Rational. software

Planning a Project with the
IBM Rational Unified Process

David West

Table of Contents

Introduction

Although the project management discipline outlined in the Rational Unified Process® is often not fully appreciated by the development team, project planning is a critical activity for software development. Good planning helps the teamwork together to achieve a set of defined goals in a defined period of time. This white paper, written for project managers and software development managers who are about to embark on the process of planning for a software development project, describes a practical approach to project planning based on the RUP project management discipline. It discusses how to create a coarse-grained project plan and an iteration plan for each iteration, while customizing the planning process to fit a particular development environment. It also provides a detailed description of how to employ the planning guide tool that's available from IBM Rational to help plan a software development project.

Many of the concepts presented herein have been introduced in the book *Software Project Management: A Unified Framework*, as well as in the RUP. See those resources for more details.

About the Project Plan

One of the biggest problems managers face when dealing with a software development project is that by its very nature, the project is invisible and non-tactile. It's not like building a bridge, where everyone can see the progress that's being made. Because the physical result of the software development project — a running application — and its ongoing progress isn't readily visible, it can be very difficult for the team to visualize and assess the project's status. To deal with this invisibility, the primary practitioners on the project use abstraction. The architect has a UML model, the analyst has a requirements model (use cases), the tester a test plan. The project plan is the equivalent tool for the project manager. It provides an abstraction or model for the project manager to work with, share with the team, and use to perform impact analysis.

In the modern software development environment, it's crucial to have a shared and dynamic vision of the project for the team to access and share. This is the project plan.

A good project plan performs these functions:

- helps the manager plan the cash flow and schedule for the project

- communicates what's going to be delivered and when

- identifies which resources should be available and when they're required

- helps avoid clashes between competing resources on different activities

- helps the team know who's doing what on the project

- provides a basis for measuring progress and expenditures

- gives the planner some baseline to support replanning activities

- helps the customer and management to see what went wrong when a project runs aground

A good project plan has these key characteristics:

- The plan is target based — that is, it identifies something that must be delivered on the project. If the plan is to be used as an aid to motivating the team toward a defined goal, it must provide clear targets for both the team and individuals to measure their performance against the plan.

- The plan enables the project manager to understand which team members are working on which tasks and what the dependencies are between those tasks.

- The plan provides many different views of the information, as required by different stakeholders (customers, team members, and management). For example, it might offer a coarse-grain plan, an artifact plan, a delivery plan, and a worker to-do list, among other things.

- The plan is measurable from a time perspective as well as a project delivery perspective. Often when asked about progress, a project manager can report how much time and money has been spent but can't quantify how much of the system has been delivered. It's important for the project manager and the entire team to know the current state of the project, which key deliverables have been completed, and which key deliverables are forthcoming.

- The plan is up to date. It's connected to the actual tasks being performed on the project and is the first place a project manager looks when assessing progress. If a project plan becomes secondary when assessing performance, it's not being used correctly.

Characteristics of a RUP Project

A RUP project has these two primary aspects that are important to the project plan:

- RUP projects are iterative.
- Project progress is measured against clear milestones.

Iterative Development

The majority of RUP projects are, by definition, iterative. The RUP is an incremental process whereby the overall project is broken down into phases and iterations. The iterations are risk driven — that is, oriented toward mitigating risks — and each one should deliver executable software that's demonstrable and testable against the project's requirements and use cases.

The project manager uses iteration plans to manage the project. Generally, work that falls outside of an iteration plan shouldn't be undertaken. An iteration plan:

- provides a detailed description of the upcoming phase of work,

- defines the worker roles involved, necessary activities, and artifacts to be delivered in that iteration,

- outlines a very clear set of measurement criteria by which progress can be assessed during the iteration and success can be measured at the end, and

- defines specific start/end dates and delivery dates.

Judging the size and number of iterations required for a project is described later in this paper.

Clear Milestones

The RUP identifies four phases for development projects. Each phase focuses the project team on a particular aspect of the project and has associated with it a number of milestones. These milestones help the project manager assess project progress and ensure that the project will deliver required features and will have quality built in. The phases and what they focus on are as follows:

1. Inception — The focus of this phase is understanding the scope of the project.

2. Elaboration — The architecture as well as the requirements of the product being built must be understood by the end of this phase.

