The Rational Unified Process – An Enabler for Higher Process Maturity

Annie Kuntzmann–Combelles, Q-Labs France

Philippe Kruchten, Rational Software Canada

A Rational Software and Q-Labs White paper

Q-Labs®

Rational®
the e-development company™

TABLE OF CONTENTS

Abstract

This paper highlights the key concepts that a mature organization – a Level 3 development unit – has to demonstrate and how the Rational Unified Process® (RUP®) components match these requirements.

Both high project maturity and organization maturity are addressed. In addition, section 4 provides some good ideas to get started with the RUP and some of the major benefits observed by early adopters in various contexts are reported.

Are you efficient enough?

Today, companies must continually search for ways to achieve a higher degree of performance. The challenge of the future is to compete at a global level. These common symptoms become even more obvious when bidding on the worldwide market:

- **Need for performance**: too many bugs are observed at the validation phase and an unstable version of the product is put on the market to meet the delivery target
- **Need for efficiency**: projects overcome budget and planning
- **Loss of market share**: competitors perform better and deliver better quality
- **Lack of competent resources**: the turnover is high and/or hiring new engineers is extremely difficult
- **Need to integrate evolving technologies with minimal risks**: technologies supporting products have to follow the trends; teams are not always experienced enough to meet the challenge

More generally, we can say there is a strong need to optimize all facets of the developed product.

Therefore, making the widely accepted assumption that the process to develop the product impacts heavily upon the results, one of the most common scenarios would be to address the software process to improve products.

Organizations facing this type of situation will probably come to the following solution to succeed.

- The first part of the solution is to understand the software value for the market. If software continues to be a key competitive advantage for products and projects, then processes to specify and implement them have to be carefully defined, documented, and optimized over time. We observe that senior management is still frequently convinced that software creates problems and arrives too late; value-added analysis for software has not been performed.

The second part of the solution is to dedicate adequate investment to fix the main problems of your processes:

- Assess the process to identify improvement opportunities
- Don’t reinvent the wheel: learn from others’ experiences and accelerate the breakthrough
- Consider both the management and the technical perspective on any project
- Manage the skills you will need tomorrow: learn, share, and grow

Several roadmaps exist to transform an organization into a highly productive team; one of the best decisions you can make is to change what you can, accept what you can’t change, and borrow everything you can that has been validated by a successful software community. The Capability Maturity Model (CMM) [6] and the Rational Unified Process (RUP) [11] are two examples of widely accepted, robust tools that can accelerate your progress towards adequate processes for developing your products. We use the term tool because, while these two components are effective supports for changing and improving software practices, they let you define the best practices for your organizational context.

The next section describes these two components—CMM and RUP—as well as some other key components you have to consider to improve the process and the delivered products.

Roadmaps for improvement

In the last decade, the software community became process conscious and realized that software products cannot be evaluated independently from the process that produces them. As a result, a number of organizations created programs to improve their software development processes known as Software Process Improvement (SPI) initiatives. In most cases, software development is concentrated into one or more units or departments of an organization and is considered an independent discipline. This situation changes when the SPI program is in place, and the software teams are better integrated with the other disciplines of the organization.

What do organizations need to start an SPI program?

- A stimulus to start the program and a key person to coordinate improvement activities; in other words, to set an improvement project with a vision statement and objectives.

- A reference model against which to evaluate the project's and organization's practices and to identify any missing parts that may impede success. The CMM model developed at the Software Engineering Institute (SEI) is a de facto reference used by thousands of organizations and the SPICE framework (ISO 15504) [9] is an alternative that might soon be an official standard with which CMM will be compliant.

- A method to manage the improvement program to closure with successful results: the IDEAL [4] framework defined at the SEI has proven its strengths.

- A development process that can be adapted for the success of each project: the RUP is an acceptable off-the-shelf solution.

The CMM model [6]

The Software Engineering Institute (SEI) Capability Maturity Model (CMM) is a framework that describes the elements of an effective software process [6]. The CMM describes an evolutionary improvement path from an ad hoc, immature process to a mature, disciplined process. It presents sets of recommended practices in a number of key process areas that have been shown to enhance software development and maintenance capability. The CMM guides developers on how to gain control of their development and maintenance processes, and how to evolve toward a culture of software engineering and management excellence.

The IDEAL method [4]

Software Process Improvement is a systematic, collaborative, and long-range method to evolve the way software work is organized and performed. Improvement methods include the IDEAL method, which is an integrated approach for SPI defined by the SEI. IDEAL identifies five phases: Initiating, Diagnosing, Establishing, Acting, and Leveraging. Each of these phases is centered on a particular activity:

- Specify business goals and objectives that will be realized or supported **(Initiating).**
- Identify the organization's current state with respect to a related standard or reference model **(Diagnosing).**
- Develop plans to implement the chosen approach **(Establishing).**
- Bring together everything available to create a "best guess" solution specific to organizational needs—for example, existing tools, processes, knowledge, and skills—and put the solution in place **(Acting).**
- Summarize lessons learned regarding processes used to implement IDEAL **(Leveraging)**.

