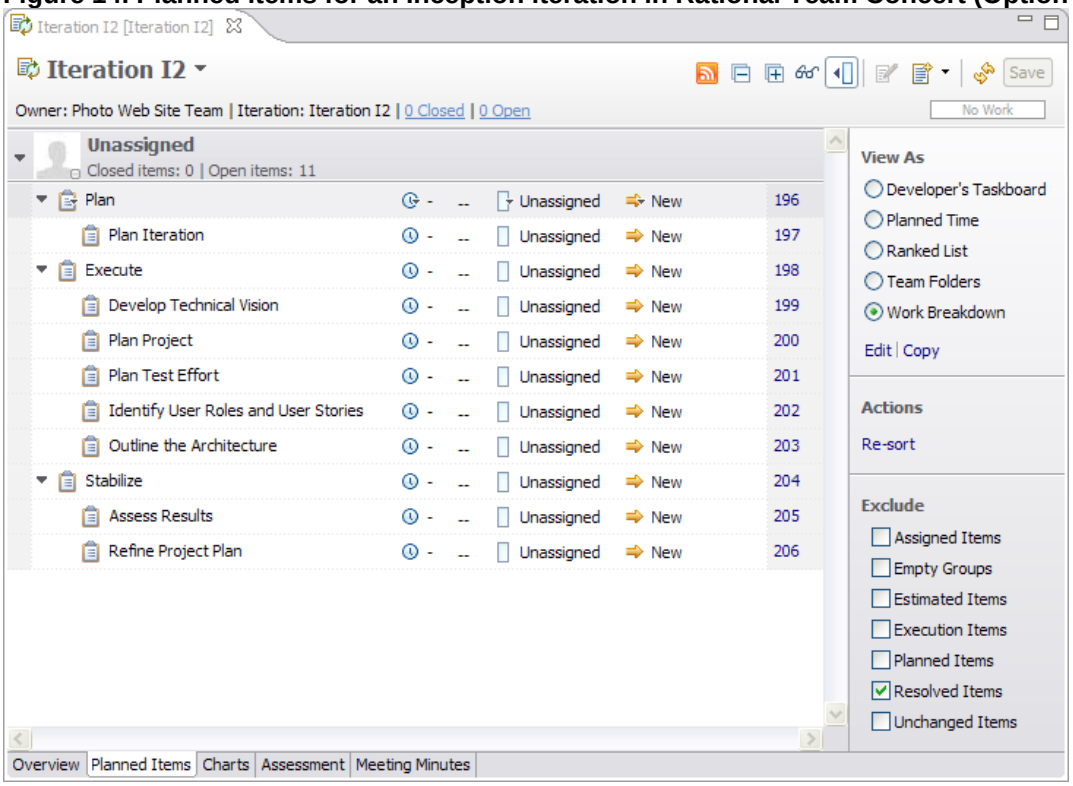


Figure 14. Planned items for an Inception iteration in Rational Team Concert (Option 3)

