Fixing a Dysfunctional Project Team
White Paper

Date:	Jul 18, 2010	
Title:	Fixing a Dysfunctional Project Team	
Code:	WSDT	
Abstract:	Sink or swim! Taking over a dysfunctional project team can be a very frustrating, yet exhilarating experience. This article summarizes some "rules of the road" that should help you navigate through shark-infested waters with the result being a highly motivated department or team delivering quality projects on time.	
Version:	1.0 (2010-07-17)	Original (excerpt from Leading Software Maniac's *Mastering Agile Software Development Leadership* boot camp)
	2.0 (2010-07-21)	Posted on LSM's web site, updated with more agile content.

Overview

Way back in 1993, I became the executive in charge of development at a software company in Dallas, Texas (where most great software was built!). I was the seventh VP in that position in just over eight years. Needless to say, the environment wasn't very stable. To make matters worse, the day I showed up for work was the day the incumbent VP was told that he was fired.

Seriously???

I walk in on my first day saying, "Howdy!" The fellow I replaced was very upset and stormed out of the building, the department's secretary was crying, and everyone else was in a state of shock. "Great," I could hear voices in the hallway, "here we go with yet another one that will try to tame us!"

What does this situation have to do with *project management*? Plenty! It isn't unusual that project managers get thrown at new projects consisting of burnt out, frustrated software/IT development teams. Whether you are faced with taking over a new project team or a software/IT department, I hope some of these best practices will help you be successful.

What does this situation have to do with *agile*? Plenty as you'll soon see. You can approach a turnaround situation in a very dictatorial, tactical, and controlling manner or you can lead a term destined for the long term by balancing a time-boxed sense of urgency while adjusting as you go.

Background

Welcome to your new nightmare—I mean opportunity! So, you're now in charge of a software development organization—congratulations! Chances are you've earned the right to lead fellow software developers, build a new team, or have been placed in a position to turn around an existing team. Your boss has just introduced you to your new team and has one simple command assignment that ricochets all over the walls in the building:

> "You have 90 days to turn this organization around!"

The Approach

You'll be under lots of pressure to "whip things into shape," yet you need to make sure that your management understands that it takes time and patience to get results that will last a long time. Figure 1 shows the simple steps we'll use.

Although "department" is mentioned throughout this document, you can apply the very same methodology to a "project team."

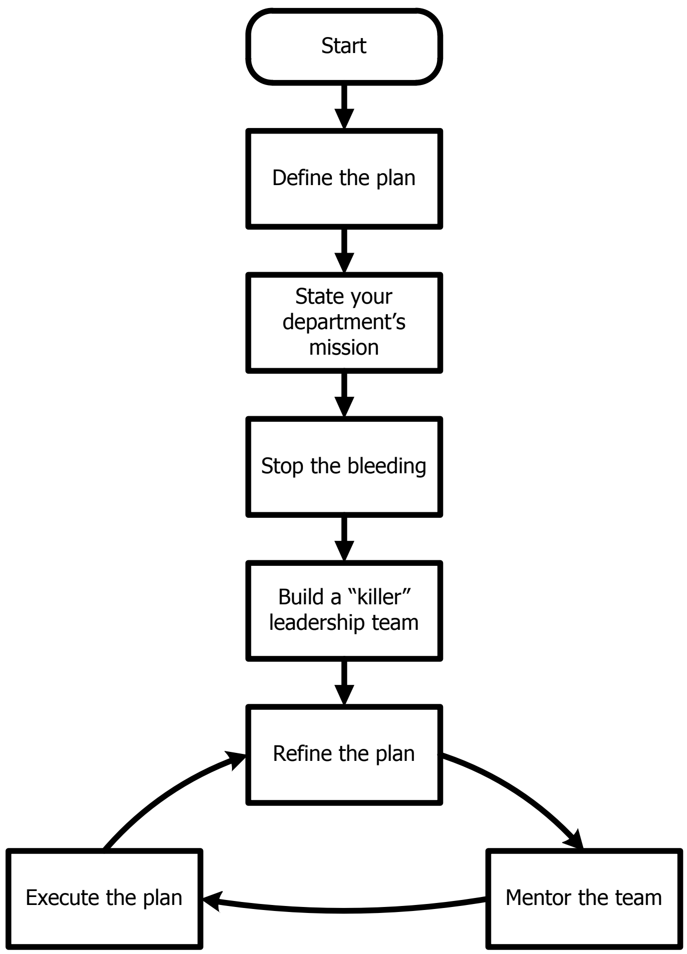

Figure 1: Steps to follow to fix up a dysfunctional organization

Preparing for Success

To lead a successful fix-up attack, there's a few rules of the road that could minimize the risk of failure:

1. Prepare, prepare, and prepare (the more time you spend preparing, the less time you have to drag your jaded team through countless meetings in attempts to come up with solutions).
2. Get buy-in along the way (this doesn't mean forcing decisions or waiting for unanimous consent).
3. Listen, Listen, *communicate*, and listen (and not just to your staff).
4. Never openly dismiss prior management as being failures EVER!!!

Note: As most of us know with agile (specifically Scrum), the more time you spend planning the less time you have to tie up your team in meetings. It isn't unusual that with a project's typical 15 minute Daily Scrum that in order for those meetings to go smoothly, you'll need to spend at least an hour or so planning.

Define the Plan

First step is to lay out a correction plan. It must be reasonable and attainable. Focus on what the department, company, and customer (or the market) needs. If you make it too complicated or detailed, it will become just another failed "salvage attempt."

In other words, keep it SIMPLE!

Communicate progress being made throughout the department's overhaul and always reinforce successes along the way. The more *visual* the communication the better as shown in Figure 2.

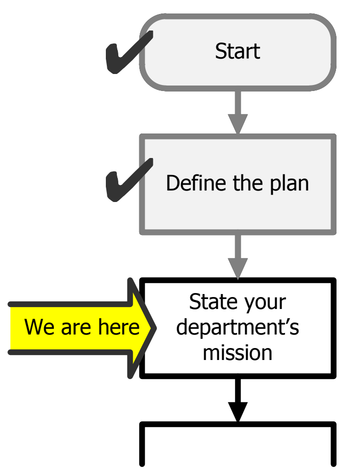

Figure 2: Showing successful completions along the way keeps everybody focused

There's a reason that the *PMBOK® Guide* proclaims that about 90% of a project manager's job should be spent on some form of communication. Agile framework management implies the same level of transparency with consistent and effective communication.

State Your Department's Mission

What's the difference between a mission and a vision? The vision focuses on the future while the mission concentrates on the present (the tactics to accomplish the vision).

Before you start making changes, you'd better get the mission straight. What's involved? A mission basically shows what you are doing and how you're going to do it. Here's what you and your team need to put into your mission statement:

1. Define the department's purpose.
2. State the business or market your organization serves.
3. Clearly identify the objectives that need to be accomplished.

Just as a point of reference, this mission stuff is analogous to what you need to define in an agile project's Project Kickoff (or Scrum Planning) meeting. Get everybody to agree to the mission of the project. Knowing why the project is needed and keeping the mission simple can give you and the team that much-needed "30 second elevator speech" you hear so much about.

How do you know you have a decent mission? Let's try one out!

> "Development's mission is to keep on the leading edge of technology in order to produce software products that our customers want."

Not too shabby! Now let's look at how well this mission compares to our "what makes a great mission statement" checklist:

	Mission should be
✓	Easy to recite
✓	Agreed upon by key stakeholders
✓	Long-lasting
	Gives the customer a sense of comfort
	Motivator

The mission is certainly simple and easy to recite. However, the "leading edge of technology" part may not resonate with customers or your sales organization. Additionally, the mission is simply too ambiguous and ambitious leaving everyone doubting that it could really be achieved. Here's another one mission statement.

> "Development's mission is to deliver on-time, quality enterprise-wide software that provides a sustainable benefit to the healthcare industry."

	Mission should be
✓	Easy to recite
✓	Agreed upon by key stakeholders
✓	Long-lasting
✓	Gives the customer a sense of comfort
✓	Motivator

Easy to recite? Sure, this statement would look good on a plaque.

