CISC/CMPE 327 Software Quality Assurance

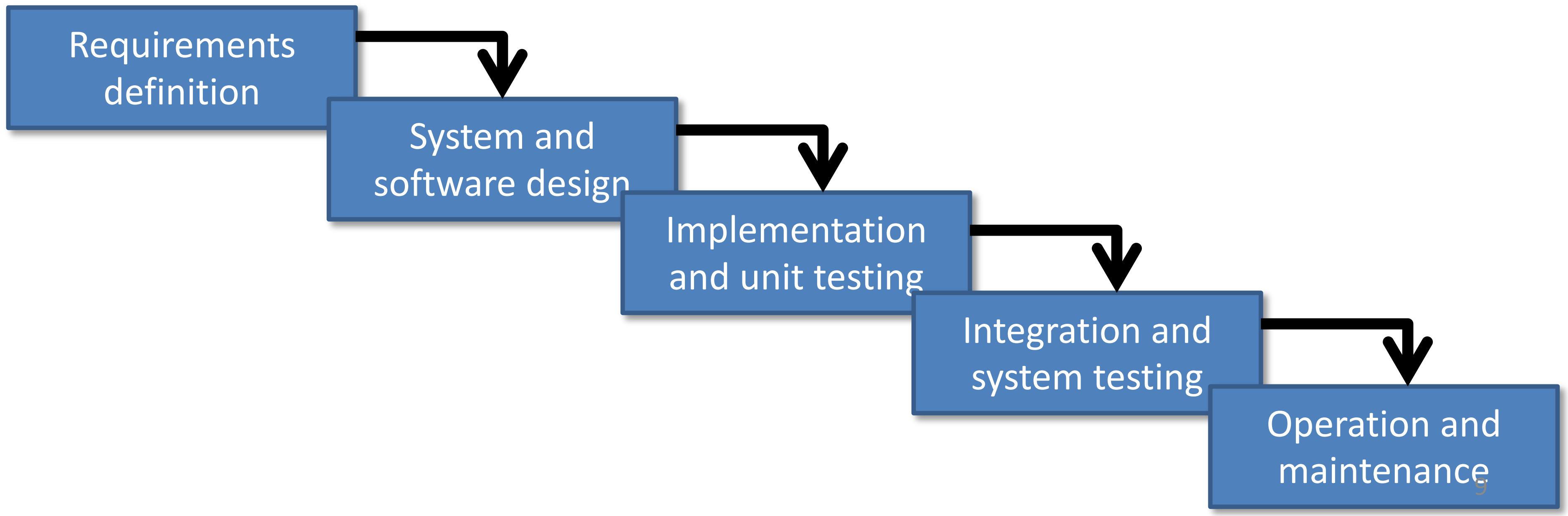
Queen's University, 2020-fall

Lecture #2
Waterfall model

CISC 327 - © 2003–2019 J.R. Cordy, S. Grant, J.S. Bradbury, J. Dunfield, S. Ding

