Chapter IV

RUP: A Process Model for Working with UML?

Wolfgang Hesse
Philipps-University Marburg, Germany

ABSTRACT
Recently, the Rational Unified Process (RUP) has been published as the second part of Rational’s Unified Method project. The RUP is advertised as an “iterative and incremental, use case-driven, architecture-centric” process model and aims to support system designers, builders and managers working with the Unified Modeling Language (UML) by a procedural guideline.

In this chapter, a brief review and a critical analysis of the RUP is attempted. Its general aim and its contribution towards more harmonisation in the software process field are acknowledged. However, its ability to reduce the complexity of software development and to clarify its interlaced structure and terminology is doubted. Major problems may result from concepts not clearly specified like workflow or architecture. In particular, RUP core concepts like phase, iteration, workflow and milestone are debated. It is argued that RUP phases and milestones do not support the requirements of modern object-oriented (and, in particular, component-based) software projects. Iteration cycles should be based on software building blocks rather than on phases and activities. As one possible alternative to the RUP, a component-based (and truly architecture-centric) process model is sketched, and a multi-variant approach to software process modelling is recommended.

INTRODUCTION: THE “UNIFIED PROCESS,” ITS HISTORY AND AIMS
In the mid-'90s Rational company has started a project trying to merge some existing methodologies for object-oriented analysis and design into a common “Unified Method.” For this purpose their chief methodologist G. Booch, joined by J. Rumbaugh and (later) by I. Jacobson, tried to combine their methods which became popular at that time. Realizing that this goal was not to be achieved, within one single step the authors reduced their ambitions and started with a common metamodel and notation, an approach which resulted
in the “Unified Modeling Language” (UML) disseminated since 1997 and now available in its version 1.3 from June 1999 (UML 1999).

In early 1999, the UML which deliberately had left aside any process aspects was complemented by the so-called “Rational Unified Process” (RUP), now documented by two books (Kruchten, 1999; Jacobson et al., 1999), in the Web (RUP, 1999) and by further presentations of its authors. With this generalised process description, the authors claim to “enhance team productivity” and to “give project managers control over schedules and deliverables.”

The RUP has various sources like Boehm’s spiral model (Boehm, 1988), Booch’s macro and micro process approach (Booch 1994) or Rumbaugh’s OMT (Rumbaugh et al., 1991). However, there is no doubt that its most important source is Ivar Jacobson’s Objectory process (Jacobson, 1993). Some of the outstanding RUP features, like the focus on use cases and on iterations or the intertwined phase and workflow structures, have been very much influenced by this predecessor process.

According to its own advertisements, the RUP has been designed with the following aims (Jacobson et al., 1999):

- Provide guidance to the order of a team’s activities.
- Direct the tasks of individual developers and the team as a whole.
- Specify what artifacts should be developed.
- Offer criteria for monitoring and measuring a project’s products and activities.

Of course, these are aims nobody would object to, at least when a process definition for a specific project is required. For a reference process model like the RUP, there are surely broader goals as, for example:

- Provide a general structure and terminology covering the ingredients of software processes like its phases, artifacts, roles of people concerned, milestones, etc.
- Define the purpose, properties and relationships of these ingredients and establish a consistent and ready-to-use terminology.
- Provide a framework of common understanding for people working at different projects to support their cooperation, exchange of experiences, comparison of results, qualification of staff, etc.

There is no doubt that the RUP, although not explicitly claimed by its authors, stands for the first serious attempt to set up a standard for such a reference model in the “object-oriented world.”

- It has set up definitions and given examples for central concepts of process technology such as phases, activities, artefacts, workflows, actors and roles.
- It is based on the experience of various software development teams that have worked in large and diversified projects.
- It is clearly oriented towards and closely connected with UML as a modeling language, and it supports working with this language and its tools.

Regarding the so far rather chaotic landscape of software life-cycles and process models, such an attempt has to be acknowledged. Surely it will influence the way of carrying out and managing software projects in the next decade.

Nevertheless, I think that a critical analysis is required covering the goals cited above (Where and to what degree is standardisation of software processes useful?) as well as the RUP way to achieve these goals (Are the concepts and guidelines offered by the RUP appropriate to match the goals?).

Concerning the goals, some of them already seem to be too specific, e.g., “Provide guidance to the order of a team’s activities” or “Specify what artefacts should be