National University
of computer and emerging sciences

Foundation for Advancement
of Science and Technology

SOFTWARE DESIGN AND ANALYSIS
CS 3004

Iterative development and The Unified process

Fall 2021
Dr. Muhammad Bilal

Outline

1. The Unified process
2. The UP Disciplines
3. Agile Modeling
4. The Rational Unified Process
5. UP phases are iterative & incremental
6. Advantages & Disadvantages of RUP

Learning Objectives

- Overview UML and visual agile modeling
- Understand Agile manifesto and Principles
- Understand Waterfall and Unified Process

Process ?

Delivering of a software involves many steps, ranging from understanding of the problem to design, testing and implementation. A software development process takes you through these steps in a methodical way

Discipline	Artifact Iteration-*	Incep. 11	Elab. El. .En	Const. CL.Cn	Trans. T1..T2
Business Modeling	Domain Model		s		
Requirements	Use-Case Model	s	r		
	Vision	s	r		
	Supplementary Specification	s	r		
	Glossary	s	r		
Design	Design Model		s	r	
	SW Architecture Document		s		
	Data Model		s	r	
Implementation	Implementation Model		s	r	r
Project Management	SW Development Plan	s	r	r	r
Testing	Test Model		s	r	
Environment	Development Case	s	r		

Table 2.1 Sample Development Case of UP artifacts, s - start; r - refine

Waterfall Method

- Requirements Analysis
  - Analysis Specification
- Design Specification
- Coding from Design Specification
- Testing
  - Unit Testing
  - System Testing
  - UAT Testing
  - Ship It (????)
- Measuring rod is in the form of formal documents
- (specifications).

Waterfall Process Assumptions

- Requirements are known up front before design i.e. Users or management exactly know what they want.
- Requirements rarely change
- Design can be conducted in a purely abstract space, or trial rarely leads to error
- The technology will all fit nicely into place when the time comes.

Waterfall Process Limitations

- The proof of the concept is relegated to the very end of a long singular cycle. Before final integration, only documents have been produced.

- Late deployment hides many lurking risks:
  - technological (well, I *thought* they would work together...)
  - conceptual (well, I *thought* that's what they wanted...)
  - personnel (took so long, half the team left)
  - User doesn't get to *see* anything real until the very end, and they always hate it.
  - System Testing doesn't get involved until later in the process.

The Unified Process

- A standardized approach to analysis and design helps to ensure that all necessary tasks are understood and completed in software development

The Unified Process

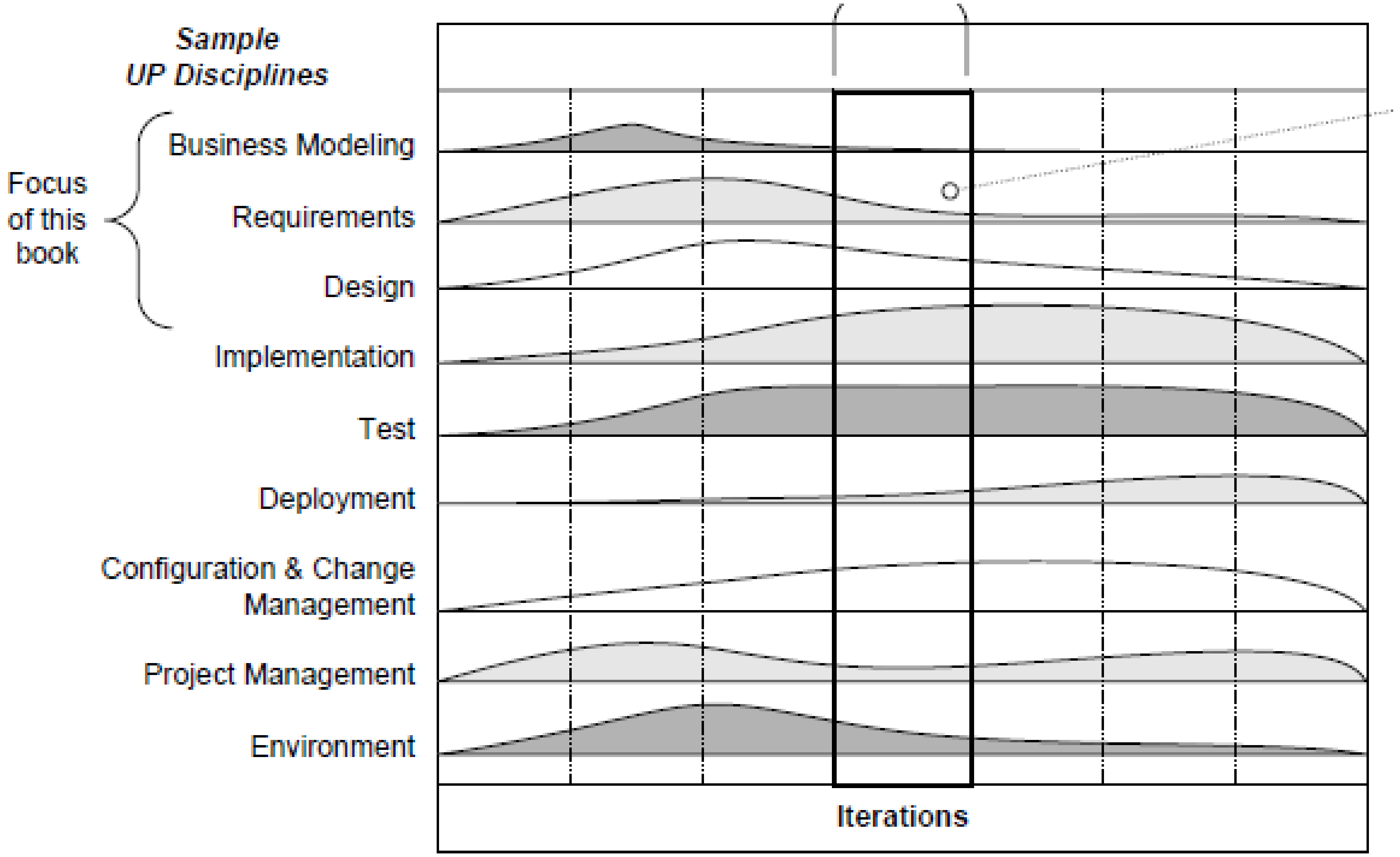
- The Unified Process has emerged as a popular and effective software development process.
- In particular, the Rational Unified Process, as modified at Rational Software, is widely practiced and adopted by industry.

The UP Disciplines (Workflows)

- Business Modeling
- Requirements
- Design

The UP Disciplines (Workflows)

- Business Modeling—The Domain Model artifact, to visualize noteworthy concepts in the application domain.
- Requirements—The Use-Case Model and Supplementary Specification artifacts to capture functional and non-functional requirements.
- Design—The Design Model artifact, to design the software objects.


Sample UP Disciplines
Focus of this book
Business Modeling
Requirements
Design
Implementation
Test
Deployment
Configuration & Change Management
Project Management
Environment
Iterations

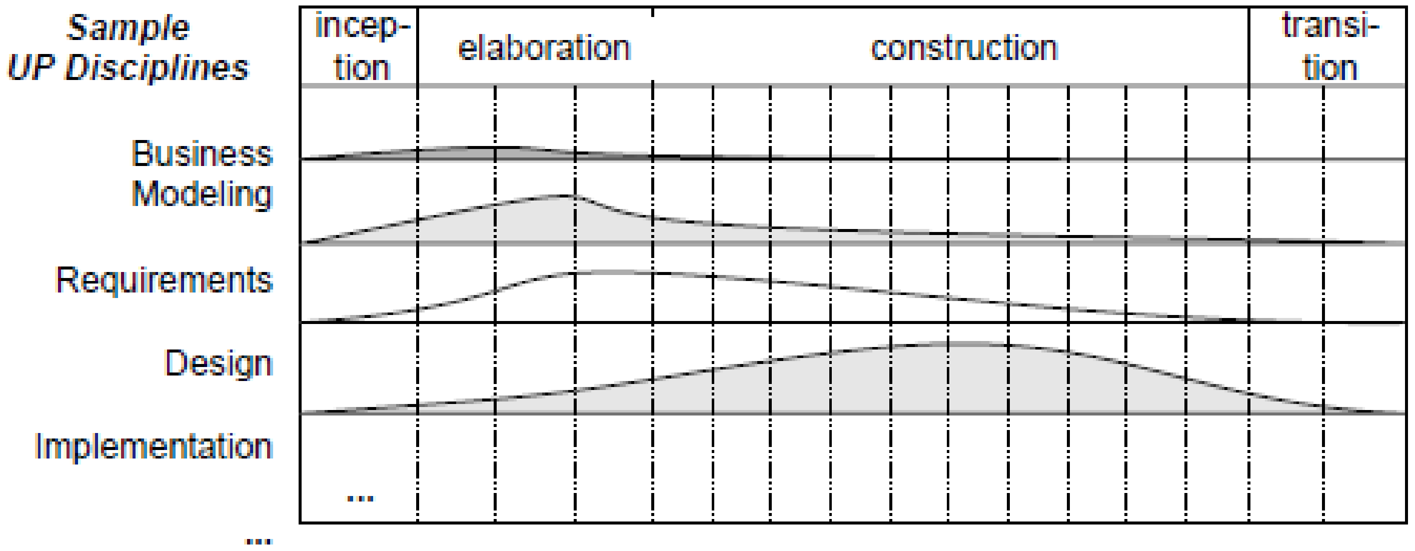


Figure 2.5 Disciplines and phases

Agile Modeling

- It focus on:
  - Minimize documentation
  - Accommodate changes
  - Iteration and time boxing
- Rational Unified Process is Agile Version of UP

Agile Methods

- Typically lightweight
  - WRT commitment to phases and documentation
  - Versus waterfall models which require “heavy” documentation of each phase before proceeding
- Flexible, Adaptable, Iterative
- Examples: RUP or UP, Extreme Programming (XP), Scrum

The Agile Manifesto

Manifesto

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

The Agile Principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter time scale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.

The Agile Principles

8. Agile processes promote sustainable development.

9. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

10. Continuous attention to technical excellence and good design enhances agility

11. Simplicity—the art of maximizing the amount of work not done—is essential.

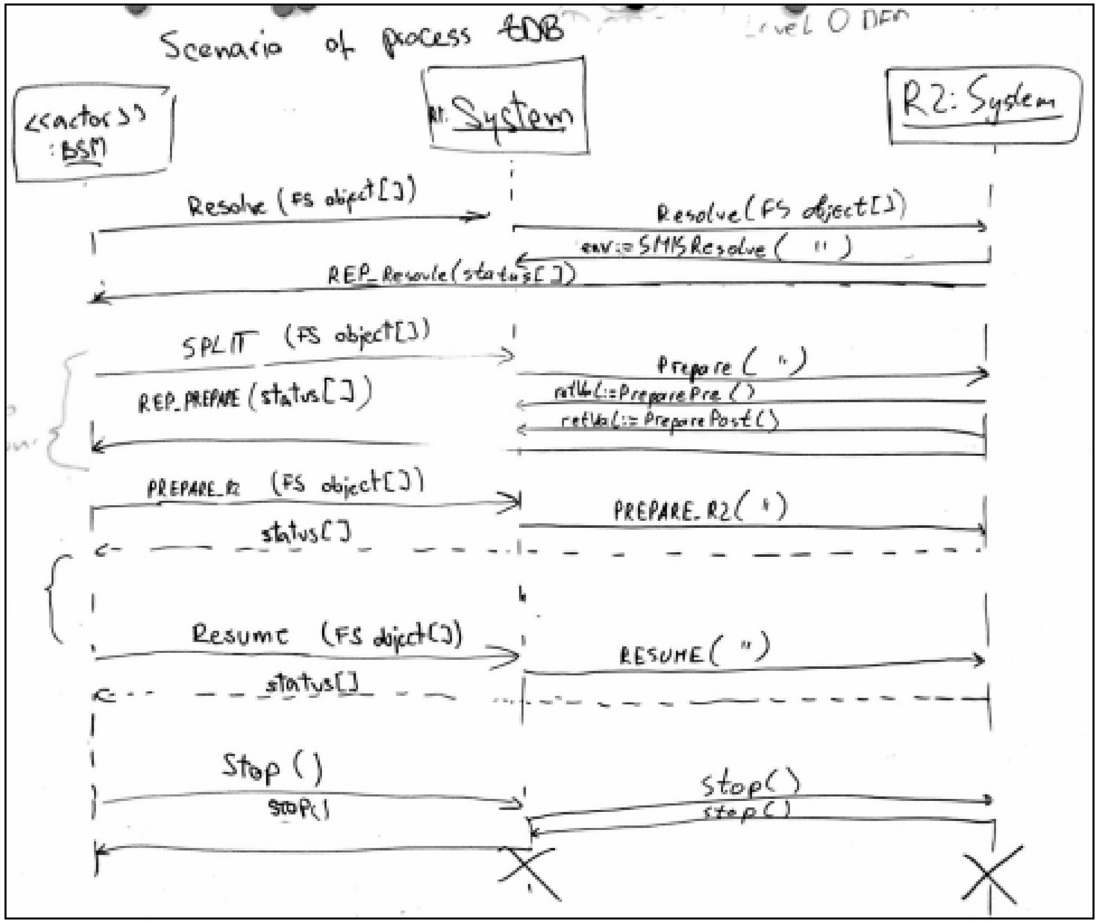
12. The best architectures, requirements, and designs emerge from self-organizing teams.

13. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

What is an Agile UP?

- The UP was not meant by its creators to be heavy or un-agile, although its large optional set of activities and artifacts have understandably led some to that impression. Rather, it was meant to be adopted and applied in the spirit of adaptability and lightness—an agile UP.

Agile UP-Example



The Most Important Concept

- The critical idea in the Rational Unified Process is *Iterative Development*.
- Iterative Development is successively enlarging and refining a system through multiple iterations, using feedback and adaptation.
    - Each iteration will include requirements, analysis, design, and implementation.
- Iterations are *timeboxed*.

What is Risk-Driven and Client-Driven Iterative Planning?

- The UP (and most new methods) encourage a combination of risk-driven and client-driven iterative planning.
- This means that the goals of the early iterations are chosen to

1) identify and drive down the highest risks

2) build visible features that the client cares most about

Why a new methodology?

- The philosophy of process-oriented methods is that the requirements of a project are completely frozen before the design and development process commences. As this approach is not always feasible, there is also a need for flexible, adaptable and agile methods, which allow the developers to make late changes in specifications.

What is Rational Unified Process (RUP)?

- RUP is a complete software-development process framework , developed by Rational Corporation.
- It's an iterative development methodology based upon six industry-proven best practices.
- Processes derived from RUP vary from lightweight—addressing the needs of small projects—to more comprehensive processes addressing the needs of large, possibly distributed project teams.

The Rational Unified Process

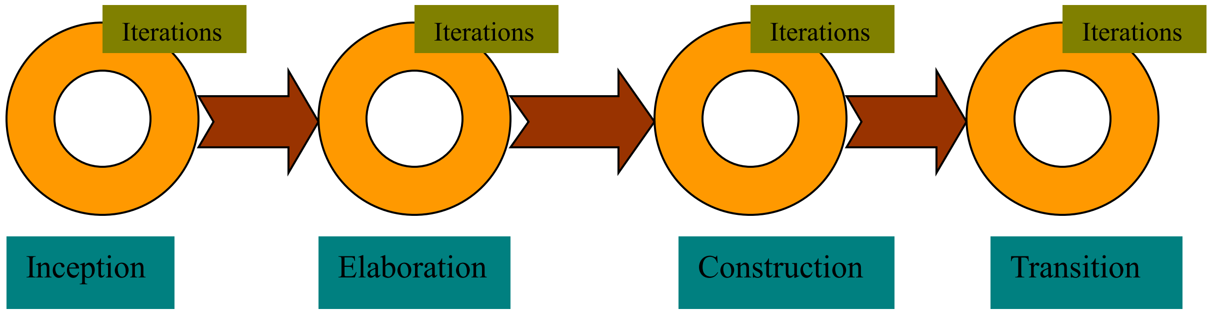
- RUP is a method of managing OO Software Development
- It can be viewed as a Software Development Framework which is extensible and features:
    - Iterative Development
    - Requirements Management
    - Component-Based Architectural Vision
    - Visual Modeling of Systems
    - Quality Management
    - Change Control Management

