CS 430 - Lecture 08 - Teams II

Collin Roberts

October 3, 2023

Outline

Synchronize and Stabilize Teams

1. Recall that so far, the synchronize-and-stabilize model has only been used within Microsoft.
2. **Rule #1:** The developers must adhere strictly to the agreed upon time to check their code in for that day's synchronization.
3. **Rule #2:** If a developer's code prevents the product from being compiled for that day's synchronization, then the problem must be fixed immediately, so that the rest of the team can test and debug.

Remark

The culture of the organization must fully support Rules #1 and #2 before this life-cycle model and team organization can have any success.

Strengths

1. Encourages individual programmers to be creative and innovative, a characteristic of a **democratic team**.
2. The synchronization step ensures that all programmers work together for a common goal, a characteristic of a **chief programmer team**.

Weaknesses

1. There is no evidence yet that this model can work outside of Microsoft.
   A Possible Explanation: There is something unique about Microsoft's culture, which has yet to be replicated elsewhere.

Advantages of **pair programming**

1. "Two heads are better than one."
2. It should produce high quality code.
3. Fewer typos/small bugs.
4. Programmers do not test their own code.
5. All knowledge is not lost if one programmer leaves. The remaining programmer from the pair can train a new pair programmer.
6. Less experienced programmers can learn from more experienced programmers.
7. The technique promotes group ownership of the code, a key feature of **egoless programming**.

Disadvantages of **pair programming**

1. Twice the person-hours: more expensive.
2. It can be slow; programmers can become distracted.
3. Subjective disagreements can waste time.
4. Each programmer must regard the other as an equal.
5. Feedback given by teammates may not always be constructive.
6. Extremely shy people might dislike this technique - they must speak up while (pair) programming and during (daily) meetings. Overbearing people might dominate.

Remarks

1. More research is needed to determine whether the benefits outweigh the costs.

Reasons why people would **not** want to participate in an open source project

1. unpaid
2. philosophical disagreements about direction.
3. Since the programmer does not own the code, he/she cannot monetize the work at all, even after the development is done.
4. You must give up control over the finished product (or even your own piece of it).
5. Intellectual property problems: You give away what you produce during such an effort.

Reasons why people choose to participate in an open source project

1. You are empowered to fix problems.
2. It benefits everyone to have some successful open-source products available.
3. the sheer enjoyment of accomplishing a worthwhile task. "Making the world a better place."
    1. Volunteers must continue to perceive that the project is worthwhile; they will drift away if the project begins to seem futile.

Reasons why people choose to participate in an open source project

4. the learning experience
    1. Employers frequently view experience gained working on a large, successful open source project as more desirable than additional academic qualifications.
    2. Hence it is crucial that the project be perceived as possibly successful to retain its programmers.
5. An organization depends on an open source application, and hence is motivated to devote resources to supporting the open source team.

Reasons why people choose to participate in an open source project

- **In summary,** an open source project must be viewed at all times as a "winner" to attract and retain volunteers to work on it.
- **Corollary:** The key individual behind the project must be a superb motivator.

Morals

1. For success, top-calibre programmers are required. Such programmers can succeed, even in an environment as unstructured as an open-source one.
2. The way that a successful open-source project team is organized is essentially irrelevant to the success/failure of the project.

People Capability Maturity Models

1. Recall that P-CMM was the capability maturity model for People. It describes best practices for managing and developing the workforce of an organization.
2. Similarly to SW-CMM, an organization progresses through five levels of maturity with the aim of continuously improving individual skills and engendering effective teams.
3. Also similarly to SW-CMM, P-CMM is a framework for improving an organization's processes for managing and developing its workforce, and no specific choice of team organization is put forward.

Choosing an Appropriate Team Organization

Here is Figure 4.7 from the text:

Team Organization	**Strengths**	**Weaknesses**
Classical Chief Programmer Teams (§4.3)	-Major success of NYT project	-Impractical
Democratic Teams (§4.2)	-High quality code as a consequence of positive attitude towards finding faults -Particularly good with hard problems	-Experienced staff resent their code being appraised by beginners -Cannot be externally imposed
Modified Chief Programmer Teams (§4.3.1)	-Many successes	-No success comparable to the NYT project
Modern hierarchical programming teams (§4.4)	-Team manager / Team leader obviates need for chief programmer -Scales up -Supports decentralization when needed	-Problems can arise unless team manager / leader responsibilities are clearly delineated

Choosing an Appropriate Team Organization

Here is Figure 4.7 from the text:

Team Organization	**Strengths**	**Weaknesses**
Synchronize and Stabilize Teams (§4.5)	-Encourages creativity -Ensures that a huge number of developers can work towards a common goal	-No evidence so far that this method can be used outside Microsoft
Agile Process Teams (§4.6)	-Programmers do not test their own code -Knowledge is not lost if one programmer leaves -Less experienced programmers can learn from others -Group ownership of code	-Still too little evidence regarding efficacy
Open Source Teams (§4.7)	-A few projects are extremely successful	-Narrowly applicable -Must be led by a superb motivator -Required top-calibre participants

Choosing an Appropriate Team Organization

1. There is no one choice of team organization that is optimal in all situations. Different strengths / weaknesses will matter more at different times.
2. In practice most teams are organized according to some variant of the (modified) chief programmer model.