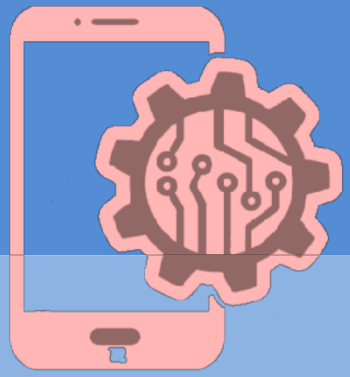
Lecture 1.3
Water Fall model

Introduction to Software Engineering



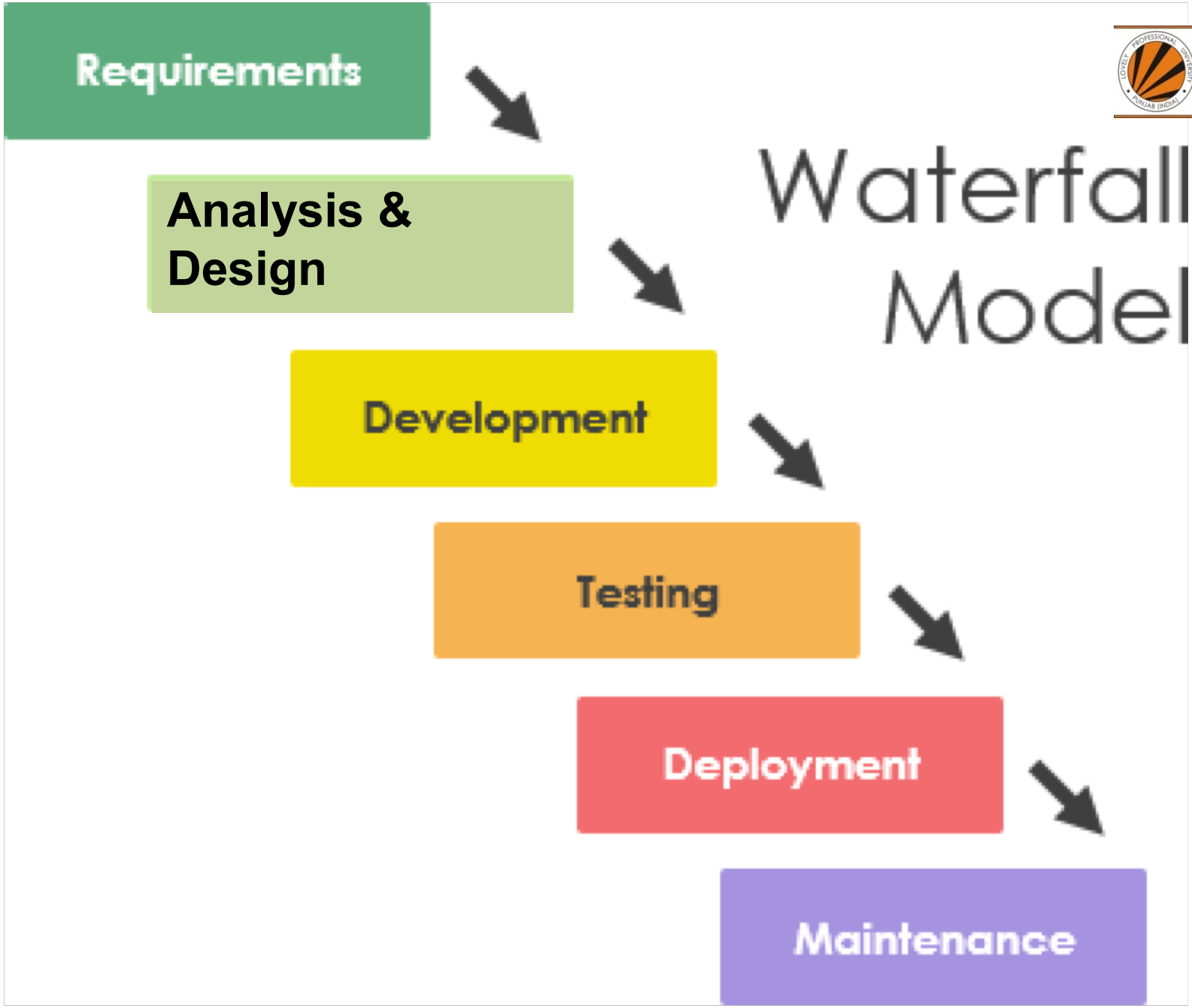
Different Process Models

- Waterfall Model (Linear Sequential Model)
- Incremental Process Model
- Prototyping Model
- The Spiral Model
- Rapid Application Development Model
- Agile Model

Water Fall Model

Water Fall Model

- Also called the *classic Software life cycle or Linear Sequential model,*
- *S*uggests a systematic, sequential approach to software development.
- It has following **Phases**.
  1. Software requirements analysis.
  2. Design
  3. Code Generation / Development
  4. Testing
  5. Deployment
  6. Maintenance


Requirements
Analysis & Design
Development
Testing
Deployment
Maintenance
Waterfall Model

System/information engineering modelling

- Work begins by establishing requirements for all system elements.
- Leads to allocating subset of these requirements to software.
- This system view is essential when software must interact with other elements such as hardware, people, and databases.

Software requirements analysis

- Is intensified and focused specifically on software.
- Analyst must understand

1. Information domain for the software,
2. Required function,
3. Behaviour
4. Performance
5. Interface.

- Requirements for both the **system** and the **software** are **documented** and **reviewed** with the customer

Design

- A multistep process that focuses on four distinct attributes of a program:

1. Data structure
2. Software architecture
3. Interface representations
4. Procedural (algorithmic) detail

- Design process:
- Translates requirements into software representation that can be assessed for quality before coding begins.
- Is documented and becomes part of the software configuration.

Code generation / Development.

- The design must be translated into a machine-readable form called code generation.
- If design is performed in a detailed manner, code generation can be accomplished mechanistically.

Testing

- Starts after code generation.
- Focus on checking:

1. Logical internals of the software:
   - ensuring that all statements have been tested/
2. Functional externals
   - conducting tests to uncover errors and ensure that defined input will produce actual results that agree with required results.

- In testing we perform both verification & validation.

Maintenance/ Support

- Software will undergo change after it is delivered to the customer.
- Change will occur because:

1. Errors have been encountered
2. Software must be adapted to accommodate:
   a) changes in its external environment
   b) customer needs for functional or performance enhancements.

- Software support/maintenance reapplies each of the preceding phases to an existing program.

Advantages

- This model is simple and easy to understand and use.
- Easy to manage due to the rigidity of the model –
- Phases are processed and completed one at a time.
- Phases do not overlap.
- Works well for smaller projects where requirements are very well understood.

Limitations:

1. **Real projects rarely follow the sequential flow that the model proposes.**

--Although the linear model can accommodate iteration, it does so indirectly.

--As a result, changes can cause confusion as the project team proceeds.

2. It is often difficult for the customer to state all requirements explicitly.

--The linear sequential model requires this and has difficulty accommodating the natural uncertainty that exists at the beginning of many projects.

3. The customer must have patience.

--A working version of the program(s) will not be available until late in the project time-span.

--A major blunder, if undetected until the working program is reviewed, can be disastrous.

