13. Teams and Projects

What if you have lots of little projects to get done? How should you allocate those projects to the programmers? What if you have one really huge project to get done?

Does It Blend?

I have consulted for a number of banks and insurance companies over the years. One thing they seem to have in common is the odd way they partition projects.

Often a project at a bank will be a relatively small job that requires one or two programmers for a few weeks. This project will often be staffed with a project manager, who is also managing other projects. It will be staffed with a business analyst, who is also providing requirements for other projects. It will be staffed with some programmers who are also working on other projects. A tester or two will be assigned, and they too will be working on other projects.

See the pattern? The project is so small that no individual can be assigned to it on a full-time basis. Everybody is working on the project at

50, or even 25, percent.

Now here's a rule: There is no such thing as half a person.

It makes no sense to tell a programer to devote half their time to project A and the rest of their time to project B, especially when the two projects have two different project managers, different business analysts, different programmers, and different testers. How in Hell's kitchen can you call a monstrosity like that a team? That's not a team, that's something that came out of a Waring blender.

The Gelled Team

It take time for a team to form. The team members start to form relationships. They learn how to collaborate with each other. They learn each other's quirks, strengths, and weaknesses. Eventually the team begins to *gel*.

There is something truly magical about a gelled team. They can work miracles. They anticipate each other, cover for each other, support each other, and demand the best from each other. They make things happen.

A gelled team usually consists of about a dozen people. It could be as many as twenty or as few as three, but the best number is probably around twelve. The team should be composed of programmers, testers, and analysts. And it should have a project manager.

The ratio of programmers to testers and analysts can vary greatly, but 2:1 is a good number. So a nicely gelled team of twelve might have seven programmers, two testers, two analysts, and a project manager.

The analysts develop the requirements and write automated acceptance tests for them. The testers also write automated acceptance tests. The difference between the two is perspective. Both are writing requirements. But analysts focus on business value; testers focus on correctness. Analysts write the happy path cases; testers worry about what might go wrong, and write the failure and boundary cases.

The project manager tracks the progress of the team, and makes sure the team understands the schedules and priorities.

One of the team members may play a part-time role of coach, or master, with responsibility for defending the team's process and disciplines. They

act as the team conscience when the team is tempted to go off-process because of schedule pressure.

Fermentation

It takes time for a team like this to work out their differences, come to terms with each other, and really gel. It might take six months. It might even take a year. But once it happens, it's magic. A gelled team will plan together, solve problems together, face issues together, and *get things done*.

Once this happens, it is ludicrous to break it apart just because a project comes to an end. It's best to keep that team together and just keep feeding it projects.

Which Came First, the Team or the Project?

Banks and insurance companies tried to form teams around projects. This is a foolish approach. The teams simply cannot gel. The individuals are only on the project for a short time, and only for a percentage of their time, and therefore never learn how to deal with each other.

Professional development organizations allocate projects to existing gelled teams, they don't form teams around projects. A gelled team can accept many projects simultaneously and will divvy up the work according to their own opinions, skills, and abilities. The gelled team will get the projects done.

But How Do You Manage That?

Teams have velocities.[1] The velocity of a team is simply the amount of work it can get done in a fixed period of time. Some teams measure their velocity in *points* per week, where points are a unit of complexity. They break down the features of each project they are working on and estimate them in points. Then they measure how many points they get done per week.

Velocity is a statistical measure. A team might get 38 points done one week, 42 done the next, and 25 done the next. Over time this will average out.

Management can set targets for each project given to a team. For example, if the average velocity of a team is 50 and they have three

projects they are working on, then management can ask the team to split their effort into 15, 15, and 20.

Aside from having a gelled team working on your projects, the advantage of this scheme is that in an emergency the business can say, "Project B is in crisis; put 100% of your effort on that project for the next three weeks."

Reallocating priorities that quickly is virtually impossible with the teams that came out of the blender, but gelled teams that are working on two or three projects concurrently can turn on a dime.

The Project Owner Dilemma

One of the objections to the approach I'm advocating is that the project owners lose some security and power. Project owners who have a team dedicated to their project can count on the effort of that team. They know that because forming and disbanding a team is an expensive operation, the business will not take the team away for short-term reasons.

On the other hand, if projects are given to gelled teams, and if those teams take on several projects at the same time, then the business is free to change priorities on a whim. This can make the project owner insecure about the future. The resources that project owner is depending on might be suddenly removed from him.

Frankly, I prefer the latter situation. The business should not have its hands tied by the artificial difficulty of forming and disbanding teams. If the business decides that one project is higher priority than another, it should be able to reallocate resources quickly. It is the project owner's responsibility to make the case for his project.

Conclusion

Teams are harder to build than projects. Therefore, it is better to form persistent teams that move together from one project to the next and can take on more than one project at a time. The goal in forming a team is to give that team enough time to gel, and then keep it together as an engine for getting many projects done.

Bibliography

[RCM2003]: Robert C. Martin, *Agile Software Development: Principles, Patterns, and Practices*, Upper Saddle River, NJ: Prentice Hall, 2003.

[COHN2006]: Mike Cohn, *Agile Estimating and Planning*, Upper Saddle River, NJ: Prentice Hall, 2006.