3. Construction — The software must be constructed in this phase.

4. Transition — The software must be rolled out to customers during this phase.

In the context of iterative development, the milestones for a phase provide a focus for the iterations. Each iteration moves the project through certain milestones. For example, an iteration within the inception phase would be structured around the need to understand the scope of the project; the iteration(s) would provide the management framework for the team to explore the system boundary, implications of a possible solution, and the size of that solution. The number of iterations would depend on how difficult it might be to define the scope of the project. If the scope were very hard to understand or could be grouped into easily defined pieces, more than one iteration might be needed. If, as is normally the case, it would take one clear piece of work to understand the scope, one iteration would be appropriate.

The milestones defined in the RUP are of necessity quite general; the project manager will need to refine the milestones so they focus the team on the needs of the project in its particular organizational context. In addition, because the aim of an iteration is to mitigate risk, during an iteration the team will be resolving issues that apply not only to the focus of the phase but also to other disciplines, such as architecture, testing, change management, or construction. The manager combines the iterative, risk-oriented approach with the refined milestones to determine the structure of the project plan.

How to Develop a Project Plan

For planning purposes, the RUP clearly distinguishes between project planning and planning for a specific iteration. Fundamentally, project planning involves developing a coarse-grained plan for the entire project, whereas iteration planning deals with developing a fine-grained plan for the specific upcoming iteration. In this section, we'll discuss in detail how to develop a project plan.

Perform Project Start Activities

At the start of any project, you need to determine whether it's in your organization's best interests to engage in the project. This basic question is often overlooked or taken for granted, but the exercise of answering it will result in the all-important two C's: consensus and commitment. The RUP refers to this project planning stage — which consists of determining the economic viability of the project (developing a business case), making a start at identifying and assessing risks, and initiating the project — as the *Project Start*.

Developing a Business Case

In developing a business case, the project manager documents the economic value of the proposed product. The artifact resulting from this activity is the instrument through which funding for the project is obtained and on which the key stakeholders agree. This artifact can be one page or one hundred pages. The important element here is to ensure that you've performed your due diligence as a manager to ensure that it's indeed in your company's best interests to undertake this project.

Keep in mind that the business case is often "make or break" for the product or project, so time spent laying it out is time well spent. Consensus by stakeholders on the economic fundamentals and market need for the product will be essential to gaining funding, resources, and commitment from your organization. This commitment will ideally serve to drive your project's development in the coming weeks or months through completion.

The RUP recommends taking the following steps to develop the business case for your project:

1. Describe the product and the need it fulfills.

2. Define the business objectives and intended market for the product.

3. Define the product or project objectives at a high level.

4. Develop a financial forecast including projected revenues and costs for the project.

5. Describe the project constraints that could potentially impact risk and cost.

6. Describe the options that could impact the project's success.

Identifying and Assessing Risks

Identifying and assessing project risks is an essential startup task. The resulting artifacts will serve as the basis for risk mitigation and the development of iterations in the forthcoming elaboration and construction phases. The RUP recommends that project managers take these steps:

1. Identify potential risks that would decrease the likelihood that the development team will be able to deliver the project with the right features, the specified level of quality, on time and within budget.

2. Analyze and prioritize the risks by estimating the impact of each and the likelihood of its occurrence to determine the risk exposure for each risk.

3. Identify risk avoidance strategies to reorganize the project so that you can reduce or eliminate risks.

4. Identify risk mitigation strategies.

5. Identify risk contingency strategies.

6. Revisit risks frequently within iterations and subsequent phases.

Initiating the Project

The *Initiate Project* activity is carried out after the project's business case is approved. This activity sets up the necessary executive management and project planning teams (if applicable), and also sets out the criteria that will be used to determine when the project has been successfully completed. It consists of these steps:

- Assign a project review authority responsible for overseeing the project. For small projects, this authority can be a single person.

- Assign a project planning team, a group of project team members who will carry out the work of planning the project, maintaining the project plan, and reporting on the ongoing project status.

- Approve project acceptance criteria — objective criteria that will be used by the customer or key stakeholders to determine when the artifacts delivered by the project are acceptable.

Define the Project Organization and Staffing

Assuming that your project is considered viable after the *Project Start* activities, your next activity will be to define an organizational structure for the project and to define staffing requirements based on effort estimates.