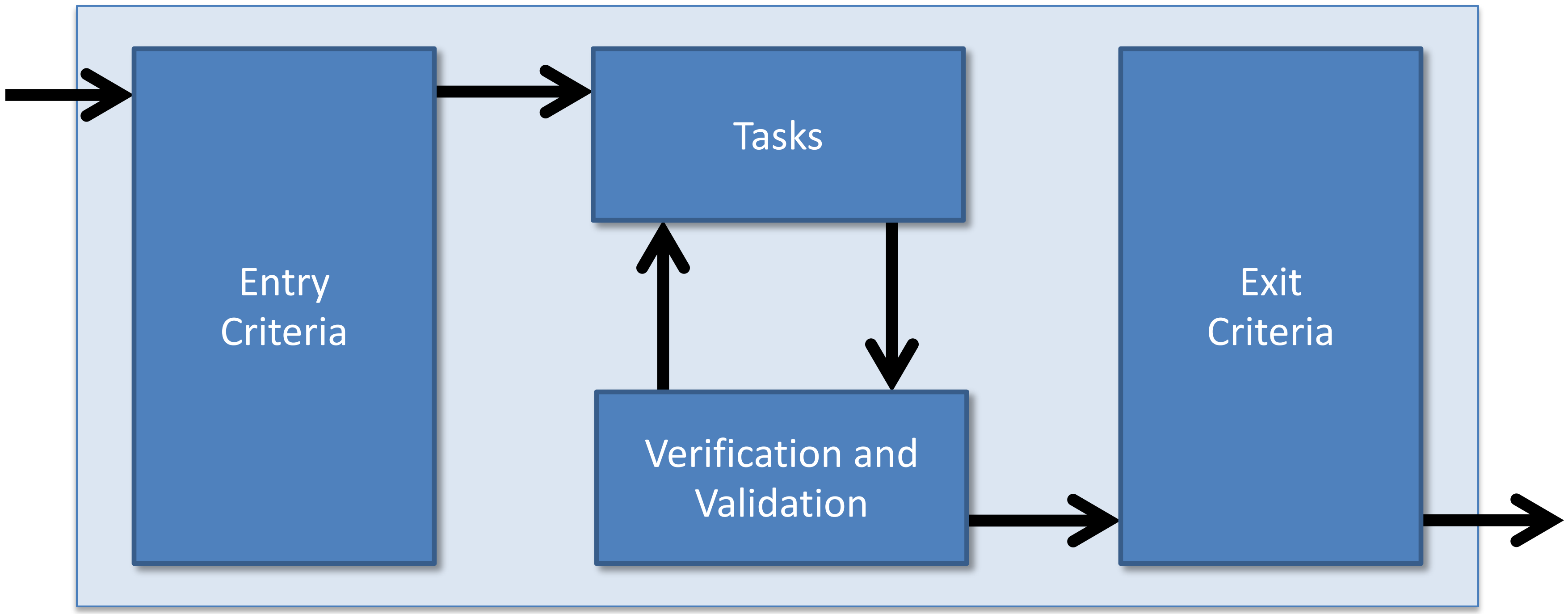
The Waterfall Model

- Original Waterfall Model
  - First explicit model, derived from other engineering processes
  - Cascade of phases, carried out in order, with sign-off of each before proceeding to the next


Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

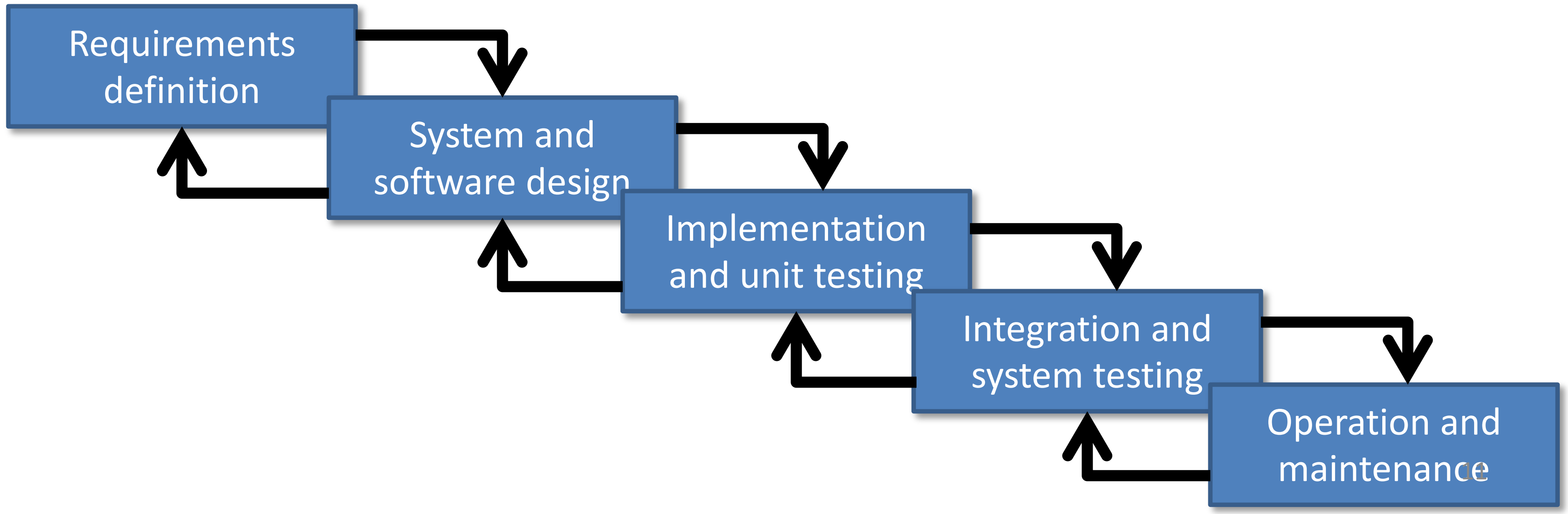
The Waterfall Model

- Organizes quality control
  - IBM's ETVX - Entry, Task, Validation, eXit at each step



The Waterfall Model

- Iterative Waterfall Model
  - Refined to be more realistic with practice
  - Go back up waterfall to revisit previous steps as necessary
  - Still work on one step at a time, cascade to next as completed


Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

The Waterfall Model

- 1. Requirements Analysis and Definition
  - System's required services, constraints, and goals are established by consultation with users/customers
  - Expressed in a way understood and agreed to by both users and developers
    - often test cases or scenarios
  - Quality control
    - requirements reviews (inspection)

The Waterfall Model

- 2. System and Software Design
  - Partitions into hardware and software subsystems
  - Establishes overall system and software architecture
  - Establishes functional specifications for components of the architecture
  - Quality control
    - Design reviews (inspection)

The Waterfall Model

- 3. Implementation and Unit Testing
  - Design realized as a set of programs and program components (units) to implement components of the architecture
  - Verify that units meet functional specifications
  - Quality control
    - Unit testing, component testing

The Waterfall Model

- 4. Integration and System Testing
  - Integrate individual programs and program units into complete system
  - Validate system that system meets requirements
  - Quality control
    - Integration testing, acceptance testing

The Waterfall Model

- 5. Operation and Maintenance
  - Normally longest phase of software life cycle
  - Install system and put into use
  - Maintenance involves correcting errors discovered in practice (“failures") and improving system units (e.g., performance tuning) and enhancing services in response to new requirements
  - Quality control
    - Regression testing, acceptance testing

The Waterfall Model

- 6. Retirement and Decommissioning
  - System is retired and replaced with a new one
  - Rarely done now because of cost and risk of replacement
    - Continuous evolution more common

Drawbacks of the Waterfall Model

- Early Freezing
  - In practice, frequent iterations back up the waterfall make it difficult to identify checkpoints and track progress
  - Therefore it is normal to freeze parts of the development, such as requirements and design, and move on to the later stages quite early without feedback

Drawbacks of the Waterfall Model

- Early Freezing
  - Premature freezing of requirements may mean that the system won't end up doing exactly what the users want
  - Premature freezing of designs often leads to badly structured systems as design problems are worked around using implementation tricks

Drawbacks of the Waterfall Model

- Early Freezing – Integration Issue

Drawbacks of the Waterfall Model

- Inflexible Partitioning
  - The inflexible partitioning into distinct stages, while a management advantage, often leads to undesirable technical results
  - Delivered systems are sometimes unusable, do not meet users' real requirements (as opposed to their original guesses)

Drawbacks of the Waterfall Model

- But…
  - The waterfall model reflects common engineering practice
  - Likely that this process model will still remain the norm for some time