Review

- Introduction to SE
  - Three Elements
    - Process
    - Methodology
    - Tool

Chapter 2
Software Processes

Objectives

- To introduce software process models
- To describe two generic process models and when they may be used
- To explain the Rational Unified Process model
- To introduce CASE technology to support software process activities

Topics covered

- Software process models
- Process activities
- The Rational Unified Process
- Computer-aided software engineering

The software process

A process defines **Who** is doing **What**, **When**, and **How**, in order to reach a certain goal.

The software process (continued)

- A structured set of activities required to develop a software system.
- There are many software processes, but some fundamental activities are common to all software processes:
  - business modelling
  - requirement modelling
  - analysis modelling
  - design modelling and implementation(coding)
  - test
  - maintenance……

Generic software process models

- A software process model is an abstraction representation of a process
  - The waterfall model (software life cycle)
    - Separate and distinct phases of different activities
  - The Iterative model
    - business, requirement, analysis and design are interleaved.

Waterfall model phases

- business → requirement →analysis → design

final version

- The main drawback of the waterfall model is the difficulty of accommodating change after the process is underway. One phase has to be complete before moving onto the next phase.[1]

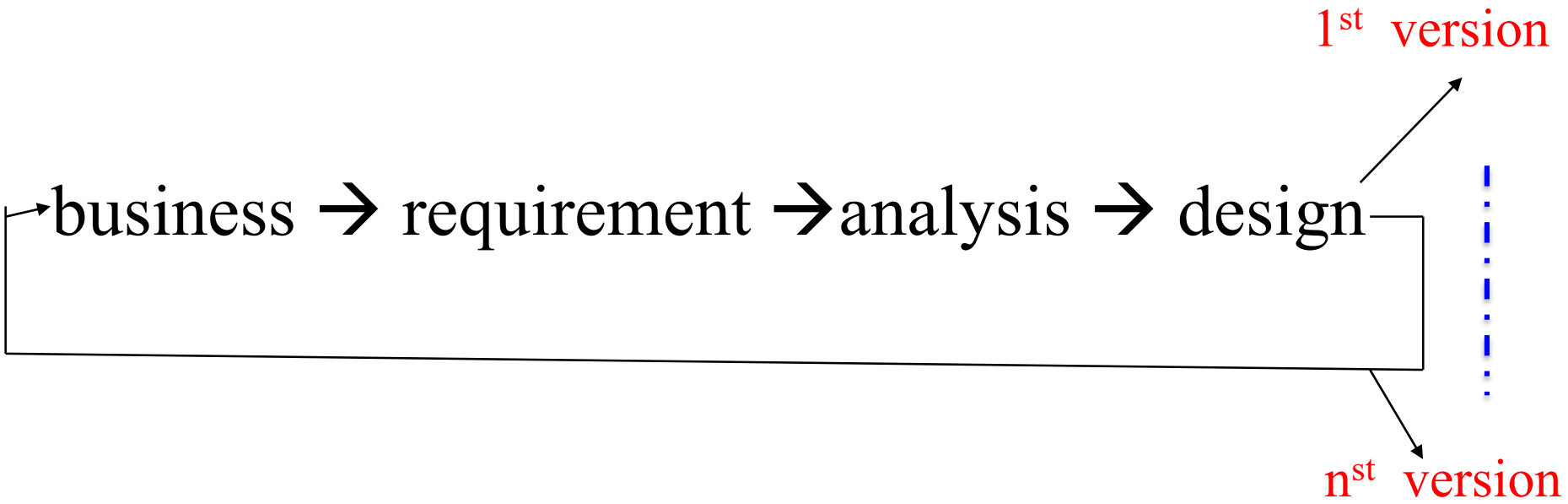
Waterfall model problems

- Key false assumption[1]
  - requirement is stable
- IKIWISI effect
  - I'll know it when I see it.
- resultant product
  - deviate from user's real requirements

Waterfall model problems (continued)

- Inflexible partitioning of the project into distinct stages makes it difficult to respond to changing customer requirements.
- Therefore, this model is only appropriate when the requirements are well-understood and changes will be fairly limited during the design process.
- Few business systems have stable requirements.

Iterative development (try marriage)



2/8 principle ,先做最重要、最难的事情[1]

Iterative development (continued)

- based on the idea of developing an initial version as soon as possible
- Exposing the versions to user for feedback
  - IKIWISI effect
- Refining the version by user's feedback
- Iterative many versions until an adequate system has been developed.

Process activities are interleaved rather than separate, with rapid feedback across activities.

