Ain shams University
Department of Computer and systems Engineering
Postgraduate

Advanced Software Engineering
CSE608

Dr. Islam El-Maddah

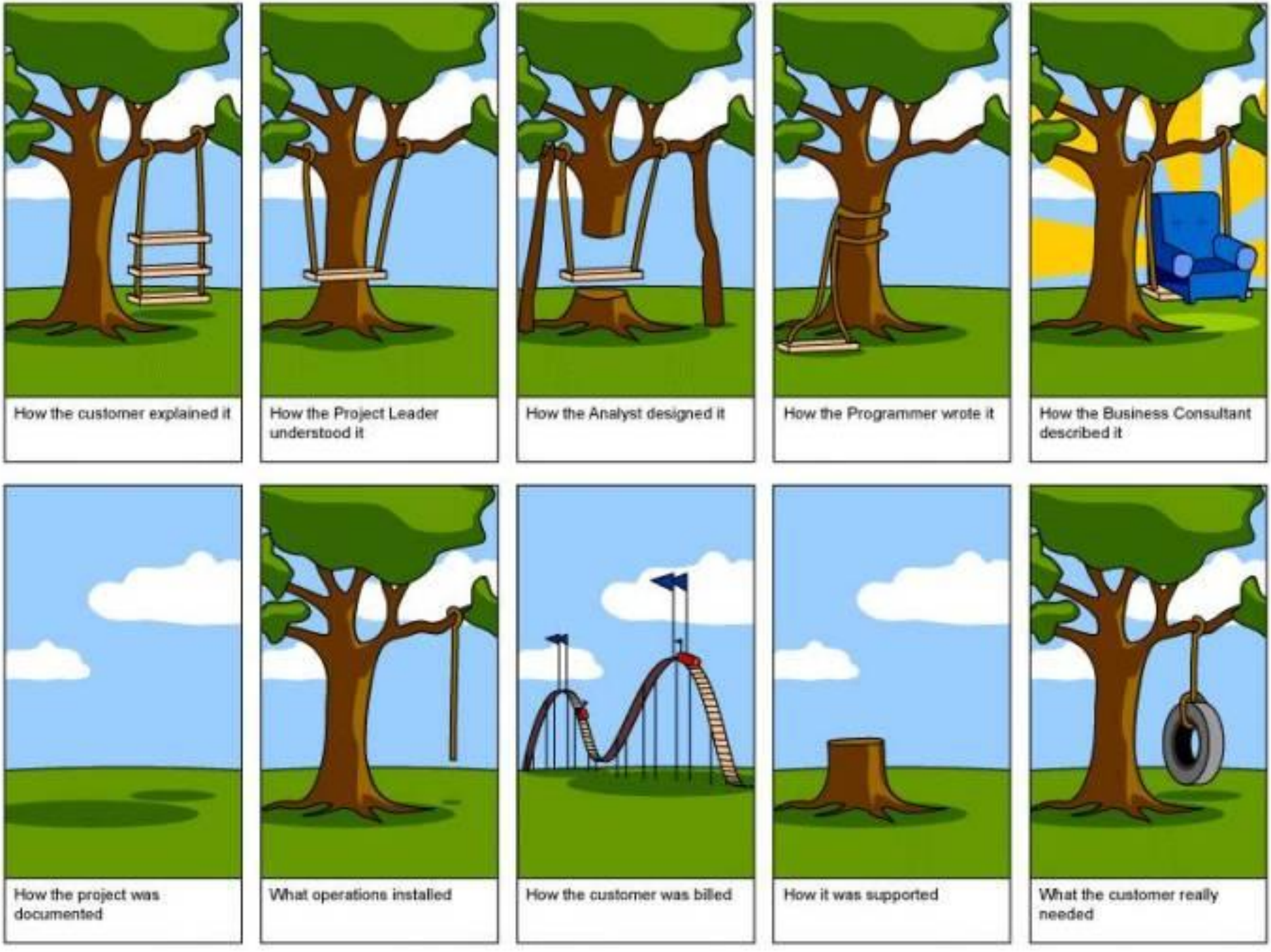
- Email address: islam_elmaddah@yahoo.co.uk

