Deliver Better Results

How to Unlock Your Organization's Potential **GIL BROZA**

Supplementary resource for the book *Deliver Better Results*. More at DeliverBetterResultsBook.com

How to Stabilize a Value Delivery System

To stabilize your system, improve choice-making regarding work items' movement through the workflow. Here are 13 ideas and principles, which have stood the test of time, for doing that (be sure to apply them across the system!). Many of them come from Kanban and Agile methods, although they're also useful for more plan-driven methods. You *don't have to use all of them*, certainly not from the get-go.

1. **Visualize the work.** Create mechanisms for seeing where all the work is and what's happening to it. The typical artifact for this is a board (whether physical or electronic): columns represent workflow stages, and items move from left to right as they proceed through the workflow. It's often useful to subdivide boards horizontally into "swimlanes" to call out different classes of items. Effective visualization makes visible several risks to stability, such as items aging or being blocked, bottlenecks, and exceeding of intended limits.
2. **Define explicit intake and completion standards.** For better stability, reduce the *variability of wasteful rework*. Do so for *preventable rework* by defining standards or policies for moving items from one workflow stage to the next. It's often helpful to capture such standards as checklists that reflect the item's readiness for the next stage.
3. **Cultivate learner safety.** To do their job properly, people should feel safe to ask for clarifications, say "We don't know," study what they need, try things out, and own up to mistakes.
4. **Increase intrinsic quality.** Improve areas of the product's design and construction where further development is particularly problematic; this usually requires breaking down, decoupling, or even replacing large components. Enhance the testing approach to catch breakage early and efficiently, likely with the help of automated checks. Upgrade artifacts that are used in the making of the product (such as such as specs, user stories, and docs) to be more consistent, clear, and organized.
5. **Break work down into smaller meaningful pieces.** Instead of decomposing each portfolio item into a sequence of functional tasks with integration at the end, deliver it in an iterative and incremental (evolutionary) sequence of work items that produce mini-outcomes. It may be enough to decompose large work into medium work; the pieces don't have to be tiny.
6. **Get to "done."** Do what you can to finish what's started with minimal delays; don't let items age too much. If you also apply the previous principle (breaking work down), get to "done" on each piece.
7. **Reduce bottlenecks.** Identify the worst bottleneck and which people and tools are involved. Reduce work they do elsewhere, so they may move more work through the bottleneck. Additionally, modify the process to reduce dependency on the bottleneck.
8. **Constrain work intake.** Impose artificial constraints so that teams and individuals avoid taking on more work than they can realistically complete. Two useful and popular constraints, which may be combined but don't have to be, are to plan work and execute it in small time-boxes and to limit WIP. Time-boxes need to be short enough, and WIP limits low enough, to produce effects that increase system stability. Such effects include: encouraging people to focus, collaborate, and break work down; replacing "push" with "pull"; and reducing excessive context-switching.

9. **Turn as much unplanned work into planned work as possible.** Even with solid management of priorities and intake, some demand will be placed on the system in the form of *unplanned work*. Part of that demand is literally unplanned because the requesters don't think of it beforehand or try to bypass the intake mechanisms. Sometimes, explaining the consequences to people is enough to make a significant improvement; at other times, you'll need to make process changes.
10. **Determine how to handle the rest of unplanned work.** Categorize all this work and decide how to deal with each category.
11. **Keep spare capacity.** Make sure that individuals and teams keep enough spare cycles to deal with unplanned work while maintaining a sustainable pace. And whenever they don't put those spare cycles toward unplanned work, they engage in activities that indirectly produce value: improve the process, reduce debt, learn something, collaborate with colleagues.
12. **Manage high-variability delays.** Collect data on how long work items take to get from start to finish, and see which kinds of items take much longer than they should based on the actual touch time. Map out their flow, and identify periods of excessive waiting for experts, reviews, approvals, vendors, integration, testing, etc. Focus on delays whose occurrence and duration are not predictable enough, and deal with their causes.
13. **Enable people to contribute outside narrow specialties.** If tasks *always* go to the people who are experts in them, some experts will get overloaded and the system will have bottlenecks and delays. Since not all tasks require deep specialization, create opportunities for people to expand their repertoire and to contribute to finishing work.