CMSC 128

Introduction to Software Engineering

2nd Semester AY 2016 – 2017

4.1 – Software Development Lifecycle Models

Reginald Neil C. Recario
Institute of Computer Science
University of the Philippines Los Baños

0

CMSC 128
Introduction to Software Engineering

CMSC 128 Introduction to SE

- Game and Energizer!
- The need for a process
- How does a process differ from a model
- The Software Development Life cycle
- Software Development Models
- References
- Credits

Note

Components of this slide deck is directly lifted from the slide deck by K. A. Saleh "Software Engineering" with modifications.

1

Game and Energizer

Game and Energizer

Listen to your lecturer for instructions!

2 The need for a process

The need for a process

- What is a process?
- Why is there a need for a process?

The need for a process

- **A process is** “*a series of actions that produce something or that lead to a particular result.*”

The need for a process

A process helps in determining:

- What are done and what are not done
- What should not be done and what should be done
- What are needed and those that are not need

The need for a process

Example: What do you do to change your electives?

(volunteers, please!)

3

How does a process differ from a model

How does a process differ

The terms are typically used as if they refer to the same thing.

How does a process differ

Model	Process
○ Abstractions of processes ○ No specifications how things are done	○ Shows a specific way of creating the project ○ Implementation

4

The Software Development Life cycle

The Software Development Life Cycle

- **The software development life cycle according to Saleh (2009)** “*defines the framework under which a software product is going to be developed*”.

The Software Development Life Cycle

- A life cycle model defines, “at a high level, the phases that the product under development will go through”.

The Software Development Life Cycle

- **At a lower level,** “*the activities involved in each of the model phases and their respective deliverables are identified*”.

The Software Development Life Cycle

- The following phases are common to SDLCs:
  - Analysis
  - Design
  - Implementation
  - Testing and Integration
  - Installation/ Deployment and Maintenance

The Software Development Life Cycle

Analysis

- Identify requirements and define specifications
- Functional and Non-functional requirements
- Documents are produced (lots of them)

Analysis (cont…)

- An important phase – consider risks, assumptions, constraints, etc.

The Software Development Life Cycle

Design

- High-level database, architectural and interface
- ERD/Class Diagram, Network design, Use Case, Activity, Mockup design, prototypes, etc.

The Software Development Life Cycle

- Implementation
- Transform the design into executable code considering industry and firm standards.

The Software Development Life Cycle

Testing and Integration

- Modules and components are tested for compliance
- Modules and components that passed the test are integrated.

Installation/Deployment and Maintenance

- The final product is deployed in the client’s/end user’s environment.
- May have to be supported for peroidic updates and fixing of bugs.

The Software Development Life Cycle

End-of-Life (EOL)

- Stop use of thre product (in this case, the software)
- No support for updates and fixes

5

Software Development Models

Software Development Models

Some of the popular and well-known

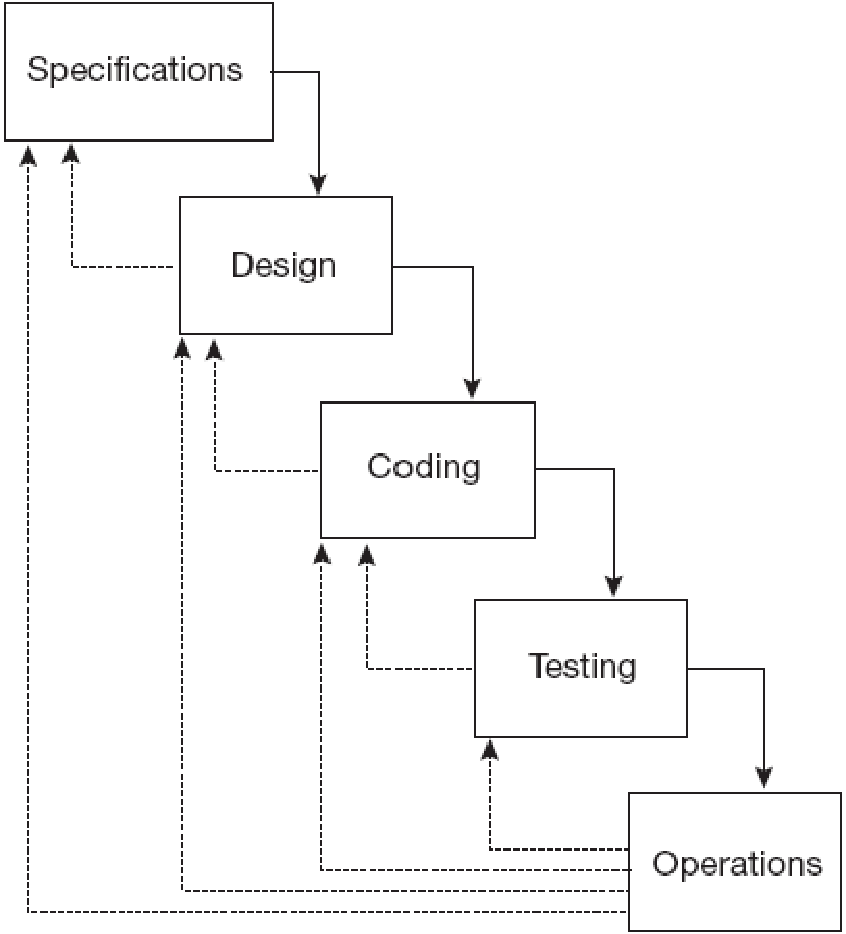
- Waterfall
- Prototyping
- Spiral
- Object-Oriented
- Incremental and Iterative
- Agile

Waterfall Model

○ Presented by Winston Royce

○ Each phase must be finished first before moving to the next phase

Software Development Models



Source: Kassem Saleh "Software Engineering"

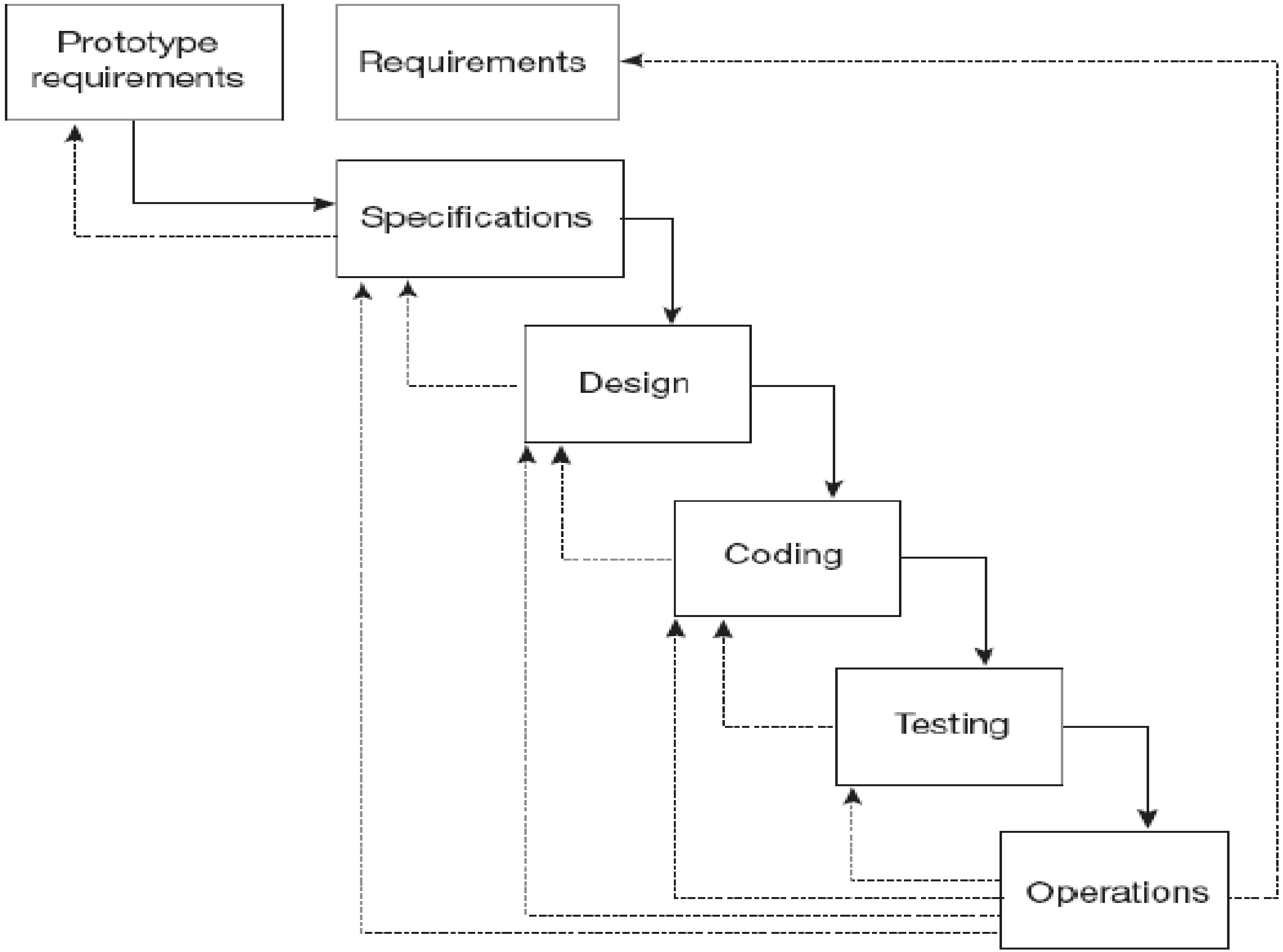
Modified for Classical Waterfall

Software Development Models

- Prototyped waterfall model
- Allows visualization of requirements
- Find out requirements errors earlier