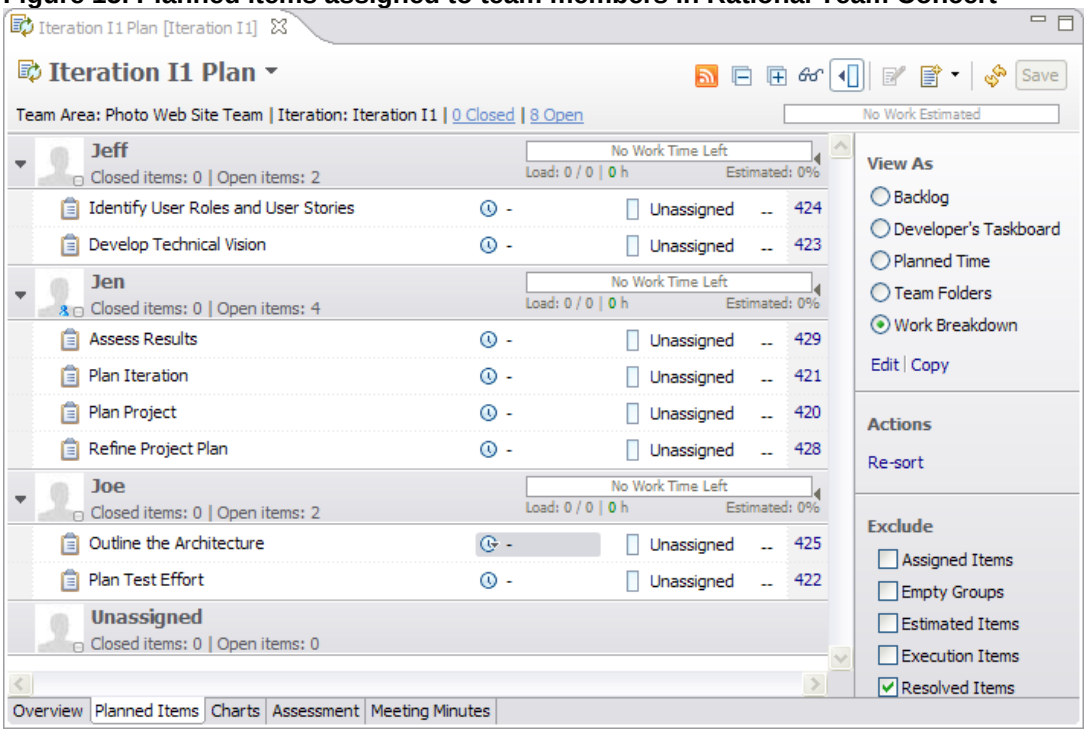


Assign work items to team members

The Team Lead then assigns each task to each person in the team, or each person in the team signs up to perform the work according to roles in the team (Team Lead, Analyst, Architect, Developer, or Tester).

The “View As Work Breakdown” option (Figure 15) lists the team members who can have tasks assigned to them.

Figure 15. Planned items assigned to team members in Rational Team Concert



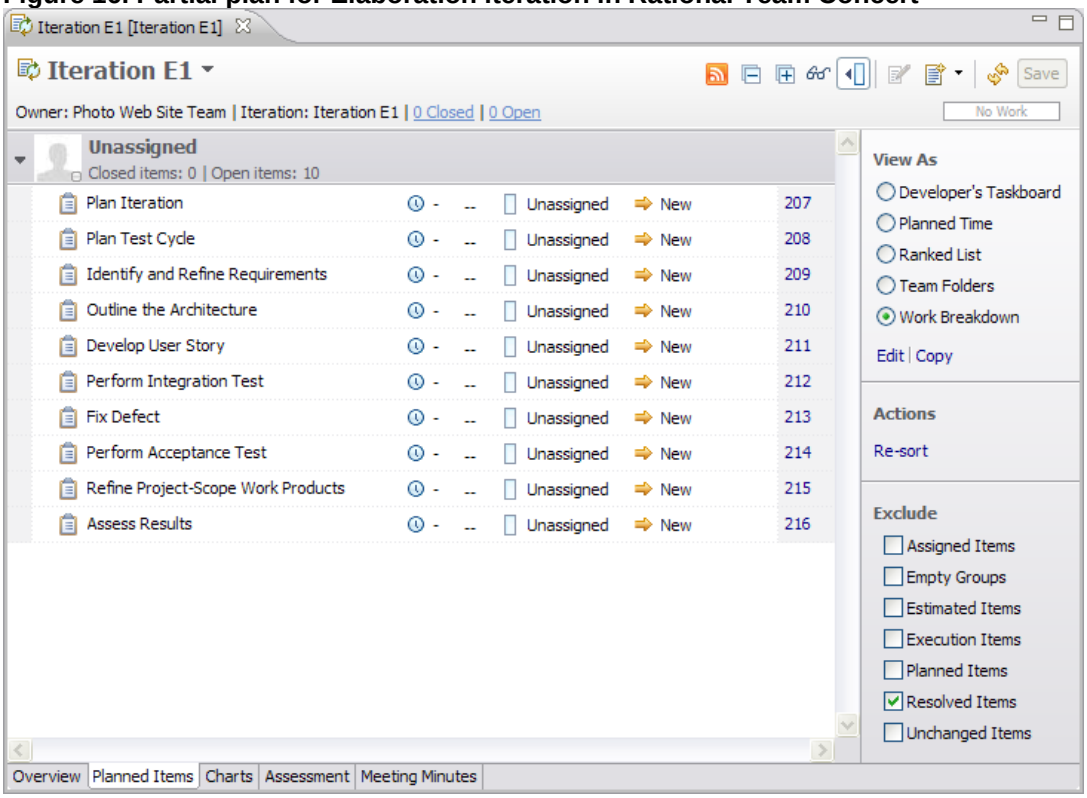
Create an iteration plan for an iteration in Elaboration (E1)

In order to create a plan for Iteration E1, you follow steps similar to those that you performed to create the Inception iteration plan (covered in this article from Create an iteration plan… through Assign work items…). You can also follow the same steps later to create plans for iterations in the Construction and Transition phases.

