Business Software Development Productivity

Blake McBride

August 27, 2019[1]

This essay attempts to identify business attitudes and practices that very significantly lead to waste. It then proposes a new model for business productivity in the area of software development. Many of these concepts can be used in other areas of business.

1 The Scourges Of Business

We will start by identifying three ideas that significantly harm a business' productivity. On the face of it, identification of these three ideas will likely appear insulting in today's society. However, if you delay your initial response and read to the end you will likely agree. The three scourges of business productivity are:

1. Equality
2. Teamwork
3. Diversity

On the face of it, these three are wonderful ideas. However, like many other things, too much of a good thing is bad. This essay argues that these three ideas, while not always, are too often taken to a harmful degree.

1.1 Equality

Although most would agree that each human life is equally valuable, few would argue that we all have the same talents, drive, or capabilities. Even with similar educational backgrounds, few would argue that each has the same ideas or abilities. The notion of equality arose in response to racist and misogynistic attitudes. Trying to solve those problems, we have taken the concept too far.

While business rarely treats workers at different levels equally, they often treat workers at the same level as if they are totally equal. The truth is no two individuals are equal (have equal potential benefit to the company). While it is important to treat each individual equally fairly, not taking advantage of those who can benefit the company greater is foolish indeed.

1.2 Teamwork

Being able to combine the talents of a number of workers in order to accomplish a task greater than the capabilities of any individual is critical to the success of a company. This is called teamwork. Teamwork involves working together in a cooperative way.

However, taken too far, members of a team are treated like individual parts that can be exchanged at will. Individuals are not important; the team is. The problem with this attitude is that the individual contribution to the team effort is rarely equal. Giving extra weight to those who have greater potential to contribute is wise.

[1] Last updated 5/22/2021

1.3 Diversity

Along with the notion of diversity are two fallacious notions:

1. Having a racially diverse group is greater than a less diverse group.
2. Group identity is more important than individual qualities.

What counts in all aspects of life are individual integrity, reliability, and capability. No one group has a monopoly on any of these. What's best for business is choosing the best. When a choice is made for "diversity"'s sake, damage is done to the individual chosen, the individual not chosen, the team, and the company.

2 The New Project Model

In this essay, I am proposing a new development model. The hallmarks of the new model are as follows:

1. A worker's success value (WSV) is determined primarily by their track record for success rather than:
   (a) their longevity with the company
   (b) their sex, race, or other group identity
   (c) their title or certifications
   (d) their charisma or outspokenness
2. Rather than a team operating democratically where each person has an equal vote, member votes are weighted according to their WSV.

MSV calculations are given in the next section.

2.1 Architects

Architects are those who create the overall design of a system.

Too many architects are bright, well educated, and have a lot of knowledge but are severely lacking in practice experience and a history of success. Although they can rattle off a great deal and sound authoritative, all too often they lack critical pieces of information that can make or break a project. Those in the position of architect should have an MSV value higher than anyone below them.

The architect works closely with the team lead throughout the process — perhaps before the formation of the rest of the team. However, the architect makes all of the final decisions.

2.2 Team Lead

The main job of the Team Lead (TL) is to implement the design given to him/her by the architect.

The team lead is a single person on the implementing team whose WSV value is higher than all of the other members of the team. Rather than running the team as some sort of democracy, the TL makes all of the final decisions of the team. Although there is discussion, there is no vote. The TL must ask for and receive input from the other team members prior to making a decision.

One of the ideas here is that the TL is likely to make the best decisions. Also, by having one person make all of the final decisions, the resulting system will be more consistent. Some of the benefits of this system are:

1. having a single decision point leads to consistent solutions
2. leverage off of the most experience rather than having an equal vote among members with significantly different levels of experience
3. reduce the wasted time and effort to over-debate and over-investigate issues
4. eliminate the political elements

The TL makes all estimates with input from the team. Time estimates are in hours/days/weeks and not points.

The TL is an active developer. TL's eventually get promoted to architects.

2.3 Senior Software Engineer

The senior software engineer (SSE) should be a single person with the second-highest WSV on the team. The SSE should work closely with the TL and be considered a TL-in-training. After a number of projects as an SSE, the SSE is promoted to a TL.

2.4 Software Engineers

The remaining developers on the team are either software engineers or junior software engineers. The differentiating point is whether or not the individual can largely work by themselves. Those who can are software engineers. Those who require a bit more help and over-site are junior engineers.

2.5 Testers (QC)

Utilize regression testing. The team should not use unit testing except in rare circumstances. Having a developer create unit tests means having the same person write unit tests with the same knowledge and assumptions. Unit tests take a lot of time to create and serve little purpose that regression testing doesn't reveal. In fact, regression tests are far more conclusive than unit tests because they also test the inter-unit interface.

The number of testers should be equal to the team lead + the senior engineer + the number of other software engineers divided by two and rounded up.

If a tester finds an error, it points out a developer error. If a customer finds an error, it points out a tester error.

2.6 Scrum Master

The scrum master performs the following tasks:

1. runs & tracks progress
2. facilitator
3. liaison with other teams

The scrum master does not make decisions or assign tasks. The TL does that.

2.7 Documentation

Tech docs are created during or immediately after the development. This is not done prior to the project because too often, not enough is known about the project until it is actually implemented. In other words, many things are discovered during the implementation phase that are unknown before then.

This offers significant savings in time since there is less implementation of an inaccurate document and less time creating and updating the document.

Of course, technical planning must occur before the implementation; however, this is typically done informally by the team lead prior to the coding.

2.8 Separate Teams

The following are performed by separate teams:

1. DevOps
2. CI
3. User documentation
4. First-level product support

3 Worker's Success Value Calculation

The method of calculating a worker's success value (WSV) is given in this section.

For each successful project a person is part of, they get the following points:

1 junior software engineer

2 software engineer

4 team lead in training

10 team lead

20 architect

For a failed project, the same number is subtracted from their MSV.

The following table shows how many points it takes to be promoted to the next highest level:

10 software engineer to team lead in training

30 team lead in training to team lead

80 team lead to architect (plus some technical knowledge testing)

If a person's MSV falls below the requirement for their level, they are demoted to the lower level.

Perhaps a national database should be used to track these points beyond a single company.