



Build your Project using Rational Unified Process #7 of a Series, by Pavan Kumar Gorakavi, M.S., M.B.A, G.M.C.P, C.A.P.M.

1. What is Rational Unified Process?

Rational Unified Process is a disciplined software development methodology that targets producing high quality software deliverables. The Rational unified process encourages systematic development while classifying tasks, assigning tasks and responsibilities within a development organization. Rational unified process was developed by Philippe Kruchten, Ivar Jacobsen, and others at Rational Corporation.

The setting for development of Rational Unified Process began in early 90s at Rational Software Corporation, founded by Paul Levy and Mike Devlin. Earlier Rational Software Corporation targets to develop large defense software development projects in Ada. In the 80s, to withstand competition from different open-system platforms Rational Software Corporation decided in porting their system to open-system platforms. To improve market strength, Rational Software Corporation acquired Pure-Atria which provide ClearCase configuration management tools, the Purify line of testing tools, Requisite for Requisite Pro, SQA for testing tools suites. Rational worked with "the three Amigos" [Grady Booch, James Rumbaugh, Ivar Jackobson] to develop a single unified language, UML, and drafted best software development practices which yielded the Rational Unified Process.

Rational Unified Process is a successor to the Rational Objectory Process. After merging of Rational Software Corporation and Objectory AB in 1995, Rational Objectory Process was the result of integration of the Rational Approach and Objectory Process. Rational Unified Process inherits its process structure and use cases structure from the Objectory Process. Rational Unified Process inherits its iterative development and architecture from the Rational Approach.

There are controversies about whether Rational Unified Process is agile or not. If the Rational Unified Process is tailored, we can affirm RUP as an agile practice. The following salient features are candidates for tailoring, to use RUP as an agile process:

- Choose only the most vital artifacts needed by the customer and the development team.
- Strive to keep iterations as short as possible.
- Establish Business owners as part of the team
- Complete Incremental short releases
- Assure Collaborative Efforts and Communicative flow.
- Maintain the modularity of the development model
- Remain flexible and Adaptive with emergent risks

2. Principles of the Rational Unified Process

The Rational Unified Process provides six guidelines to implement successful projects. These six best practices were developed from Rational's experience while developing large and complex systems. They were also designed to help drive the use of tools offered in Rational's product line. The designers of the RUP continue to evolve the process as methods and practices mature through their application [1]. The six basic principles are outlined below in figure 1.

The Rational Unified Process (RUP) supports an iterative approach to development that helps identify risk proactively, reduces re-factoring cost, and builds models with an easy exit strategy. Rational Unified Process recommends using use-cases and scenarios to capture functional requirements. The Rational Unified Process supports component-based software development. Components are non-trivial modules, subsystems that fulfill a clear function. The Rational Unified Process provides a systematic approach to defining an architecture using new and existing components [2].

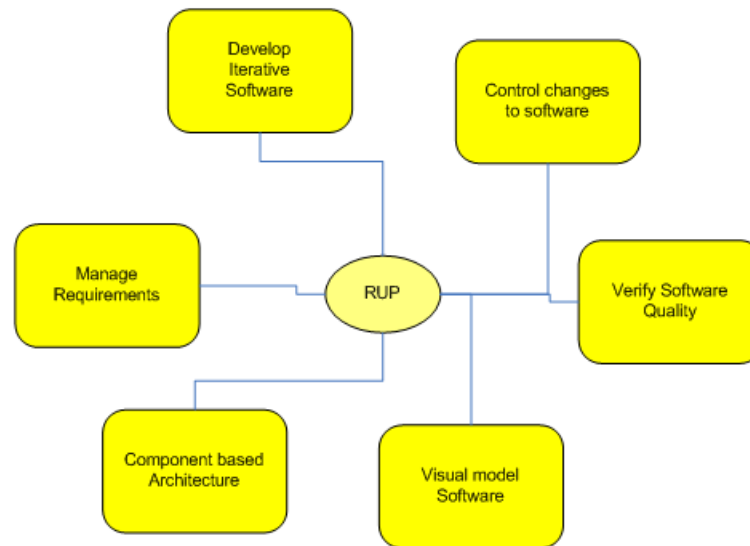


Figure 1: Principles of Rational Unified Process

RUP encourages visual software models to depict architectures and component. Frequent verification of quality should be reviewed with respect to the requirements based on reliability, functionality, application performance and system performance. Control changes to software are recommended by the process. The process describes how to control, track and monitor changes to enable iterative development.

Per Kroll and Walker Royce updated the six best practices in IBM Rational e-zine , October 2005, as follows: [3]

1. Adapt the process.
2. Balance competing stakeholder priorities.
3. Collaborate across teams.
4. Demonstrate value iteratively.
5. Elevate the level of abstraction.
6. Focus continually on quality.

