macadamian

Managing Small Projects

My  worth

Stephane Lussier

January 2005

Software experts to the world's leading technology companies.

macadamian

Agenda

- Who am I?
- What is a short project?
- Managing people
  - Aspects and role of the manager
    - Interviewing
    - Hiring mistakes I made
    - Know your developers
  - Meetings
  - The Feedback Loop
  - Management Patterns: People
- Managing time and tasks
  - Management Patterns: Tasks and time
  - What's different about short projects
  - Mistakes I made
- Processes and tools
  - Common tools
  - Development process
- Questions

Software experts to the world's leading technology companies.

macadamian

Who Am I?

- I've been a Development Manager for Macadamian Technologies for seven years.
- I worked in the Telecom industry for more than five years.
- My technical background is in computer science.
- I've managed more than 35 projects in the last four years.
  - Project sizes range from one developer for one month to 15 developers for 18 months
- I've been involved in about 50 projects at Macadamian.

macadamian

Terminology

- Customers:
  - The people who will use the project and define the requirement.
  - Clients, upper management, other divisions.
- Developers:
  - The people you're managing.

macadamian

What is a short project ?

- No strict definition. To set a range:
  - One developer for one month to five developers for six months
- Characteristics:
  - Time-oriented
  - Opportunistic
  - End goal well-defined
  - All requirements may not be well-defined
  - Originate from customer request
  - Extension of an existing product/system
  - Tools/utilities

macadamian

Managing people

- Don't forget:
  - A quality team is your **#1 asset** in your project**.**
  - The project manager is the key person to make your project a success.
- What I'm planning to cover:
  - The project manager
  - The team
  - Meetings
  - Feedback loop
  - Patterns

macadamian

A good project manager is:

- Motivated in his career
- Determined to run successful projects
- Responsive
- Interested in other people's ideas and concerns
- Able to face the customer, has people skills
- Fair, and can wear both the good and bad cop hat
- A fast-learner and induces confidence from others
- Able to understand the people the team works with (Customers, developers)
- Technically savvy

Software experts to the world's leading technology companies.

macadamian

Finding a manager for a short project

- If the project is too small to afford a manager full time, find a person who will fit a developer/manager role.
- Short projects are ideal for grooming new project managers.
- Avoid giving a manager more than three small projects. You will loose productivity.
- Look for somebody technical, who can be the architect.
- Look for someone with the qualities of a good manager, but not the experience.
- Don't be afraid to put somebody less experienced if he has the right qualities.

Software experts to the world's leading technology companies.

macadamian

The role of a project manager is to:

- Make sure the wheels are always turning.
- Respond to the needs of others.
- Follow up with the team.
- Set clear goals.
- Maintain communication channels.
- Maintain the customer relationship.
- Reward/reprimand teammates.
- Resolve difficult problems.
- Help teammates on the critical path.

macadamian

Build your team

- Look for (in this order):
  1. Intelligence
  2. Motivation
  3. Experience
- Ask yourself if the person is in the top 20% for the position. Set the bar high.
- Don't rush when hiring: Wait for the right candidate. Don't hire the best of the worst.
- Think beyond the project timeframe.
- Good developer: Productivity = +10 (Adds to the productivity of himself and others)
- Bad developer: Productivity = -10 (Wastes time of others on the team)
- Put an interview process in place.
- Avoid people with big ego, look for team player.

macadamian

Interview process (Example)

- Phone screening candidates.
- Three interviewers, thirty minutes each.
- Each interviewer needs to ask his set of questions to be able to make a GO or NO GO decision.
- You need three GOs to hire the candidate.
- Try using always the same three people.
- Use interviewers that have the qualities you're looking for.
- If somebody thinks it's a "Maybe", this is normally a sign not to hire.
- If the first interviewer says NO GO, stop there.
- If you get three GOs, do a reference check.
- Make an offer with a short period of decision time.

macadamian

Looking for intelligence (Interview)

- Is the candidate a fast learner?
- Does the candidate understand the concepts *behind* the technologies he worked with?
- Ask a few problem solving questions.
- Can he explain you in a few minutes something you don't know. (Clarify, summarize)
- Test his general knowledge: Ask questions about technologies not on his resume.
- Ask questions to drill-down on specific topics: Don't permit surface answers and assume he knows the details.

macadamian

Looking for motivation

- What's his ideal job?
- What kind of company would he start?
- Listen to candidate voice tone. Is he enthusiasm?
- Why does he want to join your team?
- Ask him to talk about the best project he worked on. Was he passionate about it?
- Do you think the candidate will be happy in the position you're offering?

macadamian

Looking for experience

- This is the easy part.
- Don't be afraid to ask technical question.
- Don't provide the answer yourself: Go to your next question.
- Once you got the answer, feel free to interrupt the interviewee and ask your next question.
- Don't go through the whole list of previous work experience, you're losing your time. Pick one, maybe two, talk about technologies.
- Put the interviewee in some work scenario, and ask him what he would do.

macadamian

Hiring mistakes (the ones I did)

- Hiring an experienced developer with good technical skills but not motivated.
- Assuming the candidate is a good developer with some language problem when you didn't completely understand the answers. Don't give the benefit of the doubt.
- Taking the best of a bad lot without asking if he reaches the standard.
- Hiring on a rush.
- Only asking questions about related experience. Not asking enough technical questions.

macadamian

Know your developers

- **Cowboys**
  - Problem solvers.
  - Prefer flexibility, common sense over processes.
  - Focus on solving one problem at a time.
  - Look for shorter path.
  - Aggressive in their estimates.
  - Watch out for:
    - Hacks in the code.
    - Unit testing.
    - Big code changes near a release.
- **Perfectionists**
  - Looking for the perfect solution.
  - Care about details.
  - Enjoy re-factoring code.
  - Conservative in their estimates.
  - Enjoy processes/structures in place.
  - Good for writing documentation.
  - Watch out for:
    - Over-design.
    - Lack of feedback.

For more interviewing information

- Please visit Macadamian.com and subscribe to our (Free) (Non-spam) email newsletter, The Critical Path. (It's short.)

Meetings

- Good reasons to have meetings:
  - Exchange information. Chance to ask questions.
  - Review work, documents.
  - Brainstorm on a topic.
- Bad reasons to have meetings:
  - Resolve problems (with 4+ persons).
  - Do round table status.
  - Anything where e-mail would have been more efficient.

macadamian

Three phases of a good meeting

- Before
  - Goal of meeting should be defined.
  - Identify the chair for the meeting.
  - Develop agenda.
  - Distribute agenda.
  - Circulate background material prior to meeting so members will be prepared.
  - All attendees need to come prepared.
  - Set the time limit and stick to it.
- During
  - Start on time, end on time.
  - Review agenda.
  - Stick to the agenda.
  - Encourage group discussion to get all viewpoints and ideas.
  - Keep the conversation on topic.
  - Summarize the discussion and draw out action items.
- After
  - Prepare and distribute the minutes within 24 hours.
  - Clearly identify action items.

Software experts to the world's leading technology companies.

macadamian

Feedback loop

- Responsibility for everybody to share information.
- Respond quickly.
- Good news should fly as fast as bad news.
- Praise/reprimand actions, behaviors (not people).
- Be proactive.
- Don't drop any balls.
- Be transparent.

Software experts to the world's leading technology companies.

macadamian

Management Patterns: Part I
People

1. A developer finds his job **boring**, he doesn't like what he's doing.
   - This is a serious issue, if you don't address it at a certain point, you will loose this person.
   - Could you switch the different tasks assign to the different developers and keep the same level of productivity?
   - Understand what would be the ideal scenario.
   - Explain the importance of the contribution.
   - The project is short, the situation won't be forever.
   - We can't always what do we want and be successful.

macadamian

Management Patterns: Part I
People (Continued)

2. A developer is **under achieving**.
   - Don't let this drag on. Address it.
   - This requires a face-to-face meeting.
   - Describe clearly your expectations.
   - Explain the current performance is not meeting your expectation.
   - Understand the cause of the problem
     - Your perception is wrong.
     - There is a motivation problem.
     - The person doesn't have the qualification/talent/potential.
     - This person has personal issues.
   - Take action (based on the cause of the problem).