The process

Some organizations have no process; some have processes based on their experience. There are some off-the-shelf processes available, such as OPEN and the Rational Unified Process (RUP). The OMG Group may propose a generic process model very soon. The term "process" is largely used in the Software Engineering Community, but the assumptions about the components of the software process are very heterogeneous. Let's look at the definition given by the SEI: a set of activities, methods, practices, and transformations that people use to develop and maintain software and associated products. The process is one key element that ensures a project matches its defined objectives.

It is important to notice that this definition includes activities, technologies (that is, methods and tools) and people. None of these three components is more crucial than the others.

The Rational Unified Process

The RUP is one example of a pre-existing process framework that benefits from long project experience [10, 11].

The RUP emphasizes addressing high-risk areas very early, by rapidly developing an initial version of the system, which defines its architecture. It does not assume a fixed set of firm requirements at the inception of the project, but allows you to refine the requirements as the project evolves. It expects and accommodates changes. The process does not put either a strong focus on documents or 'ceremonies', and it lends itself to the automation of many of the tedious tasks associated with software development. The main focus remains the software product itself and its quality, as measured by the degree to which it satisfies its end-users and meets its return on investment objective altogether.

It is generic enough to be tailored to a wide variety of software products and projects, both in size and application domain and it is centered around three areas: people, process, and tools or methods [10,11]

ISO Documents

Finally, to complete this big picture, we must mention ISO documents such as ISO 9001, ISO 12207, ISO 15504 (SPICE) [9], which are available references so organizations can compare their software development know-how with other organizations. They form the generic framework with which the other previously mentioned components are compliant.

ISO 15504 (also known as SPICE) is another reference model to analyze software processes. It assumes that many fully realized assessment models (CMM is one) and many assessment methods (the method defined by SEI is one) will be able to be mapped to the normative parts of ISO 15504.

In the rest of this paper, we focus on CMM, which is the de facto standard of the software community. CMM is a fully realized model that supports hundreds of software process improvement initiatives. Everything that is discussed is compliant with SPICE.

Using the RUP for SPI

Considering the key elements listed in the previous section—the CMM reference model, the method to manage the improvement program to closure (IDEAL), and the adaptive development process to increase competitiveness and meet the software global challenge—we now map the RUP concepts to CMM requirements and reflect upon the potential of the RUP in achieving particular software capability.

Software capability really means any one of the following for an organization:

- The right decision for the project succeeding at the right time
- Succeeding and surviving in business
- Taking risks and controlling the project output

- Getting the right quality of the product
- Integrating the most advanced software technologies in the products

Software capability is quite a sophisticated mix of these preceding statements, but there are some concepts that must be considered as part of the software capability:

- At the project level:

  Lifecycle issues: The project lifecycle is not limited to the implementation and maintenance of products. There is a management perspective that deals with the business, finance, and strategy of a project.

  Process selection and tailoring: The standard process of off-the-shelf and/or reflecting past experience cannot match objectives for a variety of projects. Efficiency results from defining an adequate development process for a particular project.

  Risks management: Any project faces risks. A capable organization anticipates the risks and makes the decisions to reconfigure the project to minimize the impact of risk occurrence.

  Measurements: One key to long-term progress in a software organization is collecting historical data to analyze software quality and productivity. Collecting some historical data for each project goes a long way. Data about effort, schedule, program size and functions implemented, and defect count form a solid basis for planning future projects and improving predictability. Predicting performance is one sign of maturity.

- At the organization level – seen as a collection of projects:

  Process improvement: Capability implies that an organization is able to learn from the past, especially from others' mistakes, and can translate these lessons learned into process evolutions. Improvement iteration cycles will only be complete if measurements are defined to quantify the improvement.

  Resources and skills: An organization's capability is tightly coupled with the capability of the people applying processes and developing products.

The RUP matches CMM requirements at the project level

The CMM describes the status of projects in organizations reaching a particular maturity level.

Projects in Level-2 organizations have installed basic software management controls. Realistic project commitments are based on the results observed on previous projects and on the requirements of the current project. The software project managers track software costs, schedules, and functionality; problems in meeting commitments are identified as they arise. Software requirements and the work products developed to satisfy them are baselined, and their integrity is controlled. Software project standards are defined, and the organization ensures they are faithfully followed. The software project works with its subcontractors, if any, to establish a strong customer-supplier relationship.

At the Defined Level (Level-3), the standard process for developing and maintaining software across the organization is documented. Projects tailor the organization's standard software process to develop their own defined software process, which accounts for the unique characteristics of the project. This tailored process is referred to in the CMM as the project's defined software process. A defined software process contains a coherent, integrated set of well-defined software engineering and management processes. A well-defined process can be characterized as including readiness criteria, inputs, standards and procedures for performing the work, verification mechanisms such as peer reviews, outputs, and completion criteria.

Because the software process is well defined, management has good insight into technical progress on all projects.

