

# Overcoming Fear with Trust

Brian Button  
Asynchrony Solutions, Inc.  
Brian.Button@asolutions.com

## Abstract

*This is a story of how one project succeeded in delighting their customer, despite being put into the most difficult of all situations for an agile team – a fixed price contract with an initially fearful customer. The next few pages will describe how we used the agile values and principles to create trust, how our practices had to be changed to help us accomplish this, and how that trust allowed us to succeed as a team.*

## 1. Introduction

I am the project manager and coach for a team that is about to deliver its first release of a website to our customer. My people are dedicated, skilled, and experienced, and our customer is about the best I could ever imagine having. But it didn't start this way.

Our customer, AB, had just come off a bad breakup – she had been involved with another company that professed to be agile, but was not having great success with it, so she had to let them go. Because of this, and justifiably so, she was left a bit suspicious of this *agile* thing. But she was willing to give it one more try, which was when she came to us.

The website we were being asked to build is a real estate search site for very high-end clientele. The properties listed by this company are among the most beautiful and most expensive in the U. S. Given the high-end nature of their clients, they were expecting a high-end website. It had to be graphically stunning, modern and interactive, and allow their users to browse their thousands of properties with ease. They had particular ideas about the concepts they wanted, but many of the details were still undecided. Given the timeline, the uncertainty around their requirements, and the difficulty of creating a visibly stunning site, it seemed that an agile process was the right choice.

There were some difficulties, though, based on their previous experiences. The rest of this paper will describe how their past relationship colored how we worked together, how we adopted and tailored specific agile practices to overcome these difficulties, and how

we build an atmosphere of trust between our customer and us.

## 2. Past Experiences

The website we are building and about to deliver represents this client's third try at fielding this site. Their first attempt was successful in getting a site fielded, but one that only covered a part of their market.

Their second attempt was less successful, mostly due to the contractor's agile implementation. This contractor was very new to agile, and seemed not to have fully grasped some of the spirit and principles behind some of the practices. These shortcomings in their implementation led directly to some of the challenges that we had to overcome in the early days of our project, so it is important to at least note them here.

### 2.1 Opaque iterations

The basic problem with the agile implementation was that the customer didn't have enough visibility into what was happening on a daily and weekly basis, which created a feeling of being out of control. The iterations were 2 weeks long, which is a reasonable length, but too long to go without the customer seeing progress. At the conclusion of these iterations, the customer was frequently surprised at what was completed and delivered. This lack of transparency led to several other, dependent issues.

### 2.2 Unkept promises

For various reasons the project team was frequently not able to deliver on the functionality they promised at the start of the iteration. This meant that choices had to be made about what they would finish. The problem was that the project team was making these decisions on their own and surprising the customer with what they had finished, and even what they had been working on, during the iteration.

This does not reflect the agile value of allowing the customer to drive, and added to the customer's feeling of lack of control.

### **2.3 Lack of progress**

The biggest reason that the project team had difficulty delivering was that their architecture never really settled down. The developers spent a large part of their time refactoring rather than adding new functionality. This was viewed as normal on an agile project, since design wasn't done up front. What was missed was that the architecture should settle down fairly quickly, and large-scale iterations should become very rare. They didn't, which affected delivery.

## **3. Creating our Bid**

Now this brings us to where my company, Asynchrony Solutions, comes in. We're a project-based consulting company that makes its living using agile methods to deliver solutions to our customers. We've been doing this for years, quite successfully.

We were contacted by this client in November, 2006, and asked to put together a proposal to build the website described above. So we sat down with them to understand the work that was involved, us with our experience in delivering custom applications using agile methods, and them with their previous unhappy experience with agile.

### **3.1 Constraints on our approach**

Due to the previous uncertainty about delivery, we were asked to define exactly what the content of our delivery would be, and put a price on what it would take to make it happen. Given their previous experience with agile, this was a completely understandable request, but it made our job very difficult in coming up with our bid.

### **3.2 Story writing for scope definition**

Our client came to visit us during this phase of bid creation to help us understand what we needed to do. She had some spreadsheets that had been worked out with her previous consultants that we turned into about 150 user stories. For the most part, these stories defined the functionality that she needed well enough that we could put a price tag on it.

