Cornell University
Computing and Information Science

CS 5150 Software Engineering

Three Types of Software Process

William Y. Arms

Types of Software Process

The basic **process steps** can be combined in many ways to create a successful **software process** for software development. This course emphasizes three types of process:

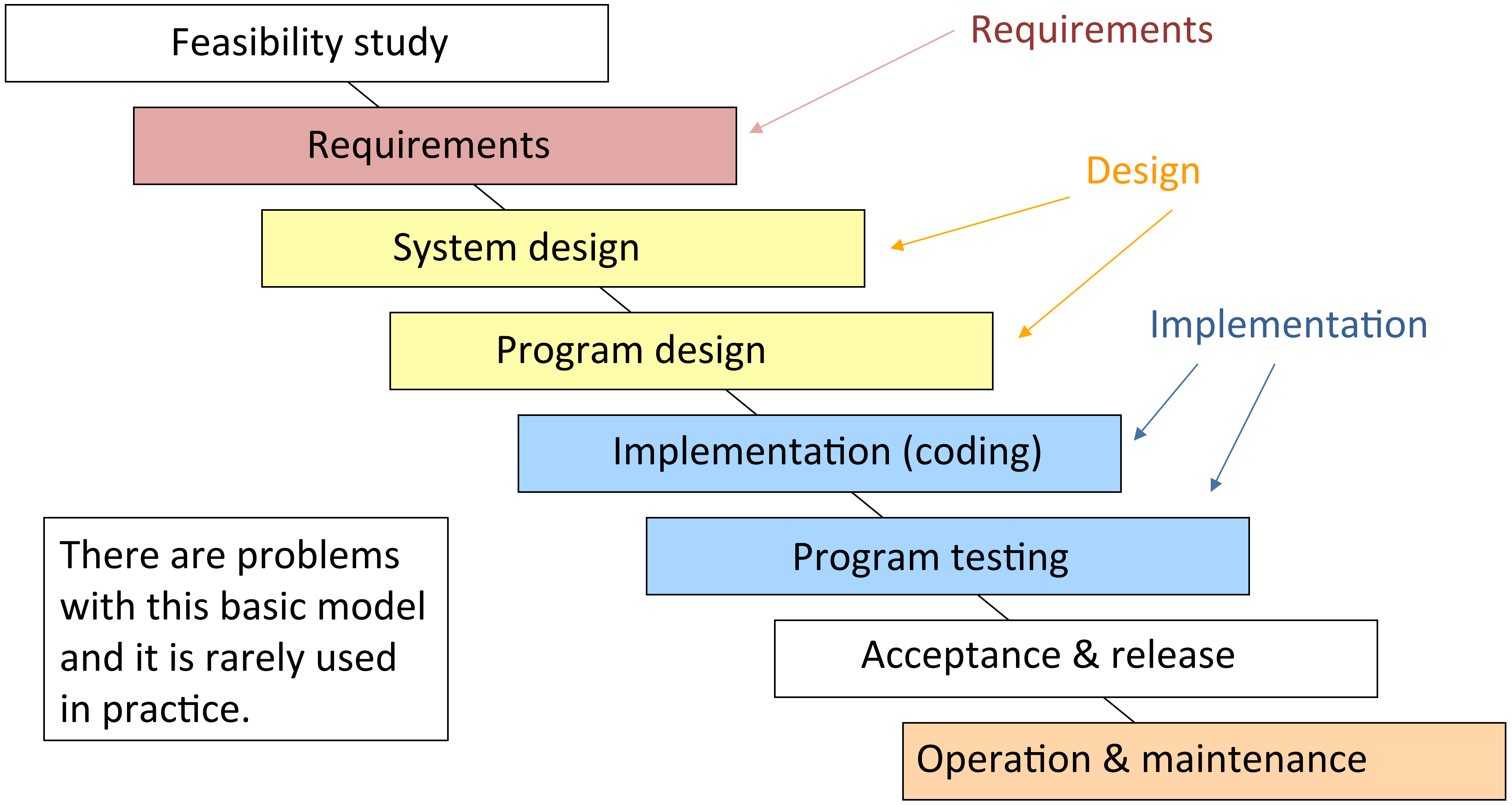
- Sequential (Modified waterfall model)
- Iterative refinement
- Incremental (Agile)

The process chosen for a particular system depends on many factors including the type of software product and the business practices of the client's organization.

The development of a major system may use all three methods in various combinations, including:

- Phased development
- Spiral development

Sequential Development: The Waterfall Model



There are problems with this basic model and it is rarely used in practice.