Iterative development versus waterfall

For a large system, a mixed process that utilizes each the best features:

- Developing a throwaway prototype using an Iterative approach to resolve uncertainties in the system specification
- Developing the well understood parts of the system using waterfall-based process.

Process activities

- business modeling
- requirement modeling
- analysis modeling
- design and implementation
- test
- maintenance
- ……

Business modeling

- purpose
  - get correct requirements[1]
- content
  - examine organization
    - individual
    - organization
    - an unit of an organization
    - group of people …….
  - current workflow and what's problems exist
  - with the new system, the new workflow

Requirement modeling

- The process of establishing what services are required and the constraints on the system's operation and development.
- Requirements sub-process
  - Feasibility study
  - Requirements elicitation
  - Requirements specification
  - Requirements validation

Feasibility study

- Mainly three aspects
  - Technology
  - Cost
  - Social etc.
- The result should inform the decision of whether to go ahead or stop
- Should be relatively cheap and quick

Requirements elicitation

- Elicitation
  - business modeling results
    - system use case model
  - communication for requirements specification

Requirements specification

- Document for
  - view
    - Customer or end-user oriented
  - model
    - Software developer oriented
  - view vs. model[1]

Requirements validation

- Check for
  - Realism
  - Consistency
  - Completeness
- Correct the problems discovered in the activity

Requirements analysis

- Analysis[1]
  - What computer[2] should do to reach requirements and stakeholder's goal.
    - PO
    - OO

Review

Software design and implementation

- Software design
  - Design a software structure that realises the requirements;
  - Decision, strategy
- Implementation
  - Translate this structure into an executable program;
- The activities of design and implementation are closely related and may be interleaved.

Design process activities

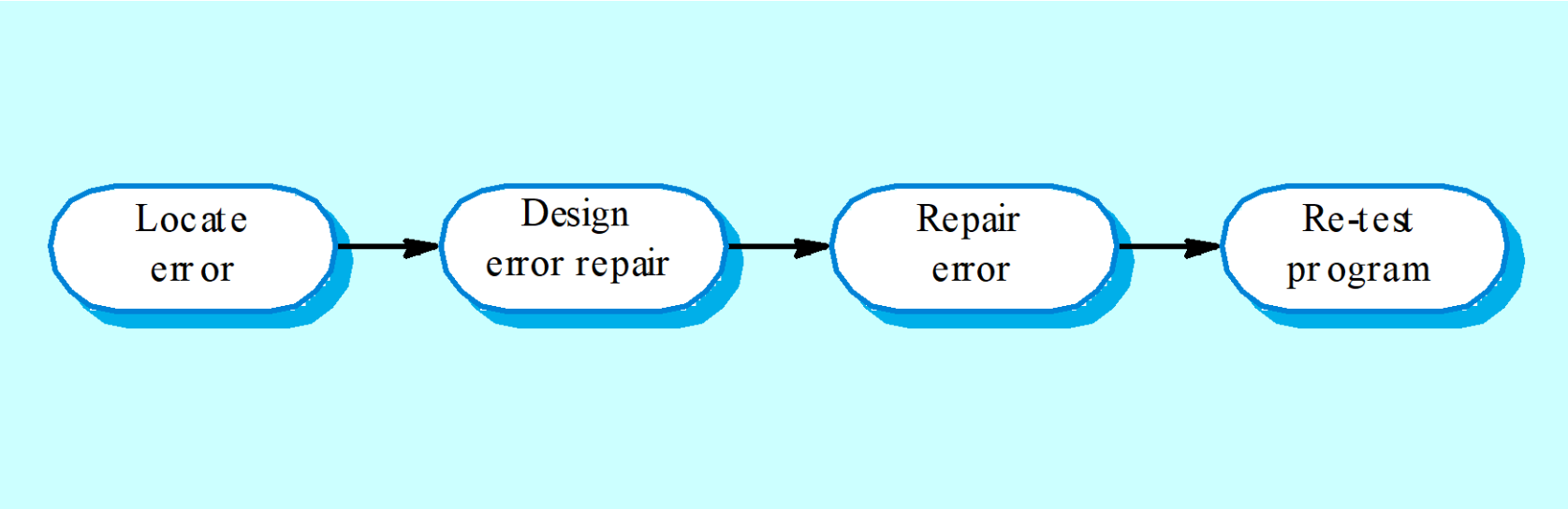
- Architectural design
- Interface design
- Component design
- Data structure design
- Algorithm design

Practical process may adapt it in different ways.

