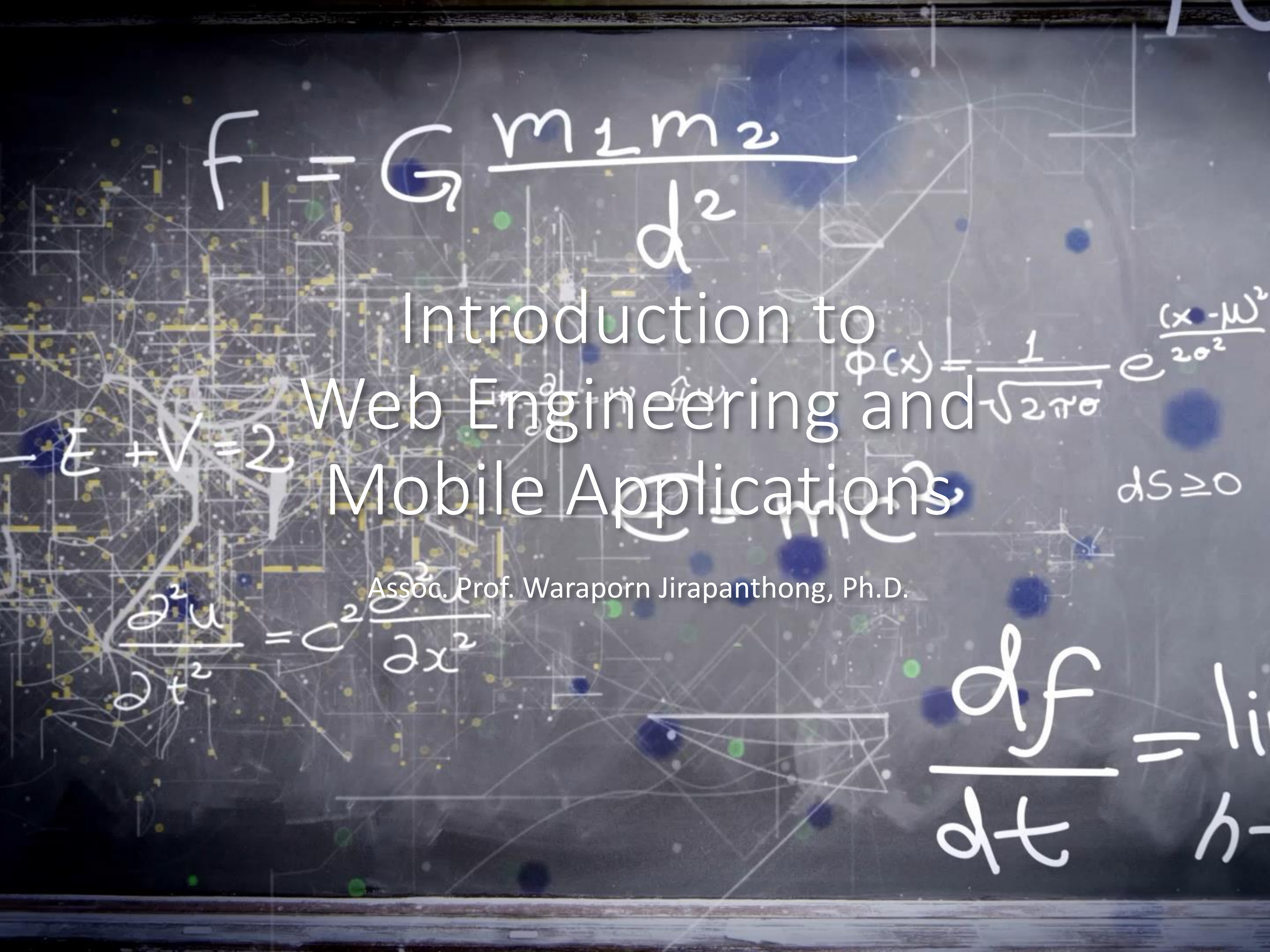

$f = G \frac{m_1 m_2}{d^2}$
$E + V = 2$
$\phi(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-\mu)^2}{2\sigma^2}}$
$dS \geq 0$
$e = mc^2$
$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}$
$\frac{df}{dt} = \lim_{h}$

Introduction to Web Engineering and Mobile Applications

Assoc. Prof. Waraporn Jirapanthong, Ph.D.

Topics

Nature of software

The evolution of software and its development process

Software development elements & process

Software engineering definition

Software engineering principles and profession

What is Software?

Software is:

- *instructions (computer programs) that when executed provide desired features, function, and performance;*
- *data structures that enable the programs to adequately manipulate information.*
- *documentation that describes the operation and use of the programs.*

- Software is developed or engineered, it is not manufactured in the classical sense.

The nature of software

Software is much intangible than other artifacts.

Duplicate pieces of software is trivial.

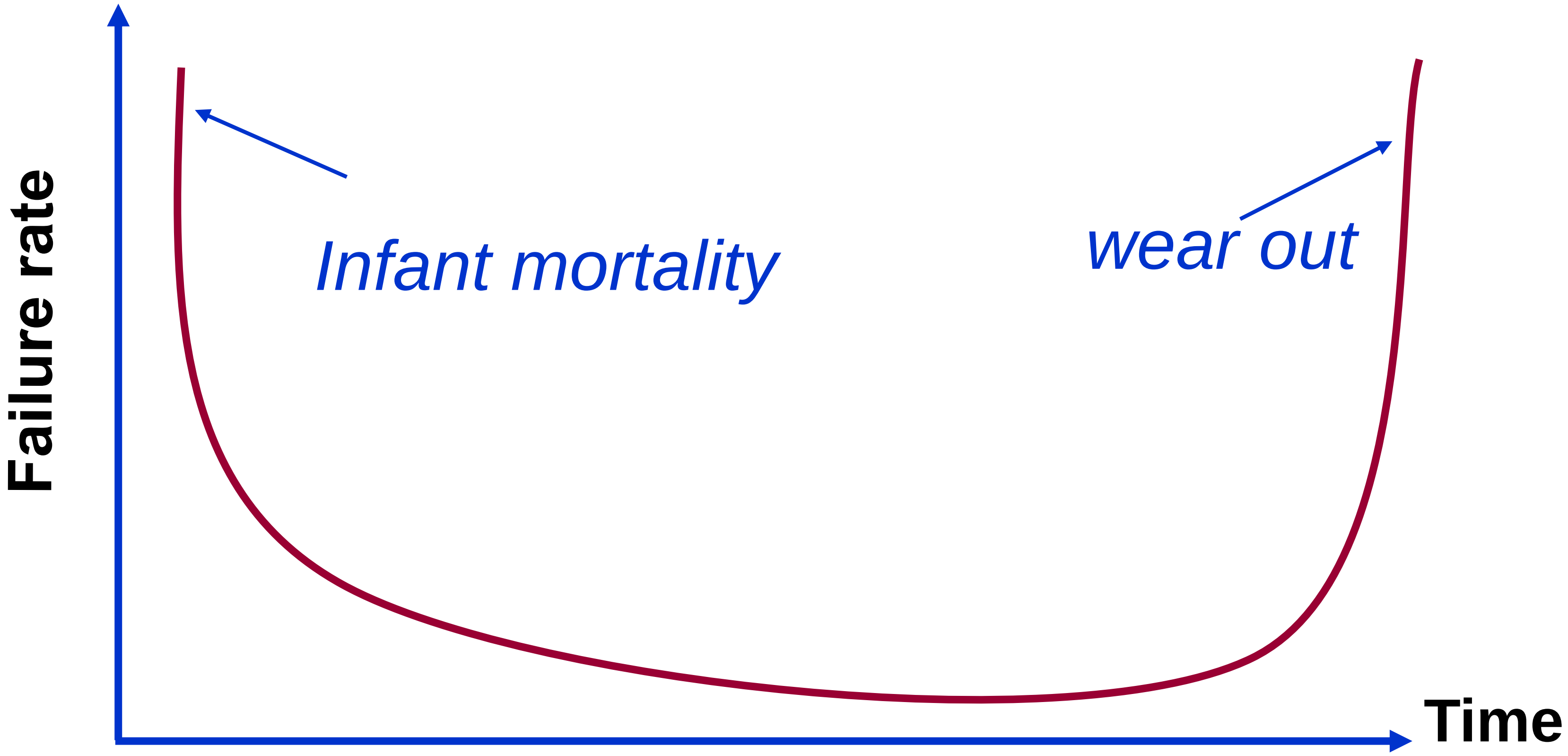
The software industry is labour intensive.