Software Development Models

- Prototyped waterfall model (cont…)
- Better quality – user interface
- Interactive with client / user
- Easy and quick to build prototype
- Prototype to throw away – do not use it to continue building the software



Source: Kassem Saleh "Software Engineering"

Object oriented model

- Analysis – Identification of problem domain objects, object attributes and methods (operations), object relationships: aggregation (composition

Software Development Models

Object oriented model (cont…)

- inheritance and communication, scenarios using sequences of interactions

Object oriented model (cont…)

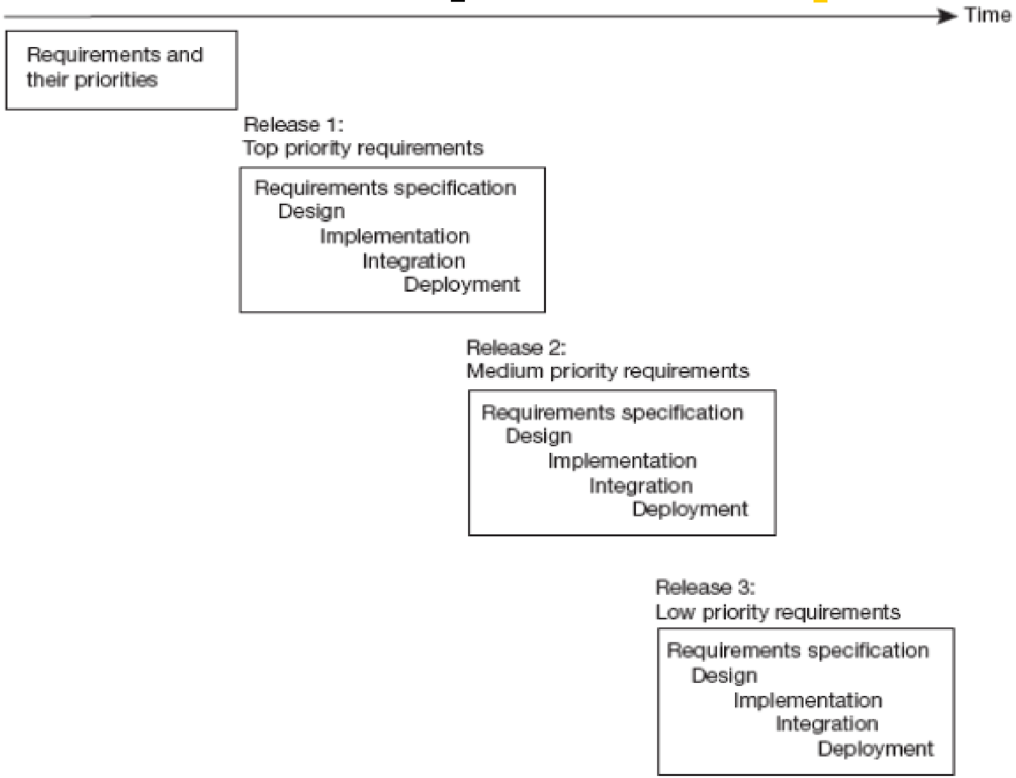
- Design – solution domain classes and interrelationships, entity, interface and control classes.
- Implementation – use Java, C++, C#
- Use the Unified Modeling Language (UML)

Software Development Models

Incremental and iterative model

- Prioritizing requirements
- Identify requirements for different releases
- Use a model to develop each release

Software Development Models



Time

Requirements and their priorities

Release 1:
Top priority requirements

Requirements specification
Design
Implementation
Integration
Deployment

Release 2:
Medium priority requirements

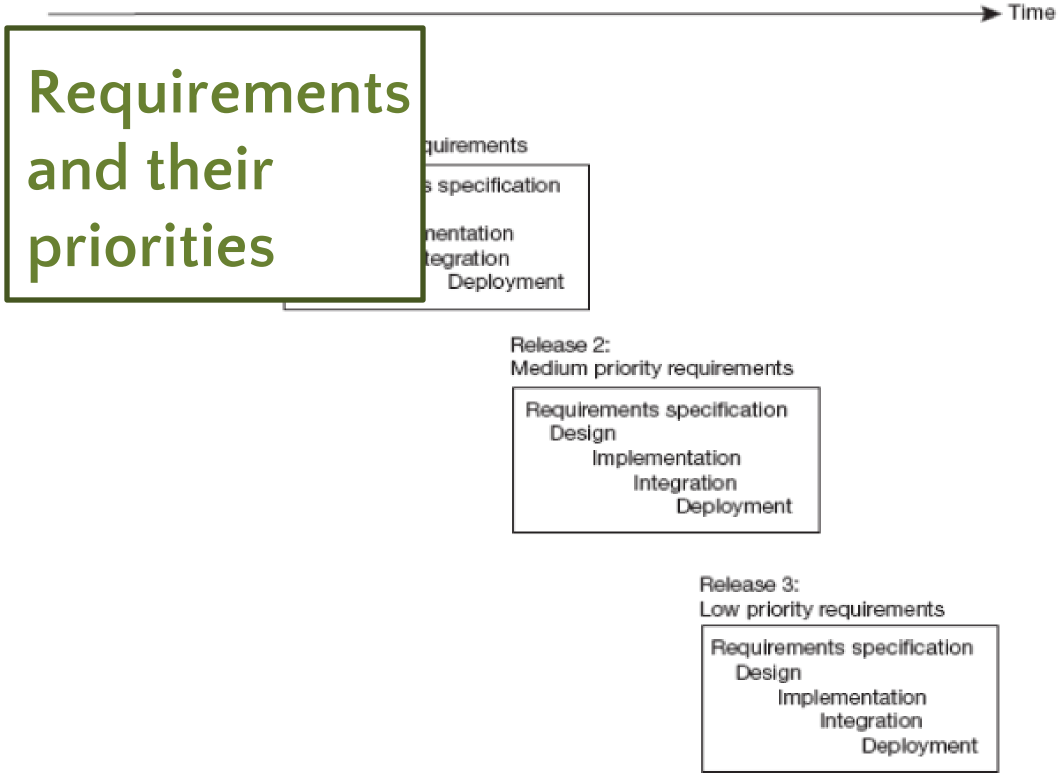
Requirements specification
Design
Implementation
Integration
Deployment

Release 3:
Low priority requirements

Requirements specification
Design
Implementation
Integration
Deployment

Software Development Models

Requirements and their priorities



Time

quirements

s specification

nentation

tegration

Deployment

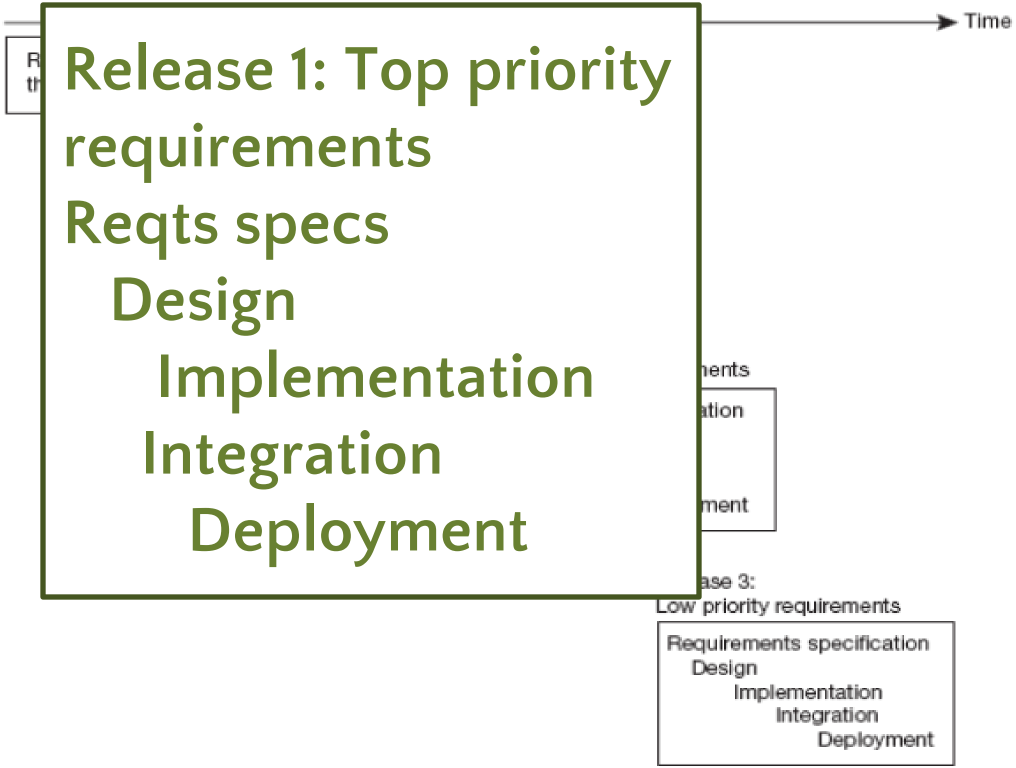
Release 2:
Medium priority requirements

Requirements specification
Design
Implementation
Integration
Deployment

Release 3:
Low priority requirements

Requirements specification
Design
Implementation
Integration
Deployment

Software Development Models



Release 1: Top priority requirements

Reqts specs

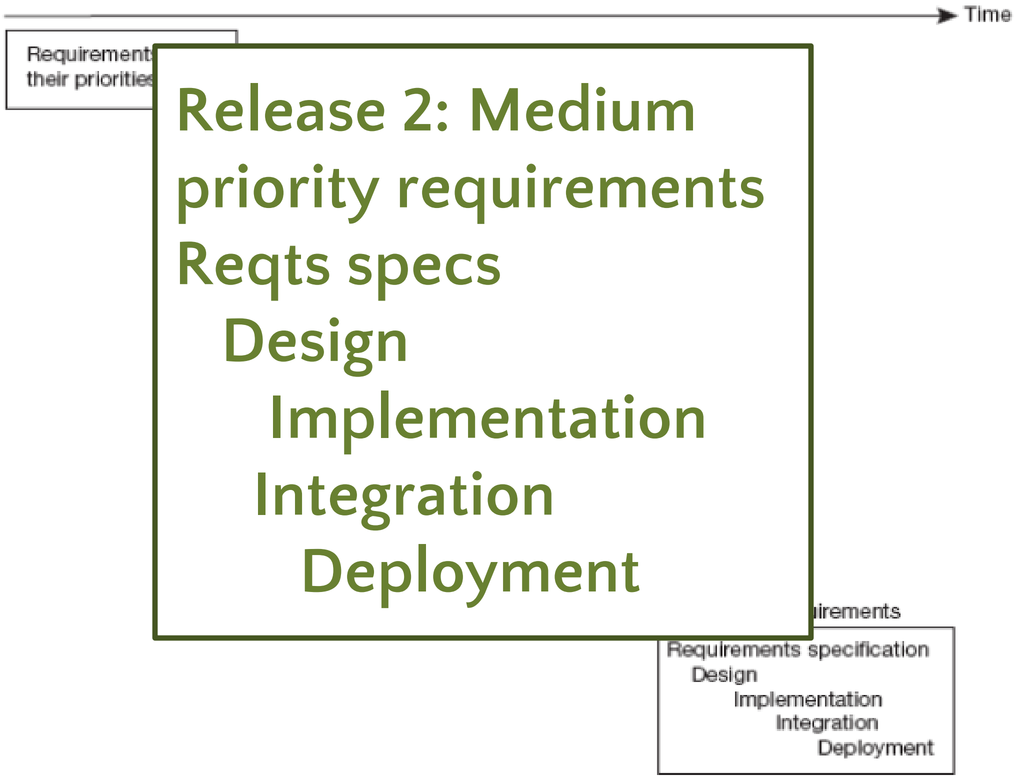
Design

Implementation

Integration

Deployment

Software Development Models



Release 2: Medium priority requirements

Reqts specs

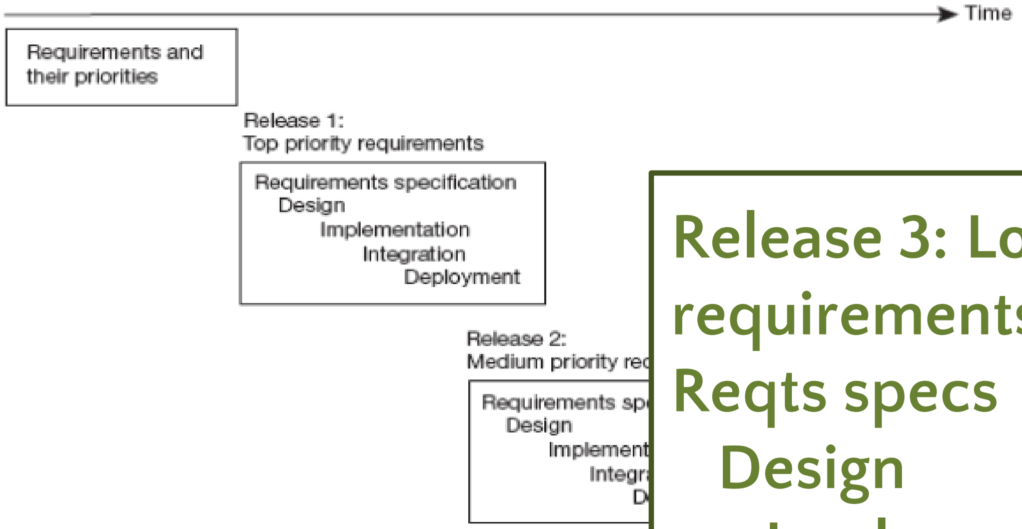
Design

Implementation

Integration

Deployment

Software Development Models



Release 3: Low priority requirements

Reqts specs

Design

Implementation

Integration

Deployment

Software Development Models

Spiral model

- Introduced by Barry Boehm in 1988.
  - Model addresses the weaknesses of the waterfall model with respect to the treatment of software development risks.

Spiral model (cont…)

- Model embeds risk management activities within the development activities.

Spiral model (cont…)

- Software risks and the lack of a clear and continuous risk management strategy are the main reason for software project failures.

Spiral model (cont…)

- Continuous consideration of risks embedded within the software development process would contribute to enhancing the quality of software.

Software Development Models

Spiral model (cont…)

- Each cycle in the spiral model involves the repeated execution of 4 steps at each phase:

Spiral model (cont…)

- (1)Identification of objectives, alternatives and constraints that are relevant at that phase

Spiral model (cont…)

- (2)Evaluation and assessment of the alternatives with respect to constraints and identified potential risks

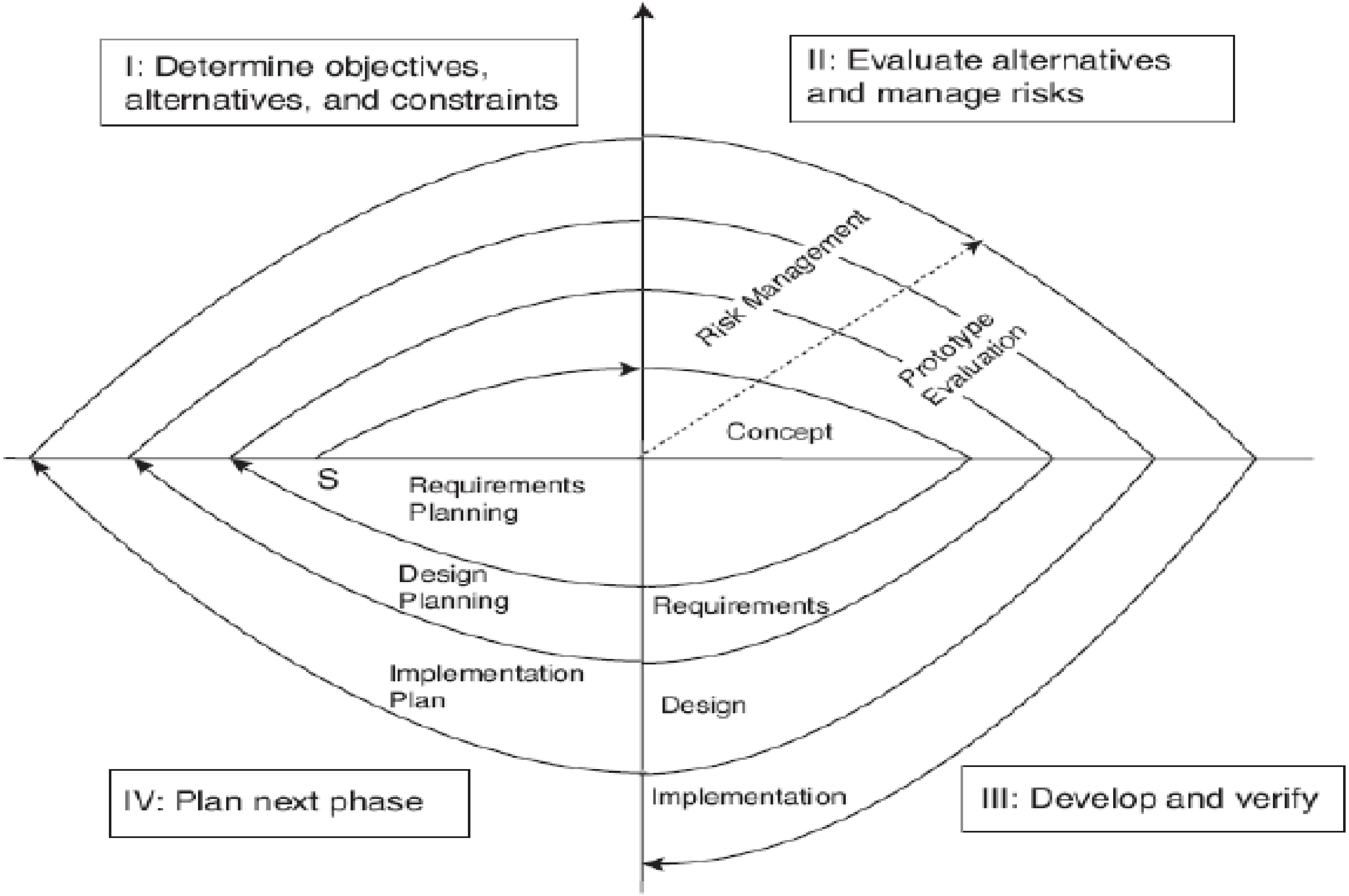
Software Development Models

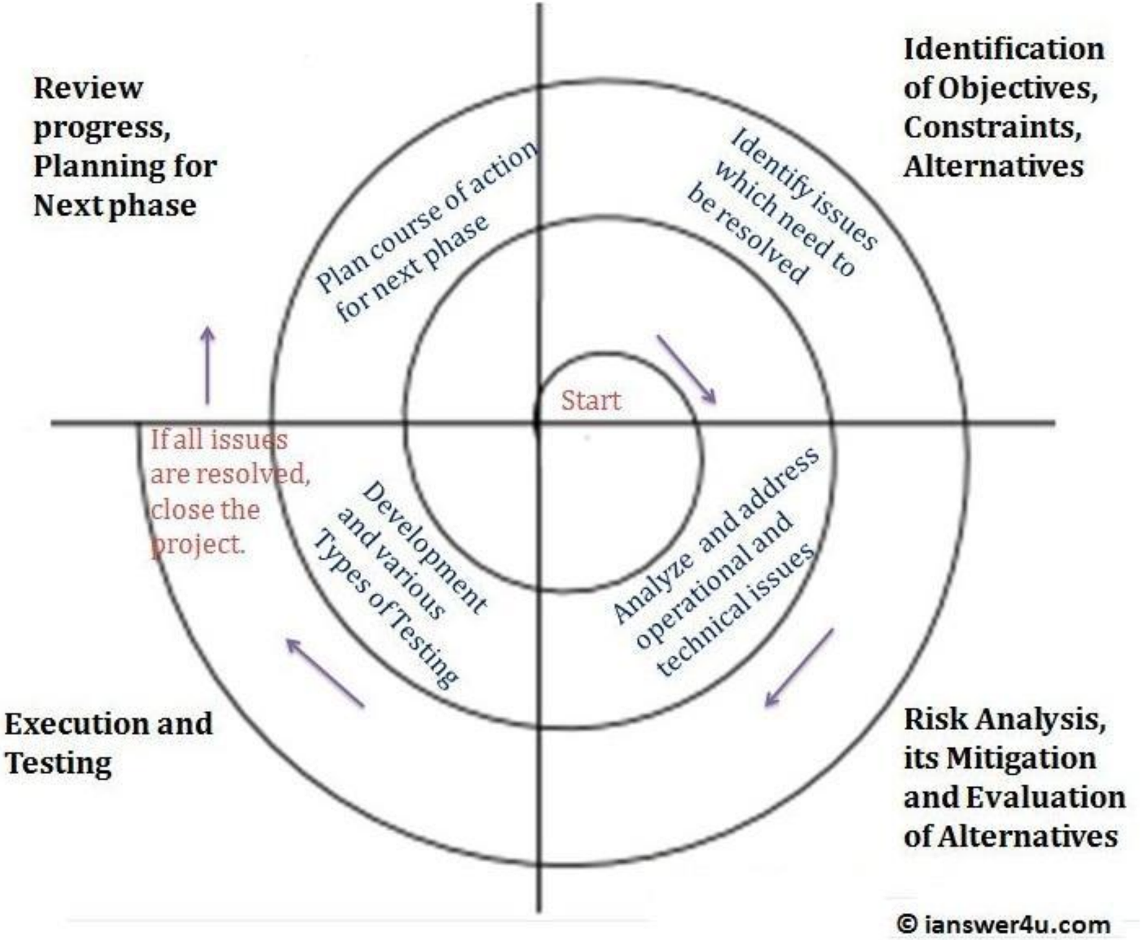
Spiral model (cont…)

- (3)Develop the phase deliverable by performing its activities, followed by a review of these deliverables

Spiral model (cont…)

○ (4) Preparation and planning for the following phase


I: Determine objectives, alternatives, and constraints
II: Evaluate alternatives and manage risks
Risk Management
Prototype Evaluation
Concept
S
Requirements Planning
Design Planning
Requirements
Implementation Plan
Design
Implementation
IV: Plan next phase
III: Develop and verify

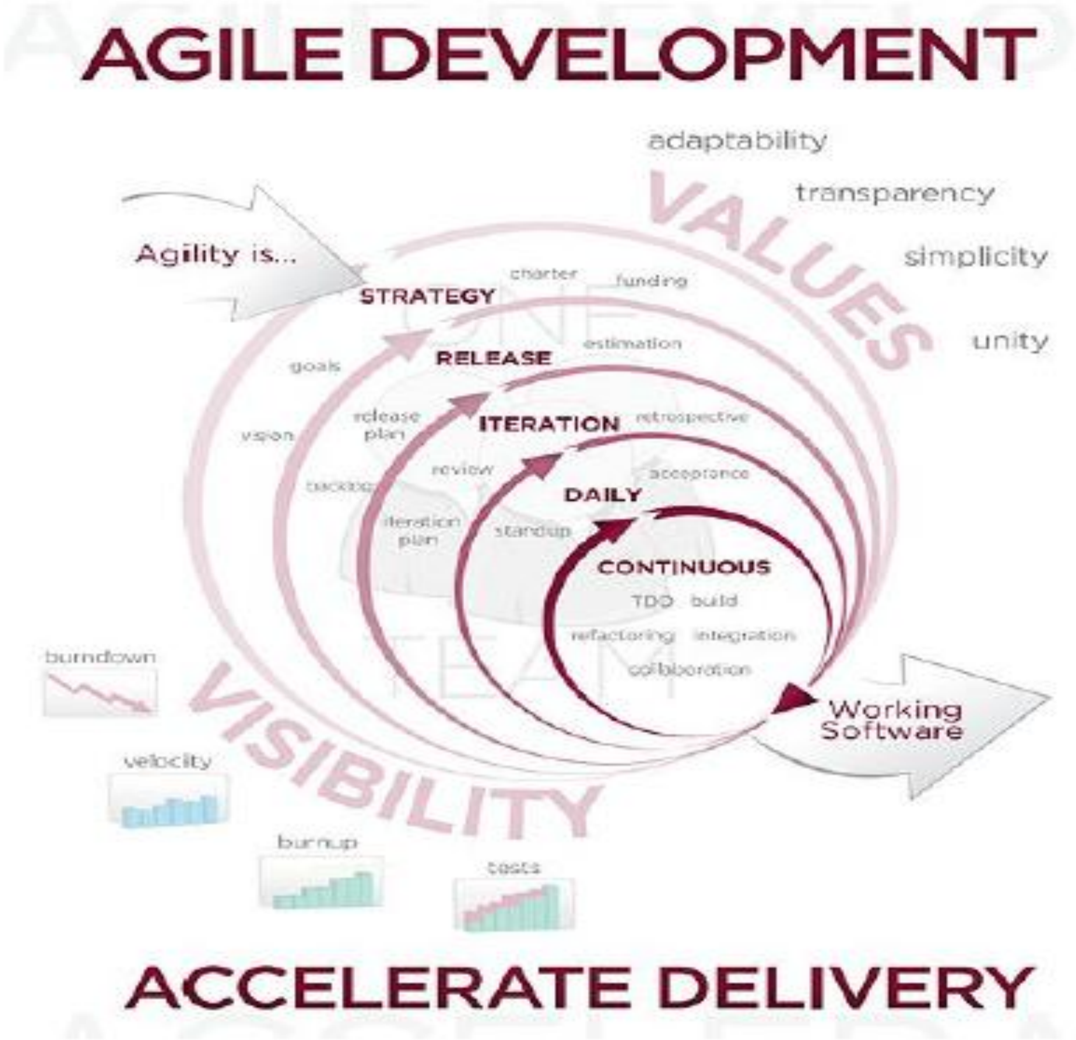

Review progress, Planning for Next phase
Identification of Objectives, Constraints, Alternatives
Plan course of action for next phase
Identify issues which need to be resolved
Start
If all issues are resolved, close the project.
Development and various Types of Testing
Analyze and address operational and technical issues
Execution and Testing
Risk Analysis, its Mitigation and Evaluation of Alternatives
© ianswer4u.com

Source: http://4.bp.blogspot.com/-DrUmxAA-0ow/Tuel4-sLo1I/AAAAAAAAAIE/MziZ7Wnu1F8/s1600/Spiral_model.JPG

Software Development Models

Lightweight Software Development Methods

- Agile Modeling
- Scrum
- Rapid Applications Development (RAD)
- Extreme Programming (XP)
- Adaptive Software Development
- Dynamic Systems Development Method
- Others


AGILE DEVELOPMENT
adaptability
transparency
simplicity
unity
VALUES
Agility is...
STRATEGY
charter
funding
goals
RELEASE
estimation
vision
release plan
ITERATION
retrospective
review
backlog
acceptance
DAILY
iteration plan
standup
CONTINUOUS
TDD
build
refactoring
integration
collaboration
ONE TEAM
Working Software
burndown
velocity
VISIBILITY
burnup
tests
ACCELERATE DELIVERY

6 References

References

Pressman, Roger. Software Engineering: A Practioner's Approach (7th ed). 2010.

Saleh, Kassem. Software Engineering. 2009.
Sommerville, Ian. Software Engin eering (10th ed). 2015.

Sommerville, Ian. Software Engineering (9th ed). 2011.

And other Software Engineering books and auxiliary online references

7

Credits

Thanks!

Any *questions* ?

You can find me at

- @RegRecario
- rcrecario@up.edu.ph

Credits

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash