Software project planning

The Plan, project estimation, decomposition, tools

(Pressman / Software Engineering / Chapter 5)
Effective team meetings

• Use an AGENDA, distributed in advance
  – People should know what is to be discussed
• Use team meeting for
  – Analyzing, reporting what has been done
  – Plan what should be done next
  – Making decisions
  – NOT FOR DOING THE WORK
    • Exception: ”brain-storming activities”
Simple AGENDA

GROUP A MEETING, DC 3101 Nov 7. at 10.15
Present: NN, NN, NN, NN

AGENDA:
* Code status (dev manager)
* Decision on testing tools
* The documentation templates (process manager)
* Test plan (testing manager)
* Next meeting

Agenda distributed 1-10 days before meeting
The process

• Some process is normally ;) used for generating software
  – linear
  – prototyping
  – RAD
  – incremental
  – spiral
  – WINWIN
  – component-based
  – formal methods
Process / product

• Example: Software process involving
  – Customer communication
  – Planning
  – Risk analysis
  – Engineering
  – Construction and release
  – Customer evaluation

• Map each of these activities to the product functions
Project lifecycle relationships

Business Life Cycle

<table>
<thead>
<tr>
<th>Policy Planning</th>
<th>Needs Identification</th>
<th>Project Conception</th>
<th>Realization</th>
<th>Product in Service</th>
<th>Disposal</th>
</tr>
</thead>
</table>

Sales process

Product Life Cycle

| Feasibility | Acquisition | Operations | Disposal |

Software process

Project Life Cycle

| Concept | Development | Implementation | Termination |

Project planning

• Provides a framework for the project
  – Software scope
  – Resources
  – Expectations

• Make it possible to make reasonable estimates for
  – Resources
  – Schedules
  – Costs
Why a project plan?

• Forces the making of necessary decisions
  – Estimates, scheduling, budgeting
• Communicates decisions to other
• Checklist / action plan
• Who, what, when, why, where, how
• The ”rules” of the project written in text, no misinterpretation
The project plan

• The very minimal project plan
  – Project objectives
  – Project deliverables
  – Project schedule
  – Supporting plans
Project objectives

I Stakeholders

– Name/contact information of stakeholders
  • Project manager(s)
  • Customer – the ones who receive deliverables
  • Supplier – the ones doing the work
  • Sponsors – paying for the thing
  • Other parties

II Objectives

– Objective I
– Objective II ...
Project deliverables

III Project deliverables

– List of things that the project has to deliver in order to meet the objectives

– Can be

  • System requirements
  • System design
  • Implementation
  • Installation
  • Documents
Schedule

IV Project schedule

– List of tasks/activities that has to be performed
– The name of person / role that is responsible for the activity
– Estimated times for tasks & activities
– Milestones
– Can use GANTT chart
<table>
<thead>
<tr>
<th>Controls</th>
<th>Apr</th>
<th>May</th>
<th>Jun</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>31/02</td>
<td>7/02 4/14/02</td>
<td>5/02 5/12/02</td>
</tr>
<tr>
<td></td>
<td>14/02</td>
<td>21/02 4/28/02</td>
<td>19/02 5/26/02</td>
</tr>
<tr>
<td></td>
<td>28/02 5/5/02</td>
<td>12/02 6/2/02</td>
<td>9/02 6/16</td>
</tr>
<tr>
<td>SC-USA</td>
<td>6 hrs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>hrs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMUser</td>
<td>6 hrs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>hrs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SC-India</td>
<td>6 hrs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>hrs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Colour coding:**
- **Time available**
- **Time allocated**
- **Overallocation**
Supporting plans

• V Roles & responsibilities
  – persons / organizations involved in the project and their role in the project

• VI Communication plan
  – To whom should what information be sent and when?

