# From Chaos to Order: Dealing with Mid-iteration Changes

By Tom Kubit
tomk@genecodes.com
Gene Codes Forensics, Inc.

While we would all like to have a project that runs perfectly and all aspects align with just what the book says, we all know that the real world just is not that way. Unforeseen challenges, emergencies, or simply just the way the business works causes the process to break down from time to time. For a team trying to ramp up to being a “full-fledged XP team”, these set-backs can have a significant negative effect on team morale and add to the stress level of the project. How we deal with the adversities we face can mean the difference between continued success and spiraling out of control.

My current project ran into just this kind of dilemma early on. The project involves writing an application to identify victims of the World Trade Centerdisaster through DNA analysis techniques. It was decided from the start that we would deliver the software on a weekly basis. Being 500 miles away from New York Citypresented its own set of challenges for meeting this schedule. The company president would deliver the software each week and get the requirements for the next delivery. He would then act as the project Customer and relay the new stories to the developers upon his return. The software for the prior iteration was delivered near the beginning of the next iteration. Spending time with the users would usually result in requests for new features that the analysts felt would help them out the most for the next set of challenges. Many times these requests required a quick turnaround. Helping the analysts to identify victims more quickly made it paramount to get the new requests back to them as soon as possible. However, by the time the new requests would get back to the development team, the iteration was nearing the mid-way point. Following the rules of iteration planning, this meant the new request would be added during the planning for the next iteration and be delivered two weeks after the request was made. There was a strong need to be able to respond to these requests in the current iteration.

The team struggled early on with these heavy changes in priorities occurring within one-week iterations. We were continually leaving tasks half done and the development process felt haphazard and out of control. We made a number of adjustments to the process to try to deal with this situation but probably the most important were prioritization of stories within the iteration and the weekly retrospectives. The retrospectives allowed us to evolve our process, provided a good barometer of team morale, and allowed everyone to participate in a very open discussion about both what could be done better and what went well over the course of the iteration. Many of the changes described below were the result of discussions during the retrospectives.

Given that mid-iteration changes weren’t going away, we had to deal with them as best we could. A second planning session, usually the first day the Customer returned, would take place for the new stories. We would estimate and task the stories, and then attempt to fit them into the current iteration. The original, now lower priority stories would be moved to the top of the list for the next iteration. However, due to the way we took on programming tasks, it was difficult to fit in the new stories without losing a lot of development effort.

We originally did a “standard” XP task plan: identify the tasks, give them estimates, and sign up for tasks. When we signed up for tasks, each person would sign up for all the tasks that they thought they could get done during the iteration. This would have the desired effect of distributing the tasks across the whole team. However, once the mid-iteration stories started coming in, there was no room for new tasks because everyone was booked. It was hard to tell who was doing what, what was half-done, and what had yet to be started. To solve this we came up with a task signup rule. We agreed to only sign up for one task at a time; that is, to not sign up for the next task until the current one is crossed off.

This made it a bit easier to fit in new stories since many of the remaining tasks were not selected. The team would lean towards taking points off the board from the stories that had no tasks started to make room for the new stories. It was, of course, the customer’s decision which stories were removed. It was now more obvious what the effect would be on the team’s velocity if stories that were already started were chosen for moving to the next iteration.

Even though we were only signing up for one task at a time, we would still pick tasks across the range of stories in the iteration. The new stories were easier to add in, but we still tended to throw away a significant amount of work from one or more of the stories being dropped. The retrospectives discussions focused on how we were still struggling with the amount of work that was either lost or postponed. Another consequence of losing work was our velocity measurements were dropping and it was hard to tell what our base point estimation meant anymore. We couldn’t get the feel for what a 5-point iteration meant in comparison to one with 8 points. This made it difficult to come up with consistent estimates for the next planning session.

Management understood XP’s rules, and was willing to take a hit in velocity to get same-week delivery of some new features. Yet the development team still felt a loss of control over how we got those features delivered. We were still fighting too many fires, especially near release day at the end of each week. At this point in time the stress level of the team was also rising since it was becoming harder and harder to feel like we were able to accomplish what we set out to do at the beginning of the week without losing a bunch of work. The retrospectives were key in helping the team discuss the issues. We determined we needed a way to limit the number of stories being worked on at any one time.

We added another rule: the customer would explicitly prioritize stories for the iteration, and we would try to do the tasks for the highest priority stories first. (We allowed judgment, so it was permitted to skip to the next story if the remaining tasks in the highest priority story would interfere with each other.) With this rule, new stories are much less disruptive. And it has another beneficial side effect: when we’re really clicking, and we get three pairs swarming around a story, we can really “crank it out.”

With this new rule, in combination with the single task sign-up and second planning session, we finally had a clear picture of what was being worked on and where the new stories folded into the mix. While the second planning session would take time away from development, we felt it was essential in keeping everyone on track. By about the sixth iteration we began to feel our customer and planning process was in a much more manageable state.

It has been about a year and a half since this rule was put in place. The frequency of mid-iteration changes has decreased dramatically. However, we continue to prioritize our stories within each iteration to make sure we stay on track. It allows us to focus on the most important stories and continually deliver the highest value to the customer. Furthermore, when we do run into those tough weeks every now and then, we are in a good position to handle just about anything that comes our way. We would recommend that any team use prioritized stories within an iteration as way to stay organized and poised for embracing change.