The partially completed iteration plan for the first iteration in the Elaboration phase (Iteration E1) could look like the plan in Figure 16, which shows a flat list of tasks assigned to the iteration.

This iteration plan is not complete yet. You need to allocate the user stories — which are initially added to the backlog — to this specific iteration. Also, any defects that need to be worked on during an iteration need to be created as work items to be assigned to the team. The sections that follow show you how to do this.

Figure 16. Partial plan for Elaboration iteration in Rational Team Concert



Manage user stories

During the initial project iterations, user stories are identified and documented. There are different ways to capture information for user stories and to track the work associated with each user story.

The Story work item in Rational Team Concert is used to manage the work associated with developing a user story. In Rational Team Concert, you manage the assignment of work to team members who are responsible for capturing the user story statement and its details and then developing and testing the solution for the user story. These various tasks could be assigned to different team members who are playing different roles (such as Analyst, Developer, and Tester), or they can be assigned to one person who plays multiple roles, which might be the case in small or agile teams.

If a team has other tools to capture the user story description, the work item in Rational Team Concert could point to the user story description in the other tool, or it could point to a high-level requirement or feature that originated the user story. Similarly, the Story work item could include a link to detailed information, such as storyboards that are stored in another tool, such as IBM® Rational® Requirements Composer, which can be used in conjunction with Rational Team Concert in this situation. More details on the integration between Rational Team Concert and Rational Requirements Composer are available in the Collaborative Application Lifecycle Management (C/ALM) scenarios (see the citations in References).

For the purposes of this article, the user story content is documented as part of the Story work item in Rational Team Concert. The description of the user story is captured in the work item description itself. Agile teams typically add the identified stories to the backlog (or work items list) for prioritization, further refinement, and capture of details needed about user stories so the development work can start.

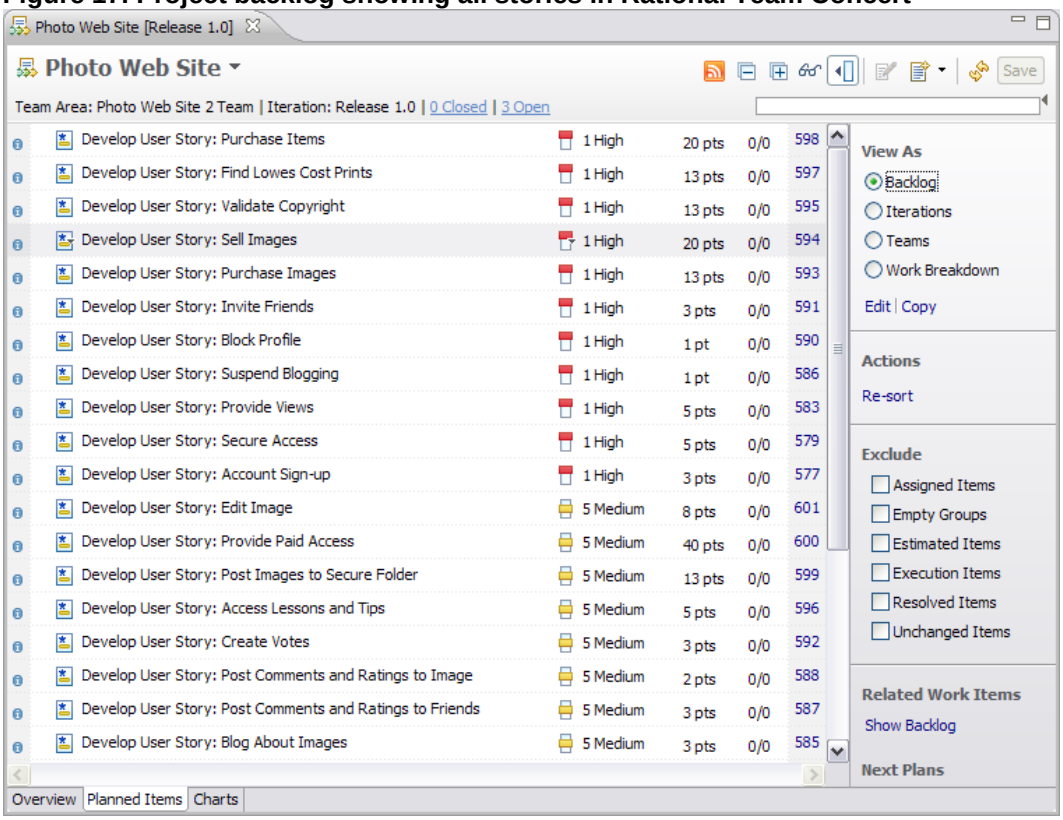
In this article, we want to show you two possible ways to create user stories and assign them to the iteration plan:

- **Option 1:** Start with the Develop User Story task that you added to the iteration plan in the steps that you followed in the previous section. Change its work item type from **task** to **user story**.
  1. Then you can add the user story name to it, for example: Develop User Story: `Purchase Items`.
  2. After that, you add the user story statement to the description field.
  3. You can duplicate this work item to create new user stories as needed.
- **Option 2:** Create new story work items either manually or by importing a batch of user stories from a .csv (comma-separated value) file.

Whether you use Option 1 or 2, the *backlog* (or work items list) is initially populated with user stories, as Figure 17 shows.

From the backlog, you can move the user stories (for example, by dragging them) to the iteration plan when they are ready to be developed and tested.

Figure 17. Project backlog showing all stories in Rational Team Concert



If you used Option 1, you end up with a flat list of user stories shown in your iteration plan (Figure 18).

If you used Option 2, you end up with a list of user stories organized under the “Develop User Story” task (Figure 19).

Regardless of the option that you selected, each user story has an owner who is responsible for its overall development. The owner might prefer (or need) to involve other team members to help develop parts of the user story, such as one person to develop database access, another to design the user

interface, and a third to automate test scripts, and so on. With that process, fine-grained tasks can be added under the Story work item to specify the work needed to develop the various aspects of each user story and to unit-test the user story (each of these tasks can be assigned to a different team member).

Figure 18. Iteration Plan showing flat list of stories in Rational Team Concert (Option 1)

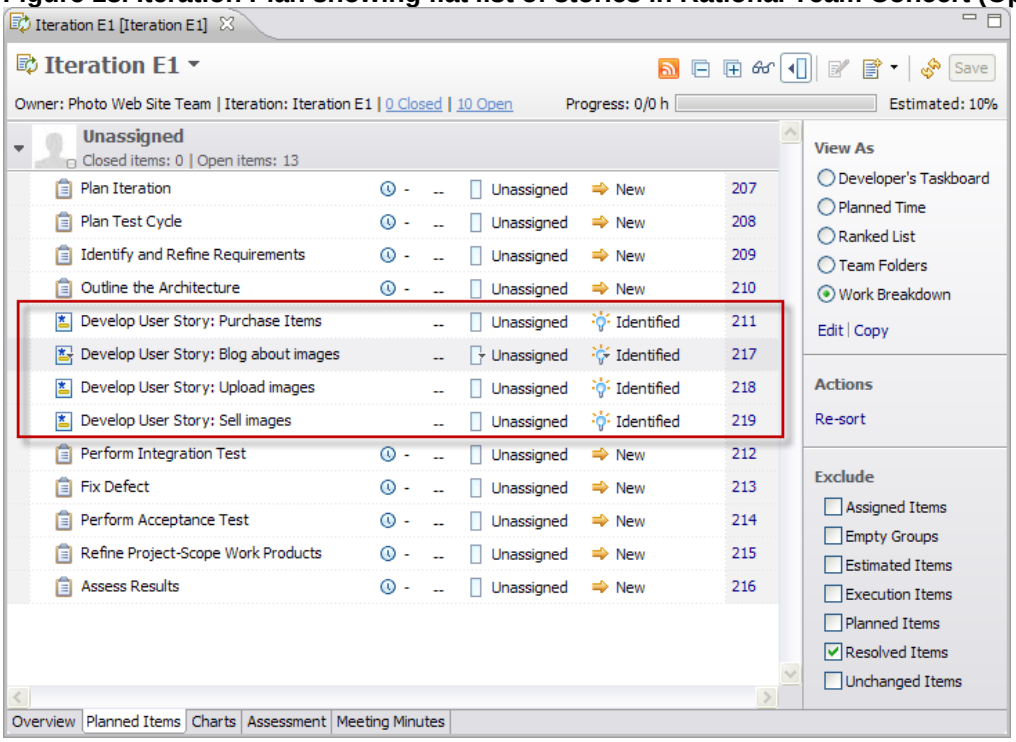
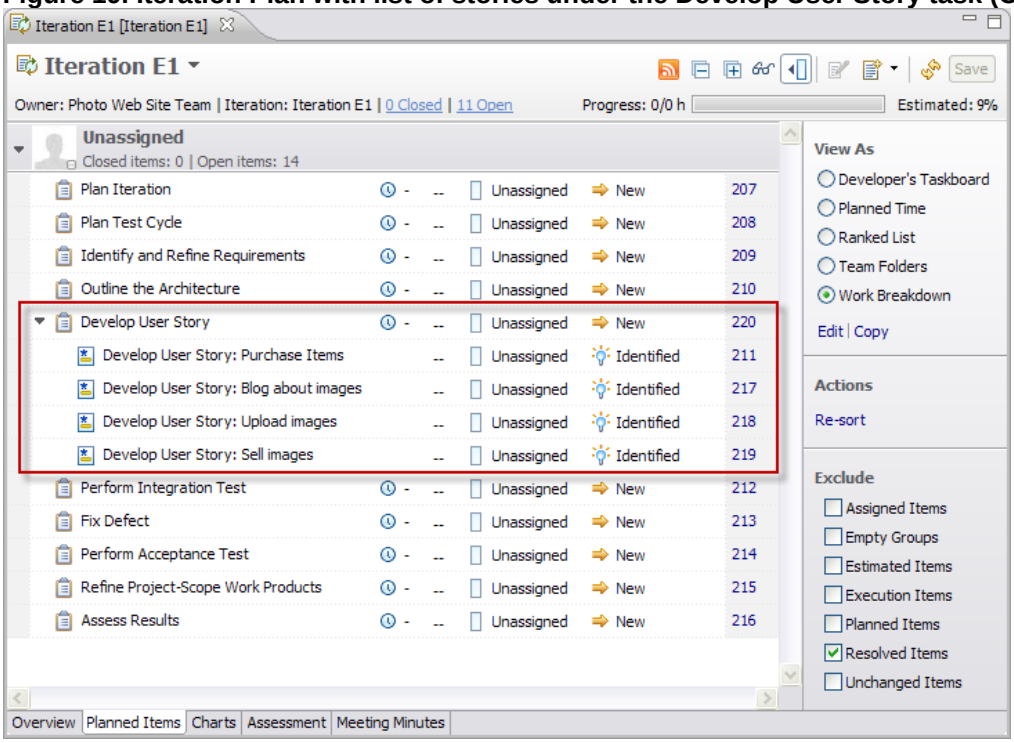
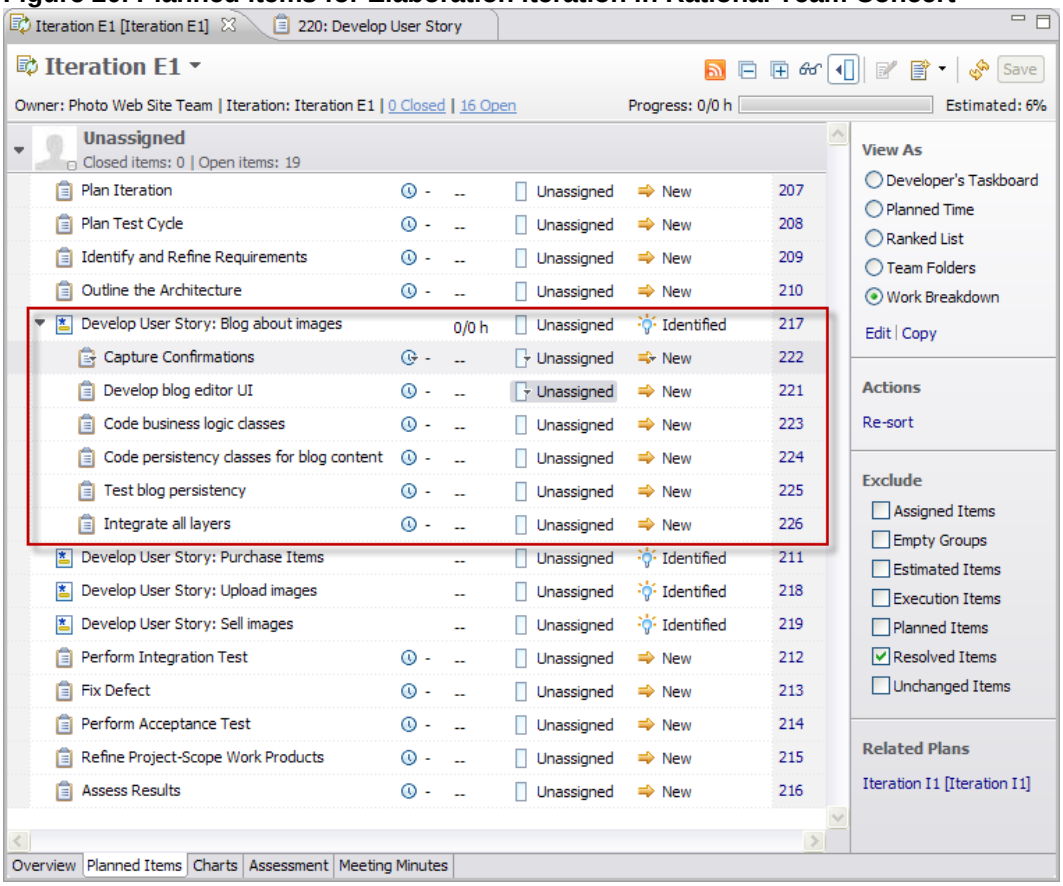


Figure 19. Iteration Plan with list of stories under the Develop User Story task (Option 2)



The plan in Figure 20 adds more information to the partially complete plan shown in Figure 19. It shows how a complete iteration plan would look, including the work items assigned to the iteration and the tasks associated to develop the specific aspects of each user story.

Figure 20. Planned items for Elaboration iteration in Rational Team Concert



For estimating purposes, the sub-tasks should contain the estimates of effort (time, typically in hours or days) needed to complete the work for that user story, which in turn calculates the aggregated amount of time for all the work needed to complete that user story.

For the purposes of tracking the completeness of a user story, each user story is marked as “implemented” after all sub-tasks are marked as “closed.” All individuals who perform the tasks under a user story are responsible for closing their tasks. The user story owner then marks the user story as “implemented.” However, integration and acceptance tests still need to be performed before the user story is marked as “done,” (finished), as discussed in the next section.

Manage tests and defects

In the iteration plan shown in the previous section, there are different tasks that represent test work that needs to be accomplished during an iteration, including planning the test effort, performing tests, and monitoring the results of testing. Those tasks are assigned to team members who are responsible for the testing work.

When it comes to testing user stories, the test cases and test script artifacts used for the testing of each user story can be captured in a separate tool, such as IBM® Rational® Quality Manager. By linking test

artifacts in Rational Quality Manager to Story work items in Rational Team Concert, testers gain transparency into the status of the development work. Developers gain transparency into test coverage and test status for each work item in the iteration plan. This enables the team to collaborate on the iteration work with insight on the effort of the whole team. You can find more details on the integration between Rational Team Concert and Rational Quality Manager in the C/ALM scenario in "Aligning requirements, development, and test," cited in References.

Only when the user story is marked as "implemented" (meaning that the development effort, which includes unit testing and integration work has been done, as described in the previous section), can the user story can go through integration testing and acceptance testing. If the user story passes the tests, then the user story owner marks the user story as "done."

If defects are identified as part of running the tests, the tester creates a work item of type *defect* in Rational Team Concert. This can be done in one of two ways:

- **Option 1.** Change the type of the Fix Defect task to a Defect work item, rename the work item to include the defect summary in its name, and duplicate that work item for as many defects that you need to assign to the iteration. This option shows a flat list of defects in the iteration plan WBS.
- **Option 2.** Create a new Defect work items and assign them as child items of the Fix Defect" task. This method shows the Defect work items nested under the Fix Defect task in the iteration plan WBS.

As part of fixing the defect, the developers might need to "reiterate," which means that the user story reverts to the Inn Progress state and goes through the same workflow already described until it is finished.

