Planning and estimating a RUP project using IBM Rational SUMMIT Ascendant

Per Kroll

This article describes how to plan and estimate a RUP project using IBM Rational SUMMIT Ascendant, an integrated suite of software engineering practices, project planning and estimating tools, and techniques guidance, delivered through a Web interface.

[Author's note: This article was a collaborative effort with my colleagues Carlos Goti, Michael Hanford, Bruce MacIsaac, Gordon Schneemann, and Bruce Wallman.]

IBM® Rational® SUMMIT® Ascendant® is an integrated suite of software engineering practices, project planning and estimating tools, and techniques guidance, delivered through a Web interface. Used with the software project management best practices in IBM Rational Unified Process®, or RUP®, it can help a project manager estimate high-level cost and schedule early in the project, create detailed plans for each iteration, and manage cost and schedule as the project proceeds.

This article describes how to plan and estimate a RUP project using the IBM Rational SUMMIT Ascendant toolset. We will see how planning and estimation is an iterative process, requiring and enabling continuous improvements of your plans and estimates throughout the project as you gain more knowledge about 1) what work needs to be done, and 2) the effectiveness of your team.

We can divide planning and estimating a RUP project into the following three main activities:

- **Perform initial project sizing.** Understanding a project's size is part of the start-up. This allows you to understand the project in terms of work effort and provides a basis for high-
level resource planning by skill type. You can augment the sizing activity using a variety of estimation techniques, many of which the IBM Rational SUMMIT Ascendant toolset supports. You can periodically revisit the initial project sizing to refine it as you learn more about project characteristics.

- **Produce a phase plan.** In the first iteration, you produce a high-level phase plan that provides initial cost and schedule "scale" information for a project go/no-go decision or possible re-scoping. Later, you can use this high-level plan to check and refine the overall scale of the project using the actual results of each iteration as input.

- **Produce iteration plans.** For each iteration, you produce an iteration plan and use it to guide work and track progress. The iteration plan specifies what deliverables you should produce, who creates them, and how to assess the results, primarily through executable testing.

See Figure 1 for a sample RUP project that demonstrates where these activities occur.

![Figure 1: When to plan and estimate a RUP project](image)

**An overview of software estimation**

Before we go through how to plan and estimate a RUP project, let's walk through estimation in general, and estimation in IBM Rational SUMMIT Ascendant specifically.

**Commonly used estimation techniques**

There are several techniques for estimating and sizing a project. The more commonly used techniques include:

- **Algorithmic models.** These models produce an estimate as a function of several variables, which are considered to be the major drivers of work effort. Algorithmic models include:
  - **COCOMO II**, a method initially coordinated by Barry Boehm and maintained by the Center for Software Engineering at the University of Southern California ([http://sunset.usc.edu/research/COCOMOII/](http://sunset.usc.edu/research/COCOMOII/)).
  - **Function points**, a language-and implementation-independent approach that counts specified user functions.
• **Use-case points**, an extension of function point analysis for projects using use-case-based requirements approaches.

• **QIF-based estimates**, a method developed for use within IBM Rational SUMMIT Ascendant, based on "Quantitative Influencing Factors" derived through IBM’s extensive software estimation experiences working with Fortune 500 companies.

• **Expert judgment**. Experts may be asked to estimate the amount of effort required for the project. Wide-band Delphi is a popular method for collaborative expert judgment estimation.

• **The Analogy-based estimation technique**. The estimator compares a new project to one or more past projects of a similar nature.

• **Parkinson/Effort to win**. This technique is used during the Inception phase only. The definition of project scope, and project inclusions, is "backed into," based on available funding or resources.

You should try to use at least two of these techniques. One technique serves to validate the other. If, however, you discover major differences in two sets of results for project size, you should question whether the goals, scale, and desired results are properly understood.

There are also two models for distributing estimates across a Work Breakdown Structure (WBS):

• With the **top-down** approach, you specify an estimate for the entire project, and then distribute the estimate across the various WBS members. You can combine this approach with, for example, function point analysis and COCOMO II, to provide the top-line estimate.

• With the **bottom-up** approach, you make an estimate for all low-level WBS elements, and then sum up those estimates to produce an overall estimate for the project. You can combine this approach with SUMMIT's QIF-based estimates, as well as with expert judgment, to provide estimates for the low-level WBS elements.

