About Corporate Politics

Adrian Kosmaczewski

2007-06-03

Politics are part of our daily life. Nevertheless, the word has got a bad reputation in the IT world (and elsewhere, too), thanks to famous failures and managed disasters, but the truth is that to succeed, projects need politics - and project managers should know it well.

In this posting, I will describe the theoretical and empirical direct relationship between organizational verticality, worker alienation and political struggles, all leading to poor project performances.

About the word

DeMarco and Lister talk about the meaning of the word "Politics" in chapter 34 of "Peopleware", named "The Making of Community":

> The science of making communities, making them healthy and satisfying for all, is called politics.(…) Aristotle included Politics among the five interlinked Noble Sciences that together make up Philosophy. The five are: - Metaphysics: (…) - Logic: (…) - Ethics: (…) - Politics: how we may logically extend Ethics to the larger group, the science of creating and managing such groups consistent with ethical behavior and logical recognition of the metaphysical entities - humans and the community made up of humans. - Aesthetics: (…) Aristotelian Politics is the key practice of good management.

(DeMarco and Lister, page 223)

In our 21st century world, having a job means spending at least 8 or 9 hours of your time with lots of people, with whom you may or may not be happy to work with. To work as a team, particularly in knowledge-intensive jobs as in IT teams, the construction of a community is fundamental. This construction is deeply rooted in cultural backgrounds, values, ethic principles, and varies from one country to the other.

But we live in a globalized world, and for both the good and the bad, some inherent aspects of it are present everywhere; some of them are described by Philosophy. Some other are inherent to the capitalist system, and for these,

I would like to point out the term that Marx uses to describe the separation between the worker and his work; "alienation":

> Marx identified four specific ways in which alienation pervades capitalist society. The product of labour: The worker is alienated from the object he produces because it is owned and disposed of by another, the capitalist.(…) The labour process: The second element of alienation Marx identified is a lack of control over the process of production. (…) Our fellow human beings: Thirdly, we are alienated from our fellow human beings. This alienation arises in part because of the antagonisms which inevitably arise from the class structure of society. (…) Our human nature: The fourth element is our alienation from what Marx called our species being. What makes us human is our ability to consciously shape the world around us. However, under capitalism our labour is coerced, forced labour. Work bears no relationship to our personal inclinations or our collective interests.

(Cox, 1998)

The creation of a community means making everyone in a team aware of the project scope, about the possibilities, about the vision and mission; everyone must buy it, and up to a certain level, feel responsible for it.

I think that the IT world is redefining the working rules of the capitalist society as we know it, and much of the problems about IT project management have to do with this particularly fundamental change in the worker/job paradigm.

Successful IT shops reduce alienation and encourage communities; they have, then, these particular characteristics, completely new in the management world:

- At the bottom, they tend to empower, rather than to limit, their teams; team members must feel that they own what they produce, in order to take responsibility for it
- At the top, management must learn to have a higher level of trust on their people; those in the field know things that management will never imagine. A good manager must "act like their most important job is to run around the room, moving the furniture out of the way, so people can concentrate on their work." (Joel Spolsky, March 19th, 2000)
- Agility is part of the equation:

  > We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value: Individuals and interactions over processes and tools Working software over comprehensive documentation Customer collaboration over contract negotiation Responding to change over following a plan That is, while there is value in the items on the right, we value the items on the left more.

(Agile Manifesto, 2001)

Daily Life

There are tons of anecdotes and examples about political conflicts in IT companies; my preferred is the series of articles that Joel Spolsky wrote, about the differences between Microsoft and Juno, in his blog "Joel on Software":

> Without going into too much detail, my manager decided he didn't like this. It became an issue of ego for him. First he yelled at the designer who was working on that page (without even telling me). Then he yelled at me. Then he reminded me every single day that I had to change it to the way he wanted it. Then he got the CEO of the company to review it, and made a big show out of getting the CEO of the company to criticize my new design. Even the CEO at Juno is perfectly happy to interfere in work done at the lowest level in the company, in fact, it's standard operating procedure.

(Spolsky, March 19th, 2000)

> For months later, we would have meetings where people would say things like "Charles [the CEO] doesn't like dropdown list boxes," because of something he had edited without any thought, and that was supposed to end the discussion. You couldn't argue against this fictional Charles because he wasn't there; he didn't participate in the design except for hit and run purposes. Ouch.

(Spolsky, March 23rd, 2000)

But I had also the opportunity of wrestling such "Command and Conquer" mentality in my own work as well.

Some time ago I was working for a big consulting firm in Geneva, and was told by my boss that the commercial teams had trouble figuring out the technical skills set of our consultants. The HR team did not have any database about us, at all, besides tons of paper CVs, which makes searching and filtering hard and inefficient, to say the least. Bid managers and other commercial staff members wanted a searchable database application in the intranet with that information, so they could match the incoming requests to the available skills in the company, and as such make bids and take better decisions. And of course, they wanted this application yesterday.

I proposed a quick solution using the Ruby on Rails framework (http://www.rubyonrails.org/) which I had been successfully using in personal projects; I was confident that such a tool could be done in less than 5 days, providing the needed functionality. My boss, on the other hand, saw this project as the opportunity to show the hearquarters his commitment to the MDA (Model Driven Architecture, http://www.omg.org/mda/) methodology, which was pushed by the company's headquarters, as the standard way of doing things in our company. He made a clear statement about not wanting "toy" technologies in "his" department, and that he would not accept solutions outside of the boundaries of what the

headquarters proposed.

It must be said that it was a tough year for our company, with several projects lost or failed. Time was running out and many members of my team had been fired. Needless to say, the human environment in the company was severely degraded. We all knew in the team that our boss was trying to get promoted, and that some of his wrong decisions were at the origin of some project failures in our team. To sum it up, the communication in our team, particularly between our boss and us, was broken after the layoffs.

Going back to the application, each of the solutions (Rails vs. MDA) had, of course, its long list of pros and cons, but my boss did not comment the CEO about other solutions than his own (I knew this by third people, afterwards). He got the approval and budget to create this application, and came back to me saying, basically, that I had to shut up and launch my UML editor.

This was enough for me to start the project by myself, at home, and in 3 days I had a working application, which I demoed to my peers in the office, and then proceed to set it up in one of our development servers. By the time I had the application up and running, the Java folks were just starting the first UML diagram needed to generate the application. Spontaneously, my teammates started entering their own curriculums in the application the same day (I was surprised) and came back to me with some ideas, bugs and modifications. I added the changes, commented the code out and uploaded everything to the internal version control system. The thing was ready.

Now for the interesting part: I showed the final version of the application to some commercial people, bypassing my boss, and that did it; they started using it almost immediately. They were impressed that we could deliver a solution that fast, and of course they went to my boss to congratulate him, which of course puzzled him. When he saw the thing working in the intranet, he went to the Java / UML team to congratulate them… They knew that I had been behind that one, and told my boss so.

He came back to my desk and asked me to remove the application from the server in that very moment. I refused, so he escalated this insubordination to the upper levels of the hierarchy, and when reaching the CEO, I suppose he was told to accept the thing, since I never heard about his opinion about this application again. The commercial teams started using it that same day.

A couple of months later I left the company. I heard that they are doing other applications with Rails right now, that the curriculums application has been upgraded a couple of times, and that my ex-boss was fired, for other reasons.

Conclusion

The difference between success and failure in software companies has a lot to do with the “verticality” of the structures and the level of “alienation” of their workers; the traditional, military ways of management that lead to higher verticality

and strong worker alienation do not work in the IT world, where knowledge, communication and collaboration are key to success.

I think that the rules of management are changing nowadays, and that this will ultimately turn into a major paradigm shift in our conception of the capitalist system. We need Politics, but as defined by Aristotle, not as Marx's.

References

Beck, Kent, Thomas, Dave, Fowler, Martin et al., "Agile Manifesto", 2001, [Internet] http://agilemanifesto.org/ (Accessed May 28th, 2006)

Cox, Judy, "An Introduction to Marx's Theory of Alienation", July 1998, International Socialism [Internet] http://pubs.socialistreviewindex.org.uk/isj79/cox.htm (Accessed May 28th, 2006)

DeMarco, Tom & Lister, Timothy, "Peopleware - Productive Projects and Teams, 2nd Edition", 1999, Dorset House Publishing, ISBN 0-932633-43-9

Spolsky, Joel, "Command and Conquer and the Herd of Coconuts", Joel on Software weblog, Thursday, March 23, 2000, [Internet] http://joelonsoftware.com/articles/fog0000000072.html (Accessed May 28th, 2006)

Spolsky, Joel, "Two Stories", Joel on Software weblog, Sunday, March 19, 2000 [Internet] http://www.joelonsoftware.com/articles/TwoStories.html (Accessed May 28th, 2006)