

A Scrum project – brief experience report

William C. Wake, William.Wake@acm.org, 12-30-04

I worked as part of a team on a short-term (3-4 month) project designed to shift a business to a different part of the market. The software was successful technically (enough systems were put in place to support testing the new business model), but the marketing test did not succeed and the division exited the entire business within the following year.

Without going into too many details, the business was a service business, designed around an inbound call center supporting sales and service. They had built up a good-sized business, but prices on the underlying product were eroding. This made the service less profitable to re-sell, so the business decided to try targeting a different group of people where the company's ability to assist with financing would make up for the erosion.

The project was structured as a “bet-the-company” move. To highlight the importance, and to remove the distraction of day-to-day support of the existing business, a separate team was set up.

There were several levels of participation:

- A solid core team consisted of about 35 people, from every aspect of the business: marketing, sales, legal, call center, IT, support, and so on. This group was 100% dedicated to the project.
- An extended group of about 25 more. This group varied in participation; all had some responsibility, and committed to spending at least a day a week directly with the core team.
- A support group of approximately 50 more; these people were not normally tied to the project, but could be called on for immediate help.

Environment

Early on, the executives decided to support working together in a single location. Previously, the bulk of the team was in one of two locations, about 100 miles apart. For this project, the company rented a warehouse in a town about halfway in between. The core team committed to being there 4 or more days/week; the extended group committed to being there at least one day/week.

The environment inside was sparse: perhaps a dozen folding tables in the large area, a second-floor conference room that could seat a dozen comfortably, a small room that could seat 3-4, bathrooms, and a kitchen. The large room had a basketball hoop, and enough room to play one-on-one.

The furniture and walls were cheap (by corporate standards) and noticeably informal; it had the feel of a startup. This was a conscious decision: the location would only be used

for a few months, and the startup feel was part of the project's "mystique."

There was limited phone service and only dial-in internet service. There was no access to the corporate intranet. Partly this was due to security concerns, but also partly to a conscious decision to keep the group set apart and free from the normal corporate distractions.

Empowerment

This team was explicitly told to do whatever it needed to get things done.

In some cases, this took a simple form: someone needed a printer hooked up, so they just took care of it themselves. That doesn't sound too amazing, but it wasn't "the way things get done around here" back at the office: there were people whose job it was, and the security liaison would be involved, etc.

Another incident stands in my mind: there were no snack machines, so the group just agreed to throw money in a box every time they took a coffee or a snack. Every couple weeks, somebody would feel like there was nothing they wanted, so they'd take the money that had accumulated, call out, "I'm making a snack run – what does everybody want?", and go. Again, this doesn't seem like much, but the corporate culture tended not to encourage that back at the office.

The real empowerment came from having all the right people there. Because it was such a fully cross-functional team, people were encouraged to just call out whenever they had a question. The right person was almost always there to help.

Scrum

The software people thought of the approach as Scrum; the business thought of it as new product development: "work like a startup again."

The group used familiar Scrum practices:

- Time-boxing. The project was structured as a series of deliverables, one key goal per month. The monthly delivery was assessed, and the result used to plan the succeeding deliveries.
- Product owner making backlog decisions. There was a team supporting the product owner, with someone formally in charge. Most decisions were by consensus of this team.
- Managed backlogs. This was done both at the "overall" level and by individuals and groups within the month's work.
- Scrum meetings. There was a daily meeting among core people, though not structured by Scrum's "Three Questions." There was also a weekly meeting, described below.

The deliverable of this team was intended to be more than just a software system, although that was an important part. The deliverable was a business: a marketing campaign, a plan for organizing the call center, plans for the financial groups, and plans for other sales and support groups.

Scrum Meetings

The daily Scrum meetings were held among the core team and were not particularly remarkable.

The weekly meetings seemed much more important to me. The weekly meeting involved getting the extended group (of about 60) together every Monday morning. This extended group included the core group, along with their managers and other involved people. This group would get together in a large circle (sitting in folding chairs). Each person would give their report in turn, passing if they had nothing to add. The meeting took 2-3 hours per week.

Team Transition

Even though this project was intended to be of only a few months' duration, two separate software teams were involved. Team 1 began the project the first month, both Team 1 and Team 2 worked the second month, and Team 2 alone worked the third month. Team 1 was populated with people from an outside contracting company; Team 2 was an internal company software group.

My understanding was that this transition was because of a desire to bring development into the internal corporate groups, both for saving money and because of shifts in priorities. Team 2 wasn't able to form until the project had already started (due to people being involved in other projects).

The transition between teams went OK (no major glitches in the handoff), but it cost productivity in the second month (while Team 1 spent time helping transition, and Team 2 was trying to figure out what was going on).

Conclusion

Didn't Work So Well	Worked Well
<ul style="list-style-type: none"> The software worked (it supported the marketing test) but the business failed. Like the old joke – the operation was a success but the patient died. It's definitely served me as a reminder that even perfect software isn't necessarily good enough. 	<ul style="list-style-type: none"> Full business involvement was very effective. The project would easily have taken twice as long if it had been done the "normal" way. Collocation paid off. It made the work "special," and it highlighted the importance of peoples'

<ul style="list-style-type: none"> • The transition between teams cost some momentum, and several programmer-months' money. (The money was not an issue given the circumstances.) • The collocation ended after a few months, as teams went back to working in their home sites. 	<p>commitments to each other.</p> <ul style="list-style-type: none"> • The large Scrum meeting worked well. The 2-3 hours per week paid off well in keeping the whole company in touch with the effort and in focusing everybody on the week's goals. • Monthly deliveries and daily meetings worked as usual. • The product owner team was very creative in working with the software team to find cheap substitutes for full features. They distinguished well between "this is needed to try out the business" and "this is required to scale up the business." For example, manual credit checks sufficed for the former but not the latter.
--	---

My key lessons from this project were:

- Never think that software is enough.
- Even a very large team can work well together if they're focused well.
- Creative collaboration between "customers" and developers can save huge amounts of time and money.