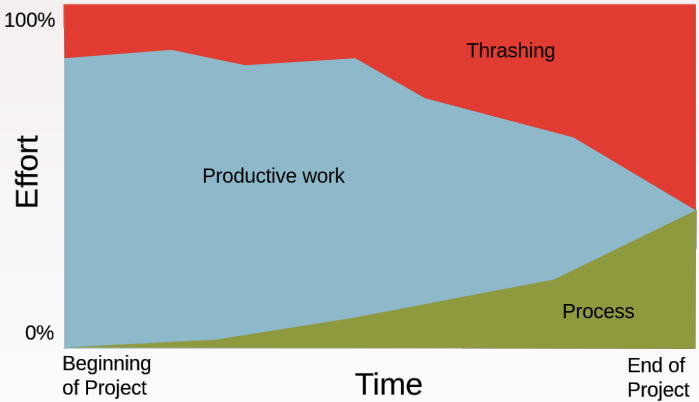
04834580 Software Engineering (Honor Track) 2024-25

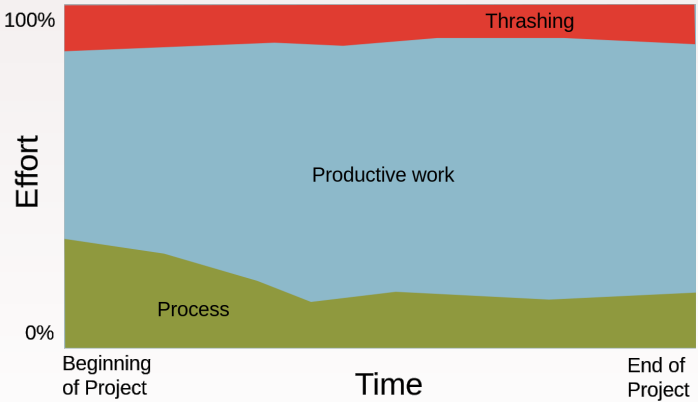
Software Development Lifecycle

Sergey Mechtaev
mechtaev@pku.edu.cn
School of Computer Science, Peking University

Little Attention to Process [1]

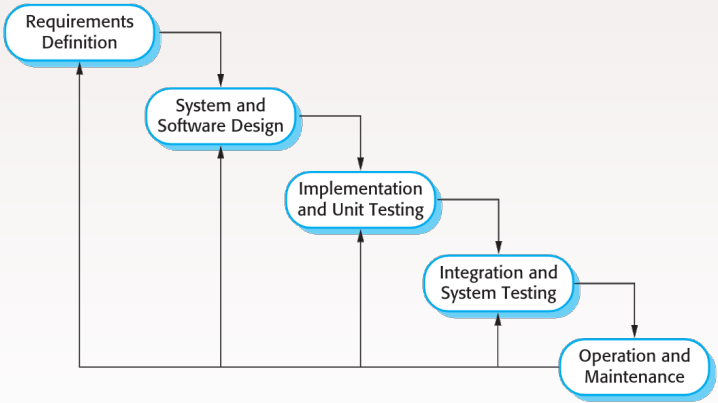


Early Attention to Process [1]



Waterfall Process Model [2]

北京大学



Development Stages [3]

- **Requirements analysis and definition:** The system's services, constraints, and goals are established by consultation with system users.
- **System and software design:** Establishing an overall system architecture, identifying and describing fundamental abstractions and their relationships.
- **Implementation and unit testing:** The software design is realized. Unit testing verifies that each unit meets its specification.
- **Integration and system testing:** The individual program units are integrated and tested as a complete system to ensure that the software requirements have been met.
- **Operation and maintenance:** The system is installed and put into practical use. Maintenance involves correcting errors, improving the implementation, and enhancing the system's services as new requirements are discovered.

Trade-offs of Waterfall Model

北京大学

Advantages

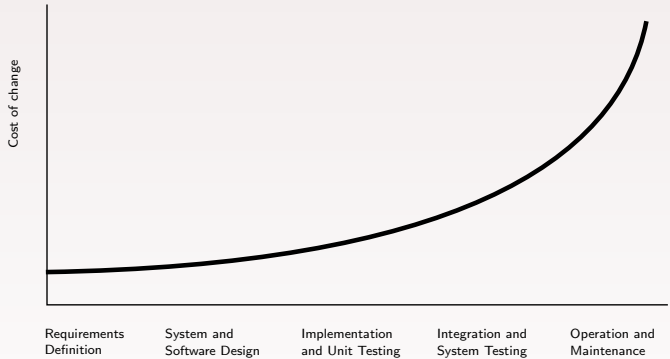
- Allows for departmentalization and control
- Easy to use
- Easy to manage

Disadvantages

- Often unrealistic: requirements constantly changing
- Lessons learned in later stages affect earlier ones

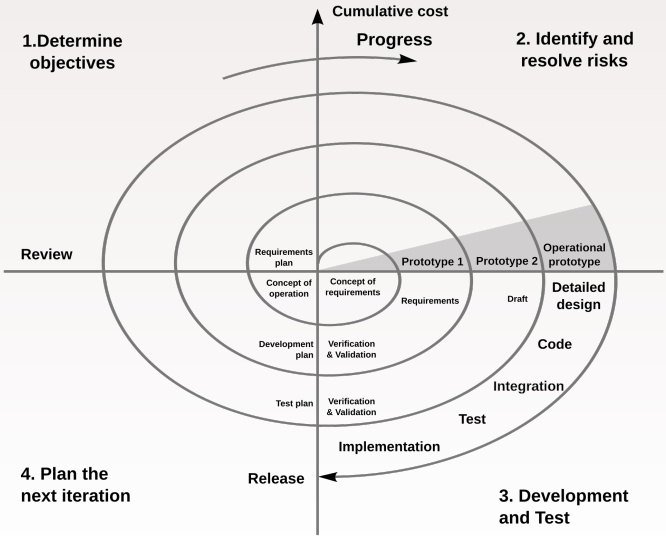
Can be useful when requirements are stable and communication cost is high.

Cost of Change



[In traditional development model,] the cost to fix a problem in a piece of software rises exponentially over time. A problem that might take a dollar to fix if you found it during requirements analysis might costs thousands to fix once the software is in production. [4]

Spiral Development Model [5]



- **Objective setting:** Specific objectives for that phase of the project are defined.
- **Risk assessment and reduction:** Identify and analyze project risks. Take steps to reduce the risks.
- **Development and validation:** After risk evaluation, a development model for the system is chosen, e.g. waterfall.
- **Planning:** The project is reviewed and a decision made whether to continue with a further loop of the spiral.

Trade-offs of Spiral Model

北京大学

Advantages

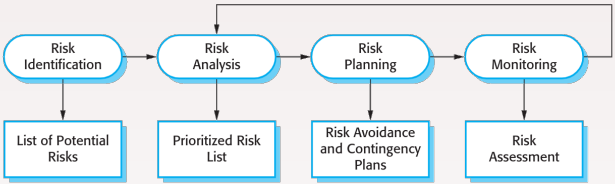
- Early indication of unforeseen problems
- Allows for changes
- The risk reduces as costs increase

Disadvantages

- More complex to run
- Requires proper risk assessment
- Requires more planning and experienced management

Risk in itself is not bad; risk is essential to progress, and failure is often a key part of learning. But we must learn to balance the possible negative consequences of risk against the potential benefits of its associated opportunity.
— Roger L. Van Scoy [6]

Risk Management Process [3]



- **Risk identification:** You should identify possible project, product, and business risks.
- **Risk analysis:** You should assess the likelihood and consequences of these risks.
- **Risk planning:** You should make plans to address the risk, either by avoiding it or minimizing its effects on the project.
- **Risk monitoring:** You should regularly assess the risk and your plans for risk mitigation and revise these when you learn more about the risk.

Risk Identification [3]

- **Project Risks:** Risks that affect the project schedule or resources. An example of a project risk is the loss of an experienced designer.
- **Product Risks:** Risks that affect the quality or performance of the software being developed. An example of a product risk is the failure of a purchased component to perform as expected.
- **Business Risks:** Risks that affect the organization developing or procuring the software. For example, a competitor introducing a new product is a business risk.

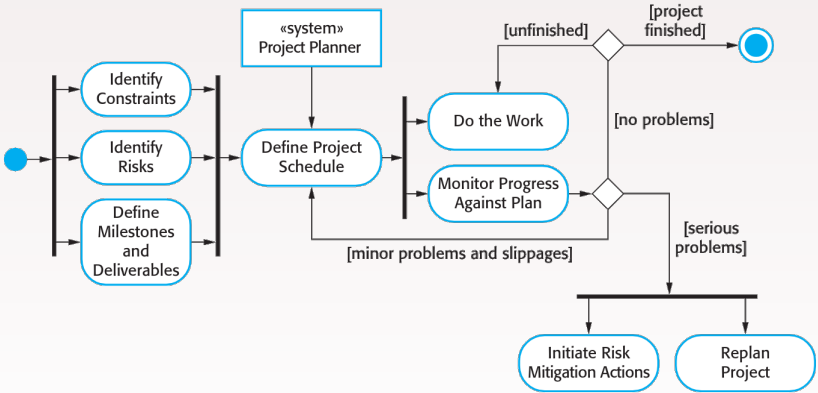
Assess **risk probability** (e.g. insignificant, low, moderate, high, very high) and **effect** (e.g. catastrophic, serious, tolerable, insignificant).

Risk	Probability	Effect
Organizational financial problems force reduction in the project budget	Low	Catastrophic
Key staff are ill at critical times in the project	Moderate	Serious
Software tools cannot be integrated	High	Tolerable
Changes to requirements that require major design rework are proposed	Moderate	Serious

$$\text{Risk Exposure} = \text{Risk Probability} \times \text{Risk Effect}$$

Risk effect \ Risk probability	High	Moderate	Low
Catastrophic	High	High	Moderate
Serious	High	Moderate	Moderate
Tolerable	Moderate	Moderate	Low
Insignificant	Moderate	Low	Low

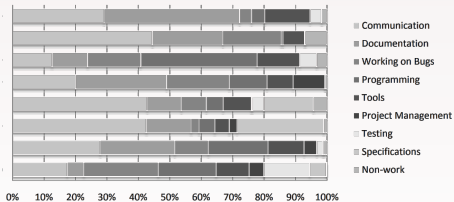
Planning Process [3]



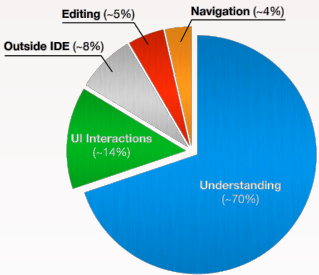
How Developers Spend Their Time

北京大学

2008's study with 8 new Microsoft developers showed that programming occupies only 10–20% of their time. [7]



2015's study with 18 developers showed that they spend 70% of their time on understanding. [8]



References I

[1] Steve McConnell.
The power of process [software processes].
Computer, 31(5):100–102, 1998.

[2] Winston W Royce.
Managing the development of large software systems: concepts and techniques.
In *Proceedings of the 9th international conference on Software Engineering*, pages 328–338, 1987.

[3] Ian Sommerville.
Software Engineering, 9/E.
Pearson Education India, 2011.

[4] Kent Beck.
Extreme programming explained: embrace change.
addison-wesley professional, 2000.

References II

[5] Barry W. Boehm.
A spiral model of software development and enhancement.
Computer, 21(5):61–72, 1988.

[6] Roger L Van Scoy.
Software development risk: opportunity, not problem.
Carnegie Mellon University, Software Engineering Institute Pittsburgh, Pa., 1992.

[7] Andrew Begel and Beth Simon.
Novice software developers, all over again.
In *International Workshop on Computing Education Research*, pages 3–14, 2008.

[8] Roberto Minelli, Andrea Mocci, and Michele Lanza.
I know what you did last summer-an investigation of how developers spend their time.
In *International Conference on Program Comprehension*, pages 25–35. IEEE, 2015.