**QIF-based estimation**

QIF-based estimation has been developed specifically for IBM Rational SUMMIT Ascendant and has been used by a number of Fortune 500 companies over the last decade. The estimation is an algorithmic model requiring numerical values for various project characteristics, such as number of use cases, number of reviewers, and so on. These numerical QIFs drive the effort estimates for each task in the WBS.

QIF-based estimation is combined with bottom-up estimation, and each WBS member is calculated as follows:

\[
\text{Estimate} = \text{Multiplier} \times (\text{QIF Count}) \times \text{Formula}(\text{Low Range, High Range}) + \text{Adjustment}
\]

Consider the following example, which is based on the activity of detailing a use case.
The QIF is:

<table>
<thead>
<tr>
<th>The QIF is:</th>
<th>Number of Use Cases</th>
</tr>
</thead>
</table>

A Labor Effort Range is associated with the QIF:

- **Low Range** = 2 hours
- **High Range** = 8 Hours

The number of use cases is estimated at 8 (the QIF Count). Note that there are two ways of setting a QIF count: 1) Allow the estimation template to set the QIF count for you, since when you choose an estimation template, values are set for all QIFs. If you choose an estimation template for a simple project, the template will, among other things, set the QIF count for use cases to 8, as in this example. 2) Alternatively, you can examine the various QIF counts provided by the template selected, and manually specify reasonable QIF counts based on your understanding of the project. For instance, the estimation template may have set the QIF count to 12, but you know that a more reasonable estimate for number of use cases for this project is 8.

The Estimator provides a variety of formulas. Using the average calculation formula, we will set the labor effort at 5 work hours associated with each use case (that is, the low range 2 hours added to the high range 8 hours for a total of 10 hours divided by 2 (the average), or 5 work hours.

So, the Estimate work effort associated with 8 use cases is -- given the average work effort of 5 hours -- approximately 40 work hours.

The Estimator allows you to use a Multiplier and an Adjustment.

You can use the Multiplier to provide some overall percentage change to all, or a selected set of, activities within a project. If the project team consists of all relatively junior staff, the team may work more slowly than an experienced team. The project manager may decide to allow an extra 10 percent of work effort to allow for this experience difference. You can do this by setting the Multiplier to 1.10.

You can use the Adjustment to change the work effort of each activity by some specific number of work hours. The project manager knows that there must be time for some on-the-job learning for the team's junior members, so the project manager allows an extra 2 work hours for each activity to allow for this learning experience. You can do this by setting the Adjustment to 2.

So, the Estimate work effort that was previously estimated to 40 work hours will now be estimated to 1.10 x 40 hours + 2 hours, or roughly 46 hours.

Figure 2 is a screen shot from IBM Rational SUMMIT Ascendant. The text boxes with blue arrows indicate where on the screen to find the key concepts discussed above.
Estimation support in IBM Rational SUMMIT Ascendant

IBM Rational SUMMIT Ascendant supports top-down as well as bottom-up estimation. SUMMIT directly supports many of the specific estimation techniques discussed earlier, including function points, QIF-based estimation, analogy, and the Parkinson/Effort to win estimation models.

If you do COCOMO II, use-case point estimation, or expert judgment, SUMMIT allows you to enter the results of your effort into SUMMIT, and you can then leverage SUMMIT’s top-down estimation support to break down a project estimate by individual component. Finally, if you use expert judgment to estimate lower-level WBS elements, you can leverage SUMMIT’s bottom-up estimation support to understand the overall project effort.

IBM Rational SUMMIT Ascendant’s support for such a broad set of estimation techniques makes it a very powerful tool. There are many ways you can leverage these capabilities for a project, each with its pros and cons. Below we will explore one recommended approach for using many of these estimation techniques when planning and estimating a RUP project.

The Inception phase

Estimation in the Inception phase involves three major steps: performing the initial project sizing; producing a phase plan; and producing iteration plans.

Performing initial project sizing

For the initial project sizing, we will leverage a combination of the analogy estimation model to rapidly produce a rough estimate correlated to similar project types, and the algorithmic QIF-based estimation to modify the estimate to better reflect the specifics of our project. Finally, we may also use expert judgment to override and improve upon the algorithmic estimates.

