Work Breakdown Structures

Identifying Manageable Activities

Agenda

1. Creating a WBS
2. Using the WBS for Estimation

Work Breakdown Structure

- Work
  - A WBS considers the work that needs to be performed
  - This includes development, but also user manuals, sales support, administration, deployment, media, etc.
- Breakdown
  - Work is broken down (decomposed) into small pieces (activities)
  - Activities are eventually broken down into tasks
  - A task is something that takes less than a week to complete
  - Activities are normally assigned to individuals
- Structure
  - Each unit of work is broken down into a number of components
  - The result is a hierarchical structure
  - The lowest layers are tasks
    - e.g. A function that generates a polynomial collision-handling hash function is completed
    - e.g. Send user manual prototype to printer for an estimate
  - The middle layers could be milestones
    - e.g. “Getting started” tutorial is completed
  - The highest layers are normally deliverables
    - e.g. Source code distribution, with configuration and makefiles, is completed

Terminology

- Activity:
  - Some behaviour that needs to be done
  - Produces some outcome (e.g. a deliverable)
  - Is often decomposed into other activities or tasks
- Task:
  - An activity that is not decomposed
  - Is at the lowest level of the WBS
  - Also called a *work package*

Advantages of a WBS

- The WBS:
  - Gives you a somewhat complete list of tasks
    - Later, this can be a checklist to show how much is still to be done, and how much is done
  - Allows you to easily assign work to team members
  - Requires you to solidify things that are still vague, even after requirements analysis
    - Generating a WBS enables you to methodically decompose the work, exposing new risks and resource requirements

WBS Process Overview

- The WBS process is basically as follows:
  - The WBS is normally created from the top down
  - The estimates are created at the bottom
  - The estimates are summed from the bottom up
  - The totals at the top are used as input for the schedule

Creating a WBS: Top-down

- The top-down approach:
  - Start with the project's overall goal
  - Decompose the goal into deliverables
  - Decompose the deliverables into modules

    …
  - When you are finished you have tasks
    - Tasks should be a few days work or less
- It is a good idea to create the WBS as a group
  - This can prevent important activities from being missed, and can add a level of peer evaluation to the process

When is a WBS done?

- Since WBS is iterative, it could go on forever
- There are guidelines for what is enough:
  - Status/completion is measurable
  - The activity/task is bounded
  - The activity/task has a deliverable
  - Time and cost are easily estimated
  - Activity/task duration is within acceptable limits
  - Work assignments are independent

Status/Completion is Measurable

- Project managers will ask team members about status
  - Status is generally how close they are to completion
- Activities:
  - Status of an activity is the ratio of completed tasks
    - e.g. I'm finished 35 of 55 tasks, so I'm 64% done
  - Completion of an activity is when all of its tasks are complete
- Tasks
  - Status of a task is generally small enough to estimate
    - e.g. I've written all the code for the class, and just need to test it, so I'm about 50% done
  - Completion of a task should take a few days or less

The Activity/Task is Bounded

- The starting and ending points of an activity should be well-known
  - How do you get started on the activity? What task to do first?
  - How do you finish the activity? What is the last task to be done?
- e.g. Optimize the search engine
  - Tasks:
    - **Determine from customers the expected wait time**
    - Measure existing search engine for comparison
    - Examine code for potential slowdowns
    - Make changes where possible
    - Investigate compiler options which could improve performance
    - Update build file to use new compiler options
    - Deploy search engine to test server
    - Measure new search engine performance
    - Verify that search engine meets customer criteria
    - Deploy search engine in public server
    - **Ask customer for review and acceptance**

The Activity/Task Has a Deliverable

- All activities should produce something
  - High-level activities produce the deliverables outlined in the requirements
    - e.g. Source code distribution, user manual, DVD media
  - Lower-level activities can produce other ‘deliverables’
    - e.g. AI engine, device API for bar code readers, a customer class

Time & Cost are Easily Estimated

- The less work is involved in an activity, the easier it is to estimate
- When we get down to task level, it should be possible to accurately estimate time and cost
  - Time: It is less work, so estimates should be accurate, particularly when the task is similar to something else done recently
    - e.g. Write the code to manage persistence of customers to/from the database
      - This is similar to other persistent code you have (or will) write, so can be accurately estimated
  - Cost: You will know if there are additional costs required
    - e.g. Licenses for an IDE, books, training
- We'll deal with estimation separately

Activity/Task Duration is Within Acceptable Limits

- Activities can take a very long time
- However, tasks (the lowest level of decomposition) should be limited in duration
  - Generally, less than 1-2 weeks is considered acceptable
- This is something that can be easily tracked
- Also, if something goes wrong, things should not go to far off track
  - e.g. A 5 day task takes 7 days to complete
  - e.g. A 10 day task was a waste of time, and needs to be re-thought

Work Assignments are Independent

- When a task is assigned to a team member, it should be possible for that team member to complete without further instructions
  - e.g. A team member should not be meeting daily with a manager or customer while working on a 10 day task
- A team member working on a task should have all they need when they begin
  - A team member building on another task's deliverables should start the task after the other task's deliverable is ready
  - e.g.
    - A team member is working on improving the design for the 3D graphical engine
    - When this is complete, another programmer might want to incorporate her code into the graphics engine
    - This should not be done until the graphics engine is complete (with respect to the re-design)

Common Sense with WBS

- Another way to ensure a WBS is complete is to use common sense
- If you were to tell a young child to brush their teeth, they might need more detailed instructions
  - Get your toothbrush and the toothpaste
  - Put a little bit of toothpaste on the toothbrush
  - Brush the front, back, tops, bottoms, and sides of your teeth
  - Spit into the sink
  - Rinse out your mouth with some water
  - Put away the toothbrush and toothpaste
- However, team members have done similar tasks before
  - If you say brush your teeth to an adult, they know what to do
    - Not only is it a waste of time to go into more detail, it is also insulting
    - This is called micromanaging

Estimation

- Estimation involves using the following information in order to make an educated guess about time or resource requirements:
  - Knowledge of the work required (expertise)
    - Ask people who know how long it should take
  - Group knowledge
    - Coming up with estimates as a group is no substitute for expertise, but sometimes expertise is not available
    - Advice from a group is generally more reliable than advice from an individual
  - Prior experience
    - e.g. It previously took 8 minutes to copy, print, seal, stamp, and address 1000 brochures
    - It *should* take about 80 minutes to get 10,000 brochures ready
  - Historical data
    - e.g. The team has worked on 3 other projects, which were all 25-50% over-budget on time
    - Therefore, expect them to go over their own estimates by a similar factor

Estimation Units

- There are several units in estimation:
  - Total time
    - e.g. It should take 3-4 weeks, with a most likely estimate of 18 days
    - This makes it obvious when the project *should* be completed
  - Human resource utilization (effort)
    - e.g. It should take 2-3 person-months, with a most likely estimate of 10 person-weeks
    - This way, you can see how adding people to the project will affect its duration
  - Lines of code (size)
    - e.g. It should be around 50,000 lines of code
    - This figure can then be used for other estimates, such as total time
    - However, few developers count lines of code anymore, so this is not very common
    - Also, not all lines of code are created equal
  - Function points (size/difficulty)
    - An estimate of the number of inputs, outputs, files, database tables, etc. that an application will require
    - e.g. This should have 6 inputs of low complexity (x3), 2 inputs of medium complexity (x4), and … for a 286 function point score

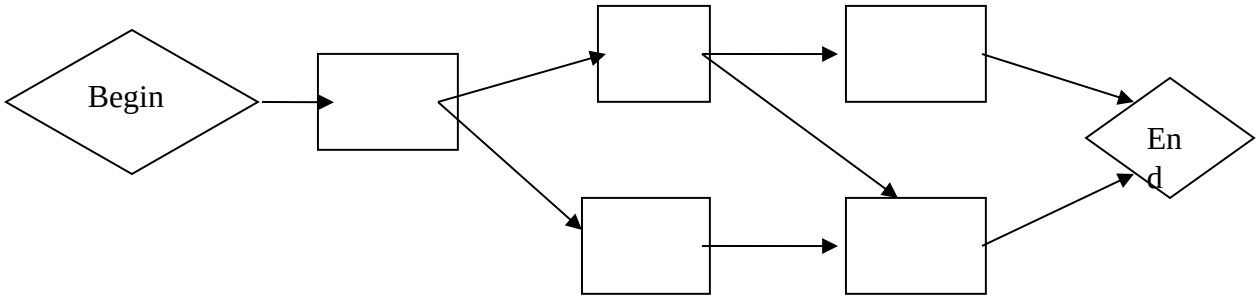
What is a schedule?

- A schedule is a description of start and end times for all the WBS' tasks
  - The schedule accommodates the plan
  - The schedule specifies all dates in terms of offsets from the start date
  - Ideally, the start date is a parameter which can be changed if the project start is delayed
    - This way, real dates can be seen
    - However, dates are not hardcoded so they can be easily changed
- An important part of the schedule is the Gantt chart

Network Diagram

- A network diagram shows task/activity flow
- Flow from one task to another may indicate:
  - Dependencies between the tasks
  - Chronological ordering between the tasks
- Parallel task flows indicate task independence
  - It is not necessarily the case that tasks may be done in parallel, but it is possible

Example :