A novice programmer can create a complex code but not easy to detect and modify.

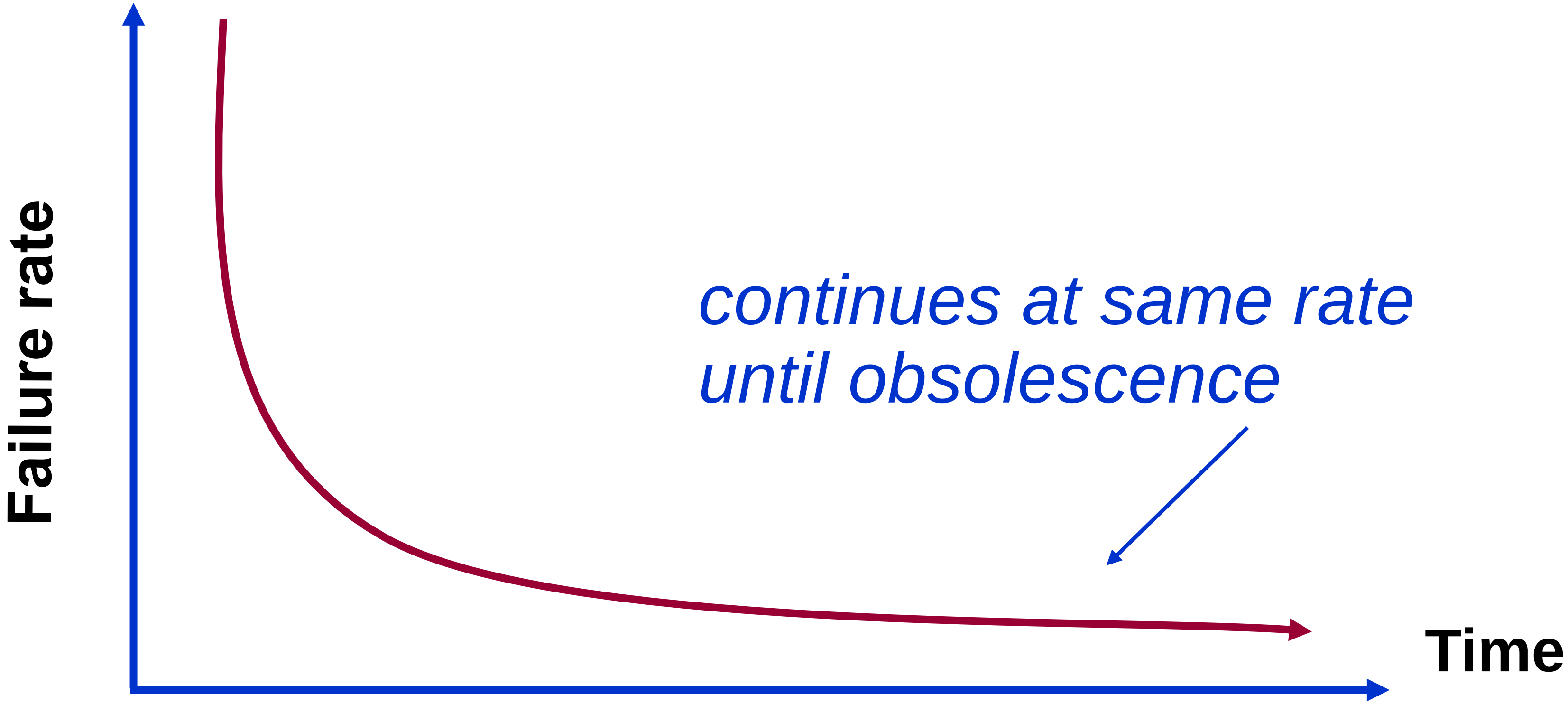
Difficult to make changes, however it will be.

