Chapter 4

- **Process Models**

Slide Set to accompany

Software Engineering: A Practitioner's Approach, 8/e

by Roger S. Pressman and Bruce R. Maxim

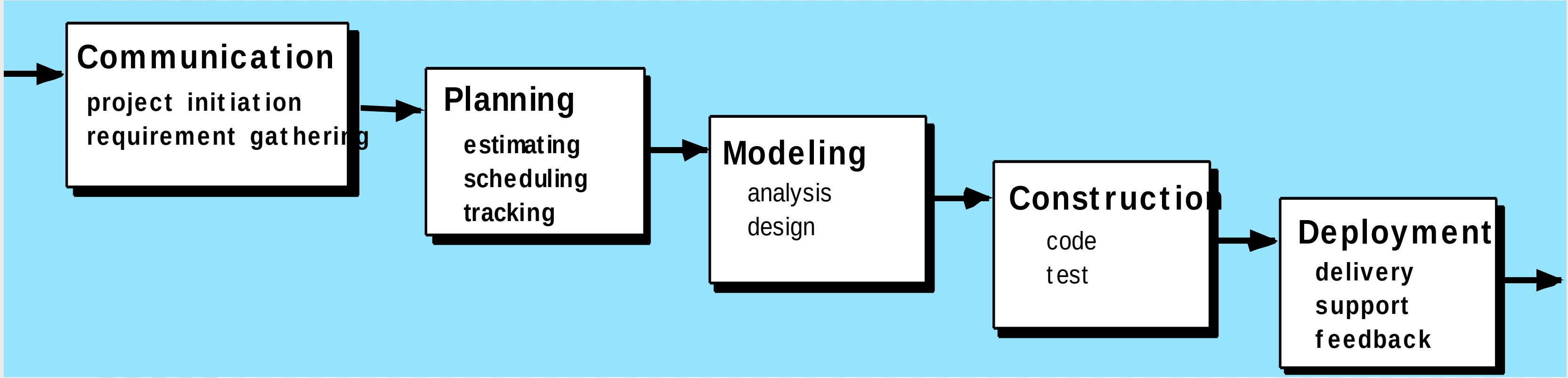
Prescriptive Models

- Prescriptive process models advocate an orderly approach to software engineering

That leads to a few questions …

- If prescriptive process models strive for structure and order, are they inappropriate for a software world that thrives on change?
- Yet, if we reject traditional process models (and the order they imply) and replace them with something less structured, do we make it impossible to achieve coordination and coherence in software work?

The Waterfall Model



Communication
project initiation
requirement gathering

Planning
estimating
scheduling
tracking

Modeling
analysis
design

Construction
code
test

Deployment
delivery
support
feedback

These slides are designed to accompany *Software Engineering: A Practitioner's Approach, 8/e* (McGraw-Hill, 2014). Slides copyright 2014 by Roger Pressman.

The Waterfall Model

- There are separate identified phases in the waterfall model:
  - Requirements analysis and definition
  - System and software design
  - Implementation and unit testing
  - Integration and system testing
  - Operation and maintenance

Advantages: Waterfall Model

- This model is simple and easy to understand and use.
- It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- In this model phases are processed and completed one at a time. Phases do not overlap.
- Waterfall model works well for smaller projects where requirements are very well understood

