Introduction
Just What is Software Engineering?

Dr. Michael F. Siok, PE, ESEP
University of Texas at Arlington (UTA)
Computer Science and Engineering

CSE 5324: Software Engineering: Analysis, Design, and Testing

Textbook: Object-Oriented Software Engineering: An Agile Unified Methodology, by David C. Kung, ISBN 978-0-07-3376257

Links

- Some links addressing the questions
  1. Why do I have to learn about UML - no one uses it!
  2. Why is software engineering different than programming - I'm a very good programmer
  3. What are some of the top challenges in software engineering today?
  4. What are some of the latest trends in software development?

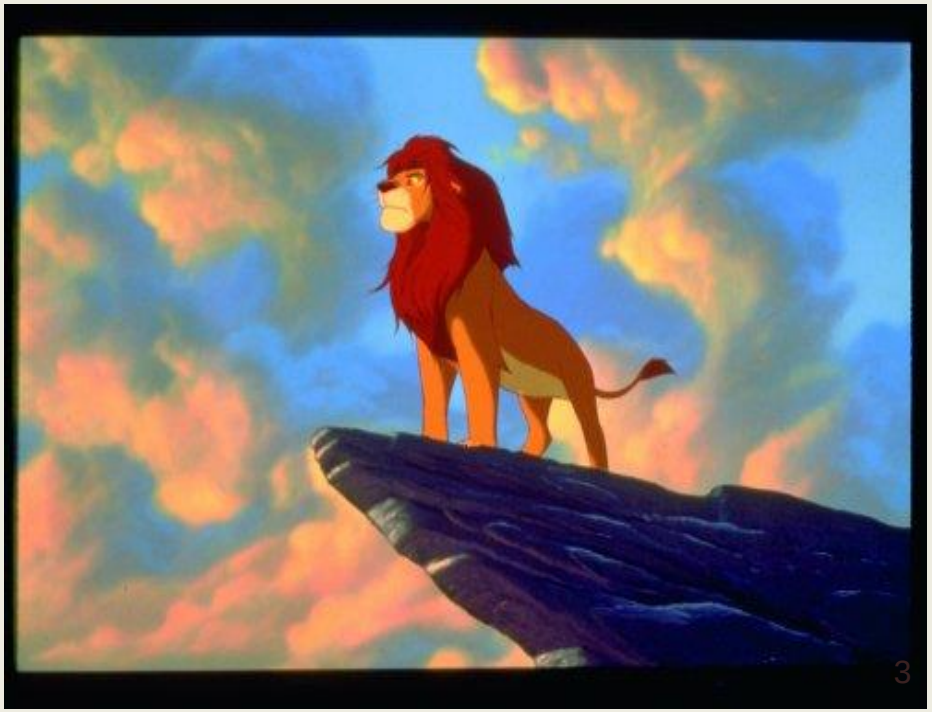
- Open Challenge (from 2011 . . . Still good)
  - http://www0.cs.ucl.ac.uk/staff/A.Finkelstein/talks/10openchall.pdf

In summary, software development is really about multiple fallible humans collaborating via some lossful natural language to precisely program essentially invisible systems based on unclear and imperfect specifications thereby creating highly compliant defect-prone systems without a definitive schedule or work predictability from which society gratefully and wishfully relies. -- Roy Oberhauser, 2013

Programming vs. Software Engineering

- ‘Programming’ relies mostly on brute force . . . the strength, perseverance, courage, and wit of the individual programmer (with little emphasis on learning, predicting, and preparing for major obstacles along the way)

- ‘Software Engineering’ is concerned with the specification, development and production of structure, machine, apparatus, manufacturing process, or product and learning from other's failures to avoid the most common pitfalls.



What is Software Engineering?

