Lecture01: Introduction

EGCI340: SOFTWARE

Meet UX Designers at Google

- https://youtu.be/116sMd5U7UY

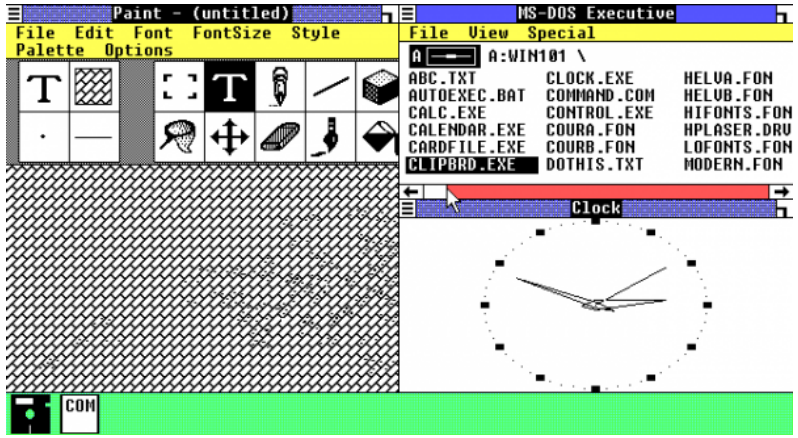
Meet UX Engineers at Google

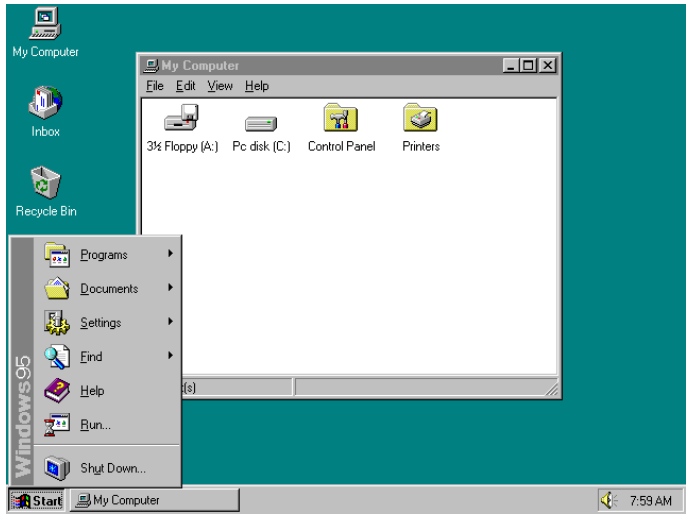
- https://youtu.be/D0ga7_HEfXs

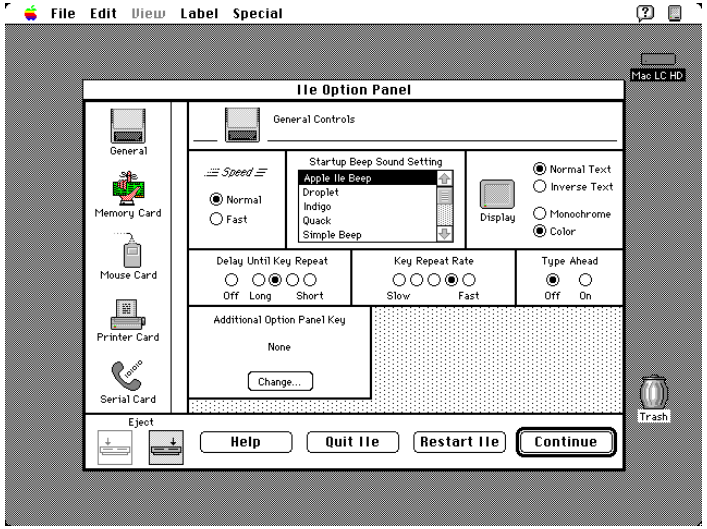
Meet a Software Engineer on the Google Hardware Team

- https://youtu.be/dyGyBTkkQHA

What are these software?








Start
Friday
24 October 2008
11:09 AM
Getting Started
Tap here to set owner information
No unread messages
No tasks
No upcoming appointments
Device unlocked
Calendar
Contacts


AT&T
6:09 PM
Messages
Calendar
Photos
Camera
Videos
YouTube
Stocks
Maps
Weather
Notes
Reminders
Game Center
Newsstand
iTunes
App Store
Settings
Phone
Mail
Safari
Music


9:02 AM
Phone
Readers Hub
Navigation
Camera
Places
Music
Video
Calendar
Thinkfree
Market
Browser
Applications
Email


12:05
T-Mobile
Messaging
Outlook
NETFLIX
Slacker Radio
XBOX LIVE

What is Software Design?

- Software design is a process to transform user requirements into some suitable form
  - Help the programmer in software coding and implementation.

- Software design is the first step in SDLC (Software Design Life Cycle),
  - Move the concentration from problem domain to solution domain.
  - It tries to specify how to fulfill the requirements mentioned in SRS.

Objectives of Software Design

- **Correctness**: Software design should be correct as per requirement.
- **Completeness**: The design should have all components like data structures, modules, and external interfaces, etc.
- **Efficiency**: Resources should be used efficiently by the program.
- **Flexibility**: Able to modify on changing needs.
- **Consistency**: There should not be any inconsistency in the design.
- **Maintainability**: The design should be so simple so that it can be easily maintainable by other designers.

Software Design Levels

Software design yields three levels of results:

- **Architectural Design** - It identifies the software as a system with many components interacting with each other.
- **High-level Design** - The high-level design breaks the single entity-multiple component concept of architectural design into less-abstracted view of sub-systems and modules and depicts their interaction with each other.
- **Detailed Design** - Detailed design deals with the implementation part of what is seen as a system and its sub-systems in the previous two designs.

Modularization

- Modularization is a technique to divide a software system into multiple discrete and independent modules, which are expected to be capable of carrying out task(s) independently.

- These modules may work as basic constructs for the entire software. Designers tend to design modules such that they can be executed and/or compiled separately and independently.

Coupling and Cohesion

- When a software program is modularized, its tasks are divided into several modules based on some characteristics.
- As we know, modules are set of instructions put together in order to achieve some tasks.
- There are measures by which the **quality of a design** of modules and their interaction among them can be measured.
- These measures are called coupling and cohesion.

What is systems development life cycle (1)

- The systems development life cycle (SDLC) is the process of understanding how an information system (IS) can support business needs, designing the system, building it, and delivering it to users.

- The key person in the SDLC is the Systems Analyst (SA) who analyses the business situation, identifies opportunities for improvements, and designs an information system to implement them.

What is systems development life cycle (2)

- The primary objective of the systems analyst is not to create a wonderful system.

- The primary goal is to create value for the organization, which for most companies means increasing profits (government and non-profit organizations measure value differently).

What is systems development life cycle (3)

* From Training Specialists Website.

What is system development methodologies?

- A methodology is a formalized approach to implementing the SDLC (i.e., it is a list of steps and deliverables).
- Category of systems development methodologies:
  - Waterfall
  - Incremental Development
  - Evolutionary Development
  - Agile Development
  - Etc.

What is system development methodologies?

- A methodology is a formalized approach to implementing the SDLC (i.e., it is a list of steps and deliverables).
- Category of systems development methodologies:
  - Waterfall
  - Incremental Development
  - Evolutionary Development
  - Agile Development
  - Etc.

Costs of Software Engineering

60% of costs are development costs (roughly)

40% are testing costs

Costs vary depending on:

- Type of developed system
- Requirements of system attributes
  - Such as performance and system reliability

Distribution of costs depends on the development model

Computer-Aided Software Engineering (CASE)

Software systems that are intended to provide automated support for software process activities.

Types of CASE Tools

- Upper-CASE Tool
  - Tools to support the early process activities of requirements and design
- Lower-CASE Tool
  - Tools to support later activities such as programming, debugging and testing

Attributes of Good Software

The software should deliver the required functionality and performance to the user and should be maintainable, dependable and acceptable

1. Maintainability
   - Software must evolve to meet changing needs
2. Dependability
   - Software must be trustworthy
3. Efficiency
   - Software should not make wasteful use of system resources
4. Acceptability
   - Software must accepted by the users for which it was designed.

Professional and ethical responsibility

Software engineering involves wider responsibilities than simply the application of technical skills

Software engineers must behave in an honest and ethically responsible way if they are to be respected as professionals

Ethical behavior is more than simply upholding the law

Issues of professional responsibility

Confidentiality

- Engineers should normally respect the confidentiality of their employers or clients irrespective of whether or not a formal confidentiality agreement has been signed

Competence

- Engineers should not misrepresent their level of competence. They should not knowingly accept work which is out with their competence

Issues of professional responsibility (cont.)

Intellectual property rights

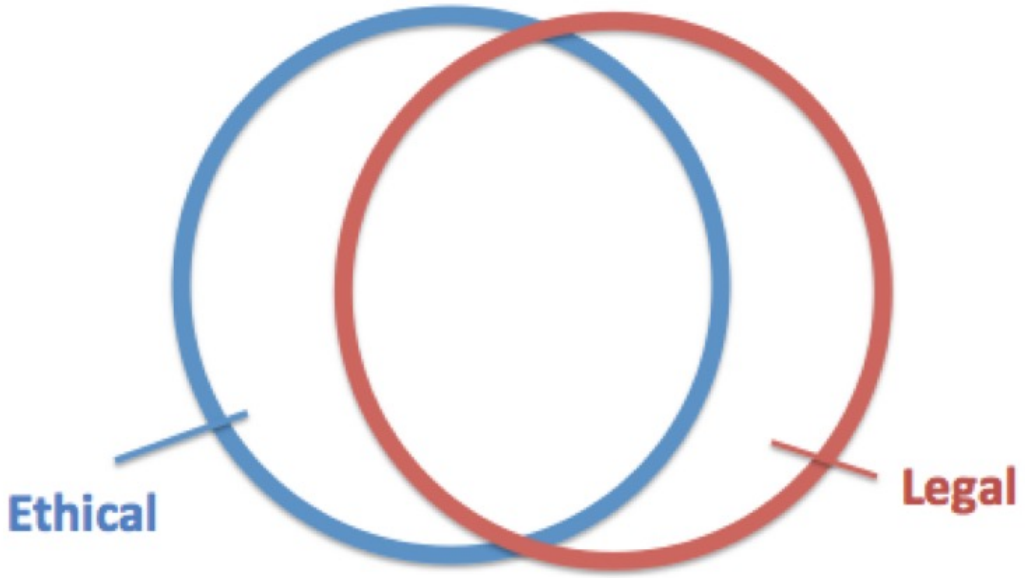
- Engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc.
- They should be careful to ensure that the intellectual property of employers and clients is protected

Computer misuse

- Software engineers should not use their technical skills to misuse other people's computers
- Computer misuse ranges from relatively trivial (game playing on an employer's machine, say) to extremely serious (dissemination of viruses)

Ethical ≠ Legal

In very basic terms, ethics are about right and wrong, while laws involve civil or criminal penalties, such as lawsuits, fines, or imprisonment. There is a lot of overlap between what's wrong (unethical) and what can subject you to civil or criminal penalties (illegal), but that doesn't meant that the two terms are equivalent.

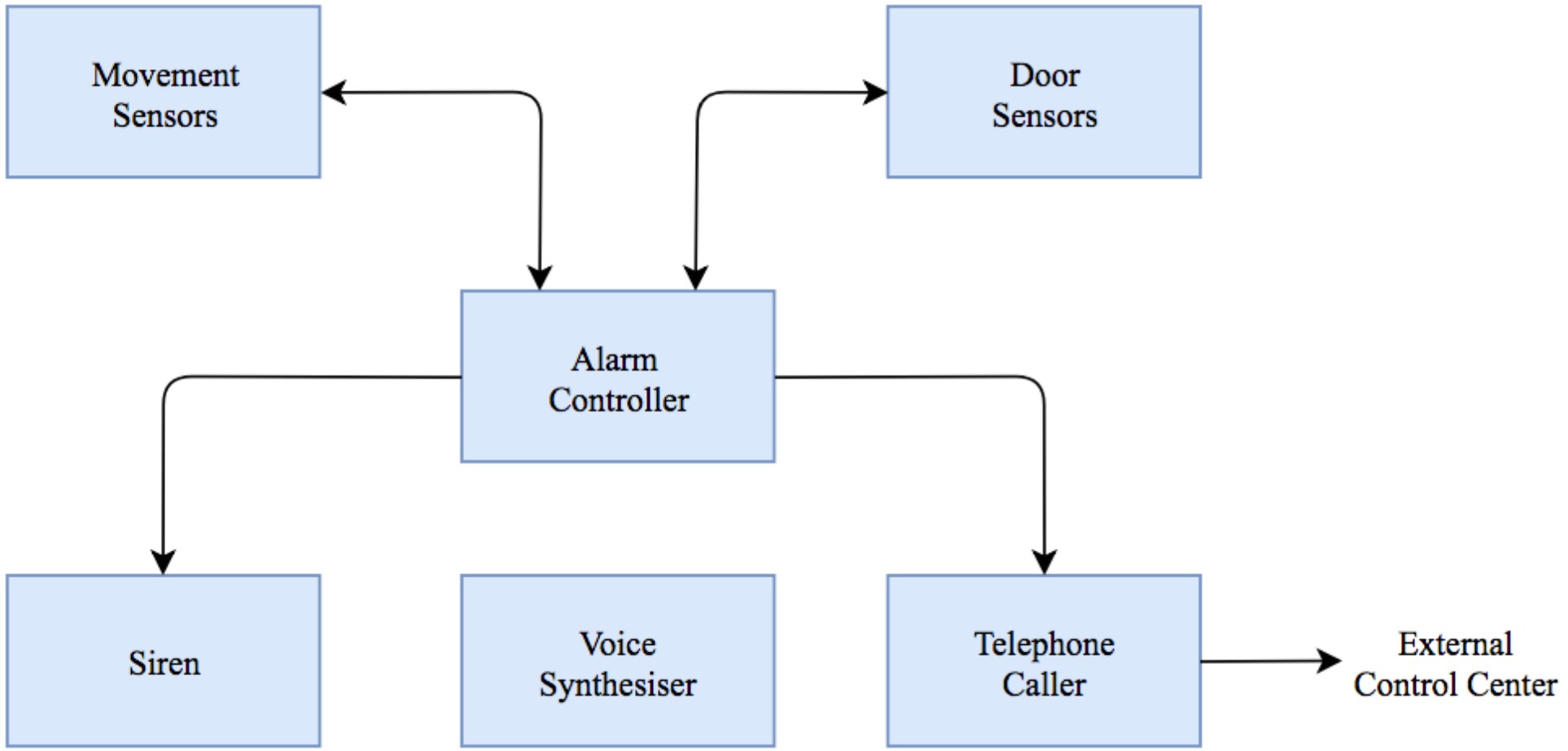


System Architectural Model

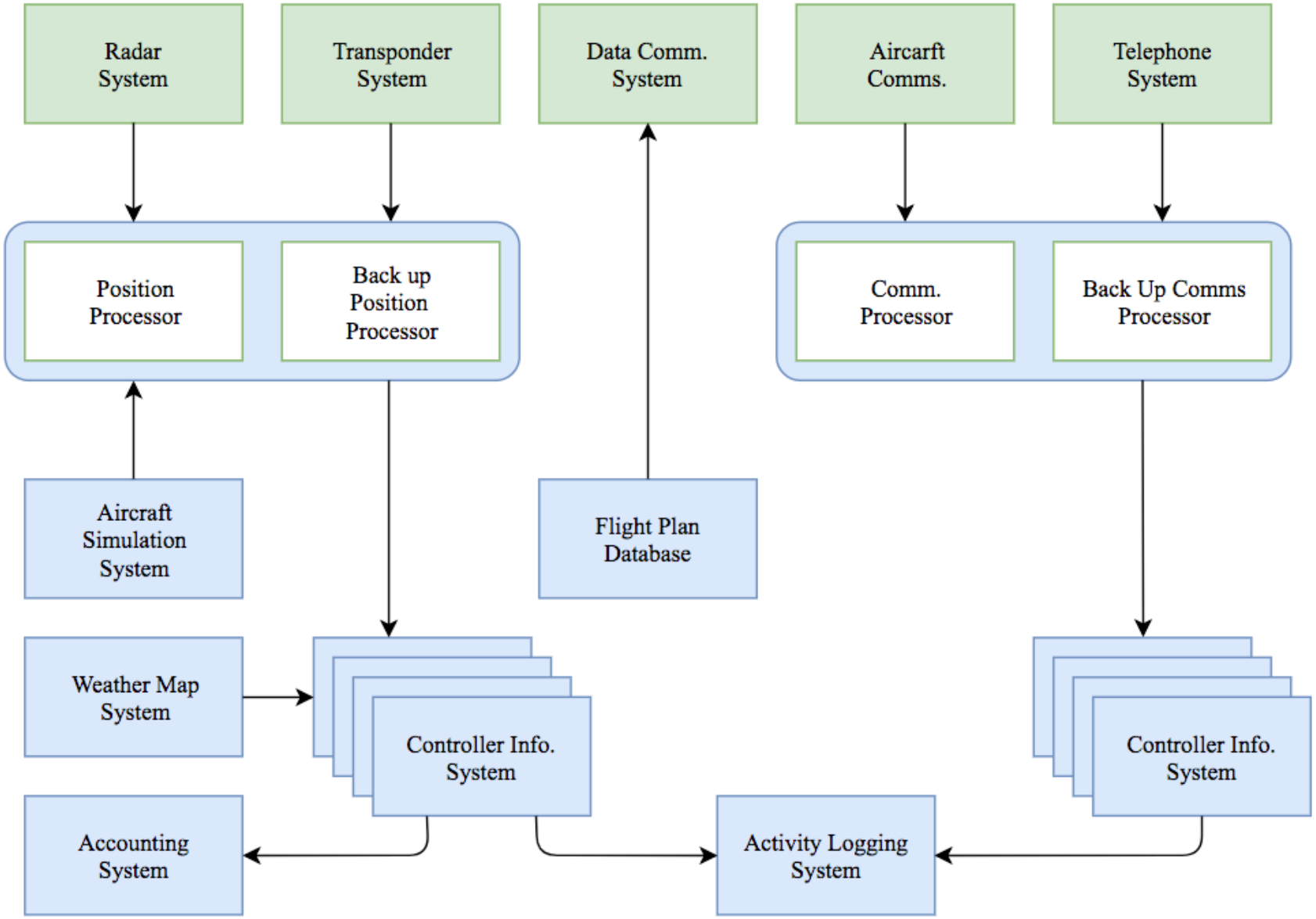
An abstract view of the subsystems making up a system

- Include major information and flow between sub-systems
- Presented as a block diagram
- Identify different types of functional components in the model

System Architectural Model: **Burglar Alarm System [1]**



System Architectural Model: **Airplane Travel Control [1]**



Legacy Systems

Socio-technical systems that have been developed using old or obsolete (not modern) technology

Crucial to the operation of a business

- Too risky to discard these systems
  - Such as Bank customer accounting system and Aircraft maintenance system

References

1. Ian Sommerville, Software Engineering 10th Edition, Pearson, April 2015

Any Questions?

:O)

Thank you