Software does not wear out

Failure Curve for Hardware

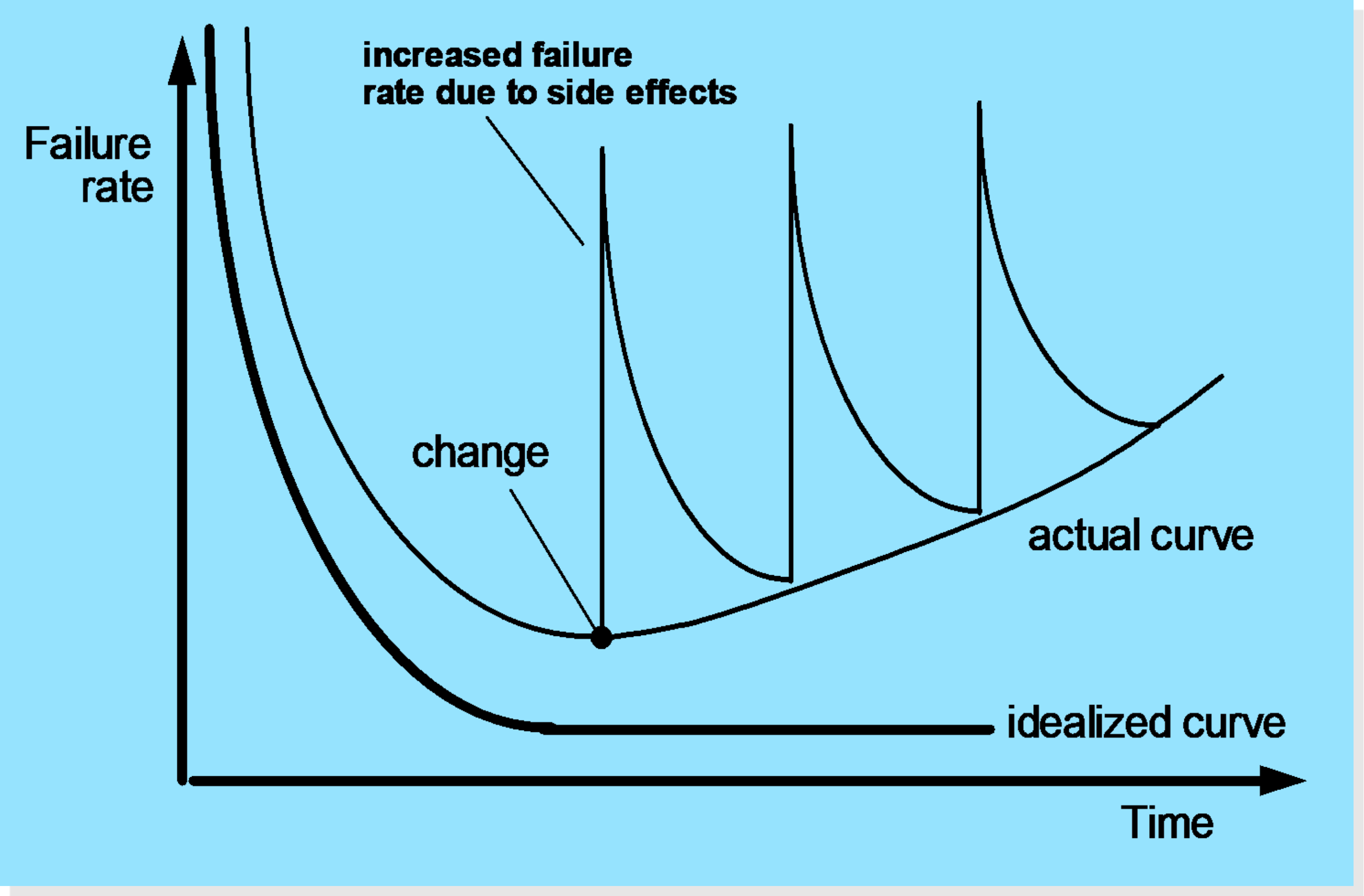


Failure Curve for Software



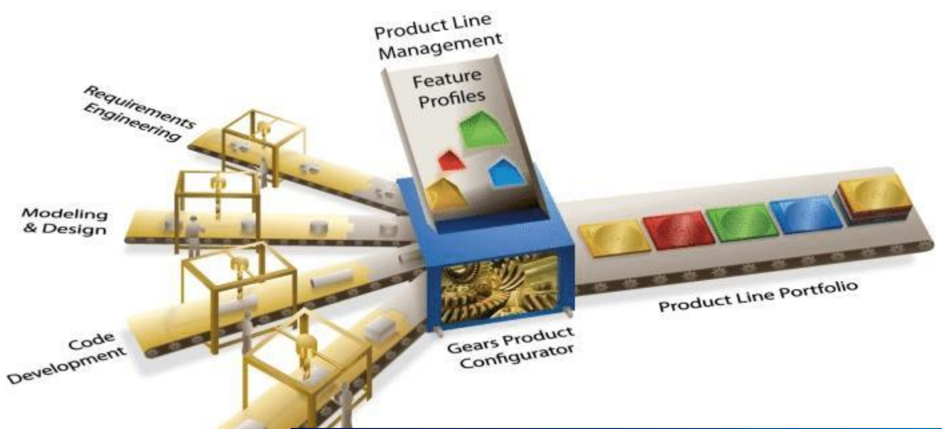
continues at same rate until obsolescence

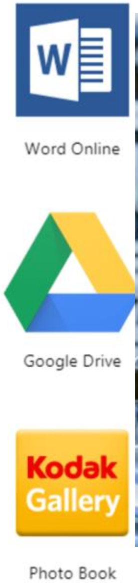
Wear vs. Deterioration

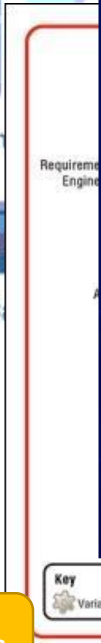


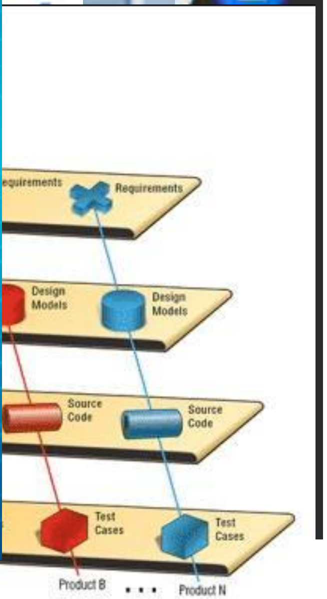
Software A

- System softwa
- Application sof
- Engineering/sc









COTS

Web Apps

What are COTS Applications?

- Commercial Off The Shelf Applications are:
  - Developed by a vendor
  - Sold, leased or licensed to business organizations
  - Typically serve enterprise-wide functions

Examples of COTS Application

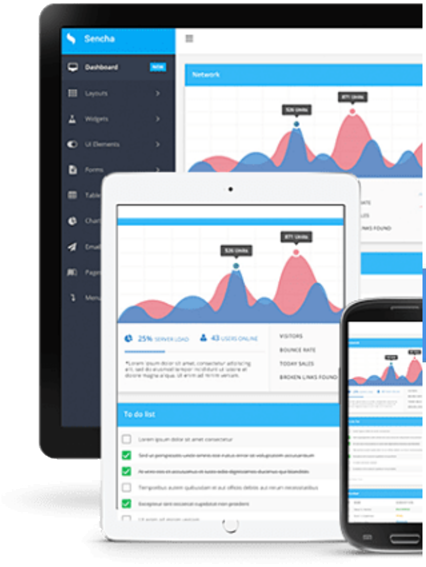
- Many are Enterprise Resource Planning (ERP) or Customer Relationship Management (CRM)
  - Workday
  - Workforce
  - SAP
  - Salesforce.com
  - Peoplesoft
  - Oracle Financials

