* 9/29/17
* Project Planning
  + Stages
    - Proposal planning
      * Bidding for contract to develop
      * Usually with only an outline of the requirements
      * Used to set a price for software
      * Estimating cost
        + Staff, hardware, software…
    - Project Startup Planning
      * Know more requirements, but not design or implementation
      * Create plan
    - Development planning
      * Project plan should be regularly amended
      * Schedule, cost-estimate and risks have to change
* Software Pricing
  + Get all of your costs
    - Hardware, software, travel, training, and effort
  + Factors affecting pricing
    - Contractual terms
      * Do you retain ownership of code?
        + If yes, may charge less
    - Cost estimate uncertainty
      * If unsure of cost estimate, may increase price above normal profit
      * If something goes wrong, have a little bit of a fallback
    - Financial Health
      * Devs in financial difficulty may lower price to get a contract
        + Better to make a small profit than to go out of business
    - Market opportunity
      * May charge low cost to get a market share
    - Requirements volatility
      * If requirements are going to change, may lower its price to win the contract. Then charge more for changes
  + Strategies
    - Under pricing
      * Gain contract to keep staff, new market
    - Over pricing
      * More risks
  + Pricing to win
    - Price according to what the developer willing to pay
    - If it is less than development costs, functionality may be reduced, with more being added later
    - Additional costs added with requirement changes
* Plan-driven Development
  + Development process is planned in detail
    - “Traditional” management process
  + Plan is created that records work to be done, who will do it…
  + Pros
    - Allows organizational issues to be taken into account
    - Problems identified before start of the project
  + Cons
    - Many early decisions have to change
  + Project plans
    - Sections
      * Introduction
      * Project Organization
      * Risk Analysis
      * Hardware and Software resource requirements
      * Work Breakdown
      * Project Schedule
      * Monitoring and reporting mechanisms
    - Supplements
      * Configuration management plan
      * Deployment plan
        + How will it be deployed to all the users
      * Maintenance plan
      * Quality plan
        + How will you check to make sure it works
      * Validation plan
        + How to you check to make sure it fills all the customer’s requirements
  + Planning Process
    - Iterative process starting with initial project plan during the startup phase
    - Changes are inevitable
  + Planning assumptions
    - Realistic rather than optimistic
    - Problems always arise
    - Account for unexpected problems
    - Include contingency plans
* Project Scheduling
  + Organize parts of the project into separate tasks
  + Estimate calendar time needed to finish, effort required, and people required
  + Activities
    - Separate tasks
    - Organize concurrently
    - Minimize task dependencies
  + Problems
    - Estimating difficulty is hard
    - Productivity not proportional to number of people
    - Adding people to a late project, just makes it take longer
    - Unexpected always happens
  + Schedule presentation
    - Graphical notations are normally used
    - Gantt Chart
* Agile Planning
  + Project Activities
    - Each activity is the basic planning element and has:
      * Duration in calendar days
      * Effort estimate
      * Deadline for activity completion
      * Defined end-point
* 10/2/17
  + I didn’t pay attention
* 10/4/17
* Requirements Engineering
  + Process of establishing services required and constraints
  + What is a requirement
    - Ranges from high-level abstract statement to detailed mathematical functional specification
    - Requirements serve a dual function
      * May be basis for contract bid - need to be open to interpretation
      * May be basis for contract terms - need to be well defined
  + Types of requirements
    - User requirement
      * In plain english (possibly with diagrams) of the services and constraints (for customer)
    - System Requirements
      * Technical description
  + System Stakeholder
    - Any person or organization who is affected by the system and who has a legitimate interest
    - Types
      * End Users
      * System managers
      * System owners
      * External Stakeholder
  + Agile methods and requirements
    - Plan will change, don’t define everything ahead of time
    - Express requirements as user stories
* Functional and nonfunctional requirements
  + Functional requirements
    - Statements of services to be provided, reaction to inputs, behavior
    - What it should or should not do
  + Non-functional requirements
    - Timing, development process, resources, standards
    - Often apply to the system as a whole
  + Domain requirements
    - Constraints on the system from the domain of operation
    - Requirements unique to this particular system (HIPA laws)
  + Requirements imprecision
    - Problems when functional requirements are not precisely stated
    - Ambiguous requirements can be interpreted differently
    - Should be both complete and consistent
* 10/6/17
* Requirements elicitation
  + Technical staff working with customers to find application domain, provided services, and operational constraints
  + Stages
    - Discovery
    - Classification and organization
    - Prioritization and negotiation
    - Specification
  + Problems
    - Stakeholders don’t know what they want
    - Express requirements in their own terms
    - Conflicting requirements from different stakeholders
    - Organisational and political factors influence system requirements
    - Requirements change during analysis, new stakeholders appear, business environment may change
  + Requirement Discovery
    - Process of gathering info about the required and existing systems and distilling the user and system requirements from this info
    - Interviewing
      * Formal or informal, part of most requirement elicitation
      * Types
        + Closed: pre-determined list of questions
        + Open: various issues are explored
      * Effective interviewing
        + Open-minded, avoid pre-conceived ideas about requirements
        + Prompt interviewee to get discussions going using a springboard question, requirement proposal, or prototype
      * In practice
        + Mix of closed and open
        + Good for getting an overall understanding
      * Problems
        + Each side uses it’s own terminology
        + Not good for understanding domain requirements

May not understand specific domain terminology

Some domain knowledge is so familiar that people find it hard to articulate

* + - Ethnography
      * Observe and analyse how people actually work
      * People don’t have to explain their work
      * Requirements that are derived from the way people actually work rather than the way in which process definitions suggest that they ought to work
      * Cannot identify new features that should be added
      * If it is normal to work around a bug, don’t want to replicate the work-around, want to fix bug
* Requirement Specification
  + Process of writing down user and system requirements in a requirement document
* 10/16/17
* Architectural Design
  + Understanding how software system should be organized and designing the overall structure of that system
  + Critical link between design and requirements engineering. It identifies main structural components in a system and the relationships between them
  + Output is an architectural model
  + Different view of the system
  + Agile and architecture
    - Early stage of agile is to design an overall systems architecture
    - Refactoring the architecture is usually expensive because it affects so many components
* 10/20/17
* Application Architectures
  + Application systems are designed to meet an organizational need
  + Use:
    - Starting point for architectural design
    - Design checklist
    - Organizing work of dev team
    - Means of assessing components for reuse
    - Vocab for talking about application types
  + Examples
    - Data Processing applications
      * Process data in batches w/out user intervention
    - Transaction processing applications
      * Take user requests and update info in database
      * E-commerce systems
      * Reservation systems
      * Structure:
        + IO processing
        + App logic
        + Transaction manager
        + database
    - Event Processing system
      * System actions depend on interpreting events from system’s environment
    - Language processing systems
      * User intentions specified in formal language that is processed by the system
      * Compilers, interpreters
  + Patterns
    - Model view controler
    - Layered
    - Repository
    - Client Server
    - Pipe and filter
  + Architectural Views
    - Logical
    - Proccess
    - Development
    - Physical
* Exam review
  + Cocomo - general view, not math
  + Tags for structured language - don’t memorize, know how to use