SIT725 –Software Engineering

Dr Shamsul Huda

E-mail:
Shamsul.huda@Deakin.edu.au

Objectives of this Unit

- To study a pragmatic process for engineering Web-based systems and applications.
- To extend your software engineering knowledge to a new and challenging area - Web engineering.
- To encourage students to critically assess various methods and theories in software and Web engineering.
- To pave a way to do further academic research in this area, applying the methods and theories in the future professional career.

Note: this unit is not a programming learning unit

Prerequisites

- No formal prerequisites.
- Highly recommended that students have basic understanding of software Engineering and OO modelling (UML) before taking this unit.

- It’s very important to read lecture notes, reading materials and textbook after each lecture, and to participate in tutorials.

Unit Information

Unit Web Site:

CloudDeakin https://d2l.deakin.edu.au/d2l/home

Unit Guide:

- On CloudDeakin
- Important unit document
- Assessment details
- Assignment due dates
- Texts and references

Unit Assignments:

Group work. 2-3 members/group

Groups are to be formed in week 1 tutorial class

Unit Information

Unit Web Site:

CloudDeakin http://www.deakin.edu.au/clouddeakin

Assignment 1

Web application analysis. Students will analyse an existing Web application using Web engineering principles and methods. It is to be completed as a group.

Task: 20%

A written analysis report is to be submitted. No length and word number limitations, but the report should cover the required aspects and provide technical details.

Due date: 5:00pm (AEST) Monday, 20 August

Unit Information

Unit Web Site:

CloudDeakin https://d2l.deakin.edu.au/d2l/home

Assignment 2

Web application design. This work is based on your analysis in assignment 1. It is to be completed as a group.

Task: 20%

A written design report is to be submitted. No length and word number limitations, but the report should cover the required aspects and provide design details.

Due date: 5:00pm (AEST), Monday, 17 September

Final Examination: 60%

2 hour examination

Miscellanea

- Make sure you can access CloudDeakin and your Deakin e-mail without problems

- If you have any problems in using CloudDeakin, please visit CloudDeakin Help on CloudDeakin

- ***Any questions?***

❑ Background

- ❑ The Web has become an indispensable technology
- ❑ It has changed ways of business, communication, education, entertainment, finance, government, industry, media, …
- ❑ The changes are supported by a **vehicle** that acquires information, structures it, builds a package presentation, and delivers it to the users.
- ❑ The vehicle is called a **Web Application** (WebApp). WebApps have evolved into sophisticated computing tools today.
- ❑ It has become crucial to use professional process – **Web**
- ❑ **Engineering**, to develop WebApps that satisfy users' needs.

Background

- But most Web developers fail to recognize that characteristics and requirements of WebApps significantly differ from traditional software, and so does their development.
- Web developers continue to view WebApps as just simple Web page creation using HTML, embodying few images and hyperlinking documents and Web pages, or as Internet/Web programming (scripting).
- They overlook system-level requirements and don’t make use of Web design and development methodologies and processes.
- They also mistakenly carry out WebApps’ development in the same manner as traditional software development.
- The poor design of WebApps is the consequence.

Background

- There is more to WebApp development than visual design and user interface.
- WebApp development involves:
  - Planning
  - Selection of an appropriate Web architecture
  - System design
  - Page design
  - Coding
  - Content creation and its maintenance
  - Testing
  - Quality assurance
  - Performance evaluation
- Web developers need to adopt a disciplined development process and sound design methodologies (Web Engineering)

Major Unit Topics

- Web application *characteristics* and Web engineering *process*
- Web engineering *formulation and planning*
- Web application *analysis models and methods*
- Web application *design and related issues*
- Web application *test process and methods*
- *Future directions*

Introduction to Web Engineering

- WebApps are computer software

- WebApps are

  -a collection of executable instructions

  - and data that provide both information and functionalities for end users.

- Web Engineering (WebE) is concerned about

  - the establishment and use of sound scientific, engineering, and management principles;

  - disciplined and systematic approaches

to the successful development, deployment, and maintenance of high quality Web-based systems and applications

WebApp Attributes—I

- **Network intensiveness.** A WebApp resides on a network and must serve the needs of a diverse community of clients.
- **Concurrency.** A large number of users may access the WebApp at one time; patterns of usage among end-users will vary greatly.
- **Unpredictable load.** The number of users of the WebApp may vary by orders of magnitude from day to day.
- **Performance.** If a WebApp user must wait too long (for access, for server-side processing, for client-side formatting and display), he or she may decide to go elsewhere.

WebApp Attributes—II

- **Availability.** Although expectation of 100 percent availability is unreasonable, users of popular WebApps often demand access on a “24/7/365” basis.
- **Data driven.** The primary function of many WebApps is to use hypermedia to present text, graphics, audio, and video content to the end-user.
- **Content sensitive.** The quality and aesthetic nature of content remains an important determinant of the quality of a WebApp.
- **Continuous evolution.** Unlike conventional application software that evolves over a series of planned, chronologically-spaced releases, Web applications evolve continuously.

WebApp Attributes—III

- **Immediacy.** WebApps often exhibit a time to market that can be a matter of a few days or weeks.
  - With modern tools, sophisticated Web pages can be produced in only a few hours.
- **Security.** In order to protect sensitive content and provide secure modes of data transmission, strong security measures must be implemented throughout the infrastructure that supports a WebApp and within the application itself.
- **Aesthetics.** When an application has been designed to market or sell products or ideas, aesthetics may have as much to do with success as technical design.
  - content (visual, videos, audios, texts, graphics, animations), interactions and functionalities that inspires the taste of target audience

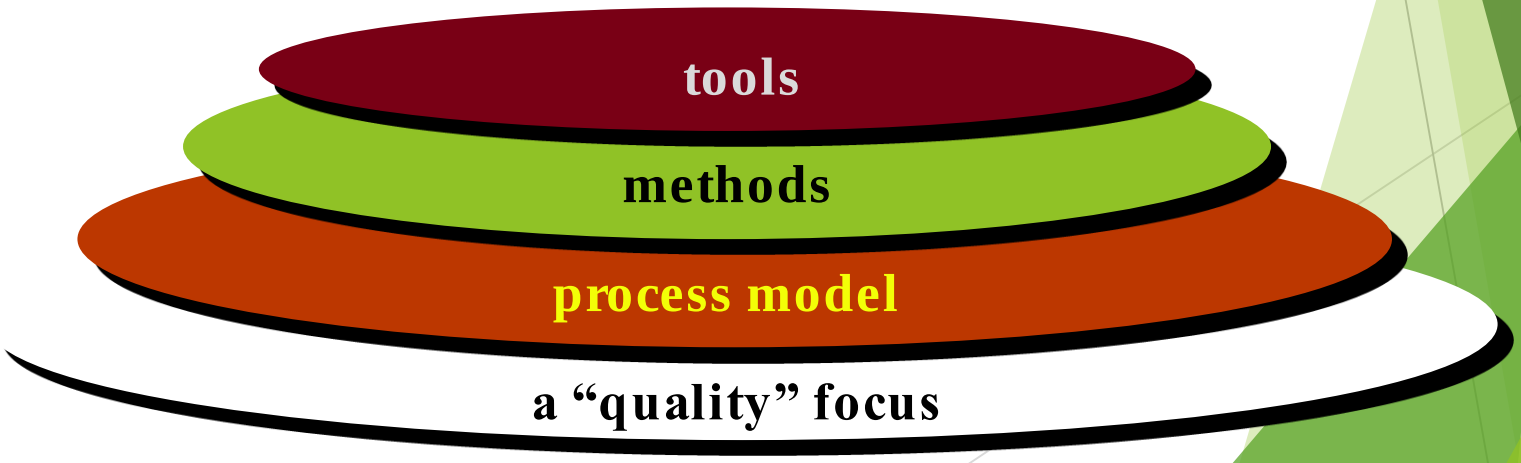
WebApp Categories

- ***informational***—read-only content is provided with simple navigation and links
- ***download***—a user downloads information from the appropriate server
- ***customizable***—the user customizes content to specific needs
- ***interaction***—communication among a community of users occurs via chatroom, bulletin boards, or instant messaging
- ***user input*** —forms-based input is the primary mechanism for communicating need
- ***transaction-oriented***—the user makes a request (e.g., places an order) that is fulfilled by the WebApp
- ***service-oriented***—the application provides a service to the user, e.g., assists the user in determining a mortgage payment
- ***Portal***—the application channels the user to other Web content or services outside the domain of the portal application
- ***database access*** —the user queries a large database and extracts information
- ***data warehousing*** —the user queries a collection of large databases and extracts information

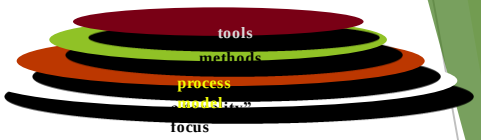
The WebE Layers

- The layered approach to WebE is conceptually identical to the software engineering layers.
  - However, the layers of software engineering must be adapted to accommodate the special characteristics of WebApps

- Please read section: ***The Components of Web Engineering*** of the textbook for details of SE layers (page 17-18)



