Unified Process

Methodology used by IBM Rational Software Architect and others

Unified Process

• In fact it is an extensible framework which should be customized for specific organizations or projects
• Grew out of Jacobson’s Objectory became Rational Unified Process (RUP), or as some prefer to call it Unified Process (UP)
• Iterative and incremental (example of spiral or evolutionary model)
• Production quality software released in piecemeal (incrementally) with each iteration
Unified Process

- Four phases
- Nine workflows or activities
- Deliverables
- Milestones

Four Phases

- Inception
- Elaboration
- Construction
- Transition
Inception

• establish business rationale
• decide project scope
• get commitment from sponsor to proceed further
• range from couple of hours chat and spreadsheet plan to a serious feasibility study over a couple of months

Elaboration

• collect more detailed requirements
• do high level analysis and design
• establish baseline architecture
• project planning
Construction

- key phase for iteration which can be in any phase
- each iteration builds production-quality software
- which is tested and integrated
- which satisfies a subset of the project requirements

Transition

- beta testing
- performance tuning
- user training
- user documentation
Four Phases - Resources

Process Structure

• Two dimensions.
• Horizontal axis represents time and shows the lifecycle aspects of the process as it unfolds.
• Vertical axis represents core process workflows or activities, which group activities logically by nature.
• 9 activities
Two dimensions of RUP

Phases

- Although most iterations will include work in most of the process disciplines (e.g. Requirements, Design, Implementation, Testing) the relative effort and emphasis will change over the course of the project.
Unified Process

• Iterative and Incremental
  – Although most iterations will include work in most of the process disciplines (e.g. Requirements, Design, Implementation, Testing) the relative effort and emphasis will change over the course of the project.

• Use Case Driven
  – In the Unified Process, use cases are used to capture the functional requirements and to define the contents of the iterations. Each iteration takes a set of use cases or scenarios from requirements all the way through implementation, test and deployment.

Unified Process

• Architecture Centric
  – The Unified Process insists that architecture sit at the heart of the project team's efforts to shape the system.
  – Since no single model is sufficient to cover all aspects of a system, the Unified Process supports multiple architectural models and views.
  – One of the most important deliverables of the process is the executable architecture baseline which is created during the Elaboration phase.
  – This partial implementation of the system serves to validate the architecture and act as a foundation for remaining development.
Unified Process

- Risk Focused
  - The Unified Process requires the project team to focus on addressing the most critical risks early in the project life cycle.
  - The deliverables of each iteration, especially in the Elaboration phase, must be selected in order to ensure that the greatest risks are addressed first.

Agile Unified Process (AUP)

- AUP is a simplified version of the IBM RUP developed by Scott Ambler.
- It describes a simple, easy to understand approach to developing business application software using agile techniques and concepts yet still remaining true to the RUP.
- The AUP applies agile techniques including test driven development (TDD), Agile Modelling, agile change management, and database refactoring to improve productivity.
Inception Objectives

- Establish software scope and boundary conditions.
  - operational concept.
  - acceptance criteria.
  - descriptions of what is and what is not included.
- Discriminate critical Use Cases of the system.
  - primary scenarios of behaviour.
- Exhibit at least one candidate architecture.
- Estimate overall cost.
- Estimate risks.

Outcome of Inception

- A ‘vision’ document, i.e., a general vision of the core projects requirements, key features and main constraints.
- A Use-Case model survey – all Use Cases and Actors that can be identified so far.
- An initial project glossary.
- An initial business case including business context, success criteria and financial forecast.
- Initial risk assessment.
- Project plan, with phases and iterations.
Other Artifacts Produced

- Initial Use Case model (10%-20% complete)
- A domain model *static picture of scope*.
- A business model (if necessary) *workflow*.
- A preliminary development case description to specify the process used.
- One or several prototypes.
  - Behavioral, Structural, Exploratory or Evolutionary.

Elaboration Objectives

- To analyse the problem domain.
- Establish a sound architectural foundation.
- Develop the project plan.
- Eliminate high-risk elements.
Elaboration objectives

• Define, validate and agree the architecture as quickly as possible.
• Agree the vision that came from the inception phase.
• Agree a plan for the construction phase.