• VII Risks & risk management
  – List of risks
  – What action should be taken to minimize risk / counter-actions
Resources

• People
  – Necessary skills
    • Organizational
    • Technical

• Reusable software components

• Hardware / software tools
# Risk analysis (i)

<table>
<thead>
<tr>
<th>Risk</th>
<th>Probability</th>
<th>Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organisational financial problems force reductions in the project budget.</td>
<td>Low</td>
<td>Catastrophic</td>
</tr>
<tr>
<td>It is impossible to recruit staff with the skills required for the project.</td>
<td>High</td>
<td>Catastrophic</td>
</tr>
<tr>
<td>Key staff are ill at critical times in the project.</td>
<td>Moderate</td>
<td>Serious</td>
</tr>
<tr>
<td>Software components that should be reused contain defects which limit their functionality.</td>
<td>Moderate</td>
<td>Serious</td>
</tr>
<tr>
<td>Changes to requirements that require major design rework are proposed.</td>
<td>Moderate</td>
<td>Serious</td>
</tr>
<tr>
<td>The organisation is restructured so that different management are responsible for the project.</td>
<td>High</td>
<td>Serious</td>
</tr>
</tbody>
</table>
## Risk analysis (ii)

<table>
<thead>
<tr>
<th>Risk</th>
<th>Probability</th>
<th>Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>The database used in the system cannot process as many transactions per second as expected.</td>
<td>Moderate</td>
<td>Serious</td>
</tr>
<tr>
<td>The time required to develop the software is underestimated.</td>
<td>High</td>
<td>Serious</td>
</tr>
<tr>
<td>CASE tools cannot be integrated.</td>
<td>High</td>
<td>Tolerable</td>
</tr>
<tr>
<td>Customers fail to understand the impact of requirements changes.</td>
<td>Moderate</td>
<td>Tolerable</td>
</tr>
<tr>
<td>Required training for staff is not available.</td>
<td>Moderate</td>
<td>Tolerable</td>
</tr>
<tr>
<td>The rate of defect repair is underestimated.</td>
<td>Moderate</td>
<td>Tolerable</td>
</tr>
<tr>
<td>The size of the software is underestimated.</td>
<td>High</td>
<td>Tolerable</td>
</tr>
<tr>
<td>The code generated by CASE tools is inefficient.</td>
<td>Moderate</td>
<td>Insignificant</td>
</tr>
</tbody>
</table>
Risk management strategies (i)

<table>
<thead>
<tr>
<th>Risk</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organisational financial problems</td>
<td>Prepare a briefing document for senior management showing how the project is making a very important contribution to the goals of the business.</td>
</tr>
<tr>
<td>Recruitment problems</td>
<td>Alert customer of potential difficulties and the possibility of delays, investigate buying-in components.</td>
</tr>
<tr>
<td>Staff illness</td>
<td>Reorganise team so that there is more overlap of work and people therefore understand each other’s jobs.</td>
</tr>
<tr>
<td>Defective components</td>
<td>Replace potentially defective components with bought-in components of known reliability.</td>
</tr>
</tbody>
</table>
Software scope

• Customer
  – Who is behind the request for this work?
  – Who will use the solution?
  – What will be the economic benefit of a successful solution?
  – Is there another source for the solution?

• The solution
  – When is the solution good?
  – What problems will this solution address?
  – Which environment
Project estimation

• Can you estimate the effort (man hours / money) needed to
  – clean your apartment?
  – prepare for an exam?
  – make a dinner?
  – build a house?
  – make a customization to an existing software?
  – make a movie reservation system?
Project estimation

• If your estimate is too small...
  – You´ll go bankrupt!! (you can´t afford to pay the salaries)

• If your estimate is too big...
  – You´ll go bankrupt!! (you´ll lose any competitive bidding)
Estimation historically

Software

Hardware

1950

Error in estimate of software costs:
Very little impact

Large impact

Software

Hardware

2000
Estimation...

• Well defined requirements
  – How to estimate if not known what to do?
• Historical data is of GREAT use
  – Software metrics for past projects
  – What worked, what caused problems?
• Update estimates during the project
  – Can refocus, remove requirements that cannot be finished on time
• Best case estimates, worst case estimates
Decomposition

• Software estimation is a form of problem solving
  – Large problems are divided into smaller problems and solved separately
    • Divide and conquer

• The software system is divided into smaller systems / tasks and estimated separately
Decomposition...

• Fine grained decomposition (e.g. very detailed tasks)
  – Errors in estimates will accumulate
    • Often estimating wrong in one direction...

• Coarse grained decomposition
  – Difficult to make fair estimated
Software sizing

• "Fuzzy "logic sizing
  – Identify type of problem, establish magnitude and scale it for this project
    • Historical data is essential
    • Experience

• Function point sizing

• Standard component sizing
  – Identify components needed and make a sum of efforts needed

• Change sizing
  – Used for existing software, how much changes are made?
Work estimation – “fuzzy”

• Two approaches
  – Systematic use of history data
  – Educated guess

• Three (usual) metrics
  – Number of function points
  – Amount of code lines
  – A metric based on subjective ad-hoc view
# Function points vs (S)LOC

<table>
<thead>
<tr>
<th>Language</th>
<th>SLOC / function points (on average)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly</td>
<td>320</td>
</tr>
<tr>
<td>C</td>
<td>128</td>
</tr>
<tr>
<td>Cobol</td>
<td>105</td>
</tr>
<tr>
<td>Fortran</td>
<td>105</td>
</tr>
<tr>
<td>Pascal</td>
<td>90</td>
</tr>
<tr>
<td>Ada</td>
<td>70</td>
</tr>
<tr>
<td>C++</td>
<td>64</td>
</tr>
<tr>
<td>Ada 95</td>
<td>53</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>32</td>
</tr>
<tr>
<td>Object languages</td>
<td>30</td>
</tr>
<tr>
<td>Smalltalk</td>
<td>22</td>
</tr>
<tr>
<td>4. generation languages</td>
<td>20</td>
</tr>
<tr>
<td>Powerbuilder</td>
<td>16</td>
</tr>
<tr>
<td>Koodigenerators</td>
<td>15</td>
</tr>
<tr>
<td>SQL</td>
<td>12</td>
</tr>
<tr>
<td>Spread sheet programs</td>
<td>6</td>
</tr>
<tr>
<td>Graphical languages (icons)</td>
<td>4</td>
</tr>
</tbody>
</table>
People and efforts

• Example:
  – 1 person 5000 LOC/year
  – 4 persons 4 * 5000 LOC / year = 20000 LOC/year ???
    • No, time needed for communication
      – corresponding to 250 LOC/year/communication path
  – 4 * 5000 – 6 * 250 LOC = 18500 LOC/year
Make / buy issues

• More efficient to acquire than to develop??
  – Estimate internal cost to develop and the delivery date
  – Identify potential candidates
  – cost of acquisition + cost of customization < cost of developing
  – can we reuse the component?
Tools

• A lot of tools available, do the following
  – Sizing of project deliverables
  – Selecting project activities
  – Staffing levels
  – Predicting software effort
  – Predicting software costs
  – Predicting software schedules

• However: They are nothing but tools, the actual planning and estimation is done by you!
Management activities

• Proposal writing.
• Project planning and scheduling.
• Project costing.
• Project monitoring and reviews.
• Personnel selection and evaluation.
• Report writing and presentations.
Management commonalities

• These activities are not peculiar to software management.
• Many techniques of engineering project management are equally applicable to software project management.
• Technically complex engineering systems tend to suffer from the same problems as software systems.
How to manage time?

- Project planning
- GANTT-charts (milestones, check-ups)
- Work effort estimation (very difficult….)
- Work distribution
- Stick to your plans (no extras….)
- Time reporting (essential for upcoming projects)
- How to avoid last-minute panic???
Project phases

• All projects are divided into phases
• All phases together are known as the project life cycle
• Each phase is marked by completion of Deliverables
• Identify the primary software project phases
Time tracking

[Image of a timesheet tracking application]

1. Add or remove tasks.
2. Utilization and task tracking.
Time tracking
Time management tools

- Microsoft project
- Other tools ....
Picking a team leader

• One person from each group should be selected as team leader / project manager

• Tasks
  – Communicate with customer
  – Arranges team meetings
  – Is responsible for the project plan
  – Is responsible for deadlines

• TEAM MUST AGREE – the team leader IS THE BOSS!!
Key points

• Good project management is essential for project success.
• The intangible nature of software causes problems for management.
• Managers have diverse roles but their most significant activities are planning, estimating and scheduling.
• Planning and estimating are iterative processes which continue throughout the course of a project.
Finally

• My personal favorite methodology:

Keep

• Break down the big picture to simple sub problems
• Solve the simple problems with simple methods
• Look on the essentials
  • What are we trying to do
  • What will be done, by when?
  • Who is responsible for a function?
  • How will the job be done
  • What resource do we need

It

Simple

Stupid!