Discussion of the Waterfall Model

The waterfall model is a heavyweight process with full documentation of each process step.

Advantages:

- Process visibility
- Separation of tasks
- Quality control at each step
- Cost monitoring at each step

Disadvantages:

In practice, each stage in the process reveals new understanding of the previous stages, which often requires the earlier stages to be revised.

The Waterfall Model is not flexible enough.

Discussion of the Waterfall Model

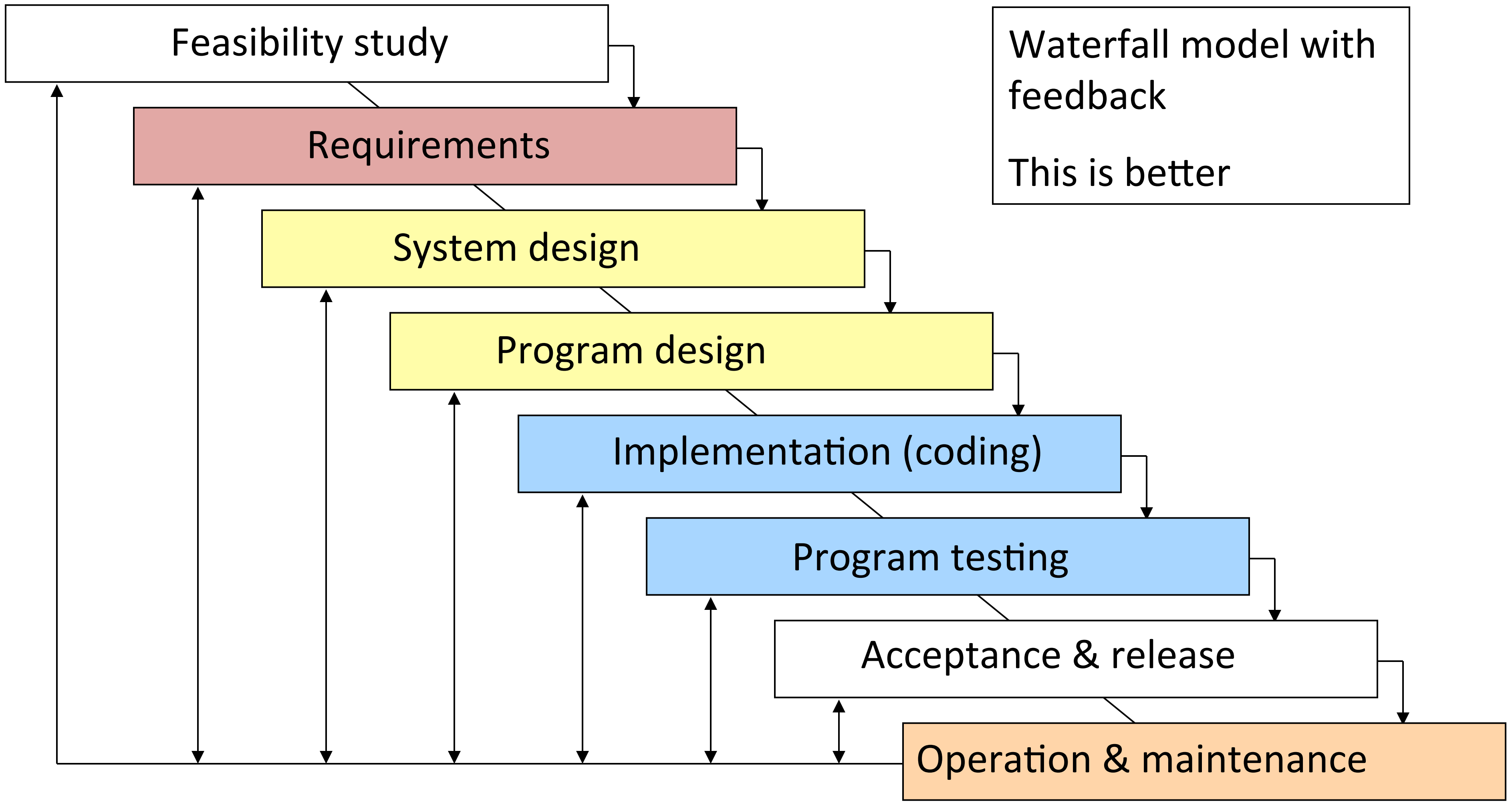
A pure sequential model is impossible

Examples:

- A feasibility study cannot create a proposed budget and schedule without a preliminary study of the requirements and a tentative design.
- Detailed design and implementation reveal gaps in the requirements specification.
- Requirements and/or technology may change during the development.