Agreed upon by key constituents including executives, employees, investors, and customers? Your company's VP of Sales and CFO would be ecstatic with the "on-time" part which usually implies an additional benefit: under budget. This mission could easily be printed in an annual report.

Long-lasting to withstand changing technology and customer needs? Even with your organization's delivery mechanism changing from shrink-wrap to Internet and to Web Services—the same mission holds.

Gives the customer a sense of comfort? Your customers would certainly appreciate the commitment to "quality" which means they will receive a product that works as expected.

Motivator to employees? Couldn't get much simpler—your staff will probably test you to ensure that key decisions you make actually support this mission.

A mission provides the foundation that bonds the organization to a common purpose or objective.

Once everybody that *created* the mission and *buys into* the mission has to realize that roles may change in order to *support* the mission. For example, if "quality delivery" is a key part of the mission and out of a staff of 20 you only have two testers, you're going to have to "anoint" some folks to become temporary QA testers!

Stop the Bleeding

It's time to go into immediate action. Getting organized demonstrates to everyone that the department prefers to operate in a "controlled sense of urgency" rather than an "always-late, ad hoc decision making, and unorganized" mode of operation:

You must quickly establish a sense of urgency. Chances are the dysfunctional nature of the organization you've inherited has evolved over a long time. You don't want to take the same amount of time to undo it.

According to Hammer and Champy's book *Reengineering the Corporation*, change management needs to take place rapidly:

> " . . . stretching it over a long time period extends the discomfort. People will become impatient, confused, and distracted. They will conclude that reengineering is another bogus program and the effort will fall apart."

Some tried and true best practices to stop the bleeding include:

- Getting the organization into a different train of thought (to being focused and successful).
- Distinguish between "must haves" versus "nice to haves" (can you guess which ones aren't worth wasting time on?).
- Establish a culture of listening (you don't have to always be yapping – listening is a very positive form of communication, too).

- Get communicating when you aren't listening … frequently. (Typically the #1 employee survey issue is "lack of effective communication" – so do something about it! But, please don't become meeting bound.)
- Reorganize if required and establish lead roles (even with a temporary leadership team).

Just how productive is your staff?

If you are interested in learning about a total program to handle everyday work flow, read David Allen's *Getting Things Done*.

According to Jonathan Spira of Basex, interrupts consume about 28% of an employee's day. His survey results published in *Time* magazine showed that knowledge workers only spend about 11 minutes on any task before being interrupted. And to top that off, over 55% respond immediately to an incoming e-mail.

And, of course, Tom Demarco and Timothy Lister's superb *Peopleware* summarizes that once interrupted, it can easily take at least 15 minutes to switch your mind back to the job at hand.

Not good! Not good at all – and yet we can all control how we manage our time.

Build a "Killer" Leadership Team

You can't do it all yourself and next up is to create an effective software development organization. There's no better feeling than to have a team that you can rely on to "deliver the goods." Some leaders consistently haul their past cohorts to form their dream team from past lives.

Don't do it!

You might have hidden talent right there in the department that is ready to step up and help lead. Even if put into roles of temporary team leads, select those that are open to change and who have the following skills:

- Have technical domain skills (they'll be credible with the rest of the team).
- Have people skills (very important especially during times of organizational fix-ups).
- Can plan (to help you ensure that decisions being made will move the organization forward).
- Understands simple software development process and willing to trust the team and adapt (the agile way).

Talk one on one to those that are removed from leadership roles or with those not selected for leadership roles. By reinforcing that the mission is what drives organizational change, they may not *like* the new role decisions being made, but they should *buy into* the reasons behind the selections.

Note: Even with small teams (less than 5 people), I usually try to designate a couple of people to have some level of responsibility outside of their normal project work tasks. Having established relations act in leadership roles actually helps "form a bridge" between the existing team and new leadership (you!). This technique can be a very useful *trust builder*.

Continuously Improve Your Organization

Finally, let's look at three steps that will become the cornerstone of everyday life going forward. The three steps presented in Figure 1 actually represent a *continuous improvement cycle* (see Figure 3).