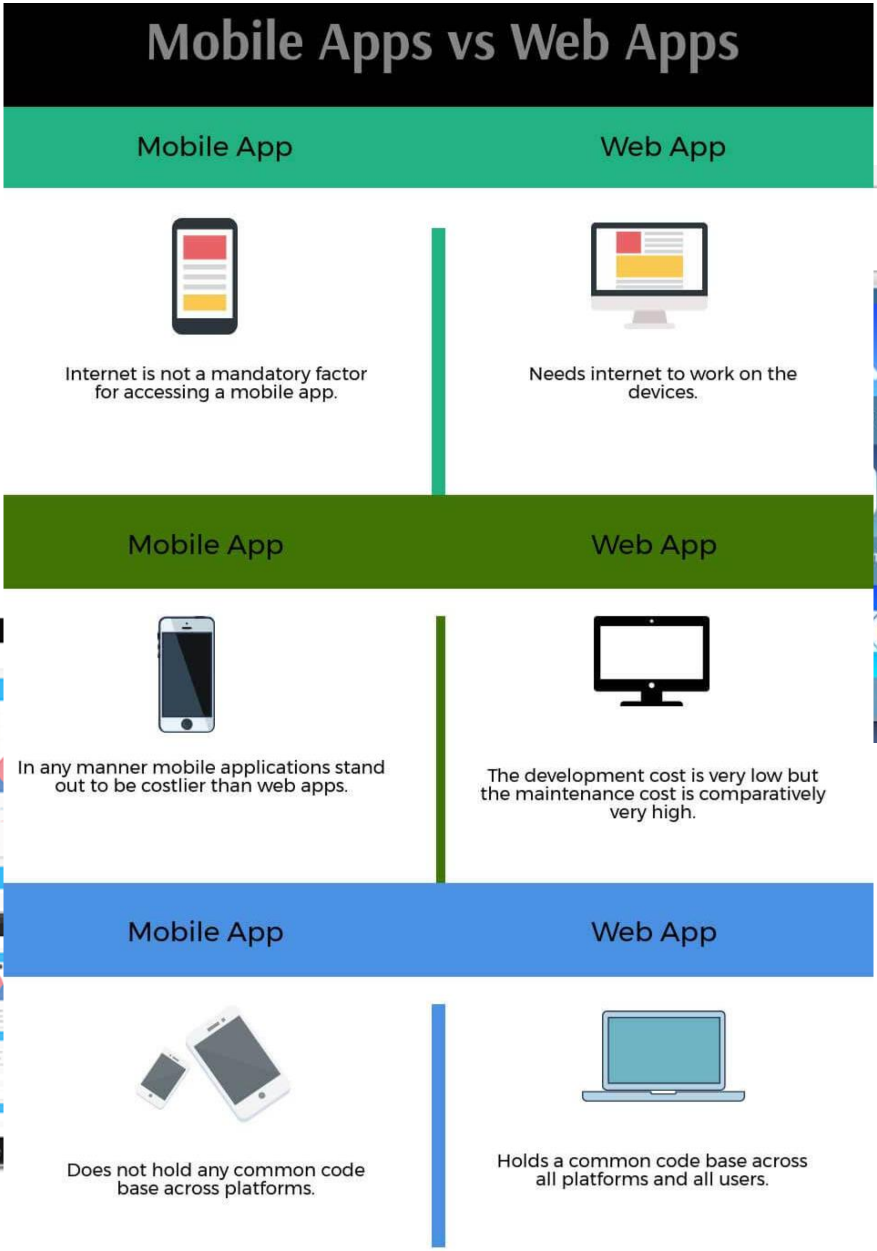
Examples of COTS Applications

- Some are smaller, niche products
  - Geospatial Information Systems (GIS)
    - SmallWorld
    - ArcGIS

Software Application

Web Apps



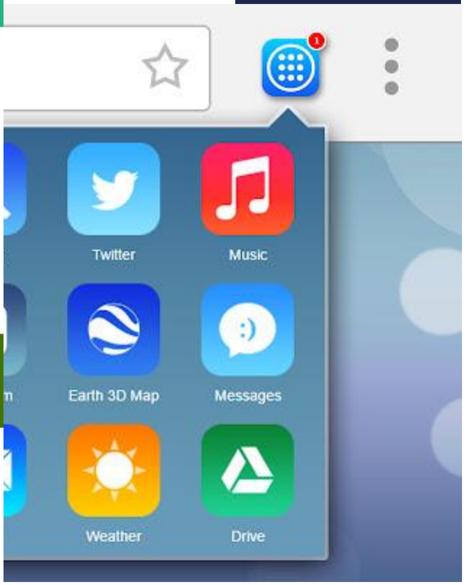


Mobile Apps vs Web Apps

Mobile App	Web App
Internet is not a mandatory factor for accessing a mobile app.	Needs internet to work on the devices.

Mobile App	Web App
In any manner mobile applications stand out to be costlier than web apps.	The development cost is very low but the maintenance cost is comparatively very high.

Mobile App	Web App
Does not hold any common code base across platforms.	Holds a common code base across all platforms and all users.



The Evolution of Software

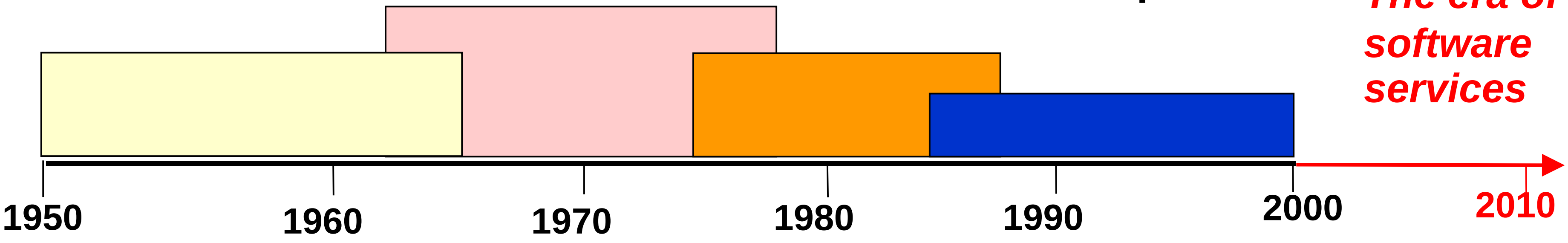
The early years
- **Batch orientation**
- **Limited distribution**
- **Customer software**

The 2nd era
- **Multi-user**
- **Real-time**
- **Database**
- **Product software**

The 3rd era
- **Distributed systems**
- **Embedded "intelligence**
- **Low cost hardware**
- **Consumer impact**

The 4th era
- **Powerful desk-top systems**
- **OO technologies**
- **Expert systems**
- **Artificial neural networks**
- **Parallel computing**
- **Network computers**

