CMPT 475: Software Engineering II

Introduction

Dr. Herbert H. Tsang, P.Eng., Ph.D.

Summer 2014

School of Computing Science
Simon Fraser University, Canada

References

- What is Software Engineering?

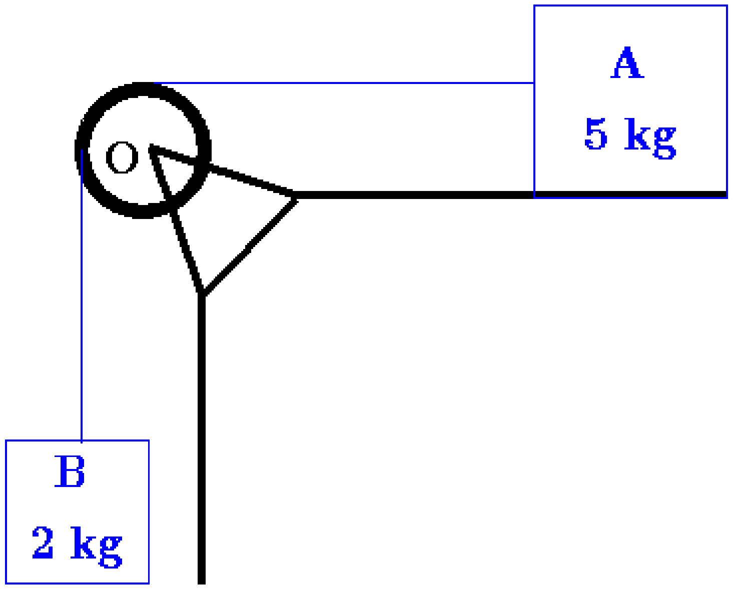
Software Engineering vs. Programming

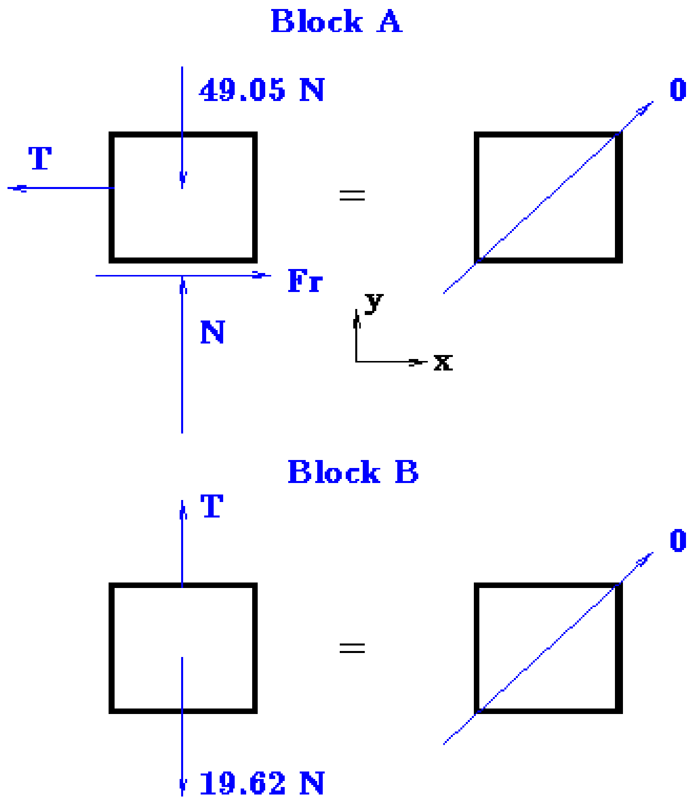
- Software engineering is about more than just programming/coding
- It is about design principles and methodologies that yield programs that are
  - Robust
  - Manageable
  - Reusable

Software vs “other” engineering

- How is software engineering *similar* to other engineering?
- **Abstraction** and **Modularity**
  - – Consider free-body diagram
  - – Thevenin/Norton
  - – Low output impedance / High input impedance
  - – Digital computer