Implementation- Programming and debugging

- Translating a design into a program and removing errors from that program.
- Programming is a personal activity - there is no generic programming process.
- Programmers carry out some program testing to discover faults in the program and remove these faults in the debugging process.

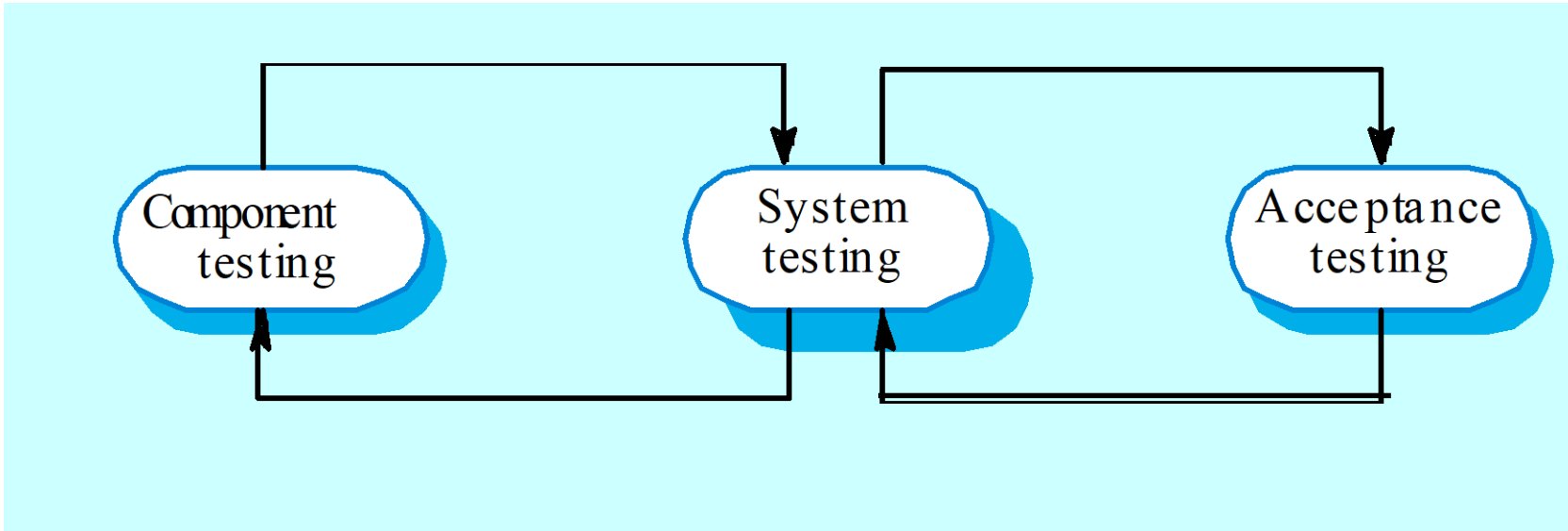
The debugging process



test

- System testing involves executing the system with test cases that are derived from the specification of the real data to be processed by the system.

The testing process



Testing stages

- Component or unit testing
  - Individual components are tested independently;
  - Components may be functions or objects or coherent groupings of these entities.
- System testing
  - Testing of the system as a whole. Testing of emergent properties is particularly important.
- Acceptance testing
  - Testing with customer data to check that the system meets the customer's needs.

Testing stages

- Acceptance testing : alpha testing
- Beta testing: system is to be marketed as a software product. Delivering a system to a number of potential customers to test.