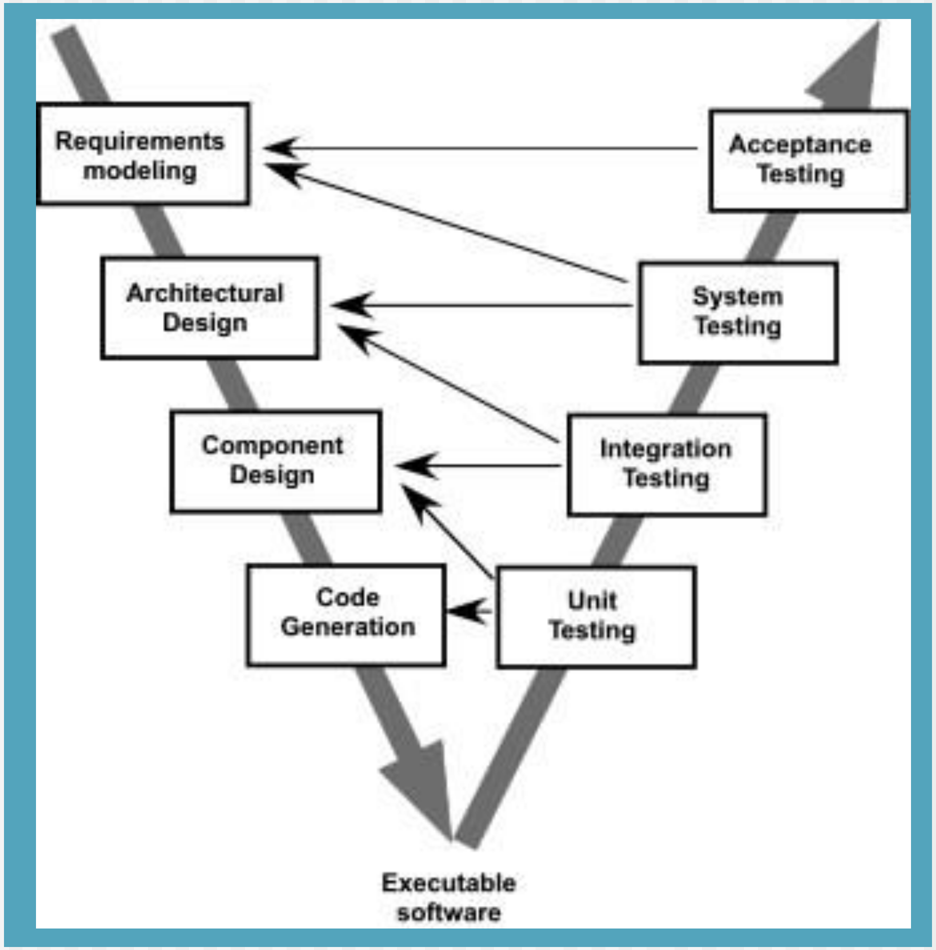
Disadvantages: Waterfall Model

- Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing.

When to Use: Waterfall Model

- This model is used only when the requirements are very well known, clear and fixed.
- Product definition is stable.
- Technology is understood.
- There are no ambiguous requirements
- Ample resources with required expertise are available freely
- The project is short.

The V-Model



These slides are designed to accompany *Software Engineering: A Practitioner's Approach, 8/e* (McGraw-Hill, 2014). Slides copyright 2014 by Roger Pressman.

The V-Model

- V- model means Verification and Validation model. Just like the waterfall model, the V-Shaped life cycle is a sequential path of execution of processes.
- Each phase must be completed before the next phase begins.
- Testing of the product is planned in parallel with a corresponding phase of development in **V-model**.

Phases: V-Model

- **Requirements** like BRS and SRS begin the life cycle model just like the waterfall model. But, in this model before development is started, a system test plan is created. The test plan focuses on meeting the functionality specified in the requirements gathering.
- **The high-level design (HLD)** phase focuses on system architecture and design. It provide overview of solution, platform, system, product and service/process. An integration test plan is created in this phase as well in order to test the pieces of the software systems ability to work together.

Phases: V-Model

- **The low-level design (LLD)** phase is where the actual software components are designed. It defines the actual logic for each and every component of the system. Class diagram with all the methods and relation between classes comes under LLD. Component tests are created in this phase as well.
- **The implementation** phase is, again, where all coding takes place. Once coding is complete, the path of execution continues up the right side of the V where the test plans developed earlier are now put to use.
- **Coding:** This is at the bottom of the V-Shape model. Module design is converted into code by developers.

Advantages: V-Model

- Simple and easy to use.
- Testing activities like planning, test designing happens well before coding. This saves a lot of time. Hence higher chance of success over the waterfall model.
- Proactive defect tracking – that is defects are found at early stage.
- Avoids the downward flow of the defects.
- Works well for small projects where requirements are easily understood.

Disadvantages: V-Model

- Very rigid and least flexible.
- Software is developed during the implementation phase, so no early prototypes of the software are produced.
- If any changes happen in midway, then the test documents along with requirement documents has to be updated.

When to Use: V-Model

- The V-shaped model should be used for small to medium sized projects where requirements are clearly defined and fixed.
- The V-Shaped model should be chosen when ample technical resources are available with needed technical expertise.