Lifecycle issues

In the CMM, the lifecycle issues are covered in more than one Key Process Area (KPA). When presenting CMM concepts for the first time in an organization, people generally misunderstand the term "software process" and assimilate it with "software lifecycle". The project lifecycle includes business and finance issues as much as development issues. Level-2 Key Process Areas and some Level-3 Key Process Areas are related to project lifecycle. They are:

- Requirements Management Goal 1: System requirements allocated to software are controlled to establish a baseline for software engineering and management use.
- Software Project Planning Goal 1: Software estimates are documented to plan and track the software project.
- Software Project Planning Goal 2: Software project activities and commitments are planned and documented.
- Software Project Tracking Goal 1: Actual results and performances are tracked against the software plans.
- Software Project Tracking Goal 2: Changes to software commitments are agreed to by the affected group and individuals.
- Software Configuration Management Goal 1: Selected software work products are identified, controlled, and available.
- Software Configuration Management Goal 2: Changes to identified software work products are controlled.
- Software Configuration Management Goal 3: Affected groups and individuals are informed of the status and content of software baselines.
- Software Product Engineering Goal 1: The software engineering tasks are defined, integrated, and consistently performed to produce the software.
- Software Product Engineering Goal 2: Software products are kept consistent with each other.

In other words, the CMM requires that:

1. A baseline is implicitly defined to consider the business environment of the application to be developed, to set priorities for the project requirements, and make all necessary decisions related to the functional and non-functional contents of the project.

2. Project plans are established and a list of activities, roles and responsibilities, milestones, and artifacts need to be set up that are realistic and reliable. Project plans take risks into account.

3. Project plans are monitored and the project team reacts in case of major deviations of feature contents or project environment/organization.

4. Any changes in the project are identified and analyzed. Further decisions are made and reported in the baseline and project plans.

5. Methods and tools are selected and applied to ensure the project's best performance.

The Rational Unified Process fulfills these requirements in the major tasks of the Inception phase as follows:

- Gets the project vision and develops a business model by matching item 1 in preceding list.
- Organizes and plans the project
- Estimates potential risks. (This item relates to item 2 in the preceding list.)
- Deploys the Unified Change Management policy, which is similar to items 3 and 4 in the preceding lists.

The overriding goal of the Inception phase is to achieve concurrence among all stakeholders on the lifecycle objectives for the project. The Inception phase is significant primarily for new development efforts, in which there are significant business and requirements risks that must be addressed before the project can proceed. For projects focused on enhancements to an existing system, the Inception phase is more brief, but is still focused on ensuring that the project is both worth doing and is possible.

At the end of the Inception phase, the project's lifecycle objectives are examined and a decision to either proceed with the project or to cancel it is reached.

Essential artifacts to review are:

- Project Vision
- Business Case
- Risk List (see the section on Risk Management)
- Software Development Plan
- Iteration Plan
- Development Case (see the section on Process selection and Tailoring)

These artifacts satisfy several goals of the CMM Key Process Areas listed at the beginning of this section.

Get the project vision and develop a business model

A business model defines the business use cases from the business workers' internal viewpoints. The model defines how people who work in the business and how the things they handle and use—"the classes and objects of the business"—should relate to one another, both statically and dynamically, to produce the expected results. The model also has an emphasis on roles performed in the business area, and their active responsibilities. Together, the objects of the model's classes need to be capable of performing all business use cases.

Based on the business model, the Rational Unified Process identifies an activity named Develop Vision, which has the following purposes:

- To gain agreement on what problems need to be solved.
- To identify stakeholders of the system.
- To define the boundaries of the system.
- To describe the primary features of the system.

Organize and plan the project

Just as the software process is influenced by the project's characteristics, so is the project organization. The default structure presented here, and illustrated in Figure 1, has to be adapted to reflect the effects of factors such as those listed here:

- Business Context
- Size of the Software Development Effort
- Degree of Novelty
- Type of Application
- Current Development Process
- Organizational Factors
- Technical and Managerial Complexity

These factors are taken into account in the RUP as process discriminants, which impact the choice of project structure.

The project structure is mainly defined through both:

1. The length of each iteration
2. The number of iterations

An iteration is a rather complete mini-project, going through all major workflows and resulting in most cases in an executable, yet incomplete, system referred to as a release.

To determine the number of iterations, many variations are possible depending on risks, size, and complexity.

If the product is intended for some totally new domain, you may need to add some iterations in the Inception phase to consolidate the concepts, show various mock-ups to a cross-section of customers or end-users, or build a solid response to a request for proposal.

If the product is large and complex, and developed over a long period of time, you need to plan on having three or more iterations in the Construction phase.

During the life of a project, the organization will evolve to support the work breakdown structure captured in the project plan. This is shown in Figure 1, [7].

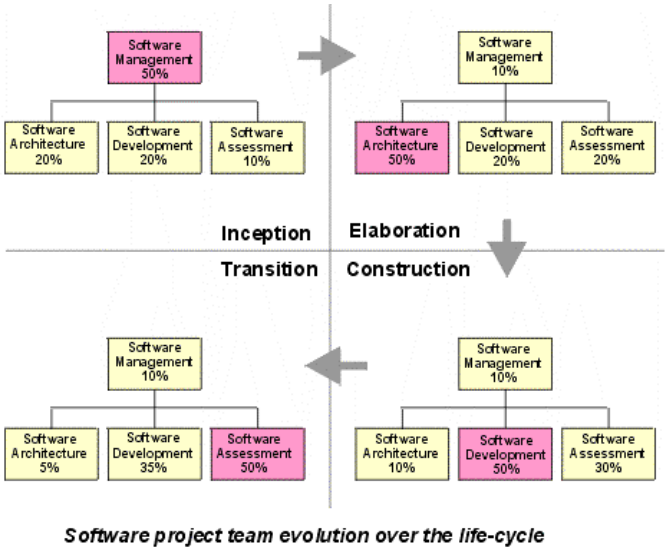

Figure 1: Software Project Team Evolution

This evolution emphasizes a different set of activities in each phase:

- Inception Team: an organization focused on planning, with enough support from other teams to ensure that the plans represent a consensus of all perspectives.
- Elaboration Team: an architecture-focused organization where the driving forces of the project reside in the software architecture team, who is supported by the software development and software assessment teams as necessary to achieve a stable architecture baseline.
- Construction Team: a balanced organization where most of the activity resides in the software development and software assessment teams.