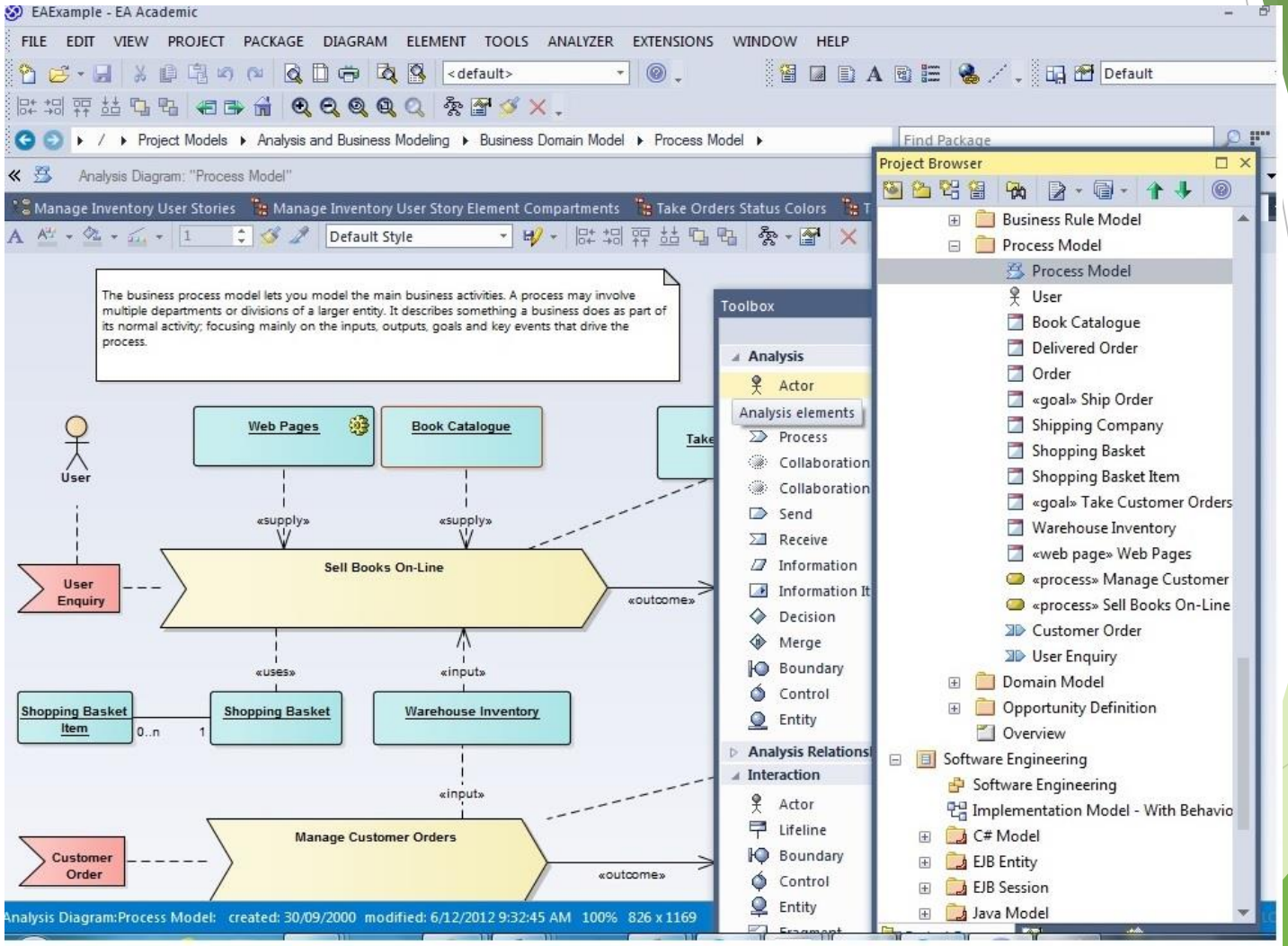
The WebE Layers



- Process layer:
- The process layer consists of a set of framework activities and the sequences of their execution which are accomplished to create a product.
  - Enables rational and timely development of software
  - Forms the basis for management control of software projects,
  - Establishes the context in which
    - Technical methods are applied
    - Milestones are established
    - Work products are produced (models and documents, actual implemented product/code)
    - Quality is ensured
    - Project tracking and control and change are properly managed
  - Process layer describes how the framework activities will be conducted and guides towards a particular process model
- Methods:
  - This provides the technical detail how-to's for building software
  - This encompasses a broader array of actions and tasks that include
    - Communication
    - Requirement analysis
    - Design modeling
    - Program construction and testing
    - support
- Tools:
  - This provides automated or semi automated support for the process and the method layers

The WebE Layers

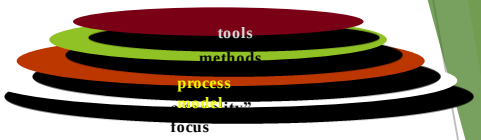
- **Tools**



The WebE Layers

- Set of process elements required for a project can vary from project to project
- Each set of process element forms a particular type of process model.
- There can be many alternative process models
- All software process models can accommodate the generic framework activities
- Each process model has a process flow :
- Process flow describes a sequence of action/ activities that are performed under the framework activities
- The final product usually needs to be corrected, refined or replaced, this requires execution of all framework activities again from the beginning.
- Thus the development process runs as a life-cycle.
- This cycle is known as a Software development life cycle (SDLC) and the steps are known as the phases of SDLC.
- [Reference-2]

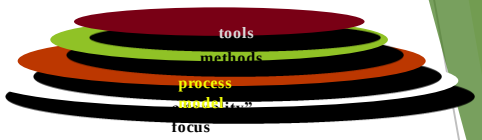
The WebE Layers



- Communication
  - Communicate with Stakeholder
  - identifying which project need to be automated
  - performing a risk analysis associated with the project
  - conducting a feasibility study of the project

- Requirement analysis
  - collect information from the project owner through discussion
  - gather information from business documents
  - interview the existing system's users
  - define a high level features of the project
  - prepare the detail features and specify the requirements
    - using different modelling approaches and computer aided software engineering (CASE) tools

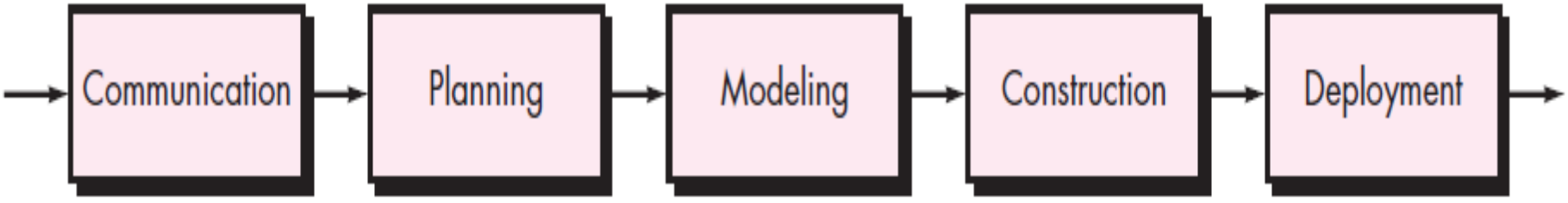
The WebE Layers



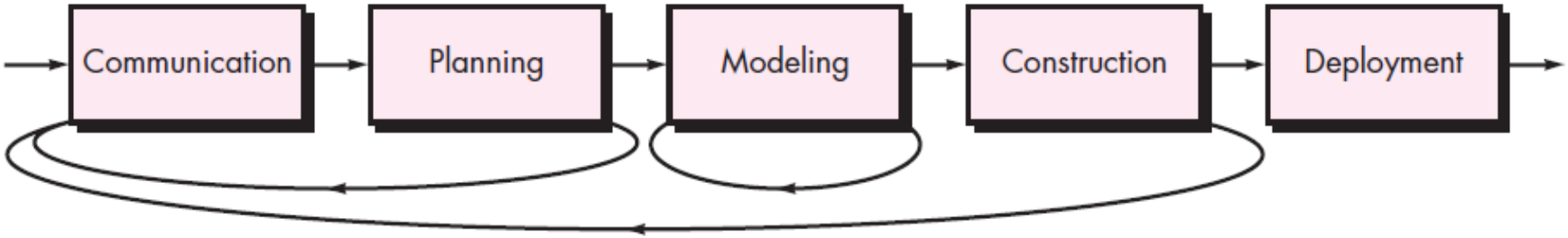
- Design modeling
  - A technical detail of the requirement specifications of the new system is prepared.
  - Involves designing the
    - application architecture
    - network architecture
    - user interface
    - Database
    - Navigation architecture and functional design
- Program construction and Testing
  - Design specification is used to write the program for construction
  - The purpose of the testing is to confirm that the system satisfies requirements
- Support and maintenance
  - New systems require monitoring, fixing bugs, modifications or minor adjustments and support while they are in operation.
  - New system also may require major changes due to evolving business goal and policies.
  - Changes in business policies may happen due to changes in the market conditions or changes in the government policies which organisations need to comply with.
  - Some changes may require major upgrades of the system.

The WebE Layers
Different process models

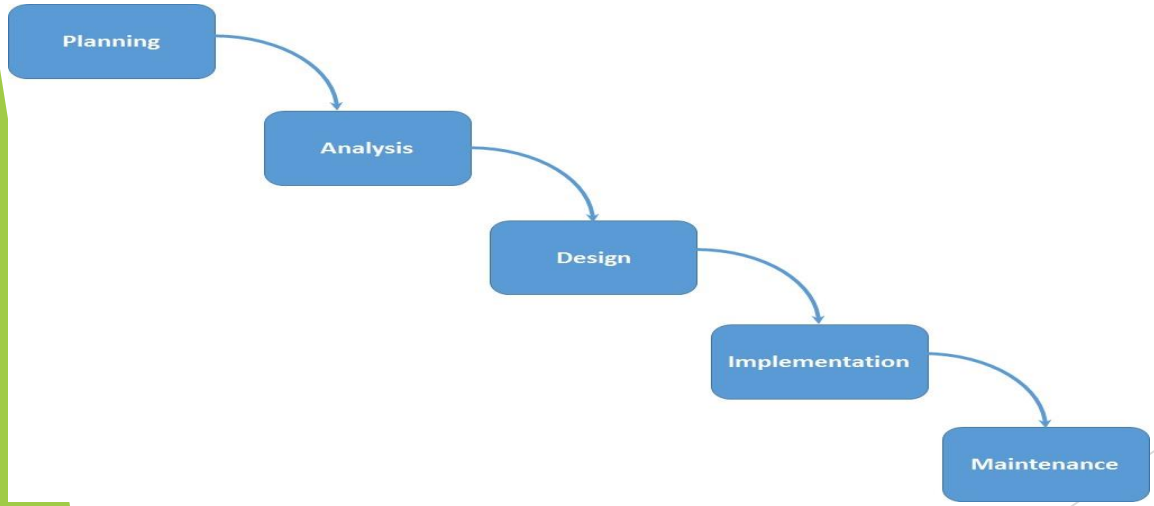
- Figures are from [Reference-2 mentioned in reference section]



(a) Linear process flow

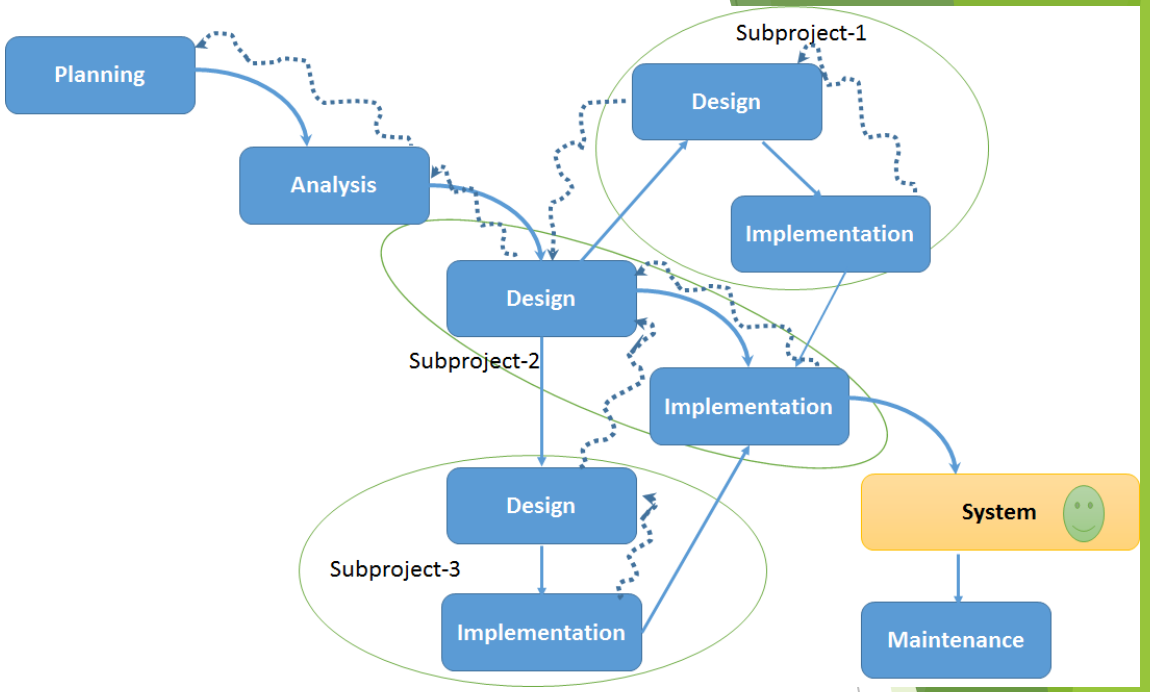


(b) Iterative process flow

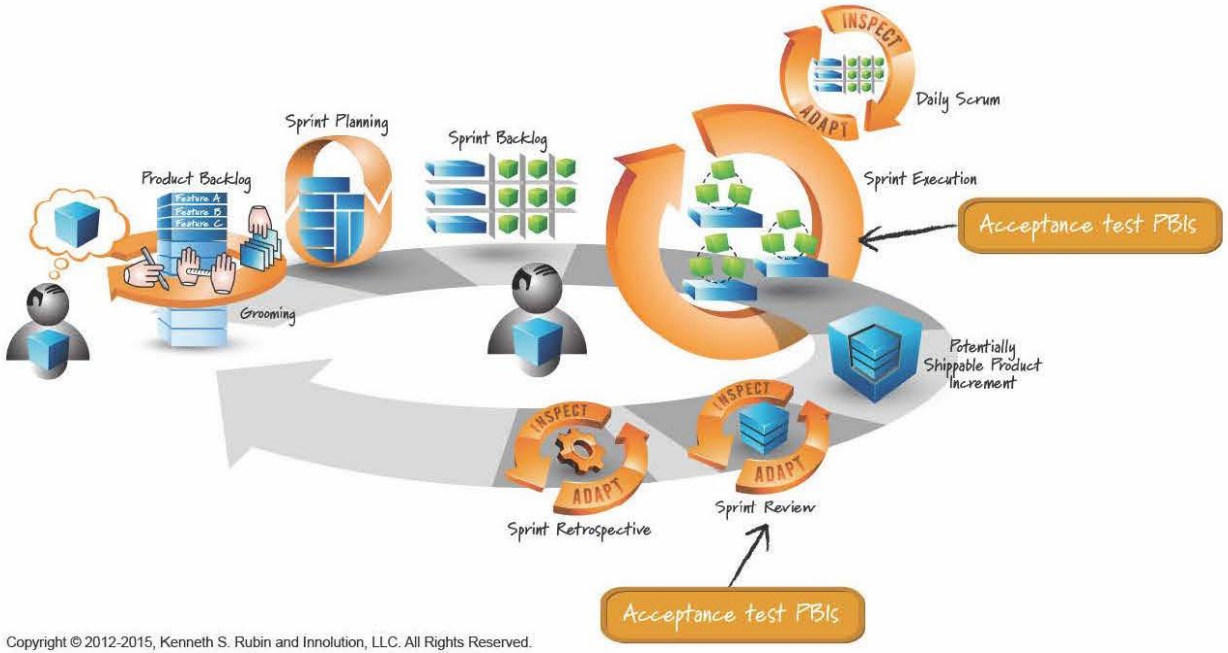


- Waterfall process

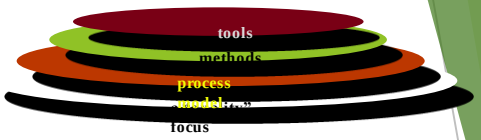
The WebE Layers parallel model



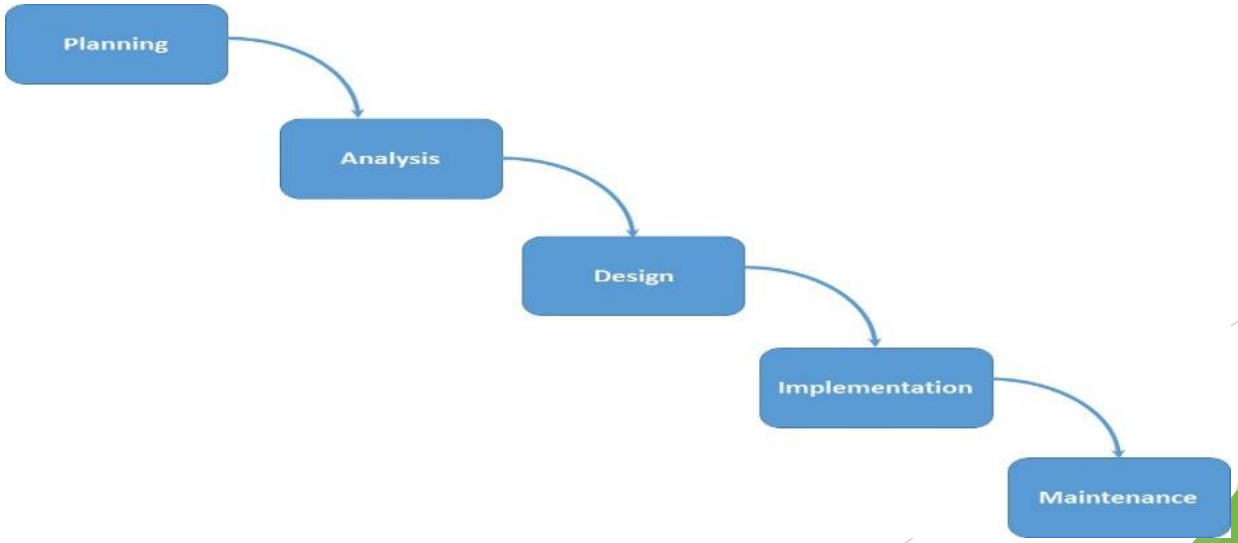
Scrum process model



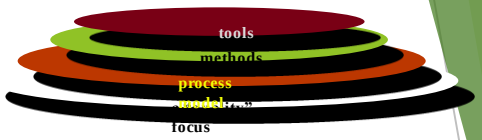
The WebE Layers



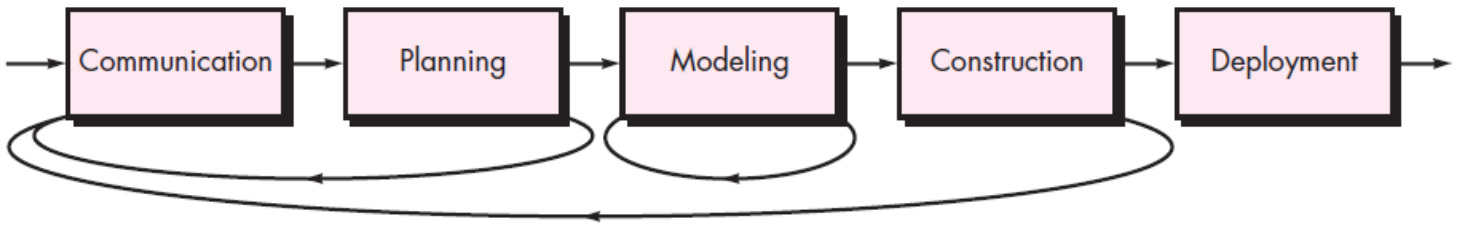
- Waterfall process
  - The waterfall approach is easy to understand and easy to use.
  - However, it considers that all of the requirements can be identified at the beginning,
  - which has many drawbacks and makes this model suitable to small or simple projects with requirements that can be determined at the beginning.
  - In practice, in big or complex projects requirements can change and some remain unknown until the implementation phase, when users trial the system.
  - With the waterfall approach there is not much opportunity for identifying issues before the implementation phase.
  - If any issues are detected at the final stage, then much rework has to be done, this is not cost-effective or efficient.



The WebE Layers

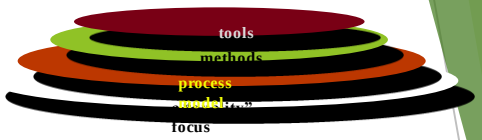


- Iterative process
  - In the iterative approach, a software is developed iteratively.
  - The first iteration goes through all or part of the activities
    - and produces an initial prototype which is a small, scaled down version of the system.
  - The second iteration starts with this first prototype and goes through all or part of the activities again, starting with the planning phase.
  - As the iteration progresses the prototype is refined until, in the final iteration, it becomes the final system.
  - A typical iteration in this approach will take some months or even years.
  - The number of iterations can vary and is determined according to the size of the project. This approach is also known as a spiral approach.

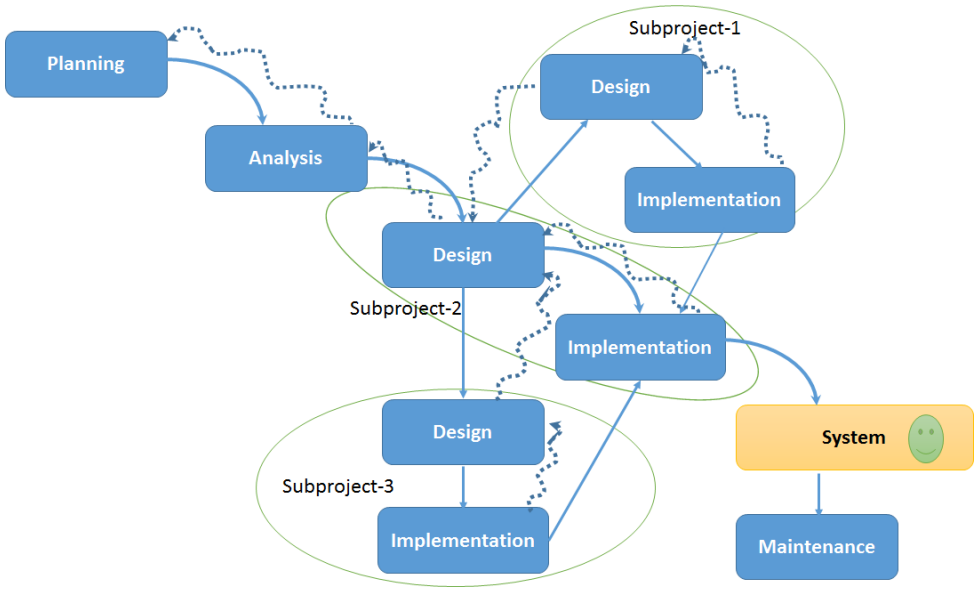


(b) Iterative process flow

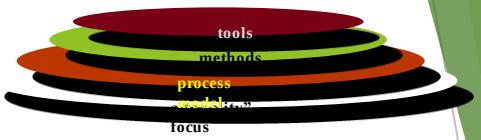
The WebE Layers



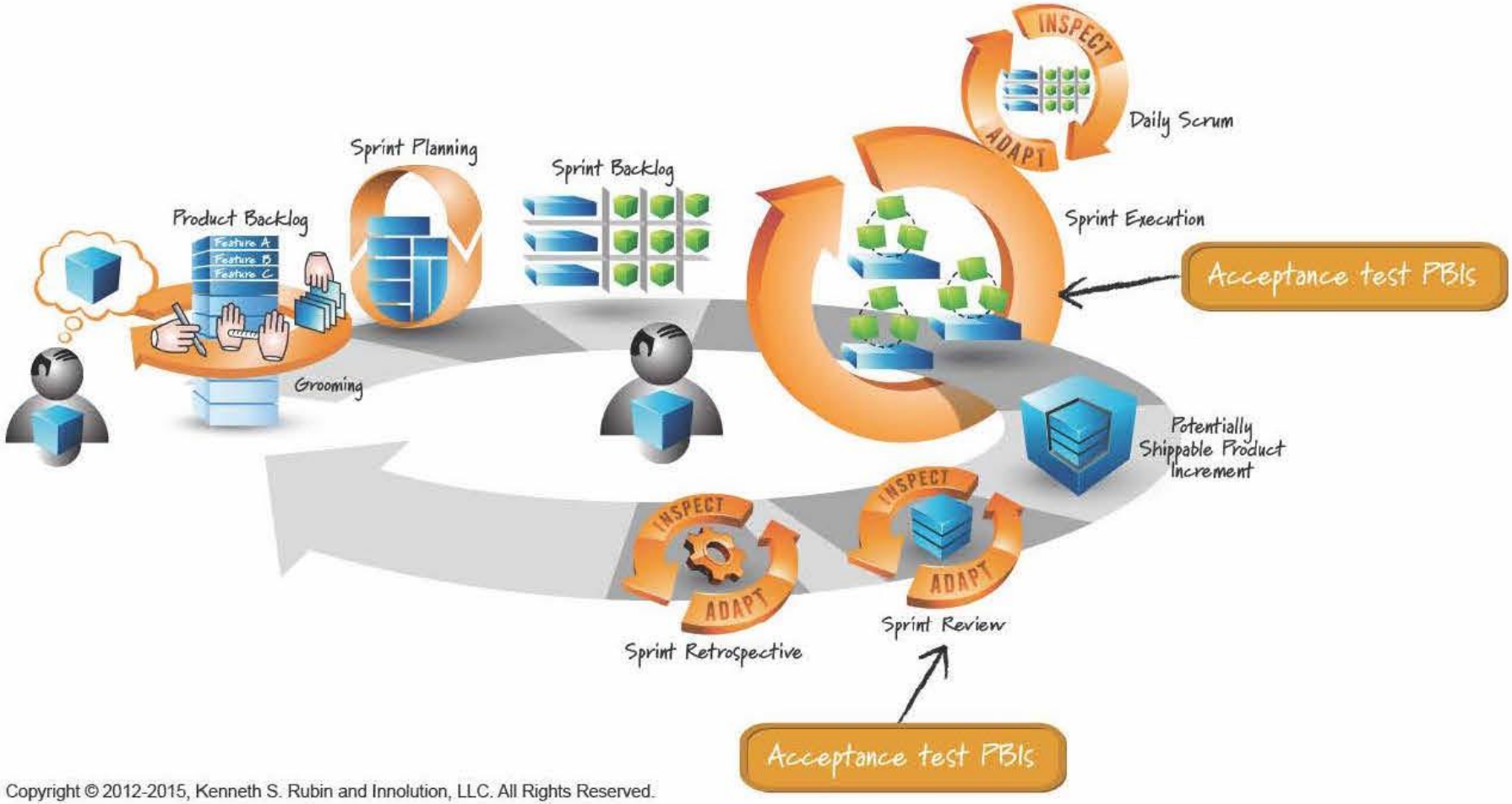
- Parallel process
  - In the parallel approach, after the analysis phase,
  - the system is divided into a number of smaller projects, or subprojects.
  - First, subprojects are designed and implemented,
  - after which all subprojects are integrated into a complete system.
  - This approach may create a problem of integration at the final stage.
  - Figure below shows the execution flow of the different activities in the parallel approach.



The WebE Layers



- Scrum process
- Will go next class



Readings

1. R. S. Pressman and D. Lowe: *Web Engineering, A Practitioner's Approach,* McGraw-Hill, 2009*.*

- **Chapter 1: Web-Based Systems**
- **Chapter 2: Web Engineering**

2. Roger S. Pressman : Software Engineering: A Practitioners Approach. (8th Edition, 2014)

Papers and other reading materials in "Week 1 Readings" folder on CloudDeakin.