macadamian

Management Patterns: Part I
People (Continued)

3. A developer is **not proactive**, doesn't live by the feedback loop.
   - This is something that can be learned.
   - Day-to-day mentoring on how to keep up the feedback loop.
   - Follow-up with this person. Explain to the person why you're doing more follow-up.
   - Assign smaller tasks to this person.
   - If there is no progress, see "Developer is underachieving" pattern.

macadamian

Management Patterns: Part I
People (Continued)

4. A customer is not responsive
   - Always keep the customer in the loop (even if he doesn't seem to care)
   - Try different communication channels.
   - Be persistent, but stay cordial.
   - Try scheduling a meeting to discuss the responsiveness issue (Not the other project issues).
   - Is the customer too busy with higher priority tasks?
     - Keep this person in the loop, but try to reduce the dependency on that person.
     - Ask if the customer can delegate somebody else to manage the issue.
     - Propose, instead of asking. Take leadership into your own hands.
   - Expose the impact of the non-responsiveness in measurable terms (money, time).
   - Track pending issues that need to be addressed.

Software experts to the world's leading technology companies.
© 2005
Macadamian Technologies

macadamian

Management Patterns: Part I
People (Continued)

5. A customer **blames you** for a problem
   - Cool down before taking any action.
   - Ask yourself if the customer is right. Ask someone for a second opinion.
   - If the customer is right:
     - Admit your mistake.
     - Explain the root cause of the problem.
     - Explain how you'll make sure it won't happen again.
   - If the customer is wrong:
     - Don't counter-attack. There's a good chance, it will get worse.
     - Make your point and ask for more explanation.
     - Make your point a second time.
     - Move on. In the interest of the project, look ahead.

macadamian

Management Patterns: Part I
People (Continued)

6. Some developers have a mentality of us (the team) against them (the customers)
   - You need to understand the point of view of both the customer and the developer.
   - Explain the motivation behind customer action/decision.
   - Promote the fact that everybody is part of the solution.
   - The customer is always right: He has the final word.
   - Be transparent, regularly share information about the customers and their decisions.

Software experts to the world's leading technology companies.
© 2005
Macadamian Technologies

macadamian

Managing time and tasks

- Gather requirements
- Estimate
- Assign tasks
- Design
- Set milestones
- Make decision
- Track
- Test
- Deal with the overtime issue
- Patterns

macadamian

Gather requirements

- Always meet with the customer, even if you got the requirements on paper.
  - Come prepared with a list of questions.
  - There's no stupid question. Ask everything.
- Don't let the future implementation drive the requirements. Think as a end-user.
- Gather information about the expected timeframe.
  - When does it need to be ready? Be clear about expectations.
- Ask about quality.
  - Carefully define terminology: "prototype," "alpha," "beta," etc.
- Define who will be the first users of the product.
- Identify unclear requirements. Develop them.
- Write down requirements.

macadamian

Estimate

- First, investigate requirements/features for which feasibility is not a sure thing.
- For short project, use a task breakdown approach.
  - Keep every task below five person-days.
  - Tasks don't equal features.
  - There are some unknown tasks, tasks that won't be listed.
  - Use a spreadsheet.
- Consult the developer who will do the task for his estimate.
- Identify assumptions, limitations.
- Regroup tasks per milestones.
- Associate time to tasks
  - Be conservative. Don't assume best guy will do it.
  - Don't try to reach a specific target date.
  - Wait thirty minutes, then do a second pass of time estimation.
- Have you allotted time for testing, documentation, management, learning curve, vacations, install program, deployment, bug fixing, build infrastructure?
- Ask somebody to review the estimates.
  - Give him/her the task breakdown without the time estimation.

macadamian

Assign tasks

- Assign tasks for the current milestone only.
  - Finish a milestone as a team before tackling the next one.
- Lower the risk to your project by assigning riskier tasks to your best developer.
- Happy developers are more productive: Try to assign tasks based on the developer interest.
- A short project with only two mature developers: Don't assign the tasks, let them do it.
- Set weekly objectives on Monday.
- Make sure no one runs out of tasks.