- Software Engineering is . . .
  - The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software – ***IEEE SWEBOK***
- According to the SWEBOK, software engineering activities can be divided into roughly these ten:

Software Requirements	Software Configuration Management
Software Design	Software Engineering Management
Software Development	Software Development Process
Software Testing	Software Engineering Tools
Software Maintenance	Software Quality

- The Textbook Definition – *Software Engineering* is a discipline focused on the research, education, and application of engineering processes and methods to significantly:
  - Increase software Productivity (P) and software Quality (Q)
  - Reduce software operating Costs (C) and Time-to-Market (T)

What is the State of Software Development in Industry?

- Famous quotes from some unnamed but well respected Software Engineers
  - *"The way we build software is in the hunter gatherer stage."*
  - *"Cave art. It's Primitive. We're computer scientists, but there's no 'science' here at all."*
  - *"If you bought a car with 5,000 defects, you'd be very upset."*

- Douglas Coupland's "Microserfs"
  - A hilarious but frighteningly real look at life in the software fast lane
    - *"Shipping hell continued today. Grind, grind, grind. We'll never make it. Have I said that already? Why do we always underestimate our shipping schedules?"*
    - *"In at 9:30 am; out at 11:30 pm - Dominos for dinner. And three diet Cokes."*

How would you answer the question -
"*If we are 'software engineers' what makes us an 'engineer'?*"

Questions Addressed to a Typical Software Developer in the Industry

- Some typical questions asked:
  - *"How long will it take us (you) to make the change?"*
  - *"Why does it take so long to produce our (your) software?"*
  - *"What can we do to increase our (your) productivity?"*
  - *"Why can't we (you) just short-cut the process and get this into test (or release)?"*
- How can you satisfactorily answer these questions?
  - Hint: this is where the term "*Engineering*" comes into play
- How do we distinguish Science from Engineering?
  - The latter is primarily concerned with the specification, design, manufacture, and production of an economical structure, machine, apparatus, process, product, or system
  - The former is the systematic study of the nature and behavior of the material and physical universe based on observation, experiment, and measurement, and the formulation of laws to describe these facts in general terms

A Definitive Survey of The Software Industry

- Based on the 1,200+ organizations assessed by the Software Engineering Institute
  - 46 percent have no defined processes
  - 32 percent have some basic process defined but at a project level only
  - 16 percent have all processes defined at a site/organization level
  - 4 percent manage projects using quantitative methods
  - 2 percent have continuous improvement processes

- How effective is a football or dance team that doesn't have a play book or a choreograph?

- How can work be standardized when almost half of the organizations have no defined software processes?

- For a partial list of extremely costly software failures visit http://en.wikipedia.org/wiki/List_of_software_bugs

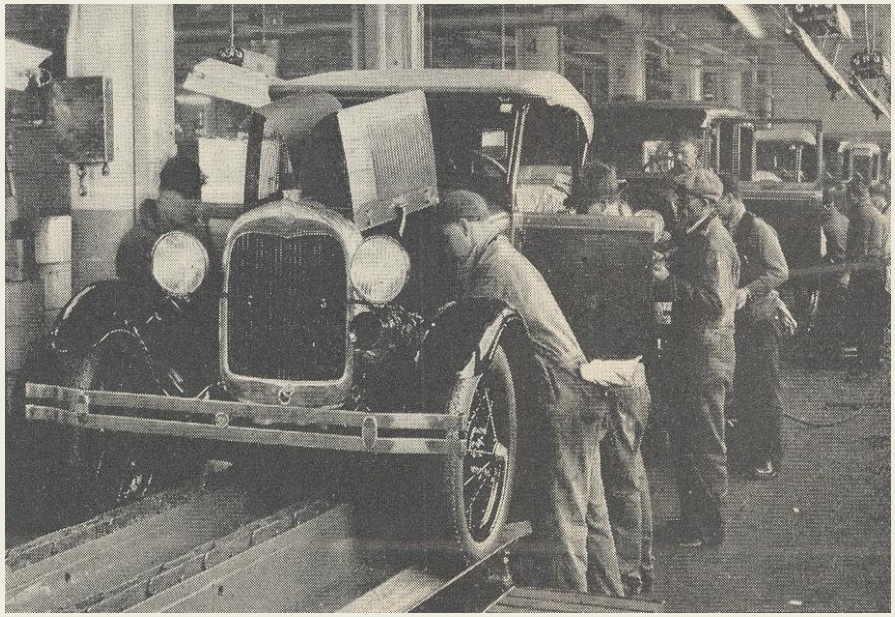
Studying the Production Line

- How is the study of the general production line and techniques related to engineering?
- Industrial Engineering is concerned with the application of science to the design of processes and production systems
- This is accomplished by using engineering techniques to study and model production line
  - Rates (production, defect, and efficiency)
  - Processes (quality, bottlenecks, cycle time, push/pull systems)
  - Tools (efficiencies, steps, work procedures)
- This is the essential quality of software engineering – development of models, laws, tools, or theorems to drive the construction of a product in a systematic, disciplined, and quantifiable manner, leading to improvements in:
  - (P)roductivity
  - (Q)uality
  - (C)osts
  - (T)ime

The Ford Production Line

-- A Look At Standardizing Work

Ford's line assigned workers to a specific production task



This work is in the public domain

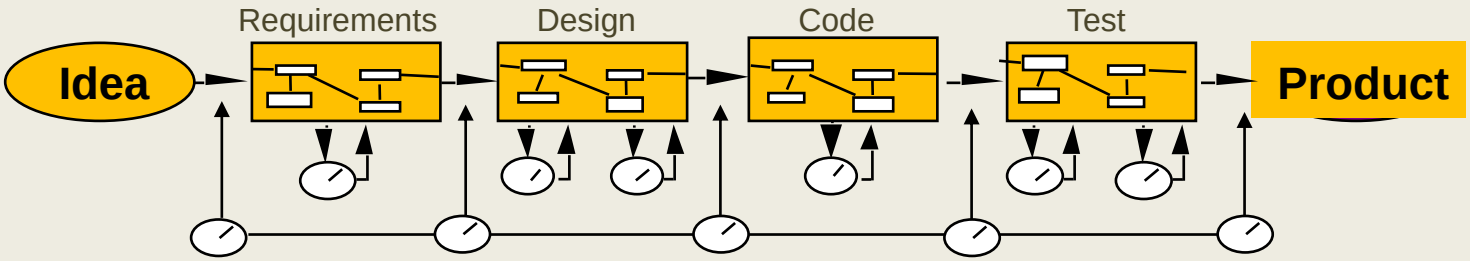
- Each production task had a station where specific
  - Work standards, tools, and measures were utilized

- Standardization of work
  - This is the key concept
  - The lack of Work standards, tools, and measures is equivalent to custom building the automobile

The production line revolutionized the industry producing cars 8x as fast as before – replacing hand-crafted cars with cars developed by the standardization of work

Software Development As A Production Line

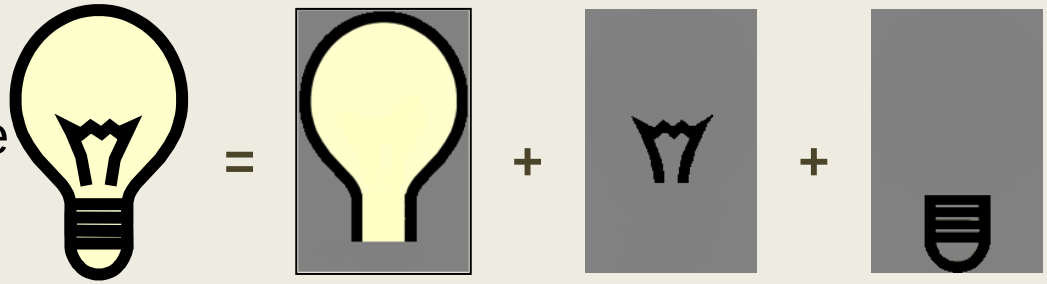
- Software development is a very complicated endeavor in industry
  - "The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software…"
  - Unlike an assembly line software development is a creative activity, but shares many similarities to the assembly line
    - Team development is the norm
    - Large complicated systems require piecewise refinement of requirements , design, etc.
    - Work moves from one step to another until a complete product is developed
    - At each step specific tools, techniques and measures are employed to ensure quality – at each step the work product is transformed