Elaboration activities

• The vision is elaborated and a solid understanding is established of the most critical Use Cases that drive the architectural and planning decisions.
• The Process, the infrastructure and the development environment are elaborated, and the process, tools and automation support are put into place.
Elaboration activities

• The architecture is elaborated and components are selected.
  – Potential components are evaluated.
  – make / buy / reuse decisions determine the construction phase cost and schedule.
  – Architectural components integrated and assessed against primary scenarios.
  – This is done iteratively.

Outcome of Elaboration

• Use Case model (at least 80% complete).
  – All Use Cases identified.
  – All Actors identified.
  – Most Use-Case descriptions developed.
• Supplementary requirements.
  – (non-functional or not associated with a Use Case)
• Software architecture description.
Outcome of elaboration

• Executable architectural prototype.
• Revised risk list and revised business case.
• Development plan for overall project.
  – coarse grained project plan, with iterations and
    evaluation criteria for each iteration.
• Updated development case that specifies process to
  be used.
• Preliminary user manual (optional).

Evaluation criteria at end

• Is the vision of the product stable?
• Is the architecture stable?
• Does the executable demonstration show
  that major risk elements are addressed?
• Is construction phase sufficiently planned?
• Do all stakeholders agree that current vision
  is achievable, using current plan with
  current architecture?
• Is the cost acceptable?
Construction

• All remaining components and application features are developed and integrated into the product.
• All features are tested thoroughly.
• *Emphasis is placed on managing resources and controlling operations to optimise cost, schedules and quality.*
• Parallel construction can accelerate the availability of deployable releases.

Construction

• categorise the use cases
• customer divides uses cases according to business value:
  • high, medium and low
• developers divides use case according to risk, e.g. high risk means very difficult to do, may have big impact on system design or not well understood
Construction

- next, developer estimates time for each use case to nearest person-week including time for all stages analysis, design, coding, unit testing, integration and documentation
- high risk use cases involving a lot of project time may require more elaboration

Construction objectives

- Minimise development costs by optimising resources and avoiding unnecessary scrap and rework.
- Achieve adequate quality as rapidly as possible.
- Achieve useful versions (alpha, beta or other test releases) as rapidly as practical.
Construction activities

• Resource management, resource control, process optimisation.
• Complete component development and testing against the defined evaluation criteria.
• Assessment of product releases against acceptance criteria for the vision.

Outcome of construction

• A product ready to put into the hands of end users.
• The software product integrated on the adequate platforms.
• The user manuals.
• A description of the current release.
Evaluation criteria at end

• Often called the beta release, is it ready?
  – Is the product release stable and mature enough to be deployed in the user community?
  – Are all stakeholders ready for the transition into the user community?
  – Are the actual resource expenditures v planned expenditures still acceptable?

• Transition may have to be postponed by one release if the project fails to reach this milestone.

Transition

• This moves the software project to the user community.

• After release, issues usually arise that require new releases, either to correct problems or finish features that were postponed.

• This phase is entered when a baseline is mature enough to be deployed in the end-user domain.

• This means that some usable subset of the system has been completed to an acceptable level of quality and that user documentation is available.
Transition phase includes

- Beta testing to validate the new system against use expectations.
- Parallel operation with the legacy system that the project is replacing
- Conversion of operational databases.
- Training of users and maintainers.
- Rollout of the product to the marketing, distribution and sales teams.
- It concludes when the deployment baseline has achieved the completed vision.

Transition objectives

- Achieve user self-supportability.
- Achieve stakeholder concurrence that deployment baselines are complete and consistent with the evaluation criteria of the vision.
- Achieve final product baseline as rapidly and cost-effectively as practical.
Transition activities

• Deployment-specific engineering, i.e. cutover, commercial packaging and production, sales rollout, and field personnel training.

• Tuning activities, including bug fixing and enhancement for performance and usability.

• Assessing the deployment baselines against the vision and the acceptance criteria for the product.

• The activities depend on the goal
  – For fixing bugs, implementation and testing are usually enough.
  – For new features, iteration is similar to construction phase.