



OpenUP development process

Improving quality of software with agile approach

10. 07. 2008

Inhalt

Inhalt	2
What is OpenUP	3
OpenUP Domain	3
OpenUP Knowledge Base	4
OpenUP Principles	4
OpenUP Fundamentals - Roles.....	5
OpenUP Fundamentals - Disciplines	6
Requirements.....	6
Project Management	6
Configuration and change management.....	6
Architecture.....	7
Development.....	7
Test.....	7
OpenUP disciplines compared to RUP and OEP.....	8
OpenUP Fundamentals – Artifacts	8
OpenUP Fundamentals – Iterations and phases.....	9
Iterations	9
Phases	10
OpenUP tools and plugins.....	12
OpenUP and Quality	13
Iterative and incremental process.....	13
Artifact templates	13
Checklists	13
Guidelines	14
Discipline Test	14
Quality evaluation.....	14
Quality measurement.....	15
OpenUP and CMMI.....	Fehler! Textmarke nicht definiert.
Insono contribution to OpenUP	16
Assumptions management.....	16
Process and solutions improvement.....	16

What is OpenUP

OpenUP is a minimally sufficient software development process. It has been developed by IBM Rational employees as a part of Eclipse Process Framework Project. It is based on more complex Rational Unified Process (RUP) and it uses the basic RUP elements, such as:

- Clear definitions of phases, roles, artifacts and tasks
- Iterative and incremental approach
- Integration of risk management
- Focus on architecture
- Making quality the integral part of the entire process

When compared to RUP OpenUP is a minimal process – it provides only fundamental content. Thus, it does not provide guidance on many topics that software projects may deal with, such as large team sizes, compliance, contractual situations, safety or mission critical applications, technology-specific guidance, business modelling etc.

OpenUP is extensible – this means additional content may be added or tailored for specific needs of a given project or organization.

OpenUP is an agile process – this means enforcing communicating with one another providing a shared understanding of the project. Agile methods draw attention to the importance of coordinating understanding, benefiting stakeholders over unproductive deliverables and formality.

OpenUP Domain

OpenUP is a suitable process for projects which meet the following criteria:

- the project deliver software as a main product
- the project don't require high level of formality
- business users (stakeholders) are involved in the project significantly
- the requirements are not complete and business users are willing to communicate them during the project
- the project is small or medium size

OpenUP might not be a suitable process for projects which meet the following criteria:

- the project deliver other artifacts which has higher priority than software
- the project require high level of formality
- the business users (stakeholders) are isolated from the project
- the project is large

OpenUP Knowledge Base

Users of OpenUP process are guided by the OpenUP knowledge base which is a set of HTML pages. The pages may be deployed to a project or an organization intranet or website to be easily accessible for all participants.

The knowledge base covers all areas of OpenUP process (disciplines, roles, artifacts, lifecycle) and provides one common access point for it to all participants.

The following screenshot shows OpenUP knowledge base:

The screenshot displays the 'Inception Phase Iteration' page within the OpenUP Knowledge Base. The page features a navigation menu on the left with items like 'Introduction to OpenUP', 'Getting Started', 'OpenUP Disciplines', 'OpenUP Work Products', 'OpenUP Roles', 'OpenUP Lifecycle', 'About', and 'OpenUP Copyright'. The main content area is titled 'Capability Pattern: Inception Phase Iteration' and includes a description: 'This iteration template defines the activities (and associated roles and work products) performed in a typical iteration in the Inception phase.' Below the description are tabs for 'Description', 'Work Breakdown Structure', 'Team Allocation', and 'Work Product Usage'. The 'Work Breakdown Structure' tab is active, showing a workflow diagram with activities: 'Initiate Project', 'Plan and Manage Iteration', 'Identify and Refine Requirements', and 'Agree on the Technical Approach'. Below the diagram is a 'Work Breakdown' table.