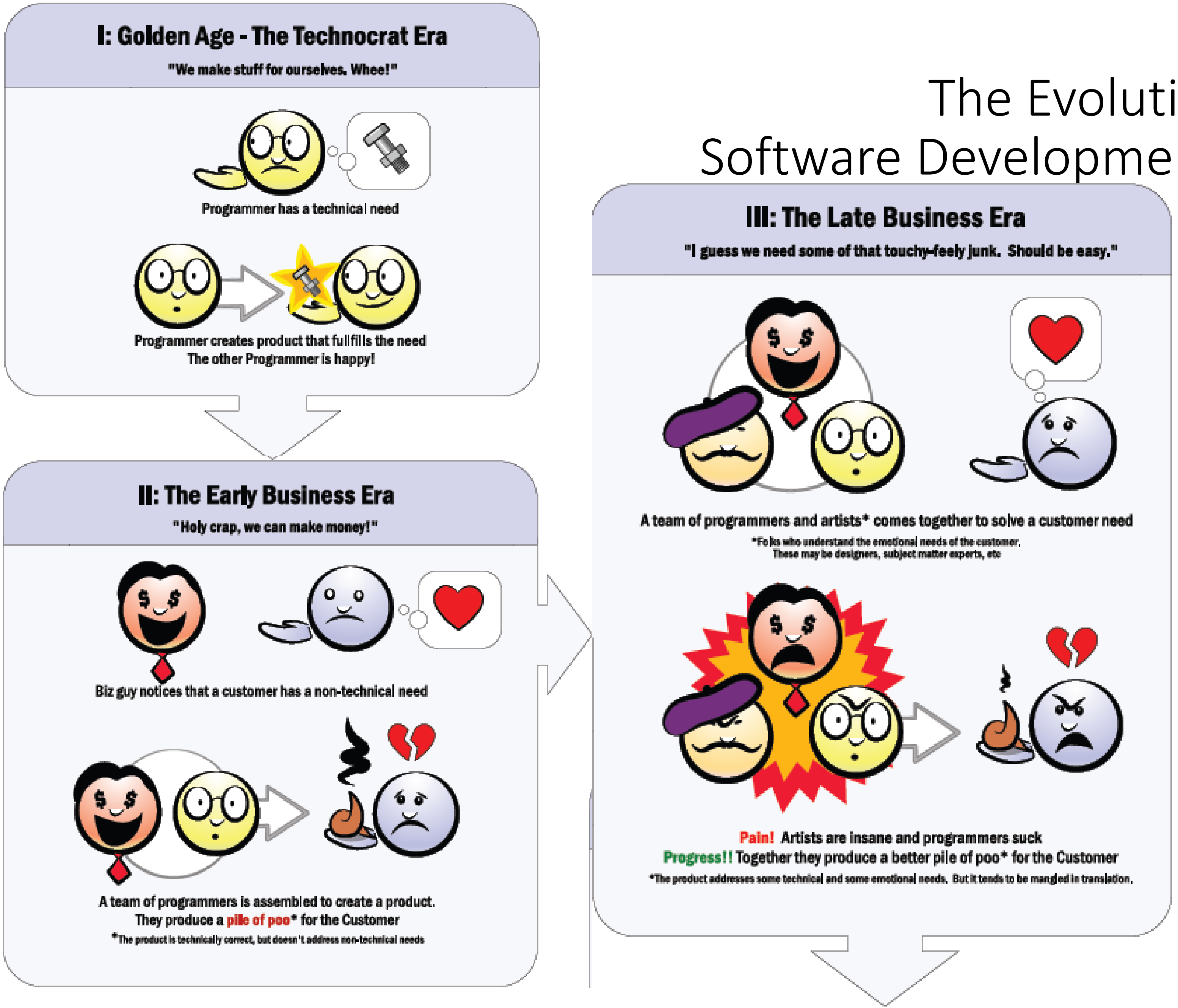
The era of software services



Software New Categories

- Open source - ”free” source code open to the computing community (a blessing, but also a potential curse!)
- Open world computing - pervasive, distributed computing
- Ubiquitous computing - wireless networks
- Netsourcing - the Web as a computing engine
- Software as a Service - a software distribution model in which applications are hosted by a vendor or service provider and made available to customers over a network, typically the Internet.
- Internet of Things (IoT) - the network of physical objects or "things" embedded with electronics, software, sensors, and connectivity to enable objects to exchange data with the manufacturer, operator and/or other connected devices based on the infrastructure of International Telecommunication Union's Global Standards Initiative. [Internet of Things Global Standards by ITU]

The Evolution of Software Development (1)



I: Golden Age - The Technocrat Era

"We make stuff for ourselves. Whee!"

Programmer has a technical need

Programmer creates product that fullfills the need
The other Programmer is happy!

II: The Early Business Era

"Holy crap, we can make money!"

Biz guy notices that a customer has a non-technical need

A team of programmers is assembled to create a product.
They produce a pile of poo* for the Customer

*The product is technically correct, but doesn't address non-technical needs

III: The Late Business Era

"I guess we need some of that touchy-feely junk. Should be easy."

A team of programmers and artists* comes together to solve a customer need

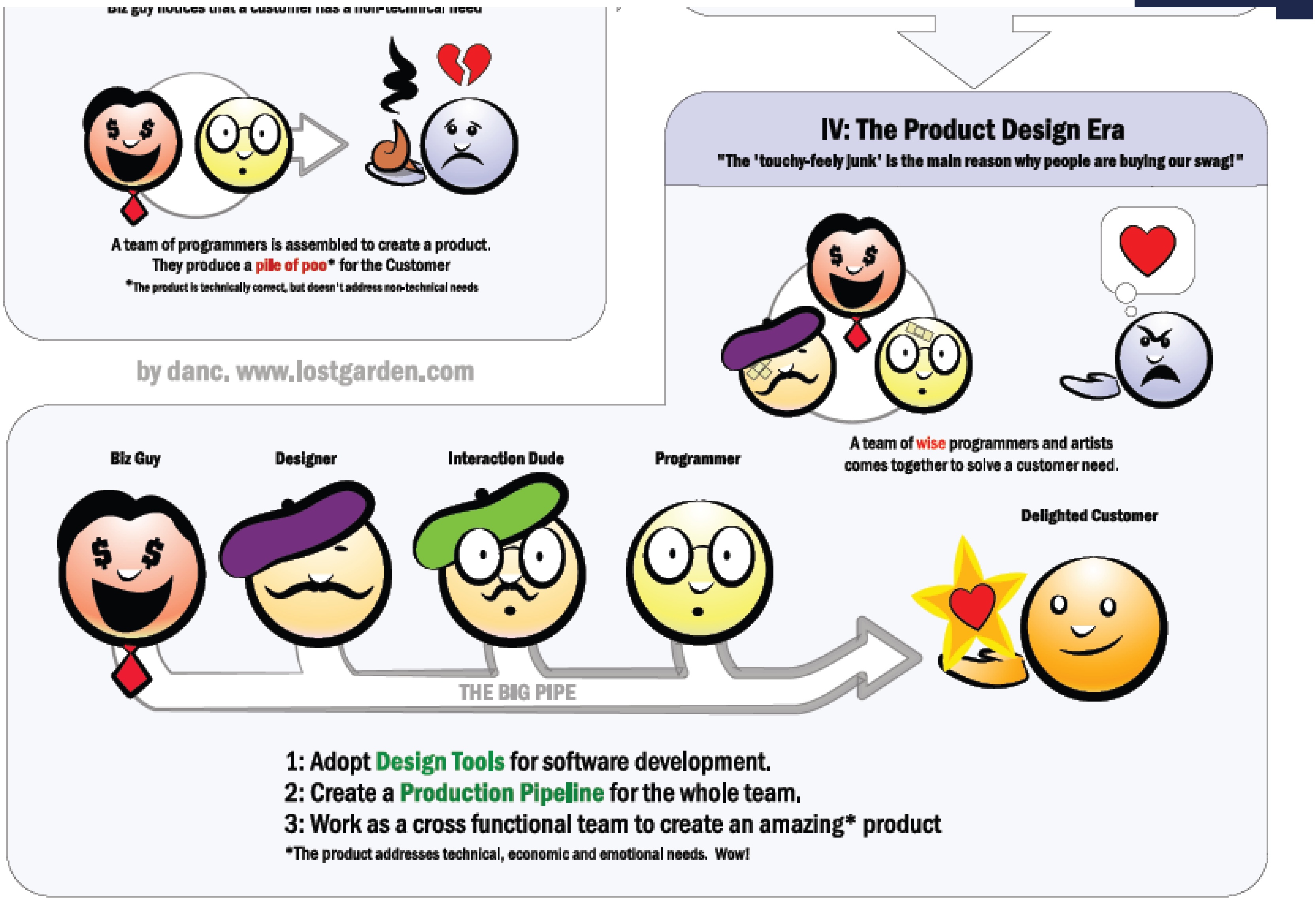
*Folks who understand the emotional needs of the customer.
These may be designers, subject matter experts, etc