Phases in RUP

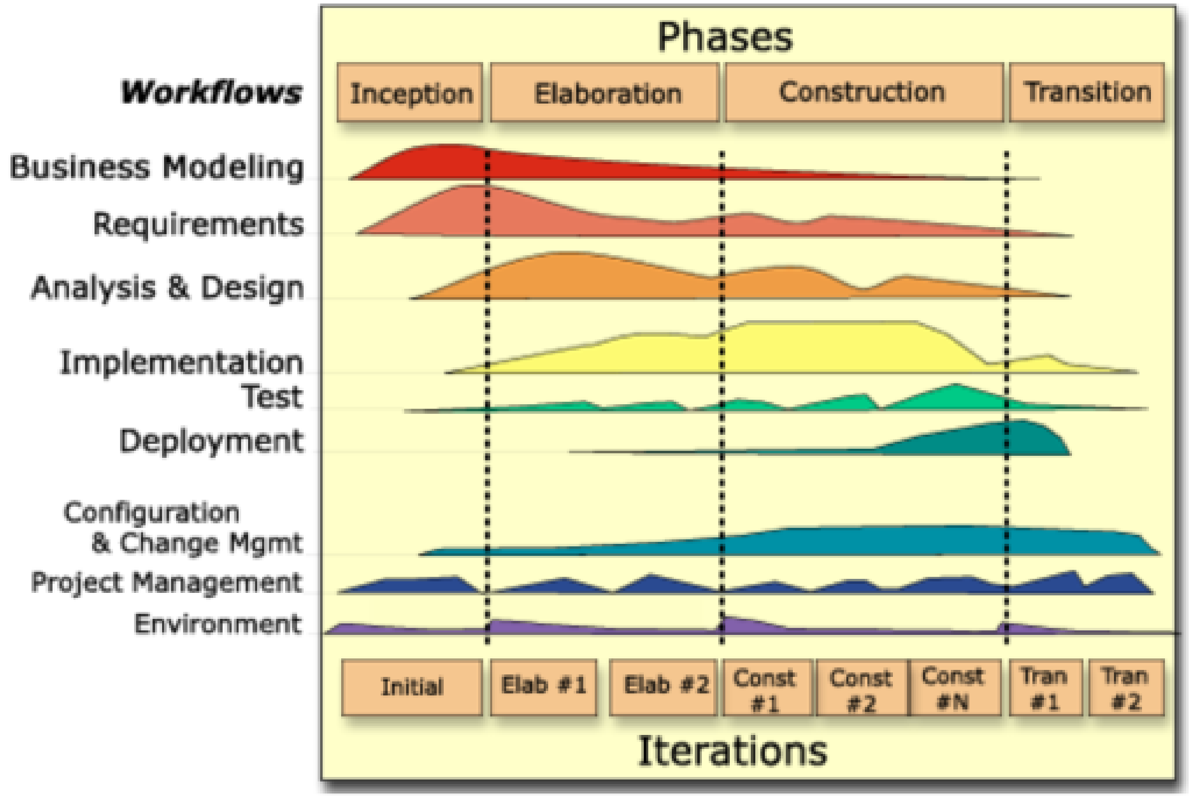
- RUP is divided into four phases, named:
  - Inception – "Daydream"
  - Elaboration – "Design/Details"
  - Construction – "Do it"
  - Transition – "Deploy it"
- Phases are not the classical requirements/ design/coding/implementation processes
- Phases iterate over many cycles

Iterations

Each phase has iterations, each having the purpose of producing a demonstrable piece of software. The duration of iteration may vary from two weeks or less up to six months.



How do traditional stages iterate?



Workflows look traditional, but they iterate in four phases

Unified Process best practices

- Get high risk and high value first
- Constant user feedback and engagement
- Early cohesive core architecture
- Test early, often, and realistically
- Apply use cases where needed
- Do some visual modeling with UML
- Manage requirements
- Manage change requests and configuration

Inception

- The life-cycle objectives of the project are stated, so that the needs of every stakeholder are considered. Scope and boundary conditions, acceptance criteria and some requirements are established.

(Present rough idea about software)

Inception - Activities

- **Formulate the scope of the project.**
  - Needs of every stakeholder, scope, boundary conditions and acceptance criteria established.
- **Plan and prepare the business case.**
  - Define risk mitigation strategy, develop an initial project plan and identify known cost, schedule, and profitability trade-offs.
- **Synthesize candidate architecture.**
  - Candidate architecture is picked from various potential architectures
- **Prepare the project environment.**

Inception - Exit criteria

- An initial business case containing at least a clear formulation of the product vision - the core requirements - in terms of functionality, scope, performance, capacity, technology base.
- Success criteria (example: revenue projection).
- An initial risk assessment.
- An estimate of the resources required to complete the elaboration phase.

Elaboration

- **An analysis is done to determine the risks, stability of vision of what the product is to become, stability of architecture and expenditure of resources.**

(Detail Requirements, Design)

Elaboration - Entry criteria

- The products and artifacts described in the exit criteria of the previous phase.

- The plan approved by the project management, and funding authority, and the resources required for the elaboration phase have been allocated.

Elaboration - Activities

- **Define the architecture.**

Project plan is defined. The process, infrastructure and development environment are described.

- **Validate the architecture.**
- **Baseline the architecture.**

To provide a stable basis for the bulk of the design and implementation effort in the construction phase.

Elaboration - Exit criteria

- A detailed software development plan, with an updated risk assessment, a management plan, a staffing plan, a phase plan showing the number and contents of the iteration , an iteration plan, and a test plan
- The development environment and other tools
- A baseline vision, in the form of a set of evaluation criteria for the final product
- A domain analysis model, sufficient to be able to call the corresponding architecture ‘complete’.
- An executable architecture baseline.

Construction

- The Construction phase is a manufacturing process. It emphasizes managing resources and controlling operations to optimize costs, schedules and quality. This phase is broken into several iterations.

(Coding or Development)

Construction - Entry criteria

- The product and artifacts of the previous iteration. The iteration plan must state the iteration specific goals
- Risks being mitigated during this iteration.
- Defects being fixed during the iteration.

Construction - Activities

- **Develop and test components**.

Components required satisfying the use cases, scenarios, and other functionality for the iteration are built. Unit and integration tests are done on Components.

- **Manage resources and control process**.
- **Assess the iteration**

Satisfaction of the goal of iteration is determined.

Construction - Exit Criteria

- The same products and artifacts, updated, plus:
- A release description document, which captures the results of an iteration
- Test cases and results of the tests conducted on the products,
- An iteration plan, detailing the next iteration
- Objective measurable evaluation criteria for assessing the results of the next iteration(s).

Transition

- The transition phase is the phase where the product is put in the hands of its end users. It involves issues of marketing, packaging, installing, configuring, supporting the user-community, making corrections, etc.

Transition - Activities

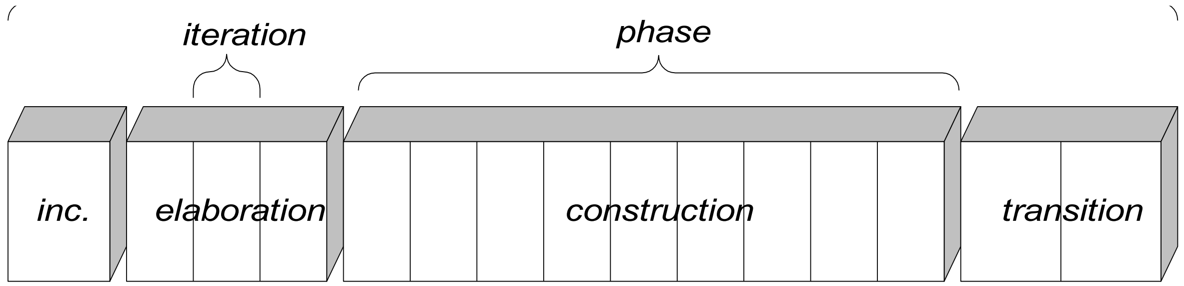
- Test the product deliverable in a customer environment.
- Fine tune the product based upon customer feedback
- Deliver the final product to the end user
- Finalize end-user support material

Transition - Exit criteria

- An update of some of the previous documents, as necessary, the plan being replaced by a “post-mortem” analysis of the performance of the project relative to its original and revised success criteria;
- A brief inventory of the organization’s new assets as a result this cycle.

UP phases are iterative & incremental

- Inception
  - Feasibility phase and approximate vision
- Elaboration
  - Core architecture implementation, high risk resolution
- Construction
  - Implementation of remaining elements
- Transition
  - Beta tests, deployment



Advantages of RUP

- The RUP puts an emphasis on addressing very early high risks areas.
- It does not assume a fixed set of firm requirements at the inception of the project, but allows to refine the requirements as the project evolves.
- It does not put either a strong focus on documents
- The main focus remains the software product itself, and its quality.

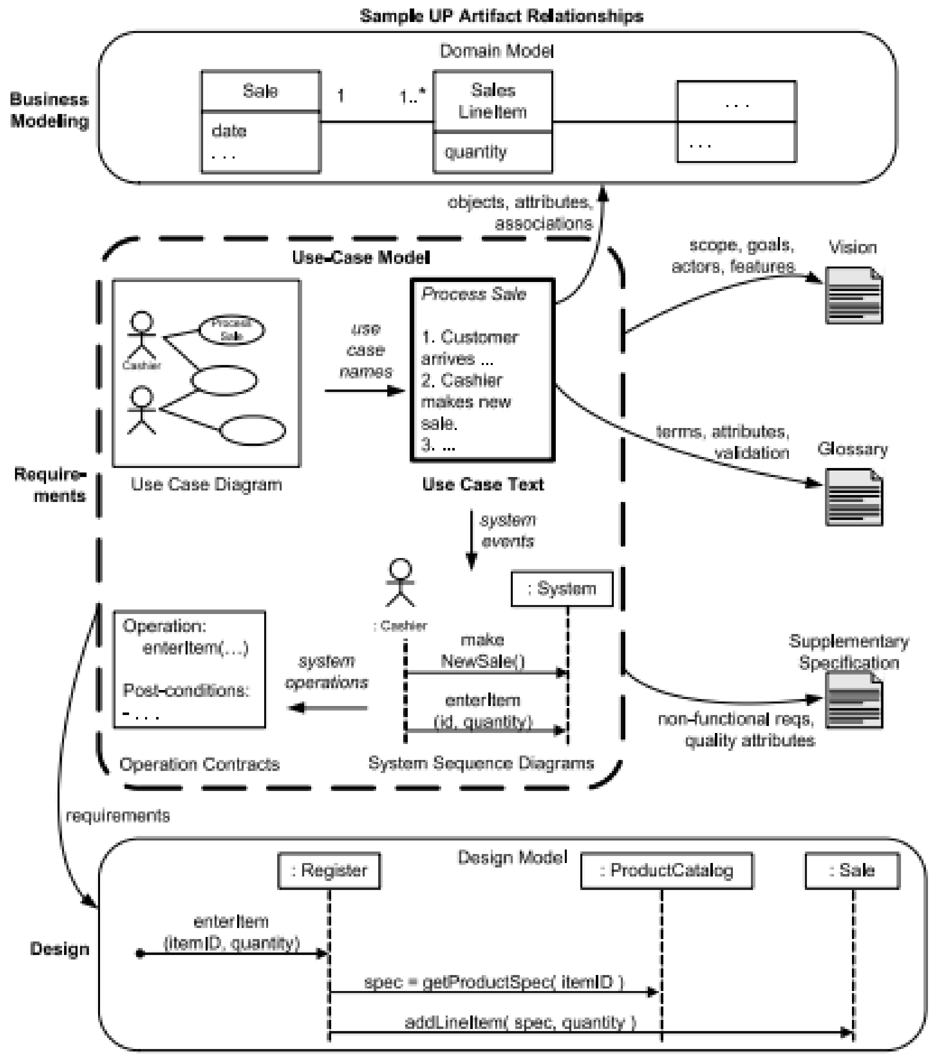
Drawbacks of RUP

- RUP is not considered particularly “agile” However, recent studies have shown that by adopting the right essential artifacts RUP is agile.
- It fails to provide any clear implementation guidelines.
- RUP leaves the tailoring to the user entirely.

UP artifacts

- The UP describes work activities, which result in *work products* called *artifacts*
- Examples of artifacts:
  - Vision, scope and business case descriptions
  - Use cases (describe scenarios for user-system interactions)
  - UML diagrams for domain modeling, system modeling
  - Source code (and source code documentation)
  - Web graphics
  - Database schema

UP Artifacts Example


Sample UP Artifact Relationships
Business Modeling
Domain Model
Sale
date
. . .
1
1..*
Sales LineItem
quantity
. . .
. . .
objects, attributes, associations
Use-Case Model
Process Sale
Cashier
Use Case Diagram
use case names
Process Sale
1. Customer arrives ...
2. Cashier makes new sale.
3. ...
Use Case Text
scope, goals, actors, features
Vision
terms, attributes, validation
Glossary
Require-ments
system events
: Cashier
: System
make NewSale()
enterItem (id, quantity)
system operations
Operation: enterItem(…)
Post-conditions:
- . . .
Operation Contracts
System Sequence Diagrams
Supplementary Specification
non-functional reqs, quality attributes
requirements
Design
Design Model
: Register
: ProductCatalog
: Sale
enterItem (itemID, quantity)
spec = getProductSpec(itemID)
addLineItem(spec, quantity)