Organic Cross-Training

Incentives for reducing hidden bottlenecks



"Best Cross Training Shoes for Flat Feet" (CC BY 2.0) by gm.esthermax

When is cross-training appropriate and desired? Cross-training done wrong can be an unnecessary time sink. What are the conditions that make cross-training necessary? Should cross-training be formal or informal? The answers to these questions are not straightforward because the incentives for cross training are hidden, and only show themselves when we make work visible. The problem is compounded in self organized groups, because these initiatives must emerge from a joint recognition of the need for cross-training instead of as a top down directive.

Why Cross-Train?

Some organizations, especially consultancies, have a strong bias for individual metrics[1]. That is to say individuals are graded by the quality and quantity of their own work. This incentivizes individual instead of team commitment and disincentivizes cross-training. The reasons for cross-training stem from the problems that arise during a work pipeline[2][3][4]. Too much work in

[1] "Focusing on utilization prevented collaboration. People were incentivized to not help others. Why should Alan go help Russ if it means the work in Alan's lane will take longer to complete.", Degrandis, Dominica/ Demaria Tonianne (FRW). MAKING WORK VISIBLE: Exposing Time Theft to Optimize Work & Flow. Natl Book Network, 2017.

[2] Goldratt, Eliyahu M., and Jeff Cox. The Goal: a Process of Ongoing Improvement. Productivity and Quality Publishing, 2013.

[3] Kim, Gene, et al. The Phoenix Project: a Novel about IT, DevOps, and Helping Your Business Win. IT Revolution, 2018.

[4] Degrandis, Dominica/ Demaria Tonianne (FRW). MAKING WORK VISIBLE: Exposing Time Theft to Optimize Work & Flow. Natl Book Network, 2017.

Copyright © 2016 W. Watson, Vulk LLC All Rights Reserved

progress (WIP), unknown dependencies, ambiguous priorities, and unplanned work contribute to creating blockers that keep work items in the last 10% stage of completion.

Constraints on Cross-Training

If we have habits that steal our time but are invisible, we won't allocate resources to address them. Cross-training can solve problems that arise repeatedly and are highly visible by creating redundant methods of addressing blockers. Many habits that create blockers are not visible, and therefore do not get addressed directly with cross-training but rather are looked at as individual shortcomings. When individuals and teams get cross-trained, they adopt T-shaped skills, which allow them to develop a deep understanding in a specialized skill set, as well as a wide range of broadly applicable skills that span across domains.

Work Flows

There are many ways to describe workflows such as by using kanban boards, or as value streams or software pipelines. A kanban board[5] illustrates work items as they flow from left to right throughout the various stages of implementation, like "to do," "doing," "blocked," and "done.". A value stream[6] starts when a new request is made by a customer and continues on to to the completion of the request. The request can be a new feature or a complete product. A software pipeline[7] starts from the point a developer checks in code and ends when working code is available to whoever requested it. In a well implemented pipeline, customers, power users, QA, developers, and others can 'pull' environments (and essentially work items) to them instead of waiting to have that work 'pushed' or assigned to them. Both of these workflow models try to address the common issue of wait time. Wait time can be increased by many factors, such as lack of automation, lack of expertise at key moments, and an elongated feedback cycle. The point of modeling the workflow is to isolate the bottlenecks and address them by any means necessary. Sometimes the means of addressing a bottleneck is counter-intuitive from the perspective of the individual but increases overall business value. Cross-training at the point of the bottleneck is one way to incrementally reduce the wait time created by the bottleneck and optimize the efficiency of the workflow.

Too Much Work In Progress (WIP)

---

[5] https://en.wikipedia.org/wiki/Kanban_board

[6] "Thus the process modeled by the deployment pipeline, the process of getting software from check-in to release, forms a part of the process of getting a feature from the mind of a customer tor user into the their hands. The entire process -- from concept to cash -- can be modeled as a value stream map", Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation." Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, by Jez Humble and David Farley, Addison-Wesley, 2015, pp. 106–107.

[7] "At an abstract level, a deployment pipeline is an automated manifestation of your process for getting software from version control into the hands of your users.", Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation." Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, by Jez Humble and David Farley, Addison-Wesley, 2015, pp. 106–107.

Copyright © 2016 W. Watson, Vulk LLC All Rights Reserved

Too much work in progress is a common practice popularly described in the book *The Goal*[8]. In *The Goal*, a concept known as a buffer is used to describe the work around a constraint. If there is not enough work before a constraint (because other parts of the process have too much work assigned to them), the buffer is not large enough and the constraint becomes the bottleneck. If there is not enough buffer after the constraint, downstream work can back up and cause the constraint to be a bottleneck. The final 10% of an effort to implement a new feature can be become a bottleneck because the the people who are experts in integrating with other components are plagued with too much work in progress. By making the amount of work in progress by team or by skill set visible, the need for cross-training can become visible to the whole team.

Unknown Dependencies

Every dependency doubles your chance of being delayed or late[9]. When an unknown dependency emerges, a work item can be delayed unnecessarily and unknowingly to the team. Dependencies are even more difficult when they require coordination across teams in a large organization. By making the dependencies visible, the incentive for cross-training can be known across teams and be implemented if the pain from the delay is strong enough.

Priorities

Ambiguous prioritization and prioritization methods incentivize people to add to their own personal WIP in the order of their choosing (i.e. the cool, new work items instead of the prioritized ones). When blocker for a work item arises and the next in line work item is not clearly communicated, either a non-prioritized work item or, worse, an unplanned work item can enter into a person's WIP. Prioritization of work items can be complicated by the fact that the prioritization method (highest paid person presence, first in first out, low hanging fruit first, etc) is unknown. If each priority, the prioritization method, and a strong stance towards the value of a priority are all made visible, people will stick to the priorities and become more innovative in solving blockers, which should include cross-training during the time that the priority is becoming unblocked.

T-Shaped Skills

When the causes for bottlenecks are made visible, cross-training becomes incentivized for the team. As people are cross-trained they take on a new T-shaped skill set. T-shaped skills forgo narrow disciplinary skills for skills that deliver business value, such as skills found on product teams. While there are one or many specializations in a T-shaped skill set, there are broad skills that span disciplines. Some examples of T-shaped skills on a product team are the

[8] Goldratt, Eliyahu M., and Jeff Cox. The Goal: a Process of Ongoing Improvement. Productivity and Quality Publishing, 2013.

[9] Degrandis, Dominica/ Demaria Tonianne (FRW). MAKING WORK VISIBLE: Exposing Time Theft to Optimize Work & Flow. Natl Book Network, 2017.pp 19

Copyright © 2016 W. Watson, Vulk LLC All Rights Reserved

abilities to develop, test, and deliver a product or feature in the required environments instead of completely handing off the work items as a ‘project’ to another team.

Informal Training

A study on cross functional teams described the four ways in which they learn[10]:

1. Direct knowledge or skills transfer from other members
2. Observing other members in action
3. Engaging in team problem solving
4. Expanding prior experience and reframing new insights

Instead of formal methods, such as classroom settings or online courses, these are informal methods of training. When unblocking a bottleneck, we should expect a self organized team to successfully cross-train using these methods or ones very similar (such as pair programming and peer reviews). While these methods are informal, we shouldn’t discount their effectiveness when it comes to incrementally creating a team that is capable of unblocking bottlenecks.

Conclusion

Cross-training is needed when work items are demonstrably blocked repeatedly. We can demonstrate the need for cross-training when we make the processes that lead to wait times and blockers visible. Cross-training is needed in order to address too much WIP, unknown dependencies, and unclear priorities properly. We should not discount the need for informal training even if it doesn’t seem as valuable as formal training at first glance.

[10] “Cross-Functional Teams: Working with Allies, Enemies, and Other Strangers.” Cross-Functional Teams: Working with Allies, Enemies, and Other Strangers, by Glenn M. Parker, Jossey-Bass, 2003, pp. 148–148.

Special thanks to the Vulk Team

Copyright © 2016 W. Watson, Vulk LLC All Rights Reserved