Interactive Session

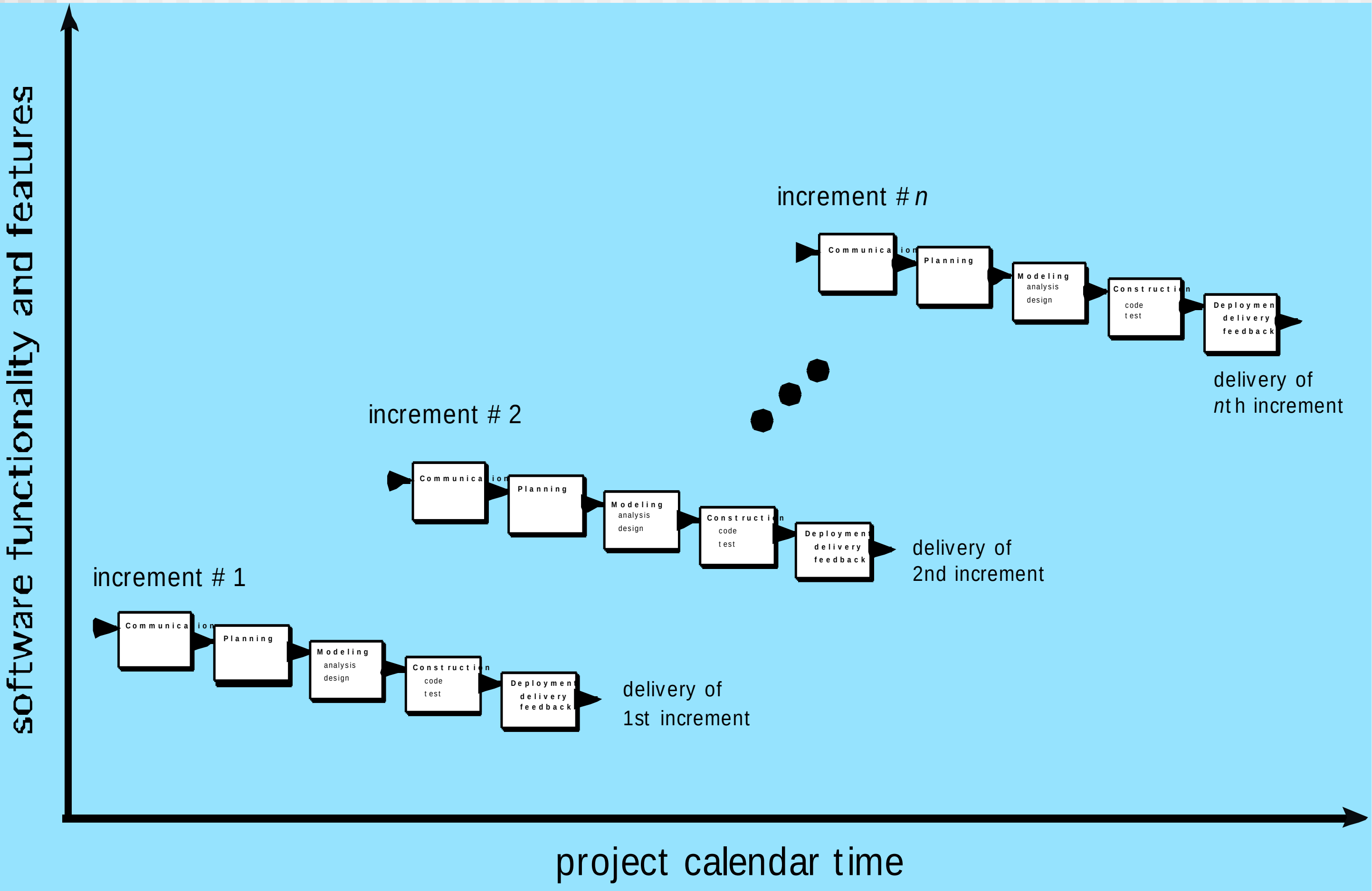
- Ask the students: what do they think about the scenarios for the case of tasks 1, 2, and 3 (shown in appendix) related to any or both waterfall model, v-model or not?

Break Time: 5 min

- Brainstorming: Which one is the most advanced and complex software system based on your best of breed knowledge?

