INTRODUCTION

Software Engineering

Dr. Raj Singh
THE NATURE OF SOFTWARE
WHAT IS SOFTWARE?

- **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

- Software doesn't "wear out"
WEAR VS. DETERIORATION

![Graph showing wear vs. deterioration over time]

- Failure rate
- Increased failure rate due to side effects
- Change
- Actual curve
- Idealized curve
- Time
SOFTWARE APPLICATIONS

- System software
- Application software
- Engineering/Scientific software
- Embedded software
- Product-line software
- Web/Mobile applications
- AI software (robotics, neural nets, game playing)
LEGACY SOFTWARE: WHY IT MUST CHANGE?

1. Software must be adapted to meet the needs of new computing environments or technology.

2. Software must be enhanced to implement new business requirements.

3. Software must be extended to make it interoperable with other more modern systems or databases.

4. Software must be re-architected to make it viable within a network environment.
Modern WebApps are much more than hypertext and images

Modern WebApps are augmented with tools, technologies, and interactive computing capability

WebApps may be standalone or integrated with corporate databases and business applications

Semantic web technologies (Web 3.0) have evolved into sophisticated corporate and consumer applications that require web linking, flexible data representation, and application programmer interfaces (API's) for access

The aesthetic nature of the content remains an important determinant of the quality of a WebApp.
CHARACTERISTICS OF WEBAPPS

Data Driven
The primary function of many WebApps is to use hypermedia to present text, graphics, audio, and video content to the end-user.

Content Sensitive
The quality and aesthetic nature of content remains an important determinant of the quality of a WebApp.

Continuous Evolution
Unlike conventional application software that evolves over a series of planned, chronologically-spaced releases, Web applications evolve continuously.

Immediacy
The compelling need to get software to market quickly is a characteristic of many application domains. WebApps often exhibit a time to market that can be a matter of a few days or weeks.

Security
Because WebApps are available via network access, it is difficult, if not impossible, to limit the population of end-users who may access the application.

Aesthetics
An undeniable part of the appeal of a WebApp is its look and feel.
MOBILE APPS

- Reside on mobile platforms such as cell phones or tablets
- Contain user interfaces that take both device characteristics and location attributes
- Often provide access to a combination of web-based resources and local device processing and storage capabilities
- Provide persistent storage capabilities within the platform
- A mobile web application allows a mobile device to access to web-based content using a browser designed to accommodate the strengths and weaknesses of the mobile platform
- A mobile app can gain direct access to the hardware found on the device to provide local processing and storage capabilities
- As time passes these differences will become blurred
CLOUD COMPUTING

Cloud Computing
Provides distributed data storage and processing resources to networked computing devices

Computing resources reside outside the cloud and have access to a variety of resources inside the cloud

Requires developing an architecture containing both frontend and backend services

Frontend services include the client devices and application software to allow access

Backend services include servers, data storage, and server-resident applications

Cloud architectures can be segmented to restrict access to private data
A set of software-intensive systems that share a common set of features and satisfy the needs of a particular market.

These software products are developed using the same application and data architectures using a common core of reusable software components.

A software product line shares a set of assets that include requirements, architecture, design patterns, reusable components, test cases, and other work products.

A software product line allow in the development of many products that are engineered by capitalizing on the commonality among all products within the product line.
SOFTWARE ENGINEERING
SOFTWARE ENGINEERING

Some realities:

• a concerted effort should be made to understand the problem before a software solution is developed
• design becomes a pivotal activity
• software should exhibit high quality
• software should be maintainable

Definition:

• Seminal: The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines.
• IEEE: The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
A LAYERED TECHNOLOGY

Software Engineering
A PROCESS FRAMEWORK

- Process framework
  - Framework activities
  - Umbrella activities
FRAMEWORK ACTIVITIES

Communication  Planning  Modeling  Construction  Deployment

Analysis of requirements  Design  Code generation  Testing

Testing
UMBRELLA ACTIVITIES

- Project tracking and control
- Risk management
- Quality assurance
- Technical reviews
- Measurement
- Configuration management
- Reusability management
- Work product preparation and production
ADAPTING A PROCESS MODEL

- the overall flow of activities, actions, and tasks and the interdependencies among them
- the degree to which actions and tasks are defined within each framework activity
- the degree to which work products are identified and required
- the manner which quality assurance activities are applied
- the manner in which project tracking and control activities are applied
- the overall degree of detail and rigor with which the process is described
- the degree to which the customer and other stakeholders are involved with the project
- the level of autonomy given to the software team
- the degree to which team organization and roles are prescribed
THE ESSENCE OF PRACTICE

1. Understand the problem (communication and analysis).
2. Plan a solution (modeling and software design).
3. Carry out the plan (code generation).
4. Examine the result for accuracy (testing and quality assurance).
UNDERSTAND THE PROBLEM

- Who has a stake in the solution to the problem?
  That is, who are the stakeholders?

- What are the unknowns?
  What data, functions, and features are required to properly solve the problem?

- Can the problem be compartmentalized?
  Is it possible to represent smaller problems that may be easier to understand?

- Can the problem be represented graphically?
  Can an analysis model be created?
PLAN THE SOLUTION

- Have you seen similar problems before?
- Are there patterns that are recognizable in a potential solution?
- Is there existing software that implements the data, functions, and features that are required?
- Has a similar problem been solved? If so, are elements of the solution reusable?
- Can subproblems be defined? If so, are solutions readily apparent for the subproblems?
- Can you represent a solution in a manner that leads to effective implementation?
- Can a design model be created?
CARRY OUT THE PLAN

Does the solution conform to the plan?

Is source code traceable to the design model?

Is each component part of the solution provably correct?

Has the design and code been reviewed, or better, have correctness proofs been applied to algorithm?
Is it possible to test each component part of the solution?

Has a reasonable testing strategy been implemented?

Does the solution produce results that conform to the data, functions, and features that are required?

Has the software been validated against all stakeholder requirements?
The Reason It All Exists

KISS (Keep It Simple, Stupid!)

Maintain the Vision

What You Produce, Others Will Consume

Be Open to the Future

Plan Ahead for Reuse

Think!
### Software Myths

- **Affect managers, customers (and other non-technical stakeholders) and practitioners**

- Are believable because they often have elements of truth,

- but … Invariably lead to bad decisions,

- therefore … Insist on reality as you navigate your way through software engineering
Every software project is precipitated by some business need—
- the need to correct a defect in an existing application;
- the need to adapt a ‘legacy system’ to a changing business environment;
- the need to extend the functions and features of an existing application, or
- the need to create a new product, service, or system.