The plan must allow for some form of iteration.

Modified Waterfall Model



Waterfall model with feedback

This is better

Sequential Development

Sequential processes work best when the requirements are well understood and the design is straightforward, e.g.,

- Conversions of manual data processing systems where the requirements were well understood and few changes were made during the development (e.g., electricity billing).
- New models of a product where the functionality is closely derived from an earlier product (e.g. automatic braking system for a car).
- Portions of a large system where some components have clearly defined requirements and are clearly separated from the rest of the system.

Contracts

Note about contracts for software development

Some organizations contract for software development by placing separate contracts for each stage of the Waterfall Model or arrange for payment after each stage. This is a very bad practice.

Iterative Processes: Requirements and Risk

Concept

- Create a **prototype** system early in the development process.
- Review the prototype with clients and test it with users, to improve the understanding of the requirements and clarify the design.
- Refine the prototype in a series of iterations.

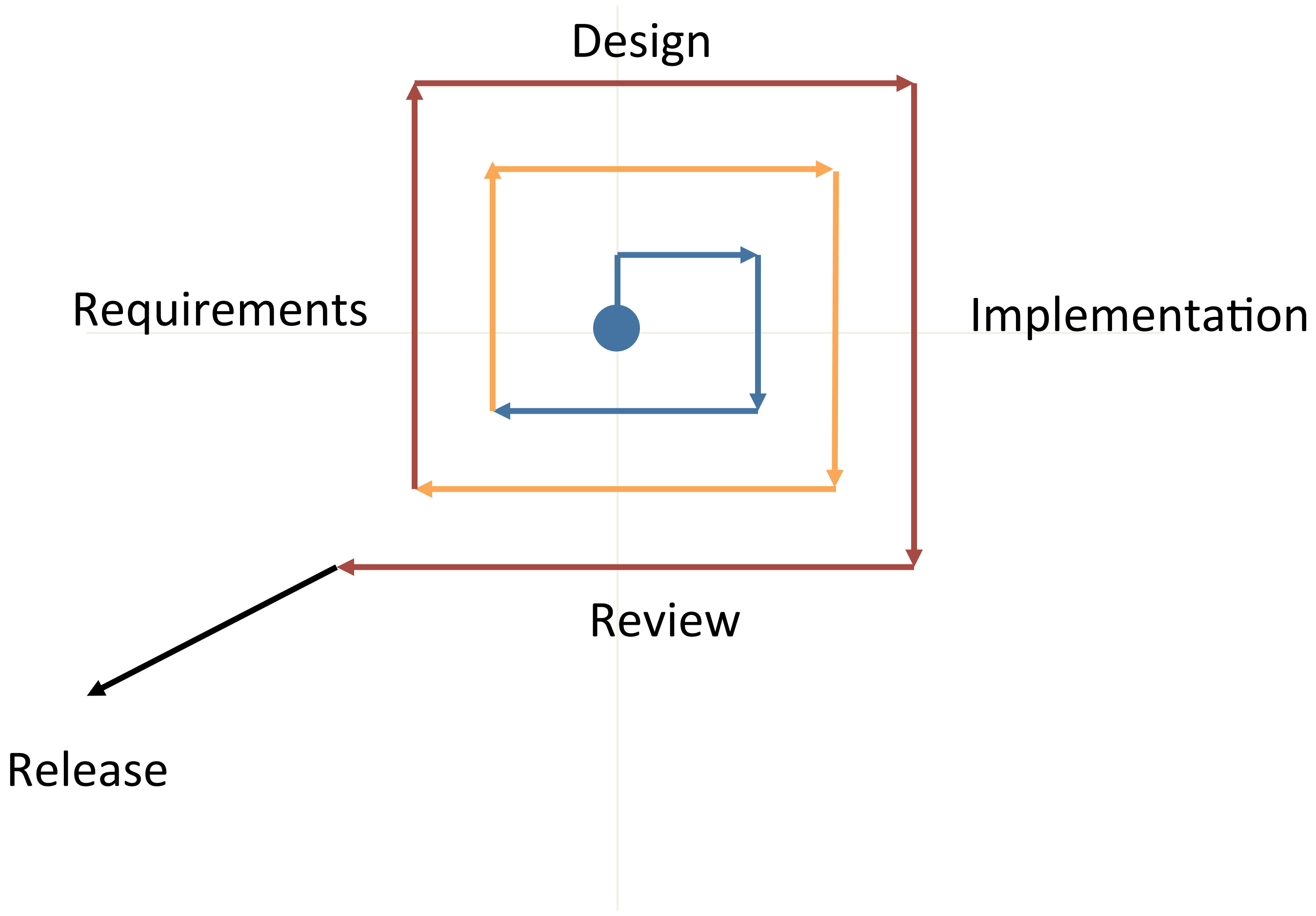
Requirements are hard to understand until there is an operational system, particularly with user interfaces.