Pain! Artists are insane and programmers suck
Progress!! Together they produce a better pile of poo* for the Customer

*The product addresses some technical and some emotional needs. But it tends to be mangled in translation.

*www.lostgarden.com/.../**Evolution**%20of%20**Software**%20Development.pdf*

The Evolution of Software Development (2)



Biz guy notices that a customer has a non-technical need

A team of programmers is assembled to create a product.
They produce a pile of poo* for the Customer
*The product is technically correct, but doesn't address non-technical needs

by danc. www.lostgarden.com

IV: The Product Design Era

"The 'touchy-feely junk' is the main reason why people are buying our swag!"

A team of wise programmers and artists comes together to solve a customer need.

1: Adopt Design Tools for software development.
2: Create a Production Pipeline for the whole team.
3: Work as a cross functional team to create an amazing* product
*The product addresses technical, economic and emotional needs. Wow!

www.lostgarden.com/.../***Evolution***%20of%20***Software***%20Development.pdf

Software Development Life Cycle

Software Development Life Cycle

Simple SDLC

1.
2.
3.
4.
5.
6.

Estimating

How much effort is required ?

Cost

Time / Scheduling

Requirements

Requirements – define and qualify system

- Defined by client, with help from engineer
- Functional – define what must be done
- Non-Functional – qualify the functional ones

Design constraints

- On design or implementation
- Programming language, platforms etc

Example: A Simple Problem

Given a collection of lines of text (strings) stored in a file, sort them in alphabetical order and write them to another file

Input format
- Character size
- Line separator

Specify Sorting
- Numbers
- Upper/lowercase

Special cases

Boundaries

Error Conditions

Performance

Real-time ?

Modifiability

User Interface
- GUI, CLI, Web …

Typical input and size

Platforms

Schedule

Programming Languages

Algorithms

Technical Issues : Systems Development

Problem and Solution Simplification

- Decomposition
- Modularization
- Separation
- Incremental iterations

Technology and tools choices

- Development platform
- Development language
- Database
- Network
- Configuration management

Process and Methodology

- Choice of process
- Choice of methodologies
- Choice tools to support the process

Non-Technical Issues: Systems Development

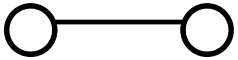
Project Effort Estimation and Scheduling

- Needs to consider and estimate more items
- Needs to coordinate more items in terms of pre-requisites and co-requisites
- Needs to consider more potentials of risks and variations

Assignments and Communications

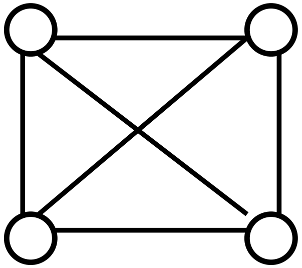
- More people with an increased variety of skills
- More communications among the people
- More errors and modifications

With the increase in system complexity, there is a corresponding increase in the “manpower” or human resources.



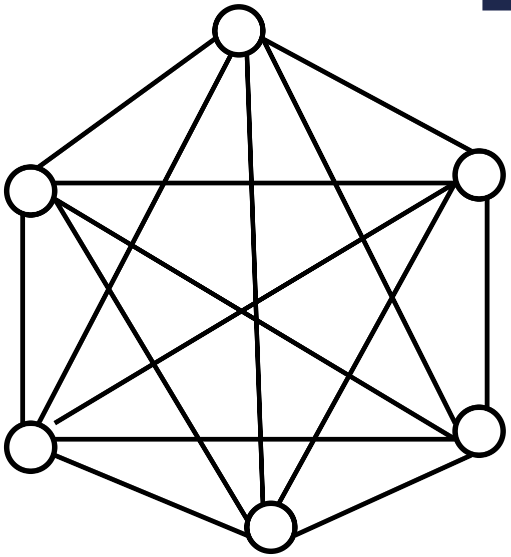
2 people:

1 path



4 people:

