L10: Project management

Motivation
Work breakdown structure
Network diagrams
GANTT charts
Cost estimation
Guidelines
What is project management

Project management has three objectives

– Complete the project on time (this lecture)
– Complete the project within budget (this lecture)
– Complete the project so that it meets the requirements (lectures 2-3)

Why should you care? (beyond your grade in the class)

– Project management consistently rated by employers as one of the most desirable skills sought in new college engineering hires
– Building complex systems is a tremendous challenge, not only technically but also managerially
Work breakdown structure

The WBS is the first step in project planning

– The WBS is ordered as a set of activities that must be completed to accomplish the project objectives
– Each activity consists of tasks and deliverables:
  • Task: actions that accomplish a goal
  • Deliverables: entities delivered to the project upon task completion
– In addition, each activity must have
  • A definition of the work to be done
  • A timeframe for completing it
  • Resources needed to complete it
  • Person(s) responsible for it
  • Dependencies
  • Checkpoints for monitoring progress

– These activities are then organized hierarchically
Example: thermometer design

There are three main tasks

– Analog interface circuitry
– LED & digital circuitry
– Integrate and test
<table>
<thead>
<tr>
<th>ID</th>
<th>Activity</th>
<th>Description</th>
<th>Deliverables / Checkpoints</th>
<th>Duration (days)</th>
<th>People</th>
<th>Resources</th>
<th>Predecessors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Interface Circuitry</td>
<td>Complete the detailed design and verify it in simulation.</td>
<td>• Circuit schematic • Simulation verification</td>
<td>14</td>
<td>Rob (1)</td>
<td>• PC</td>
<td>1.1</td>
</tr>
<tr>
<td>1.1</td>
<td>Design Circuitry</td>
<td>Complete the detailed design and verify it in simulation.</td>
<td>• Circuit schematic • Simulation verification</td>
<td>14</td>
<td>Rob (1)</td>
<td>• PC</td>
<td>1.1</td>
</tr>
<tr>
<td>1.2</td>
<td>Purchase Components</td>
<td>Identify parts • Place order • Receive parts</td>
<td></td>
<td>10</td>
<td>Rob</td>
<td></td>
<td>1.1</td>
</tr>
<tr>
<td>1.3</td>
<td>Construct and Test Circuits</td>
<td>Build and test.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3.1</td>
<td>Current Driver Circuitry</td>
<td>Test of circuit with sensing device.</td>
<td>• Test data • Measurement of linearity</td>
<td>2</td>
<td>Jana (1)</td>
<td>• Test bench • Thermometer</td>
<td>1.2</td>
</tr>
<tr>
<td>1.3.2</td>
<td>Level Offset and Gain Circuitry</td>
<td>Test of circuit with voltage inputs.</td>
<td>• Test data • Measurement of linearity</td>
<td>3</td>
<td>Rob (1)</td>
<td>• Test bench • Thermometer</td>
<td>1.2</td>
</tr>
<tr>
<td>1.3.3</td>
<td>Integrate Components</td>
<td>Integrate the current driver and offset circuits.</td>
<td>• Test data verifying functionality and linearity requirement</td>
<td>5</td>
<td>Rob (1)</td>
<td>• Test bench • Thermometer</td>
<td>1.3.1</td>
</tr>
<tr>
<td>2</td>
<td>LED and Driver Circuitry</td>
<td>Make selection of A/D converter.</td>
<td></td>
<td></td>
<td>Alex</td>
<td>• Internet</td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td>Research A/D Converters</td>
<td>Make selection of A/D converter.</td>
<td>• Identify types, cost, and performance • Identify two potential converters for purchase</td>
<td>1</td>
<td>Alex</td>
<td>• Internet</td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td>Complete Hardware Design</td>
<td>Design conversion hardware.</td>
<td>• Circuit schematic • Simulation verification</td>
<td>7</td>
<td>Ryan (1)</td>
<td>• Digital circuit simulator</td>
<td>2.1</td>
</tr>
<tr>
<td>2.3</td>
<td>Purchase LED and Driver Components</td>
<td>Purchase LED and Driver Components</td>
<td>• Identify parts • Place order • Receive parts</td>
<td>10</td>
<td>Rob</td>
<td></td>
<td>2.2</td>
</tr>
<tr>
<td>2.4</td>
<td>Construct and Test</td>
<td>Test with supply voltage input.</td>
<td>• Test data showing digital output vs. voltage inputs</td>
<td>5</td>
<td>Alex (1)</td>
<td>• Test bench • Logic analyzer</td>
<td>2.3</td>
</tr>
<tr>
<td>3</td>
<td>System Integration and Test</td>
<td>Complete integration of front-end and LED driver circuitry.</td>
<td>• Test data demonstrating functionality from temp input to LED output • System linearity measurement</td>
<td>7</td>
<td>Alex (1)</td>
<td>• Test bench • Digital logic analyzer • Thermometer</td>
<td>1.3.3</td>
</tr>
</tbody>
</table>
Estimating activity duration

- Based on PERT (Project Evaluation and Review Technique)
- First, generate
  - $t_a$: the most optimistic time estimate
  - $t_b$: the most pessimistic time estimate
  - $t_m$: the most realistic time estimate
- Then compute the estimated activity duration as
  \[
  t_e = \frac{t_a + 4t_m + t_b}{6}
  \]
- The advantage of this estimate is that it forces you to consider best-case and worst-case scenarios
Network diagrams

A directed graph of the activities and their dependencies

- Activity on note (AON) form – the one we will use
  - Nodes are activities
  - Arrows (arcs) indicate the precedence relationships
- Activity on arc (AOA) form – an alternative
  - Nodes represent the realizations of project milestones (events)
  - Arcs represent the activities
The critical path

- The path from start to end with the longest duration
- Activities in this path are most important: if they experience slippage, they delay completion of the project
- Other paths can become critical paths if their activities experience sufficient slippage

float

- An estimate of the margin for each activity; to calculate it:
  - Identify all paths of the network diagram and their durations
  - The path with the longest duration \( t_{CP} \) is the critical path
  - If an activity is in the critical path, it has zero float
  - Otherwise find the longest path \( t_{LP} \) to which the activity belongs
  - The activity’s float is \( t_{CP} - t_{LP} \)

- See example 10.1 for float time calculation
Gantt charts

A bar graph representation of activities on a timeline

- An effective way to visualize the WBS
- Activity dependencies can be incorporated
Cost estimation

**Break-even analysis**

- Aims to estimate the number of units to be sold so costs and revenues are equal (i.e., no profits, no losses)

- Considers two types of cost:
  - **Fixed costs**: rent, overhead, insurance, property taxes, design and development costs, capital expenditures, market research, maybe also labor costs
  - **Variable costs**: depend on the number of items being produced: raw materials, inventory, energy costs, maybe also labor costs

- The break-even point

\[
\text{fixed cost} + n \times \frac{\text{variable cost}}{\text{unit}} = n \times \frac{\text{sale price}}{\text{unit}}
\]

\[
\frac{\text{TOTAL COSTS}}{\text{REVENUE}}
\]
break-even point

revenue = \( n \times \text{sales price/unit} \)

profit

total costs = fixed + variable costs

variable costs = \( n \times \text{cost/unit} \)

fixed costs

units sold, \( n \)

dollars
Cost models

– Costs must be accurately estimated to meet expected profits
– Components of the cost include labor, equipment and supplies
  • Equipment costs are relatively easy to estimate
  • Labor costs are tied to the project duration and are typically the largest

Estimating labor costs

– One option is to use the same equation of activity duration
  \[
  \text{cost} = \frac{\text{cost}_a + 4\text{cost}_m + \text{cost}_b}{6}
  \]
– A more formal method is to use empirically derived models
  • These models use past data to derive an equation that relates cost to a measurable variable, e.g., kilo lines of code (KLOC)
Empirical models for labor cost

- For small development projects (4-10 KLOC), IBM used a linear model
  \[ E = a \times KLOC + b \]
  - where \( E \) denotes effort
- As complexity increases, an exponential model is more realistic
  \[ E = a(KLOC)^b \]
  - IBM used data from 60 projects to estimate \( a = 5.2 \) and \( b = 0.91 \)

Limitations of these models

- The mythical man-month: adding more people to a project will not reduce development time (Brooks, 1975)
  - E.g., increasing the number of software developers by \( \times 100 \) will not reduce development time by 100
- In fact, “adding manpower to a late software project makes it later”
Potatoes picked vs. Number of people:
- Linear increase

Time taken vs. Number of people:
- Decreasing curve

D. Bell, “Software engineering for students”, Addison-Wesley, 2005
Constructive Cost Model (COCOMO)

– Uses not only KLOCs but other parameters as model inputs, such as
  • Type of technology being used and its maturity
  • Size of the team
  • Experience of the engineers, and
  • Time frame for completing the project