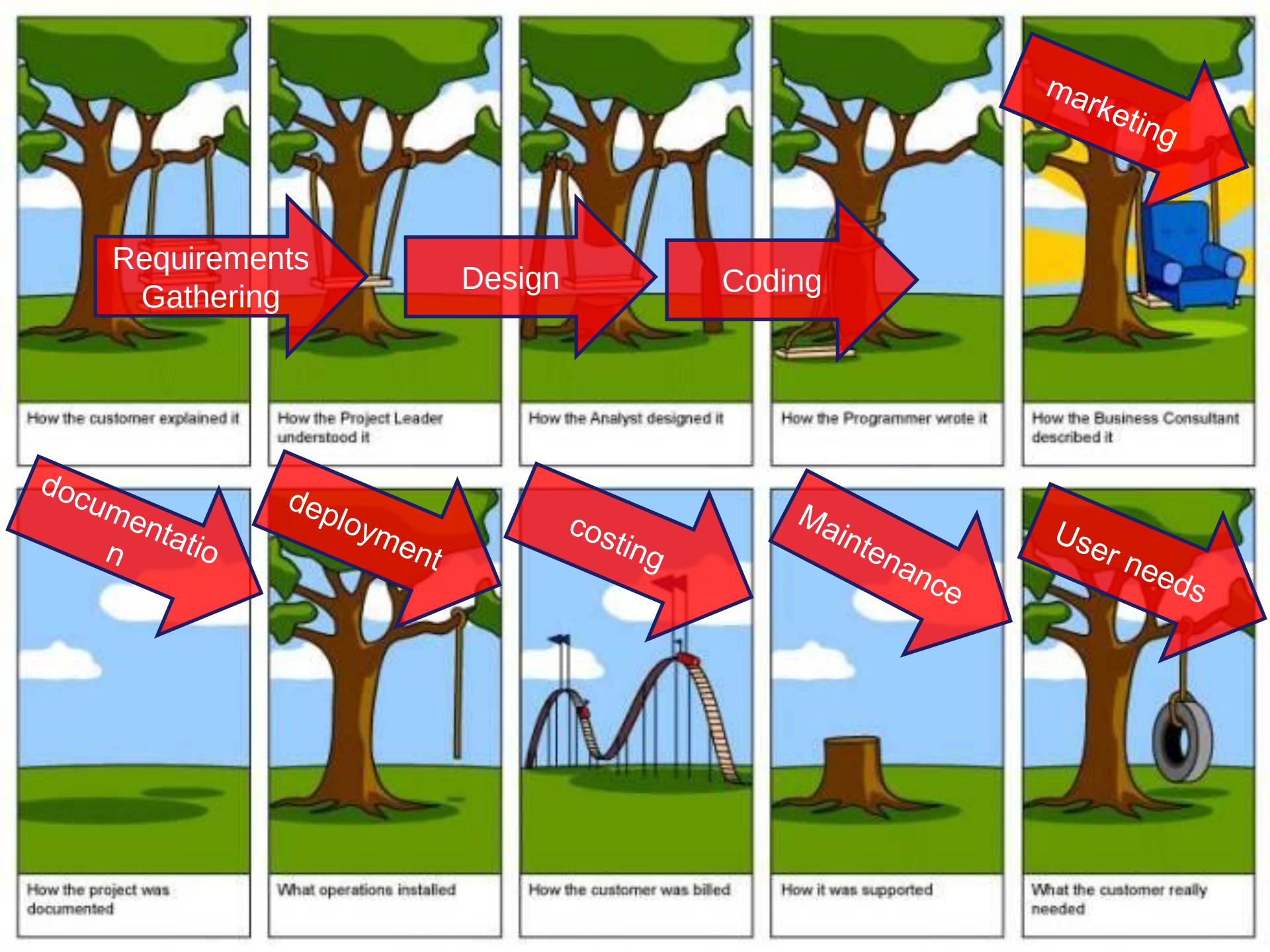
Text Books:

Software Engineering, A practitioner's approach

Roger s. Pressman 6th edition McGraw-Hill

Software Engineering Somerville 7th edition


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed


Requirements Gathering
Design
Coding
marketing
How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
documentation
deployment
costing
Maintenance
User needs
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Course contents

- Fundamental concepts of software engineering.
- Software Requirements: functional requirements, non-functional requirements.
- Software Requirements Specification (SRS) document.
- Software process models: waterfall model, spiral model, extreme programming model, and evolutionary model. Introduction to software design.
- Software Modeling: UML modeling. Class modeling. State diagram
- Software Testing
- Software Quality Assurance

Introduction to software Engineering

The Evolving role of software

- Dual role of Software
  - ★A Product
    - Information transformer-
      producing, managing and displaying
  - ★A Vehicle for delivering a product
    - Control of computer(operating system),the communication of information(networks) and the creation of other programs