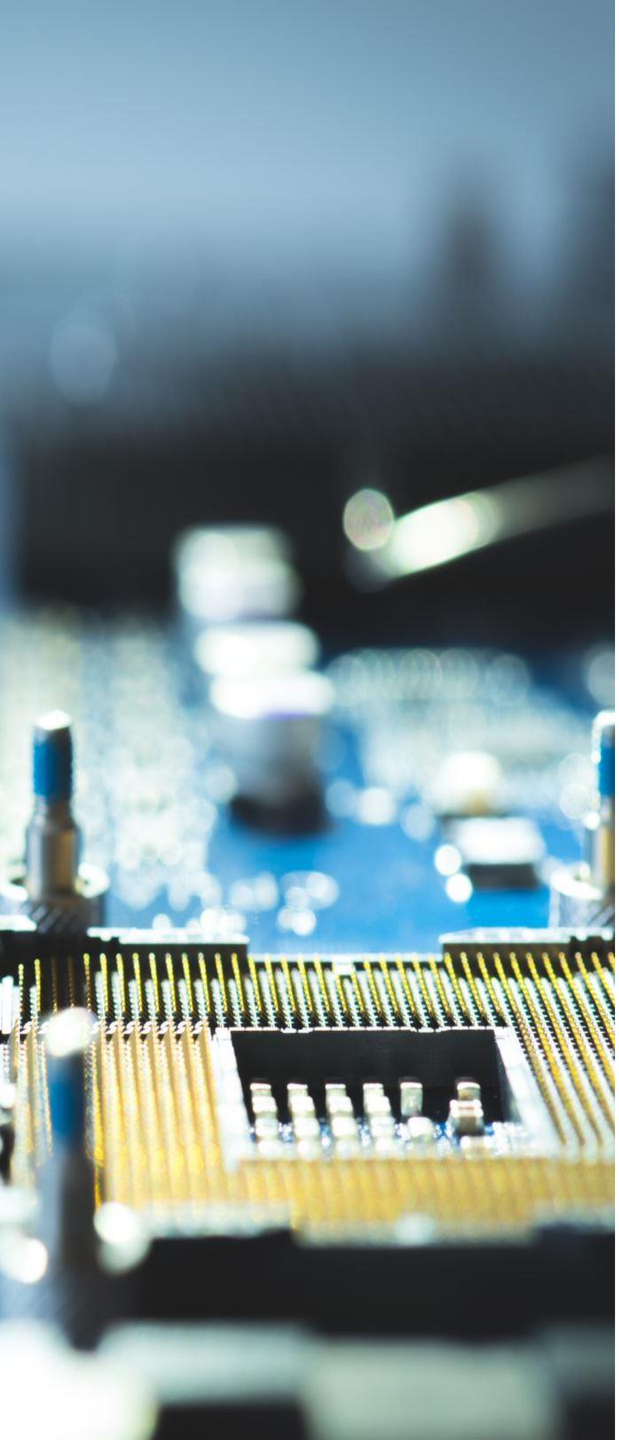
possibly 6 paths



6 people:

increase to potentially 15 paths

Increase in Amount of Communications as # of People Increases.
Also, an increase in the probability of error.



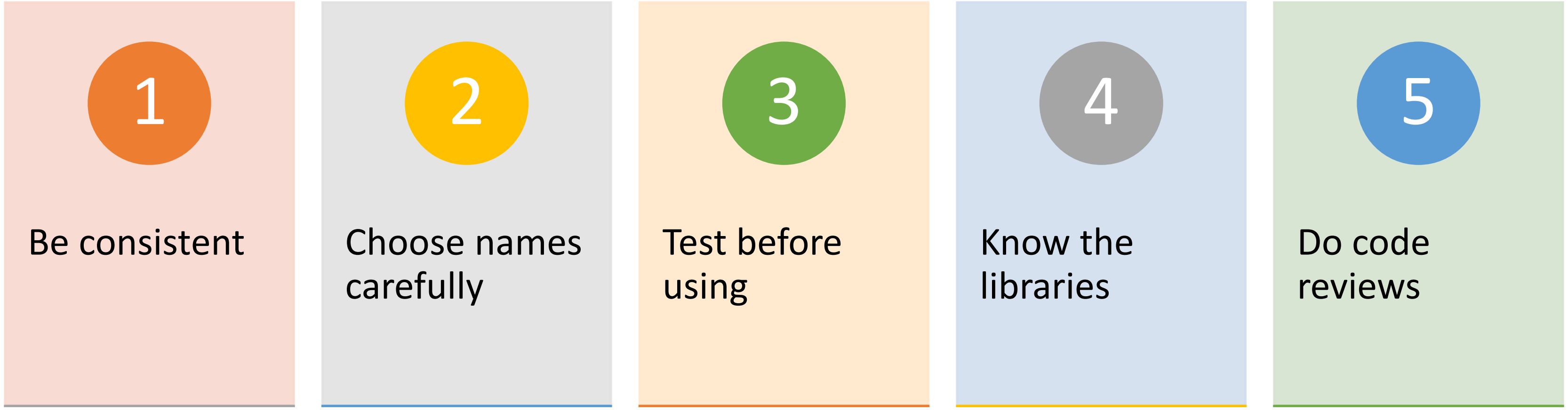
Designing and Developing a System

- What is a system?
  - Is a single program a system?
  - How many programs must be there?
  - Does it have to involve hardware, programs, business process, and others?

- Is there a difference between developing and supporting

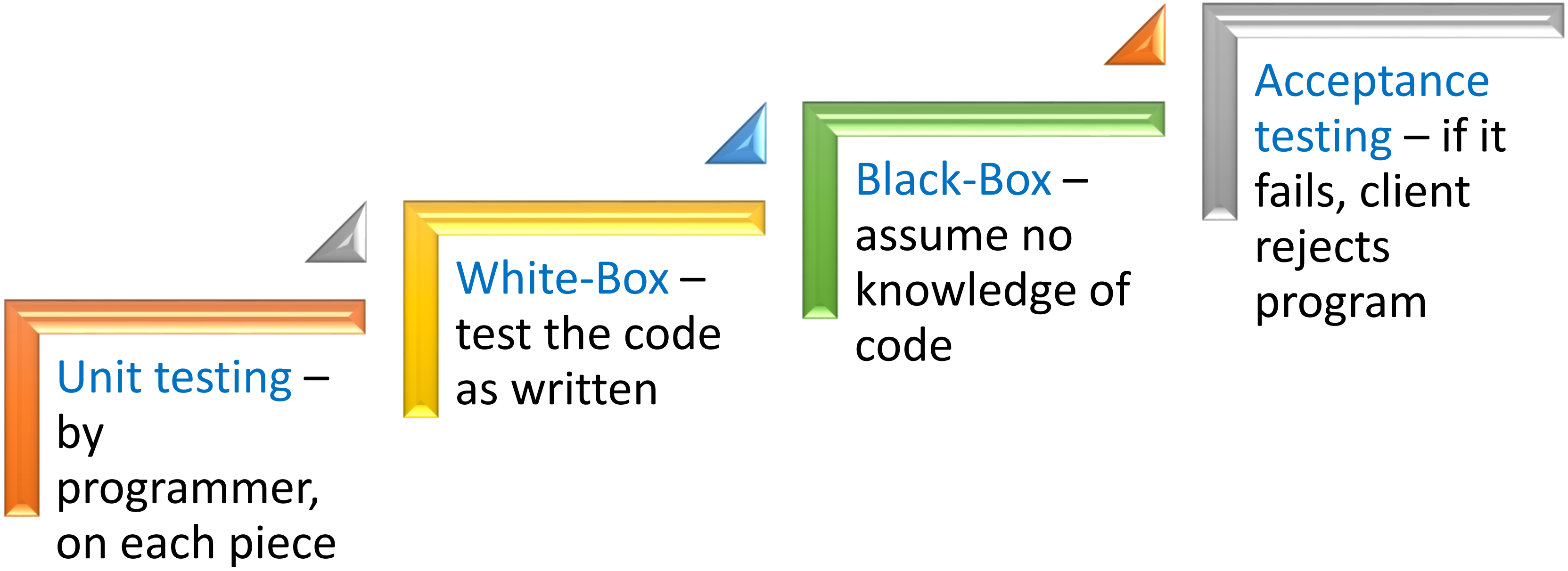
  a) a single program versus b) a system ?