– Three COCOMO levels
  • Basic
  • Intermediate
  • Detailed

http://en.wikipedia.org/wiki/COCOMO
**Basic COCOMO**

- Computes cost as a function of KLOC for three classes of projects
  - **Organic projects** - "small" teams with "good" experience working with "less than rigid" requirements
  - **Semi-detached projects** - "medium" teams with mixed experience working with a mix of rigid and less than rigid requirements
  - **Embedded projects** - developed within a set of "tight" constraints
- Model equations
  \[
  E = a \times (KLOC)^b \text{ [person/month]} \\
  D = c \times (E)^d \text{ [months]}
  \]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organic</td>
<td>2.4</td>
<td>1.05</td>
<td>2.5</td>
<td>0.38</td>
</tr>
<tr>
<td>Semi-detached</td>
<td>3.0</td>
<td>1.12</td>
<td>2.5</td>
<td>0.35</td>
</tr>
<tr>
<td>Embedded</td>
<td>3.6</td>
<td>1.20</td>
<td>2.5</td>
<td>0.32</td>
</tr>
</tbody>
</table>
Intermediate COCOMO

- Considers 15 attributes organized into four “cost drivers”
  - Product attributes, e.g., database size, required reliability
  - Hardware attributes, e.g., run-time constraints, memory constraints
  - Personnel attributes, e.g., team capabilities, experience
  - Project attributes, e.g., development schedule

- Model equation

\[ E = a \times (KLOC)^b \times EAF \text{ [person/months]} \]

- where EAF (effort adjustment factor) is the product of a tabulated effort multiplier (see next slide) for the 15 attributes
<table>
<thead>
<tr>
<th>Cost Drivers</th>
<th>Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Very Low</td>
</tr>
<tr>
<td><strong>Product attributes</strong></td>
<td></td>
</tr>
<tr>
<td>Required software reliability</td>
<td>0.75</td>
</tr>
<tr>
<td>Size of application database</td>
<td></td>
</tr>
<tr>
<td>Complexity of the product</td>
<td>0.70</td>
</tr>
<tr>
<td><strong>Hardware attributes</strong></td>
<td></td>
</tr>
<tr>
<td>Run-time performance constraints</td>
<td></td>
</tr>
<tr>
<td>Memory constraints</td>
<td></td>
</tr>
<tr>
<td>Volatility of the virtual machine environment</td>
<td>0.87</td>
</tr>
<tr>
<td>Required turnaround time</td>
<td></td>
</tr>
<tr>
<td><strong>Personnel attributes</strong></td>
<td></td>
</tr>
<tr>
<td>Analyst capability</td>
<td>1.46</td>
</tr>
<tr>
<td>Applications experience</td>
<td>1.29</td>
</tr>
<tr>
<td>Software engineer capability</td>
<td>1.42</td>
</tr>
<tr>
<td>Virtual machine experience</td>
<td>1.21</td>
</tr>
<tr>
<td>Programming language experience</td>
<td>1.14</td>
</tr>
<tr>
<td><strong>Project attributes</strong></td>
<td></td>
</tr>
<tr>
<td>Application of software engineering methods</td>
<td>1.24</td>
</tr>
<tr>
<td>Use of software tools</td>
<td>1.24</td>
</tr>
<tr>
<td>Required development schedule</td>
<td>1.23</td>
</tr>
</tbody>
</table>
Detailed COCOMO

- Effort is calculated for each part of the software life cycle
  - plan and requirement
  - system design
  - detailed design
  - module code and test
  - integration and test
Guidelines for creating a project management plan

- Build the plan after the design architecture is complete
- Take the initial time estimates for activities and double them
- Assign a lot of time for testing and integration
- Factor in lead times for part ordering
- Assign a project manager
- Do not assign all team members to all tasks
- Track the progress versus the plan
- Don’t become a slave to the plan
- Experience counts
The program manager

- Program manager ≠ the “boss”
- Responsible for planning and organizing the project
  - Develop the WBS, Gantt chart, and cost estimates
  - Monitor checkpoints and deliverables against the plan
  - Develop strategies for reacting to slippage in tasks
- Additional responsibilities may include
  - Purchasing of materials
  - Controlling spending
The project plan

You’ll need to develop one for the proposal

– Work breakdown structure
  • Activities
  • Deliverables
  • Responsibilities (only primary)
  • Duration
  • Resources, and
  • Dependencies (see table 10.1)
  • Please also summarize responsibilities in an activity matrix

– Project plan in graphical representation
  • Gantt chart: good to show timeline, and/or
  • Network diagram: good to show dependencies

– Costs
  • Tabulated list of costs for equipment, materials, and labor
  • Since this is a capstone project, estimate labor costs in worker-hours

<table>
<thead>
<tr>
<th>Activity</th>
<th>Description</th>
<th>Pete</th>
<th>Ann</th>
<th>Mike</th>
<th>Sue</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Lorem ipsum</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Lorem ipsum</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Lorem ipsum</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Lorem ipsum</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td>Lorem ipsum</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>6</td>
<td>Lorem ipsum</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Lorem ipsum</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>8</td>
<td>Lorem ipsum</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>