1. You should define the project organization based on the characteristics of the project and external constraints, such as existing organizational policy. The RUP suggests defining the organizational structure of the project in terms of positions, teams, responsibilities, and hierarchy.

2. You should define staffing requirements based on the effort estimates for the project, the desired schedule, the chosen organizational structure, and mapping of roles. The RUP recommends defining the numbers, type (skills, domain), experience, and caliber of staff required for the project. In addition, the RUP recommends adjusting the software development team from its baseline as your project moves through its development lifecycle. For instance, your team will normally be heavy on management functions, particularly project management, in the inception phase. When the project moves into the elaboration stage, there will be more architectural staff. In construction, the development staff will be larger. Finally, when your project reaches the transition stage, your team should be heavy on QA or software assessment staff.

Compile the Software Development Plan

The next critical phase of developing a project plan is the creation of the software development plan (SDP). The major activities of the SDP are defining the project structure and estimating the size of the overall project.

Defining the Project Structure

Based on more than twenty years of experience with helping customers develop literally thousands of commercial software projects, IBM Rational has derived a ballpark estimate for project planning purposes of how time should be allocated among the four RUP phases. For general purposes, most projects require 10% of the overall time to be spent in the inception phase, 30% in the elaboration phase, 65% in the construction phase, and 10% in the transition phase, as shown in Figure 1.

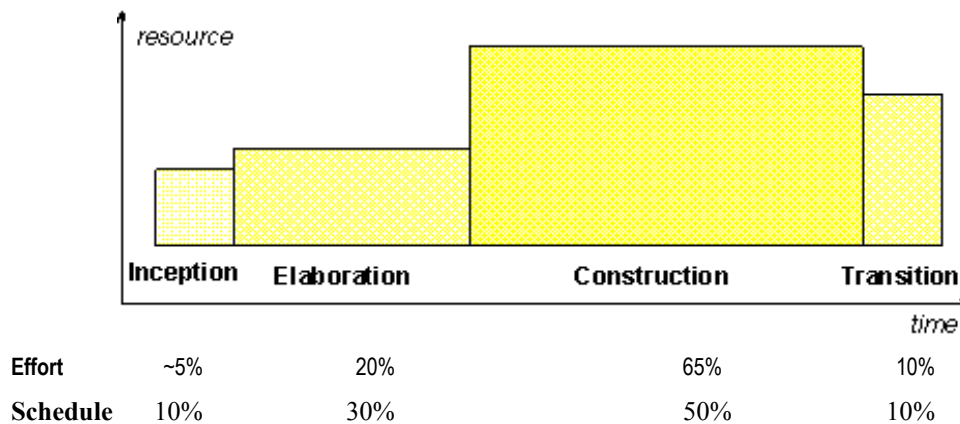|	Inception	Elaboration	Construction	Transition
Effort	~5%	20%	65%	10%
Schedule	10%	30%	50%	10%

Figure 1: Typical time allocation for the four phases of a project

But since this is only an estimate, your project structure may differ. Below you'll find key questions whose answers will shed light on the number of iterations required in each phase of your project and the time and effort required for each phase. Keep in mind that during the life of your project, the structure may change if phase milestones aren't passed or if the project runs into problems.

Inception phase:

Question	Impact
Is the project going to deliver a business system as well as a computer system? Is the project responsible for changing the business processes as well as building a computer system?	Business process re-engineering is a very complex endeavor, and the RUP only provides techniques for business modeling, not for the supporting process. If, however, the business process is simple or well understood, it's possible that work on it may be undertaken in the inception phase. If this is the case, a more complex inception phase will be required.
Is there an existing business system? If so, is it important to understand the existing business system?	If there's a need to understand the existing business system as well as the new system, two iterations are needed — one focused on the existing system and the second on the new system.
Does everyone on the project have a good understanding of the business system that's being supported by the computer system?	If there's not a good understanding on the part of your team, it's worth doing business modeling and increasing the length of the inception phase. Having a clear understanding of the problem you're trying to solve is crucial to a successful project.

If the inception phase is considered complex (as determined by the questions above), the time allocation should be increased by 5% of overall project time. In addition to the relative size of the phase, the number of iterations needs to be defined. If there's a need to investigate an existing system, the number of iterations should be at least two. If the existing system doesn't need to be understood, the inception phase normally comprises just one iteration.