Breakdown Element	Steps	Index	Predecessors	Model Info	Type	Planned	Repeatable	Multiple Occurrences	Ongoing	Event Driven	Optional	Team
Initiate Project	1				Activity	✓						
Plan and Manage Iteration	4				Activity				✓			
Identify and Refine Requirements	8	1			Activity	✓						
Agree on the Technical Approach	12	1			Activity	✓						

OpenUP Principles

OpenUP is driven by the four core principles :

1. Collaborate to align interests and share understanding. This principle promotes practices that foster a healthy team environment, enable collaboration and develop a shared understanding of the project.
2. Balance competing priorities to maximize stakeholder value. This principle promotes practices that allow project participants and stakeholders to develop a solution that maximizes stakeholder benefits, and is compliant with constraints placed on the project.

3. Focus on the architecture early to minimize risks and organize development. This principle promotes practices that allow the team to focus on architecture to minimize risks and organize development.
4. Evolve to continuously obtain feedback and improve. This principle promotes practices that allow the team to get early and continuous feedback from stakeholders, and demonstrate incremental value to them.

OpenUP Fundamentals - Roles

The following roles are defined by OpenUP:

- **Stakeholder** represents interest groups whose needs must be satisfied by the project. It is a role that may be played by anyone who is (or potentially will be) materially affected by the outcome of the project
- **Project Manager** leads the planning of the project in collaboration with stakeholders and team, coordinates interactions with the stakeholders, and keeps the project team focused on meeting the project objectives.
- **Analyst** represents customer and end-user concerns by gathering input from stakeholders to understand the problem to be solved and by capturing and setting priorities for requirements.
- **Architect** is responsible for designing the software architecture, which includes making the key technical decisions that constrain the overall design and implementation of the project.
- **Developer** is responsible for developing a part of the system, including designing it to fit into the architecture, and then implementing, unit-testing, and integrating the components that are part of the solution.
- **Tester** is responsible for the core activities of the test effort, such as identifying, defining, implementing, and conducting the necessary tests, as well as logging the outcomes of the testing and analyzing the results.
- **Any Role** represents anyone on the team that can perform general tasks.

OpenUP Fundamentals - Disciplines

OpenUP defines the following disciplines:

Requirements

The purpose of this discipline is to:

- Understand the problem to be solved
- Understand stakeholder needs (what users want)
- Define the requirements for the solution (what the system must do)
- Define the boundaries (scope) of the system
- Identify external interfaces for the system
- Identify technical constraints on the solution
- Provide the basis for planning iterations
- Provide the initial basis for estimating cost and schedule

Project Management

The purpose of this discipline is to:

- Encourage stakeholder consensus on prioritizing the sequence of work
- Stimulate team collaboration on creating long term and short term plans for the project.
- Focus the team on continually delivering tested software for stakeholder evaluation
- Help create an effective working environment to maximize team productivity
- Keep stakeholders and the team informed on project progress
- Provide a framework to manage project risk and continually adapt to change

Configuration and change management

The purpose of this discipline is to:

- Maintain a consistent set of work products as they evolve
- Maintain consistent builds of the software
- Provide an efficient means to adapt to changes and issues, and re-plan work accordingly
- Provide data for measuring progress

Architecture

- This discipline explains how to create an architecture from architecturally significant requirements. The architecture is built in the Development discipline.
- The architecture can be represented in many forms and from many viewpoints, depending on the needs of the project and the preferences of the project team. It need not be a formal document. The essence of the architecture can often be communicated through a series of simple diagrams on a whiteboard; or as a list of decisions. Choose the medium that best meets the needs of the project.
- The architecture can be expressed as a simple metaphor or as a comparison to a predefined architectural style or set of styles. It may be a precise set of models or documents that describe the various aspects of the system's key elements. Expressing it as skeletal build is another option - although this build may need to be baselined and preserved to ensure that the essence of the system can be understood as the system grows.