Software has

Instructions

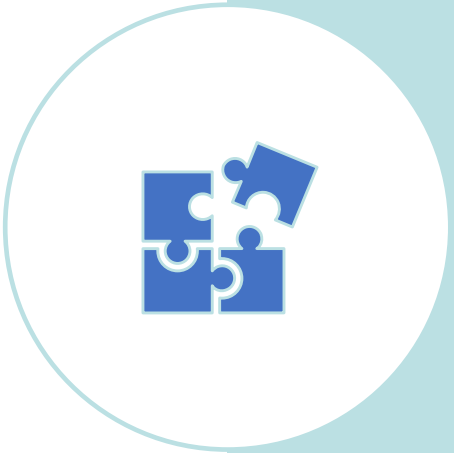
Data structures

Documents

Software Engineering

Engineering

-Application of science, tools and methods to find cost effective solution to problems



SOFTWARE ENGINEERING

- SE is defined as systematic, disciplined and quantifiable approach for the development, operation and maintenance of software

Characteristics of software

- **Software is developed or engineered**, it is not manufactured in the classical sense.
- **Software does not wear out. However** it deteriorates due to change.
- **Software is custom built** rather than assembling existing components.

  -Although the industry is moving towards component based construction, most software continues to be custom built

CHARACTERISTICS OF HARDWARE

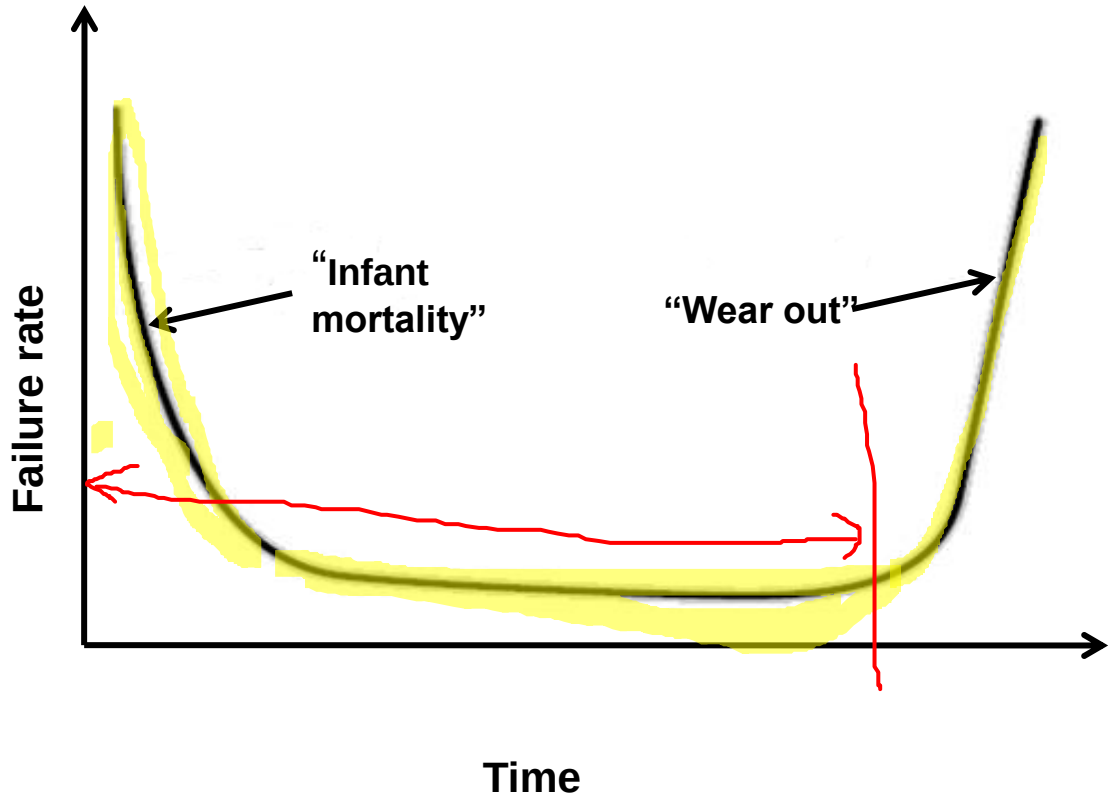


Fig: FAILURE CURVE FOR HARDWARE

CHARACTERISTICS OF SOFTWARE

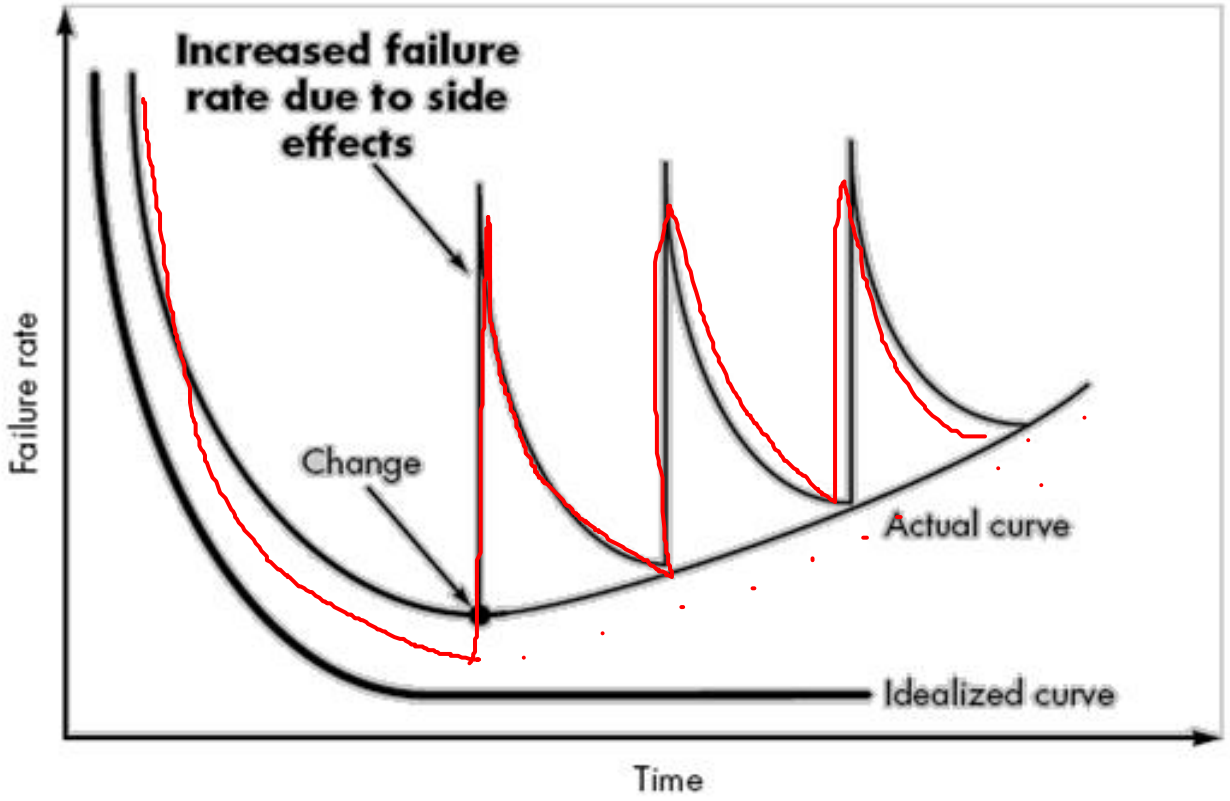
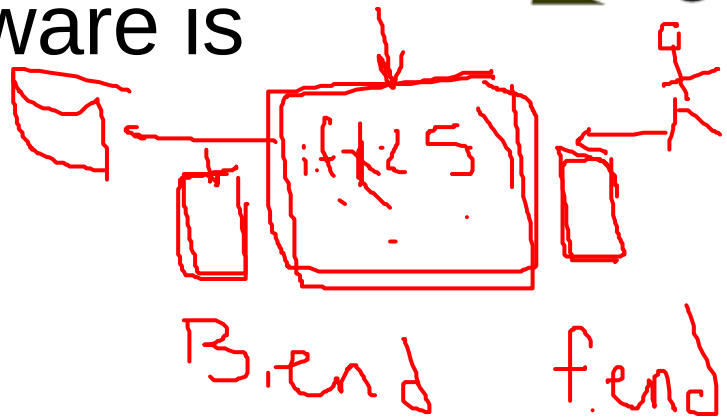


Fig: FAILURE CURVE FOR SOFTWARE

LEGACY SOFTWARE

- older programs that are developed decades ago.
- The Quality of legacy software is poor because it has
  - Inextensible design
  - convoluted code
  - poor and nonexistent documentation
  - Non existence test cases.

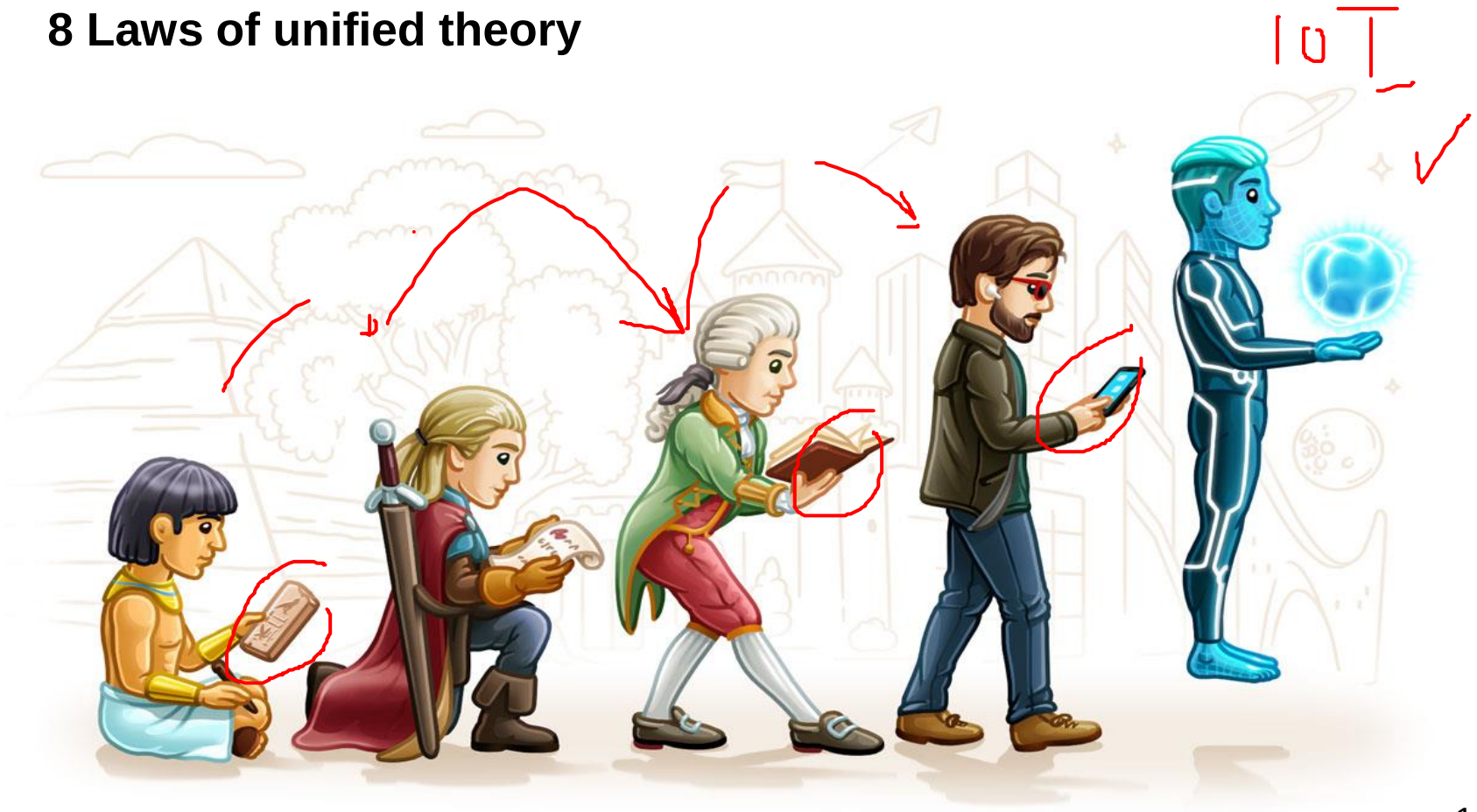


Legacy systems evolve due :

- The software must be adapted to meet the needs of new computing environment or technology.
- The software must be enhanced to implement new business requirements.
- The software must be extended to make it interoperable with more modern systems or database
- The software must be rearchitected to make it viable within a network environment.

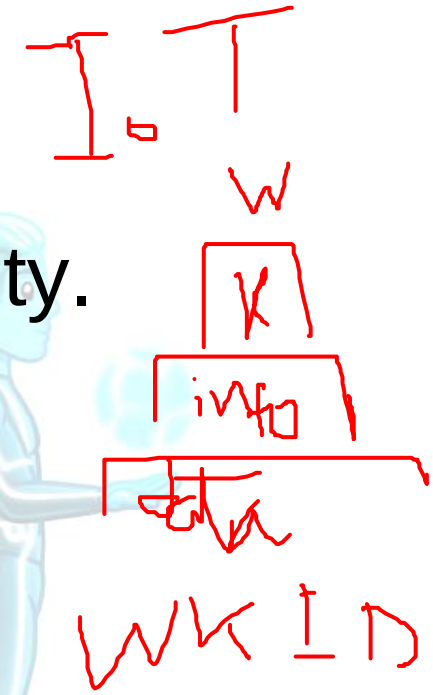
Software Evolution

- **Software evolves due to changes**
- **Changes occur due to correction, adaption and enhancement**
- **8 Laws of unified theory**



