Process Models

Diagramming the steps in a task

Software Process

- A process model depicts the tasks that are done to achieve software development
- Process models usually depict
  - tasks
  - entities (files, data, etc)
  - actors
  - decisions

Purpose of Process Models

- Understanding of what is (should be) done
  - training
  - management
- Automation
  - clerical
  - analysis

Project Planning

Management should be based on effective and thorough planning
Pressman pp165-192, 145-163

TTYP – process model – turn in

- Draw a process model for a team meeting
  Use stick figures for actors, boxes for artifacts, and ovals for processes
## Project Plans
- **Project Plan** - overall plan anticipating progress of all aspects
  - Quality Plan - how to achieve quality
  - Staff Development Plan - how to train staff
  - Validation Plan - how to test product

## Project Plan Sections
- **Intro** - objectives and constraints
- **Organization** - people, roles, reporting
- **Risk Analysis** - identify risks and risk reduction
- **Resources** - hardware and support software
- **Work breakdown** - identify tasks
- **Schedule** - dependencies and estimates

## Risk Analysis (pp145-163)
- Risk analysis will be covered more thoroughly in cis541
- Initially, consider what unique characteristics of THIS project threaten to prevent its completion.
- Identify risks and indicate what the team can do to lessen the risks (usually about order of tasks or prototypes)

## WBS - work breakdown schedule
- Dividing a task into subtasks
  - each task should be less than some max time
  - each task should have some artifact as a product
- Build tree of tasks and subtasks
  - Leftmost items can be related to lifecycle

## WBS for project
- **Requirements**
  - Develop 1 page overview
    - Review email with project statement
    - Identify issues
  - Prepare for Presentation 1
    - Develop consensus
    - Assign tasks
    - Review documents
    - Practice presentation
  - Design

## Schedules
- **Tasks**
  - resource requirements - people and time
  - dependencies - what tasks have to finish before this one can start
- **Critical Path**
  - the sequence of tasks that determine the minimum time to complete the project
Bar Chart (Gantt)

- Each task is a horizontal bar whose length represents the estimated duration and whose starting point is the ending of the tasks on which it is dependent
- Graphical depiction showing duration and parallelism

Example Gantt Chart

Example

Example Node pred time start finish crit
A 5
B 4 8
C 3
D 3
E 6

PERT Diagrams

- Program Evaluation and Review Technique

Critical Path

- If there are N tasks with dependencies between the tasks, what set of tasks determines the minimum completion time for the whole project?
- This is the critical path.

Critical Path Algorithm (part 1)

- For each node (until completion time of all nodes are calculated)
  » if the predecessors are completed, then take latest completions time and add required time for this node;
  » this is the completion time.
- The node with the latest completion time determines the earliest completion time for project.
Example

<table>
<thead>
<tr>
<th>Node</th>
<th>pred</th>
<th>time</th>
<th>start</th>
<th>finish</th>
<th>crit</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>*</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>8</td>
<td>5</td>
<td>13</td>
<td>*</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>7</td>
<td>13</td>
<td>20</td>
<td>*</td>
</tr>
<tr>
<td>D</td>
<td>A</td>
<td>4</td>
<td>5,10</td>
<td>9,14</td>
<td>*</td>
</tr>
<tr>
<td>E</td>
<td>D</td>
<td>6</td>
<td>9</td>
<td>15</td>
<td>*</td>
</tr>
</tbody>
</table>

Critical Path Algorithm (part 2)

- Starting with the node with latest completion time, mark it as critical.
- Select predecessor node with latest completion time, mark it as critical.
- Continue until reaching start node.

Example

<table>
<thead>
<tr>
<th>Node</th>
<th>pred</th>
<th>time</th>
<th>start</th>
<th>finish</th>
<th>crit</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>*</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>8</td>
<td>5</td>
<td>13</td>
<td>*</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>7</td>
<td>13</td>
<td>20</td>
<td>*</td>
</tr>
<tr>
<td>D</td>
<td>A</td>
<td>4</td>
<td>5,10</td>
<td>9,14</td>
<td>*</td>
</tr>
<tr>
<td>E</td>
<td>D</td>
<td>6</td>
<td>9</td>
<td>15</td>
<td>*</td>
</tr>
</tbody>
</table>

Critical Path Algorithm (part 3)

- Slack time is the amount of time that a specific task can be delayed without changing the project delivery time
- For each noncritical node starting with the last node, determine the earliest of the latest start times of the successor nodes. This is the latest completion time. Make the latest start time for the node to reflect this time.

Presentation 1

- Web page and on disk
  - Project statement
  - Project plan
  - Diagrams including Object models
  - Powerpoint presentation
  - Weekly task assignments
  - Individual Time Logs