Software Development Methodologies

Lecturer: **Raman Ramsin**

Lecture 6

Integrated Object-Oriented Methodologies: RUP/USDP and EUP

Rational Unified Process (RUP)

- Initial version officially released in 1998
- Revised versions introduced in 2000 and 2003
- Developed at Rational Corporation by the three principal developers of the OMT, Booch and OOSE (Objectory) methodologies: Rumbaugh, Booch and Jacobson
- A non-proprietary, somewhat less complex variant called USDP (Unified Software Development Process) was introduced in 1999

Rational Unified Process (RUP)

- UML-based
- Use case driven, a feature inherited from OOSE
- Iterative-incremental, with the overall process resembling the Micro-in-Macro process of the Booch methodology
- Covering the full generic lifecycle

RUP: Process – Phases

- Overall development *cycle* consists of four *phases*:
  - *Inception:* defining the scope and objectives of the project, as well as the business case
  - *Elaboration:* capturing the crucial requirements, developing and validating the architecture of the software system, and planning the remaining phases of the project
  - *Construction:* implementing the system in an iterative and incremental fashion based on the architecture developed in the previous phase
  - *Transition:* testing and releasing the system

RUP: Process – Iterations and Disciplines

- Each phase can be further broken down into *iterations*
- An iteration is a complete development loop resulting in a release of an executable increment to the system
- Each iteration consists of nine work areas (*disciplines*) performed during the iteration
- For each discipline, RUP defines sets of:
  - *artefacts* (work products)
  - *activities* (units of work on the artefacts)
  - *roles* (responsibilities taken on by development team members)

RUP: Process – Disciplines

1. *Business Modeling:* concerned with describing business processes and the internal structure of a business. A *Business Use Case Model* and a *Business Object Model* are developed.

2. *Requirements Management:* concerned with eliciting, organizing, and documenting requirements. The *Use Case Model* is produced.

3. *Analysis and Design:* concerned with creating the architecture and the design of the software system; results in a *Design Model* and optionally an *Analysis Model*.

4. *Implementation:* concerned with writing and debugging source code, unit testing, and build management. Source code files, executables, and supportive files are produced.

5. *Test:* concerned with integration-, system- and acceptance testing.

RUP: Process – Disciplines (Contd.)

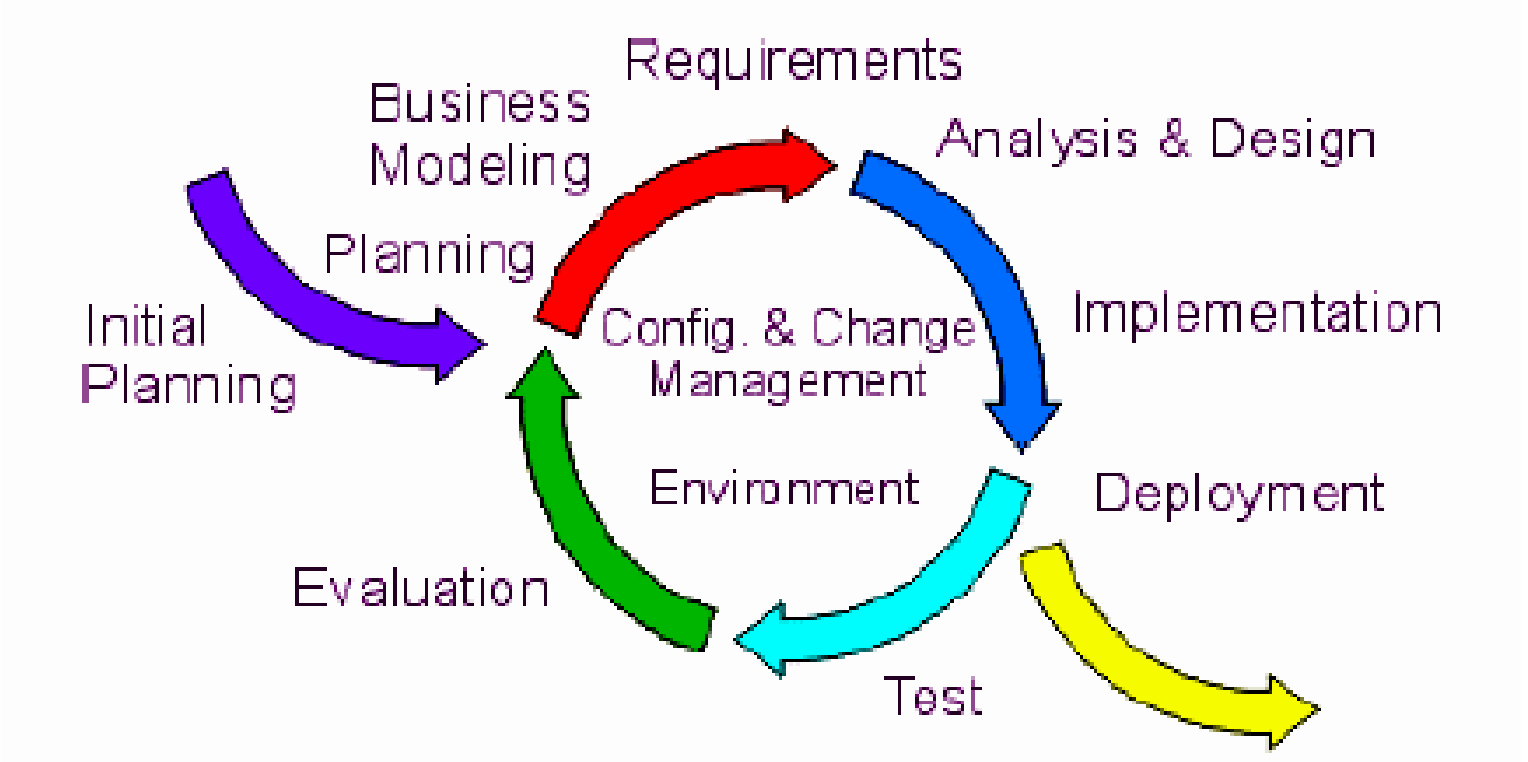
6. *Deployment:* concerned with packaging the software, creating installation scripts, writing end-user documentation and other tasks needed to make the software available to its end-users.