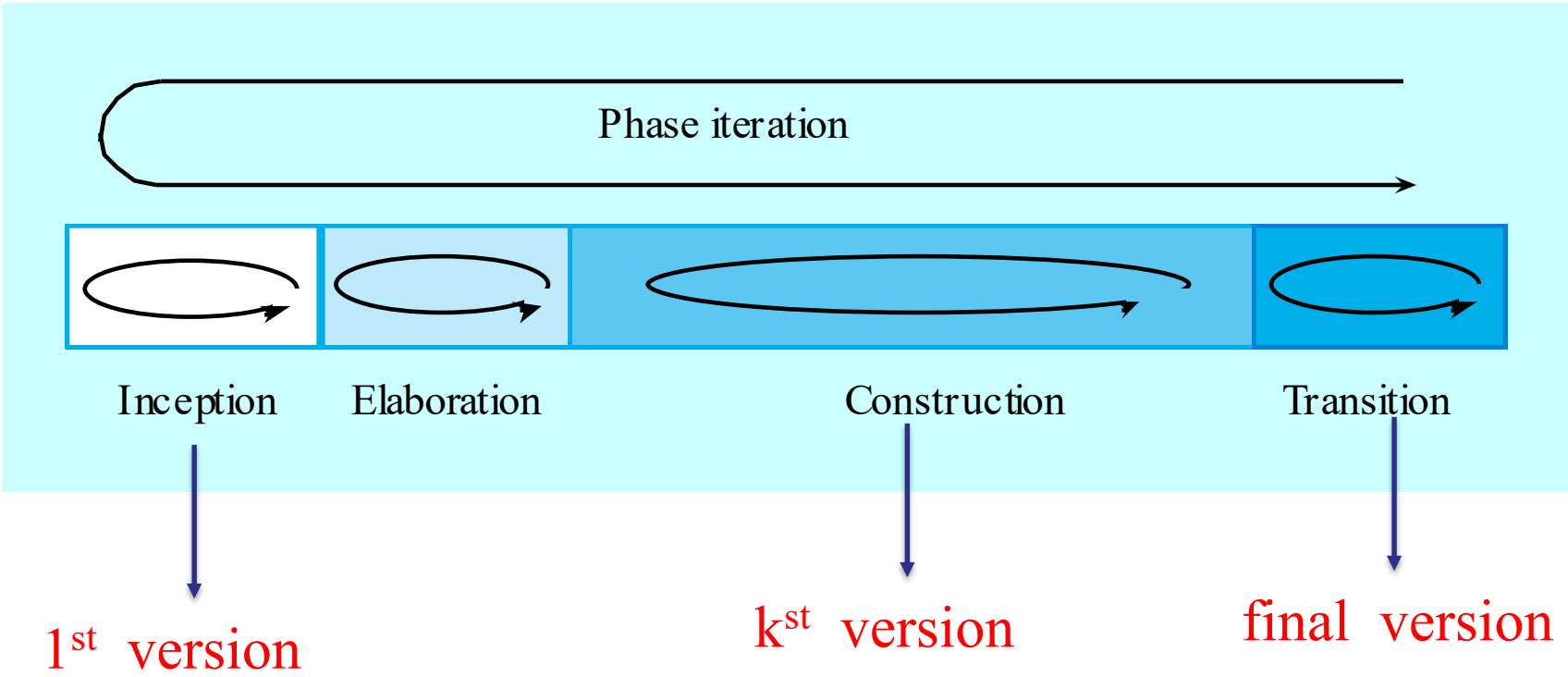
maintenance

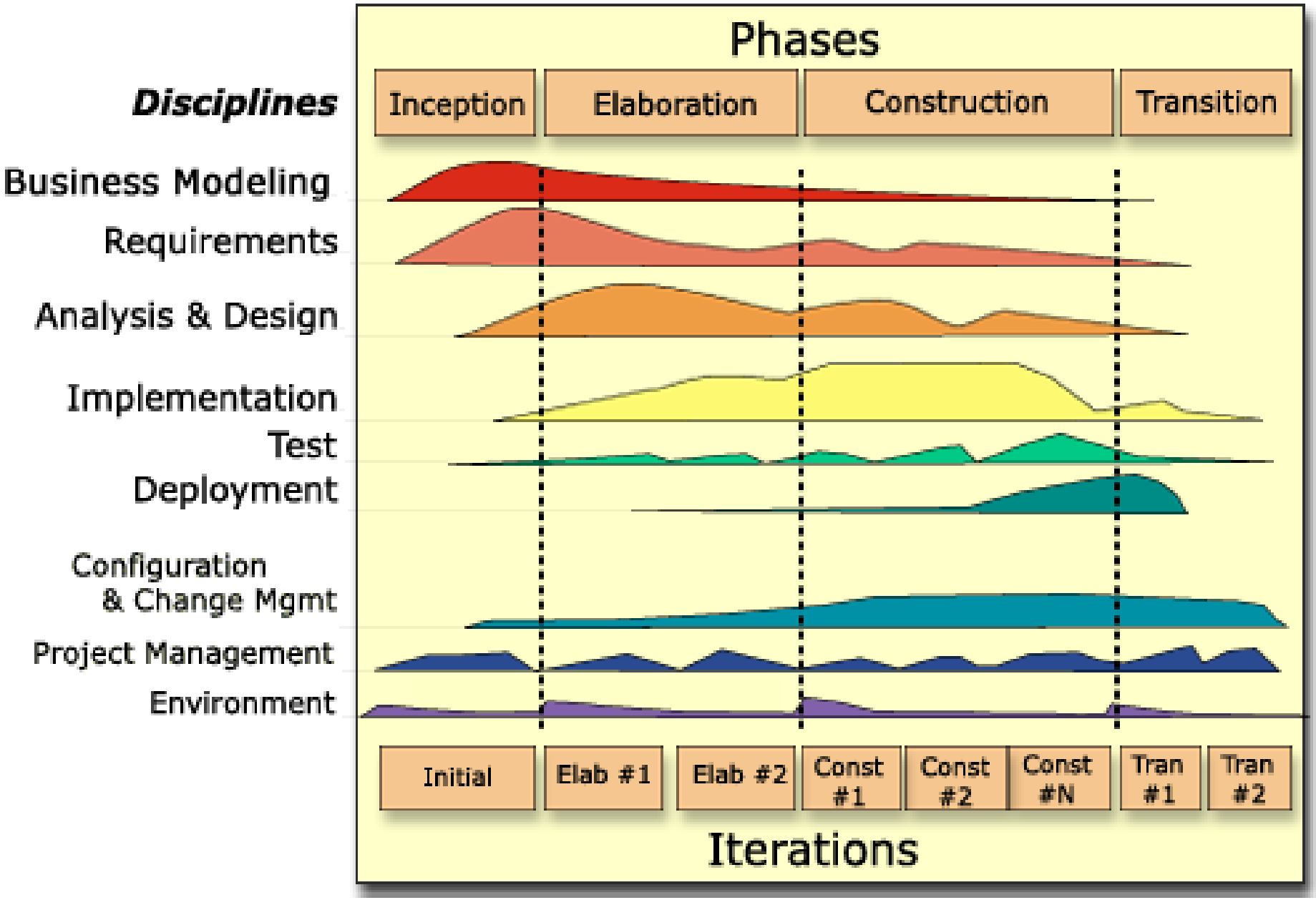
- Software is inherently flexible and can change.
- As requirements change through changing business circumstances, the software that supports the business must also evolve and change.