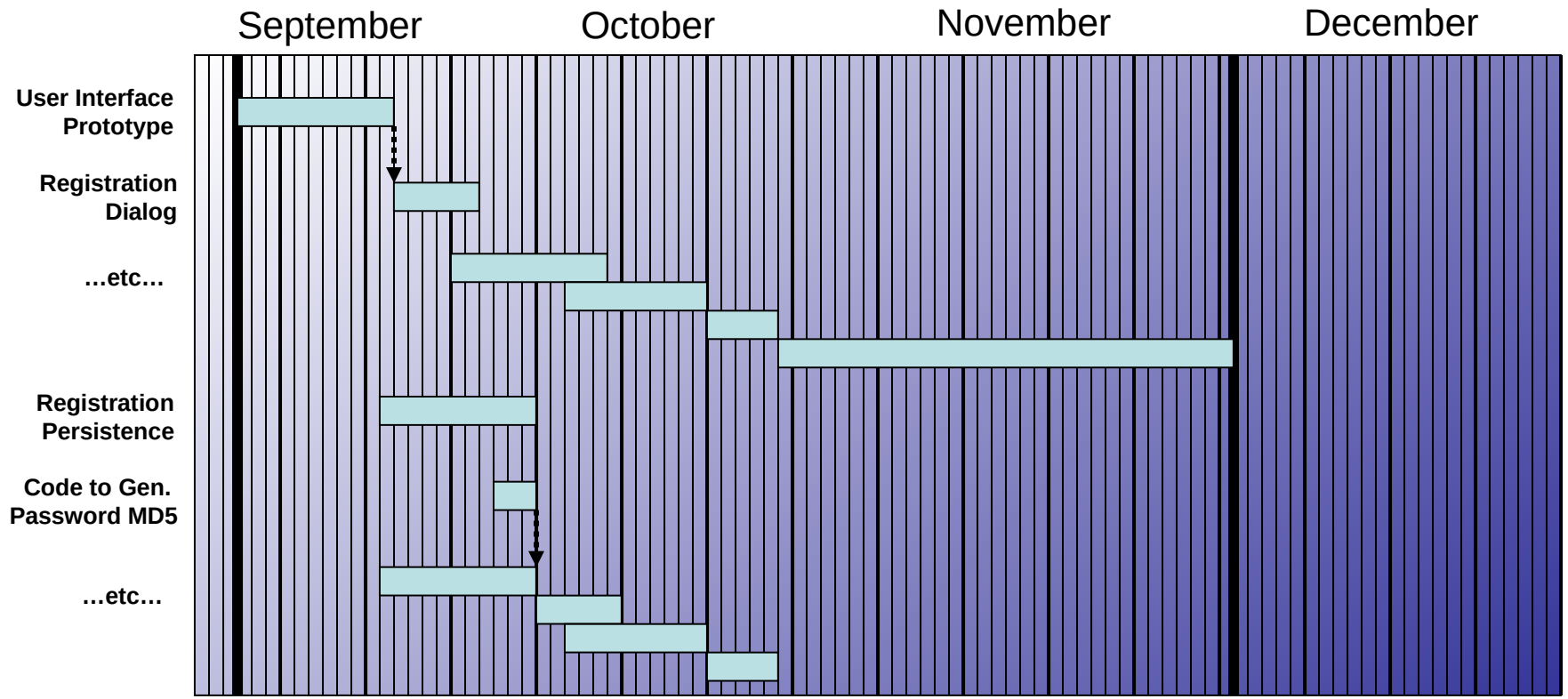


Schedules

- A schedule is an implementation of the project plan
  - However, in industry lingo, a *project plan document* normally includes the schedule
- A common schedule representation is a Gantt chart
  - A Gantt chart is a graphical depiction of the task flow, with dates
  - Dates are shown as the x-axis, so questions about start/end times can be answered
    - e.g. Relative start times of parallel tasks
    - e.g. Completion of all of an activity's tasks
    - e.g. Chronological dependencies between tasks
- However, other formats are possible:
  - A calendar, showing tasks started, active, and completing
  - A list of task descriptions, including start and expected end dates

Gantt Charts

- Visual representation can help when a project manager needs an overview:



Common Schedule Problems

- Problems with estimates or deadlines:
  - Customer or upper management set deadline without team consultation
  - Schedule is based on ‘best case’ estimates
  - Target date moved up without re-adjustment to scope, resources, or schedule
- Problems with requirements:
  - Schedule omits necessary tasks
  - Project size is impossible within allotted time
  - Project is larger than estimated
  - Effort is greater than estimated
- Problems with schedule management:
  - Schedule was based on specific team members that will not be available
  - Schedule slips are ignored when schedule is re-evaluated (velocity)
  - Delays in tasks result in delays in dependent tasks
  - Unfamiliar territory causes unexpected delays
- Problems with productivity:
  - Demotivated personnel (e.g. schedule pressure)
  - Weak personnel
  - Friction between team members