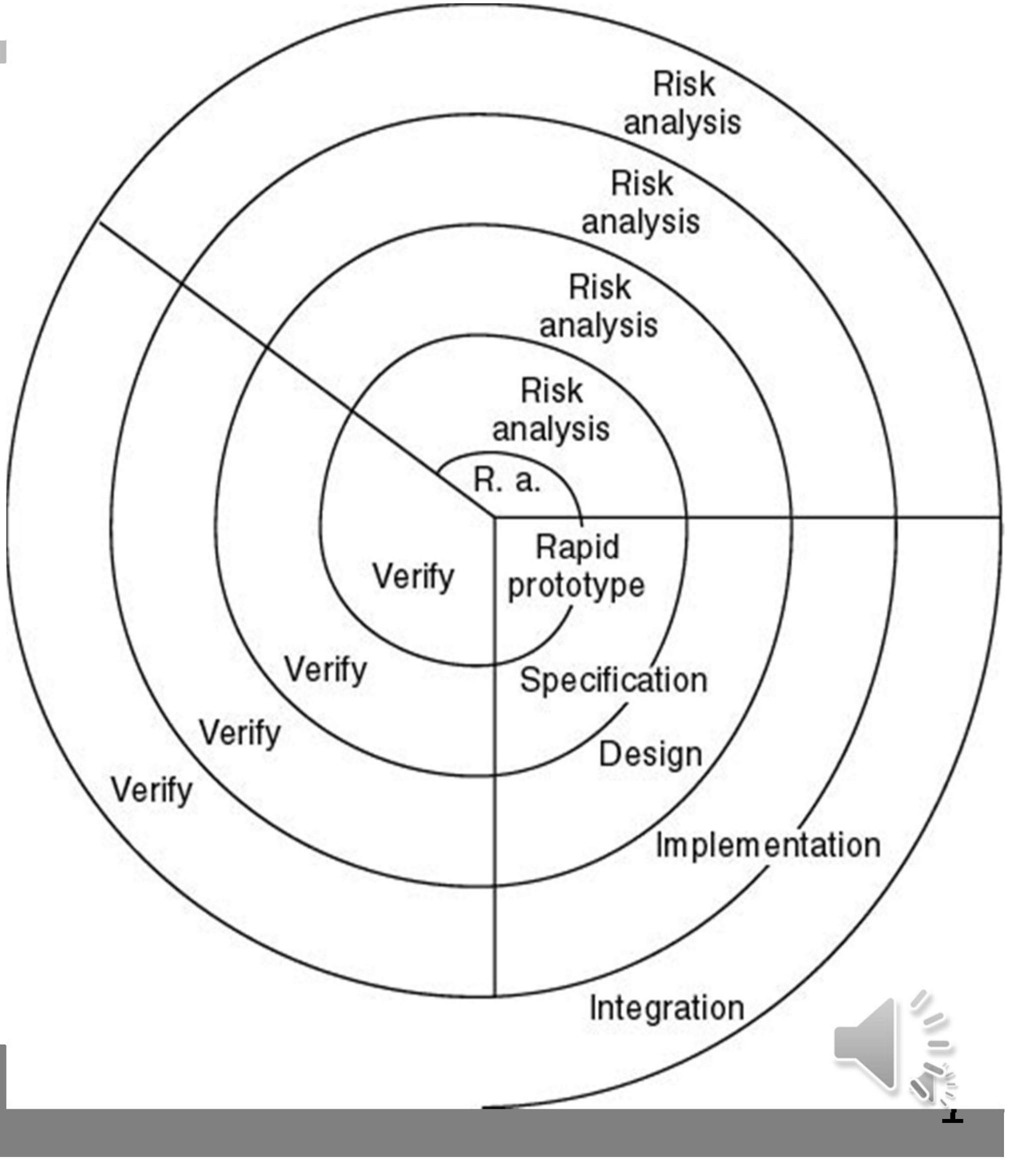
1.4 Iterative Development Processes

- Spiral Model [Boehm, 1988]
  - first iterative software development process



1.4 Iterative Development Processes

- Booch's iterative oo development process [Micro Processes]
  - 1) identifying the classes
  - 2) identifying the semantics (attributes and behaviors of the classes)
  - 3) identifying the relationships among the classes
  - 4) defining the class interface
  - 5) implementing the classes

  - Grady booch (부치) RATIONAL 소프트웨어 수석과학자. UML 개발자

1.4 Iterative Development Processes

- Booch's iterative oo development process[Macro Processes]
  - to serve as the controlling framework of the micro process.
  - analysis, modeling, design, implementation, maintenance
  - ➔ RUP(Rational Unified Process) – complete
  - ➔ XP(Extreme Programming) – lightweight..

1.4.1 Object Oriented Development Activities

- 1) Conceptualization
  - to establish [redacted] and [redacted] requirements of the software system.
  - Establishing the complete requirements of the system.

- 2) Object-Oriented analysis and modeling
  - to build models of the system's desired behavior, using ex.UML
  - use cases and class diagrams

1.4.1 Object Oriented Development Activities

- 3) Object-oriented design
  - to create [redacted] for implementation
  - in terms of objects, classes, the relationships among them.
  - key concern of OOD
    - 1) satisfy all the stated requirements and constraints
    - 2) flexible for future changes and enhancements
    - 3) feasible for implementation , can it be implemented efficiently ?

1.4.1 Object Oriented Development Activities

- 4) Implementation
  - using OOPL(ex. Java)
  - coding, unit testing, debugging
  - key issues
    - 1) correct?
    - 2) efficient and maintainable?
    - 3) robust ? (capable of tolerating faults and recovering from failures?)

1.4.1 Object Oriented Development Activities

- 5) Maintenance
  - to manage post delivery evolution
  - removing bugs
  - enhancing functionalities
  - adapting to evolving needs and environments

1.4.1 Object Oriented Development Activities

- Iterative Development Processes
  - try to facilitate and manage [redacted]
  - 1) Each iteration is relative small and can be completed in a relative short period of time
  - 2) Each iteration results in a release of an [redacted] product or component, which is a part of the final product.

1.4.2 Rational Unified Process

- RUP
  - Complete Software Engineering Process
  - Provides guidelines for every phase
  - Goal : ensure the production of high quality software that meets the needs of its end users within a predictable schedule and budget.
  - not one process
  - but a process framework that can be adapted and extended to different organizations and projects

- RUP
  - IBM의 래셔널 소프트웨어 부서에서 만든 객체 지향 개발 방법론
  - RUP는 하나로 고정되어 쓰인 프로세스가 아니라, 적응이 가능한 프로세스 프레임워크
  - 개발 조직과 소프트웨어 프로젝트 팀이 필요한 바에 따라서 프로세스의 요소들을 선택하여 조절할 수 있도록 설계됨
  - 래셔널 소프트웨어사에서 개발
  - IBM에 2003년 2월에 합병
  - 샘플 산출물과 다양한 활동에 대한 자세한 설명을 바탕으로 한 서로 연결된 지식-베이스를 포함
  - RUP는 사용자가 쉽게 개발 과정을 수정할 수 있는 IBM Rational Method Composer (RMC) 라는 제품에 포함되어 있음

1.4.2 Rational Unified Process

- The Key Practices of the RUP
  - 1) develop software iteratively
  - 2) systematically elicit, organize, and manage changing requirements.
  - 3) use component-based architecture
  - 4) visually model software using UML
  - 5) continuously verify software quality
  - 6) control changes to software

1.4.2 Rational Unified Process

- Emphasis of RUP is On Building models rather than paper documents.
- 9 models (collectively cover all the important decisions)
  - 1) Business Model : Establishes an abstraction of the organization
  - 2) Domain Model : Establishes the context of the system.
  - 3) Use Case Model : Establishes the system's [redacted] requirements
  - 4) analysis model(optional) : an idea design

1.4.2 Rational Unified Process

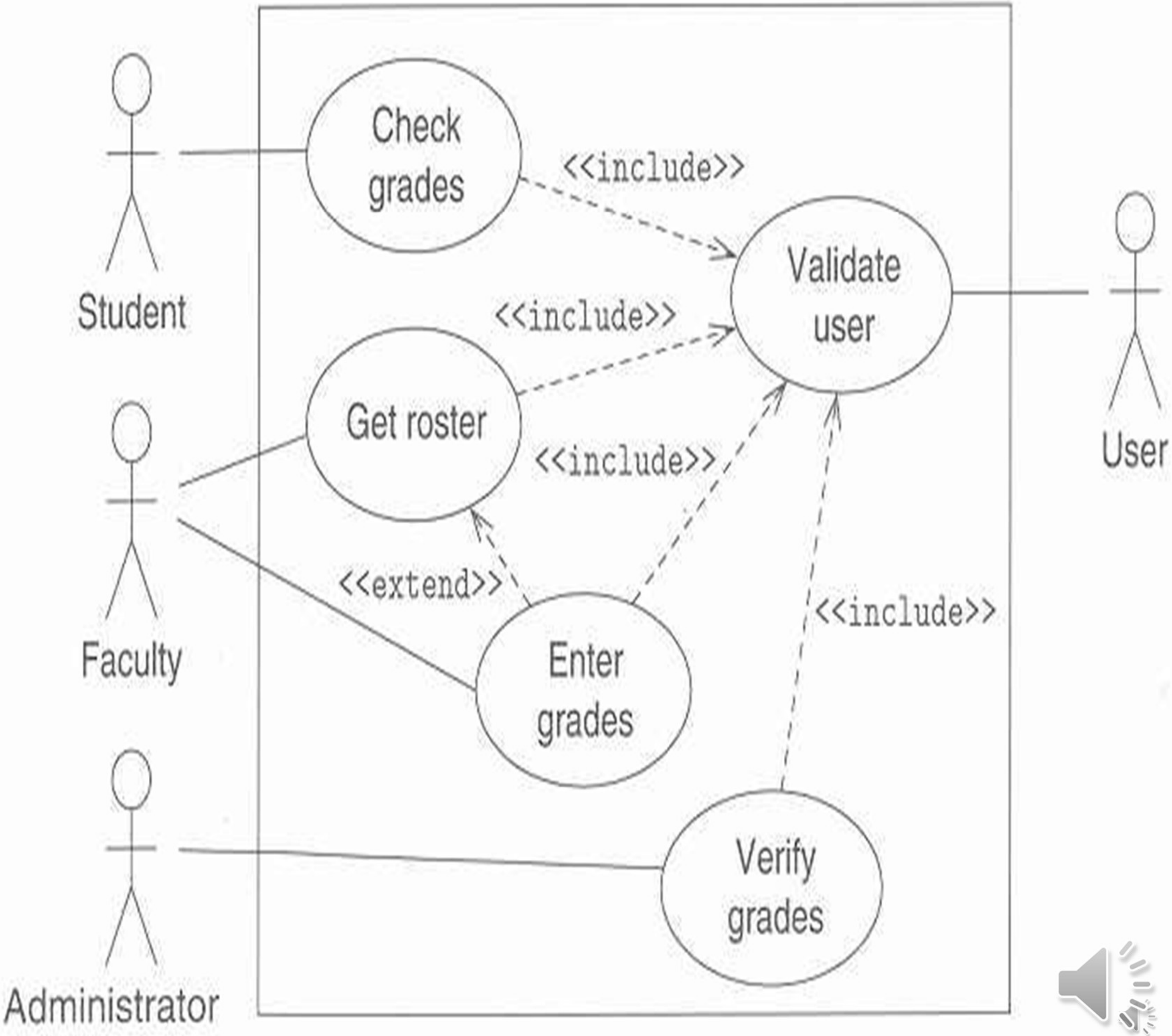
- 5) Design Model : establishes the vocabulary of the problem and its solution.
- 6) Process Model(optional) : establishes the system's concurrency and synchronization mechanisms
- 7) Deployment Model : the hardware topology on which the system is executed
- 8) implementation model : establishes the parts used to assemble and release the physical system
- 9) Test Model : establishes the paths by which the system is validated and verified.

1.4.2 Rational Unified Process

- RUP
  - use case driven
    - use cases
      - defined for system requirements
      - the foundation for all other development activities, including design, implementation and testing
  - architecture centric
    - the main focus of early iteration of the development process is to produce and validate an executable architecture prototype

Figure 2.17

Dependency relationships among use cases.



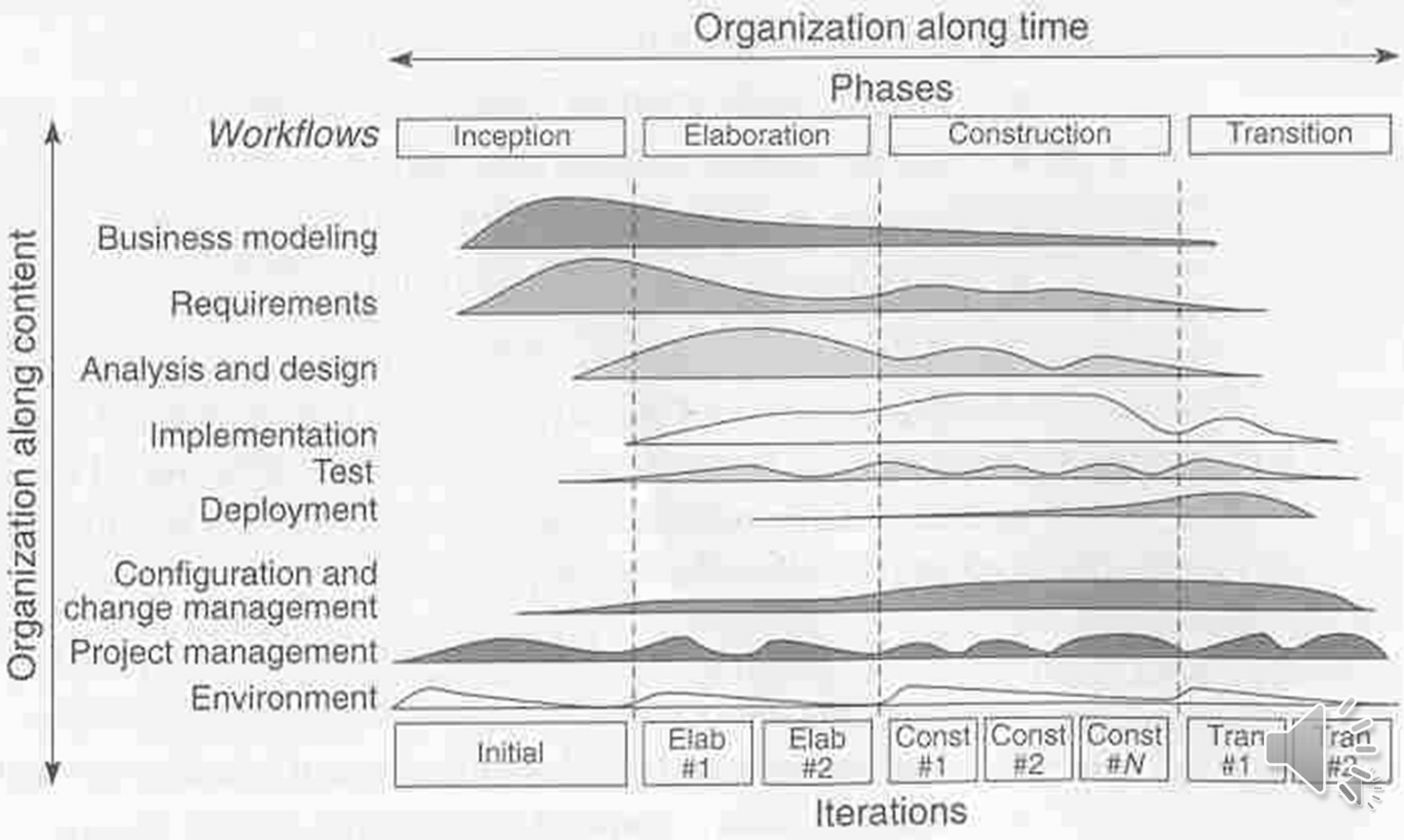
1.4.2 Rational Unified Process

- Process structure of the RUP

Figure 1.2

Rational Unified Process. (From Kruchten [2000] The Rational Unified Process, An Introduction. Addison-Wesley.)

✓



1.4.2 Rational Unified Process

- 1) First Dimension : Workflow
  - a workflow
    - consists of a sequence of activities that produce a set of artifacts, or deliverables, which can be project plans,design models, source code, tests and documentations
  - Nine process workflow
    - 1) business modeling : the structure and dynamics of the organization
    - 2) Requirements : the use case-based method for eliciting requirements
    - 3) Analysis and design : the multiple architectural views.

1.4.2 Rational Unified Process

- 4) Implementation
- 5) Test : test cases, procedures, and defect-tracking metrics
- 6) Deployment : all the deliverable system configurations
- 7) Configuration management
- 8) Project management
- 9) Environment : covers the necessary infrastructure required to develop a system.

1.4.2 Rational Unified Process

- 2) Second Dimension : Phases and iterations
  - four major Phases
    - 1) Inception : Establishes the Business case for the project
    - 2) Elaboration : Establishes a project plan and a sound architecture
    - 3) Construction : grows the system.
    - 4) Transition : supplies the system to its end users
  - more iterations
    - iterations in different phases have different emphases on process workflows.

1.4.3 Extreme Programming

- Extreme Programming
  - lightweight process for producing high-quality executable code throughout the development process.
  - focuses on [redacted] from the very beginning
  - iterative process with small iteration
    - each iteration : a few days, a few weeks
    - first iteration : produce a minimum, skeletal, and executable implementation
    - focus of each iteration : [redacted] or [redacted]

1.4.3 Extreme Programming

- emphasizes maintaining high quality in the code delivered by each and every iteration
  - 1) enhancements : new functionalities or features
  - 2) refactoring [redacted] the code to improve the quality , including extensibility and maintainability, and the structure of the software system (p. 252 – 255)

1.4.3 Extreme Programming(*)

- Step of Extreme Programming
  - 1) Development Team determines the various features(stories)
  - 2) for each such features, Team informs the clients how long & how cost to implement
  - 3) the clients selects the features using cost-benefit analysis
  - 4) the proposed build is broken down into smaller pieces(tasks)
  - 5) A programmer first draws up test cases for a task

1.4.3 Extreme Programming(*)

- 6) working with a partner on one screen, the programmer implements the tasks
- 7) test all the test cases
- 8) the task is integrated into the current version of the product
- (all members of the XP team work on specifications, design, code and testing.)

1.4.3 Extreme Programming

- Key Concepts of XP
  - Planning Game (start with a simple a plan for each <u>iteration</u>, and continually <u>refine the plan</u> as necessary)
  - Frequent and small releases
  - use Metaphor (with the customers)
  - Simple design([redacted] later if changes are necessary).
  - [redacted] First (write unit test before writing code)
  - Refactoring (refactor to make the system simpler and clearer or to <u>reduce duplication</u>) p 253.
  - [redacted](write all production code in pairs)

1.4.3 Extreme Programming

- [redacted] (Anyone may change code anywhere)
- Continuous integration
- 40-hour week
- [redacted] (have a customer available on-site and full time)
- Coding Standards (adopt common standards and conventions for naming, source code formatting, documentation, and so on)

■ RUP vs. XP

- RUP : emphasizes building OO Models with UML
- XP : emphasizes producing executable code.

끝