macadamian

Milestones

- Divide your project into milestones of 3-6 weeks.
  - Longer milestone, focus is lost.
  - Shorter milestone, it doesn't look like there's any milestone, you don't have the feeling you've accomplished something substantial at the end of a milestone.
- Clearly identify what are the deliverables of the Milestone.
- Clearly identify what is the target date for the end of the Milestone.
  - Set a realistic date.
- Flexible Milestone. (short project)
  - Before starting a new Milestone, adjust it (time, features) according to the knowledge you've cumulated since the beginning of the project.
- Plan some time at each Milestone to fix issues discover during the previous Milestone, don't wait the end of the project.

macadamian

Design (for short projects)

- First a few questions:
  - Should you do the big design up front?
  - How much effort should be put into it?
- My approach:
  - Design/document the output/input of your project up front (UI, API, etc.).
  - Don't do a detailed design up front.
  - Do a "maintenance" document near the completion of a project.

macadamian

Making decisions

- Don't wait until you're 100% sure you're making the right decision.
- If you're facing three or more choices, process by elimination until you get only two choices.
- When you're confidence level is >80%, make the decision.
  - You'll be right 9 times out of 10.
  - Understand that you will make errors. Listen for feedback, indicators.
  - The project will move faster if in the same time you take ten decisions with 90% confidence level than five decisions with 100% confidence level.
- Listen, ask questions, listen, make decisions, explain motivation, move on.
- Delegate some decisions: Decisions should be made by the most knowledgeable person.

macadamian

Tracking project

- Keep it simple, this is a short project.
- Follow up with developers every day.
  - By following up, you aren't playing the police: You're following up to understand the state of the project, so you could make the best decisions.
- Look for developers who are:
  - Blocked.
  - Going in the wrong direction.
  - Out of tasks.
- You need to share the project status with the developers and customers (Feedback loop).

macadamian

Testing

- Developers need to do the unit testing.
- Measure the costs vs. the benefits of doing automated testing or developing a test suite.
- You need a test plan that includes your test scenario. Don't make an exception for short project.
- Let the customer review the test plan.
- QC people are part of the team, don't make a distinction when managing.
- Test for usability.
- Assign other tasks to QC: Program set-up, build system, etc.

macadamian

Overtime

- Plan for **no overtime**.
- Re-adjust your plan for **no overtime**.
- Ask for 40 hours of work, fully focused.
- Sometimes you have to do overtime:
  - Picture this: You have 3 overtime tokens per year. An overtime token is good for 2 weeks once you cash it in. Spend your token wisely.
  - You have to do overtime yourself.

macadamian

Management Patterns: Part II
Tasks and time

1. The customer is changing the requirements faster than you can change your underwear.
   - Be flexible. Consider the new requirements from an end-user perspective.
   - Don't say yes to everything right away. Take the time to have an honest look at it.
   - Expose the impact of the new requirements to the customer.
     - Impact on the schedule
     - Impact on other features
   - Put down the new requirements on paper in your own words. Resubmit it to the customer to confirm that's what he wants.

macadamian

Management Patterns: Part II
Tasks and time (Continued)

2. A developer has some problem estimating the work he has to do.
   - Insist on getting an estimate, but don't insist on having an estimate that is accurate.
     - The developer will learn from this exercise.
     - You will understand better if the developer is aggressive or conservative in his estimates.
   - Explain to him how you would do it. Use a granular task break down.
   - Review the estimate with him.
   - Ask developers to estimate frequently. People get better at estimates with experience.

macadamian

Management patterns part II
Tasks and time (Continued)

3. The project is late. You won't meet the milestone deadline.
   - Is there a solution?
     - Can you add additional resources?
     - Is it time to use the overtime token?
   - Prepare yourself to tell the customer.
     - When will you hit the milestone?
     - What are the impact to future milestones?
     - What are the reasons?
     - Don't wait until you are at 100% sure you will miss the milestone.
   - Announce it to the customer and team.
     - Don't use e-mail.
     - Discuss a solution.
   - What can you learn from the situation?

macadamian

Management Patterns: Part II
Tasks and time (Continued)

