Project Management 101

Dr. Julie Greensmith

Things you might need to know...

- (Today) What project management is (very briefly!)
- The seven sins of project management
- (Tomorrow) How to manage project time and goals effectively
  - Gantt Charts and Milestone Charts
- Tips and tricks for the management of geeks


www.bbc.co.uk/comedy

Good management comes from good leadership

- Its a totally distinct and complex art form, being a good project manager
  - often play a really key role in software engineering
- Leadership qualities define the competence of the project manager
  - "Leaders do things first"
  - "Leaders have followers"
- The best project managers respect their team and are not afraid to lead by example

What is Project Management?

- A method for organising tasks within a team
- A structured framework to help a group work productively
- Tools to aid in the various tasks required to successfully manage a team of people:
  - task sequencing
  - dependency analysis
  - resource allocation
  - scheduling
- Computational tools to track progress relative to plan

Why do we need project management?

- Complex project needs a lot of coordination
- The management of people
  - requires emotional intelligence and understanding of personality and skills of individuals
- The management of resources
  - equipment, procurement, budget and expenditure
- Creation of order in the management of features and tasks
  - critical dependencies need to be managed effectively
  - Multiple decision points – approvals
  - Phased expenditure of funds
  - Matching of people/resources to tasks

Dependencies and the Critical Path

- Certain tasks within a project need scheduling
  - Sometimes task B cannot be started before task A is completed
- Other types of constraints need to be taken into account
  - availability of certain people, development lags
- Critical path – any slippage slips whole project
- Helpful to know what tasks are on the critical path
- Useful to try to shorten the critical path

Initial steps

- Generate a formal definition of the project, with goals, constraints, assumptions
- Identify project start/end dates, any mandatory milestones, including reports, signoffs, deliverables, etc.
- Be aware and factor in any constraints
  - money, equipment availability, holidays, etc.
- Identify tasks to be accomplished via macro and micromanagement
  - high level (i.e., by categories), then details within each what needs to be done

Task Refinement

- Refine detailed task list, dropping/ combining, adding things omitted
- For each task in list:
  - Estimate time (person hours, calendar period)
  - Identify dependencies among tasks
  - Identify resources (people, money, parts, etc.)

Getting cracking

- Organise task groups roughly by starting date
- List dependencies that should or MUST hold
- Use MS Project/PM software to make a chart to show dependencies and to track progress to make a GANTT chart
- Identify milestones from tasks and task grouping
- Identify critical path, see if it can be shortened (get more "slack")
- Assign person-hours and specific team member(s) to each task – identify *"task leads"*

Monitor your progress carefully

- As project progresses:
- Monitor, record progress on all tasks
  - modify your charts to show progress and adjust for resources
- Pay particular attention to those on critical path
- Revise plan as needed to take into account changes, adapt to meet milestones
- *Remember: a task expands to fill the time you give it*

The Seven Sins

of Software Project Management

The Seven Sins

- Taken from Agile Coach Mike Cohn of Mountain Goat Software
- The sins are as follows:
  - Gluttony
  - Lust
  - Sloth
  - Opaqueness
  - Pride
  - Wastefulness
  - Myopia

GLUTTONY

Number 1: Gluttony

- Project management sin of excess
- Definition:
  - Fixing all dimensions (scope, schedule, resources, and quality) of a project
- Experienced as:
  - Impossible schedules
  - "Death marches" and over the top micro management
- Leads to:
  - Trying to do too much for the resources (time, people) available
  - Cutting quality to meet other goals

Lust

Number 2: Lust

- Definition:
  - Intense or unrestrained craving for features
- Experienced as:
  - Trying to put too many features into a product during the time allowed
  - Treating all features as “critical”, need to prioritise!
- Leads to:
  - overtime, reduced quality, surprises and unexpected problems

Avoid Feature Lust!