Mistakes in the requirements are the most expensive to correct.

Example:

- Converting a national archive from paper based to computer based.

Iterative Refinement



Discussion of Iterative Refinement

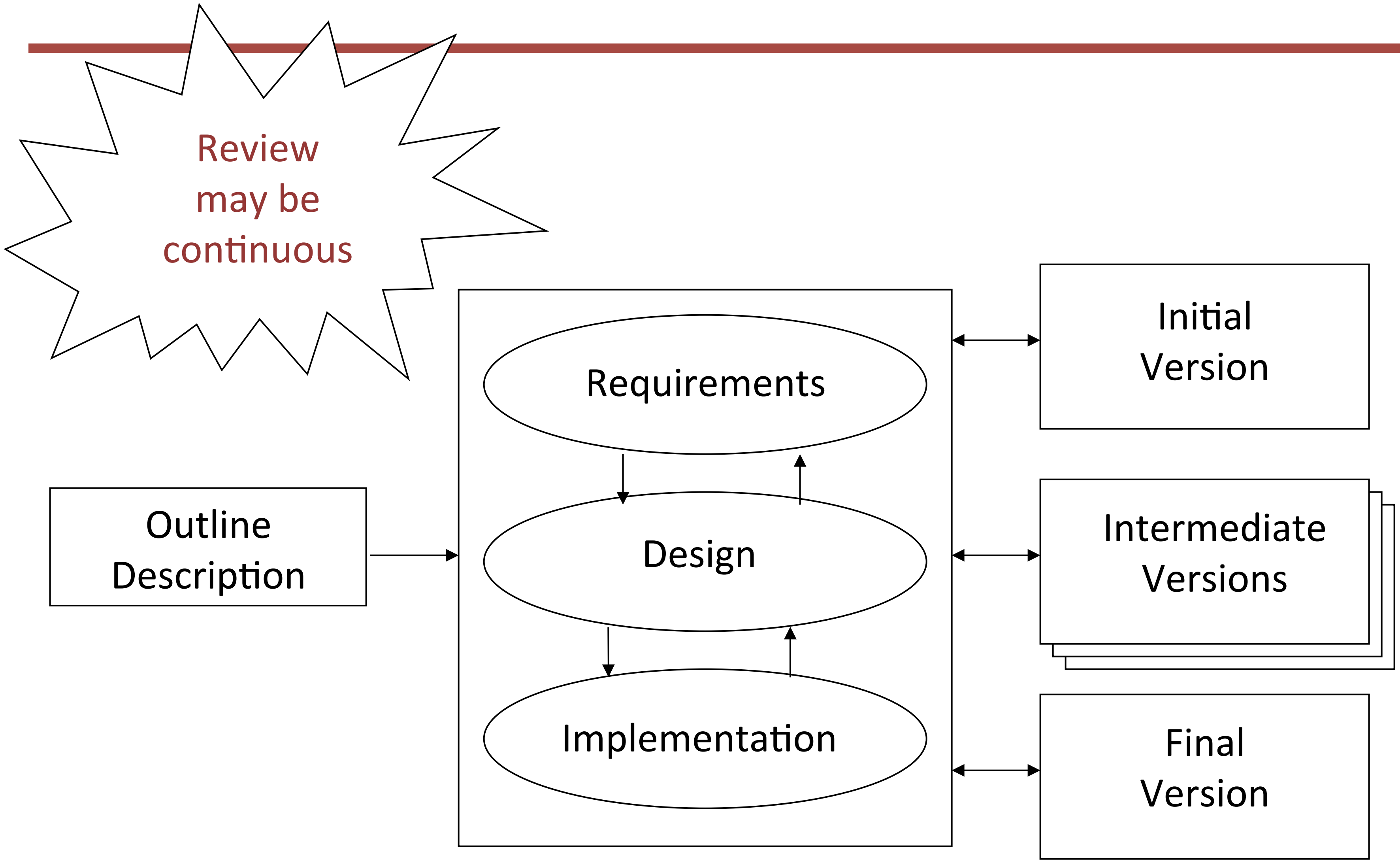
This is a medium weight process with documentation created during the process.