4. The customer imposes an unrealistic delivery date and the feature set.
   - Take a deep breath.
   - You need to explain your point of view to the customer.
     - Show when you will be ready.
     - Show what you will have done for the target date identified by the customer.
     - Identify features that may be cut.
     - Put the tasks/features in priority order. Address them in this order.
     - Assure them you will do your best to surpass what you think is realistic.
   - Give regular feedback about the milestone situation. The customer shouldn't have to ask for it.

macadamian

Management patterns part II
Tasks and time (Continued)

5. One of your developers is reassigned.
   - Understand the reason for it. It's probably for the good of the company, not for your project.
   - Expose the impact to the team and the customer.
   - If it's temporary, then establish how long this situation will last.
   - Re-assign and prioritize tasks.

macadamian

Management patterns part II
Tasks and time (Continued)

6. You don't have enough time to manage the project.
   - You need to connect developers with customers.
   - Focus on the problems in the project.
   - Follow up at least every week.
   - Stay available for the emergency issues.
   - Explain the situation to the developer and delegate some responsibilities.
     - Explain your expectations clearly to the developer.
     - Stay in the loop.
   - Make sure somebody in the team stays responsive to the customer.

macadamian

Management Patterns: Part II
Tasks and time (Continued)

7. Project is so small it only requires one developer.
    - Keep your management role to the minimum. This could result in almost nothing to do.
        - Push some management tasks to the developer.
        - The developer should be in direct contact with the customer.
        - Lightweight process.
        - Keep good communication with the customer.
        - Stay in the loop.
    - Good opportunity to groom future project leader.
    - Choose your developer wisely.

macadamian

Managing short projects. What's different?

- Be flexible on the requirements. Allow requirements to change during the project.
- Keep processes lightweight.
- Avoid designing the whole project up front.
    - Do some functional design (project as a black box) up front.
    - Write a maintenance document at the end.
- Give your developer more room (Design, relationship with the customer, etc.).
- There's a good chance you will do more than managing the project.

macadamian

Mistakes I made managing software projects

- Not enough follow-up.
- Avoided confronting ideas of developers and customers.
- Not being responsive to difficult requests.
- Didn't explain my expectations clearly enough.
- Had difficulty saying no to requests.
- Not putting enough effort on the testing.

macadamian

Tools and processes

- Essential tools
- Development process
- Communication with the customer
- New process

macadamian

Common tools

- Essential tools
  - Source control
  - Bug tracking system
  - E-mail, Phone, IM
  - Debugger
  - Space to share documents for the team.
- Helpful tools
  - Mailing list
  - Templates for test plan, UI design, detailed design, Maintenance doc, etc.

macadamian

Development process

- All code is reviewed before going into the source control.
  - Developer submits code for review to a peer.
  - Aim for one code submission a day.
  - Keep the code submission small.
  - Submit complete code.
  - Submit test cases with the code.
  - Reviews have an higher priority than code.
  - Keep code review in crunch time.
- Don't only review code, review also documents.
- Avoid fixing other people's errors. Let the author fix it.

macadamian

Communication with the customer

- Weekly status e-mail.
  - Pending issues
  - Work accomplished this week
  - Work planned for next week
- A weekly meeting with the customer.
- Initiate all the discussions between the developers and the customers. Then act as an observer.
- Ask to be CC on all e-mails exchanged between customers and the developers.
- Understand when you should use phone, e-mail or IM.

macadamian

Introducing new process

- Be flexible. Processes should change over time.
- Easy part is to create a new process, tough part is to apply it.
- Try to estimate the costs vs. the benefits.
- Always explain the motivation behind the new process.
- New processes are like new code, they need testing and debugging.
- Measure the costs and benefits of the processes after it has been in place a while.
- The document describing the process should be easily accessible.

macadamian

Questions?

- Hopefully I'll have some decent answers. ☺

macadamian

References

- Running effective meetings: http://www.csusm.edu/srl/fast_tips/running_effective.htm
- The One Minute Manager
  Kenneth Blanchard, Spencer Johnson
- The Guerilla Guide to Interviewing by Joel Spolsky http://www.joelonsoftware.com/articles/fog0000000073.html
- Debugging The Development Process
  Steve McGuire, Microsoft Press
- The Critical Path: Subscribe at www.macadamian.com. Or just read the archives.