Tell us your story

The Incremental Model



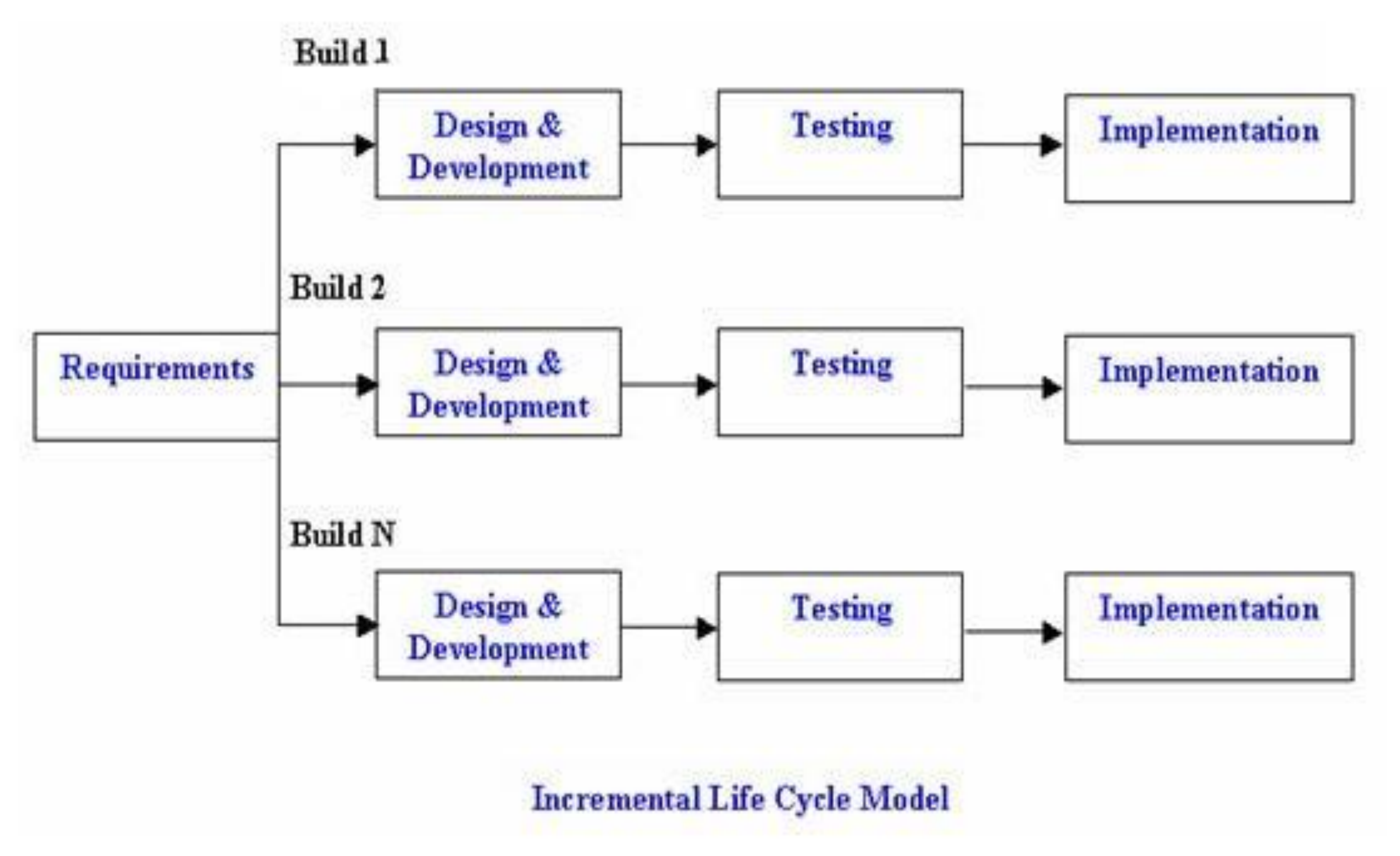
The Incremental Model

- In incremental model the whole requirement is divided into various builds. Multiple development cycles take place here, making the life cycle a “multi-waterfall” cycle.
- Cycles are divided up into smaller, more easily managed modules. Each module passes through the requirements, design, implementation and testing phases.
- A working version of software is produced during the first module, so you have working software early on during the software life cycle.
- Each subsequent release of the module adds function to the previous release.

The Incremental Model

- Rather than deliver the system as a single delivery, the development and delivery is broken down into increments with each increment delivering part of the required functionality.
- User requirements are prioritised and the highest priority requirements are included in early increments.
- Once the development of an increment is started, the requirements are frozen though requirements for later increments can continue to evolve.

How It Works: The Incremental Model



Incremental Life Cycle Model

Advantages: The Incremental Model

®Generates working software quickly and early during the software life cycle.

®This model is more flexible – less costly to change scope and requirements.

®It is easier to test and debug during a smaller iteration.

®In this model customer can respond to each built.

®Lowers initial delivery cost.

®Easier to manage risk because risky pieces are identified and handled during it’d iteration.

Disadvantages: The Incremental Model

®Needs good planning and design.

®Needs a clear and complete definition of the whole system before it can be broken down and built incrementally.

®Total cost is higher than waterfall.

When to Use: The Incremental Model

®This model can be used when the requirements of the complete system are clearly defined and understood.

®Major requirements must be defined; however, some details can evolve with time.

®There is a need to get a product to the market early.

®A new technology is being used

®Resources with needed skill set are not available

®There are some high risk features and goals.

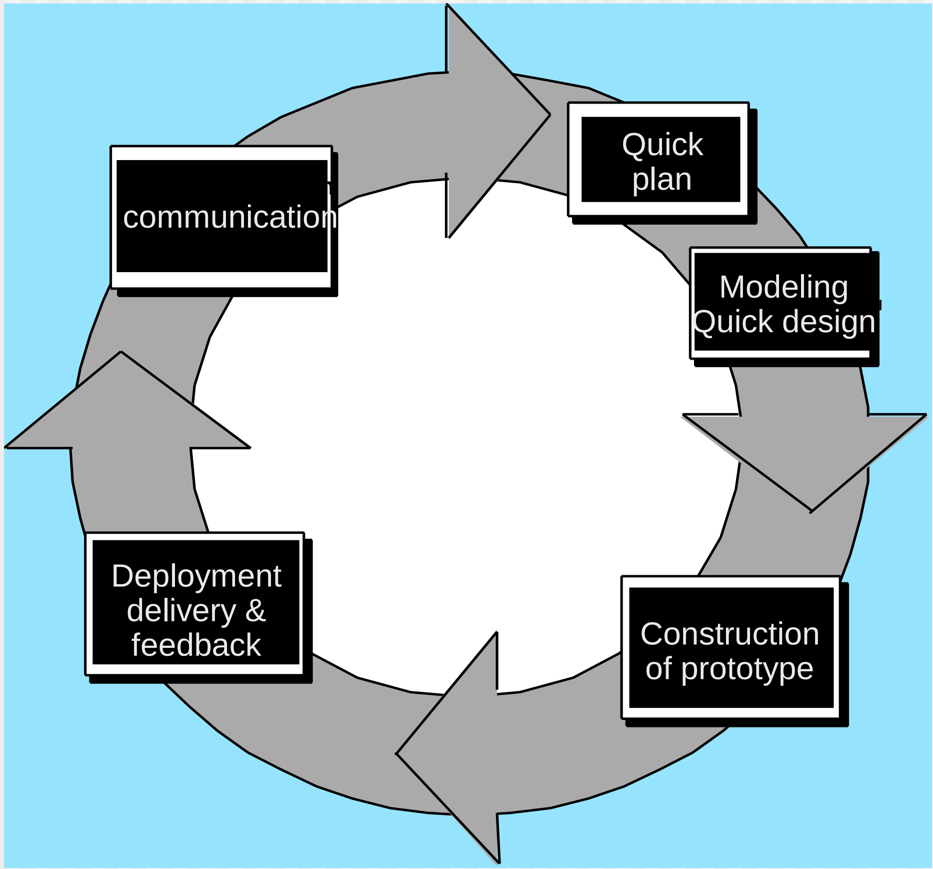
Interactive Session

- Ask the students: what do they think about the scenarios for the case of tasks 1, 2, and 3 (shown in appendix) related to incremental model or not?

Break Time: 5 min

Brainstorming: Suppose you are dealing with a large retailer company like PRAN-RFL group of industries. You are a tech expert and your job is to provide consultancy support for your valuable clients. Now, PRAN-RFL group needs your support as they want a centralized monitoring system software for their production unit. As an expert, what will be your suggestions on this matter to your client? Identify 5-6 points on this matter.

Evolutionary Models: Prototyping



These slides are designed to accompany *Software Engineering: A Practitioner's Approach, 8/e* (McGraw-Hill, 2014). Slides copyright 2014 by Roger Pressman.

Evolutionary Models: Prototyping

A prototype is an initial version of a system used to demonstrate concepts and try out design options.

A prototype can be used in:

- The requirements engineering process to help with requirements elicitation and validation;
- In design processes to explore options and develop a UI design;
- In the testing process to run back-to-back tests.

Construction of prototype

Development: Prototyping

- ❖ May be based on rapid prototyping languages or tools
- ❖ May involve leaving out functionality
  - Prototype should focus on areas of the product that are not well-understood;
  - Error checking and recovery may not be included in the prototype;
  - Focus on functional rather than non-functional requirements such as reliability and security

Construction
of prototype

Throw Away: Prototyping

- Prototypes should be discarded after development as they are not a good basis for a production system:
  - It may be impossible to tune the system to meet non-functional requirements;
  - Prototypes are normally undocumented;
  - The prototype structure is usually degraded through rapid change;
  - The prototype probably will not meet normal organizational quality standards.

Construction
of prototype

Interactive Session

- Ask the students: what do they think about the scenarios for the case of tasks 1, 2 and 3 (shown in appendix) related to prototype model or not?

