

# Rational Unified Process

Anders Hessel  
Information Technology  
Uppsala University



## What is RUP?

- RUP is a Software Engineering Process
- Several out-of-the-box instances
- Can be tuned from lightweight to comprehensive
- An answer to the strict waterfall method or Documented driven development

Informationsteknologi

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se



## RUP Components

- Six best practices
- Four phases
- Static structure of the process
- Nine workflows
  - Core process workflows
  - Core supporting workflows

Informationsteknologi

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se



## Breakdown

- In Functionality
- In Components
- Synchronized in time

Informationsteknologi

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se



## Time

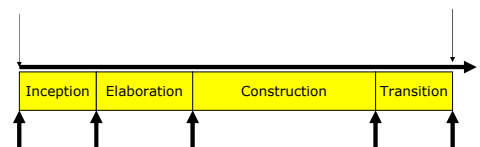


Informationsteknologi

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se



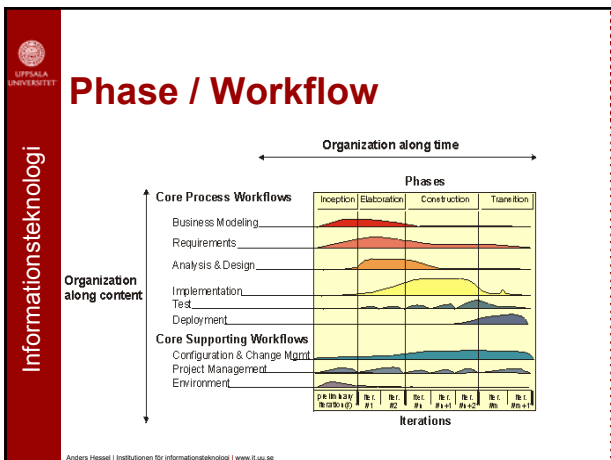
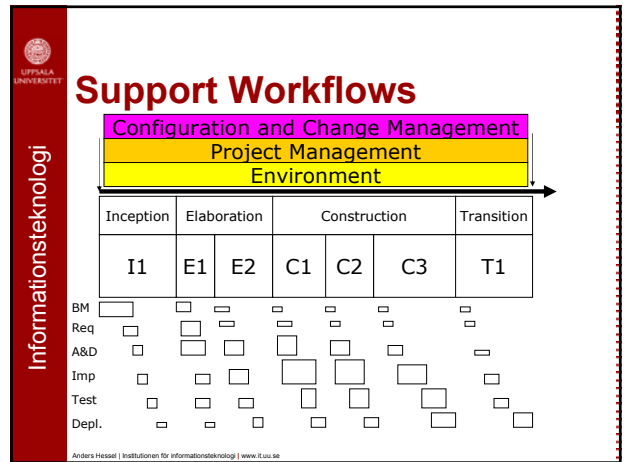
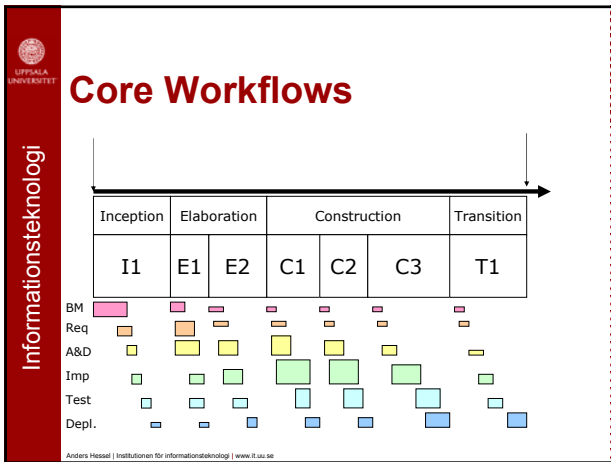
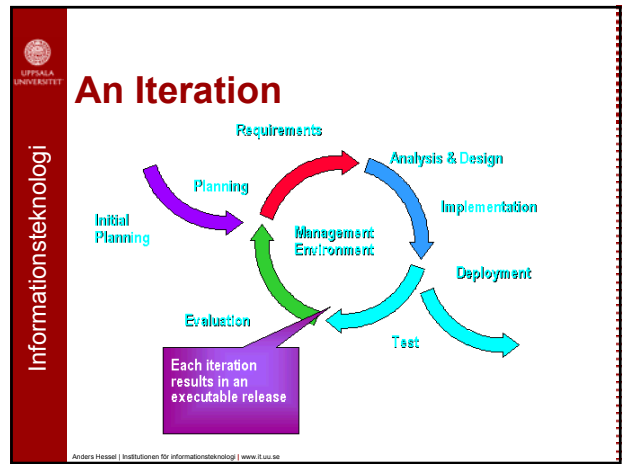
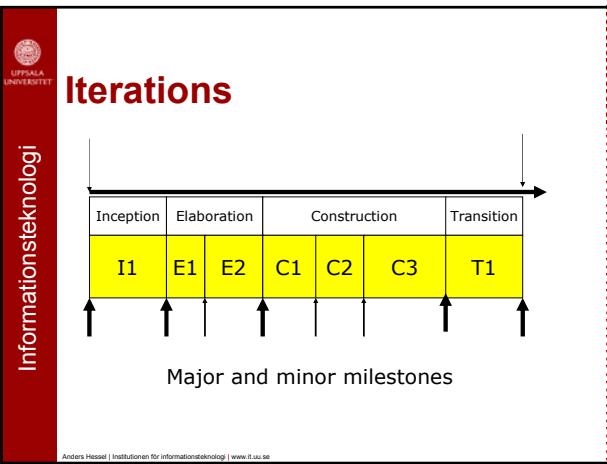
## Maturity Phases