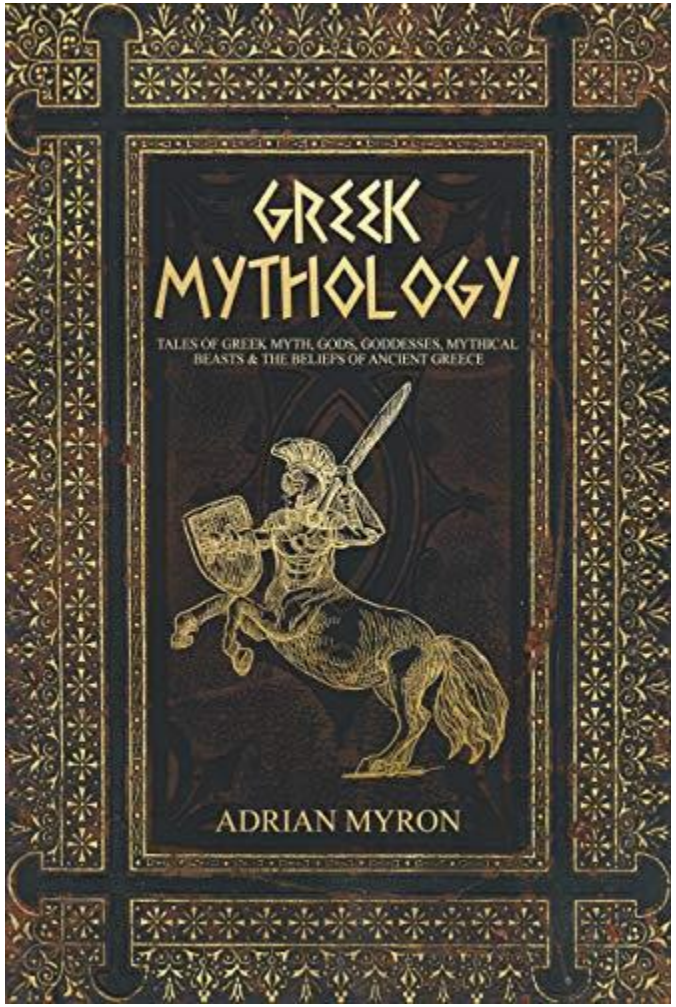
Software Evolution

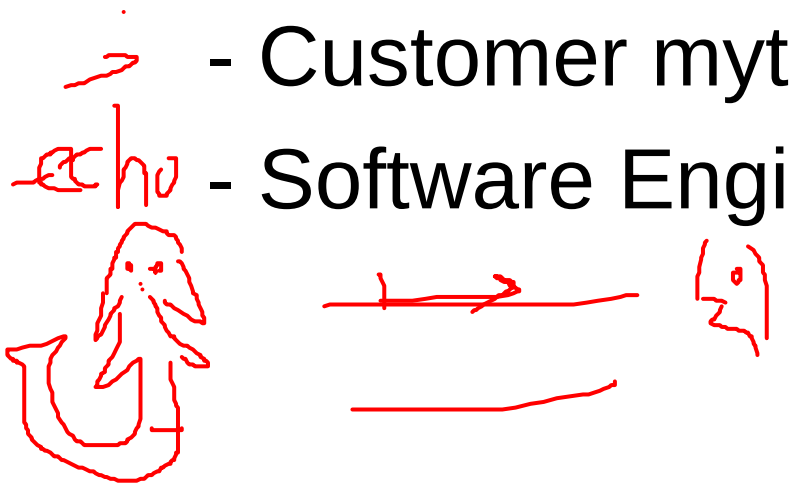
- The Law of Continuing Change.
- The Law of Increasing Complexity.
- The Law of Self-Regulation
- The Law of Conservation of Organizational Stability.
- The Law of Conservation of Familiarity
- The Law of Continuing Growth
- The Law of Declining Quality
- The Feedback System Law



SOFTWARE MYTHS

- Widely held but false view
- Propagate misinformation and confusion
- Three types of myths
  - Management myth
  - Customer myth
  - Software Engineer myth





MANAGEMENT MYTHS

- Myth(1)

  -The available standards and procedures for software are enough.

- Myth(2)

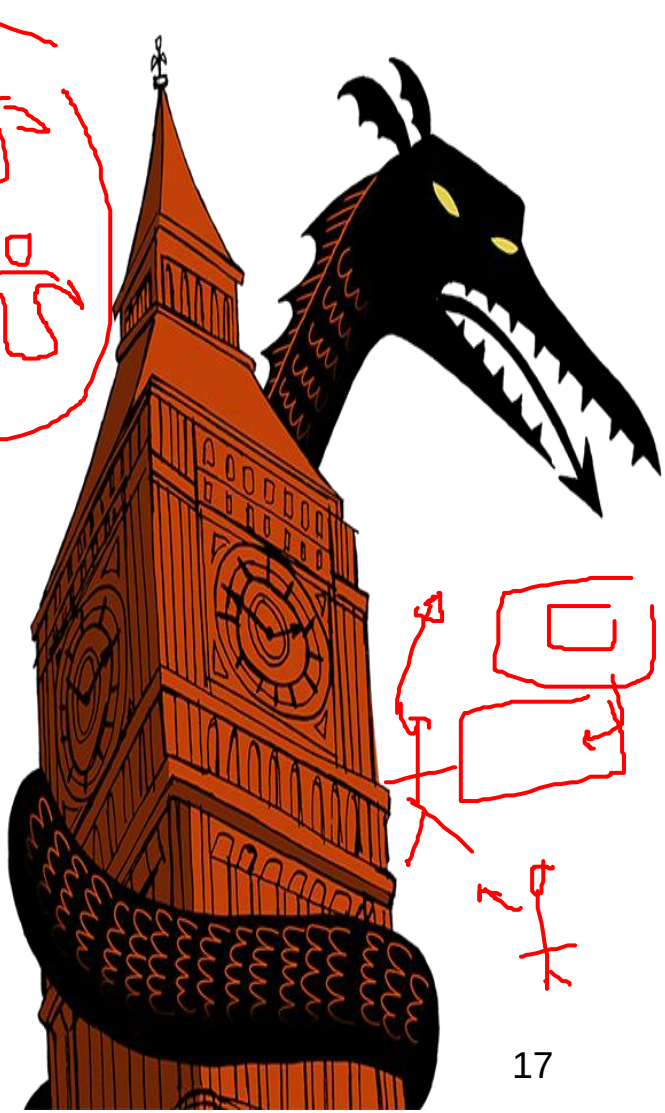
  -Each organization feel that they have state-of-art software development tools since they have latest computer.

- Myth(3)

  -Adding more programmers when the work is behind schedule can catch up.

- Myth(4)

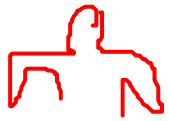
  -Outsourcing the software project to third party, we can relax and let that party build it.



CUSTOMER MYTH

- Myth(1)

  - General statement of objective is enough to begin writing programs, the details can be filled in later.

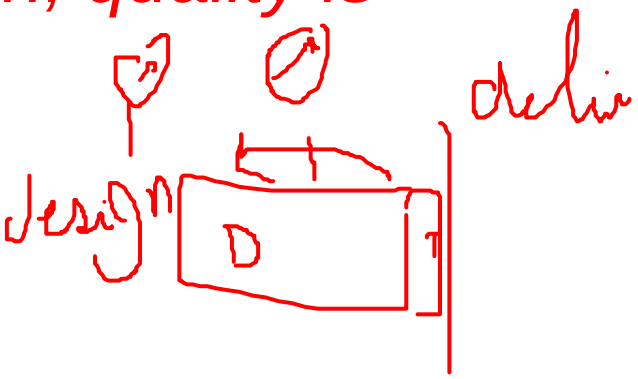


- Myth(2)

  -Software is easy to change because software is flexible

Software Engineers' Myths [Pressman]

- Myth(1):*Once the program is written, I'm done*
  - Between 60-80% of effort expended *after* delivery
- Myth(2):*Until the program is written, quality is uncertain*
  - Formal design reviews
  - Formal code reviews
  - Test-first approaches
  - Prototyping to validate requirements



- Myth(3):*The only deliverable is the program itself*
  - Lots of documentation: installation guides, usage guides, maintenance guides, API definitions and examples

Software Engineers' Myths [Pressman]

- *Myth(4): Documentation is Software-Engineering busy work*
  - Focus is on quality, not quantity
  - Documentation is hard for engineers to write.
  - Conserve energy: documented code can serve as a basis for useful documentation
    - JavaDoc
    - Doxygen