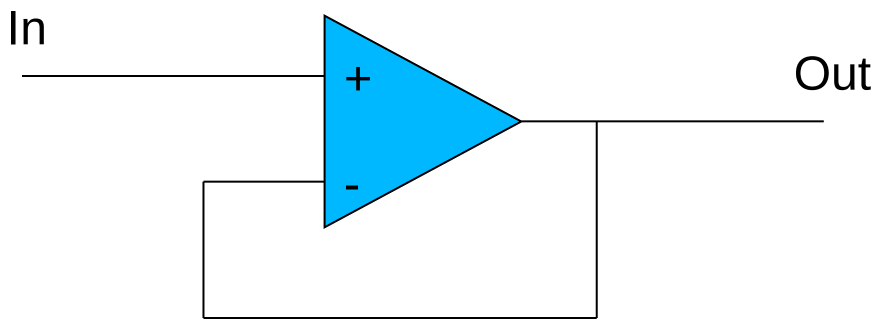
Abstraction: free-body diagram



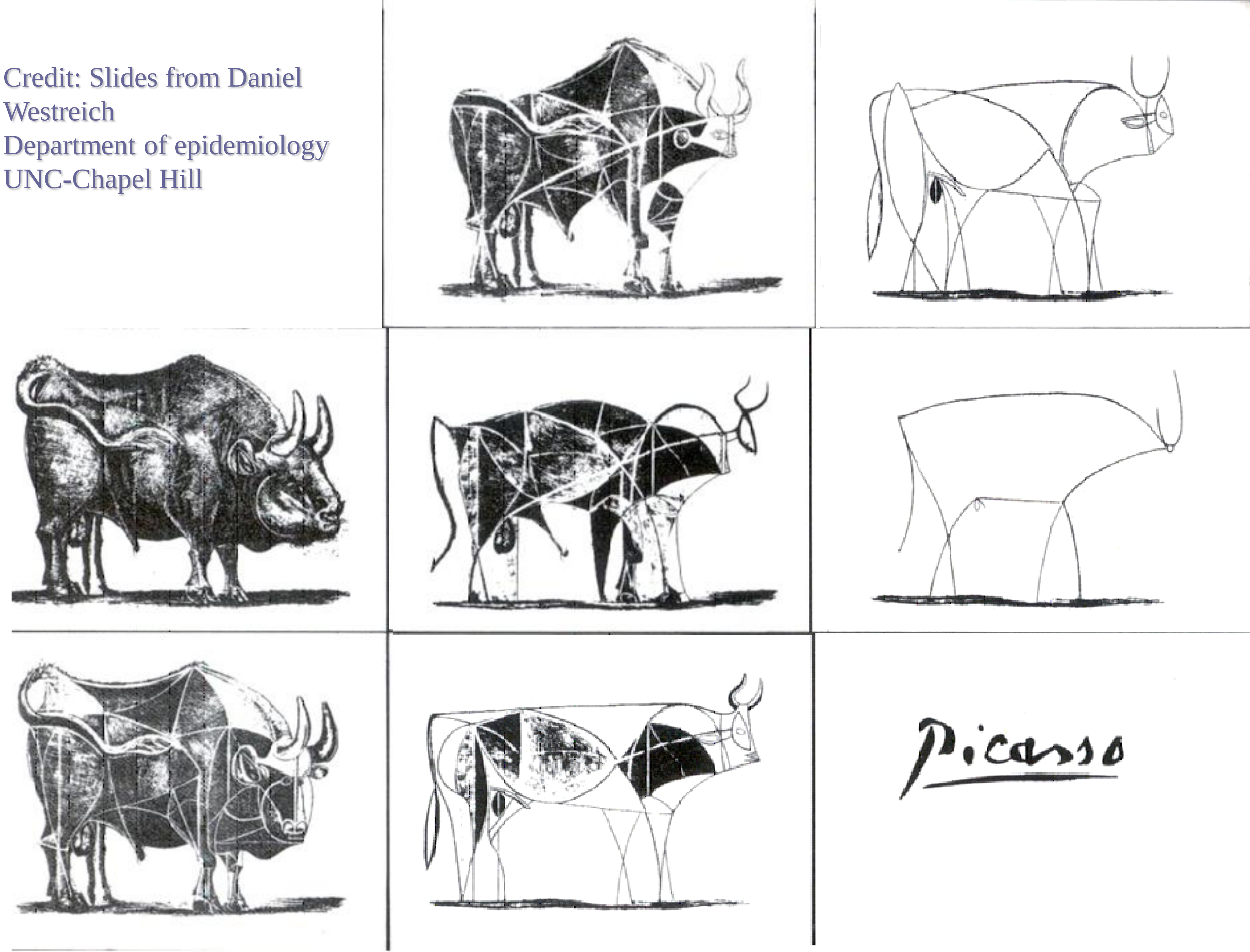


Modularity: Op-amp buffer

- Unity gain buffer
- Vout = Vin
- Very high input impedance, very low output impedance



Credit: Slides from Daniel Westreich
Department of epidemiology
UNC-Chapel Hill



What is software?

- Computer programs and associated documentation such as requirements, design models and user manuals.
- Software products may be developed for a particular customer or may be developed for a general market.
- Software products may be
  - Generic - developed to be sold to a range of different customers e.g. PC software such as Excel or Word.
  - Custom - developed for a single customer according to their specification.
- New software can be created by developing new programs, configuring generic software systems or reusing existing software.

What is Software Engineering?

The IEEE Computer Society defines software engineering as:

"the application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software, i.e. the application of engineering to software"

Source "IEEE Standard Glossary of Software Engineering Terminology," IEEE std 610.12-1990, 1990.

“A scientist builds in order to learn; an engineer learns in order to build."

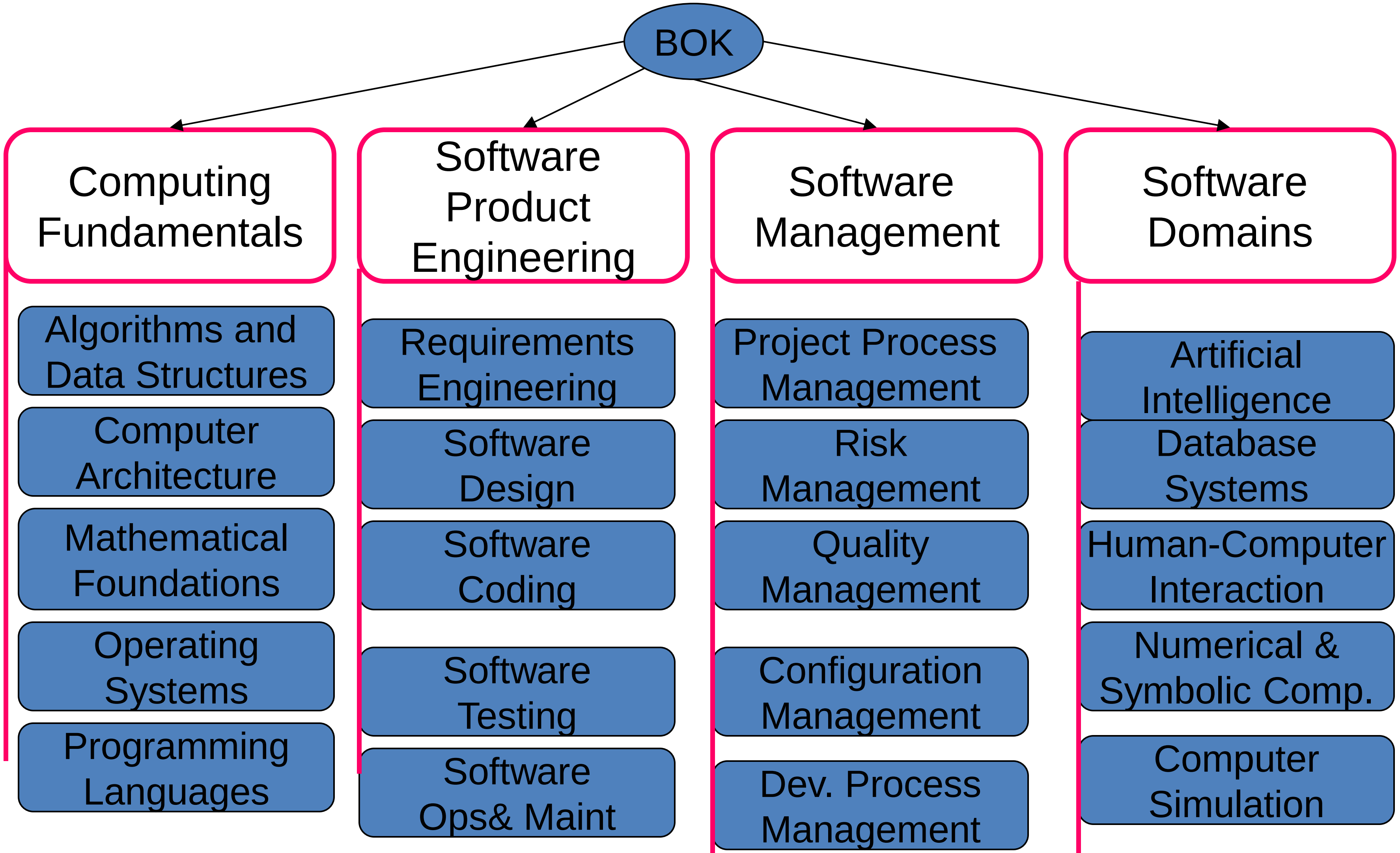
Fred Brooks

the quote appeared in the article " Software Engineering is Not Computer Science" by Steve McConnell
(http://www.gamasutra.com/features/19991216/mcconnell_pfv.htm)

What is software engineering?

- Software engineering is an engineering discipline that is concerned with all aspects of software production.
- Software engineers should adopt a systematic and organised approach to their work and use appropriate tools and techniques depending on the problem to be solved, the development constraints and the resources available.

Software Engineering Body of Knowledge



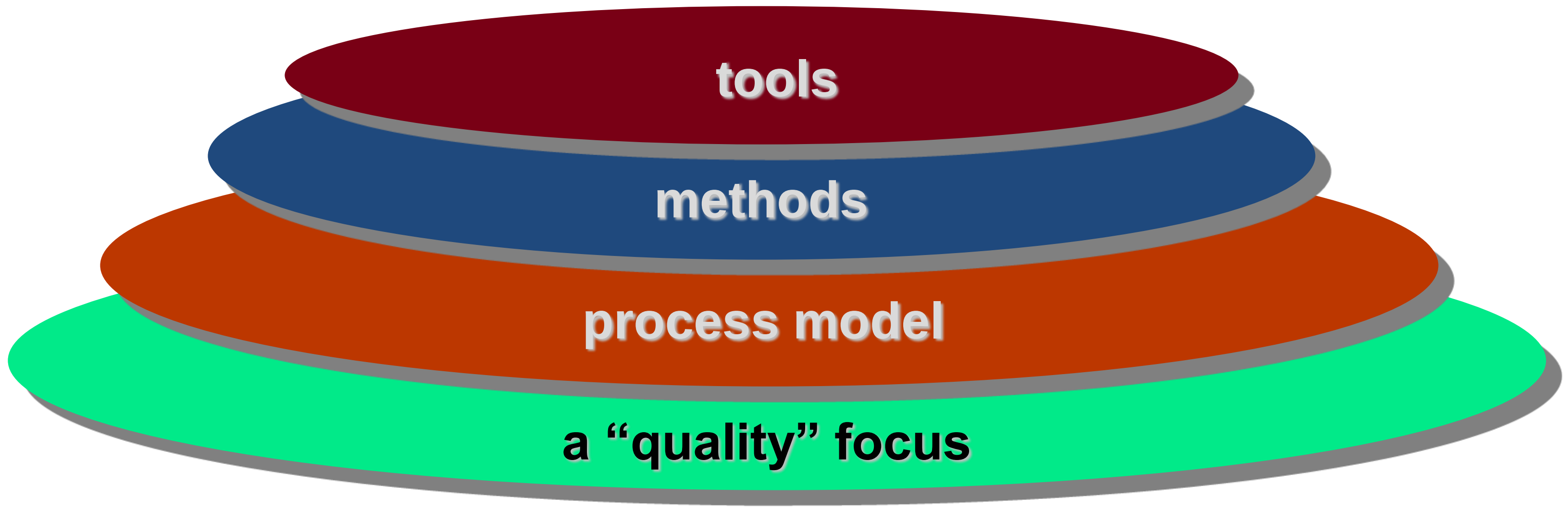
Source: http://www.sei.cmu.edu/pub/documents/99.reports/pdf/99tr004.pdf

Aspects of software engineering

- Analysis vs. synthesis of a problem
- Method or technique: procedure for producing a result
- Tool: instrument or automated system for accomplishing something
- Procedure: recipe for combination of tools and techniques
- Paradigm: style of doing something

Source http://www.cs.umd.edu/~mvz/cmsc435-s09/pdf/slides1.pdf

A Layered Technology



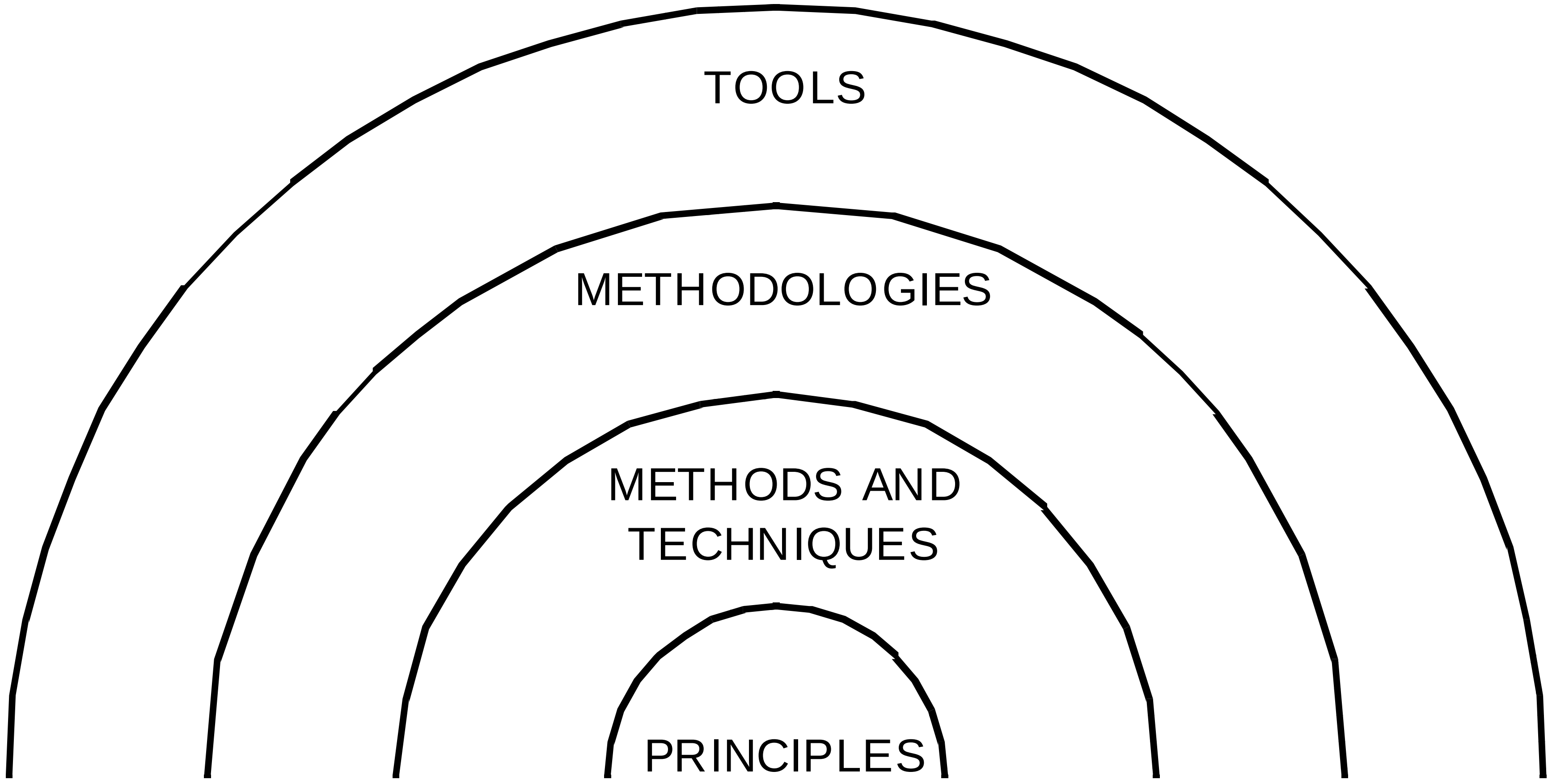
Software Engineering

These slides are designed to accompany *Software Engineering: A Practitioner’s Approach, 7/e* (McGraw-Hill 2009). Slides copyright 2009 by Roger Pressman.

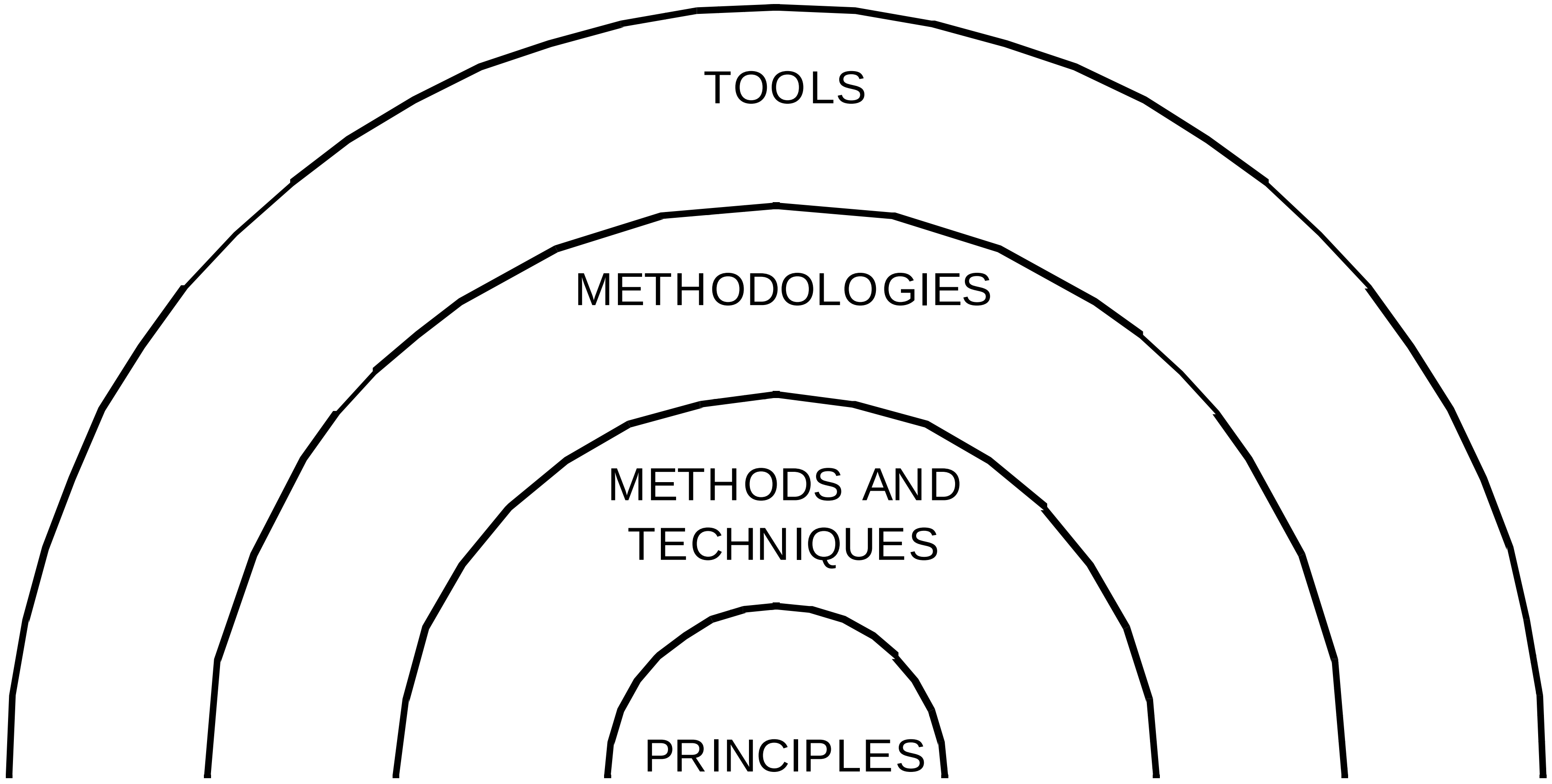
Software Engineering Principles

- Rigor and formality
- Separation of concerns
  - Modularity and decomposition
  - Abstraction
- Anticipation of change
- Generality
- Incrementality
- Scalability
- Compositionality
- Heterogeneity

From Principles to Tools



From Principles to Tools



Characteristics of Software

- SW is developed or engineered, it is not manufactured in classical sense
- Software does not wear out
- Most SW is custom built, rather than being assembled from existing components
- Typically errors are high when software is built or changed and the error rates comes down
- The cost of correction / change increases exponentially when we move ahead in the life cycle of a SW project

Objective of Software Engineering

- To organize and control software development process and produce a well-structured, accurate and useful software solution.

Software Engineering is

- A. Modeling
  - to manage details (complexity)
- B. Problem solving
  - using models to search for information and alternatives that can be completed with the available resources
- C. Knowledge acquisition
- D. Remembering rational
  - Rational preserves the context of the system, why decisions were made and options chosen earlier in development of the system

A. Modeling

- A model is an abstract representation of a system used to answer questions about that system
- The systems we make models of in software engineering are artificial systems
- Models can be made of (man made) artificial systems. These models can be used to determined if the system can be built

Modeling (con’t)

- Models can also be used to:
  - Assess the practicality and usefulness of the system
  - Estimate the cost and development time of the system
  - Develop a implementable approach to building the system with the desired functionality
  - And more …

B. Problem Solving

- The Software development method is based on the steps used to solve any other problem
  - Formulate the problem
  - Analyze the problem
  - Specify the problem clearly and in detail, capture the rational of the problem
  - Search for ways to solve the problem
  - Decide on the ‘best’, most appropriate solution
  - Specify the solution clearly and in detail, capture the rational of the solution
  - Implement the solution
  - Test/Verify the implementation
  - Deliver and maintain the resulting system

No Software Engineering?

- Why do some groups or companies, not use software engineering in their software development
- Attitude of some managers (not trained in software engineering) towards software development
  - Design and documentation is a waste of time
  - Production of lines of source code (# of LOC) or implementation of numbers of features as the sole measures of progress of project
  - End users should only be involved at the beginning and end of the software development process. OR
  - End users can add requests for additional functionality at any time

No Software Engineering?

- Poor project estimation, unreasonable deadlines
  - Causes programmers to cut corners, omit documentation, ‘patch’ or ‘hack’ rather than design and implement
  - Leads to maintenance nightmares

QUESTIONS?