Software Project Management

The art of herding cats

Some sobering stats

- 42% of technology projects are abandoned before completion
- >50% of completed projects exceed cost/time budgets by 150% or more
- 17% get solid business sponsorship

Source: Computer World, 3/22 p. 32

Why?

- Business failures
  - Poorly conceived ideas
  - Lack of real sponsorship
  - Unrealistic goals & schedules
- Technologists failures
  - Unrealistic or vague schedules
  - Failure to monitor & react appropriately
  - Just plain old bad designs

Successful projects

- Align business people & technologists
  - Engaged business leadership is essential
  - Increasingly, technologists *must* be business-savvy
- Establish climate of realism
  - What's doable
  - What's essential
- Tracking, tracking, tracking

Milestones this week

DATE	TEAM	MILESTONE
9-Apr	Seige	minimum implementation complete
9-Apr	sound images	create visual frames for 1 sec. Music at 20 fps
10-Apr	redemption	menu interface, mostly complete animation engine
11-Apr	PIM	publishing debugged, UI for new projects complete
9-Apr	At5000	functionality complete
9-Apr	trading system	market clearing, 3 strategies & basic investor interfa
12-Apr	clipbook	file I/O & mostly completed GUI
12-Apr	space dust	sound effects & wrapping
12-Apr	campus cal.	pers'lized email, visualization & initial pers'lization
15-Apr	vulcan	graphical game display
16-Apr	logic studio	basic timing in evaluator, graphic input usable
16-Apr	pacman 2000	game & main classes, 2nd ghost, keyboard input
17-Apr	direct chat	dynamic IP server complete

Are you there?

- How do you know?
  - A good milestone is concrete
  - A better one is demonstrable
- Be wary of undue optimism

If you're not done, why not?

- Harder than estimated
- More distractions
- Unplanned dependencies had to be done first
- Got behind early and stayed behind

What's the plan, now?

- Work harder….
  (popular but rarely works)
- Doesn't matter, you built in slack
- Rebalance work within the team
- Change definition of what's to be delivered

Milestones next week

DATE	TEAM	MILESTONE
19-Apr	clipbook	export to html & first version GUI complete
21-Apr	direct chat	test dynamic IP addresses
23-Apr	logic studio	timing section debugged
23-Apr	pacman 2000	ready for integration
23-Apr	Seige	test for components complete
23-Apr	trading system	market condition testing complete
24-Apr	redemption	collision dection, most of combat
25-Apr	sound images	prototype works perfectly w/ real data
25-Apr	PIM	simple C/S interaction complete & tested
26-Apr	campus cal.	multiple keyword & pattern search
30-Apr	vulcan	networking & multple game/chat
1-May	space dust	project presentable
7-May	At5000	fully functional system

Are you still ok?

- Is it time to start on the extras?
- Did you just barely make this week's deliverables and next week is hopeless
  - Looking out a week or two is often a good mechanism for checking optimism

What do project managers do?

- Nominally
  - Plan
  - Organize
  - Monitor
  - Control
- Really
  - Communicate, communicate, communicate
  - Remove excuses for failure

Planning

- Project definition
  - Negotiate features with customers
  - Ascertain doability & any assets available
  - Understand risk profile
- Staffing and other needs
- Rough cut architecture & schedule
  - Unless you're careful, this schedule will be ***the*** schedule

Feature definition

- Minimally
  - High level list of features
  - Rough functional spec for the project
  - Rough priority list for features
- Complex projects need requirements that are trackable
- All projects need unambiguous feature definitions

Doability

- How novel is the project, generally?
- Have you or your team done a similar project before?
- How much of the project can you reuse/purchase?
- How good are your resources (people, computing facilities, tools, office space etc.)?

Risk profile

- What happens if you fail?
- What constitutes failure?
  - Schedule?
  - Cost?
  - Features?
  - Which one(s)?
- Will this project get to commandeer resources or be a source for others?

Staffing & resources

- Who (individuals or expertise) do you need?
  - Requirements, development, test
- Are the right people on staff already?
- Are the skills you need hard to find?
- Do you have the right & enough computing facilities including development and test tools?