Development

The purpose of this discipline is to:

- Transform the requirements into a design of the system-to-be
- Adapt the design to match the implementation environment
- Build the system incrementally
- Verify that the technical units used to build the system work as specified

Test

The purpose of this discipline is to:

- Provide early and frequent feedback that the system satisfies the requirements
- Objectively measure progress in small increments
- Identify issues with the solution
- Provide assurance that changes to the system do not introduce new defects
- Improve velocity by facilitating the discovery of issues with requirements, designs, and implementations as early as possible

OpenUP disciplines compared to RUP and OEP

The following table shows differences in disciplines for OpenUP and other development processes - RUP (Rational Unified Process) and OEP (Open Engineering Process):

OpenUP	RUP	OEP
Architecture	not a separate discipline – architecturally significant parts of other disciplines: Requirements, A&D, Implementation, Test	Technical Architecture and System Development. Domain Architecture
Configuration and Change Management	Configuration and Change Management	Configuration management in Configuration- and environment management, Change Management in Iteration, process and change management
Development	Analysis and Design, Implementation	System Development
Project Management	Project Management	Project Management
Requirements	Requirements	Requirement Analysis
Test	Test	Quality Assurance, Test, Usability
not present	Business Modelling	Business Process Modeling
not present	Deployment	Operation, Distribution and Project External
not present	Environment	Environment and Batch

OpenUP Fundamentals – Artifacts

An artifact is something that is produced, modified, or used by a task.

OpenUP organizes artifacts activities by introducing the following:

- Roles are responsible for creating and updating artifacts.

- Artifacts are subject to version control throughout the project lifecycle.
- The 17 artifacts in OpenUP are considered the essential artifacts a project should use to capture product- and project-related information.
- Templates provide an out-of-the box, standard way to capture information.

OpenUP Fundamentals – Iterations and phases

Iterations

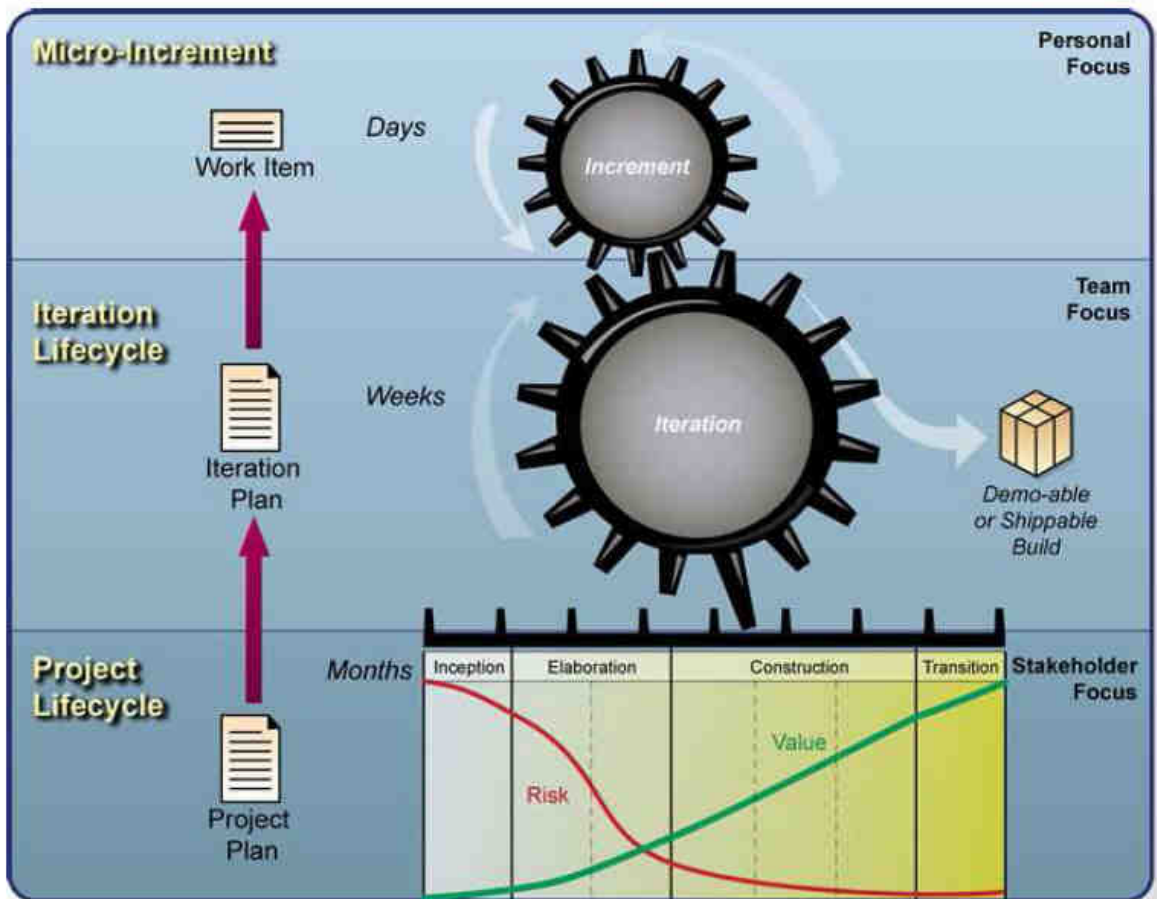
An iteration is a set period of time within a project in which you produce a stable, executable version of the product, together with any other supporting documentation, install scripts, or similar artifacts necessary to use this release. The executable is demonstrable, allowing the team to demonstrate true progress to stakeholders, and get feedback on how they are doing so that they can improve their understanding of what needs to be done and how to do it.

Typical iteration in OpenUP lasts from 1 week to 6 weeks.

Each iteration builds upon the results of the previous iteration, and will produce a product increment one step closer to the final product. Iterations are timeboxed, meaning that the schedule for an iteration should be regarded as fixed, and the scope of the iteration's content actively managed to meet that schedule.

Each iteration should address the most critical risks, and implement the highest-priority Work Items. This ensures that each iteration adds maximum stakeholder value, and reduces uncertainty. Iterative development is typically combined with frequent or continuous integration: as unit-tested components become available, they are integrated, then a build is produced and subjected to integration testing. In this way, the capability of the integrated software grows as the iteration proceeds, towards the goals set when the iteration was planned.

The following illustration shows the role of iterations in the overall project lifecycle:



Phases

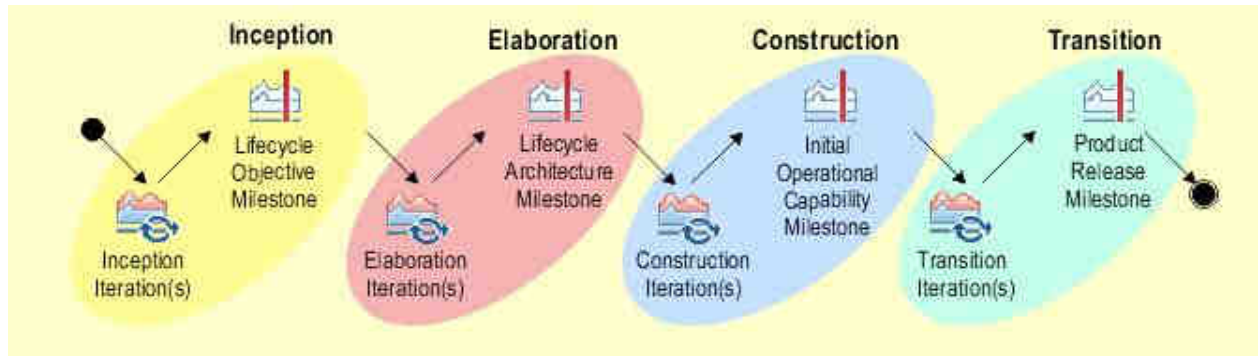
OpenUP organizes iterations into a set of phases. Phase is a set of iterations – a phase may consist of just one iteration (typically for inception phase) or many of them.

Each phase ends with a milestone aimed at providing oversight by raising and answering a set of questions that are typically critical to stakeholders:

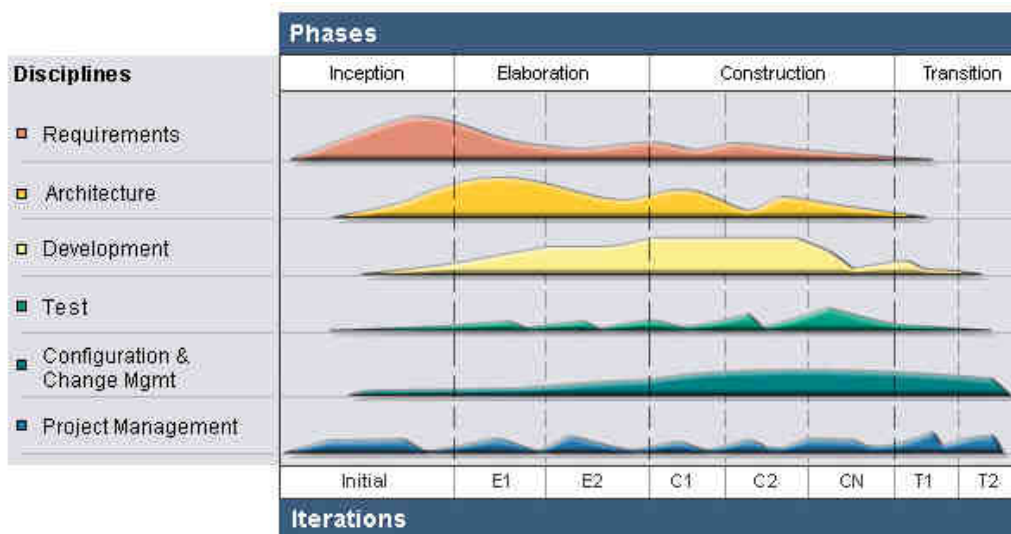
- **Inception.** Do we agree on project scope and objectives, and whether or not the project should proceed?
- **Elaboration.** Do we agree on the executable architecture to be used for developing the application and do we find that the value delivered so far and the remaining risk is acceptable?
- **Construction.** Do we find that we have an application that is sufficiently close to being released that we should switch the primary focus of the team to tuning, polishing and ensuring successful deployment?
- **Transition.** Is the application ready to release?

If the answer is Yes to the above questions at the phase review, the project continues. If the answer is No, the phase is delayed (usually by adding an extra iteration) until a satisfactory answer is received, or the stakeholders may determine that the project should be cancelled.

The following illustration shows the phases and milestones in OpenUP:



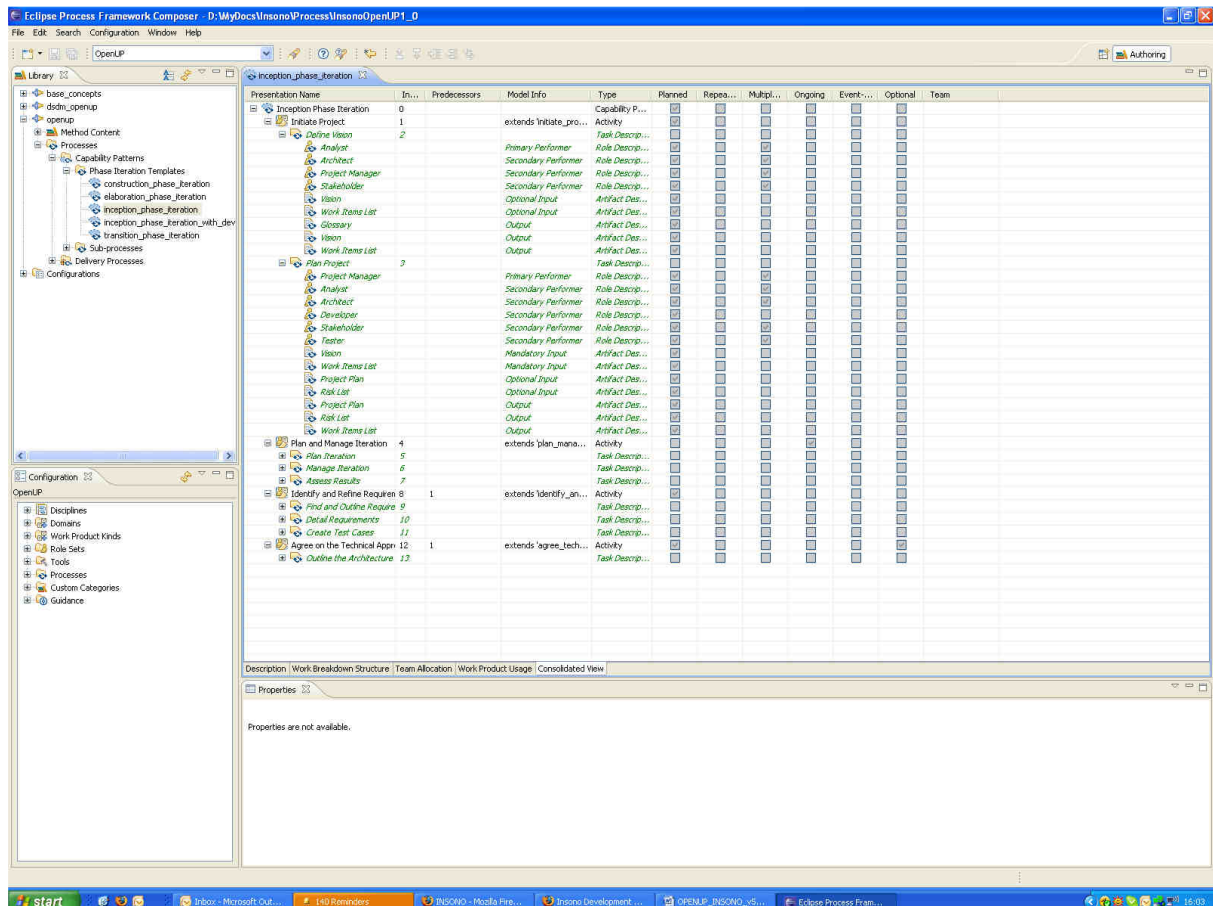
The following illustration shows effort spent on various disciplines during the phases of a project:



OpenUP tools and plugins

The base OpenUp can be extended with Eclipse Process Framework Composer tool. Additional content can be added to the base knowledge base or company specific document templates can be provided. The existing content can be changed (tailored) to meet specific needs of particular project or company.

The following screenshot shows Eclipse Process Framework Composer tool:



Independent tool vendors can use Eclipse Process Framework Composer tool to create plug-ins that attach tool mentors to tasks giving step-by-step instructions on how the tool can be used within the context of the process.

The examples of plugins available:

- The Agile Database Techniques (ADT) Plug-In
- The Agile Model Driven Development (AMDD) Plug-In
- APG TOGAF Process Library (ATPL)
- Agile Enterprise Architecture

OpenUP and Quality

The basic OpenUP helps achieving quality by:

- Iterative and incremental process enforcing frequent feedback about quality
- Providing artifact templates
- Specyfing quality criteria for artifacts – checklists
- Specyfing techniques which helps achieving quality – guidelines
- Providing QA related process elements – discipline Test

Additionally the basic process can be extended by the following elements of RUP:

- Quality evaluation
- Quality measurement (metrics)

Iterative and incremental process

Iterative process improves quality by enforcing early and frequent feedback about software development results. The feedback is related to the real executable software not just intermediate documentation. Thus the value of such feedback is much higher because even serious flaws of requirements or design documentation may not be obvious until the software is created.

Incremental deliveries results in better prioritization of QA activities based on importance and risk related to system use cases. Use cases which are more important or more risky are put under QA activities earlier and there is more time until the end of the project to improve their quality.

Technique which takes incrementality to the extreme is continuous integration. It allows checking the quality of integrated software just after addition of new software elements by developer, at specified times (eg. 1am) or at specified time intervals (eg. 1 hour).

Artifact templates

OpenUP provides templates for most of the documents needed in the process. The templates help identifying information necessary to create documents meeting the quality standards.

The templates provided by OpenUP may be extended or tailored according to the needs of the project or organization which uses it.

Checklists

In OpenUP most of the artifacts have checklist. Checklists are useful in a number of contexts: they help you decide what to do, they help doing it, they help assess the quality of the work, and they help understand how a particular artifact relates to the rest of the process.