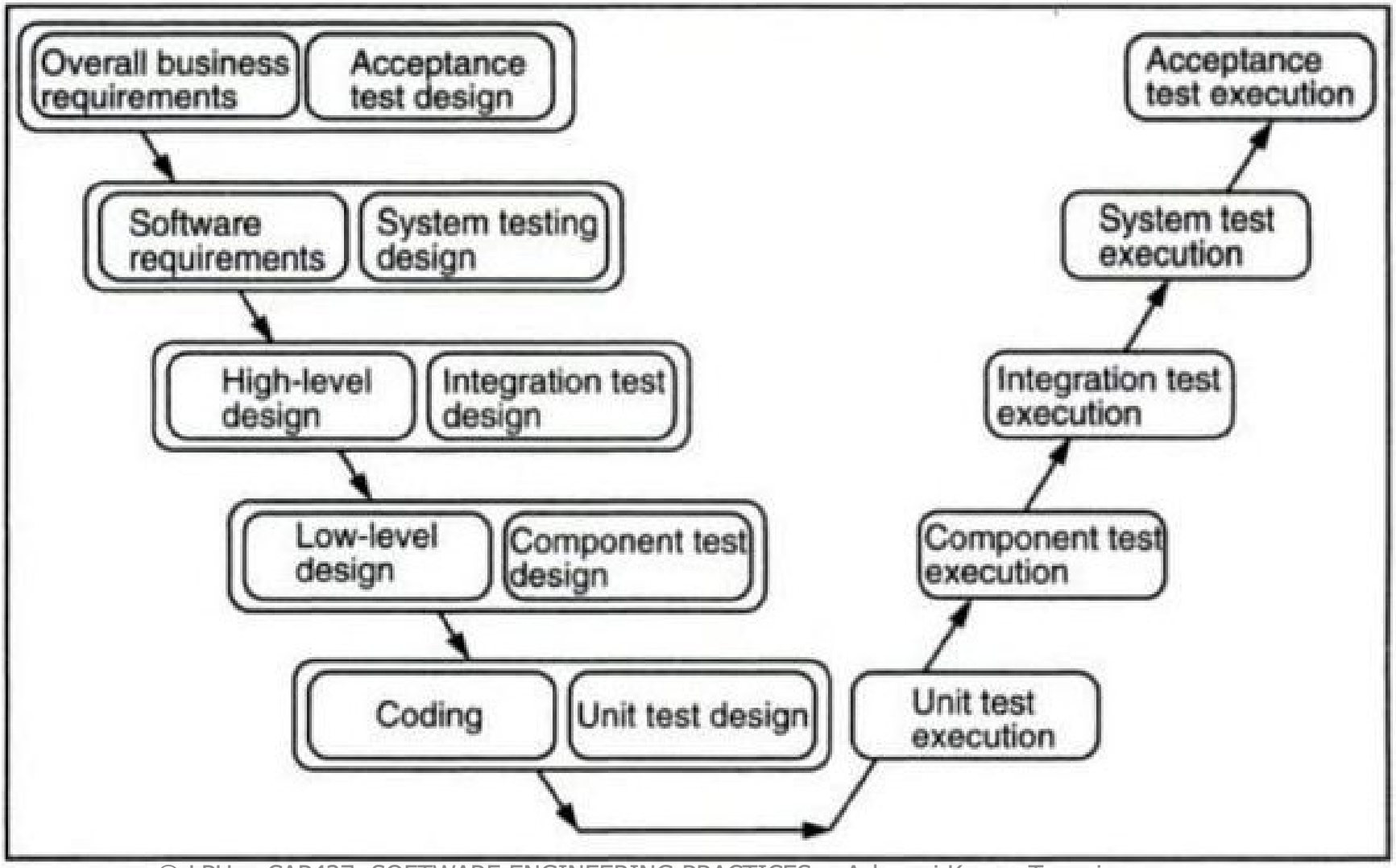
4. Freezing the requirements usually requires choosing the hardware

5. It is a document-driven process that requires formal documents at the end of each phase.

V- Model

- SDLC model where execution of processes happens in a sequential manner in V-shape.
- Also known as Verification and Validation model.
- Extension of the waterfall model
- Association of a testing phase for each corresponding development stage.
- Highly disciplined model and next phase starts only after completion of the previous phase.

V- Model


Overall business requirements
Acceptance test design
Software requirements
System testing design
High-level design
Integration test design
Low-level design
Component test design
Coding
Unit test design
Unit test execution
Component test execution
Integration test execution
System test execution
Acceptance test execution

V- Model

- **Requirements** like BRS and SRS begin the life cycle model
- Before development starts, a system test plan is created.
- Test plan focuses on meeting the functionality specified in the requirements gathering.
- **The high-level design (HLD)** phase focuses on system architecture and design.
- Provide overview of solution, platform, system, product and service/process.
- Integration test plan is created to test the pieces of the software systems ability to work together.

V- Model

- **Low-level design (LLD)** The actual software components are designed.
- Defines the actual logic for each and every component of the system.
- Component tests are created.
- **Coding phase:** All coding takes place here.
- Output:
- No. of program units,
- Individual program units need to be tested independently before they are combined to form components.

Advantages & Disadvantages of V-model:

Advantages

1. Simple and easy to use.
2. Testing activities like planning, test designing happens well before coding, saving time
3. Higher chance of success over the waterfall model.
4. Proactive defect tracking – that is defects are found at early stage.
5. Avoids the downward flow of the defects.
6. Works well for small projects where requirements are easily understood.

Disadvantages

1. Very rigid and least flexible.
2. No early prototypes of the software are produced.
3. If any changes happen in midway, then the test documents along with requirement documents has to be updated.

When to use the V-model:

- For small to medium sized projects where requirements are clearly defined and fixed.
- When ample technical resources are available with needed technical expertise.
- High customer confidence is required.