CIS 422/522

Standup Progress Report

Project Planning

Stand-up Meeting

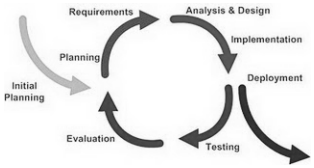
- Technique used in Agile developments
  - Team meets daily
  - Meet standing up to promote efficiency
  - Provides frequent, high-bandwidth feedback
- Report on
  - What have you done?
  - What will you do next?
  - Are there any impediments to progress?
- Answer for each team (2 minutes)

Review: Need to Organize the Work

- Nature of a software project
  - Software development produces a set of interlocking, interdependent work products
    - E.g. Requirements -> Design -> Code -> Test
  - Implies dependencies between tasks
  - Implies dependencies between people
- Must organize the work such that:
  - Every task gets done
  - Tasks get done in the right order
  - Tasks are done by the right people
  - The product has the desired qualities
  - The end product is produced on time

Projects Use Iterative Process

- Process viewed as a sequence of iterations
  - Essentially, a *series of waterfalls*
  - Each iteration builds on the previous one
  - Each iteration produces complete set of work products including deliverable software
- Addresses key risks (provides feedback, have something to deliver)



From Process to Plan

- Process manifests itself in the project plan
  - Process definition is an abstraction
  - Many possible ways of implementing the same process
- *Project plan makes process concrete*, it assigns
  - People to roles
  - Artifacts to deliverables and milestones
  - Activities to tasks over time
- *Project plan* is itself a product of the process
  - Activity: project planning
  - Artifact: the Project Plan
  - Roles: Project Manager (owner), team members
- Evolves as the project proceeds

Project Plan

- Purpose: specifies how project resources will be organized to:
  - Create each deliverable
  - Meet quality goals
  - Address developmental goals (e.g., mitigate risk)
- Audience: answers specific kinds of questions for specific types of users, e.g.:
  - Customers: When will the product be delivered?
  - Stakeholders: What is the development approach? How does it address project risks?
  - Managers: When will tasks be completed? What is the current progress against the plan?
  - Developers: What should I be working on and when?

Plan Outline

- Plan contents (template)
  - Purpose and audience
  - Project background
  - Team roles and responsibilities
  - Risks and risk mitigation
    - What are the key risks?
    - Which mitigation strategies will the project deploy
  - Process: development process being used and its rationale
  - Mechanisms, methods, and techniques
    - What kinds of methods and tools will be used?
    - E.g., planning tools, requirements method, design method IDEs, etc.
  - Detailed schedule and milestones
  - Resources and references

Detailed Schedule and Milestones

- Maps people to tasks over time such that
  - Delivery meets schedule
  - Personnel are fully engaged (time is not wasted)
- Answers: "Who is working on which tasks, what is their progress, and when will they be finished?"
- Inputs
  - Set of artifacts to be created (superset of deliverables)
  - Dependencies/precedence between tasks
  - People filling roles that perform tasks
  - Time budget for each task
- Output
  - Current project schedule
  - Deadline for each task
  - Sequencing among tasks
  - Allocation of people to tasks

Project Plan Template

- Use the template provided in your Assembla team workspace
- This should be a *living document*
  - Changed as the project progresses
  - Ideally, always gives a current view of the *progress against the plan*
    - Shows planned activities
    - Gives snapshot of the current project state
    - This is what I am looking for (or any manager)

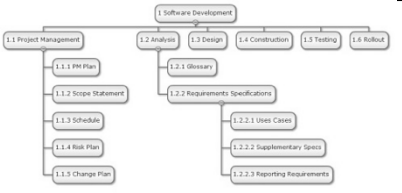
Project Planning Tools

Work Breakdown Structure (WBS)
PERT Chart
Gantt Chart

Work Breakdown Structure

- Structured technique for decomposing work into individual tasks with the goals:
  - Identify the complete set of tasks in the project
  - Provide units of work (for individuals or teams)
  - Provide units of work for scheduling and costing
- Identify hierarchy of tasks and subtasks
  - Identify major tasks in project
  - Decomposing each element into component parts
  - Continuing to decompose until manageable work packages can be mapped to roles
- Works best when:
  - Tasks correspond to key deliverables
  - Sum of tasks is 100% of the work
  - Tasks do not overlap
  - Each leaf task takes about the same amount of time

Work Breakdown Structure



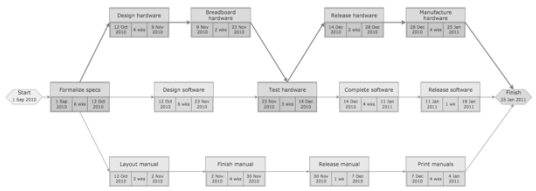
1. Software Development
   1. Project Management
   2. Analysis
      1. Glossary
      2. Requirements Specification
         1. Use Cases
         2. Supplementary Specs…

Equivalent list format

Pert Chart

- Network analysis or PERT is used to identify dependencies between the tasks in the work breakdown structure
- Helps identify where ordering of tasks may cause problems because of precedence or resource constraints
  - Where task B cannot begin before task A ends
  - Where one person cannot do two tasks at the same time
  - Where adding a person can allow tasks to be done in parallel, shortening the project