Every project differs in RUP's appeal. Complexity, size, dependent variables, and distributed versus co-located team are some of the factors that impact the project. Identifying dependent factors helps in adjusting the process to fit it. Project size must be optimal and must be decided based on the requirement. Process must be introduced slowly in initial part of the project and it must be implemented intensely at later parts of the project. Periodic reviews in RUP helps to identify risk proactively and continuously improve the process. Appropriation of process strength is dependent on project size, distributed teams, complexity, stakeholders and other dependent factors. More formal control is required when a project meets the following criteria:

- Project members are distributed in different places
- User community is large and distributed.
- Project is Large or Very Large.
- Many stakeholders.

Defining and understanding business needs is an important aspect of implementing RUP. Key business players must identify the business needs, and try to prioritize business and stakeholder needs. Per Kroll and Walker Royce argue strongly about *centering development activities around stakeholder needs*. Performing value analysis on leverage of assets, balancing user needs and reusing the assets are some of their suggested best practices. Collaboration across the team is required to build a high efficient team. Per Kroll and Walker Royce mention that effective collaboration can be achieved by:

- Motivating individuals on the team.
- Encouraging cross functional collaboration.
- Integrated collaboration across business, software and operation team.
- Providing feasible environment for effective collaboration.

Iterative value-based-development is one of the best practices. Delivering incremental value helps obtain early and continuous feedback. This early feedback helps reduce re-factoring cost. Per Kroll and Walker Royce argue to elevate the level of abstraction by reusing existing assets. Reusability can be impacted by system complexity, loose coupling, and cryptic architectures. Reusability can be maximized by having modular design, and simple architecture.

3. Life Cycle of Rational Unified Process

According to Kruchten, RUP consists of four sequential processes or phases during design and development of a business solution. The four sequential processes includes: Inception, Elaboration, Construction and Transition. These phases are iterative in nature and yield products in each field. Iterations are between two weeks to six months. The RUP lifecycle is illustrated in figure 1.

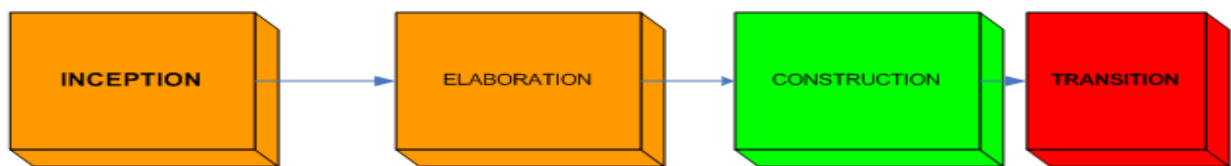


Figure 2: Life Cycle of RUP

Phase 1: Inception

This phase prepares overall domain descriptions, and organizes documented requirements with proper use-cases and technical specifications. During this phase the business case for the system is established and the project scope delimited. All key entries are identified and communication is identified at higher levels. Identifying use cases is an important step in this phase. The Business Case includes success criteria, risk analysis, resource requirements, and game plan to be implemented.

The Inception phase yields use-cases, vision documents, high level business case and glossary, higher level risk analysis, project plan, and prototype. Evaluation is performed iteratively in each phase. Stakeholders conduct regular meetings to discuss scope definition and cost estimates. Conflict of interests and initial risk assessments are resolved while stakeholders are in process of working with the team. Cost estimates, Risk factors, task estimations, resource estimations and schedule estimates are evaluated in these meetings. Prototypes developed in phase 1 are evaluated for their depth and breadth. Relative analysis of actual expenditure vs. planned expenditure is performed as part of evaluation criteria. The Project can be rethought, cancelled, or approved to proceed in this phase.

Phase 2: Elaboration Phase

This phase plays a vital role in laying a foundation for the project's architecture. This phase helps analyze problem domains, develop a flexible project plan, develop a flexible architectural foundation, and identifies risks and eliminating them. The Architectural foundation can be developed from the results of phase 1, i.e., scope definition, requirements, and major high level functionality. This phase is helpful in identifying risk analysis, and value analysis and the decision can be made whether or not to commit to further phases: construction and transition. Executable architecture is built in iterations depending on project size, scope, cost, and time lines.

After this phase , most use cases are defined, actors are identified, Software architecture is described, and a prototype is created. Evaluation is performed on the product stability, architecture stability, relative analysis of actual resource expenditure vs. planned expenditure, and risk factors identified. The project may be aborted or approved to proceed, based on the result of milestones achieved in this phase.

Phase 3: Construction Phase

In the Construction phase all remaining components and application features are developed and integrated in the product. All developed features are thoroughly tested. The Construction phase is a manufacturing phase where focus should be on managing resources, controlling operations, optimizing cost, meeting the timeline, and maintaining acceptable quality. A successful Elaboration phase yields a successful Construction phase. A robust architecture developed in Elaboration helps as a skeleton for the Construction phase.

There are scenarios where parallel construction increments can be spawned. The Construction phase results in software products integrated on an adequate platform, User manuals, release notes, and other proprietary documentation. The results of the Construction phase: either alpha, beta, or final, are developed iteratively and rapidly which helps develop a high quality product with a rapid pace. The Construction phase is evaluated considering the following salient features:

- Stability of the product released.
- Maturity of the product developed.
- Stakeholder acceptance.
- Relative analysis between actual resource expenditure vs. planned expenditure.
- Can this be deployed in user community?
- Does user documentation stand as per industry standards?
- Release notes have up-to-date information.
- Legal liability resolved for the finished products.

Phase 4: Transition Phase

This phase transitions the software deliverables to the user community. This phase begins when the product reaches suitable maturity and stability. As the product development is iterative in nature, defects are resolved in subsequent releases. This phase primarily focuses on activities required to release stable products, product tuning, product configuring and resolving issues. The primary objective of the Transition phase includes achieving baseline targets, establishing user confidence and self supportability, and concurrence with vision documents.

The Transition phase is evaluated based on following salient features.

- User satisfaction.
- Relative analysis between actual resource expenditure vs. planned expenditure.
- Self-supportability of the product.
- Value analysis based on cost effectiveness, and product baselines.
- Acceptance based on deployment baselines.

4. Different Actors of RUP

The RUP process can be classified into three main components: Activities, Artifacts, and Workers. An **Activity** is a unit of work that an individual is asked to perform in his or her role. It can also be described as a granular, modular functionality which a role expects. An activity can vary from few hours to few days. Finding actors, developing modular functions, writing use cases are some of the examples of Activities.

An **Artifact** is an information center that is developed, modified or used by a process. Activities are derived from artifacts. Artifact can be in various shapes. It can be in a use case model or a design model or business case or software architecture document, or executables, or source code. A workflow is a meaningful sequence of activities that produce a valuable result. Workflows are generally expressed as sequence diagrams, collaboration diagrams, or activity diagrams.

Worker defines responsibilities of an individual, a person's role with a description or title. A set of sample Worker roles is in figure 3, below.

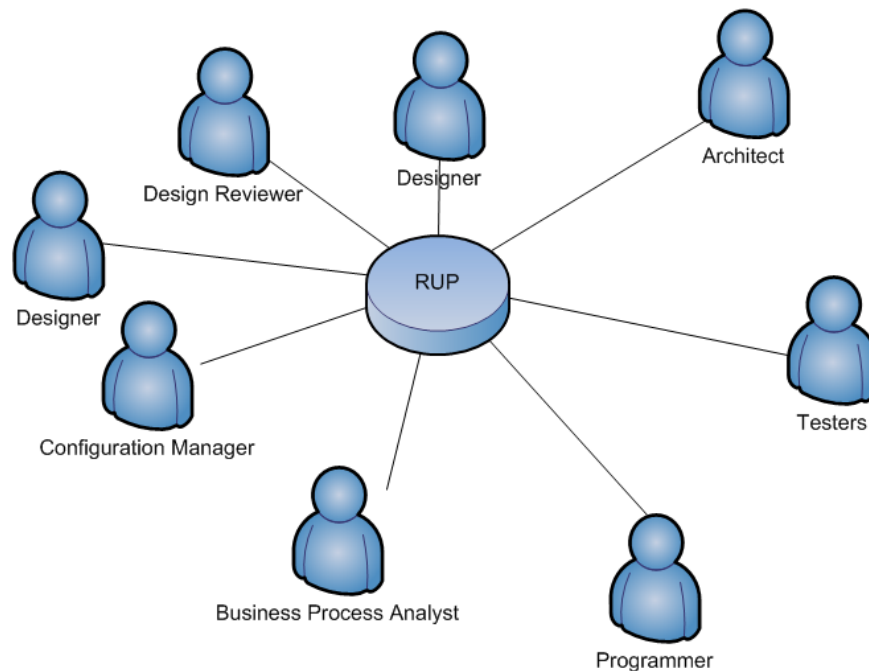


Figure 3: RUP stakeholders

RUP is an activity-driven development method. Roles are basically defined based on the activities defined in the process. As the diagram shows above, RUP has workers like Developers, Architects, Designers, Design Reviewers, Configuration Managers, Business Process Analyst, Business Reviewers, Business Designers, Course Developers, Toolsmith, Testers, Project Managers, and other Stakeholders.

5. Conclusion

According to Kruchten (2000), Rational Unified Process can be adopted as a whole or in parts. RUP has been successfully implemented across many organizations and it is well supported by Rational tools. RUP can be tailored to make the process agile. RUP's process-based approach helps to develop a robust system.

About the Author

Pavan Kumar Gorakavi is working as a Senior Software Developer in Dallas, TX. He is settled in Dallas, TX with his family (wife Swapna Gorakavi and son Anish Gorakavi). He is VP - programs for *asapm* Young Crew. He is also acting as Associate Director [Marketing] for PMI-ISSIG.

Pavan earned his Bachelor's degree in computer science from Jawaharlal Nehru Technological University and Masters in computer science from Lamar University. He did his MBA from University of Texas at Dallas and GMCP from Southern Methodist University. Pavan holds SUN, IBM and PM Institute certifications.

Pavan Gorakavi authored a book on 'Artificial Intelligence' published by Rahul publications - India, and 'Digital Electronics' published by Subhash publications, India. His research interests are Artificial Intelligence, Agile methodologies, and Software development in ADA, Prolog and Java. You can reach Pavan at gorakavi@gmail.com.