- Transition Team: a customer-focused organization where usage feedback drives the deployment activities.

Migration between teams during this evolution ensures that knowledge and capability is retained in the project.

Deploy the Unified Change Management policy

Unified Change Management (UCM) is Rational Software's approach to managing change, from requirements to release, in software system development. UCM spans the development lifecycle, defining how to manage changing requirements, design models, documentation, components, test cases, and source code.

A key aspect of the UCM model is that it unifies the activities used to plan and track a project's progress and the artifacts undergoing change. The UCM model is realized by both the process and the tools. Rational ClearCase® and Rational ClearQuest® are foundation technologies for UCM. ClearCase manages all artifacts produced by a software project, which includes both system artifacts and project management artifacts. ClearQuest manages the project's tasks, defects, and requests for enhancements (generically referred to as activities), and provides the charting and reporting tools necessary to track a project's progress.

Process selection and tailoring

Maturity Level-3 includes three KPAs that target the overall organization and four KPAs that target the project organization, management, and engineering. The software process for both managing and engineering activities is documented, standardized, and integrated into an organization-wide software process. All projects use a documented and approved version of the organization's process for developing and maintaining software. Organization Process Focus (OPF), Organization Process Definition (OPD), and Training Program (TP) KPAs ensure that the organization identifies the project's best practices at Level-2 and documents them as a standard for the organization. They also focus on the skills management based on the needs identified by the projects, lessons learned from past projects, and the organization's mission and vision for the future. However, the maturity does not satisfy Level-3 requirements if there are no guidelines to adapt the standard process for a particular project to achieve success, manage the risks, and improve performance.

So, for most lead assessors and the CMM community, Level-3 is characterized by the Integrated Software Management (ISM) KPA, which has the purpose of integrating the software engineering and management activities into a coherent, defined software process. This defined process is the best suitable customization of the standard process that helps satisfy customer needs and constraints, market demand, and business strategy.

ISM is defined by two objectives in the CMM model. The first one involves tailoring whereas the second one requires project management. Activity 10 in the ISM: "the project's software risks are identified, assessed, documented, and managed according to a documented procedure" is the second key indicator for a Level-3 organization. Risk management is required at Level-2, but Level-3 organizations have succeeded in better anticipating risks and are able to demonstrate an entrepreneurial decision system. The Risk Management section below develops these aspects.

In the RUP, the key concept of Development Case and the Environment Workflow fulfill these requirements. A Development Case is a static configuration of the Rational Unified Process product; that is, it's a business process for software engineering tailored to a specific project, product, and organization. The Development Case focuses on what to do and how to do it, and also provides an overview of the process to be followed, which can be understood by everyone on the project.

The Rational Unified Process lists constituents of a process that are likely to be modified, customized, added, or suppressed in a given Development Case.

- **Core process workflows**
  A software project rarely skips one of the core process workflows—such as Analysis & Design, Implementation, and so on—completely

- **Artifacts**
  Projects are far more likely to differ by the artifacts that they have to produce, update, and deliver.

- **Activities**
  Activities may vary for at least two reasons: activities that use artifacts as input and produce, or update, artifacts as output are affected by the modification of these artifacts. In particular, if some artifact, or some element of information in an artifact, is no longer necessary, the corresponding steps may be suppressed or significantly modified. Activities are also modified to introduce specific techniques, methods, and tools that pertain to the specific application domain or development expertise such as design steps, programming languages, automatic code generation tools, measurement techniques, and so on.

The process engineer is responsible for configuring the process, deciding how the development process will look, "installing" the development case in the development organization (team, project or company), and teaching the developers how to use it.

Once a development case is set up by the process engineer, the project manager instantiates and executes it for the given project. This is often called process enactment.

As the process unfolds, lessons are learned during the process itself, which are then used by the process engineer as feedback to improve the process.

Risk management

Risk Management is widely covered in the CMM, primarily at Level-2 and more deeply at Level-3. Specific risk activities are listed to satisfy the goals of Key Process Areas Software Project Planning, Software Project Tracking and Oversight, and Integrated Software Management. CMM requirements follow the practices observed in many organizations: at Level-2, projects generally identify and assess risks on a regular basis. But there is a very low proactivity. At level 2, risk management is not yet well covered/understood, and there are problems with projects follow up and early identification of risks.

On the contrary, in organizations at Level-3, risks are identified, assessed, and mitigated, and teams demonstrate a rather good anticipation. On most advanced teams, quantitative measures are used to make decisions.

Our long experience in CMM-based appraisals and Software Process Improvement highlights the difficulties faced by software units in managing risks. This is one weakness still observed despite all of the effort dedicated to project control and measurement.

Using the RUP and the iterative approach is very helpful to succeeding in managing project risks. The entire Rational Unified Process is driven by Risk Management.