PERT Chart Example



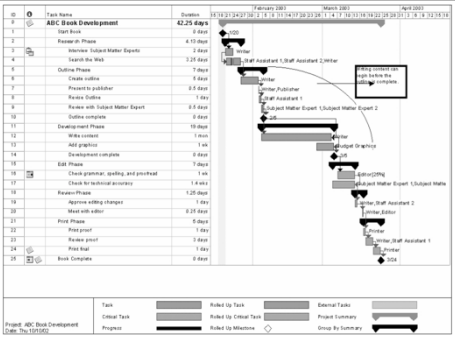
- Which tasks can we start on?
- Which tasks can be done concurrently?
- Which tasks depend on which other tasks?
- *Critical Path*: which path has the longest duration?
  - Gives minimum length of project

http://www.conceptdraw.com/samples/project-chart

Gantt Charts

- Method for *visualizing a project schedule* in one chart showing
  - The set of tasks
  - Start and completion times
  - Task dependencies
  - Responsibilities
- PERT charts can be reformatted as Gantt charts
- Typically requires a tool, e.g., http://www.ganttproject.biz/

Example Gantt Chart



16

Project Milestone Planning

- Milestone planning is used to show the major steps that are needed to reach the goal on time
- Milestones typically mark completion of key deliverables or establishment of baselines
  - Baseline: when a work product is put under configuration management and all changes are controlled
- Often associated with management review points
  - E.g., Requirements baseline, project plan complete, code ready to test
- Can use Gantt or PERT charts to show milestones

A Simple Alternative

Week 1:

Date Assigned	Due Date	Task	Person Responsible	Status	Date Completed
2/3	2/5	Brainstorm project ideas	Everyone	Complete	2/5
2/3	2/4	Set up meeting w/ instructor	Heidi	Complete	2/3
2/3	2/6	Decide on project	Everyone	Complete	2/6
2/6	2/10	Create Git repository	Heidi	Complete	2/6

Week 2:

Date Assigned	Due Date	Task	Person Responsible	Status	Date Completed
2/10	2/10	Decide on software requirements	Everyone	Complete	2/10
2/10	2/15	Plan and design 1st iteration	Everyone	Complete	2/13
2/10	2/10	Set up meeting w/ Kathleen Freeman-Hennessy	Heidi	Complete	2/10
2/13	2/15	Write ConOps	Nicole, Heidi	Complete	3/2
2/13	2/19	Write project plan	Nicole, Heidi	Complete	2/19
2/13	2/22	Write software requirements	Nicole, Heidi	Completed	3/2
2/15	2/24	Implement 1st iteration	Dex, Hans, Yakun	Complete	2/24

How much planning?

- Planning itself consumes resources; how much planning is enough?
- Enough that:
  - Everyone knows what they should be doing
  - Everyone knows what other people are supposed to be doing
  - Everyone knows when specific deliverables should be finished
    - Can track dependencies between their tasks and others
    - Know when task inputs will be available
  - It is easy to determine the current status of the project against plan
- In practice, detail decreases with distance

Project Deliverables

Deliverables Walkthrough

- Consider: What kinds of questions should your documents answer?
  - Assume a manager unfamiliar with the project is reviewing your status
  - Would your documents answer key questions about the project goals and current status?
- *Team page*: Who is on the team and what are their skills?
- *Project plan*
  - Who is responsible for which tasks?
  - What are the anticipated risks and what are you doing to mitigate them?
  - What is your development process and how does it help address the risks?
  - Detailed Schedule & Milestones
    - What is the project schedule of tasks and deliverables?
    - What is the current status relative to schedule?

Walkthrough (2)

- *Software Requirements*
  - 2. ConOps: What capabilities will the software provide the user or customer?
  - 3. Behavioral Requirements: What are the detailed technical requirements?
    - Specific inputs accepted & outputs generated
    - Detailed behavior of any computation (e.g., sort, error responses)
  - 4. Quality Requirements: objective requirements for software qualities (e.g., reliability, performance)
- *Software Design*
  - Architecture: How is the software organized into components? Important relationships between components?
  - Module Interfaces: What are the component interfaces?

Walkthrough (3)

- *Quality Assurance*: How will you check whether the software satisfies functional and quality requirements?
  - Reviews: Which artifacts/properties will be checked by review?
  - Test Plans: How will you test the software?
- *User Documentation*: How will users understand how to install and use the application?
- *Code Documentation:* What do I need to know to find parts of the code responsible for implementing any given requirement or part of the design?
  - How is the code organized in the repository?
  - What does this code component do?

Client Meeting

- Meet teams Wed. in class or after
- Two purposes
  - Progress report
    - Instructor in role of Product Manager
    - Team describes plan, progress against plan
  - Requirements Elicitation
    - Instructor in role of customer/user
    - Ask questions about requirements
    - Schedule review of requirements documents

Questions?

Instructor Meetings

Set up instructor meeting for this week. Will go over progress, plans, any issues

1. What is the plan for delivery?
2. What is the team's current status (progress against plan)?
3. Are you building what the customer wants?
   1. How do you know?
   2. What sorts of activities are planned to check?

Exercise: Plan to Bake a Cake

- How many cooks does it take to bake a cake?
  - Can more people produce a cake faster?
  - Is there a limit to how fast?
  - How could you optimize baking several cakes?