Implementation Rules



1. Be consistent
2. Choose names carefully
3. Test before using
4. Know the libraries
5. Do code reviews

Testing



Unit testing – by programmer, on each piece

White-Box – test the code as written

Black-Box – assume no knowledge of code

Acceptance testing – if it fails, client rejects program

Supporting a System

Pre-release education and preparation

- Number of expected users
- Number of known problems and expected quality
- Amount of user and support personnel training
- Number of fix and maintenance cycle

Post-release user and customer support

- Call center and problem resolutions
- Major problem fixes and code changes
- Functional modifications and enhancements

Software Projects

Key success factors:

- User involvement
- Executive management support
- Clear requirement statements
- Proper planning

Top failure reasons:

- Lack of user input
- Incomplete requirements
- Changing requirements

Source of Software Product Problems

Code errors	:	38.33%
Design errors	:	24.17%
Documentation errors	:	13.33%
Requirements errors	:	12.50%
Bad-fix errors	:	11.67%

Should we worry about coding more or requirements more, why?

Software Engineering

What is needed to develop large and complex software products and what is needed to control such projects ?

More “discipline” is needed in this field:

“SOFTWARE ENGINEERING” (*NATO conference - 1968)*

What is Software Engineering

Sommerville –

an engineering discipline whose focus is the cost-effective development of high quality software system”

Pfleeger –

application of computing tools to solving problems

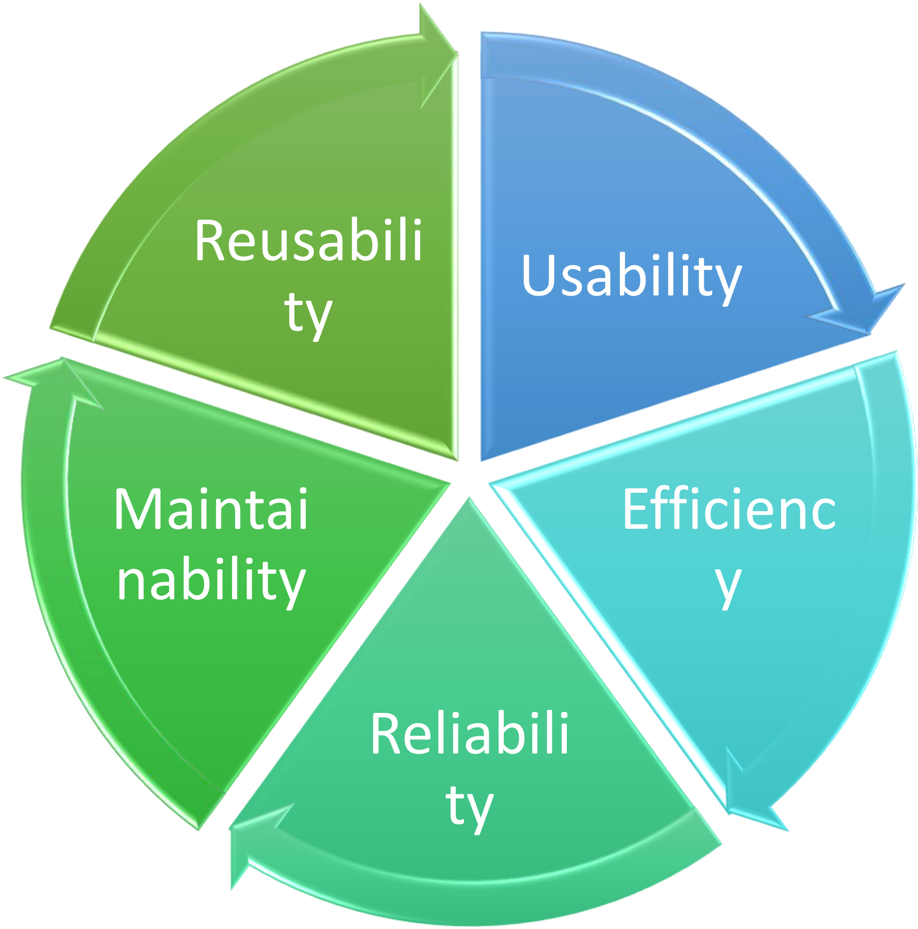
CMU/SEI-90-TR-003 –

form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problems

Timothy C. –

the process of *solving customers’ problems* by the *systematic development and evolution* of *large, high-quality software systems* within *cost, time and other constraints.*

Software Engineering: Software Quality



Reusabili ty

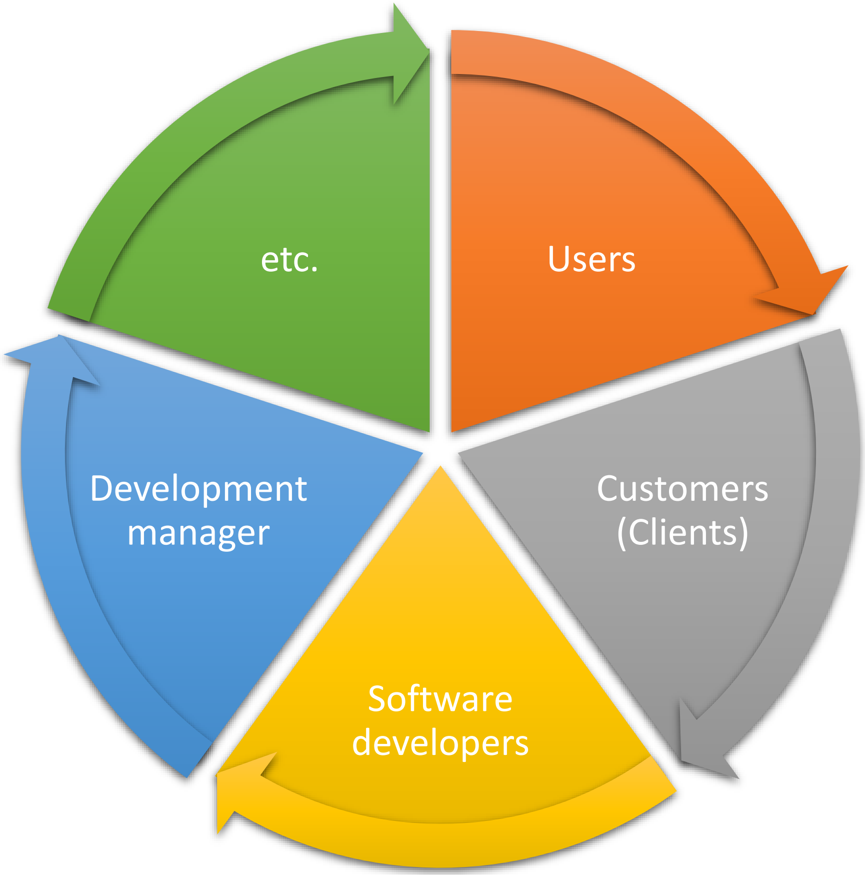
Usability

Efficienc y

Reliabili ty

Maintai nability

Stakeholders in Software Engineering



Software Engineering Profession

Software is a serious business

- Reached $180 billion in 2000
- It is ubiquitous across multiple industries

Software is a commodity of increasing “Value”

The business of software has graduated from a “garage” operation to an “enterprise” profession

We need to treat software engineering as an engineering profession

Classwork

- Identify a software project regarding web/mobile application development.