- Keep realistic with what you can achieve
  - Develop features in priority order
  - Use agile to get instant gratification
  - Keep a check on the amount of "overtime" you have
  - Kent Beck (Agile Guy) states:
    - "Overtime is a symptom of a serious problem on the project. The XP rule is simple–you can't work a second week of overtime. For one week, fine, crank and put in some extra hours. If you come in on Monday and say 'To meet our goals, we'll have to work late again,' then you already have a problem that can't be solved by working more hours."

Sloth

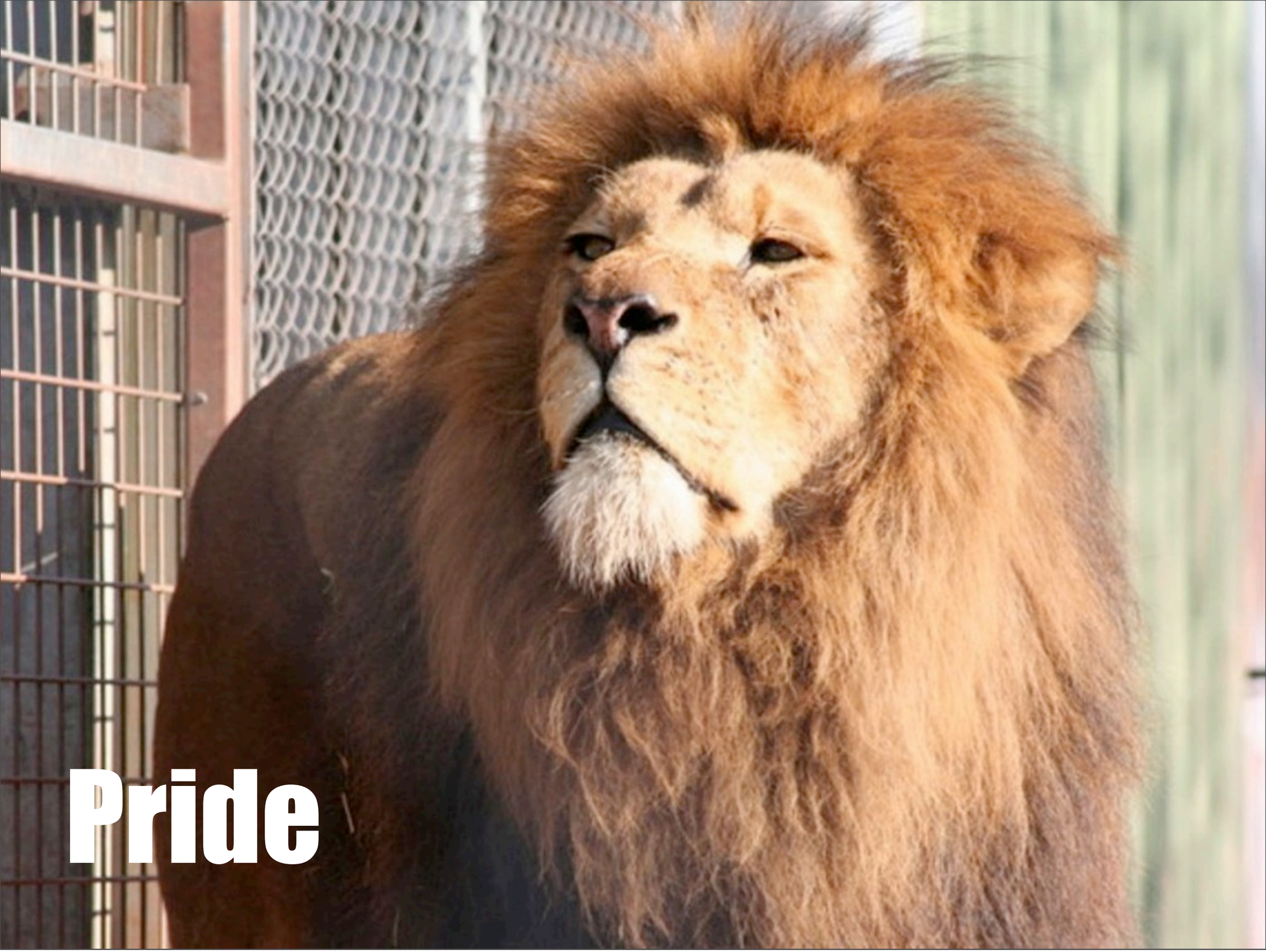
Number 3: Sloth

- People can be really lazy!! You have to fight it.
- Definition:
  - Failing to do high quality work at all times
- Experienced as:
  - Testing quality in at the end
  - Instability during development
- Leads to:
  - Big delays
  - Unpredictable schedules and cost overruns