7. *Project Management:* concerned with project planning, scheduling and control.

8. *Configuration and Change Management:* concerned with version- and release management and change-request management.

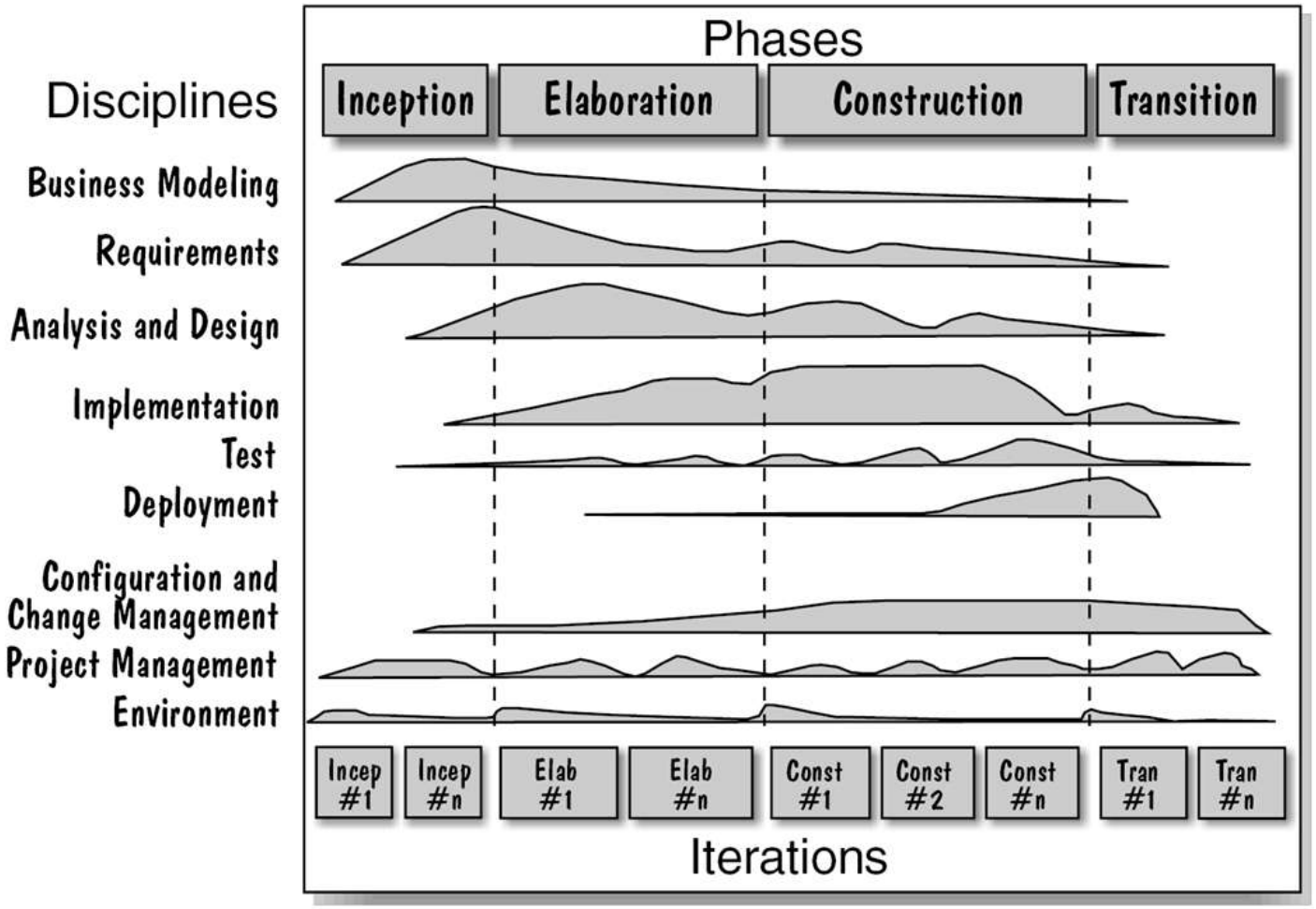
9. *Environment:* concerned with adapting the process to the needs of a project or an organization, and selecting, introducing and supporting development tools.

RUP: Process – Disciplines in Iterations



[Kruchten 2003]

RUP: Process – Disciplines in Iterations and Phases



[Kruchten 2003]

RUP: Process – *Inception* Phase

- Tasks include:

1. Describe the initial requirements.
2. Develop and justify the business case for the system.
3. Determine the scope of the system.
4. Identify the people, organizations, and external systems that will interact with the system.
5. Develop initial risk assessment, schedule, and estimates.
6. Configure the initial system architecture.

RUP: Process – *Elaboration* Phase

- Tasks include:

1. Produce an architectural baseline for the system.
2. Evolve the requirements model to 80% completion.
3. Draft a coarse-grained project plan for the construction phase.
4. Ensure that critical tools, processes, standards, and guidelines have been put in place for the construction phase.
5. Understand and eliminate high-priority risks of the project.

RUP: Process – *Construction* Phase

- Tasks include:

1. Describe the remaining requirements.
2. Develop the design of the system.
3. Ensure that the system meets the needs of its users and fits into the organization's overall system configuration.
4. Complete component development and testing, including both the software product and its documentation.
5. Minimize development costs by optimizing resources.
6. Achieve adequate quality.
7. Develop useful versions of the system.

RUP: Process – *Transition* Phase

- Tasks include:
  1. Test and validate the complete system.
  2. Integrate the system with existing systems.
  3. Convert legacy databases and systems to support the new release.
  4. Train the users of the new system.
  5. Deploy the new system into production.

RUP: Strengths and Weaknesses

- ***Strengths***

  - Iterative-incremental process
  - Well-documented process
  - Based on functional, behavioural, and structural modeling of the problem domain and the system
  - Traceability supported through use cases
  - Seamlessness (though with hiccups, e.g. transforming use cases to sequence diagrams)
  - Architecture-centric process (which necessitates early specification of an architectural blueprint)

RUP: Strengths and Weaknesses

- ***Strengths*** **(Contd.)**
  - Customizability addressed
  - Risk-based development, aimed at mitigating the risks before undertaking the tasks
  - Support for structural, behavioural and functional modeling at all levels (problem domain to objects; logical to physical)
  - Rich modeling language (UML), especially in structural and behavioural modeling features
  - Support for formalism (through UML/OCL)

RUP: Strengths and Weaknesses

- ***Weaknesses***
  - Very complex process
  - The process is confusing to those involved. The iterative-incremental nature of the process further complicates the issue.
  - Although advertised as customizable, configuring the process is a formidable task in itself.
  - Since the process is very complex, not having a maintenance phase, on the grounds that it can be performed by iterating the whole process as a cycle, is not convincing.
  - Prohibitive number of models
  - Strict adherence to UML, which is not necessarily constructive, especially since UML is not perfect and can exacerbate the model inconsistency problem.

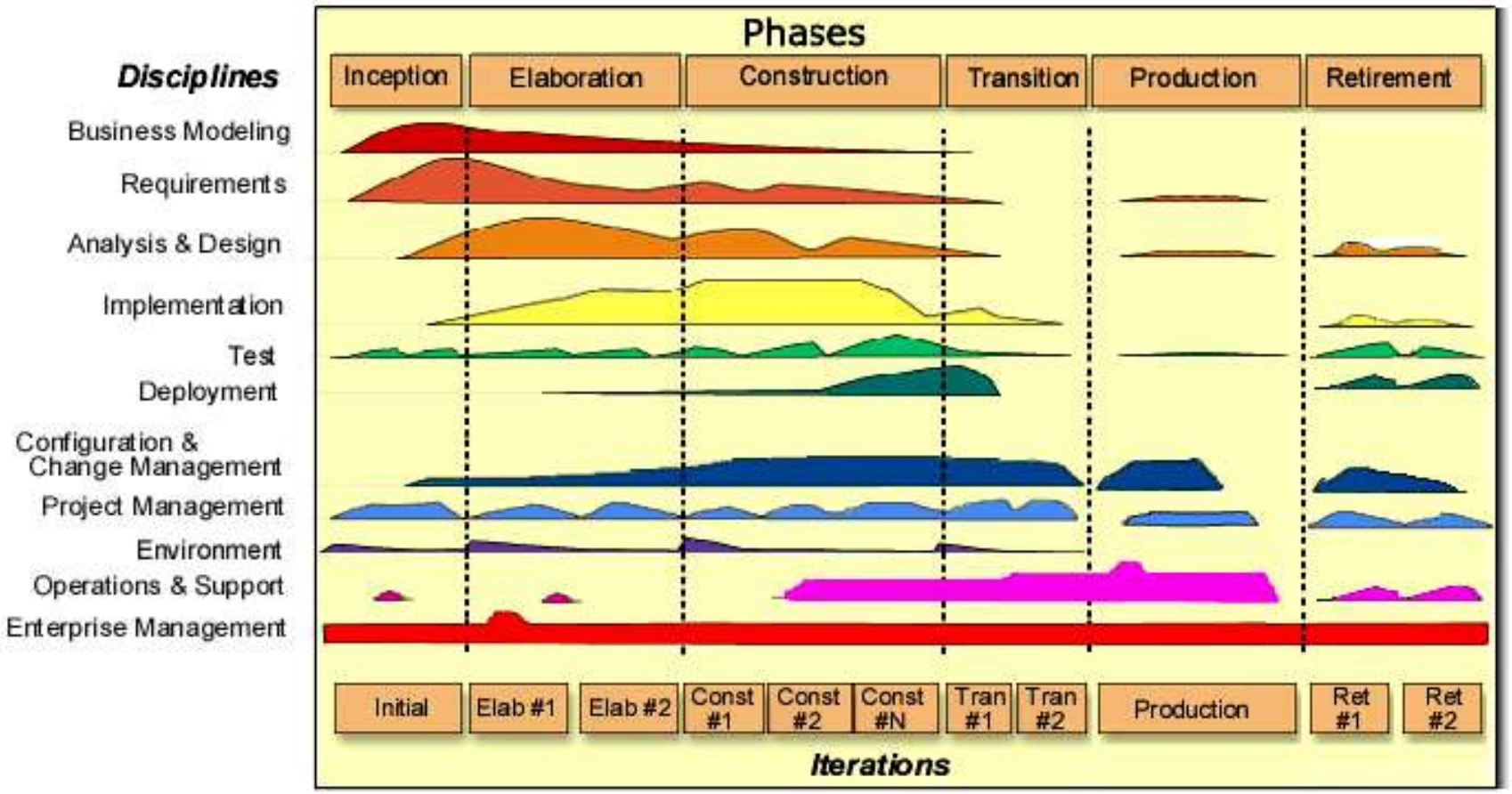
Enterprise Unified Process (EUP)

- Introduced by Ambler and Constantine in 2000 as an extended variant of RUP
- A revised and refactored version was introduced in 2005
- Motivated by the belief that RUP suffers from serious drawbacks:
  - RUP does not cover system support and eventual retirement.
  - RUP does not explicitly support organization-wide infrastructure development.
  - The iterative nature of RUP is both a strength and a weakness, since the iterative nature of the lifecycle is hard to grasp for many experienced developers.
  - Rational's approach to developing RUP was initially tools-driven; hence the resulting process is not sufficient for the needs of developers.

Enterprise Unified Process (EUP)

- Extends RUP by adding two new *phases* and two new *disciplines*, one of which was further broken down into seven disciplines in the 2005 version of the methodology
- Extends the activities in some of the old disciplines of RUP
- Whereas RUP advocates adherence to UML, EUP makes use of some older modeling notations too; e.g. the use of Data Flow Diagrams for business modeling.
- EUP stresses that use cases are not enough for modeling the requirements; consequently, use cases in EUP do not have the pivotal role they have in RUP.

EUP: Process – Disciplines in Iterations and Phases



[Ambler et al. 2005]

EUP: Process – *Production* Phase

- Focus is on keeping the software in production until it is either replaced with a new version (by executing the lifecycle all over again), or retired and removed.
- There are no iterations during this phase.
- Somewhat similar to the maintenance phase in the generic lifecycle, in that it is mainly concerned with the operation and support of the system.
- Unlike classic maintenance, any need for changing the system (even a bug fix) will result in the reinitiation of the development cycle.

EUP: Process – *Retirement* Phase

- Added in 2002 as the sixth phase
- Focus is on the careful removal of a system from production, either because it is no longer needed or is being replaced. This typically includes:
  - Identification of the existing system's coupling to other systems.
  - Redesign and rework of other systems so that they no longer rely on the system being retired.
  - Transformation of existing legacy data.
  - Archival of data previously maintained by the system that is no longer needed by other systems.
  - System integration testing of the remaining systems to ensure that they have not been broken via the retirement of the system in question.

EUP: Process – *Operations and Support* Discipline

- Concerned with issues related to operating and supporting the system

- Spans several phases, not only the production phase:
  - During the *construction* phase, and perhaps as early as the *elaboration* phase, the development of operations and support plans, documents, and training manuals is initiated.
  - Artefacts are enhanced and perfected during the *transition* phase, where the discipline will also include the training of the operations and support staff.
  - During the *production* and *retirement* phases, the discipline covers classic maintenance activities.

EUP: Process – *Enterprise Management* Discipline

- Concerned with the activities required to create, evolve, and maintain the organization's cross-system artefacts, such as:
  - Organization-wide models (requirements and architecture)
  - Software process
  - Standards
  - Guidelines
  - Reusable artefacts
- Broken down into seven disciplines in the 2005 version of the methodology

EUP: Process – *Enterprise Management:* Seven Disciplines

- Added in 2005, these disciplines prescribe enterprise management activities in a more fine-grained fashion:

1. Enterprise Business Modeling
2. Portfolio Management
3. Enterprise Architecture
4. Strategic Reuse
5. People Management
6. Enterprise Administration
7. Software Process Improvement

EUP: Strengths and Weaknesses

- ***Strengths***
  - Same benefits as RUP
  - Addresses enterprise-level issues
  - Maintenance is a phase in its own right.
  - Attention is given to post-mortem activities when retiring the project (in the form of a new *Retirement* phase).
  - Not strictly adherent to UML; other modeling languages such as DFDs are also used.

EUP: Strengths and Weaknesses

- ***Weaknesses***
  - Like RUP, EUP is
    - very complex
    - encumbered with a prohibitive number of models
    - suffering high potential for model inconsistency
    - confusing as to the process used
    - hard to customize
  - EUP has added further complexity to RUP by adding two new phases and two new disciplines.
  - Adding the maintenance phase is not sufficient, since any change needed will result in a restart of the development process.

References

- Kruchten, P., *Rational Unified Process: An Introduction*, 3rd ed. Addison-Wesley, 2003.
- Ambler, S. W., Nalbone, J., Vizdos, M. J., *The Enterprise Unified Process: Extending the Rational Unified Process*. Prentice-Hall, 2005.