Construction of prototype

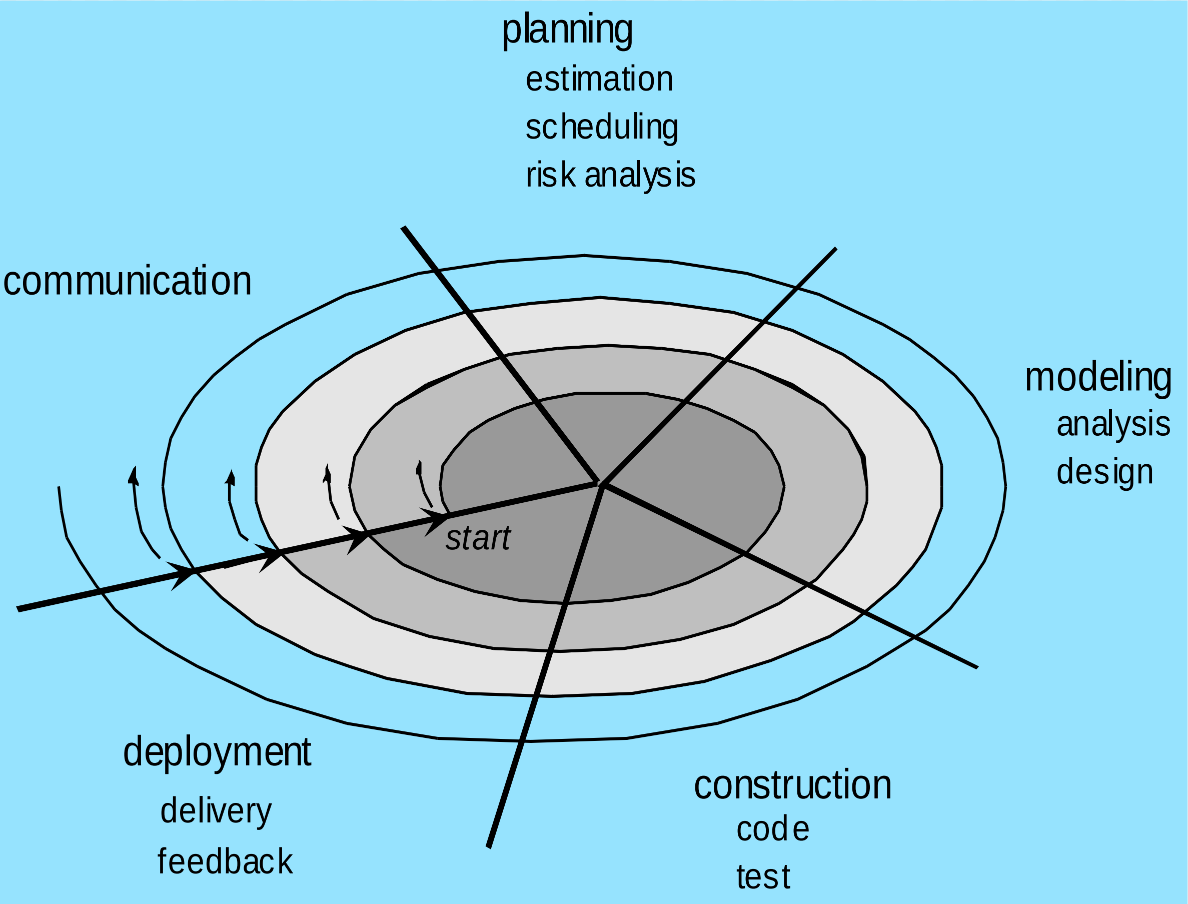
Break Time: 5 min

- Brainstorming: Progress in business depends largely on the executive decisions taken by the higher authorities. Imagine, you have given the role of CTO for a start-up software company. You can consider that you have a limited number of staffs under your supervision. Recently, the head of the HR has informed you that the total marginal revenue for this fiscal year is about 23% uprising than the previous year. As, you have a limited number of staffs, to increase software production, you can either recruit new staffs or can store this revenue to your company capital. What will be your probable decision on this matter?

Construction of prototype

These slides are designed to accompany *Software Engineering: A Practitioner’s Approach, 8/e* (McGraw-Hill, 2014). Slides copyright 2014 by Roger Pressman.

Evolutionary Models: The Spiral

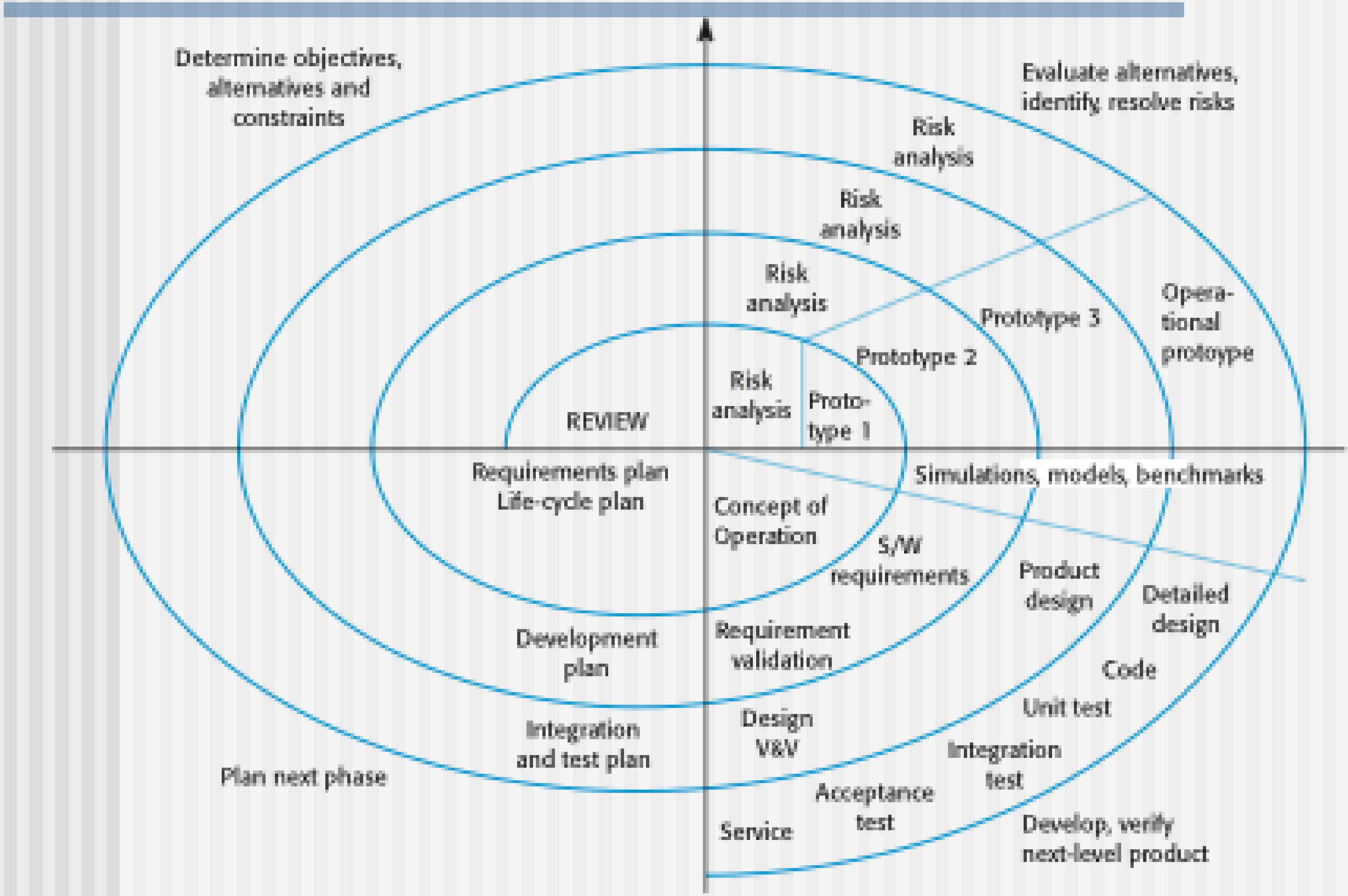


These slides are designed to accompany *Software Engineering: A Practitioner's Approach, 8/e* (McGraw-Hill, 2014). Slides copyright 2014 by Roger Pressman.

Evolutionary Models: The Spiral

- Process is represented as a spiral rather than as a sequence of activities with backtracking.
- Each loop in the spiral represents a phase in the process.
- No fixed phases such as specification or design - loops in the spiral are chosen depending on what is required.
- Risks are explicitly assessed and resolved throughout the process.

Deep World: The Spiral



These slides are designed to accompany *Software Engineering: A Practitioner's Approach, 8/e* (McGraw-Hill, 2014). Slides copyright 2014 by Roger Pressman.

Sectors: The Spiral

- Objective setting

  Specific objectives for the phase are identified.
- Risk assessment and reduction

  Risks are assessed and activities put in place to reduce the key risks.
- Development and validation

  A development model for the system is chosen which can be any of the generic models.
- Planning

  The project is reviewed and the next phase of the spiral is planned.

Usage: The Spiral

- Spiral model has been very influential in helping people think about iteration in software processes and introducing the risk-driven approach to development.
- In practice, however, the model is rarely used as published for practical software development.

Interactive Session

- Ask the students: what do they think about the scenarios for the case of tasks 1, 2 and 3 (shown in appendix) related to spiral model or not?

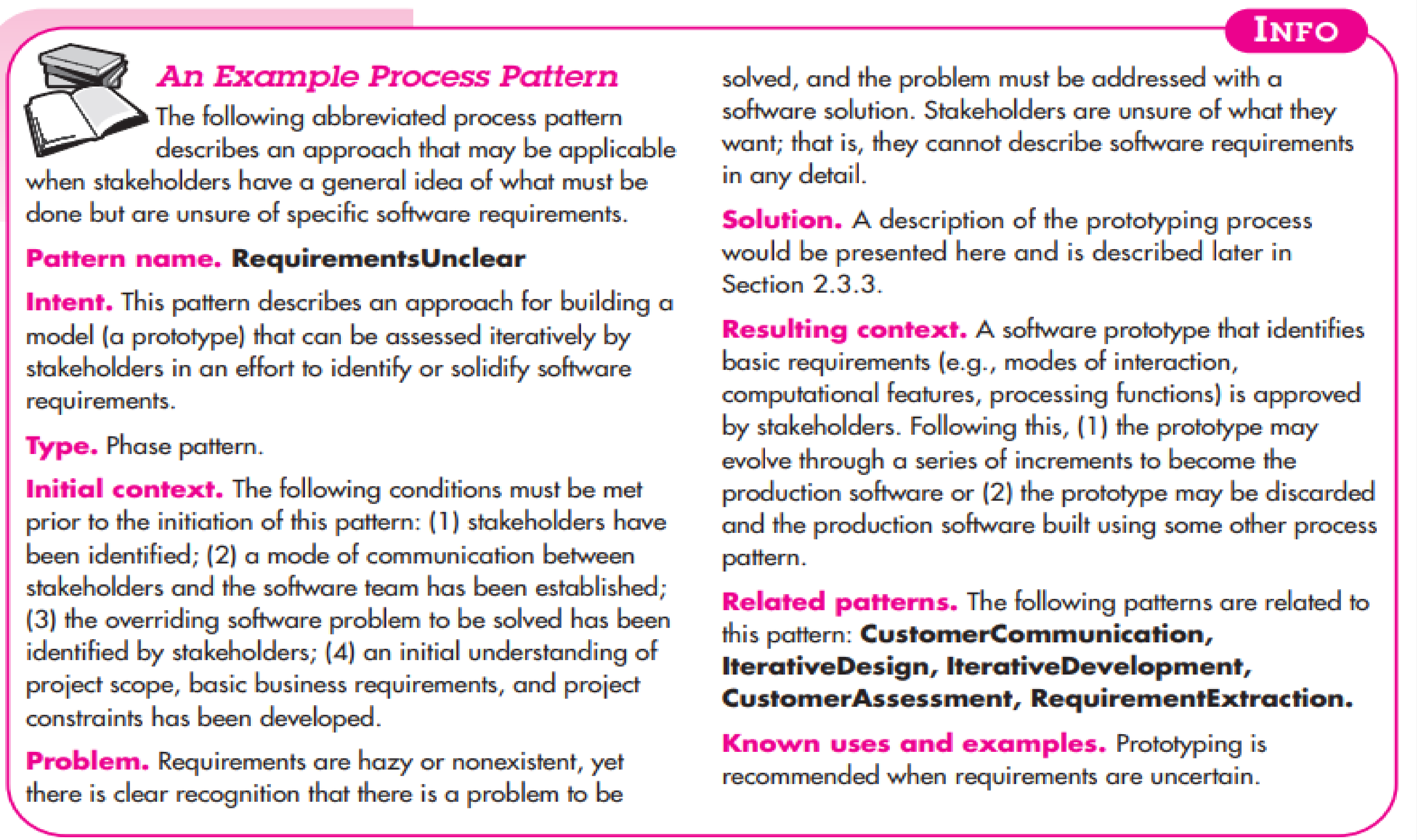
Open-Ended Task: 10 min

A common problem during communication occurs when you encounter two stakeholders who have conflicting ideas about what the software should be. That is, you have mutually conflicting requirements. Develop a process pattern.

See the figure (M).

(2.1.3, 37 p, R. Pressman)

Figure M: 10 min



INFO

An Example Process Pattern

The following abbreviated process pattern describes an approach that may be applicable when stakeholders have a general idea of what must be done but are unsure of specific software requirements.

Pattern name. **RequirementsUnclear**

Intent. This pattern describes an approach for building a model (a prototype) that can be assessed iteratively by stakeholders in an effort to identify or solidify software requirements.

Type. Phase pattern.

Initial context. The following conditions must be met prior to the initiation of this pattern: (1) stakeholders have been identified; (2) a mode of communication between stakeholders and the software team has been established; (3) the overriding software problem to be solved has been identified by stakeholders; (4) an initial understanding of project scope, basic business requirements, and project constraints has been developed.

Problem. Requirements are hazy or nonexistent, yet there is clear recognition that there is a problem to be solved, and the problem must be addressed with a software solution. Stakeholders are unsure of what they want; that is, they cannot describe software requirements in any detail.

Solution. A description of the prototyping process would be presented here and is described later in Section 2.3.3.

Resulting context. A software prototype that identifies basic requirements (e.g., modes of interaction, computational features, processing functions) is approved by stakeholders. Following this, (1) the prototype may evolve through a series of increments to become the production software or (2) the prototype may be discarded and the production software built using some other process pattern.

Related patterns. The following patterns are related to this pattern: **CustomerCommunication, IterativeDesign, IterativeDevelopment, CustomerAssessment, RequirementExtraction.**

Known uses and examples. Prototyping is recommended when requirements are uncertain.

Highlighted Topics

- Case Study: Software Models
- Details of Incremental, Spiral, V-model, Waterfall, Prototype

Appendix

Task 1: Suppose you are the team leader of a software company. Under your supervision, a team has been formed. Now, one of your clients wants to develop antivirus software from your company. This software needs to identify potential threats and take the necessary actions. So, your team needs to release the software with the necessary precautions. Several times of the testing process to be applied to make this software operational. The client is not technically equipped. **Predict** which process model will be selected for this development with reasons and also demonstrate the whole building process.

Appendix

Task 2: Suppose you are the team leader of a software company. Under your supervision, a team has been made. Now, a client wants to develop a security software from your company. This software is based on artificial intelligence. So, your team needs to release the software with the necessary precautions. **Predict** which process model will be selected for this development with reasons.

Appendix

Task 3: Suppose you are the project manager of a software company. Under your supervision, a team has been assigned. Now, one of your clients wants to develop an e-learning based mobile application from your company. So, your team needs to collect sample data from different educational institutions and then develop along wise. Several times of testing process need to be applied to make this software operational. This mobile application should be able to generate performance of progress report. Predict which process model will be selected for this development with reasons and also **demonstrate** the whole building process..