Opaqueness

Number 4: Opaqueness

- Having a lot of bulls*&t
  - kidding yourself that the project is running ok when in fact you are totally behind schedule
- Definition:
  - Obscuring the progress, quality or other attribute of a project
- Experienced as:
  - Not knowing the true state of the project
- Leads to:
  - Surprises
  - Poor decisions


Pride

Number 5: Pride

- This is the arrogance of the manager and the developer - you can always learn new stuff!!
- Definition:
  - Believing that we know everything to build the product
- Experienced as:
  - A lack of stakeholder and user involvement
- Leads to:
  - Failure to solicit feedback
  - Failure to learn which is unacceptable in the technology sector

Wastefulness

Number 6: Wastefulness

- Definition:
  - Misuse of critical resources
- Experienced as:
  - Losses of creativity, motivation, and time
- Leads to:
  - Project malaise and developer apathy (the biggest enemy on a project)
  - Delays and potentially costly overruns
  - Not effectively reusing code or existing libraries

Myopia

Number 7: Myopia

- This means "shortsightedness" or "project blindness"
- Definition:
  - Not seeing beyond your own work
  - Affects both managers and developers, but especially new graduates!!
- Experienced as:
  - Teams who don't see the big picture
  - Individuals who work only within their roles
- Leads to:
  - Unsuccessful products
  - Delays and overruns (again)

What to do?

- Be aware of the sins
  - try to avoid them
- Get to know your team, share some empathy!
  - this builds trust and avoids opaqueness
  - keep motivation high to avoid apathy
- Don’t be afraid to rework a plan
- Keep track of progress using charting techniques
  - we will go through these and some practical examples tomorrow

Geek Management

Specialist needs of the software developer

Tips for Geek Management

- As IT and CS is central to most industries it pays to keep geeks happy in your workforce
- I'm not saying that geeks have special needs, but they have "specialist needs" that a good manager will be able to address
  - have a particular work ethic
  - have a different style of working
  - highly contrasted to managing people in a sales team

Dealing with geek personalities

- Value their training
- Give them credit where it is due
  - be aware what that nasty two line function does and reward them appropriately
- Don’t assume that they don’t have a life away from the computer: minimise overtime as burnout is getting increasingly common
- Be inclusive to discourage the ‘lone coder’ attitude
- Not all geeks are equal!

Using language carefully

- Avoid jargon filled management speak at all costs:
  - singing from the same hymn sheet
  - providing ballpark figures
  - “technical debt”??????
- Be consistent with instructions
  - documentation cant say one thing but you mean something else
  - Dont preach to the geek. Or try to appear smarter. Ever.

Do not undermine the geek!

- Include them in IT and infrastructure related issues
  - the decision of which technologies and equipment to procure, use and develop with
- Give them the tools they need
  - be prepared to spend the money on better computers than you would roll out across a standard workforce
  - this means paying for their software requirements too
  - Give them the space, and flexibility to solve their problems

Respecting creativity

The Personal Touch

- Muck in with the geeks
  - agile really lends itself to this
  - they will respect you in turn for it
- Get to know your workforce
  - thats the same as with any management role
- If you are a geek managing other geeks don't rub it in that you are now the junior boss!

Understanding perfectionism

- Geeks can get feature lust
  - an over-enthusiasm to build in as many features
- Help to recognise when something is “done”
  - have structured goals and defined milestones to stop projects dragging on
- Clear communication regarding expectations can eliminate most of these problems
- Beware of procrastination based perfectionism!