- Software Engineering is both a development (production) process and a creative process

Measuring and Addressing Quality Is Essential

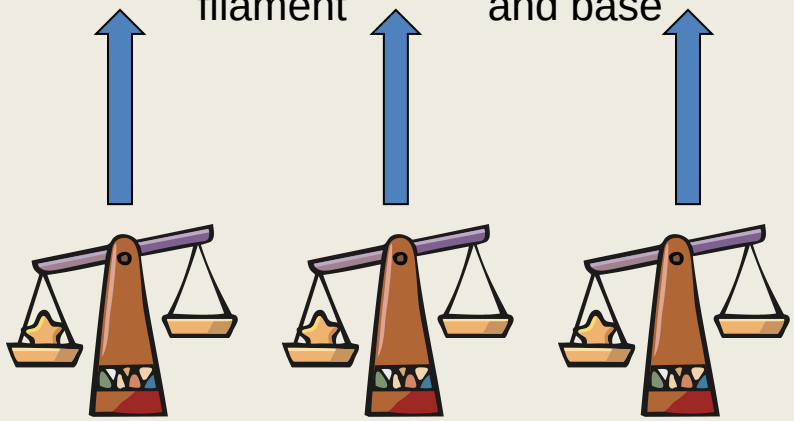
Example Production Line



What happens if I have a new person blowing the glass bulbs?

- How does this affect the later steps?
- Where does the assembly error appear to come from?

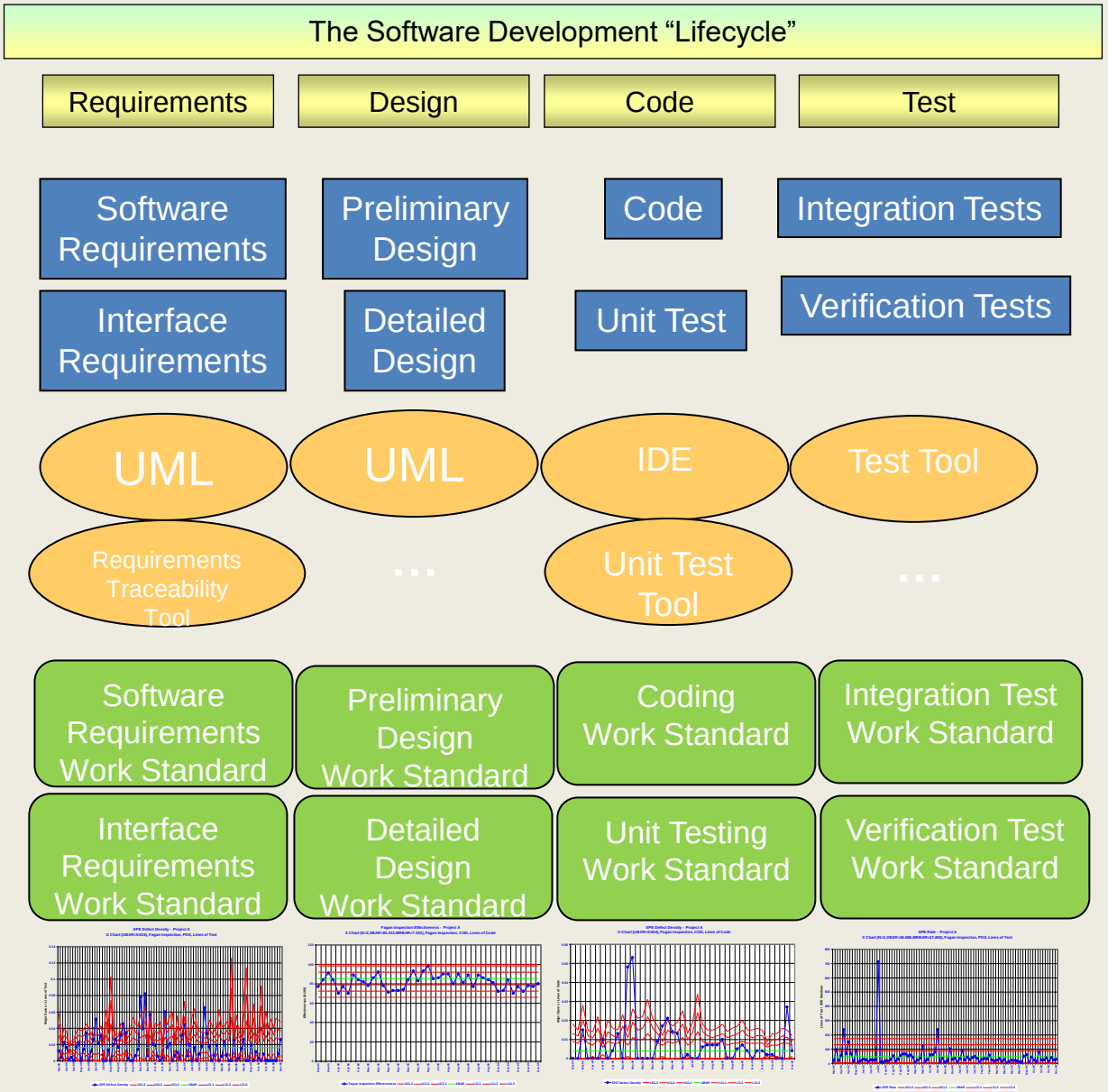
- Quality Control vs. Quality Assurance



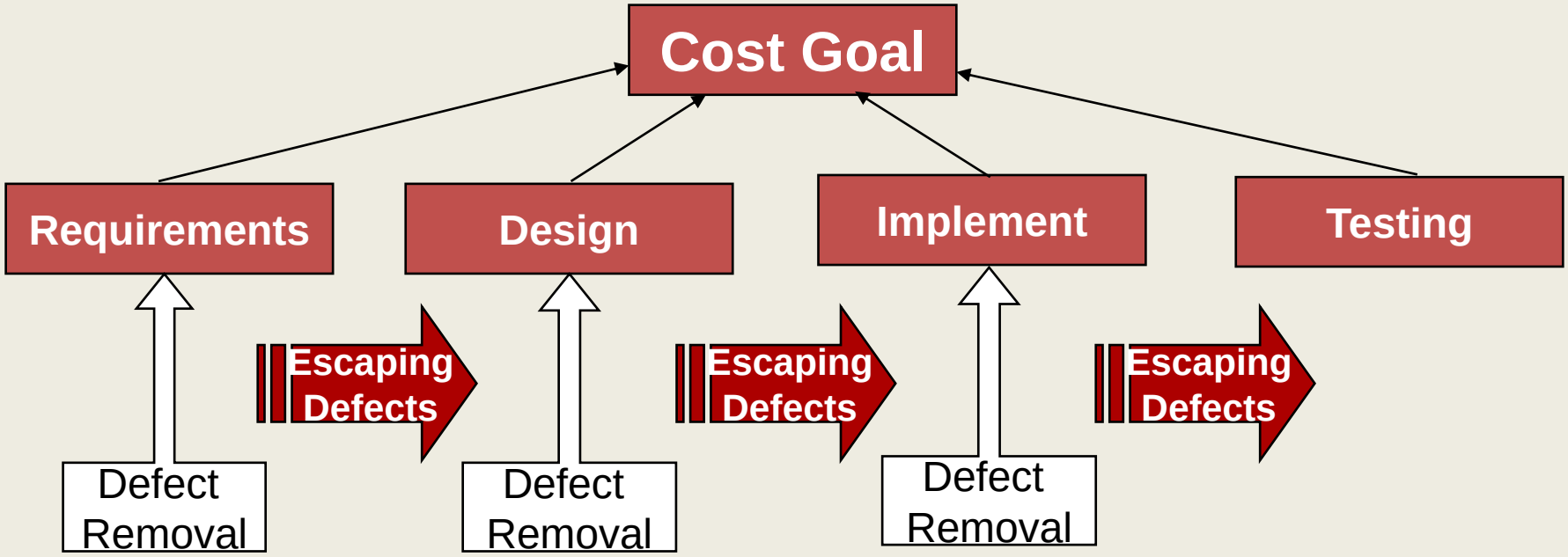
Tools and procedures help to ensure a product can be produced. Standards and measures help to ensure product can be efficiently produced.

The “Best” Production Lines have all four:
Tools, Procedures, Standards, and Measures

Software Lifecycle As A Production Line



Production Techniques Must Address Work Defects



- Need to understand how many errors we will have and the re-work cost associated with them
- Need to remove as many defects as the budget/schedule will allow and the delivered product can tolerate (*Can we find them all?*)
- We will develop and use life-cycles, processes, and methodologies that deliver the best approach that meet cost, schedule, and quality needs

"How do I develop a software product from customer concept into a working product?"

- Should I just slam out some code? How well will that work?
- How do I communicate with the customer and capture the essential things they need/(want) in the software?
- How do I decide what the software architecture should look like? Does it matter?
- How do I model user interaction?
- How do I design user interfaces?
- What are design patterns and how should I use them?

How Does Software Differ from Other Engineering Disciplines?

- **Student comments:**
  - Newer discipline
  - Easier to make revisions to software
  - Innovation/pace of change
  - Not physical – more ways software can break
  - Laws underneath are more complex
  - Can have many purposes and can change
  - Not as much time spent testing
  - Not required to be as robust
  - Management difficult because hard to measure quality/intangible

How Does Software Differ from Other Engineering Disciplines?

- **My answers:**
  - Software is designed, not manufactured
  - Some re-use achieved in practice
  - Software needs to be practiced and managed like an Engineering discipline
  - Software is malleable
  - Can apply to huge variety of problems
  - Software doesn’t wear out, but it can become obsolete
  - All problems/bugs are "designed in"

Can we even get the software job done?

- Outside of small jobs, software develop… …y in industry
  - Small job – 10,000 Delivered sour…
    - SW Engr production rate – 100 DS…
      - *Requirements, design, code, te…*
    - 10,000 DSLOC/(100 DSLOC/day) = 10…
    - 19 to 21 working days per month ≅ 5 m…
  - Medium size job = 100,000 to 500,000 D…
    - 500,000 DSLOC/100 DSLOC/day = 5,000 da…
  - Very Large job = 1,000,000 to 10,000,000 D… …years

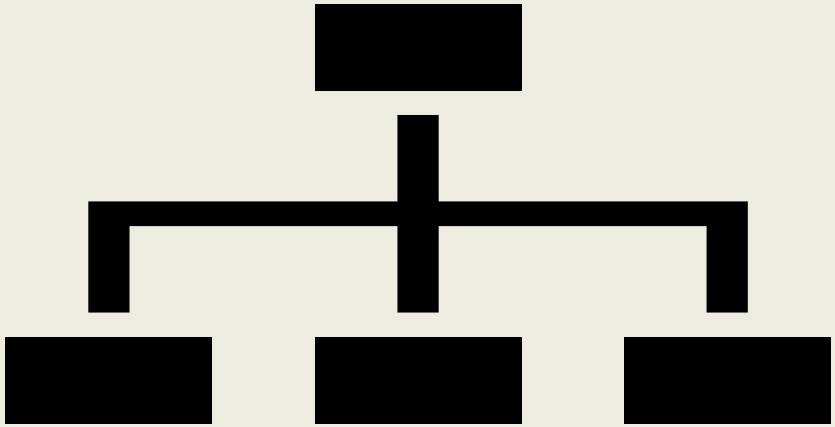
By contrast, COCOMO II estimates a 500,000 DSLOC software project will take a staff of 60 people 3.75 years to complete the software product design, programming, and test with an average productivity rate of 183 DSLOCs/person/day for a software development cost of about USD41M (at USD100/hr loaded labor rate).