From quality point of view checklist help decide whether an artifact meets the common quality criteria for this artifact type.

Guidelines

A Guideline usually focuses on how to perform a particular task or grouping of tasks (for example, grouped together as activities) or provides additional detail, rules, and recommendations on artifacts and their properties. Guidelines can include details about a variety of topics including:

- Practices and different approaches for doing work
- How to handle particular kinds of artifacts
- Information on different subtypes and variants of artifacts and how they evolve throughout a lifecycle
- Discussions on skills the performing roles should acquire
- Measurements of progress and maturity, etc.

Discipline Test

OpenUP provides entire part of the process for testing software. This include:

- Tester role with specified areas of responsibility
- Artifacts: Test Case, Test Script with templates
- Checklists and guidelines for Test Case and Test Script
- Description of test related activities performed by Tester role
- Guidelines for test related activities

Quality evaluation

Inspections, Reviews, and Walkthroughs are specific techniques focused on evaluating work products and are a powerful method of improving the quality and productivity of the development process. Conducting these should be done in a meeting format, with one role acting as a facilitator, and a second role recording notes (change requests, issues, questions, and so on).

The IEEE standard Glossary (1990 Ed.) defines these three kinds of efforts as:

- **Review** - a formal meeting at which an work product, or set of work products are presented to the user, customer, or other interested party for comments and approval.
- **Inspection** - a formal evaluation technique in which work products are examined in detail by a person or group other than the author to detect errors, violations of development standards, and other problems.
- **Walkthrough** - a review process in which a developer leads one or more members of the development team through a segment of a work product that he or she has written while the other members ask questions and make comments about technique, style, possible errors, violation of development standards, and other problems.

Quality measurement

The quality criteria may be stated in many ways and may include more than one measure. Common acceptance criteria may include the following measures:

- Defect counts and / or trends, such as the number of defects identified, fixed, or that remain open (not fixed).
- Test coverage, such as the percentage of code, or use cases planned or implemented and executed (by a test). Test coverage is usually used in conjunction with the defect criteria identified above).
- Performance, such as the time required for a specified action (use case, operation, or other event) to occur. This is criteria is commonly used for Performance testing, Failover and recovery testing, or other tests in which time criticality is essential.
- Compliance. This criteria indicates the degree to which each work product, activity, task, or step must meet an agreed upon standard or guideline.
- Acceptability or satisfaction. This criteria is usually used with subjective measures, such as usability or aesthetics.

Although OpenUP doesn't provide quality measurement out of the box, it can be extended with appropriate technique and tools like: JIRA, Coverage, JMeter.

Insono contribution to OpenUP

Insono has extensive knowledge and experience gained from demanding software development and systems integration projects. We used this knowledge and experience to extend the basic OpenUP with the following elements:

- assumptions management
- process and solutions improvement

Assumptions management

Some projects suffer from incomplete or vague requirements. Processes like OpenUP, RUP or OEP do not help with such kinds of project problems.

Insono developed its custom method of dealing with problems related to requirements discipline.

By :

- introducing formal concepts of open point and assumption
- including the concepts in the OpenUP Requirements discipline
- providing set of techniques for managing open points and assumptions

Insono significantly improved effectiveness of requirements management for OpenUP and project management in general.

Process and solutions improvement

Insono extended the document templates for Project Management discipline with sections related to process and solutions improvement. This helps with creation of organizational asset – base of knowledge gained during projects which is documented and may be reused in the future .

Insono offer for RUP

For project which requires high level of formality Insono offers Rational Unified Process expertise. RUP is also a suitable choice for companies aiming at high level process standardization with CMMI.

CMMI for Development (CMMI-DEV) is the most popular approach which defines process practices for products and services development that should be implemented in an organization to better perform and successfully achieve its business goals.

The study published on IBM developerworks website:

<http://www.ibm.com/developerworks/rational/library/dec05/grundmann/>
shows that RUP covers 97 percent of the CMMI requirements for Maturity Level 2