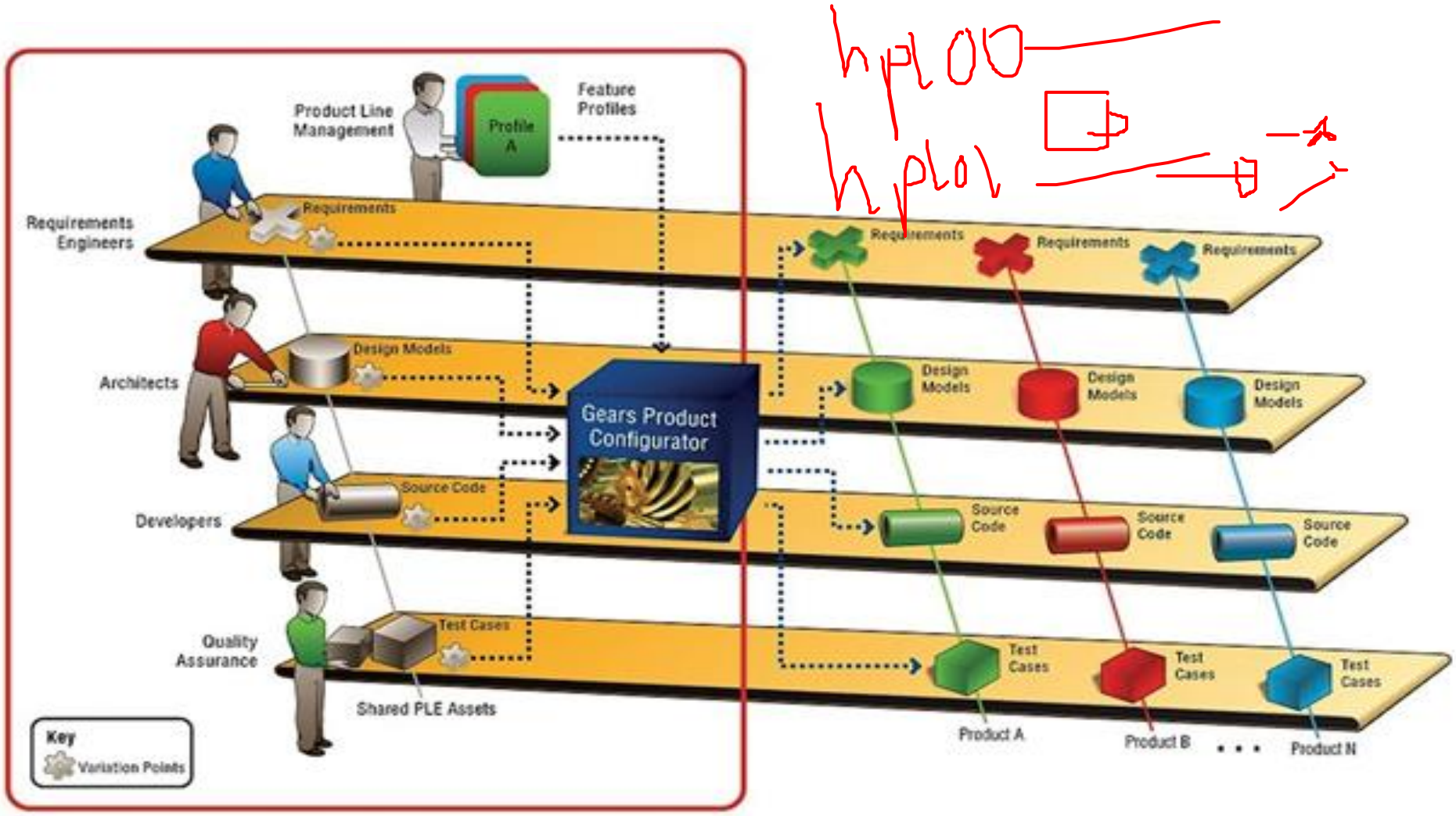
Architecture vs Interior Design

MINDLER

Software Architecture vs Design

Design	Architecture
The process of creating a specification of a software artifact that helps to implement the software	The process of creating high level structures of a software system
Creates a software artifacts describing all the units of the system to support coding	Converts the software characteristics into high level structure
Creational, structural and behavioral are some software design patterns	Microservice, serverless and event driven are some software architecture patterns
Helps to implement the software	Architecture helps to define the high level infrastructure of the software

Software Product Line



Product Line Engineering (PLE)

- could be the difference between success and failure when building most anything.
- The basic concept behind PLE is creating and managing the processes that allow a related set of products to share engineering assets and effort to achieve an efficient means of production.
- PLE makes creating products (services, software, etc.) easier and more cost effective.