Major milestones

Informationsteknologi

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se



- Informationsteknologi
- # Six Best Practices
- Develop Iteratively
  - Manage Requirements
  - Model Visually
  - Use Component Architecture
  - Verify Quality
  - Control Changes
- Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se

## Develop Software Iteratively

- Hard to define the entire problem
- Iterations allows increasing understanding
- Reduce risk by demonstrating progress frequent
- Early start of implementation
- Up and running (part by part)
- Deliver testable systems
- FUN

## Manage Requirements

- Accept that requirements change
- Elicit, organize, and document required functionality and constraints
- Track and document tradeoffs and decisions
- Evaluate impact of changes
- Capture and communicate business requirements

## Requirements

- Functionality
- Usability
- Reliability
- Performance
- Supportability
- Design constraint

## Use Component Architecture

- Early development of robust executable architecture
- Intuitively understandable
- Promotes effective software reuse
- Components are non-trivial modules, subsystems with clear functions
- Stable independent modules with clear interfaces isolates software dependencies

## Model Visually (UML)

- Visual abstractions help to:
  - Communicate different aspects of the software
  - See how the elements fit together
  - Make sure that building blocks are consistent with code
  - Maintain consistency between design and its implementation
  - Promote unambiguous communication

## Models

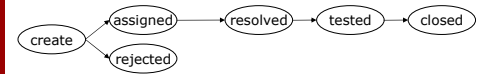
- Use-Case Diagram
- Class Diagram
- Object Diagram
- State Diagram
- Component Diagram
- Collaboration Diagram
- Sequence Diagram
- Activity Diagram
- Deployment Diagram

## Verify Quality

- Built into the whole process
- Use objective measurements
- Pays off to find problems early
- Real test gives safe status
- Test high risk areas more thoroughly

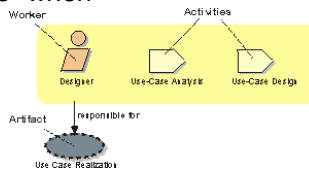
## Control Changes to Software

- Each change is acceptable
- Track changes
- Secure workspace for each developer
- E.g. Trouble report / Change request



## Static Structure of the Process

- Workers, the “who”
- Activities, the “how”
- Artifacts, the “what”
- Workflows, the “when”



## Worker



- A worker is a “hat” not a person
- One person can have many hats
  - Worker
    - Designer
    - Use-Case Author
    - Use-Case Designer
    - Design Reviewer
    - Architect
  - Activities
    - Object Design
    - Detail Use-Case
    - Use-Case Design
    - Review the Design
    - Arch. A&D

## Activities



- Unit of work for a specific worker
- Activity has a clear purpose e.g. creating or updating an artifact
  - Activity
    - Plan an iteration
    - Find Use-Cases and actors
    - Review the design
    - Execute performance test
  - Worker
    - Project Manager
    - System Analyst
    - Design Reviewer
    - Performance tester

## Artifacts



- Piece of information (produced, modified or used) by a process
  - A model: Use-Case or Design
  - A model element: Class, Use-Case, or Subsystem
  - A document: Business Case or SAD
  - Source Code
  - Executables

Informationsteknologi

Uppsala Universitet

## Workflows

- A WF is a sequence of activities that produces a result of observable value.

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se

Informationsteknologi

Uppsala Universitet

## Core Workflows

- Business Modeling
- Requirements
- Analysis and Design
- Implementation
- Test
- Deployment

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se

Informationsteknologi

Uppsala Universitet

## Business modeling

- Active during inception and elaboration
- Workers
  - Business process analyst
  - Business designer
  - Business model reviewer

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se

Informationsteknologi

Uppsala Universitet

## Requirements

- Describe system functionality
- Make agreement with developer and customer
- System analyst:
  - Common vocabulary
  - requirement management plan
  - find actors and Use-Cases
  - develop vision
  - elicit stakeholders requests
- Software architect: Prioritize Use-Cases
- Req. reviewer: Review req.

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se

Informationsteknologi

Uppsala Universitet

## Modern SRS Package (SRS)

- Modern Software Requirements Specification Package

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se

Informationsteknologi

Uppsala Universitet

## Analysis & Design

- Show how the system will be realized
- Architect:
  - Architectural analysis
- Designer:
  - Use-Case analysis
  - Use-Case design
  - Subsystem design
  - Class design

Anders Hessel | Institutionen för informationsteknologi | www.it.uu.se

## Implementation

- **Code and test the system**
- **Implementor:**
  - Implement a component
  - Fix a defect
  - Perform unit test
- **Integrator**
  - Plan system integration
  - Plan subsystem integration
  - Integrate subsystem
  - Integrate system
- **Code reviewer**
  - Review code

## Test

- Test cases procedures and other verification methods are created and described
- **Test designer**
  - Plan/design/implement test ...
  - Evaluate test
- **Tester:**
  - Execute test
- **Designer:**
  - Design test classes and packages

## Deployment

- Pack, distribute and install the software at the end-user
- **Deployment Manager:**
  - Deployment plan
  - Manage acceptance test
  - Define bill of materials
  - Write release notes
- **Technical writer**
  - Develop support materials

## Core Supporting Workflows

- Project Management
- Configuration and Change Management
- Environment

## Configuration and change Management

- **Monitor and administrate changes in the projects artifacts** (keep consistent with requirements)
- **Configuration Manager (CM):**
  - Responsible for release and version control
  - Convener of the CCB
  - Set up CM environment
  - Establish CM policies
  - Write CM plan
- **Change Manager**
  - Establish change control process
  - Review change request

## Project Management

- Elicit suitable strategies for the iterative process
- Balance competing objectives
- Manage risks
- Overcome constraints to deliver a product which meets the needs of customers and users

## Project Manager

- Initiate project
- Develop iteration plan
- Monitor project status
- Schedule and assign work
- Report status
- Handle exceptions and problems

## Project Reviewer

- Project approval review
- Project planning review
- Iteration plan review
- Iteration evaluation criteria review

## Environment

- **Provide the software development organization – both process and tools – needed.**
- Process engineer
  - Development case
  - Project specific templates
- Software architect:
  - Design guidelines
  - Programming guidelines
- Tool specialist:
  - Tool guidelines
  - Tools

## RUP Phases

- Inception (from TG0 to TG1)
- Elaboration (from TG1 to TG2)
- Construction (from TG2 to TG4)
- Transition (from TG4 to TG5)
  
- Iterations 1+2+3+2

## Inception Phase

- Inception means Start
- Formulate objectives
  - High level system interaction with actors
  - Business case includes:
    - Success criteria
    - Risk assessments
    - Estimate of resources
    - Create phase plan with major milestones

## Inception Outcome

- Vision document
- Initial Use-Case Study (10%-20%)
- Initial project glossary
- Initial business case
- Initial risk assesment
- Project plan
- Business model
- Prototype(s)

## Milestone Lifecycle Objectives

## Elaboration Phase

- Analyze the problem domain
- Establish sound architectural foundation
- Develop project plan
- Eliminate high risk elements

**Mile wide and inch deep  
view of the system**

## Elaboration Outcome

- Use-Case model (at least 80%)
  - All identified, most developed
- Supplementary requirements captured
- A Software Architecture Description
- Executable architectural prototype
- Revised risk list and revised business case
- A development plan for the overall project (course grained project plan showing iterations and their evaluation criteria)
- **A preliminary user manual** (optional)

## Milestone Lifecycle Architecture

## Construction

- All remaining components and application features are:
  - Developed and integrated to the product
  - Thoroughly tested

## Construction Outcome

- **A product ready to put in the hands of the end users**
- Consists of (at minimum):
  - SW product integrated on the adequate platform
  - The user manual
  - A description of current release
- **Considered as beta-release**

## Milestone Initial Operational Capability

## Transition

- Take the SW product to the user community
- Issues that usually arises:
  - New releases
  - Corrections
  - Finish the features that was postponed
- Beta-testing to validate new system against user expectations

## Milestone Product Release

~END~