Software

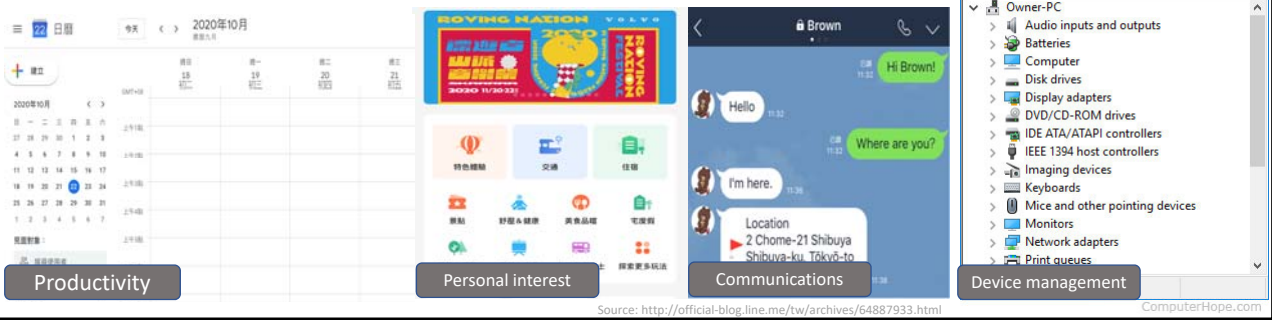
Shih-Yi (James) Chien
Assistant Professor
Dept. of Management Information Systems
National Chengchi University, Taiwan
sychien@nccu.edu.tw

1

Purpose of Apps

Application software: programs that can help you perform specific tasks when using a computer or device

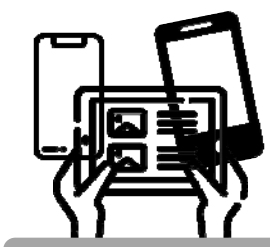
- Productivity apps: create documents for commercial or personal use
- Graphics and media apps: interact with digital media
- Personal interest apps: tools to pursue interests
- Communications apps: tools for sharing information
- Device management apps: tools to maintain a computer or device



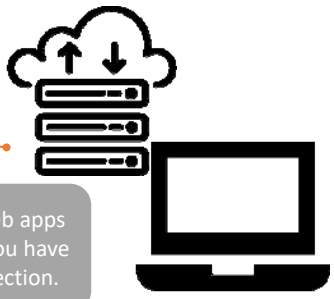
Types of Apps

Types of Apps

- **Local apps**: installed and run on computer hard drive
- **Portable apps**: run from external storage (e.g., flash drive) or cloud
- **Web apps**: programs accessed through Internet via browser or mobile device
- **Mobile apps**: apps run on the mobile device (e.g., smartphone or tablet)



Mobile apps run on your smartphone or tablet.



You can access web apps anywhere when you have an Internet connection.

3

You are about to develop a new program for various platforms within a limited time, which one should go first? Why?

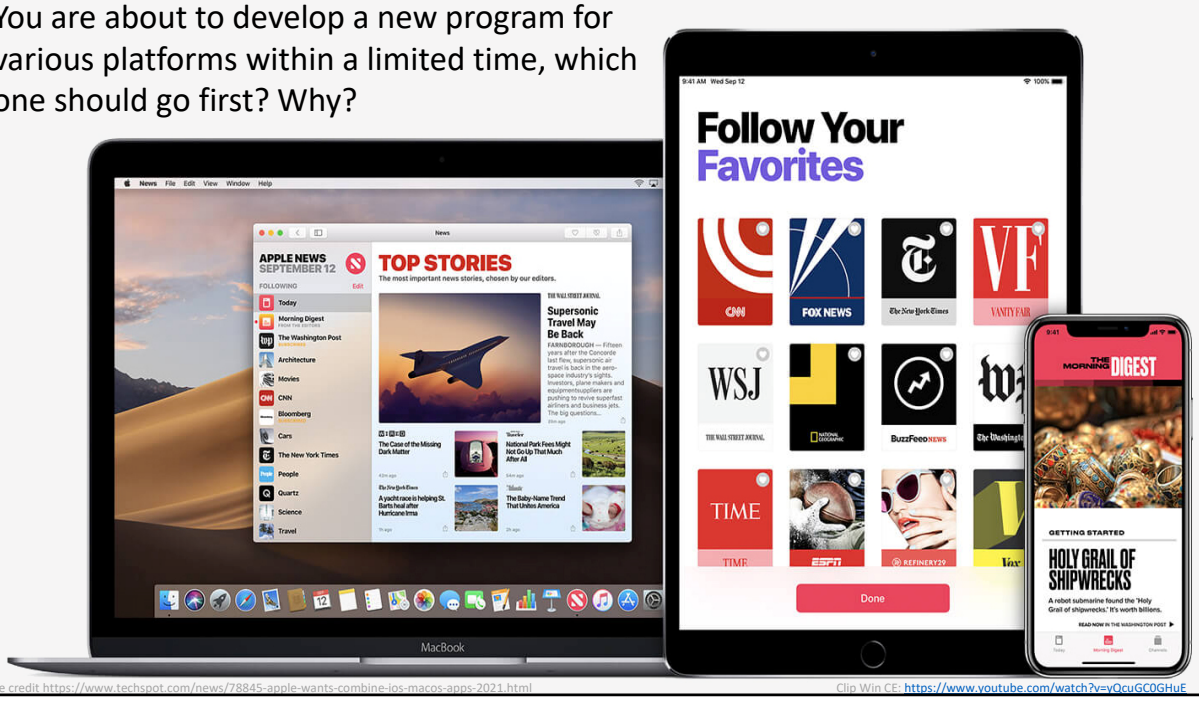


image credit https://www.techspot.com/news/78845-apple-wants-combine-ios-macos-apps-2021.html

Clip Win CE: https://www.youtube.com/watch?v=yQcuGC0GHuE

Current Trends in App Development

Mobile first design

- Build apps to work on mobile devices first as they come with more restrictions
  - Limited resources to deliver system services
  - Smaller screen or less computing power to enhance system functionalities
- Focus on the key elements
  - Understand the users' most important tasks
  - Quickly access key functions on a small screen
- Extend the functions of the tablet or desktop version
  - Deleting menu items vs. expanding or adding menu items
- Cross platform mobile development tools
  - Mobile commerce (m-commerce)

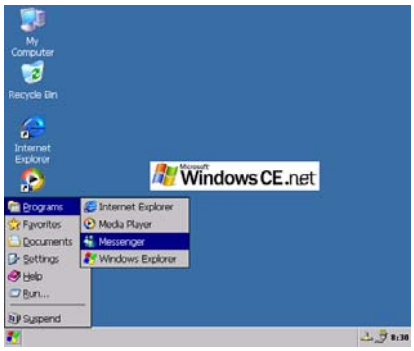
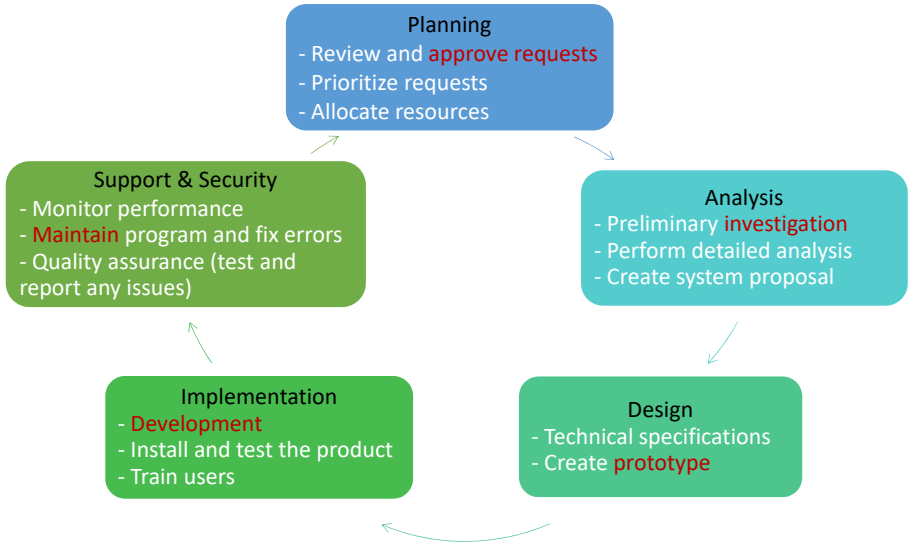


image credit: https://guidebookgallery.org/guis/windowsce/screenshots
Clip Win CE: https://www.youtube.com/watch?v=yQcuGC0GHuE

Software Development Life Cycle (SLDC)

SLDC produces the fastest, least expensive, and highest quality products



6

Analysis Phase

- Determine if the project is worth pursuing
  - Operational feasibility: the operating status of the application; whether it meets user requirements
  - Schedule feasibility: project's timetable and deadline
  - Technical feasibility: the skills and resources required to complete the planned functions
  - Economic feasibility: cost/benefit analysis
- Specify what the software will do without determining how it will be done

7

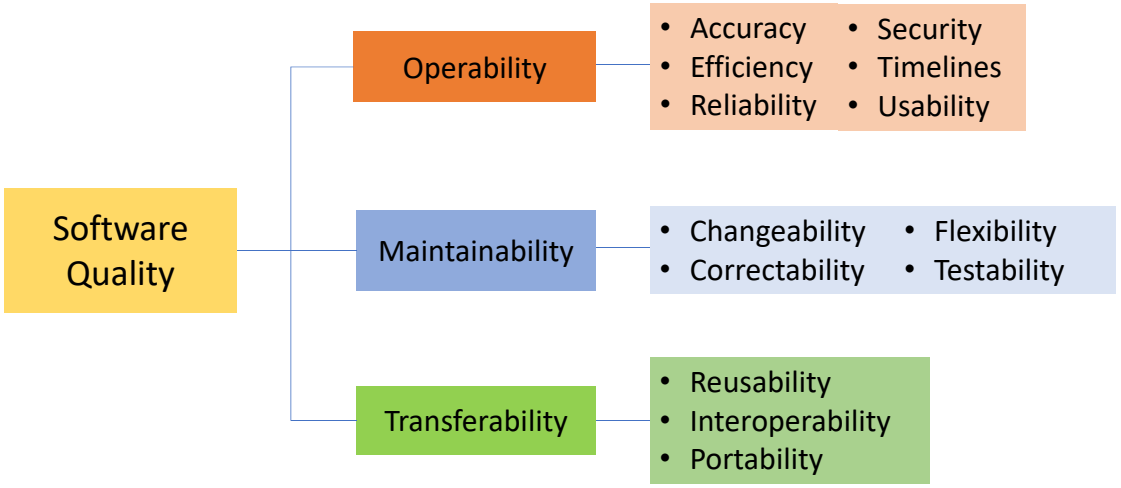
Design Phase

- Determine how the system will accomplish what was defined in the analysis phase
- Obtain necessary hardware and develop the details of the finished product (prototype)
  - All system components need to be defined

8

Implementation Phase

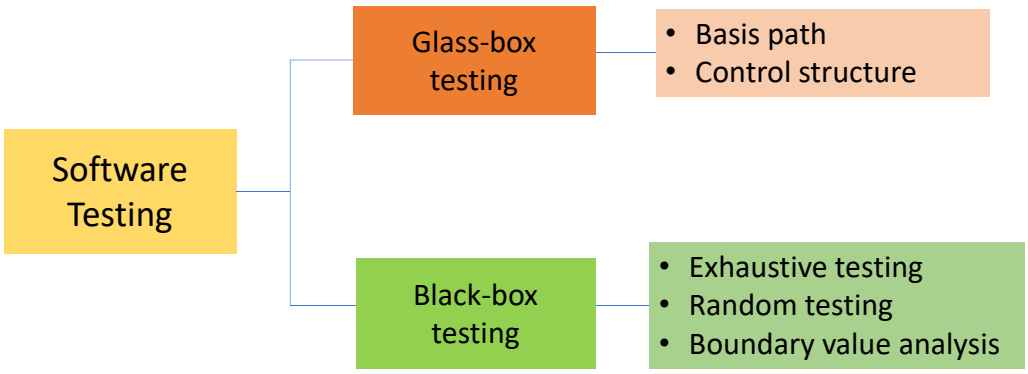
Develop and test the product, and convert it to the new system



9

Testing Phase

Develop and test the product, and convert it to the new system



10

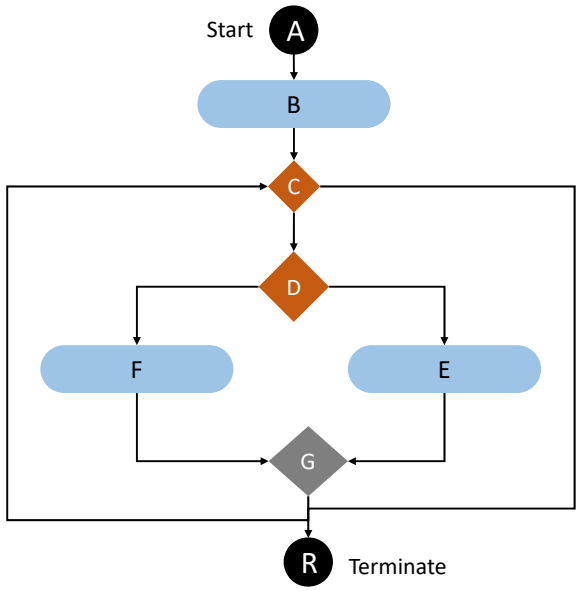
Glass-box testing (white-box testing)

- Goal: determine whether all components of the software work according to their design purpose
- Assumes that the tester knows everything about the software
  - Software is like a glass box in which everything inside the box is visible
  - Testing is done by the software engineer
- All independent paths in each module are tested at least once
- All the decision constructs (two-way and multiway) are tested on each branch
- Every loop construct is tested
- All data structures are tested

11

Basis Path Testing

- Create a set of test cases that each statement in the software is executed at least once



Independent paths:
Path-1: (A, B, C, R)
Path-2: (A, B, C, D, E, G, R)
Path1: (A, B, C, D, F, G, R)

12

Control Structure Testing

- More comprehensive than basis path testing
- Use different types of tests
- Condition testing
  - Apply any condition expression in the module
  - Simple condition vs. Compound condition
- Loop testing
  - All types of loops (while, do, for)

13

Black Box Testing

- Test without knowing the internal functions of the software and how the software works
- Test the functionality of the software
  - Functions that the software should complete (e.g., input and output)

14

Exhaustive Testing

- Best black-box test method
- Test all possible input values of the software
- Difficult to apply in complex software (input domain is too huge)

15

Random Testing

- Test a selected subset of values in the input domain
- A subset of the selected values are evenly distributed in the input domain
- In this case, using a random number generator can be helpful

16

Boundary Value Testing

- Inputs must be greater than or equal to the boundary value
  - Greater than, equal, lower than
  - $x \geq 100000$, $x > 100000$, etc.

17

Predictive Development

Waterfall model takes each step separately and completes it before continuing to the next stage

- Pro: each stage knows exactly what to do because they have the complete results of the previous phases
- Con: difficult to locate a problem, the entire process must be checked

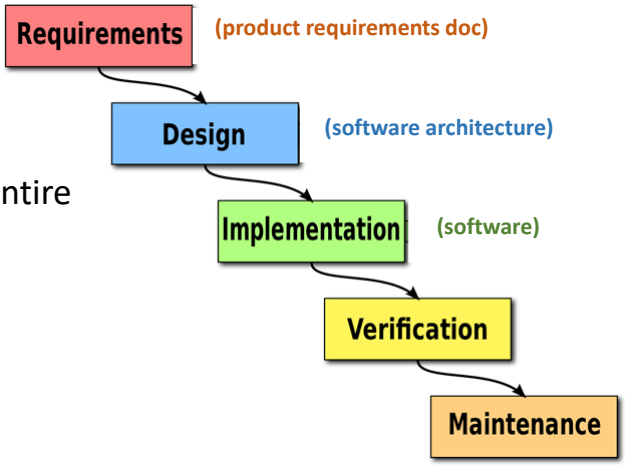


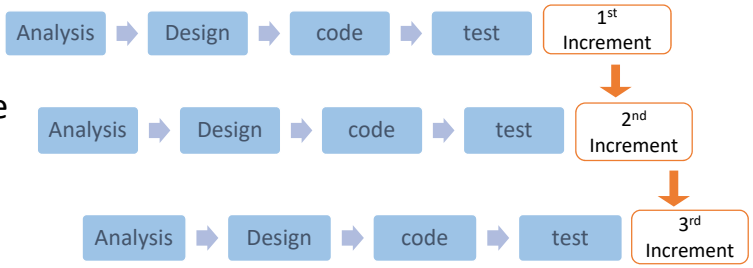
image credit: https://en.wikipedia.org/wiki/Waterfall_model#/media/File:Waterfall_model.svg

18

Incremental model

Software is developed in a series of steps

- First: a simplified version of the system
  - Represent the entire system by little details
- Second: Include more details
- Pro: easy to find problems
- Con: problems might cause due to system architecture fails to accommodate all requirements for the entire software lifecycle (need good planning and designing)



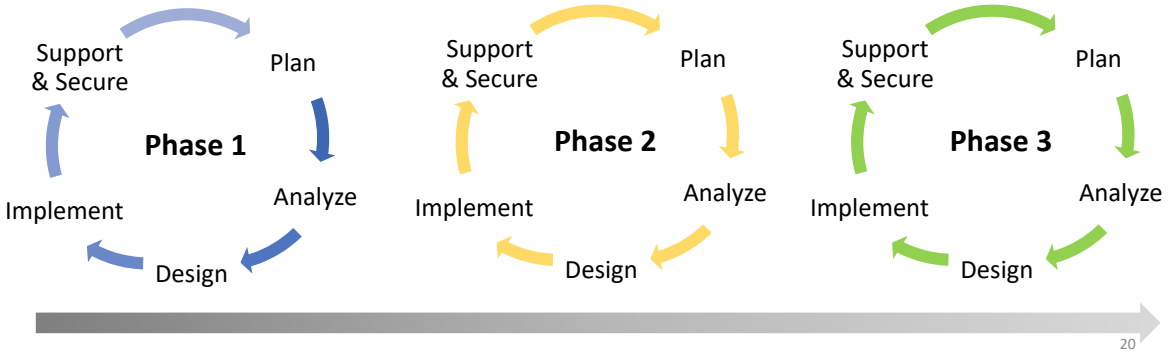
19

Agile Development (Adaptive Development)

Incorporate the iterative and incremental software development

Increase flexibility to the project goals and scope

- Evolving in phases
- Adding components according to user needs or requirements
- Incorporate testing and feedback from users at all processes and phases
  - Change rapidly to adapt to the market



20

科目代號(Course #)：306005001
科目名稱：計算機概論
Course Name：Introduction to Computer Science
授課教師：簡士鎰
Instructor：CHIEN SHIH-YI
系所：資管一甲、資管一乙
上課時間 (Session)：五23 (fri09-11)

科目代號(Course #)：306005011
科目名稱：計算機概論
Course Name：Introduction to Computer Science
授課教師：簡士鎰
Instructor：CHIEN SHIH-YI
系所：資管一甲、資管一乙
上課時間 (Session)：五D5 (fri13-15)