**CPTS 322 Software Engineering**

1/6

* Software Engineering: holistic process of software development, distilling common principles for constructing and maintaining quality software

1/8

* Identifying Software Characteristics:
  + Intangibility: Unlike physical hardware, developed through programming and coding
  + Complexity: Software systems can be extremely complex
  + Changeability: Software is inherently flexible
  + Reusability: Software can be reused again and again
  + Non-Wearable: Does not wear out on its own
* Failure Modes: hardware fails due to physical wear while software fails due to bugs and logical errors
  + Hardware failure curve makes a bathtub shape
    - Infant Mortality: where the failure rate is high at the start due to manufacturing defects
    - Useful Life: then become stable during a period with low failure rate
    - Wear-Out: increasing failure rate as components age
  + Software failure curve
    - Development and deployment: high failure rate due to bugs and initial testing
    - Operational phase: decreasing failure rate as bugs are fixed and software stabilizes
    - Evolution and Maintenance: potential spikes in failure rate due to updates, new features, or environmental changes
* Types of Software
  + System: operating systems (windows, linuxO
  + Application: programs designed to perform specific tasks for users
  + Development:
  + Embedded:
  + Web: Applications that run on the internet
  + Mobile: Applications designed for mobile devices
* Software Development Life Cycle (SDLC): The process followed to develop software
  + Requirements Analysis: the process of determining user expectations for a product
  + Design: Creating a blueprint for software architecture
  + Implementation: Writing actual code according to design specification
  + Testing: Evaluating software to ensure it meets the quality standards
  + Maintenance: overseeing the software’s deployment and providing ongoing support and updates

1/13

Software Processes and Flow

* + A generic software process framework involves 5 activities: communication, planning, modeling, construction, deployment
    - Communication: collecting the requirements from client
    - Planning:
    - Modeling:
    - Construction:
    - Deployment:
  + Possible direction of flows:
    - Linear: executes in a sequential fashion
    - Iterative: repeats one or more of the activities before proceeding
    - Evolutionary: executes each action in a spiral or circular manner
      * Each circuit through the five activities leads to a more complete model
    - Parallel: executes one or more activities at the same time (not used often)
  + Waterfall Model is purely linear which is a natural process of water falling down from process to process
    - Design the system in such a way that there is no need to move backwards in development
    - It is costly to move back to a previous stage
    - Use this method when work follows a linear fashion, when well-defined adaptations must be made, and when development will be stable

1/15

* Prototyping Process Model
  + The initial prototype includes basic features for users. The second prototype adds features to improve product. Subsequent prototypes add more features and continue to improve products.
  + Development: Prototypes are often “throwaway” versions used to understand and show the client. They are built to understand and clarify the requirements but are not typically part of the final product
  + Advantages: flexibility, early user involvement, early issue detection, improved communication
  + Disadvantages: complication in later stages, increased development time, expensive to repeat stages
    - Generally faster and cost-effective in initial stages, but becomes slow and costly with many prototypes
    - Better for smaller projects but bad for larger, more complicated projects
* Evolutionary Model: very similar to prototyping process model
  + Purpose: main goal is to gradually build a complete system by adding features and improving functionality over time.
  + Advantages: flexibility, user involvement, risk mitigation, quality improvement
  + Development: Each version is more complete than the last
    - It can take more time and costs upfront but will reduce long-term time and costs
    - Better for larger projects

1/17

* Agile Model
  + Thrives on flexibility, responsiveness, collaboration and delivering values swiftly
  + Break down projects into smaller, manageable iterative cycles called sprints
  + Characteristics:
    - Adaptive Planning: avoids long-term, rigid planning; instead focuses on short-term iterations that deliver tangible results
    - Continuous Improvement: promotes culture of constant improvement, teams regularly reflect on processes and outcomes
    - Continuous Integration: fast feedback to adapt and make changes quickly; involves regular integrating code changes and testing them
    - Continuous Deployment: Delivers software frequently
  + Sprints are of the same time duration (2-8 weeks)
  + The end of each sprint delivers a working product
* Scrum
  + Scrum Process:
    - Product Backlog: comprehensive list of desired features 🡪
    - Backlog Refinement: review and prioritization of tasks 🡪
    - Sprint Planning & Backlog: choose features to work on in the sprint🡪
    - Sprint (repeat)
      * Planning
      * Execution
      * Daily Scrum: daily stand-up meetings to track progress
      * Review
      * Retrospective
    - 🡪 Valuable product
* Continuous Integration

1/22

* GitHub
  + SSH Protocol: Method to secure remote login
  + Repository: Contains project folders
  + Fork: Copy of a repository