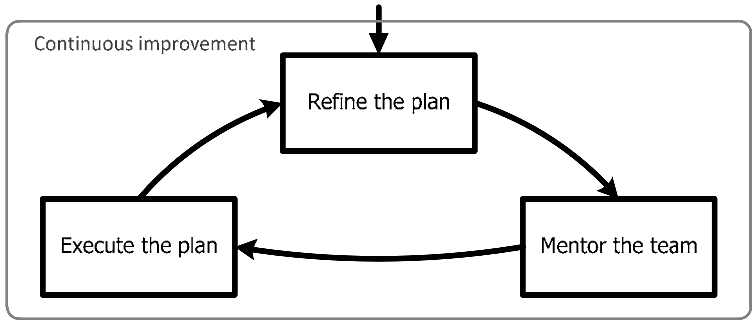

Figure 3: Continuously improving how the department (team) performs

The steps are simple:

1. **Refine the plan:** Adjust the original plan as you learn more about the business, your teams, your products, and your customers. Not tweaking the plan will definitely be an alert to your staff that you are inflexible. Also, you may wish to consider defining metrics for the plan so that you know what success will feel like!
2. **Mentor the team:** Take every opportunity to coach your staff at all levels on how to make better, more timely, well-informed decisions, and on being decisive while getting buy in (this isn't contradictory).
3. **Execute the plan:** This is one of the most overused (and universally hated) of all phrases in the *PMBOK® Guide*:

   "Plan the work then work the plan."

   There is, however, some merit to this catchy jingle. You need to establish credibility by being clear on what you and your team need to do. Then, just do it! Also, don't forget to validate the metrics you planned (step 1 in the continuous improvement cycle) were achieved.

So when does this continuous improvement cycle ever end? *Simple put—it doesn't.*

A Final Word

There are various events that can trigger a dramatic department or team change. The steps outlined in this document have worked for me time and time again. You'll probably find your own formula for success, but whatever you use, please keep the following in mind:

1. Create a plan and get total buy in to the plan with total transparency. Don't be secretive (even you feel like you need to be).
2. Ensure that your mission is clearly understood which will provide the guidance used in decisions.
3. Insist on a culture of listening, communication, productive use of time, and early visible successes. Demonstrate a sense of urgency and time box everything!
4. Surround yourself with positive influencers acting in leadership roles (even if temporary). Pump up the team and don't take credit for your own actions. Your role is to mentor and guide.

5. Finally, set up the organization to continuously improve (even if small, baby steps are taken) – get everyone used to becoming more effective.

Avoid at all costs embracing a *command and control* style of decision making – this is the time to be agile and flexible.

Bibliography

Allen, David. *Getting Things Done: The Art of Stress-Free Productivity*. New York: Penguin Group, 1991.

Demarco, Tom and Timothy Lister. *Peopleware: Productive Projects and Teams, 2nd Edition*. New York: Dorsett House, 1999.

Hammer, Michael and James Champy. *Reengineering the Corporation: A Manifesto for Business Revolution*. New York: HarperCollins Publishers, 1993.

Project Management Institute, Inc. *A Guide to the Project Management Body of Knowledge: PMBOK® Guide, 4th Edition. Newton Square, PA: Project Management Institute, 2008.*

Spira, Jonathan B. "The High Cost of Interruptions," *KMWorld*, Sept. 1, 2005, http://www.kmworld.com/Articles/News/News-Analysis/The-high-cost-of-interruptions-14543.aspx.

Whitaker, Ken. *Principles of Software Development Leadership: Applying Project Management Principles to Agile Software Development.* Boston: Course Technology PTR, 2009.

Bio

Ken Whitaker of Leading Software Maniacs™ (LSM) has more than twenty-five years of software development executive leadership and training experience in a variety of technology roles and industries. He has led commercial software teams at Software Publishing (remember Harvard Graphics?), Data General, embedded systems software companies, and enterprise software suppliers. Ken is an active PMI® member, Project Management Professional (PMP)® certified, a Lewis Institute instructor, and a Certified ScrumMaster (CSM). Sources for LSM's presentations come from case studies, personal leadership experience, the PMI *Project Management Book of Knowledge* (*PMBOK® Guide*), and Ken's two books: *Managing Software Maniacs* and *Principles of Software Development Leadership*.

Leading Software Maniacs is proudly associated with: