HOUSE KEEPING

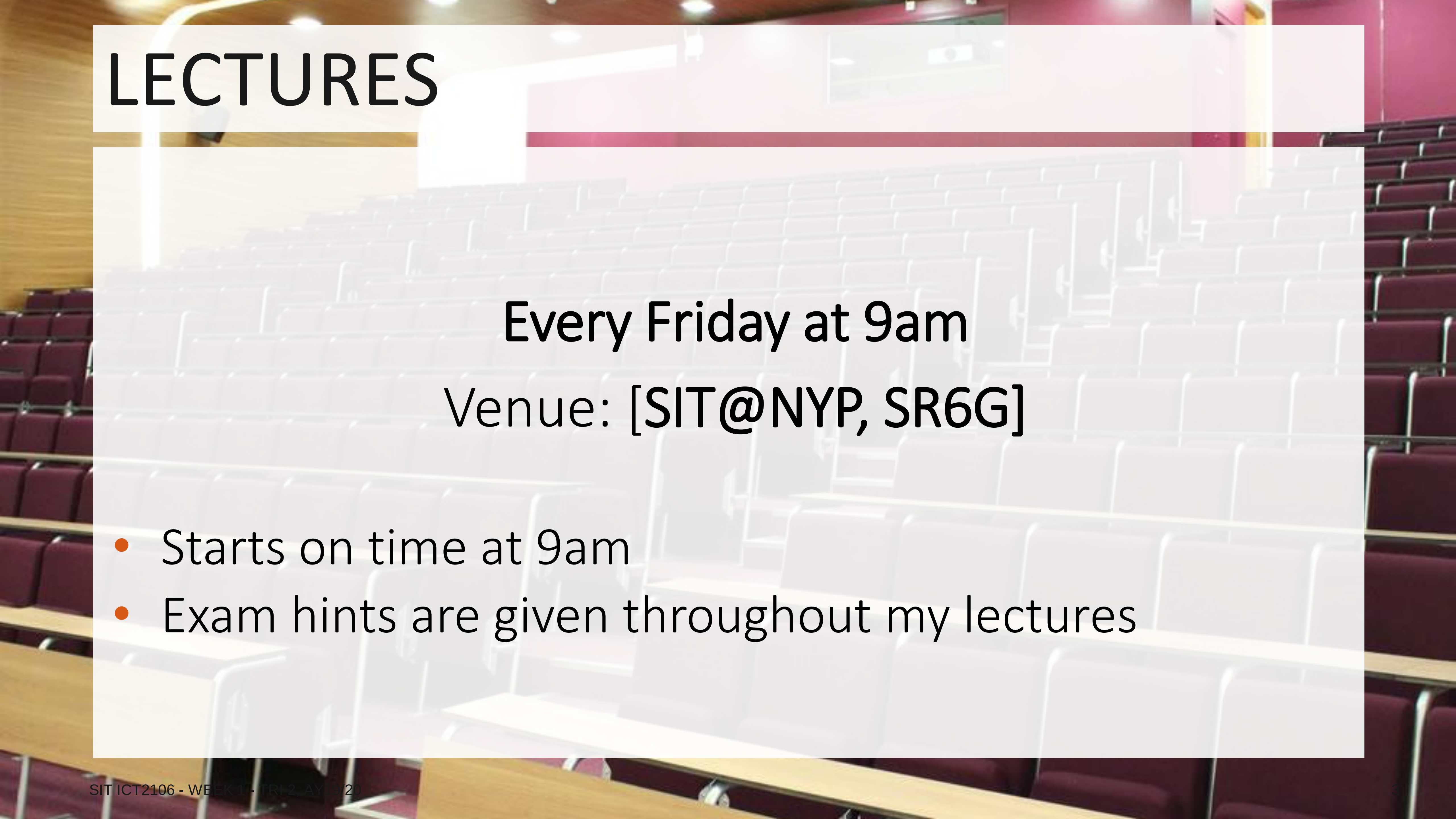
ICT2106 Software Design

Dr. Alex Q. Chen

Assistant Professor

Alex.Q.Chen@singaporetech.edu.sg

DID: 6592 1401



LECTURES

Every Friday at 9am

Venue: [SIT@NYP, SR6G]

- Starts on time at 9am
- Exam hints are given throughout my lectures

…HOW TO LEARN BEST IN CLASS

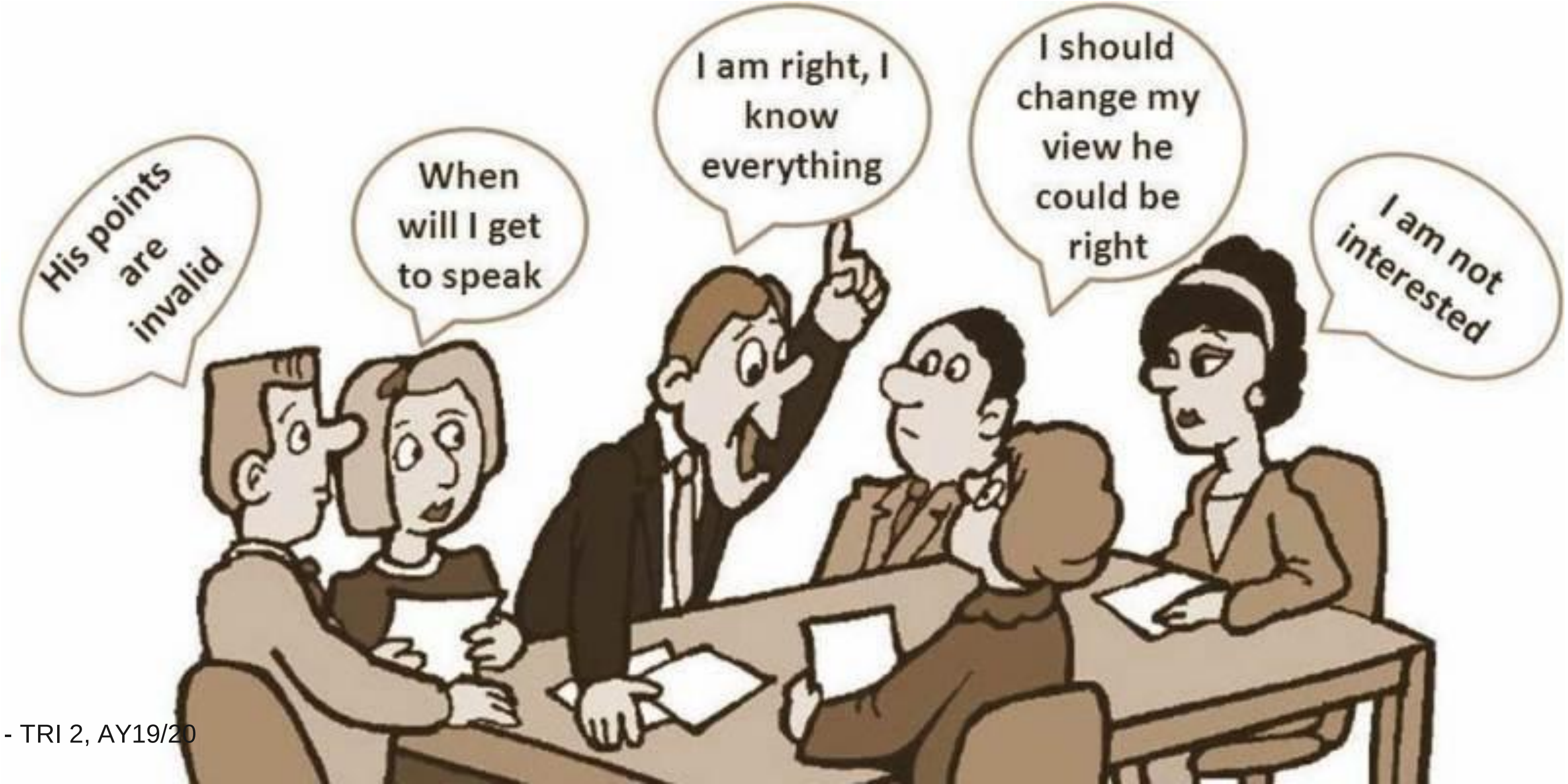
Are you listening?

…HOW TO LEARN BEST IN CLASS

Asking questions

…HOW TO LEARN BEST IN CLASS

Engage in class activities


His points are invalid
When will I get to speak
I am right, I know everything
I should change my view he could be right
I am not interested

WHAT IS EXPECTED OF YOU?

- Be on time for lectures and labs
- Do self-readings to supplement your understanding of the module
- Be attentive in lectures
- Silent mobile phones during classes
- When you are not clear about some topics
  - Read up the lecture and additional reading materials
  - Discuss the materials with me

    Alex.Q.Chen@singaporetech.edu.sg
- Always try to attempt all questions first

SUPPLEMENTARY READINGS

1. Carlos E. Otero. *Software Engineering Design*. CRC Press, Boca Raton FL, 2012 – Chapter 1
2. Software Engineering Book of Knowledge – SWEBOK: http://www4.ncsu.edu/~tjmenzie/cs510/pdf/SWEBOKv3.pdf – Software Design Chapter
3. UML – Information from the OMG portal http://www.uml.org and http://www.omg.org/spec/UML/

TEACHING PLAN – PART 1 – ICT2106

Week	Morning Session	Afternoon Session	Due
1 10 Jan	Lecture 1	Form your project team and Team's module selection	
2 17 Jan	Lecture 2	• Team's module confirmation • Lab 1	• Project team members • Team's module
3 24 Jan	Lecture 3	Lab 2 (CNY eve)	• Lab assignment 2 at 6pm • Project Proposal (Tues, 28 Jan at 1pm)
4 31 Jan	Lecture 4	• Lab 3 • Project Proposal feedback	• Lab assignment 3, Monday after your lab • Module Assignment Part 1 (Wed, 05 Feb at 6pm)
5 07 Feb	• Lecture 5 • Quiz 1	Lab 4	• Lab assignment 4, Monday after your lab
6 14 Feb	Lecture 6	Catch up week	Project Prototype 1 (TBC)

The teaching plan is subject to minor changes

PRE-REQUISITES

- Knowledge of fundamentals of Software Engineering
- UML
- Good knowledge of Web technologies and development
- Fundamental knowledge of Object-oriented Programming
- Basic understanding of Human-Computer Interaction

INTRODUCTION TO SOFTWARE DESIGN

ICT2106 – Software Design – Week 1

AGENDA

1. Motivation for Software Engineering
2. Importance of Software Design
3. Software Design Process
4. Software Design & Quality Attributes
5. Key issues in Software Design

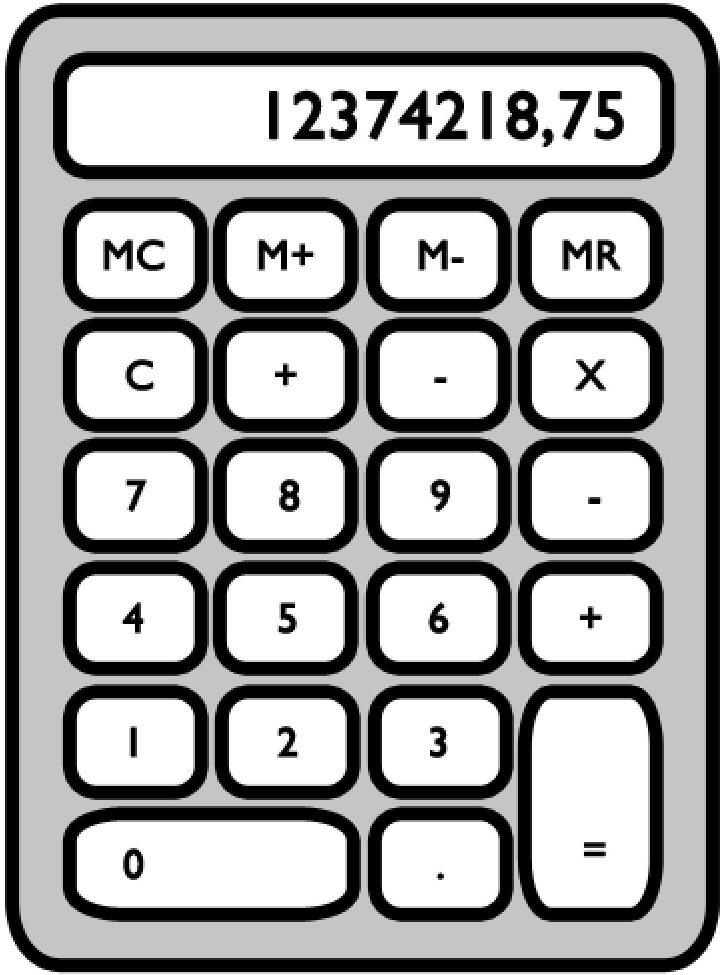
SOFTWARE ENGINEERING is …

the application of a **systematic**, **disciplined**, and **quantifiable** approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

SOFTWARE ENGINEERING

What do **systematic**, **disciplined**, and **quantifiable** meant?

MOTIVATION: THE MIGHTY CALCULATOR



Do we need software engineering to develop this product?

MOTIVATION: VEHICLES

Do we need software engineering here?

MOTIVATION: SELF-DRIVING VEHICLES

WITHOUT SOFTWARE IS IMPOSSIBLE

Notes from the video and discussions:

MOTIVATION: ARIAN 5

Notes from the video and discussions:

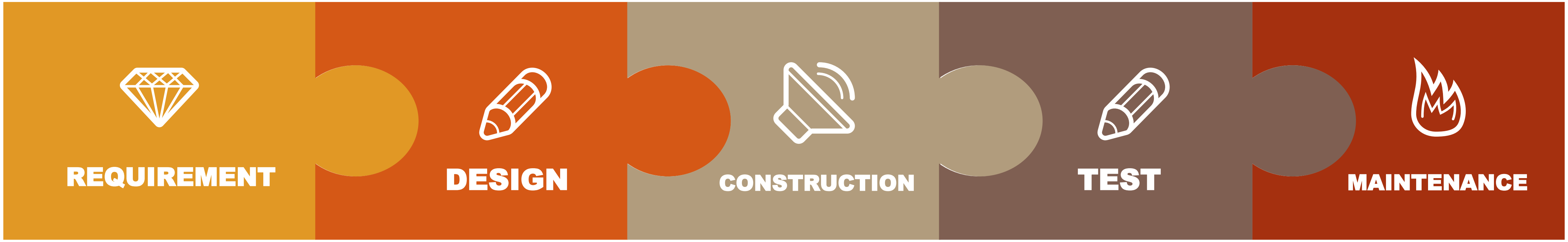
DO WE NEED SOFTWARE ENGINEERING HERE?

It's becoming evident that not all software systems are equal!

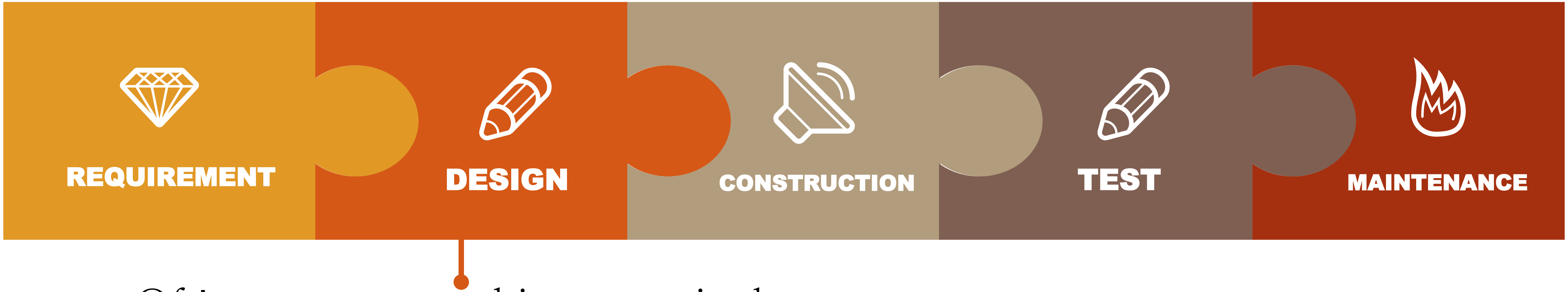
Some are more **complex** than others and software failure may result in **catastrophic events**, including loss of human life!

X-47B

ENGINEERING SOFTWARE



ENGINEERING SOFTWARE



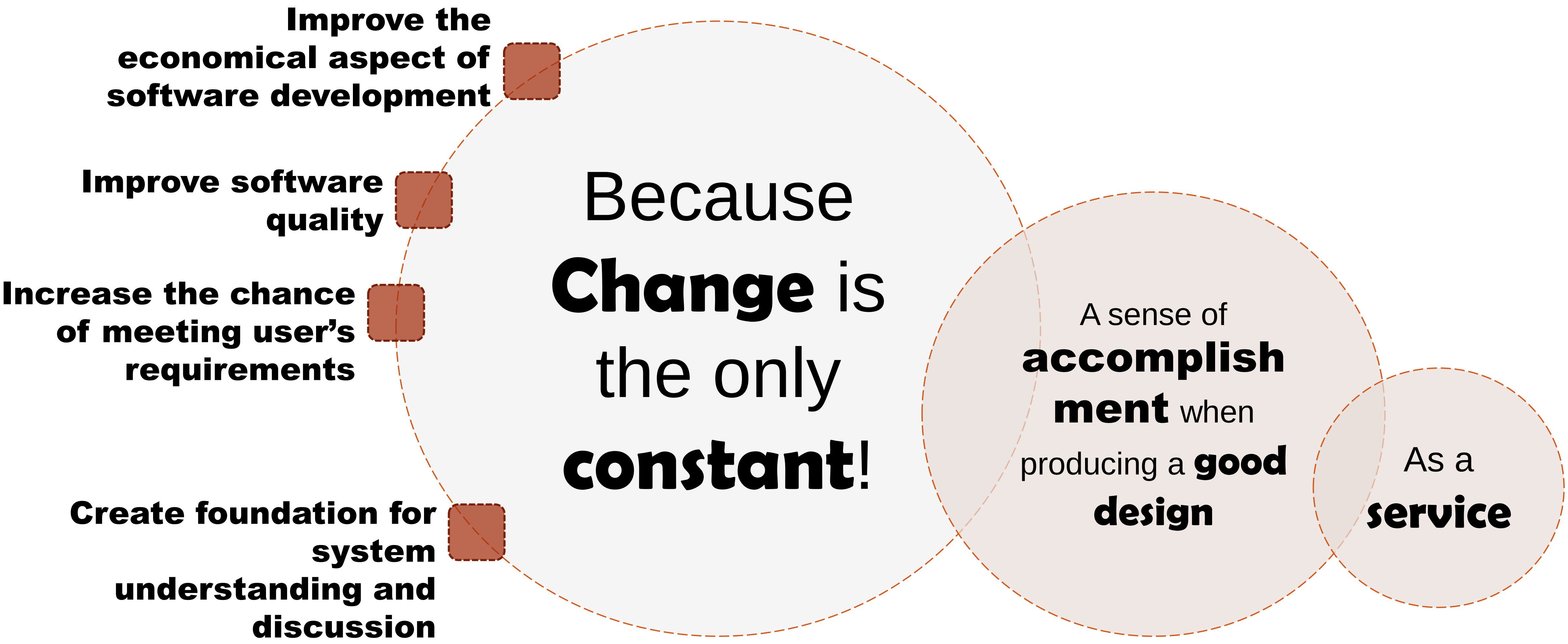
Of importance to this course is the **design *phase***, where requirements are used to create a blueprint of the software to be constructed

SOFTWARE DESIGN IS…

The **process** of identifying, evaluating, validating, and specifying the architectural, detailed, and construction models required to build software that meets its intended functional and non-functional requirements; and,

The **product** of such process

WHY SOFTWARE DESIGN?



Improve the economical aspect of software development

Improve software quality

Increase the chance of meeting user's requirements

Create foundation for system understanding and discussion

Because **Change** is the only **constant**!

A sense of **accomplishment** when producing a **good design**

As a **service**

DESIGN THINKING

Is there a wrong problem to solve?

SOFTWARE DESIGN PROCESS

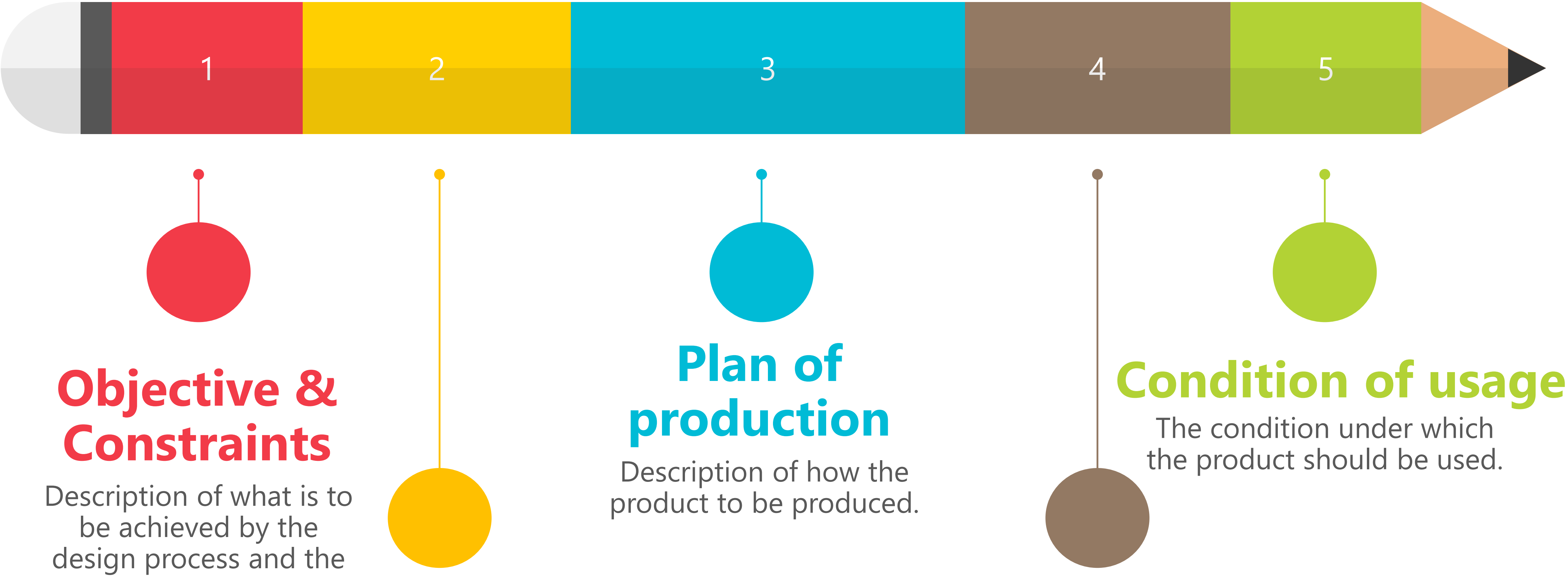
CASE STUDY 1 – MOVING HOUSE

Notes from discussions:

Suppose you are 60 years old and you have been living in 1 house for 60 years of your life. Now you are moving to a brand new house and you are out of $$$ to hire a planner or interior designer.

List the STEPS that you are going to do when moving to the new house.

DESIGN PROCESS



Objective & Constraints

Description of what is to be achieved by the design process and the contraints

Description of the product

Plan of production

Description of how the product to be produced.

Rationale of the design

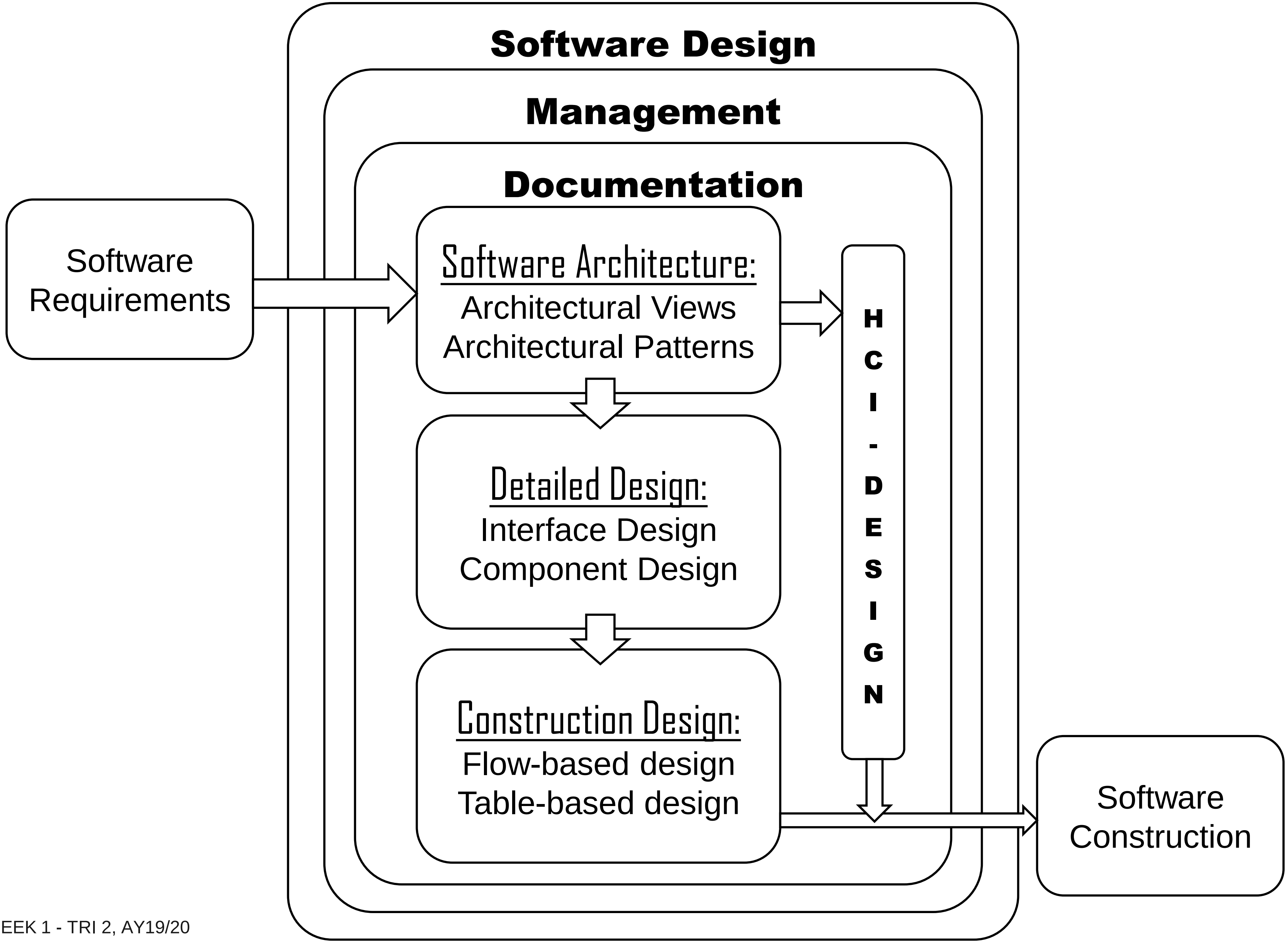
Statements that justify why the designed product and its associated production plan can achieve the objectives and satisfy the constraints

Condition of usage

The condition under which the product should be used.

SOFTWARE DESIGN PROCESS

A software design process is a set of **activities** and **controls** that specify how resources work together for the production of software design artefacts.



Software Design

Management

Documentation

Software Requirements

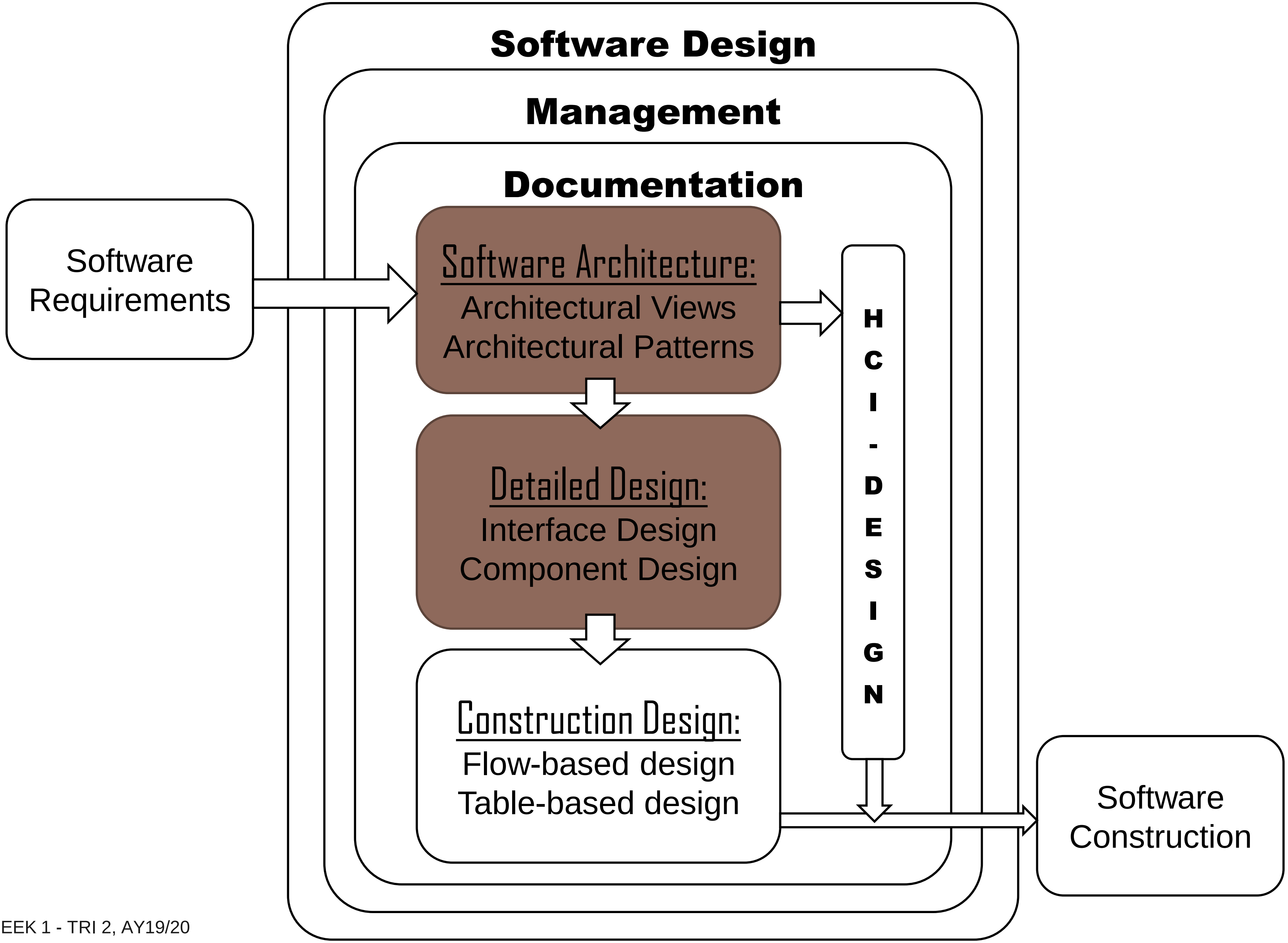
Software Architecture:
Architectural Views
Architectural Patterns

Detailed Design:
Interface Design
Component Design

Construction Design:
Flow-based design
Table-based design

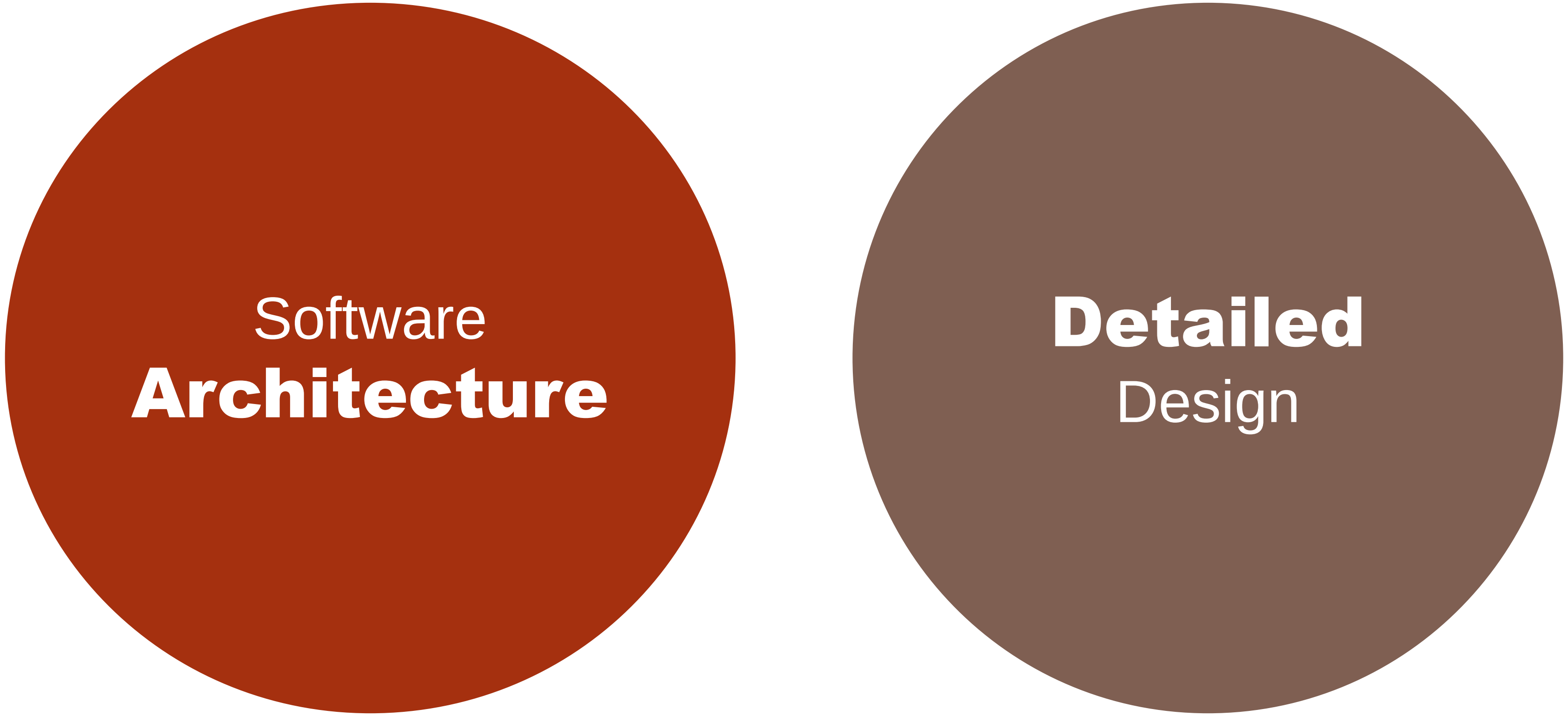
HCI-DESIGN

Software Construction


Software Requirements
Software Design
Management
Documentation
Software Architecture:
Architectural Views
Architectural Patterns
Detailed Design:
Interface Design
Component Design
Construction Design:
Flow-based design
Table-based design
HCI-DESIGN
Software Construction

SOFTWARE DESIGN PROCESS

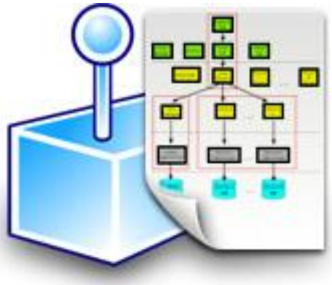
Two major activities of the software design process include:



Software **Architecture**

Detailed Design

SOFTWARE ARCHITECTURE



STRUCTURAL

It provides the major structural components and interfaces of the system.

COMMUNICATION

They serve as important communication, reasoning, and analysis tool that supports the development and growth of the system

QUALITY

It focuses on the quality aspects of the system before detailed design or construction can begin.

FOUNDATION

Lays the foundation for all subsequent work in the software engineering lifecycle

DETAILED DESIGN

Internal design

detailed design focuses mostly on the internals of those components and interfaces.

White-box

Builds on the software architecture to provide a white-box approach to design the structure and behavior of the system.

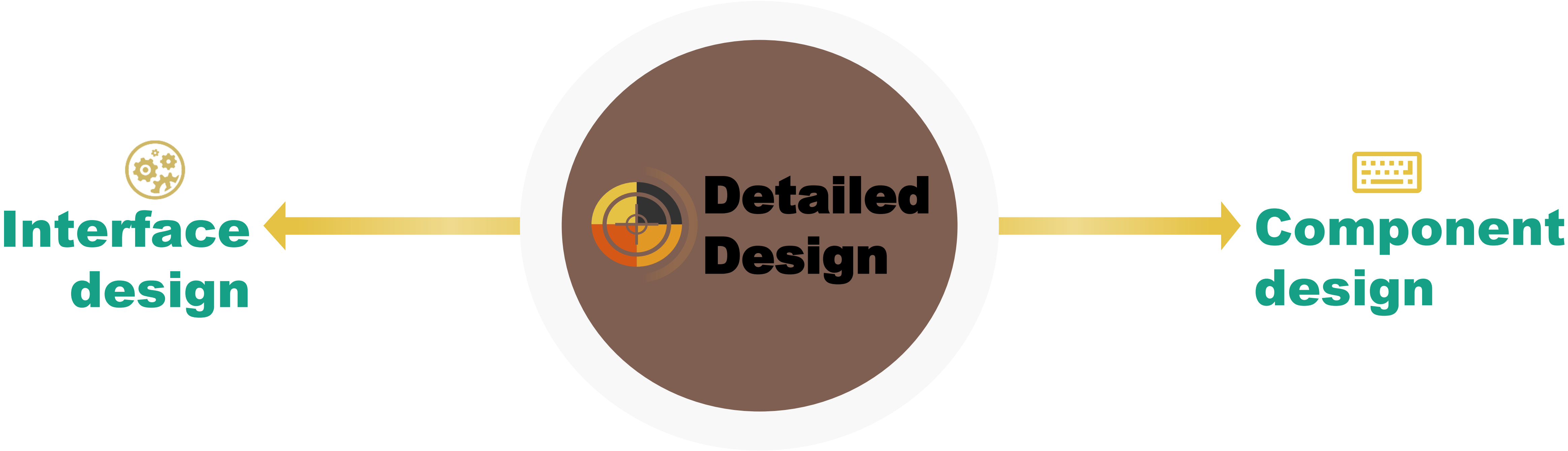
Functional

Focuses on functional requirements, whereas the architecture focuses mostly on non-functional, or quality, requirements

Follows architecture

Refines the architecture to reach a point where the software design, including architecture and detailed design, is deemed sufficiently complete for the construction phase to begin.

DETAILED DESIGN



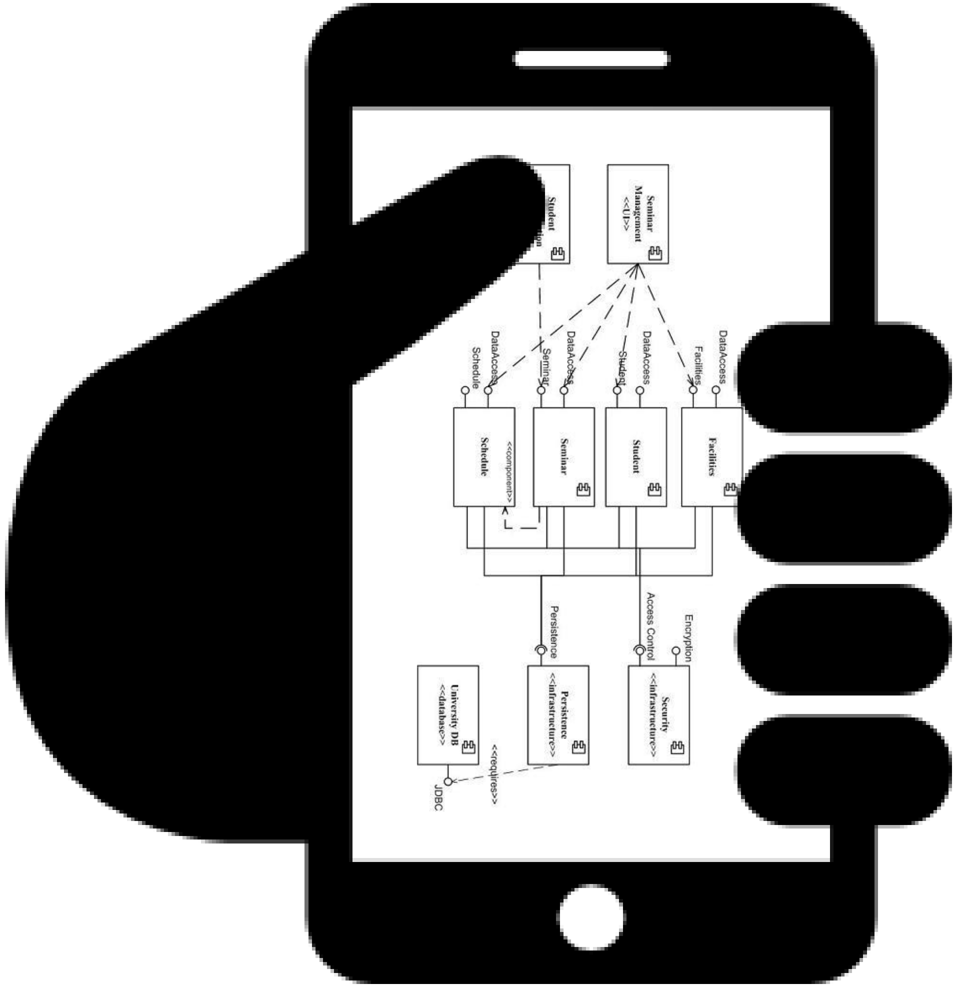
INTERFACE (NOT GUI) DESIGN

Specification of **interfaces** between **components**

Provide standardized way of **accessing** software **components**

Allow multiple **development** effort to occur in **parallel** as long as interfaces are obeyed.

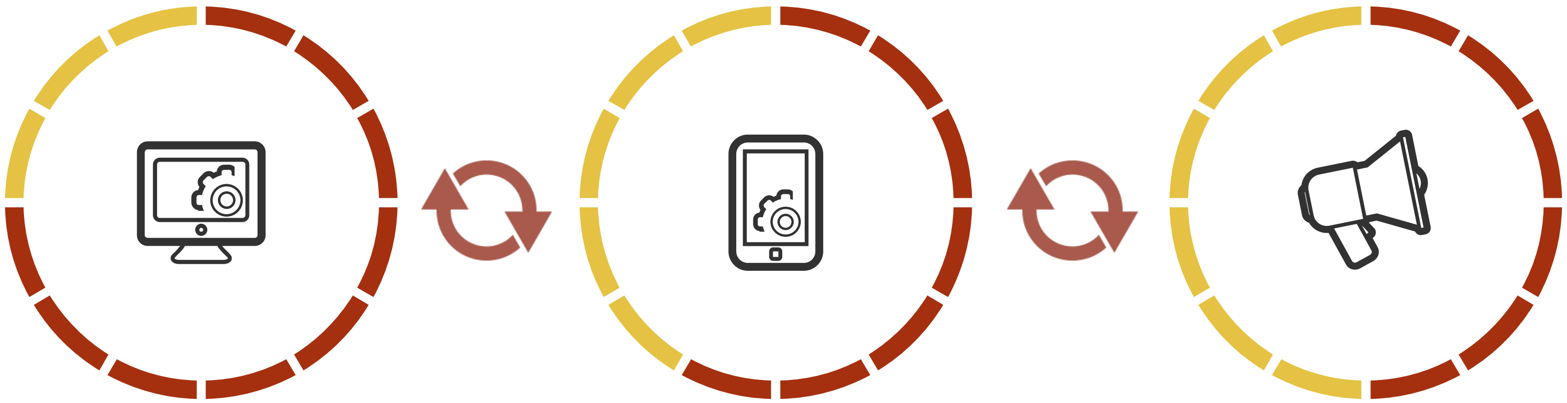
COMPONENT DESIGN



Internal design (**structure** & **behaviour**) of components is created.

In OO design, components are designed by using class diagrams and sequence diagrams.

SOFTWARE DESIGN PROCESS



ARCHITECTURE DESIGN

DETAILED DESIGN

CONSTRUCTION DESIGN

It is seen that architectural designs are elaborated through detailed design, which are further elaborated through construction design. In practice, design activities are not carried out in such orderly and controlled fashion. Quite a bit of iteration occurs between these activities.

SOFTWARE DESIGN & QUALITY ATTRIBUTES

SINGAPORE INSTITUTE OF TECHNOLOGY

WHAT IS A GOOD PRODUCT?

Notes from the video and discussions:

FUNCTIONAL VS. QUALITY - EXAMPLE

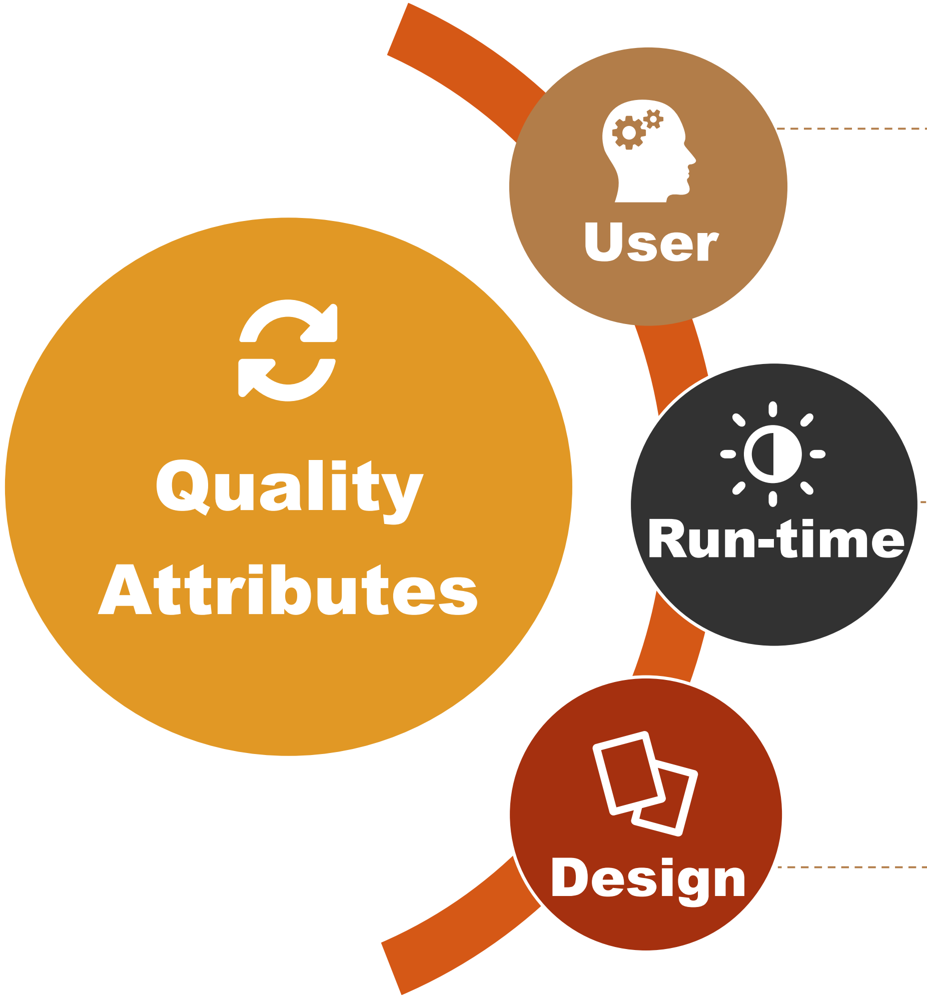
“When the user presses the green button, the Options dialog appears”

how quickly the dialog will appear

how often this function will fail,

how quickly it will be repaired

MICROSOFT QUALITY ATTRIBUTES



Usability: How well the application meets the requirements of the user and consumer, resulting in a good overall experience.

Availability: Defines the proportion of time that the system is functional and working

Performance: Indication of the responsiveness of the system.

Reliability: The ability of the system to remain operational over time

Scalability: Ability of a system to handle increasing load or to be readily enlarged.

Security: Capability of a system to prevent malicious or accidental actions

Conceptual Integrity: Defines the consistency and coherence of the overall design

Maintainability: Ability of the system to undergo changes with a degree of ease.

Reusability: The capability for components and subsystem to be suitable for use in other applications and in other scenarios.

Testability: How easy it is to test the application

DESIGN TACTICS

I want usability!

Customer

Software Architect

Tactic #1: support undo operations
Tactic #2: Provide cancel option

Tactics for Security [1]

- Resisting Attacks: Authenticating users, Limit exposure, Limit access, only on need-to-know basis, etc.
- Detecting Attacks: Intrusion detection

Tactics for Testability

- Event logging: Log data and operations throughout the system. Allow testers to enable/disable this feature so that when enabled, events are displayed in the console to give insight into the system's operations and data.

Tactics for Maintainability[1]

- Localize changes: Modularization, abstraction, encapsulation
- Prevention of ripple effects: Encapsulation, reduce coupling

Tactics for Availability:

- Redundancy, Task monitor, Watchdog timer, etc.

Tactics for Performance [1]

- Increase computation efficiency, reduce computational overhead, introduce concurrency, etc.

PROCESS FACET OF DESIGN QUALITY

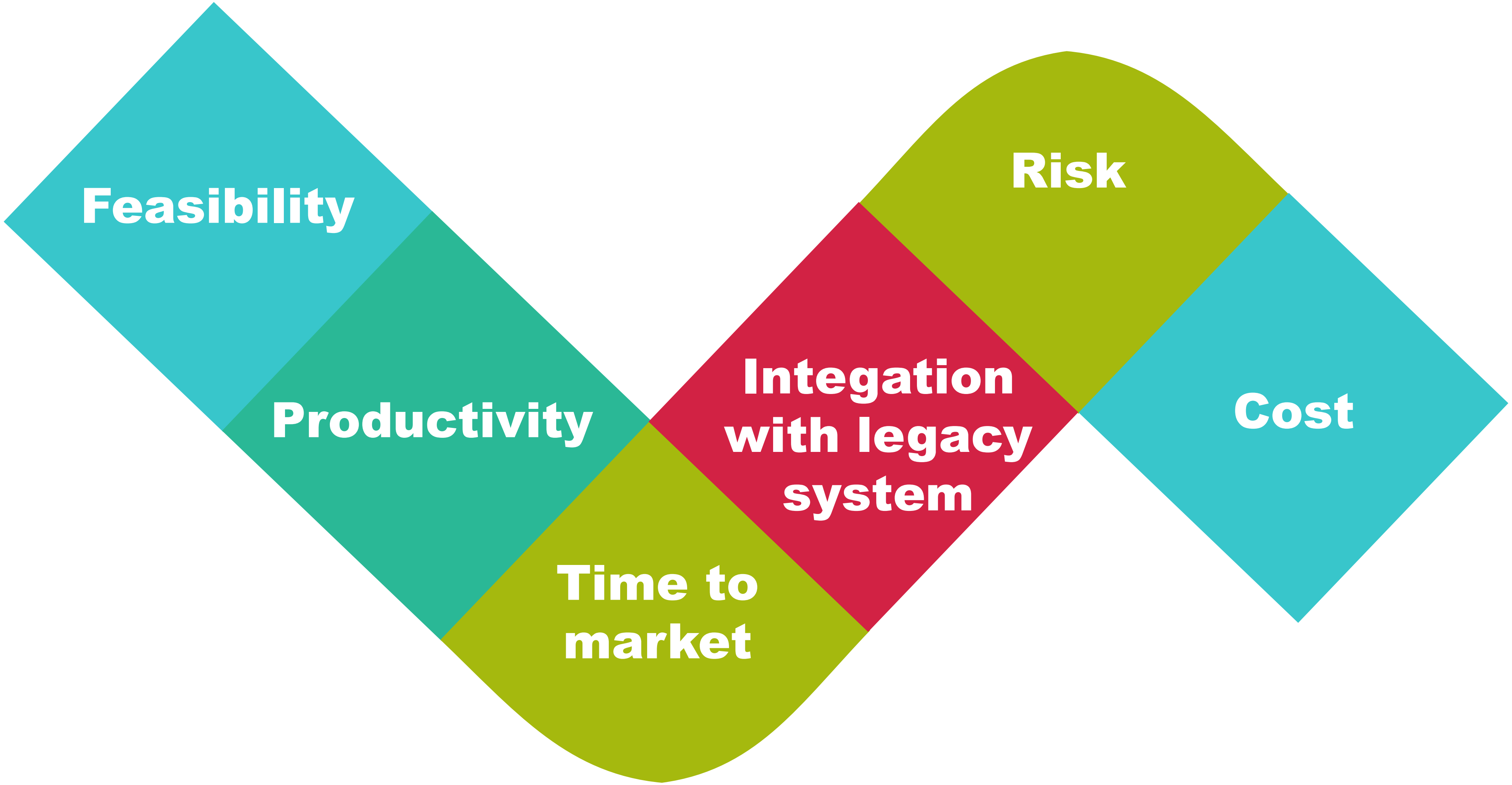
Software design does have a process facet such as:

- Project Schedule & Milestones
- Development team organization
- Test plan etc.

This topic has been studied in many modules of this degree program

- ICT2101 Introduction to software engineering
- ICT3104 Software management

PROCESS-ORIENTED QUALITY ATTRIBUTES



Feasibility

Productivity

Time to market

Integation with legacy system

Risk

Cost

ELECTRONICDEALS CASE STUDY

Can be downloaded from LMS.

QUALITY ATTRIBUTES & DESIGN TACTICS

Scalability

- Peak user load: 10k users/hour in peak load, 100 users per second average load.
- Concurrent user load: 5000 users

Design consideration for scalability

- hybrid clustering at all layers: Web server, application servers, database servers have a primary clyster and a standby cluster.
- Failover, replication. Redundancy

QUALITY ATTRIBUTES & DESIGN TACTICS

Availability

- The online application should be available 99.999% of the time across all the geographies where ElectronicsDeals Inc. operates.
- The services exposed by ElectronicsDeals Inc. should be available 99.99% of the time.

Design Tactics

- Very high availability for the global gateway page and product home page: Cache the two pages. CDN server
- High availability for services: distributed ESB server. Cluster ESB, the robustness of the ESB would further increase because it provides redundancy and failover support.

QUALITY ATTRIBUTES & DESIGN TACTICS

Some design Tactics for performance:

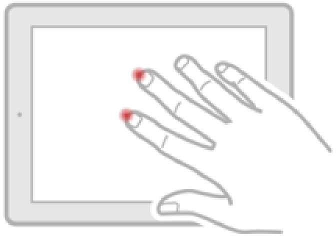
- Responsive design: So the application can be delivered for all desktop browsers and devices.
- Lightweight design and asynchronous on-demand data loading: using client-side AJAX technologies and load the content asynchronously.
- Layer-wise caching: A host of caching techniques are used to provide optimal performance. Search results for keyword are also cached.
- Using a lot of caching for optimization.
- End-to-end performance testing to monitor.

KEY CONSIDERATIONS IN SOFTWARE DESIGN

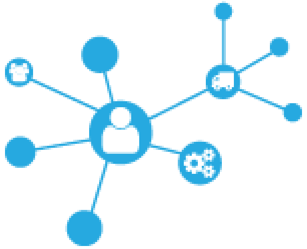
KEY CONSIDERATIONS IN SOFTWARE DESIGN

Concurrency

Event Handling



Distribution of Components



Security

Exception & Error handling

Interaction & Presentation

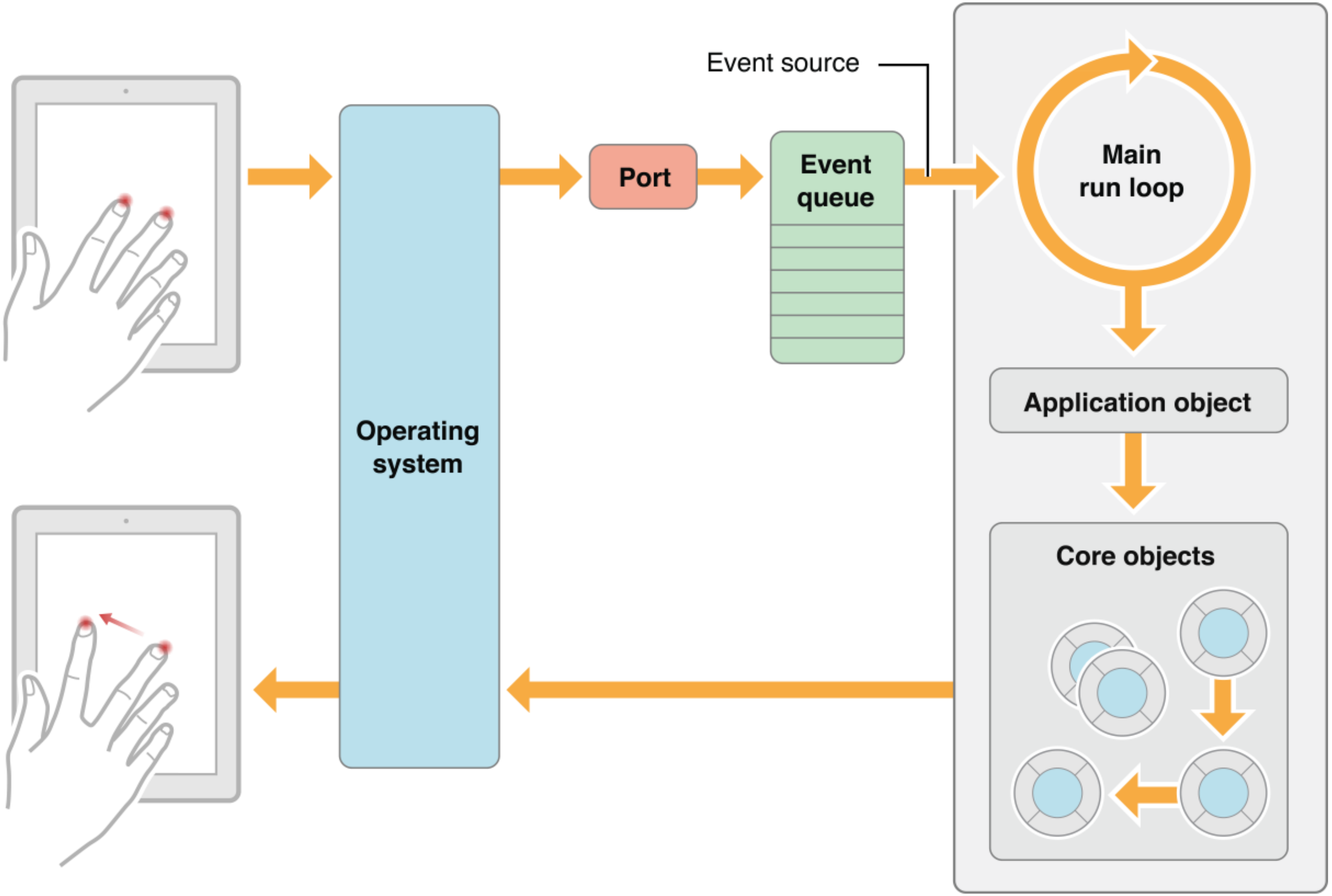
Data & Persistent storage

CONCURRENCY

Concurrent use of shared resources may also lead to issues such as **deadlock** and **starvation**

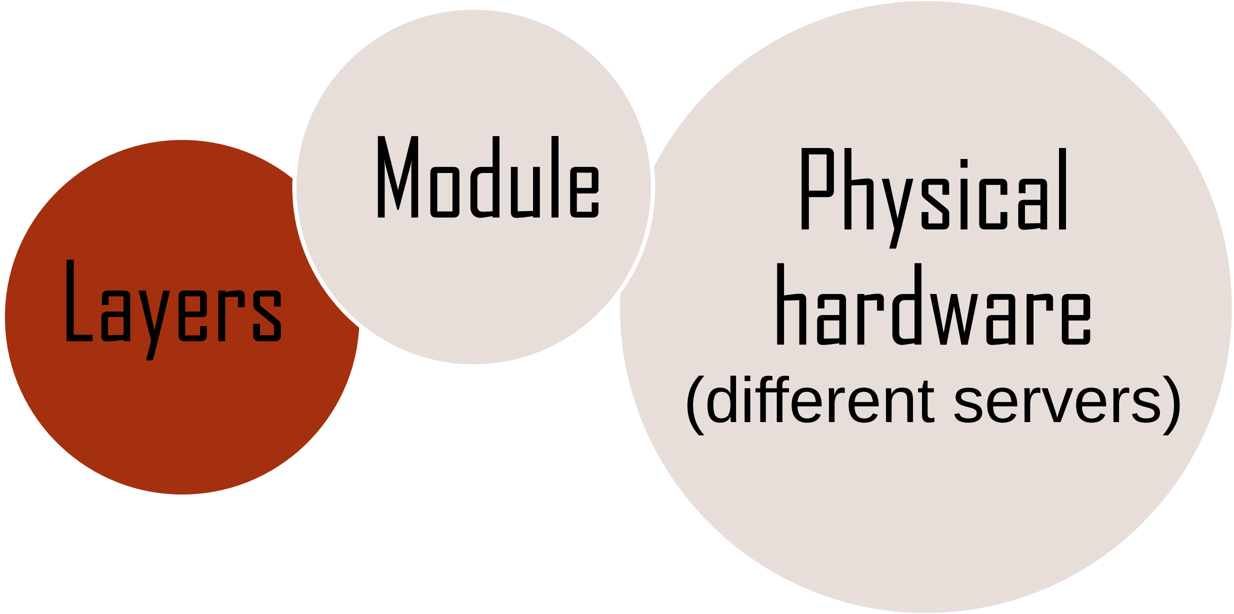
Stop blaming me for being too SLOW. You are the one who still type with one finger!

CONTROL AND HANDLING OF EVENTS



DISTRIBUTION OF COMPONENTS

Software components
can be distributed into:

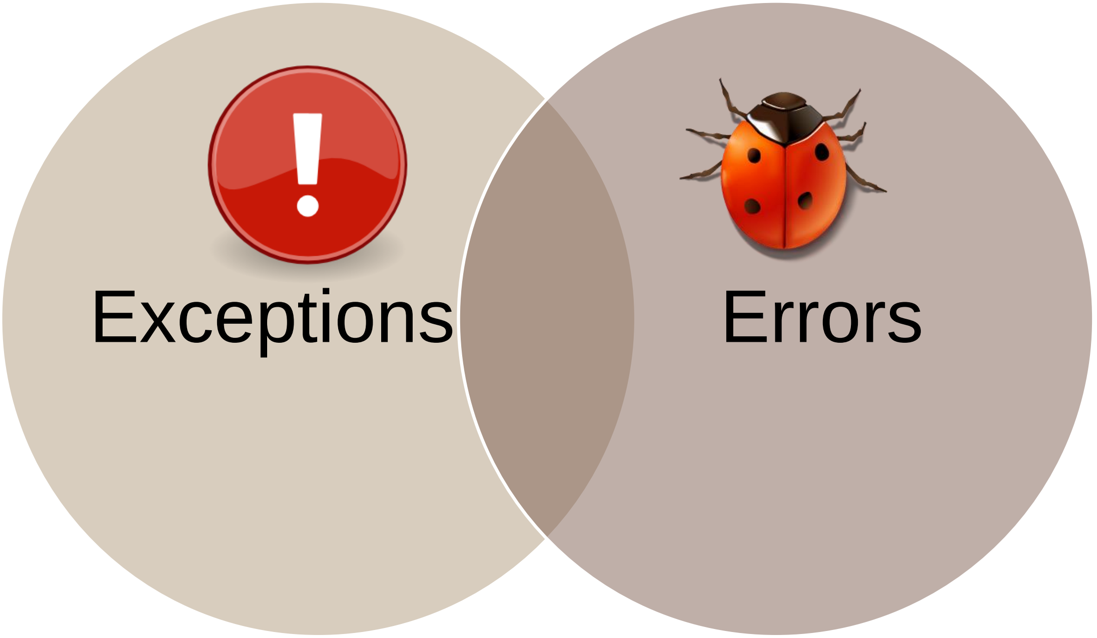


Layers

Module

Physical hardware
(different servers)

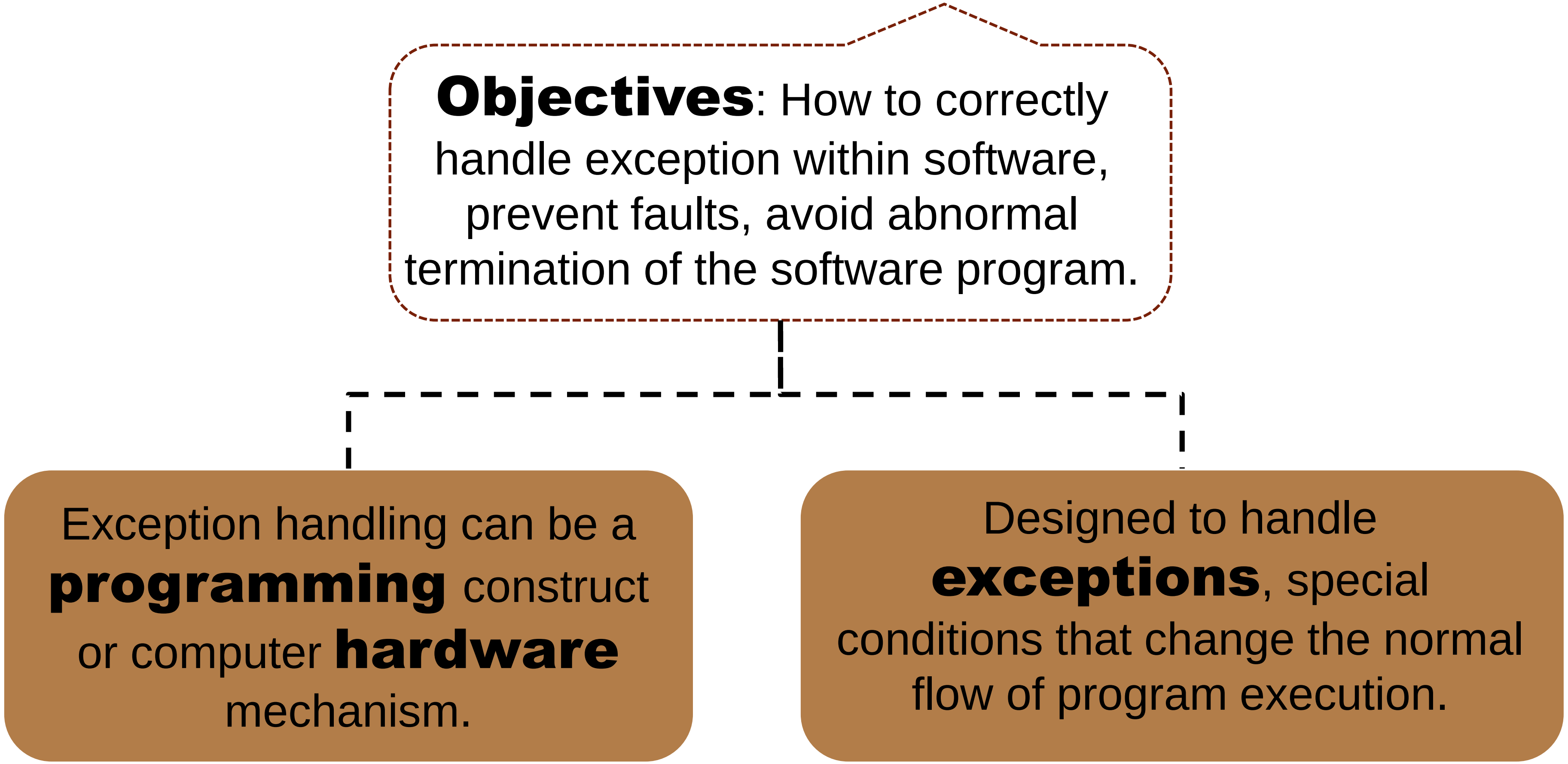
EXCEPTION & ERROR HANDLING



Exceptions

Errors

EXCEPTION HANDLING



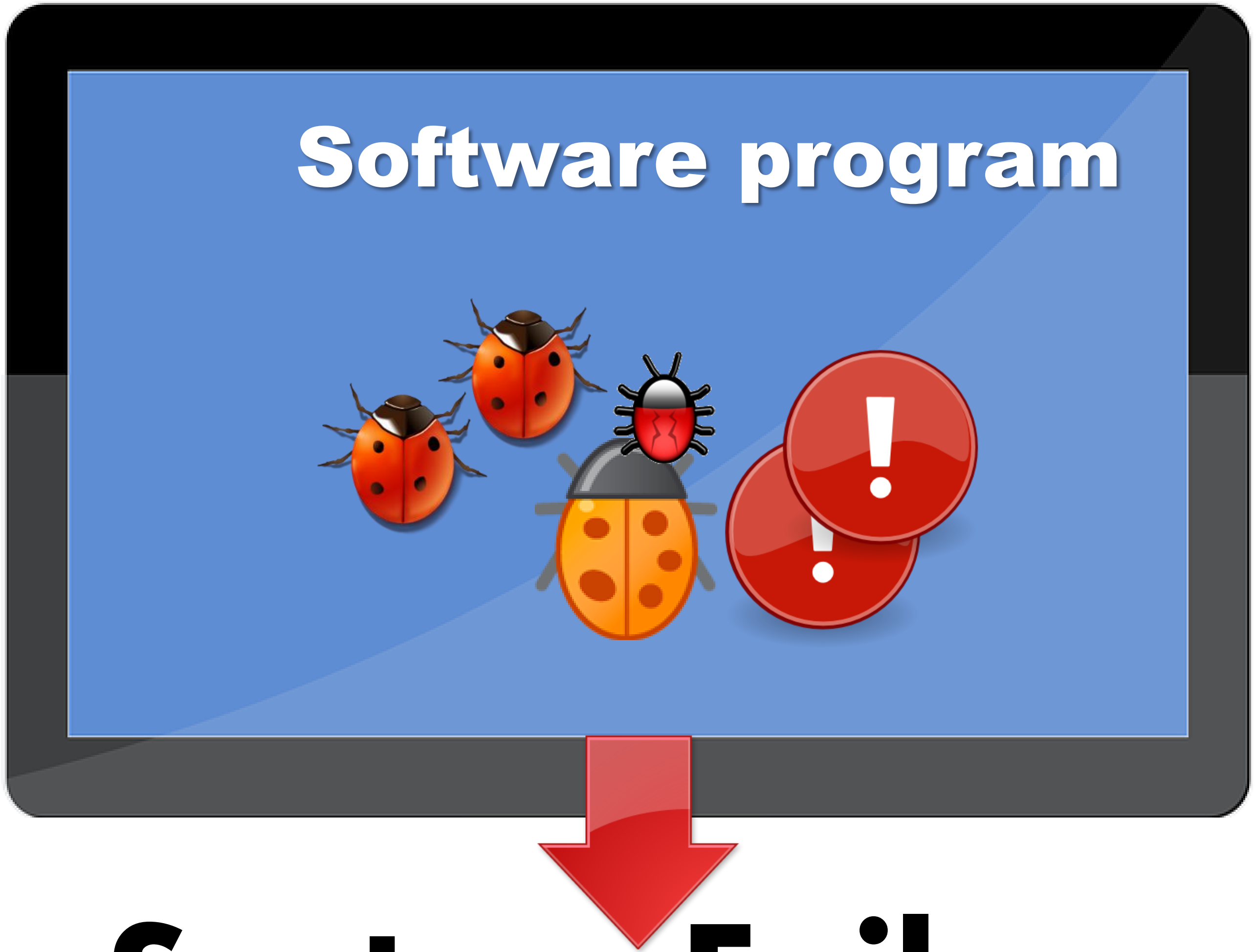
Objectives: How to correctly handle exception within software, prevent faults, avoid abnormal termination of the software program.

Exception handling can be a **programming** construct or computer **hardware** mechanism.

Designed to handle **exceptions**, special conditions that change the normal flow of program execution.

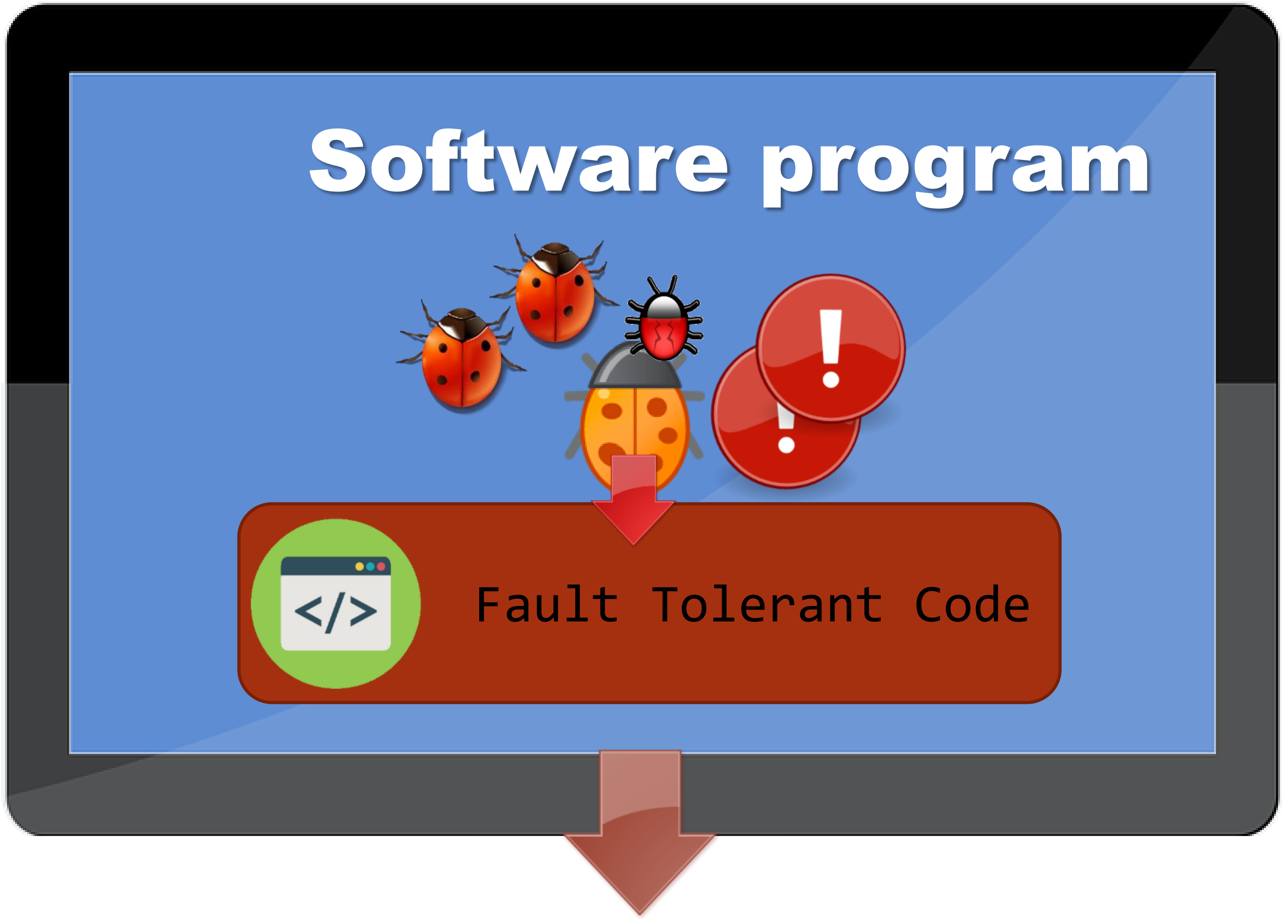
ERROR HANDLING

Software bugs, describe a flaw, mistake, failure or fault in a computer program or system that produces an incorrect or unexpected result, or cause it to behave in unintended ways.



Software program

System Failure



Software program

Fault Tolerant Code

Normal processing

INTERACTION & PRESENTATION

To work with a system, users have to be able to **interact** and control the system. How to **structure** and **organize** with **users**? How to present information to users?

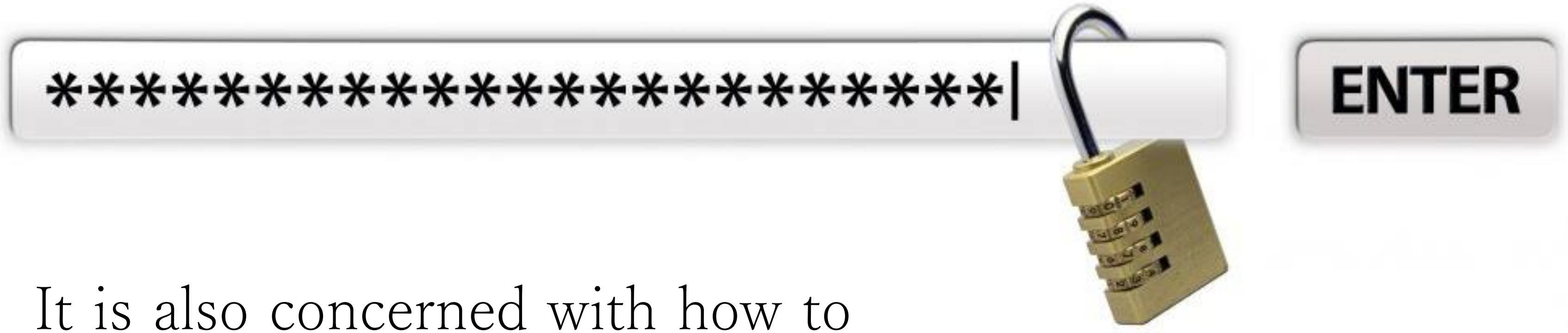
DATA PERSISTENT AND STORAGE

Persistence: The actual storage of data from a software program in a database during the execution of a software program.

SECURITY

Design for security is concerned with how to **prevent unauthorized** disclosure, creation, change, deletion, or **denial of access** to information and other resources.
(More in *Secure Software Development*)

SECURITY



It is also concerned with how to **tolerate** security-related **attacks** or violations by limiting damage, continuing service, speeding **repair** and **recovery**, and failing and recovering securely.

WE HAVE COVERED…

1. Motivation for Software Engineering
2. Importance of Software Design
3. Software Design Process
4. Software Design & Quality Attributes
5. Key Considerations in Software Design