Summary

This article showed an approach to mapping a Rational Method Composer delivery process and a corresponding process template in Rational Team Concert, emphasizing the use of practices in the IBM Practices for Agile Delivery.

Rational Team Concert users can create the project and iteration plans and populate those plans with work items of Task, Story, or Defect types. Those work items are then assigned to team members who will perform the work. The team lead can have the view of all work assigned to an iteration and track the status of work (at each iteration and at the project level) in Rational Team Concert.

References

Rational Team Concert:

- Rational Team Concert page on Jazz.net and Jazz.net forums
- Getting Started with Work Items in Rational Team Concert
- Getting Started with Project Areas and Process in Rational Team Concert 2.0
- Rational Team Concert Information Center
- IBM developerWorks page for Rational Team Concert, with links to many other resources
- Webcast: Using Rational Team Concert in a globally distributed team
- Demo: Dashboards and reports
- Podcast: IBM Rational Team Concert and Jazz
- Trial downloads (free):
  - Enterprise Edition
  - Express Edition
  - Standard Edition
  - Rational Team Concert Quick Reference

Rational Method Composer page on ibm.com

Jazz Collaborative ALM (C/ALM): Aligning requirements, development and test (requires free registration)

About the authors

Ricardo Balduino is a senior software engineer at IBM, leading and contributing to the development of solutions such as Eclipse Process Framework, IBM Practices, IBM Measured Capability Improvement Framework, Jazz-based solutions, and the Rational Unified Process. His 15 years of experience in the software industry also includes developing software for industrial process automation and financial services, as well as delivering training and consulting services to help organizations adopt formal and agile software development practices. Ricardo is a certified Scrum Master and certified Project Management Professional (PMP®). He holds a B.S. degree in computer sciences from Sao Paulo State University, Brazil, and an M.S. degree in software engineering from San Jose State University, USA.

Bruce MacIsaac is the development manager for IBM Methods, including IBM Practices for Agile Delivery and the Rational Unified Process. He has more than 20 years of software development and process development experience. Bruce has been a driving force behind the Rational Unified Process and the Eclipse Process Framework for the last eight years. He co-authored *Agility and Discipline Made Easy: Practices from OpenUP and RUP* (Addison-Wesley Professional, 2006). His current focus is providing complete solutions for such areas as agile development, systems engineering, and enterprise modernization, and enabling such practices to be automated on the Jazz platform.

Dr. Peter Haumer is a solution architect at IBM Rational software, responsible for defining next-generation product solutions and service offerings related to process improvement and management. He works closely with clients and IBM stakeholders and also leads development projects that realize products and solutions for enabling development teams to adopt, refine, and continuously improve their agile development practices. One key focus of his research is the vision to directly support IBM® agility@scale™ and continuous process assessment and improvement, as well as practices-based lean governance across all Rational products. He is an in-depth technical expert on the Jazz platform and its collaborative application lifecycle management products. He helps and mentors many client and IBM-internal development teams implement and tailor Jazz-based solutions to support their development processes. Peter received his doctorate in computer science from the Technical University Aachen, Germany.

Appendix: Comparison of IBM Practices for Agile Delivery and scrum process templates

The IBM Practices for Agile Delivery process template in Rational Team Concert supports projects that want to adopt the agile practices mentioned previously in Section 2. Here's a comparison between the IBM Practices for Agile Delivery process template and the Scrum process template:

- Terminology

Scrum process template	Agile Delivery process template
sprint	iteration
release plan	project plan

- Work items
  - The agile delivery process template includes a work item type for tracking *risks*; whereas, what the scrum process calls "impediments" are tracked with a work item type called *issue*.
  - The process template includes these types of work items:
    - Epic
    - User Story
    - Task
    - Defect
    - Risk
    - Issue
    - Retrospective
  - The workflow and states of the Story work item type aligns with the *state transitions* that occur while performing the tasks recommended in the IBM practices content
- Roles
  - Roles are aligned with those of the IBM practices. The agile delivery process template defines the following roles:
    - Team Lead
    - Analyst
    - Architect
    - Developer
    - Tester
    - Stakeholder
- Process documentation
  - We added the Rational Method Composer published process configuration to the Agile Delivery process template, so that team members can access process guidance while they are performing their tasks in Rational Team Concert

Trademarks and Legal Notices