Elaboration phase:

Question	Impact
Is the system under consideration a new system? Has the team built something similar to this system before?	If there's no existing architecture, it will take time to create one. New systems make the elaboration phase more complex.
Is a pre-built framework going to be used for construction?	The use of a pre-built framework provides a number of assumptions about the overall system architecture and really reduces the amount of effort required in the elaboration phase.
Is the technology to be deployed new to the team?	If the technology is new to the team, there's often a steep learning curve associated with its adoption. This will lengthen the elaboration phase.
Does the system need to support concurrent threads of control or have to respond to asynchronous events?	Additional activities are associated with real-time systems development and will lengthen the elaboration phase.

If the elaboration phase is considered complex (as determined by the questions above), the time allocation should be increased by 5% of overall project time. The number of iterations in the elaboration phase is determined by the technical architectural risks that are to be mitigated. Experience indicates that if the architecture is complex or the team is new to the technology to be deployed, at least two iterations will be needed during the elaboration phase.

Construction phase:

Question	Impact
Do you have a separate integration test environment?	If there's a separate integration environment, it may be possible to have multiple construction releases to this environment for integration testing. If there's no integration environment, the number of releases needs to be kept to a minimum.
Is the development team spread over many locations, so that members will be working in parallel?	A development team that's spread over many locations increases the complexity of the build process and lengthens the construction phase.
Are the developers involved with the construction phase familiar with the approach and technology?	If the team is new to the technology or methodology being applied, the construction phase might be difficult and will last longer.

If the construction phase is considered complex (as determined by the questions above), the time allocation should be increased by 5% of overall project time. The number of iterations in the construction phase is determined by both the capability of the team to deliver incrementally and the underlying technical environment. If both the team and the environment are capable of supporting rapid incremental delivery, the number of iterations is defined by the logical pieces of functionality (collection of use cases) that can be delivered.

Transition phase:

Question	Impact
Are you planning to run a beta program?	If a beta program is planned, at least two transition iterations are needed.
Can you deliver the system incrementally? If so, how many increments?	Many business environments can't support a rollout that's undertaken incrementally. If the supporting business system must change or the technical environment is being replaced, the software product must be released in one big bang. It may be possible to determine deployment increments based on the geographical distribution of the business (for example, deploy in France before the UK).

If there are a large number of deployments (increments), the time allocation for the transition phase may need to be increased to 15% of total project time.

Estimating Project Size

At the start of any software development project, sponsors require some sort of estimate of the time and budget that will be required. Estimating the size of a software project at the start is particularly difficult, though. Fredrick P. Brooks, in his book *The Mythical Man Month*, points out three basic characteristics that make software development inherently hard to manage and size:

1. Invisibility — In software development, progress isn't as visible as it is when building a bridge or a factory.

2. Complexity — Software development seems to be inherently more complex than other engineering processes.

3. Flexibility — Software is inherently easy to change. This is often cited as one of its biggest benefits, but the flip side is that software is always being changed because it's so malleable.

These two characteristics also make software project estimation very difficult:

1. Unique applications of software — Traditional engineering disciplines rely on the fact that, because one endeavor is similar to other endeavors, it's possible to estimate one project based on the experience of previous projects. Because of the flexibility of software, each solution is often unique, making it very difficult to compare two projects and use that information to help estimate the size of the project.

2. Changing technology — Even on the same project, technology changes; between projects, the change is often massive. Each time new technology is adopted, it's impossible to estimate how long things will take.

It's true that estimation improves during the course of the project development lifecycle as the team's knowledge and understanding improves, but any approach to estimation must be applied considering the above factors. For detailed discussions of estimation, see *Software Project Management: A Unified Framework* and *Software Cost Estimation with Cocomo II*.

How to Develop Iteration Plans

Once the project manager has completed the coarse-grained project plan, she or he develops an iteration plan at a more detailed level before every iteration. Each iteration in a phase moves the project toward the phase milestones, and iteration plans are the mechanism by which this movement happens. An iteration plan defines a clear objective and associates with it the activities required to accomplish that objective. This section discusses how to develop iteration plans.

Consider What Phase You're In

The phase you're in helps determine the objective of your particular iteration. For example, if the project is in the inception phase, the objectives will be as follows:

1. Establishing the project's scope and boundary conditions, including an operational vision, acceptance criteria, and what's intended to be in the product.