The Rational Unified Process

- IBM Rational
- A hybrid process model: best practices
- Modern: OO based
- A process model derived from the work on the UML and associated process.
- Normally described from 3 perspectives
  - A dynamic perspective that shows phases over time;
  - A static perspective that shows process activities;
  - A practice perspective that suggests good practice.

RUP phase model




Phases
Disciplines
Inception
Elaboration
Construction
Transition
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration
& Change Mgmt
Project Management
Environment
Initial
Elab #1
Elab #2
Const
#1
Const
#2
Const
#N
Tran
#1
Tran
#2
Iterations

RUP phases

- Inception: focus on feasibility
  - Establish the business case for the system.
- Elaboration: focus on requirements
  - Develop an understanding of the problem domain and the system architecture.
- Construction: focus on design
  - System design, programming and testing.
- Transition
  - Deploy the system in its operating environment.

RUP good practice

- Develop software iteratively (r priority)
  - The most difficult part has top priority(先做最难做、着重要的事情）
    - 2/8 law
- Manage requirements
- Use component-based architectures
- Visually model software
- Verify software quality
- Control changes to software

Computer-aided software engineering

- Computer-aided software engineering (CASE) is software to support software development and evolution processes.
- Activity automation
  - Graphical editors for system model development;
  - Data dictionary to manage design entities;
  - Graphical UI builder for user interface construction;
  - Debuggers to support program fault finding;
  - Automated translators to generate new versions of a program.

CASE technology

- CASE technology has led to significant improvements in the software process. However, these are not the order of magnitude improvements that were once predicted
  - Software engineering requires creative thought - this is not readily automated;
  - Software engineering is a team activity and, for large projects, much time is spent in team interactions. CASE technology does not really support these.

CASE classification

- Classification helps us understand the different types of CASE tools and their support for process activities.
- Functional perspective
  - Tools are classified according to their specific function.
- Process perspective
  - Tools are classified according to process activities that are supported.
- Integration perspective
  - Tools are classified according to their organisation into integrated units.

CASE integration

- Tools
  - Support individual process tasks such as design consistency checking, text editing, etc.
- Workbenches
  - Support a process phase such as specification or design, Normally include a number of integrated tools.
- Environments
  - Support all or a substantial part of an entire software process. Normally include several integrated workbenches.

Note

- CASE helps professionals productive but turns a novice into a novice.

  such as computer mouse.

summary

- process models
  - waterfall
  - iterative
- process activities
  - BM
  - RM
  - AM
  - DM
  - CM