Initial schedule

- Often, the schedule is dictated
- Until you really *know* what you're doing try not to commit to dates
  - Intervals are often acceptable at this stage
- Always qualify any schedule to indicate confidence

Thoughts on schedules

- Schedule, cost, features. Pick any two.
  - Most projects are constrained on all three.
  - Most projects fail…..
- Don't kid yourself
- Consider 2 sets of books, padding the external schedule by at least 20%
- Learn how to communicate constraints effectively

Organize

- Get the right people and tools
- Assign tasks to teams/individuals
- Establish reporting structure
  - Who tracks progress
  - How often
  - What mechanism (meetings, email, paper)
- Establish other project processes

Team structures

- Individuals almost always specialize
  - By kind of work (requirements, test etc.)
  - By functional area (UI, DB, various application domains etc.)
- Teams are composed of individuals
- Teams are charged with some goal

Kinds of teams

- Functional teams are organized by their specialties and tend to persist
- Feature teams form and reform to accomplish a specific task
- A hybrid approach is feature teams matrixed from functional team members

Functional teams

- Do best when the project area is very complex and requires a high degree of personal depth
  - Tend to be used by more mature projects
- Can result in people aligning with their specialty instead with the overall goals

Feature teams

- Do best in responding to rapid change
  - Tend to be used when things are new
- Can result in team members letting their expertise languish—the needs of the team outweigh the needs of individuals to keep current

Hybrid teams

- Can have either the advantages or disadvantages of both
- They are harder to manage and require a substantial commitment from management and staff to work well
- When they work well, they can be the most effective approach

Minimal project processes

- Feature definitions
- Architecture control
- Change control
- Source control
- Schedule creating and tracking
- Testing
- Trouble tracking

Architecture

- Surprisingly few projects do this well
- It's more than a picture
- Essentials
  - Interfaces, including specification of how data/control will cross interface boundaries
  - Hardware & software components
  - Software guidelines

Change control

- Most essential for identifying new requirements
  - New feature requests
  - Features masquerading as bug reports
  - Features generated by the team itself
- Usually requires some kind of record keeping and periodic review

Change control process

- Must ensure
  - Changes are identified
  - Conscious decisions are made on cost/benefit of the change
  - Impacts of any change are understood and planned for
  - Changes are communicated to those who need to know

Monitor

- How do you know you're on target?
  - Right features being built
  - On schedule
  - On budget
- Pay special attention to risks and high priority features
- Too much and too little are both bad

Schedules

- Measurable milestones
  - Start, original and current end dates
  - Dependencies
  - Owner
- Rule of thumb: Milestones should be no more than 2-weeks apart
  - Internet projects should apply normal "internet time" factor and divide by 4 (i.e. every couple of days)

Intermediate deliverables

- Plan check-points where things are working and you can use the system
- Show what you've got to your customers
- As the project manager, use the system yourself and judge how good is it really

Testing

- Early and often
  - Test progress is the best way to know what you've got and counter optimism
- Test development should begin no later than when development starts
- Weekly builds when more than 2 months from delivery, daily thereafter

Meetings….

- A necessary evil
- Remember both parts
  - Necessary: Without them it's too hard to know what's going on or to communicate to the team what they need to know
  - Evil: It's easy to let meetings drone on and accomplish little

Effective meetings

- Start on time
- Have an agenda with a strict end-time
- Have an owner who drives the meeting to achieve its objectives within the allotted time
- Have minutes published within a day and disseminated widely

Reasonable meetings

- Monthly all-hands meetings to discuss project goals and high-level status
- Weekly team meetings to review milestones & open issues
- Weekly management meetings to track overall progress & issues
- Periodic meetings with non-development partners

Daily status meetings

- Only when the project nears completion
- Objective to identify any issues that may delay completion
- To be effective:
  - Issues must close quickly
  - Must be short (15 - 45 minutes)
- Attendance is a good indicator of whether the meetings are working

Control

- Stuff happens
- Control is what you do about it

Effective control

- Depends on
  - Risk analysis/priorities defined during planning
  - Early warning given from monitoring systems
- Waiting to act usually makes the solution worse, not better

Options

- Add resources
  - Often, adding resources to a late project makes it later, but sometimes this works
- Redefine the deliverable
  - Drop features
  - Relax schedule constraint
  - Reduce quality (this one usually just delays the problem)

Wrap up

- Good project management is hard
- The essence:
  - Know what's important
  - Know what's happening
  - Do something about it