The Unified Process

Robert B. France
Dept. of Computer Science
Colorado State University
PROCESS REVIEW
Essence of Software Development

- Requirements Specification
- Design
- Implementation

© Robert B. France
Validation vs. Verification

• Validation is concerned with establishing that a design or an implementation satisfies users
  – Are we building the right thing?

• Verification is concerned with establishing that a development artifact (e.g., design or code) satisfies formal specifications
  – Did we build it right?
What is a Process and why do we need a systematic process?

• A software process is a sequence of steps required to develop or maintain software (Watts Humphrey)

• A process is a series of steps involving activities, constraints, and resources that produce an intended output

• Good people + good process = lower risk of project failure
Evolution of Software Process Models

- “Code and Fix” Model
- Waterfall model
- Spiral model
- Incremental/iterative agile processes
- Unified process
Choosing a Model

• Choice depends on nature of project:
  – Are requirements clearly defined and stable?
  – Is there pressure to produce a working product quickly?
  – Are the consequences of operational errors serious?
Radical vs. Conservative Models

• More radical models suitable when:
  – quick results are needed
  – requirements are fuzzy or unstable

• More conservative models suitable when:
  – consequences of errors are very serious
  – requirements are well-understood and stable
Unified Process:
Introduction to an Iterative, Incremental OOA/D Process
The Unified Process (UP)

• The Unified Process is an industry standard software engineering process
  – It is the generic process for the UML
  – It is free - described in "The Unified Software Development Process",
    ISBN:0201571692"

• UP is:
  – Use case (requirements) driven
  – Iterative and incremental

• UP is a generic software engineering process. It has to be customised (instantiated) for your project
  – In house standards, document templates, tools, databases, lifecycle modifications, …

• Rational Unified Process (RUP) is an instantiation of UP
  – RUP is a product marketed and owned by IBM
  – RUP also has to be instantiated for your project
UP history

- 1967: Jacobson working at Ericsson
- 1976: Jacobson establishes Objectory AB
- 1987: Rational acquires Objectory AB
- 1995: Rational Unified Process (RUP)
- 1997: UML becomes an industry standard
- 1998: Rational Unified Development Process
- 1999: Ongoing RUP development
- 2001: RUP 2001
- 2004: Ongoing RUP development
UP Basics

- Small steps, feedback and evolution
- Iterative, incremental, time-boxed
- Risk-driven

© Robert B. France
Key Practices

• Deliver product in increments developed in iterations
• Iterations are carried out in a fixed time
  – Developers can choose to drop features but should not extend iteration
• High risk and high value aspects tackled in early iterations
  – Cohesive architecture implemented in early iterations
• Customers continuously engaged in evaluation, feedback and requirements elicitation
• Continuously verify quality; test-driven code development
• Model software
Motivating Time-boxing

• “Work expands so as to fill the time available for completion” (Parkinson’s Law)
• Forces prioritization of tasks and risks
• Gain confidence of customers
• Build team confidence/satisfaction

© Robert B. France
Iterations

• Iterations are the key to the UP
• Each iteration is like a mini-project including:
  – Planning
  – Analysis and design
  – Integration and test
  – An internal or external release
• We arrive at a final product release through a sequence of iterations
• Iterations can overlap - this allows parallel development and flexible working in large teams
  – Requires careful planning
• Iterations are organised into phases
Iteration Structure

Requirements phase.

- Conceive, Plan, Requirements, etc.
- Analyze
- Design
- Test
- Implement

Iterative phase.
Development cycles

© Robert B. France
Iteration workflows

UP specifies 5 core workflows

Requirements → Analysis → Design → Implementation → Test

Each iteration may contain all the core workflows but with a different emphasis depending on where the iteration is in the lifecycle.

An iteration

Planning → Project specific → Assessment

other workflows

© Clear View Training 2010 v2.6
Baselines and increments

• Each iteration generates a baseline
• A baseline is a set of reviewed and approved artefacts that:
  – Provide an agreed basis for further review and development
  – Can be changed only through formal procedures such as configuration and change management
• An increment is the difference between the baseline generated by one iteration and the baseline generated by the next iteration
  – This is why the UP is called “iterative and incremental”
Each phase can include several iterations
- The exact number of iterations per phase depends on the size of the project! e.g. one iteration per phase for small projects

Each phase concludes with a major milestone.
For each phase we will consider:
The focus in terms of the core workflows
The goal for the phase
The milestone at the end of the phase

© Clear View Training 2010 v2.6
Phases

• Inception
  – Early exploration of problem to determine project feasibility
  – What’s the perceived business value?
  – What are the risks?

• Elaboration
  – Requirements detailing (major requirements identified)
  – Iterative implementation of “core” architecture

• Construction
  – Iterative development of remaining low-risk elements
  – Prepare for deployment

• Transition
  – Beta tests
  – Deployment
# Inception

Inception is the initial phase of the project lifecycle where the groundwork is laid for the project. The focus is on establishing a business case, identifying key requirements, designing a proof of concept or technical prototypes, and implementing these prototypes. The amount of work in each core workflow is as follows:

### Focus

<table>
<thead>
<tr>
<th>Focus</th>
<th>Inception</th>
<th>Elaboration</th>
<th>Construction</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements – establish business case and scope. Capture core requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis – establish feasibility</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design – design proof of concept or technical prototypes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementation – build proof of concept or technical prototype</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test – not generally applicable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Goals

- Establish feasibility of the project - create proof of concept/technical prototypes
- Create a business case
- Scope the system - capture key requirements
- Identify critical risks
Inception - milestone

- Life Cycle Objectives - conditions of satisfaction:
  - System scope has been defined
  - Key requirements for the system have been captured. These have been defined and agreed with the stakeholders
  - An architectural vision exists. This is just a sketch at this stage
  - A Risk Assessment
  - A Business Case
  - Project feasibility is confirmed
  - The stakeholders agree on the objectives of the project
# Elaboration

<table>
<thead>
<tr>
<th>Focus</th>
<th>Requirements – refine system scope and requirements</th>
<th>Inception</th>
<th>Elaboration</th>
<th>Construction</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Analysis – establish what to build</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Design – create a stable architectural baseline</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Implementation – build the architectural baseline</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Test – test the architectural baseline</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Goals                  | Create an executable architectural baseline         |           |             |              |            |
|                        | Refine Risk Assessment and define quality attributes (defect rates etc.) |           |             |              |            |
|                        | Capture use cases to 80% of the functional requirements |           |             |              |            |
|                        | Create a plan with sufficient detail for the construction phase |           |             |              |            |
|                        | Formulate a bid which includes resources, time, equipment, staff, cost |           |             |              |            |
Elaboration - milestone

- Lifecycle Architecture - conditions of satisfaction:
  - A resilient, robust executable architectural baseline has been created
  - The Risk Assessment has been updated
  - A project plan has been created to enable a realistic bid to be formulated
  - The business case has been verified against the plan
  - The stakeholders agree to continue
## Construction

<table>
<thead>
<tr>
<th>Focus</th>
<th>Requirements – uncover any requirements that had been missed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Analysis – finish the analysis model</td>
</tr>
<tr>
<td></td>
<td>Design – finish the design model</td>
</tr>
<tr>
<td></td>
<td>Implementation – build the Initial Operational Capability</td>
</tr>
<tr>
<td></td>
<td>Test – test the Initial Operational Capability</td>
</tr>
<tr>
<td>Goals</td>
<td>Complete use case identification, description and realization</td>
</tr>
<tr>
<td></td>
<td>Finish analysis, design, implementation and test</td>
</tr>
<tr>
<td></td>
<td>Maintain the integrity of the system architecture</td>
</tr>
<tr>
<td></td>
<td>Revise the Risk Assessment</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Focus</th>
<th>Inception</th>
<th>Elaboration</th>
<th>Construction</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements – uncover any</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>requirements that had been missed</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis – finish the analysis model</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design – finish the design model</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementation – build the Initial</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operational Capability</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test – test the Initial Operational</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Capability</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© Clear View Training 2010 v2.6
Construction - milestone

• Initial Operational Capability - conditions of satisfaction:
  – The product is ready for beta testing in the user environment
# Transition

<table>
<thead>
<tr>
<th>Focus</th>
<th>Requirements – not applicable</th>
<th>Analysis – not applicable</th>
<th>Design – modify the design if problems emerge in beta testing</th>
<th>Implementation – tailor the software for the user site. Fix bugs uncovered in beta testing</th>
<th>Test – perform beta testing and acceptance testing at the user site</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goals</td>
<td>Correct defects</td>
<td>Prepare the user site for the new software and tailor the software to operate at the user site</td>
<td>Modify software if unforeseen problems arise</td>
<td>Create user manuals and other documentation</td>
<td>Provide customer consultancy</td>
</tr>
</tbody>
</table>

© Clear View Training 2010 v2.6
Transition – milestone

• Product Release - conditions of satisfaction:
  – Beta testing, acceptance testing and defect repair are finished
  – The product is released into the user community
Summary

• UP is a risk and use case driven, architecture centric, iterative and incremental software development process

• UP has four phases:
  – Inception
  – Elaboration
  – Construction
  – Transition

• Each iteration has five core workflows:
  – Requirements
  – Analysis
  – Design
  – Implementation
  – Test