Risk drives the iteration plans; iterations are planned around addressing specific risks, attempting to either bound the risk or reduce it. The Risk List is periodically reviewed to evaluate the effectiveness of risk mitigation strategies, which in turn drive revisions to the project plan and subsequent iteration plans.

The key to managing risk is not to wait until a risk materializes (and becomes a problem or a failure) before deciding what to do about it. Just as a change of a few degrees in the path of a transcontinental flight has a large effect on where the plane lands, managing risk early is usually less costly and painful than cleaning up after the fact.

Four categories of risks are identified here:

1. Resource Risks
   - Organization
   - Funding
   - People
   - Time
2. Business Risks
3. Technical Risks
   - Scope risks
   - Technological risks
   - External dependency risk
4. Schedule Risks

The Risk Management Plan and the Risk List are two artifacts identified by the Project Management workflow of the RUP. They recommend that you identify and assess the risks.

Measurement

Measurement is one common feature in the CMM. A common feature exists for each Key Process Area from Level-2 to Level-5 and indicates when a practice is institutionalized. Metrics are so important that in the new version of the SW-CMM—that is, the one embedded into the CMMI framework, the Integrated CMM System engineering + Software engineering—there is a measurement-based goal for each Key Process Area.

What is key in the CMM is to measure processes to determine their adequacy at Level-2 and their effectiveness at Level-3. The shift from reporting to analyzing and acting on metrics is generally difficult, but a clear sign that the projects are more mature. Small and early successes managing a project quantitatively lead to accepting and understanding the benefits of measurement.

In addition to these general requirements, some KPAs have specific requirements for measures such as Software Project Planning, Software Project Tracking and Oversight, and Integrated Software Management. These are related to project data that enable project estimates and project control.

The RUP provides guidance for using metrics. The Measurement Plan and the Measurement artifacts have to be produced in the Inception phase.

The Project Measurements artifact stores the project's metrics data. It is kept current as measurements are made or become available. It also contains the derived metrics calculated from the primitive data and should also store information, such as procedures and algorithms, about how the derived metrics were obtained. Reports on the status of the project—for example, progress towards goals (functionality, quality, and so on), expenditures, and other resource consumption—are produced using the project measurements. Frequent, or even apparently continuous, displays of project status are possible using the Rational Project Dashboard approach, for example, where automated software data collection agents feed real-time displays of project status.

The Rational documentation edits an initial set of metrics, which is simple enough to start with. Metrics for certain aspects of the project include:

- Progress of size and complexity
- Stability of rate of change in the requirements or implementation, size, or complexity
- Modularity of the scope of change
- Quality of the number and type of errors
- Maturity of the frequency of errors
- Resources of project expenditure versus planned expenditure

The RUP matches CMM requirements at the organization level

The software process capability of Level-2 organizations can be summarized as disciplined because planning and tracking of the software project is stable and earlier successes can be repeated. The project's process is under the effective control of a project management system, following realistic plans based on the performance of previous projects.

At the Defined Level (Level-3), the standard process for developing and maintaining software across the organization is documented, including both software engineering and management processes, and these processes are integrated into a coherent standard. The standard process is referred to throughout the CMM as the Organization's Standard Software Process. Processes established at Level-3 are used, and changed, as appropriate; to help the software managers and technical staff perform more effectively. The organization exploits effective software engineering practices when standardizing its software processes. There is a group that is responsible for the organization's software process activities; for example, a software engineering or SEPG. An organization-wide training program is implemented to ensure that the staff and managers have the knowledge and skills required to fulfill their assigned roles.

The software process capability of Level-3 organizations can be summarized as standard and consistent because both software engineering and management activities are stable and repeatable. Within established product lines cost, schedule, and functionality are under control, and software quality is tracked. This process capability is based on a common, organization-wide understanding of the activities, roles, and responsibilities in a defined software process.

Process improvement

Capability implies that an organization is able to learn from the past, especially from mistakes, learn from others, and translate lessons learned into process evolutions. Improvement iteration cycles will only be complete if measurements are defined to quantify the improvement.

When maturity of the organization improves, the standard process changes. Getting from Level-2 to Level-3 implies that all good practices within projects are institutionalized and that there is an assessment process in place that will help to identify the best practices across projects, which will be documented into the Organization Standard Software Process (OSSP). These are the requirements of the Organization Process Focus (OPF) Key Process Area. The OSSP will then be refined on the basis of lessons learned within projects across the organization. Because projects use the same standard, the repository of experiences and lessons learned is easily enriched, and the OSSP itself can benefit from the various experiences.

The RUP Environment workflow develops a similar approach. The "Implementing a process in an Organization" concept explains what you do at the organizational level to implement process and tools in a development organization.

Implementing a new process in a software-development organization can be described using the four phases of the RUP: Inception, Elaboration, Construction and Transition.

The steps to implement process and tools in an organization

The OSSP of the CMM is a new process in the organization and its definition can follow the four phases of the RUP.

The decision whether to develop an organization-wide development environment that each software development project can use with the necessary adaptations is key, but a certain maturity level must exist.

If you decide to develop an organization-wide environment, you must initiate a project to develop the organization's development environment. And if you decide to initiate such a project, it must be made clear that this project team will work very closely with the software development project teams. The RUP also recommends that you consider it as a specific project. This fulfills CMM requirements here again.

The process-implementation project is divided into a number of phases where all four steps are performed in each phase until the project is ready and the process and tools are deployed and successfully used by the entire organization.

A process-implementation project can be divided into phases

The four phases address:

- Phase 1: Sell the process-implementation project to the sponsors.
- Phase 2: Handle the major risks.
- Phase 3: Complete everything—templates, guidelines, examples of Development Cases are ready, and a training curriculum is in place.
- Phase 4: Deploy it to the entire organization.

These phases can be named Inception, Elaboration, Construction, and Transition respectively, as they are for a software development project using the Rational Unified Process.

The same project phases will be defined each time the standard process has to evolve in order to increase its performance based on lessons learned in a particular project or according to a particular technology.

The RUP also defines the Managing Organizational Change concept, which is the overall background of Software Process Improvement. The recommendations for successfully implementing a process change are to:

- Identify change agents at various levels in the organizations.
- Plan the change in small, reasonable, and measurable steps.
- Communicate the changes using ground level language appropriate to the level of the organization.

These recommendations are similar to those given by the IDEAL approach, from which they are inspired.

Finally, the RUP defines a specific actor called “Mentor”. A mentor is someone who teaches and guides the project teams about what they need and when they need it. Typical ways of mentoring are:

- Workshop leader
  Some activities are best performed in a group; for example, finding actors and use cases during use-case modeling. Throughout such activities, it's valuable to have a modeling leader who is a process expert.

- Process expert
  The process expert is an on-site support person for the project. The process expert's task is to help the developers use the process and model as well as possible.

- Project manager support
  A process expert can help the project manager to plan and steer the project. Sometimes the project manager has little or no experience about the process in question.

- Reviewer
  A cost-effective way to transfer knowledge is to have a process expert review the results of each phase. A process reviewer also brings value in reviewing any process adaptation performed by one project.

Does this Mentor fulfill a different role than the SEPG (Software Engineering Process Group) in the CMM? No, it is the same type of role.

Resources and skills

At the organization level, the CMM identifies a "Training Program" Key Process Area, which could be better named as "skills management" as that's the real concern here. A mature organization has to identify and plan the needs for skills in the mid to long term, and manage them to get them in due time.

The purpose of a Training Program is to develop the skills and knowledge of individuals so they can perform their roles effectively and efficiently. Training is an organizational responsibility, but the software projects should identify the necessary skills and provide the necessary training when the project's needs are unique. The management workflow in the RUP considers these issues when staffing a project.

The Training Program Key Process Area has three goals:

- Goal 1: Training activities are planned.
- Goal 2: Training is provided for developing the skills and knowledge necessary to perform software management and technical roles.
- Goal 3: Individuals in the software engineering group and software-related groups receive the training necessary to perform their roles.

Several aspects of the RUP fulfill these requirements. Roles are precisely identified and their competencies are defined. A role is typically realized by an individual or a small set of individuals working together as a team. A project team member typically fulfills many different roles; just as a person can wear many 'hats', a person may perform many different roles.

Roles are not individuals; instead, they describe how individuals should behave in the business and what responsibilities these individuals should have.

Although most roles are realized by people within the organization, people outside of the development organization also play an important role; for example, that of the stakeholder of the project or product being developed.

Each type of role is carefully defined by skills and knowledge a team member should demonstrate. Based on these definitions, an organization is able to derive the missing skills and the characteristics of the training necessary to get this role performed efficiently.

The RUP guidelines form a good set of rules to be provided as training. The course developer is one of the role categories listed.

The only aspect that is not entirely covered by the RUP and is required by the CMM is identifying, planning, and delivering training to develop new skills for the future. Imagine that the strategy of the organization is to center its activity on e-commerce one year from now; this objective has to be further decomposed and will probably impact the type of developments performed as well as the type of environments used. Teams have to be prepared for this change and the necessary training must be defined, developed, or acquired externally. The Training Key Process Area at Level-3 requires that these issues are carefully managed, tracked, and recorded.

- Section three of this paper gives a detailed mapping of some of the maturity profiles with the RUP features. This paper could have made a systematic parallel between each goal of the CMM reference and the corresponding guidelines of the RUP; we deliberately did not do that and instead chose to focus on the key signs of high maturity.

- Software quality is addressed within the RUP through the test activity on the one hand and through the Develop Quality Assurance Plan Workflow on the other. Our demonstration, however, highlights our experience and observation of what high maturity really means: what is different in Level- 3 organizations compared to Level-1 or low Level-2 organizations.

- The next section is dedicated to some feedback of early adopters of these combined technologies.

Pragmatics: how to get started with the RUP

The most common question we are asked is: Has anyone done that before?

This question will probably be asked more than once when the previous concepts are presented to managers and practitioners. Getting more mature, more process-oriented, managing risks, and being successful is attractive to any software business. However, is it really possible in a moving context, with requirements that are changing quickly and teams who need to apply technologies on which they are not trained?

Rational customer stories include the experience of Computing Devices International: Computing Devices is a leading defense electronics and information company. Information continues to be a key competitive advantage in almost every industry. For Computing Devices International, it is not just information—their mission is to provide end-to-end solutions that deliver mission-critical information on demand anytime, anywhere.

They need to ensure consistent, on-schedule delivery of high quality solutions. The key benefits of their experience include the following:

- An acceleration of the development process through an iterative design approach
- A decrease in development time from 3 years to 19 months
- The ability to ensure reliable, on-time delivery
- An increase in customer satisfaction
- A 33% reduction of development costs

The sophisticated systems that Computing Devices develops for its customers are highly dependent on software that has to be designed to stringent requirements. Computing Devices first turned its attention to the process of writing safety critical software.

Different business units within the organization were using a variety of tools, based on structured design or "waterfall" methods. Consequently, the development process was slow and lacked a consistent approach. This led to slipped deadlines, higher costs, and software quality that did not always meet their high standards. Computing Devices realized they needed to take action—fast.

Skandia-IT is another example from Rational’s portfolio. In that organization, nine large insurance systems were delivered either on time or ahead of schedule, in a single 12-month period.

They accelerated the recruitment of new talent by attracting developers and consultants with a state-of-the-art development process, ensured customer satisfaction by employing use cases to solicit requirements, model systems, and direct development, and quickly developed a flexible, 3-tier architecture by wrapping legacy systems and using them as components. These were some of the challenges Skandia-IT conquered.

Managers at Skandia-IT explained, "There were several reasons for why we changed the whole process. The most important was that we reorganized our entire business. We changed the development process in parallel to the change from a product-oriented organization to a customer-oriented organization. Our new systems give the customer the possibility to handle a lot of things themselves, with telephone or via the World-Wide Web. Our employees at Skandia can focus more time on becoming insurance specialists rather than learning how to use the system. Time saved is invested in customer relations."

Skandia-IT has invested heavily in the installation of the new process. It's an investment that has paid off. Better quality, predictable delivery times, and side effects like the fact that today it’s easier to recruit experienced project leaders and developers.

Other examples of companies focusing on processes and starting with the RUP to accelerate the deployment exist. Q-Labs is working with Lysis, a fast growing company that develops software for Internet broadcasting. In a start-up environment, process technologies and methodologies that focus on advanced levels of process maturity can be out of place. This type of organization is primarily fast-paced, reactive, and innovative, and the contrast between the need for speed in getting products out and the need for maximizing process control raises the question: to what extent does the theme of process accommodate this diversity of viewpoints?

The characteristics of this company are:

- They are a fast growing software unit.
- They develop faster (less than 6 months).
- They deliver better products using leading object technologies.
- They adapt to dynamic markets.

This environment is not really different from other software industries except that the time scale is completely different, predictability is equivalent to survival or death and teamwork has to change radically very fast. Time for learning is almost nonexistent and the best option is to start from existing, well-proven practices. This is exactly what Lysis did when they adopted both CMM to guide their growth and the RUP as a background to their install processes.

The need to define software practices is not questioned. It serves understanding, communication, execution and management. It also forms a basis for formulating relationships with other organizations, both internally and externally. In the case of Lysis, relationships with marketing and sales and with potential software subcontractors are the targets. The defined processes have to remain flexible: changes may occur rapidly in personnel, infrastructure, release schedules, and adopted technologies. The development processes needs to accommodate them. A rapid and timely response is required. Iterative processes with short iteration cycles, risks-oriented gates, and constructive management are recommended. The ability to repeat a process across components of product lines is necessary. Process definitions help inform new personnel who join the company as it grows.

In the CMM, process definition (maturity Level-3) follows process repeatability (maturity Level-2), but in such a company starting from a standard process like RUP is helpful to promote and establish repeatability, thereby bringing significant benefits.

The approach taken by Lysis started with an appraisal of the development practices against CMM requirements. This resulted into a good buy-in of process orientation by the teams who were most interested by programming, object technologies, and tools environment. Based on the observations, an action plan to define processes was established with strong time constraints. Business goals, set by the top management to support SPI, include:

- Shorten the development process in order to be the first on targeted markets
- Speed-up transition from project mode to product mode
- Manage company growth
- Satisfy people
- Keep start-up reactivity
- Improve predictability and visibility (for IPO, necessary quarterly results)
- Quantitative improvement of the products quality
- Manage cost and potential profits
- Optimize customer satisfaction
- Identify and manage interface with partners (for sub-contracting part of the work)

The SPI program itself has been established as a project using the RUP framework where:

- Inception = Assessment

- Elaboration = Project planning + Web-based repository definition and prototyping
- Construction = Processes defined + Implementation and tooling
- Transition = Pilot project + deployment

Work packages are defined to address CMM Key Process Areas. Each of them has objectives, activities to be performed, deliverables, and a review process. As an example, objectives for Software Planning are:

Objective of the action:

- Document the current good practice of software planning
- Define the generic organization set for projects
- Define roles and responsibilities for planning
- Define the commitment process for project level
- Define a generic project lifecycle and tailoring guidelines for a standard set of project processes
- Define a standard Work Breakdown Structure (WBS)
- List standard list of deliverables
- Define template and guidelines for Development Plan
- Write the project planning process
- Define skills and training necessary for planning

Choosing the RUP to start with will speed up some of these actions and will ensure that an organization benefits from a robust experience. The SPI initiative is not completed, but some visible success factors have already been observed: development of new products is better structured; plans exist and are regularly revisited. Corrective actions are taken and the visibility outside of the software team has greatly improved.

Rational Software reports about a similar case with E-corporation. The E-corporation is a European Interactive Architect, which is guiding companies all over the world to the digital business frontier. The E-corporation combines strategic, artistic, and software skills to conceive, create, and implement Internet and e-business ventures.

Key benefits of E-corporation deploying the RUP are:

- Better communication is provided between team members, which increases their efficiency.
- Customer's expectations are more accurately met.
- A framework for meeting high-quality and high-speed demands is provided.
- Risk with iterative development is reduced and a proven process based on expert knowledge and industry-acknowledged best practices is used.

The greatest advantage The E-corporation has experienced from working with one common process on all projects is the effective reuse of components, experiences, and activities. The Rational Unified Process can constantly be fine-tuned when the organization gains experience in various projects. Because all activities are documented following a consistent structure and language, much experience can be taken from previous projects to new ones.

The E-corporation plans to roll out the Rational Unified Process to all of its projects at all sites to further leverage the unification of its teams. The Rational Unified Process gives their development teams the power to apply proven principles to optimize all efforts, simplify communication, and assure quality solutions that meet all requirements.

These are examples of how various organizations have observed benefits by coupling the Rational Unified Process and the CMM guidelines. We are currently working with other organizations where the RUP is considered the generic structure that will be populated with the organization's process asset where it fits.

Conclusion

What we have described about mapping the RUP and CMM is equally applicable for ISO 15504. The concepts underlying these two references are similar and the key factors we have highlighted to explain a maturity profile support the capability scale for ISO 15504.

This paper has clarified the extent to which the RUP concepts match the goals of a highly mature organization. Therefore, adopting the RUP is a way to successfully fulfill CMM requirements.

A recent article published by Telcordia Technologies gives some quantitative results of high maturity on the business. Telcordia, recently assessed at maturity Level-5, reports that:

- Reduction in field fault density has reached 94% since the start of its SPI initiative.
- More than 98% of software releases have been delivered on time since 1995.
- The cost of testing a line of code has been reduced by 64%.

Telcordia's quality journey took six years to reach this maturity level.

For software intensive companies in the e-business and e-commerce fields, time to deliver cannot afford a five or six year effort to optimize processes. Therefore, the only possible scenario to reach the maturity needed to remain competitive is to:

- Understand the value of software process
- Adopt an existing framework such as the RUP
- Adapt that framework to the characteristics of the market environment
- Measure its impact on business
- Learn and enrich the customized process model

Because the RUP model is generic, it's important you don't get lost. Therefore, Rational identifies the "Essential RUP", which includes those aspects to focus on when adopting the model [11, 12]. The default RUP can be used, but everything does not have to be implemented the first time. The important features are:

- Business case
- Schedule
- Vision document
- Risks
- Architecture
- Change requests and how to handle defects
- Tests
- The software product itself
- User support documents
- Project assessments

Executing the RUP is not a goal in itself; it helps solve painful points in groups and on teams, and helps meet competitive objectives.

Authors' contact information

Annie Kuntzmann-Combelles, Executive VP
Q-Labs France
28 Villa Baudran 94742 Arcueil cedex
tel +33 (0)1 49 08 58 00
akc@objectif.fr

Philippe Kruchten, Rational Fellow
Rational Software Canada
pbk@rational.com

References

[1] K. Pulford, A. Kuntzmann-Combelles, and S. Shirlaw, 1995. *A Quantitative Approach to Software Management—The AMI Handbook*. Addison Wesley Longman.
[2] Barry W. Boehm, 1996, "Anchoring the Software Process," *IEEE Software*, July 1996, pp.73-82.
[3] Philippe Kruchten, 1996. "A Rational Development Process," *CrossTalk*, 9 (7), July 1996, p.11-16.
[4] Robert McFeeley, 1996. *IDEAL: A User's Guide for Software Process Improvement*. Software Engineering Institute, Pittsburgh, PA, CMU/SEI-96-HB-001.
[5] Steve McConnell, 1997. *Software Project Survival Guide*. Redmond, WA: Microsoft Press.
[6] Mark Paulk, et al. 1993. *Capability Maturity Model for Software, Version 1.1*. Software Engineering Institute, Pittsburgh, PA SEI-93-TR-024.
[7] Walker Royce, 1998. *Software Project Management: A Unified Framework*. Addison Wesley Longman.
[8] Jas Madhur et al, 1998. *Reaching CMM Levels 2 and 3 with the Rational Unified Process* white paper, Rational Software.
[9] Alec Dorling et al., 1999. *SPICE, the Theory & Practice of Software Process Improvement*, IEEE Computer Society.
[10] Philippe Kruchten, 2000. *The Rational Unified Process—An Introduction, 2nd ed.* , Addison Wesley Longman.
[11] *Rational Unified Process, version 2000.02.10,* Rational Software Corporation
[12] Leslee Probasco, 2000. "The Ten Essentials of RUP", *The Rational Edge*, December 2000, http://www.therationaledge.com.

Dual Headquarters:
Rational Software
18880 Homestead Road
Cupertino, CA 95014
Tel: (408) 863-9900

Rational Software
20 Maguire Road
Lexington, MA 02421
Tel: (781) 676-2400

Toll-free: (800) 728-1212
E-mail: info@rational.com
Web: www.rational.com
International Locations: www.rational.com/worldwide