

## Today

Last time:

Logical Architecture (ref.: chapter 30)

Theme: layers

This time:

Architectural Analysis (ref.: chapter 32)

Architecture is very important!

More important than implied by the "32"

After class please see me: Tamim

## Architectural Analysis, Software Engineering, and OOA/D – Why Know About Them?

- Many ECE students (including EEs)
  - Will gain employment in software development
  - Will have co-workers...
    - ...who understand S.E. ideas (younger ones, often)
    - ...who understand little about S.E. (older ones, often)
  - Its good to be able to speak the "language"

## Architectural Analysis, Software Engineering, and OOA/D – Why Know About Them?

- The ideas in this course are
  - "ahead of the curve"
  - That is, iterative development / UP is
    - newer than Waterfall-type models
    - Likewise, OOA/D is newer than classical software eng.
      - (But still, it's pretty well established at this point)
- Therefore industry/gov't will use
  - OOA/D and UP
  - Increasingly over the next several years

## Architectural Analysis, Software Engineering, and OOA/D – Why Know About Them?

- Thus ideas in this course will become...  
...increasingly useful over time
- So they will help you stay at the cutting edge
  - ...over most of a career
  - (compared to, say, taking an "industry-approved 7-week programming course")
    - A college degree in ECE is better!
    - ...and a college degree in ECE needs to include:  
**software engineering ideas with future relevance**

## Architectural Analysis, Software Engineering, and OOA/D – Architectural Analysis

- Recall **logical architecture** from last time:
  - It involves structuring the software with respect to
    - Non-functional requirements
    - These include reliability, security, performance, maintainability, etc.
    - Maintainability, for example, is extremely important
      - (and is supported by OOA/D and the layers pattern)
- Let's look more at developing **architectures**

## Architectural Analysis – Some Vocabulary & Concepts

- Software architecture
  - Organization of a software system (p. 448)
- Architectural investigation
  - Process of determining what (non-)functional requirements impact system design (p. 448)
- Architectural design
  - Process of determining a software structure to match the architectural investigation (p. 448)
- Architectural analysis
  - Architectural investigation and design (p. 448)

## Architectural Analysis – Thinking About It

- Software architecture
  - What is it?
- Architectural investigation
  - Process of determining what (non-)functional requirements impact system design (p. 448)
    - What are some non-functional requirements
- Architectural design
  - Process of determining a software structure to match the architectural investigation (p. 448)
  - What is the *product* of this *process*?
    - Hint: it is mentioned on this slide
- Architectural analysis
  - Architectural investigation and design (p. 448)

## Architectural Analysis – Concepts & Vocabulary

- Software architecture, Architectural investigation  
Architectural design, Architectural analysis
- Which one is the result of architectural analysis?
- Which two are parts of architectural analysis?
- Which one occurs after another?
- Which one occurs before another?

## Architectures – More Concepts/Vocabulary

- Architectural patterns (i.e. design patterns)
  - These are the patterns we have been learning
  - One is low coupling
    - ...which also applies to non-functional design (as we've seen)
    - ...does it apply to functional design?
  - A key pattern discussed recently is **layers**

## Architectures – More Concepts/Vocabulary

- Architectural patterns, design patterns, low coupling, functional, non-functional, layers
  - Which one(s) are architectural patterns?
- Which pattern is a property of which other pattern?
- Which term is a special focus of most patterns?

## Architectures – More Concepts/Vocabulary

- Architectural *view*
  - A perspective from which an architecture is seen
  - Examples (pp. 501-502)
    - Logical architecture view (e.g. the layers)
    - Deployment architecture view (e.g. which modules run on which machines)
    - Process architecture view (e.g. processes and threads)
    - Data architecture view (e.g. persistent data – files/DB/etc.)
    - Use case architecture view (e.g. most important use cases and their non-functional requirements)
    - Implementation architecture view (e.g. list of code files, documentation, etc.)
  - These are defined in the UP (other views are allowed too)

## Architectures – More Concepts/Vocabulary

- Logical architecture view
- Deployment architecture view
- Process architecture view
- Data architecture view
- Use case architecture view
- Implementation architecture view
- Which go with which???
  - (e.g. most important use cases and their non-functional requirements)
    - (e.g. list of code files, documentation, etc.)
    - (e.g. the layers)
    - (e.g. persistent data – files/DB/etc.)
    - (e.g. which modules run on which machines)
    - (e.g. processes and threads)



## An Architecture Example

- For the POS system, here is an architectural decision:
  - Have a single, remote tax calculator
  - Have a copy running on every cashier's computer
- Which is better?
  - Consider **reliability**, **response time**, ability to calculate and display tax on **each item** vs. tax for the **entire sale**, ease of **updating** the calculator, **licensing cost** of 3<sup>rd</sup>-party tax modules, and **any other factors** you think of
  - Think-pair-share –
    - take 2 minutes to think
    - then I'll call time and you can discuss with 0-n neighbors
    - lucky groups then can tell the class their conclusions



## Architectures – Further Concepts & Vocabulary

"Architecture" is clearly a many-dimensional idea!

- Architectural drivers:
  - Non-functional (and functional) requirements that impact architecture
  - A bell/whistle is not an architectural driver
- Architectural factors: Architectural drivers
- Architectural decisions:
  - These define solutions to the architectural drivers, such as: Software structure, removal of requirement, terminate project, etc.



## Architectures – Further Concepts & Vocabulary

- Architectural drivers
- Architectural decisions
- Architectural factors
- Do any refer to software structures?
  - If so, which?
  - Exclusively so?
- Are any synonymous?
  - If so, which?
- Which is/are the result of which?



## Architectures and FURPS+

- Recall FURPS+
  - Functionality, usability, reliability, performance, supportability, etc.
  - In the UP framework, each requirement is categorized in a FURP+ category
  - Which letter goes with which category?
- These help define the factors in a **factor table**
  - (See Table 32.2)