2. Exhibiting, and maybe demonstrating, at least one candidate architecture against some of the primary scenarios.

3. Determining the critical use cases of the system, the primary scenarios of operation that will drive the major design trade-offs.

4. Estimating the overall cost and schedule for the entire project (and, in more detail, for the elaboration phase that will immediately follow).

5. Estimating potential risks (the sources of unpredictability).

6. Preparing the supporting environment for the project.

The factors defined earlier in the planning process will help the manager determine which of these objectives to focus on. Each of these objectives is associated with a deliverable.

Determine the Deliverable(s)

When you understand the objectives of the phase, you understand the deliverables for the iteration — that is, the things produced during that iteration. A deliverable may be something like a vision (of the system) or a series of use cases. Each deliverable has associated with it a workflow template identifying the activities, resources, and other deliverables that will be required to complete this deliverable. It also has associated with it an artifact, meaning the documentation that the project produces for that deliverable and that can be used to review the deliverable at the end of the iteration or workflow.

The following deliverables are samples, based on a "normal" RUP project:

- Business architecture — Describes the business processes and how those business processes are realized within the activity system being investigated. This deliverable is needed only if business modeling is being undertaken.

- Business case — Provides benefit and cost information about the product being built and defines success criteria for the project.

- Vision — Defines what the product does, the market it's aimed at, and the main features of the product.

- Use case model — Defines the functional requirements of the system.

- Supplementary requirements — Defines the supplementary or nonfunctional requirements of the system.

- Use case — Describes a service provided by the system.

- User interface prototype — Simulates the user interface, as defined and testable by users. This is very important if the system has a very complex GUI or has a series of nonfunctional requirements relating to usability.

- Analysis model — Identifies key abstractions that make up the system.

- Subsystem design — Describes the design of subsystems of the analysis model, which will consist of a series of components.

- Risk list — Describes the risks to be mitigated during each iteration.

- Component — Consists of an executable unit of code to be deployed in the executable system.

- Build — Groups a series of components into an entire system.

- Release — Consists of the particular collection of activities that defines a particular release.

- Functional test — Tests the functionality needed to meet a particular requirement.

- Performance test — Tests the performance of the system.

- Test environment — Sets up the test environment for a particular iteration.

- Defect/enhancement — Outlines the process for resolving defects or performing enhancements in response to user feedback on the other deliverables.

- Development environment — Sets up the development environment and manages changes to this environment.

Select the Appropriate Workflow Template

For each deliverable, there's an associated workflow template in the RUP that identifies the activities required to realize this deliverable. An activity includes roles, artifacts, and guidance (help steps). Certain activities may not be appropriate for your particular project or iteration, and these can be omitted. The amount of time allowed for activities will depend on the total time allocated for this iteration in the project plan.

Associate Resources with Activities

Resources need to be associated with each particular activity in the iteration plan. The workflow template provides hints as to which resources are required by defining the roles that need to be played in each activity. The planner must fill the roles with actual workers.

Define Monitoring and Control Processes

How do you know when your project is off track? What do you do to get a wayward project back on course? To answer these questions, the project manager needs monitoring and control processes that check vital signs of the project with regard to the software development plan. Typically, the project manager will be concerned with indicators that apply to the project's scope of work, progress, budget, quality, and risks. The RUP suggests that project managers do the following:

1. Define project indicators to alert the project manager to instigate corrective actions as required.

2. Define sources for project indicators.

3. Define and communicate a procedure and reporting frequency for project team members to report their status.

4. Define thresholds for the project indicators.

5. Define a procedure for project status reporting.

Assess Iteration

After completing an iteration, the project manager and the development team should assess the success or failure of the iteration and capture lessons learned, so they can be applied to modify aspects of the project or to improve the process. This very important step will also help ensure that future projects will reflect the lessons learned in the iteration.

Project managers should do the following:

1. Collect metrics and progress information on the project so that actions can be taken that may involve re-planning, retooling, training, or reorganizing your process, team, and tools.

2. Assess the results of the iteration as compared with expected results, in terms of functionality, performance, capacity, and quality measures.

3. Examine the evaluation criteria to ensure that the goals aren't set too high or too low, that the requirements are still valid, and that the features are still economically justified.

4. Use the results of the assessment to generate change requests for the vision, the risk list, the project plan, subsequent iteration plans, or the development case and requirements, if called for.

How to Use the Planning Guide Tool

The RUP Planning Guide for Microsoft® Project 2002 is a planning guide tool, available for download exclusively on the Rational Developer Network (www.rational.net), meant to simplify iterative project planning for project managers. Using the tool will enable you to schedule the right deliverables and activities for each iteration in your project. To use the RUP Planning Guide as described here, you'll need to install both Microsoft Project 2002 and the RUP Planning Guide template.

This section will show you how to use the RUP Planning Guide to plan an iterative software project. You'll learn how to:

*   jump-start planning of iterative projects with a customizable, pre-populated project plan based on IBM Rational's proven software development best practices,

*   select only the deliverables and activities you need to plan your iterative project, and

*   use the RUP as a knowledge resource for your team based on the experiences of thousands of projects and customers.

The RUP icon — — marks instructions to follow in using the RUP Planning Guide software.

Plan Project Phases and Iterations

Creating a coarse-grained project plan involves planning the number of iterations in each project phase. The RUP Planning Guide helps you jump-start this process by giving you a customizable, pre-populated project plan.

1. When you open the RUP Planning Guide, click RUP Project Plan to get to the guide's main page.

2. Begin by saving your project under a title of your choice.

3. Click Plan RUP Phases to start planning your project. You'll see a default number of iterations set for each phase. These defaults are based on the average project and you can use them as is or change them according to your project's needs. For example, for a small project with a small number of architectural risks, you only need one iteration in the elaboration phase.

4. Click Create the Phases and Iterations. You'll see that the phases and iterations have been created as you specified in the previous step.

5. Click Done to go back to the main menu.

You've now created the coarse-grained project plan that clearly shows the number and order of iterations.

Right-size Your Process

Now you'll begin creating the iteration plan for your project. The RUP Planning Guide will help you to right-size your project plan to reflect your unique project needs. As you customize your process, you can easily customize the guide to your modified process.

By analyzing where in the project your iteration is, the guide understands what should typically be delivered for that iteration and presents a list of potential deliverables. As you specify deliverables, the guide helps you to schedule the necessary tasks for producing these deliverables, all based on experience from thousands of successful iterative projects captured in the RUP. You can choose the level of granularity at which to plan, and as your project progresses you can augment already-made plans.

First you'll create an activity-centered iteration plan. While planning the activities for each iteration, keep in mind the milestones you need to achieve.

1. Click Plan RUP Iteration. You'll notice that there's a link to the RUP tool mentors from this page. These tool mentors are available at various stages of the project plan to guide you through the process. If you click on the tool mentors here, you'll see that they help you make decisions regarding the number of iterations and overall structure of your project. The tool mentors also provide links to guidance in the RUP as well as step-by-step instruction on the planning process.

2. Select the iteration you want to plan and click Create Detailed Iteration Plan. For example, you might begin by planning the first inception iteration, which focuses on understanding the scope of your project.

3. The RUP Planning Guide dynamically presents a list of discipline-based activities you could consider adding to your plan, based on the objectives of the iteration. Tasks that are recommended for the particular iteration you've selected are indicated with a check mark. You can select or deselect activities from the default lists.

4. When you're satisfied with the activities you've selected, scroll down and click Add Tasks. You can see the tasks that have been created for the iteration.

Now you need to decide what artifacts you're going to deliver in this iteration based on a default list that's held in the template. This list can be tuned to reflect the needs of your organization.

1. Select the artifact you want to add from the Artifact list. For example, select Vision from the list to deliver the vision of the system you're building.

2. Enter a name for the artifact. Notice that the activities to deliver the artifact are shown at the bottom of the artifact screen.

3. Click Create to add the artifact as a deliverable. You'll see that the artifact is added to the iteration you had selected.

You can continue to add artifacts and activities as needed. You'll choose the level of granularity by selecting Artifact, Role, or Activity as appropriate. For instance, in some plans you may want to include a very detailed list of the activities necessary to deliver an artifact, while in some other plans it may be enough just to list the artifact. The planning guide lets you choose from three levels of granularity:

- Artifact — This is the coarsest level of granularity, providing one task for each artifact to be delivered.

- Role — At this level of granularity, all the roles involved in working on the artifact are listed to help the project manager understand in a bit more detail who needs to do what, but no detail is provided on what needs to be done.

- Activity – At this most detailed level of planning, a detailed list of all the activities required is provided.

As an example of adding artifacts, let's add a Use Case for capturing requirements to the inception iteration.

1. Select Use Case from the artifact list.

2. Enter a name for the artifact, as before.

3. Click Role and notice that tasks are scheduled for all roles involved. Again, you can select or deselect roles from the default list. You can do the same for Activity.

4. Click Create to add the artifact. Notice that the tasks listed under Use Case are added to your iteration.

5. Click Done when you've finished adding tasks and deliverables to your iteration plan.

You now have a project plan for your first detailed iteration!

Project Lifecycle Guidance from the RUP

The RUP Planning Guide offers guidance to project managers on how to plan and manage iterative development, and to team members on how to perform all the tasks in the project plan. The guide also enables team members to reach templates jump-starting the work of producing required deliverables. This tight linkage between project plan and knowledge base makes the project plan a central focus for all team members.

One form of guidance is available through the tool mentors mentioned earlier. Another option is available through hyperlinks next to each task or deliverable in the project plan.

1. To access guidance for the vision added earlier, click on the hyperlink next to it. This opens a page for the Vision Artifact from the RUP. This artifact provides a link to templates that assist in creating this artifact.

2. To get guidance on how to carry out the Develop Vision task, scroll down to the Output from Activities section and click Develop Vision.

It's possible to explore the whole RUP knowledge base by displaying the navigation tree.

1. Click Display Treebrowser on the top right corner of the page.

2. Select Where am I? to see where this process guidance fits relative to the whole process.

3. From the tree browser, browse the overview of the RUP.

Planning to Succeed

Good project planning is critical to the success of any software development effort. The RUP's project management discipline outlines a process that project managers and software development managers can use in order to produce a good project plan — that is, one that's target based, measurable, flexible, and connected to actual progress, and that defines responsibilities and dependencies for the team.

As you've learned in this whitepaper, there are two plans in RUP projects: a coarse grained project plan and an iteration plan for each iteration. The project plan should define the phase milestones, the number of iterations, and the overall project size estimates. Iteration plans should define iteration-specific items such as deliverables, iteration length, risks to be mitigated, activities, assignment of responsibilities, and measurement criteria. Together, project plans and iteration plans help the development team visualize and assess project status so that working software can be created on time, with required features, and with desired quality.

About the Author

David West is currently the Product Manager for the Rational Unified Process platform, responsible for the definition, delivery, and rollout of the RUP product and process. He has worked in information technology for over 12 years. During that time he worked for a number of organizations, ranging from a large financial institution to consulting firms. While at Rational he has worked as a technical consultant helping organizations adopt the Rational Suite and improve their software capability, as well as a product manager.

References

*   *Software Project Management: A Unified Framework* by Walker Royce (Addison-Wesley, 1998)

*   *The Rational Unified Process: An Introduction*, 2nd ed., by Philippe Kruchten (Addison Wesley Longman, 2000)

*   *Software Cost Estimation with Cocomo II* by Barry W. Boehm et al. (Prentice Hall, 2000)

*   *The Mythical Man Month: Essays on Software Engineering*, 2nd edition, by Frederick P. Brooks (Addison-Wesley, 1995)

IBM software integrated solutions

IBM Rational supports a wealth of other offerings from IBM software. IBM software solutions can give you the power to achieve your priority business and IT goals.

- *DB2® software helps you leverage information with solutions for data enablement, data management, and data distribution.*
- *Lotus® software helps your staff be productive with solutions for authoring, managing, communicating, and sharing knowledge.*
- *Tivoli® software helps you manage the technology that runs your e-business infrastructure.*
- *WebSphere® software helps you extend your existing business-critical processes to the Web.*
- *Rational® software helps you improve your software development capability with tools, services, and best practices.*

Rational software from IBM

Rational software from IBM helps organizations create business value by improving their software development capability. The Rational software development platform integrates software engineering best practices, tools, and services. With it, organizations thrive in an on demand world by being more responsive, resilient, and focused. Rational's standards-based, cross-platform solution helps software development teams create and extend business applications, embedded systems and software products. Ninety-eight of the Fortune 100 rely on Rational tools to build better software, faster. Additional information is available at www.rational.com and www.therationaledge.com, the monthly e-zine for the Rational community.