-- but with lots of assumptions --

- Clearly, to manage a software development job of …y significant size and scope and finish in a '*reasonable*' amount of time within a costing framework to produce a needed and usable software product requires a team of developers AND a plan
  - *Leads to . . . "The Business of Software"*

3 Tracks of Software Engineering

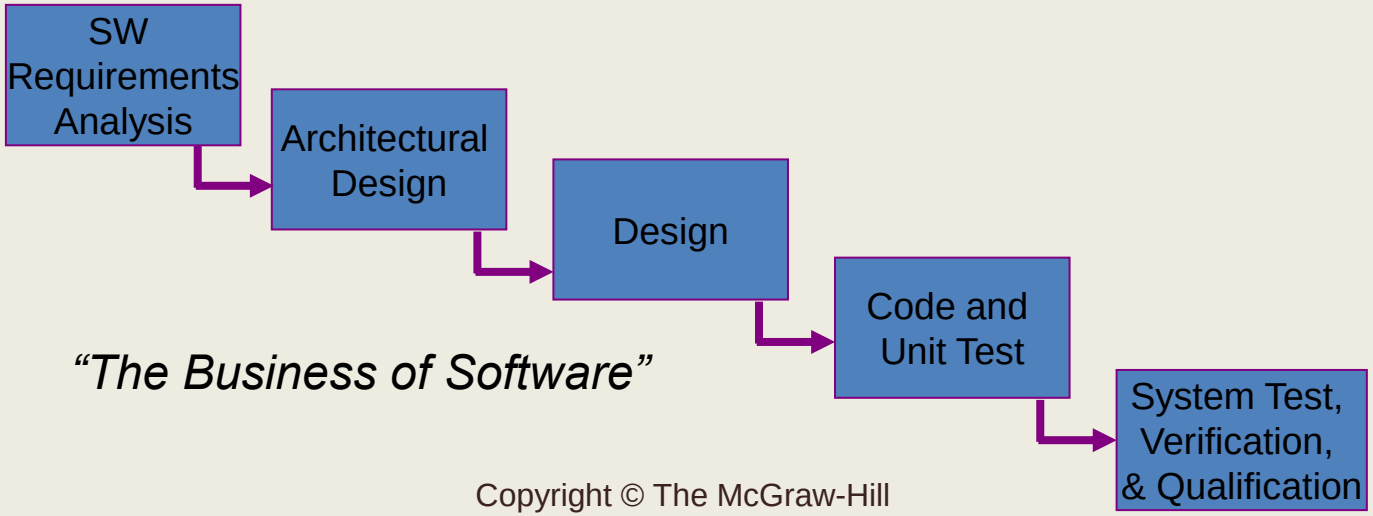
- Software engineering focuses on three major sets of activities that take place simultaneously
  - Software Development
  - Software Quality Assurance
  - Software Project Management



"The Business of Software"

Software Development

- A software development process transforms the initial system concept into the operational system running in the target environment
- It identifies the business needs, conducts a feasibility study, and formulates the requirements or capabilities that the system must deliver.
- It also designs, implements, tests, and deploys the system to the target environment
- The biggest question we will address this semester is "how do I develop a software product from a customer concept into a working product?"



"The Business of Software"

Software Quality Assurance

- Software Quality Assurance (SQA) ensures that . . . .
  - Development activities are performed properly
  - Software artifacts produced by development activities meet software requirements and desired quality standards

- Verification, Validation, and testing ensure these tasks are carried out
  - Verification – ensures products are developed 'right'
  - Validation – ensures that 'right' products are developed
  - Testing – ensures functionality implemented and available

"The Business of Software"

Software Project Management

- Software project management oversees the control and administration of software development and SQA activities
- Project management activities include:
  - Effort estimation
  - Project planning and scheduling
  - Risk management
  - Project administration, and
  - others. . . .
- These activities along with the oversight help ensure that software system is delivered on-time and within budget

"The Business of Software"

Object-Oriented Software Engineering

- Object-Oriented Software Engineering (OOSE) is a specialization of software engineering

- The object-oriented paradigm views the world and systems as consisting of (real-world) objects that relate and interact with each other
  - It provides the ability to relate the software system being developed back to the customer in terms they recognize
  - OOSE allows the customer to conceptualize users, roles, classes (objects), and top level interactions from their point of view – it eliminates a huge conceptual gap for the customer
  - It builds a conceptual model for the developers of how the customer's world operates

- OOSE encompasses:
  - OO processes
  - OO methodologies
  - OO modeling languages
  - OO tools

Ethical Responsibility

- Software Engineers practice a code of ethical conduct just like in other engineering professions
  - Code expresses the consensus of the profession on ethical issues
  - Code provides a means to educate both the public and aspiring professionals about the ethical obligations of all software engineers

- *'Software engineers will commit themselves to making software a beneficial and responsible profession'*
  - 8 Principles that define the practice

SE Code of Ethics and Professional Practice

- https://www.computer.org/education/code-of-ethics

- Developed by ACM and IEEE Computer Society jointly; published in 1999

- Review this Full Version Preamble (3 pages)
  - Understand why this was needed
  - Understand the 8 principles and clauses

SE Code of Ethics and Professional Practice (Cont'd)

- You will be faced with ethical dilemmas in your career
  - Hardly ever are the answers simple
  - Often, no right answers
  - Some issues may be relatively minor, others not so
- How you "*react*" or "*take action*" could shape the rest of your career
  - Do nothing
  - Discuss with colleagues
  - Work issues through management
  - Whistle-blower to the public
  - Resign and look for a new job somewhere else
- An appropriate ethical position depends on you and the views of others

Key Takeaway Points

- Software engineering aims to significantly improve software productivity *(P)* and software quality *(Q)* while reducing software costs *(C)* and time-to-market *(T)*

- Software engineering consists of three tracks of interacting life cycle activities:
  - Software Development Process
  - Software Quality Assurance
  - Software Project Management

- Object-oriented (OO) software engineering is a specialization of software engineering
  - It views the world and systems as consisting of objects that interact
  - It helps to effectively bridge the communication gap between developer(s) and customer(s)

Study Questions

- After study of this chapter, the student should be able to address the following questions:
  - What are the focuses of computer science and software engineering, respectively?
  - What are the educational objectives of computer science and software engineering, respectively?
  - Some authors say that software engineering is *"programming in the large."* What does this mean?
  - What is the relationship between software engineering and computer science? Can you have one without the other?

This Course's Goals

- To help you be able to develop software as a team
  - Beginning with an idea . . . and
  - Ending with the development of a successful engineered product

- To help you apply engineering to software development through:
  1. Step-wise refinement of techniques that apply proven principles to the development of software
  2. To help you become very familiar with the UML and software design techniques
  3. To provide you with a full-life cycle understanding of software development
  4. To give you an opportunity to work on a software team and develop a working Android product (a smartphone app) using our engineering techniques

Course Emphasis

- **Technical content**
  - Learn a Software Engineering approach using UML
  - Develop an Android application as part of a software team using the engineering approach for demonstration at semester end
  - Develop deliverables tied to your Android product development approach
- **Management**
  - Teamwork
  - Presentations
  - Project Planning
- **Experience**
  - Hands on work with SE processes
  - Experience developing Android applications


THE
END