Iterative refinement uses various techniques that enable the client to review the the planned system early during development:

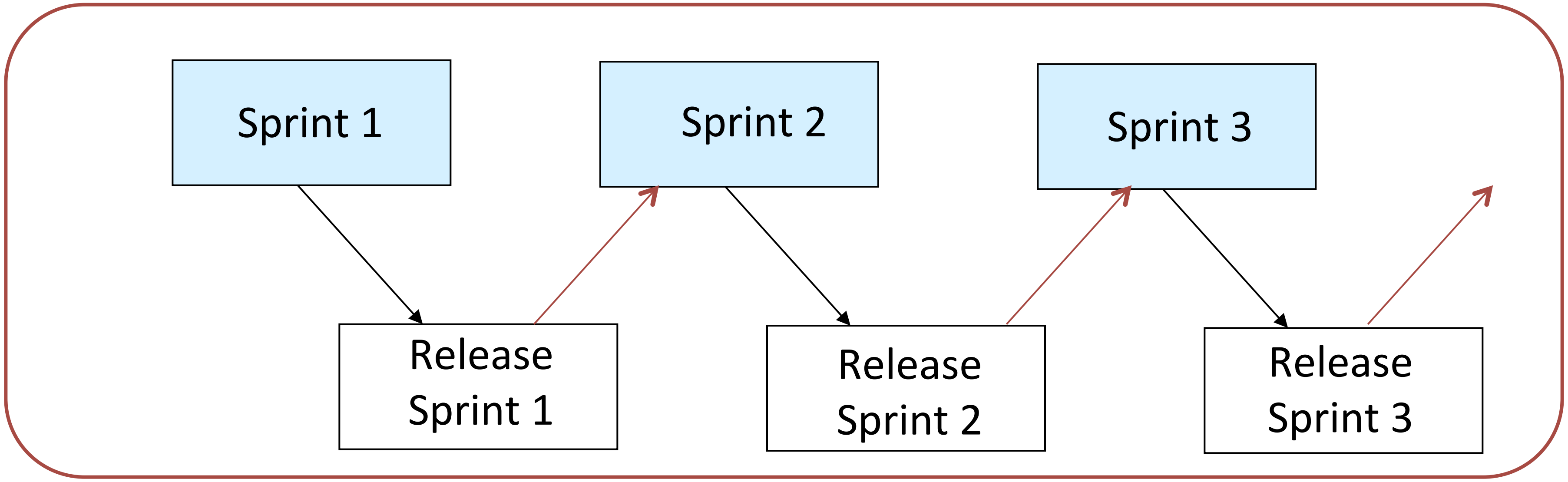
- User interface mock-up
- Throw-away software components
- Dummy modules
- Rapid prototyping
- Successive refinement

Get something working as quickly as possible, for client and user evaluation, but do not release it.

Iterative Refinement with a Large System



Incremental Development (Sprints)



- The project is divided into a large number of small tasks, known as **sprints**.
- For each sprint, a team works through a full software development cycle including planning, requirements analysis, design, coding, testing, and acceptance testing, and release.
- Each sprint is completed in a fixed time period, e.g., four weeks.
- The size of an sprint is based on team size, e.g., 5-10 people.

Incremental Development

Advantages

- Pay-back on investment begins soon.
- Requirement are more clearly understood in developing subsequent sprints – minimize waste.
- Feedback from customers and clients can be incorporated in later phases.

It is easier for small teams to develop a small print correctly than to coordinate large projects with many ramifications.

Challenge

This approach is excellent for continual enhancement of a system within an established architecture.

A high-level team must establish overall architecture and coordinate increments.

Incremental Development of Online Systems

When **software is released online** it is often possible to divide it into small increments that are developed and released in quick succession.

Example:

- Start-up company developing a web based service.

This is a lightweight process with minimal documentation created during the process.

It is not suitable for shrink wrapped software, embedded systems, or similar environments.

Mixed Processes

In practice, many large projects use processes that mix aspects of the three types of software process. For example:

- User interfaces have to be tested with users. This forces iterative development, even within an underlying sequential process.

Mixed Processes: Phased Development

Combine sequential and iterative elements

A simple system with basic functionality is brought quickly into production (Phase 1). This system may be developed using a sequential or iterative refinement.

Subsequent phases are based on experience gained from users of the previous phase.

Advantages

- Pay-back on investment begins soon.
- Requirement are more clearly understood when developing subsequent phases

Examples of Mixed Processes: Iterative Refinement + Waterfall Model

Problem: Add graphics package to a programming environment

Phase 1: Iterative refinement

Make several prototype versions by extending the current environment with a preprocessor and run-time support package. Test with users until users are pleased with function. Throw the code away.

Phase 2: Modified waterfall

Use the results of Phase 1 to specify a formal set of requirements. Write new compiler and run-time system incorporating graphics elements. Make minor adjustments to requirements as needed.

Mixed Processes: Spiral Development

In a big project, certain parts of the system may be well understood, but other parts needs iterative refinement to clarify the requirements and the design options.

Spiral development

- Create a prototype system that has the overall structure of the final product with dummy stubs for missing components.
- If a component or feature is well understood, use a sequential process to develop the final version.
- If a component or feature needs clarification, use iterative refinement to develop a series of versions.
- Add new versions of components to the prototype system when ready in such a way that there is always a functioning version of the overall system.

Corporate Processes

Large software development organizations have their own internal processes that are designed for their needs. For example:

- Amazon.com (Internet commerce) makes extensive use of sprints. Most software development is divided into increments of about four weeks elapsed time.
- Lockheed Martin (government contractor) follows a sequential process that fits with the way that the US government manages software contracts.
- SAP (business software) emphasizes the functionality that is seen by their business customers. Much of the development is suitable for a sequential process.
- Microsoft (PC software) places great emphasis on testing with a very wide variety of equipment and backward compatibility. Much of the development uses a spiral process.

Choosing a Software Process

Changes during the software development process are expensive.

- If the requirements are poorly understood, or expected to change, select a process that keeps flexibility. Iterative refinement, sprints, phased implementation.
- If a big software systems has many inter-related components, avoid major changes to the design of a system during development. Sequential process, such as the modified waterfall model.
- If the market for the software is poorly understood, use a process that gets operational software in front of customers as quickly as possible. Incremental, sprints.

Observations about Software Processes

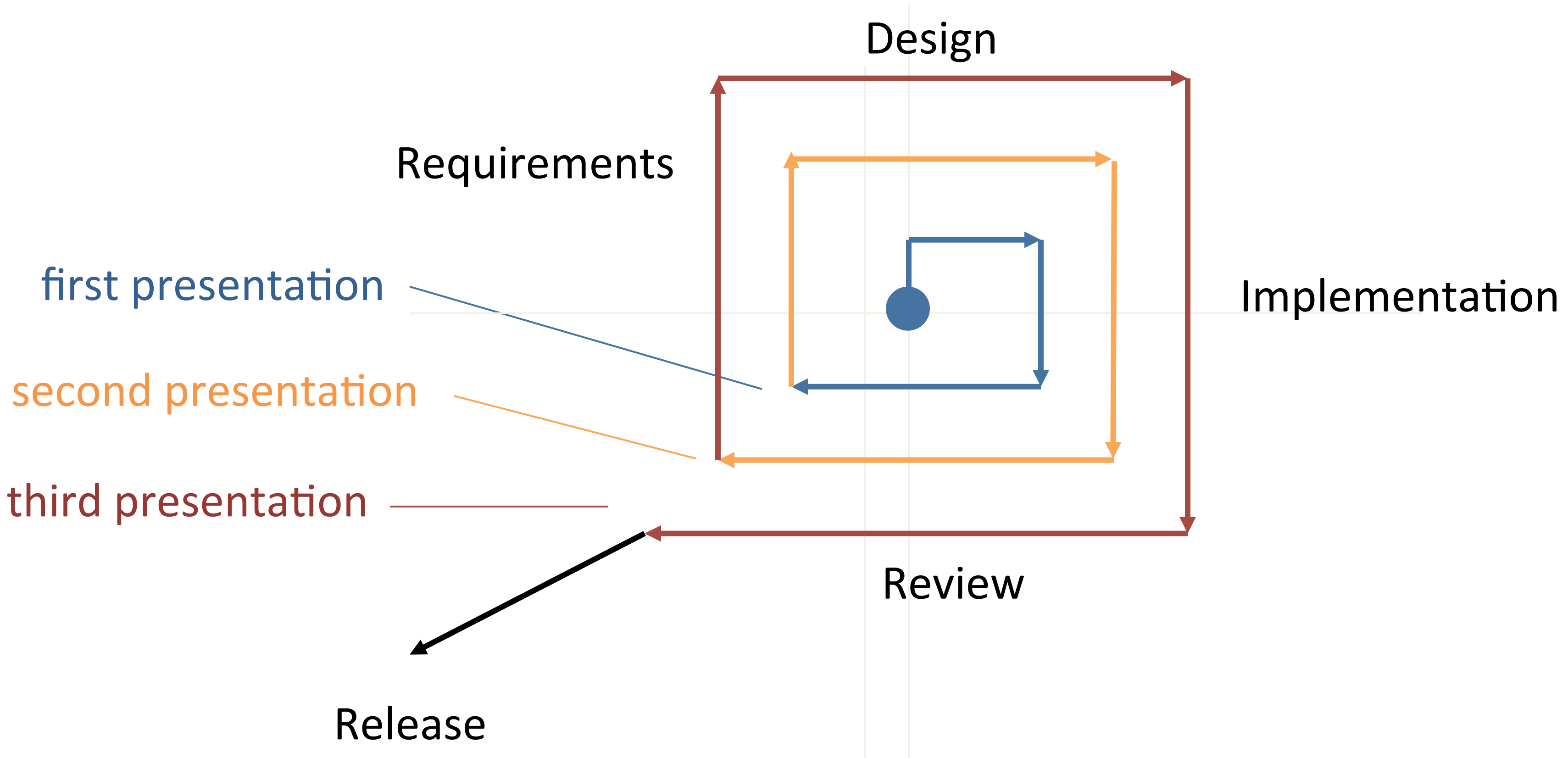
Completed projects should have included all the basic process steps *but* ... the development process is always partly evolutionary.

Risk is lowered by:

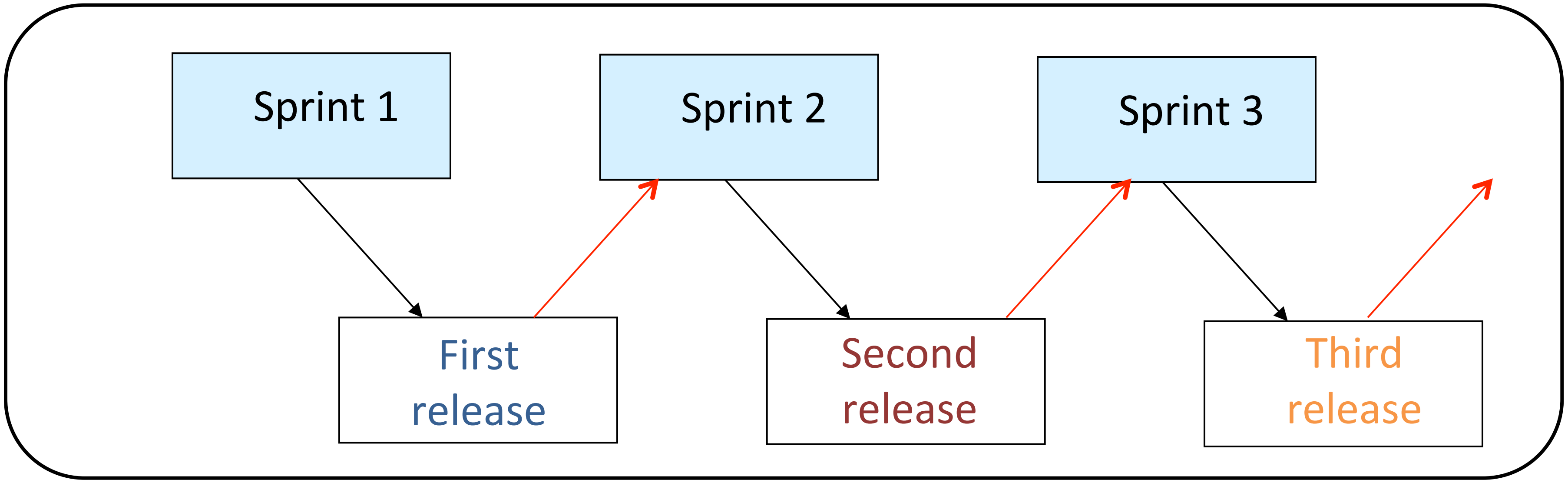
- **Prototyping** key components
- Frequent **releases**, or dividing large projects into **phases**
- Early and repeated testing with **users** and **customers**
- Following a **visible** software process
- Making use of reusable **components**

It is never possible to complete each step without provision for revision. This is known as throwing it over the wall.

CS 5150 Projects: Iterative Refinement



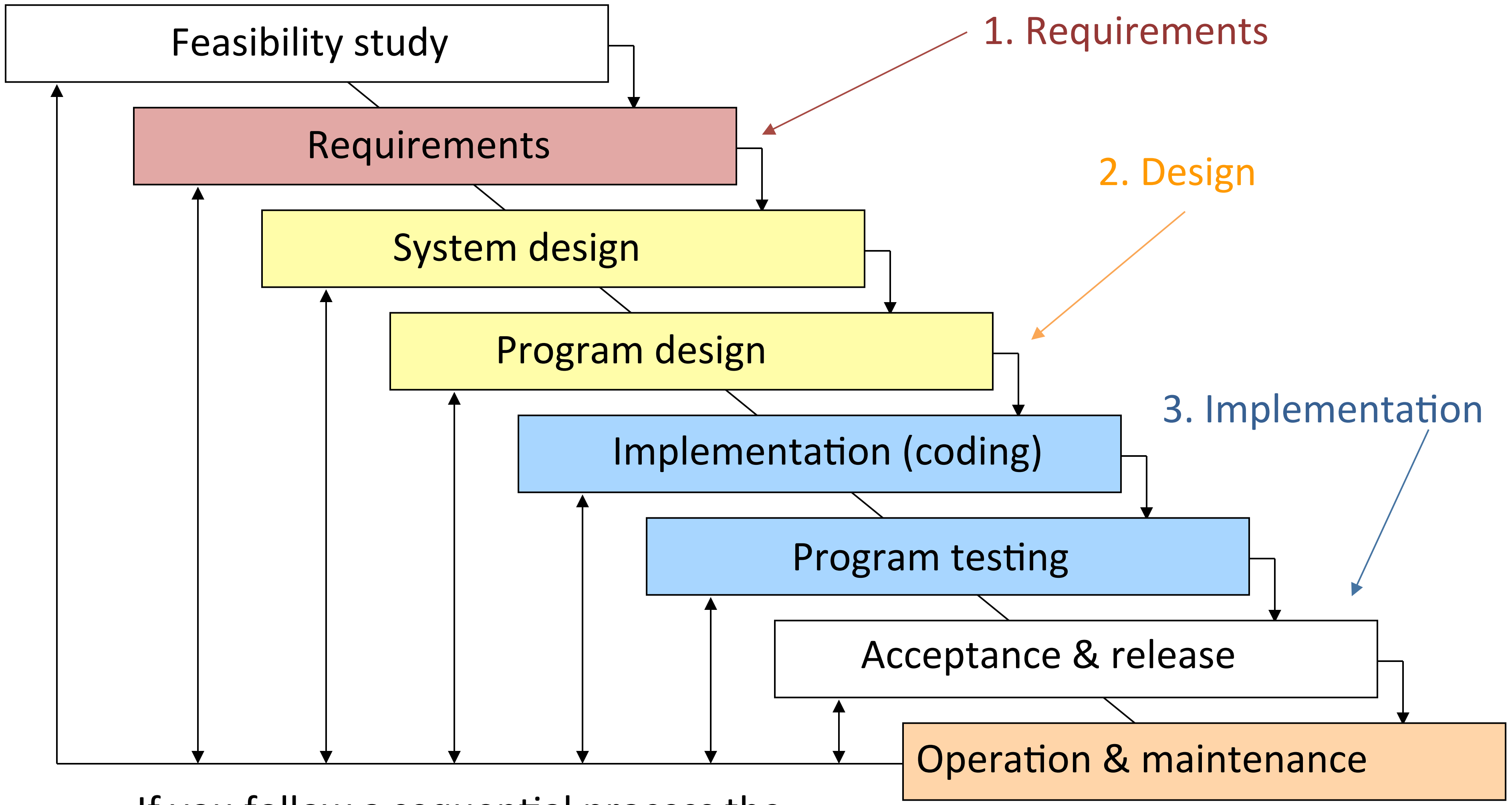
CS 5150 Project: Incremental Development



first presentation second presentation third presentation

Because of the constraints of the course, it is unlikely that you will succeed in having three releases during the semester. If you do not release the products of the first and second sprints, this becomes a hybrid between incremental development and iterative refinement.

CS 5150 Projects: Sequential Development



If you follow a sequential process the three presentations should be as shown.