The Green Shift Anti-Pattern

Scott W. Ambler

In astronomy there is the concept of red-shifting and blue-shifting: red shifting occurs when a body is moving away from you (the wavelength of the light increases) and blue-shifting occurs when a body is moving towards you (the wavelength of the light decreases). This is useful input for determining the speed and direction of something. In software development we have green shifting which occurs when people rework the information contained in status reports to make them more politically palatable to their manager.

A few years ago I was involved with a seriously challenged project. In my weekly status report to the project manager I would very bluntly list all of the problems we were experiencing. Strangely, even though myself and several others were clearly documenting the problems, all of which were out of our control, nothing ever seemed to happen. After a few weeks of this I ran into the CIO and she congratulated me on how well the project was going. I was surprised, and told her that the project was in serious trouble and summarized the critical issues for her. It was news to her.

After a bit of investigation we discovered that although myself and team members had been reporting the serious political challenges that the project team faced, when our project manager summarized our reports he decided to put a better spin on it and reported that the team found the project challenging. His manager in turn reworked it in his report that the team enjoyed the challenges it faced. This was the report which the CIO received.

The problem is that the project status "green shifted" as it rose through the various levels of management. The people on the ground were very clearly reporting a red project status, our project manager wanted to play it safe so reported yellow status, and his manager was more political yet and reported green status.

So how can you avoid this problem? In this organization the only way the CIO could find out what was really happening on the project was to go to the source " she couldn't trust the status reporting hierarchy within her organization which was why she struck up a conversation with me to begin with. Experience had told her that traditional status reporting wasn't going to provide an accurate picture of what was actually happening, but at best provided a sanitized version.

I believe that green shifting is a significant contributor to project failure within many traditional IT organizations. If senior management is unable to find out the true status of a project, then they will be unable to bring resources to bear to fix the problem(s) before it's too late. Or, worse yet, they don't find out about the "walking dead" projects which have no hope of success and therefore should be cancelled immediately; instead these projects struggle along for months wasting resources and destroying morale.

In agile development organizations this problem is greatly reduced. Instead of writing status reports, most agile teams hold a 15-minute daily stand up meeting where each team member describes what they did yesterday, what they intend to do today, and what problems they are currently dealing with (if any). This meeting is called a "scrum" meeting and it is one of the primary practices of the Scrum methodology. Everyone on the team must attend the scrum meeting to provide status, and anyone external to the team who wishes to hear the current status is welcome to attend the scrum. If you don't attend the scrum meeting you won't hear the detailed status: nobody is going to invest their valuable time writing you a status report.

This approach scales very well. Large projects are split into sub-teams, and one person from each sub-team scrum meeting attends a "scrum of scrums" where status is reported up the hierarchy. Status is reported back down the next day by this person in the sub-team's scrum. If you rotate the responsibility for attending the scrum of scrums then you decrease the opportunity for a single person to act as a political filter which green shifts, or red shifts for that matter, your team's status. Similarly, the status within a department or division can be communicated via the scrum of scrums approach.

To summarize, in written reports information will "green shift" the more political filters it goes through between the sender and the recipient. If you're looking for a more open and honest report, then reporting status publicly via scrum meetings is a very good way to do so. Everyone on the team learns about what everyone else is doing, greatly reducing coordination problems, and senior management has the opportunity to find out exactly what is going on simply by attending the scrum. There's one primary challenge with this approach: Can management actually handle the truth?

The term "green shifting" was introduced to me at the Javapolis conference in Belgium in December of 2005. Two attendees and I were talking about political problems in software development and we started talking about this idea. Unfortunately I didn't get their names otherwise I would be attributing the source of this term to them. Anyway, if they happen to be reading this newsletter, please send me an email so I can acknowledge them as the source in future writings.

Book Review: Agility and Discipline Made Easy

Many organizations want to be effective and streamlined at software development, hence agile approaches are attractive to them. At the same time the realities of modern regulatory requirements and the need for effective IT governance make the Unified Process (UP) attractive to organizations. These requirements don't have to be contradictory, as long as you're willing to take an agile approach to the UP?

The UP can be agile? Preposterous you say? Absolutely not. In *Agility and Discipline Made Easy*, Per Kroll and Bruce MacIsaac, the manager and technical lead of the UP content team at IBM respectively, describe exactly how to be agile with the UP. The book presents a collection of 20 practices, such as Embrace and Manage Change, Understand the Domain, Describe Requirements from the User Perspective, and Organize Around the Architecture, which form the heart of an agile instantiation of the Unified Process. Each practice is presented in pattern format, with a description of the problem which it addresses and how to apply the practice to varying levels of degree. The latter feature is what I find most interesting, Kroll and MacIsaac provide an easy tailoring mechanism for you to adopt the practices in a manner which reflects your situation. Because every project team is different, and because they face different situations, they're going to need their own "flavor" of the UP.

Another interesting feature, perhaps worth the price of the book all by itself, is the discussion of the Eclipse Process Framework (EPF) and the Open UP, an open source version of the UP. EPF and Open UP together represent a new vision for the Unified Process: the UP is now being developed by the people, for the people. Together we can make the UP as agile as we wish. The practices described in this book provide the guidance, the EPF the tool, and the Open UP the actual means. This is something all developers should start paying attention to.

Agility and Discipline Made Easy: Practices from OpenUP and RUP
By Per Kroll and Bruce MacIsaac
Addison Wesley, 2006
http://www.amazon.com/exec/obidos/ASIN/0321321308/ambysoftinc/
--SWA

Hot Links

The Agile Alliance homepageis the best starting point for anyone interested in learning more about agile software development.

Here's a collection of agile project planning tips of mine.

In Managers Manage I overview the Scrum methodology, an agile approach to project management.

The Eclipse Process Framework (EPF) and Open UP can be downloaded from the Eclipse Foundation.

The Agile Unified Process (AUP)can be downloaded from my site.

The Agile Models Distilled page provides links to overviews of a wide variety of models.

The principles of Agile Modeling v2 are now available.

The practices of Agile Modeling v2 are also available.

Check out the Agile Modeling mailing list.

Get agile modeling training resources here.