The technique consists of the following steps:
1. **Choose a planning template.** The template will provide the WBS that you will use for estimation. Adopting organizations can add their own templates, drawn from the actual plans used by past, successful projects. IBM provides out-of-the-box templates for RUP configurations that conform to small, medium, and large process needs. Note that at this stage we only try to understand what type of activities we are likely to spend time on in our project; we are *not* trying to produce a detailed project plan for the entire project.

2. **Select default QIF counts.** IBM provides pre-defined sets of QIF counts for projects of small, medium, and large complexity. The project manager will select one of the default QIF sets as the starting point for a series of sizing scenarios and exercises. This allows the project manager to pre-load estimation data that best reflects past stereotypical projects. You can add QIF count templates to calibrate the method to the organization's experience.

3. **Determine which estimation formula to use.** SUMMIT allows you to choose from several pre-defined formulas for sizing an entire project or project segment. In the section titled *QIF-based Estimation*, we used the average formula, but you can choose from many other formulas in order to increase or decrease your estimate based on your project's specifics. Typically, you use the same calculation for the entire WBS, but you may choose to use different calculations for different elements of the WBS.

4. **Review estimates and modify as required.** You now have an initial effort estimate for all of your WBS elements. By reviewing them, you can assess whether they are realistic or not; you can run additional sizing scenarios using different project assumptions and characteristics; and you can choose the overall sizing formula, as needed. A project manager can override the sizing of one (or more) WBS members, using their own sizing. There is also a summary report that summarizes percent effort by phase, by workflow detail, and by role, and shows the relative impact of each QIF on the estimate. This allows the project manager to better understand and validate the estimates against prior project experience.

Note that both Steps 1 and 2 accommodate necessary adjustments based on your own project or experience. Step 1 above allows you to take advantage of out-of-the-box, experience-based IBM Rational SUMMIT Ascendant WBSs. In Step 2 you have available the out-of-the-box, experience-based SUMMIT calibrations of effort (based on QIFs and QIF counts) for each WBS item. SUMMIT provides you with a variety of useful information, including *overall effort, effort by phase, resource need by role and phase, and resource need by phase and discipline*, as shown in Figure 3. In the next section, we will see how this information will help you build a phase plan.
Figure 3. Resource need by phase and discipline

The accuracy of results from application of estimation techniques varies, based upon the estimator’s knowledge and experience with the specific project. It should be clear that estimates developed early in the Inception phase are not informed by actual work effort and consumption of resources that are known later in the phase. When you create (or refine) an estimate later in the project, you can apply such information from actual experience. You should also treat the estimation of effort as an iterative process.

In other words, it is important to revisit plans and estimates periodically during the project execution. It is also important to set the expectation with the project sponsor(s) that this periodic re-estimation will occur, and that increased knowledge of a project’s scope will certainly affect the timing and sizing of the project as it progresses.

Producing a phase plan

A phase plan is normally driven by a project completion date; at a high level, the phase plan outlines the start and end dates of RUP phases and iterations, as well as the objectives of each iteration. Since most of the information involved in the phase plan concerns the objectives, and is therefore text-based, this information is best captured in a text document -- either standard word processing or spreadsheet format -- attached to the Software Development Plan; see Table 1.

Table 1: Generic phase plan

<table>
<thead>
<tr>
<th>Phase</th>
<th>Iteration</th>
<th>Primary Objective (Risks/ use cases addressed)</th>
<th>Scheduled Start/End</th>
<th>Effort Estimate (Person hours)</th>
<th>Percent effort of total</th>
<th>Staffing</th>
</tr>
</thead>
</table>
| Inception | I1        | - Define vision          
          |           | - Determine project scope  
          |           | - Define candidate architecture                                   | week1/week4          | 1000               | 9         | 6.3       |
| Elaboration | E1          | • Install and test architecture  
|            |            | • Components  
|            |            | • Validate requirements details  
|            |            | • Implement priority use cases  
|            |            | • Test proposed architecture  
|            | week5/week11 | 1400  
|            |            | 13  
|            |            | 5.8  
| E2          | • Mitigate architectural risks  
|            |            | • Complete architecture installation and test  
|            |            | • Implement additional use cases  
|            |            | • Load test architectural elements  
|            | week12/week17 | 1400  
|            |            | 13  
|            |            | 5.8  
| Construction | C1          | • Describe additional use cases  
|            |            | • Design additional subsystems  
|            |            | • Implement use cases and subsystems  
|            |            | • Integrate product and validate state  
|            | week18/week21 | 1900  
|            |            | 17  
|            |            | 11.9  
| C2          | • Describe additional use cases  
|            |            | • Implement use cases  
|            |            | • Integrate product and validate state  
|            | week22/week25 | 1900  
|            |            | 17  
|            |            | 11.9  
| C3          | • Describe additional use cases  
|            |            | • Implement use cases  
|            |            | • Integrate product and validate state  
|            | week26/week29 | 1900  
|            |            | 17  
|            |            | 11.9  

Planning and estimating a RUP project using IBM Rational
SUMMIT Ascendant
### Transition

| Transition | T1 | • Validate state  
• Plan beta program  
• Plan user manual  

| week30/week35 | 1400 | 13 | 5.8 |

### Project Summary

| Project Summary | week1/week35 | 10900 | 100 |

The basic input to the phase plan comes from the QIF-based estimation described previously. But you should manage the phase plan as a separate artifact, for several reasons:

- The phase plan may require the combined results of different estimation techniques, not just the QIF-based estimation.
- Because the phase plan will be updated as necessary with data from the iteration plans (see next section), the overall effort -- and hence the start and end dates of phases -- will likely be different from the overall effort you may have derived from the QIF-based estimate. As shown in Figure 3 earlier, there are no dates indicated, but rather labor estimates. If this data is sent to a scheduling tool such as PMOffice or Microsoft Project, then the detailed plan in the scheduling tool will require start/end dates for each project activity, and you will have to adjust the phase plan to reflect these dates.
- You typically do not want to include low-level WBS elements in your phase plan.
- In addition, the phase plan usually includes start and end dates for each iteration.

Typical steps in producing a phase plan are as follows:

1. Estimate the required staffing level for each phase. This estimate is provided through the QIF-based estimate.
2. Determine a duration for the project by summing phase durations.
   - Phase duration (weeks) = phase effort (hours, according to estimation)/number of full-time equivalent staff/staff hours per week. A more detailed analysis that considers project-specific characteristics, such as availability of staff with the right skill set, may produce a more accurate schedule. The summary report in IBM Rational SUMMIT Ascendant of work per role per phase can help with this.
3. Decide the number of iterations per phase.
• Iterations within a phase have start and end dates. The iterations do not overlap.
• Iteration duration = phase duration/number of iterations in phase.

4. Outline objectives for each iteration.
5. Update coarse project plan throughout the project.
   • This step provides an overall roadmap against which you can track progress.
   • When used in conjunction with scheduling software, this allows you to realize when you have delays and need to re-plan your project.

An important decision is how many iterations are needed in each phase. Table 2 shows project characteristics that tend to increase the number of iterations in a given phase.

Table 2: Project characteristics that can increase the number of iterations in a given phase

<table>
<thead>
<tr>
<th>Phase</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception</td>
<td>• Working with new functionality (unprecedented system for the team) &lt;br&gt; • Unknown business environment &lt;br&gt; • Highly volatile scope &lt;br&gt; • Make-buy decisions</td>
</tr>
<tr>
<td>Elaboration</td>
<td>• Working with a new system environment (new architectural features) &lt;br&gt; • Untested architectural elements &lt;br&gt; • Need for system prototypes</td>
</tr>
<tr>
<td>Construction</td>
<td>• Lots of code to write and verify &lt;br&gt; • New technologies or development tools</td>
</tr>
<tr>
<td>Transition</td>
<td>• Need for alphas and betas &lt;br&gt; • Conversions of customer base &lt;br&gt; • Incremental delivery to customers</td>
</tr>
</tbody>
</table>

Because an iteration is a milestone where the current product is integrated, tested, and validated, the duration of each iteration should be brief. If it is too long, you delay chances to adjust your plans as requirements change and the software under development is certified to work (which may not happen). Long iterations tend to reduce the effectiveness of your focus on risk mitigation. Depending on project size, the iteration duration should be between one and two months. Project size tends to increase iteration duration, but long iterations may cause the team to lose focus.

Iterations have a cost. They need to be planned and assessed, which involves team effort. Phase Plans should be a balance between effort consumed to control a project and effort to deliver results.

As we can see, IBM Rational SUMMIT Ascendant provided us with much of the critical information we need for phase planning. Teams leveraging SUMMIT to manage their iteration plans (see "Producing iteration plans" below) should consider using SUMMIT to manage the phase plan schedule information discussed above. Having all schedule-related information managed within SUMMIT facilitates, among others, status reporting and portfolio management.
Producing iteration plans

As a project manager for a RUP project, you typically, at any given time, actively work with two iteration plans: One for the current iteration, and one for the next iteration. The purpose of an iteration plan is to:

- Define the scope of the iteration, expressed in terms of concrete objectives, such as:
  - Mitigation of specific risks
  - Detailing specific use cases
  - Implementing and testing scenarios and/or subsystem functionality
- Validate and/or adjust the iteration scope by:
  - Estimating effort for each objective
  - Comparing bottom-up estimates for the objectives with overall coarse plan estimates
- Create a detailed plan of how the iteration will unfold
- Set the order and composition of builds
- Increase maturation of specific artifacts
- Create, identify, and manage task dependencies
- Assign tasks to individuals

Using IBM Rational SUMMIT Ascendant for iteration planning

IBM Rational SUMMIT Ascendant can be very helpful for:

- Creating tasks
- Assigning effort estimates to tasks
- Assigning task dependencies
- Assigning tasks to individuals

Since the underlying data format is identical to the data structure of a Microsoft Project database, you can open and edit the iteration plan can be opened and edited in Microsoft Project, as well as make changes in either Microsoft Project or in SUMMIT.

If you are using IBM Rational SUMMIT Ascendant to do iteration planning, you need to decide how tasks will be structured. One simple way to structure tasks so that you can compare them with the estimations done for the phase plan is to use the same WBS and add more detailed tasks.

For example: Suppose there are two Elaboration iterations, and you are planning the first one. A possible set of steps for creating the iteration plan using IBM Rational SUMMIT Ascendant are:

1. Copy the Elaboration phase section of the coarse estimation plan as a starting point.
2. Divide the estimates by number of iterations in the phase (in this case, divide by 2).
3. Add tasks for specific iteration goals:
   - "Detail Use Case X" becomes a subtask under Refine the System (see Figure 5).
   - "Implement Scenario Z" becomes a subtask under Implement Components.
   - Subdivided workflow details become summary tasks.
4. Provide an override estimate for each new subtask (overrides are further described in "The Elaboration phase and beyond" section below).
   - Total effort for the iteration must match available effort
5. Add dependencies between tasks as needed.

• Use builds to drive the tasks and limit the number of dependencies. Assign tasks to specific people. Use IBM Rational SUMMIT Ascendant and Microsoft Project to manage the plan, as shown in Figure 4.

![Figure 4: Example IBM Rational SUMMIT Ascendant iteration plan](image)

Other iteration planning approaches

Some project managers may prefer a different WBS structure for iteration planning: for example, a WBS organized around artifacts. Also, some project managers prefer to manage tasks more informally, such as managing tasks as a list; in this case, the team collaborates to manage dependencies and ensure that the iteration objectives are achieved.

Table 3 shows an example format for an informal iteration plan organized around artifacts.

**Table 3: Example artifact-based iteration plan**

<table>
<thead>
<tr>
<th>Inception Iteration Plan - Example Format</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Artifact</td>
<td>State of Artifact</td>
<td>Start/End Dates</td>
<td>Effort (person days)</td>
<td>Role Responsible</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Scheduled</td>
<td>Actual</td>
</tr>
<tr>
<td>Vision</td>
<td>Baselined</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supplementary Specifications</td>
<td>Baselined</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Business Case</td>
<td>Baselined</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Risk List</td>
<td>Baselined</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Development Plan</td>
<td>Baselined</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Architecture Document</td>
<td>Drafted</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Development Plan - Phase Plan</td>
<td>Baselined</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iteration Plan</td>
<td>Completed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iteration Assessment</td>
<td>Completed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use-Case Model</td>
<td>Drafted</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use Case “A”</td>
<td>Identified</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Jim

Jane

Bob

Jane

Jim
The Elaboration phase and beyond

With the conclusion of the Inception phase, you have a solid base of actual work effort to measure against the projected work effort, and you can refine and enhance the project sizing and project schedule data from actual experience. You should go back and revisit QIF counts and estimation formulas used, as well as remove WBS elements that are no longer relevant. You should consider overriding some estimations of individual WBS elements, since at this stage you probably have a much more accurate estimate for how much time you will spend on various activity types than derived earlier from the QIF-based estimation. At this point, you also have a better understanding of what resources will be available so that you can perform appropriate changes to staffing profiles, and corresponding iteration lengths based on resource leveling.

Using the revised data, we re-estimate the subsequent phases and iterations by going through each of the three activities described in the Inception section of this article, but now with improved data, as shown in Figure 5 below. In particular, we can measure productivity and use this as the basis for re-estimation. As better information becomes available, the project manager typically replaces QIF-based estimates with overrides that reflect the improved project understanding, and this replacement occurs more frequently over the course of the project. Note that this is made possible by QIF-based estimation being combined with a bottom-up estimation model, which is different from other common algorithmic models, such as COCOMO II, function points, or use-case points.

Figure 5: Example of an estimation override

For example:

- In Inception, we thought we would have 8 use cases for a system, and that we would spend roughly 5 hours (using average formula, see section QIF-based Estimation), for each use cases in the activity Detail a Use Case. In Elaboration, we realize that we will probably have

<table>
<thead>
<tr>
<th>Architectural Proof of Concept</th>
<th>Prototyped</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Risk -- Performance of legacy database</td>
<td>Prototype measured</td>
<td>Sally</td>
</tr>
<tr>
<td>Phase Reviews -- Project Approval</td>
<td>Completed</td>
<td>Jane</td>
</tr>
</tbody>
</table>

**TOTALS**

- Architectural Proof of Concept
- Prototyped
- Bob
- Risk -- Performance of legacy database
- Prototype measured
- Sally
- Phase Reviews -- Project Approval
- Completed
- Jane
11 use cases, and that we spend 8 hours describing each use case. We will update the QIF count for use case to 11, use the formula "High," and then have SUMMIT re-estimate our project.

- As subsystems are identified, and use cases are described and allocated to subsystems, you can obtain effort estimates for implementing subsystem functionality. You can also use these estimates for bottom-up effort estimation.

As these tasks are performed, we can compare the estimated effort for the task against the actual effort, and improve the estimates for those tasks that are not yet complete. You may compare estimates based on specific artifacts and project metrics with the original QIF-derived estimates, but typically supercede those estimates.

**Conclusion**

IBM Rational SUMMIT Ascendant supports a broad set of estimation models, and you can use many of these effectively when planning and estimating a RUP project.

The analogy estimation model allows us to quickly produce estimates that reflect experiences from past projects that have been captured in planning templates and default QIF-count templates.

The QIF-based estimation model allows us to specify a limited number of estimation parameters to capture the project characteristics that will have the most impact on the overall project effort. This helps us rapidly refine the analogy estimate to reflect the specific characteristics of our project.

The QIF-based estimation model provides us with valuable estimates of how many resources of what type was needed at what time in the project. This allows us to produce a more accurate phase plan.

IBM Rational SUMMIT Ascendant allows us to continuously improve our estimate as we gain a better understanding of what needs to be produced, the productivity of our team, and staff being available to us. We use this information to improve our QIF-based estimate, and we also leverage the override capability to replace QIF-based estimates with our expert judgment, as we understand what it will take to carry out low-level task.

QIF-based estimation is combined with a bottom-up estimation model, which makes it different from other algorithmic models, such as COCOMO II and function points. This is crucial to allow a continuous and incremental refinement of the algorithmic estimates using expert judgment of low-level tasks. The override mechanism in IBM Rational SUMMIT Ascendant enables capturing the expert judgment.

Finally, IBM Rational SUMMIT Ascendant allows us to manage all schedule and estimation information, including providing an audit trail for planning and estimation activities. This makes SUMMIT a powerful tool for planning and estimating a RUP project.

**Notes**

1 A RUP project consists of four phases; Inception, Elaboration, Construction, and Transition.
The Rational Unified Process provides guidelines for performing productivity metrics.