Unfortunately, there were still a few areas that were pretty raw. These areas made up the most interesting and advanced sections of the web site, and there was no way to define them without building them first. The

best that we could do given our constraints was to assume the worst case for these areas and put in placeholder stories for them, accepting the risk.

To be honest, this scared us, but we had no other choice. We were relying on the skill of our developers and our growing relationship with the client to mitigate this risk, and fortunately for us, that's exactly what happened.

### **3.3 Estimation for costing**

Three of us, Nate, Adam, and I, gathered to create some really rough estimates for these stories. We knew that these estimates would be used to create our bid, so we tread carefully. We used the Planning Poker[1] method for creating our estimates, with Nate playing the role of customer and Adam and I doing the bidding. We ran through all 150 stories in a couple of afternoons, and we were ready to create our bid.

### **3.4 Creating the Statement of Work**

After the estimates were finished, we created the statement of work. We protected ourselves against the unknown by putting in some padding. Our client knew about this, and was completely fine with it, given the unknowns in the project and the not-to-exceed nature of the contract. After a few emails back and forth, they accepted the statement of work.

Now we were ready to go!

## **4. Creating the Release Plan**

After we all agreed to the Statement of Work, AB brought her boss to our offices to meet us, and to help create the release plan.

We did it by printing all the stories on cards, creating columns on a conference table representing 2-week iterations, and handing them the cards. I explained what Release Planning was, why it was important, and how they could help us. I talked about story points, velocity, iterations, and all the background that they needed to build the plan. I said that our teams average about 1.5 story points per person per week, so our 4 person would do about 6 unit per week, and I stopped there.

They got it in a hurry, and in a matter of a few hours had their first plan laid out from start to finish. We talked about how things might change, how they could change their mind, and so on, and they were very happy at this point.

### **4.1 The misunderstandings...**

As a very experienced coach, I should really have known better, but I got caught up in the excitement of the new project. I messed up in this first release planning session, and it had consequences for our relationship early in the project.

First of all, I set the expectation that our team would be doing 6 story points per week, because that's what most of our other teams do. As I've seen time and time again, you can't compare velocity between teams, because story points are completely arbitrary and abstract units. One team's 1-point story could be 1.5 points for another team. By setting the expectation of 6 story points per week, I had set our velocity in stone before we had written a line of code.

Secondly, I didn't mention anything at all about ramp-up time. Our plan was based on finishing 6 points in the first iteration and each iteration after that. In all my years of coaching teams, I have yet to see any team produce at full velocity in their first iteration. In fact, it is not uncommon to see teams deliver nothing or almost nothing in their first few weeks as they get comfortable with the environment, team mates, architecture, etc, and then get faster over the next few weeks. I knew this, and I assumed the client did as well, but I never said it explicitly. Had I explained it before it actually happened, I don't think our lack of delivery in the first couple of iterations would have been looked at so suspiciously.

## 5. Initial iterations and Startup costs

With all that planning stuff out of the way, we were ready to roll up our sleeves and start cutting some code! Our customers had gone back to their offices and we were left alone for a bit. They were coming back out in a couple of weeks, and we really wanted to blow them away by showing them just a glimpse of what they were getting.

Our intention was to define and develop a part of the main use case of their site and to have it ready for when they came back. We worked with them to pull out three stories that defined that main workflow, as a development team we broke those stories down into tasks, talked a lot about architecture and design, tried some things, and finally got to the point where we could create the code upon which the entire rest of our project would build. My developers tested, coded, and refactored the very best that they could during that first week, and I managed and helped out as much as I could, and we did amazing things.

But try as we might, we didn't get the stories finished. In fact, we finished about ½ of the functionality that we had promised. This worked out to 3 of the 6 story points we had originally promised for

our first iteration. I had to have a very difficult call with AB to explain all of this to her, which is when the missteps that had taken place during the release planning came back to bite my team and me.

I can remember this call clearly. This is when I explained what should have been explained earlier, that teams start out slowly as they learn their environment and system, and speed up later. It was perfectly normal for us to have under-delivered and over-promised for our first iteration, and it was likely to happen again in the next week or so as we got our feet under us. It was very clear from her voice that AB didn't know what to make of this conversation, as it must have been eerily similar to conversations she had with her previous consultants. All I could do at this point was to promise as much transparency as I could, so that she could see our progress in real time any time she wanted, to avoid future surprises. This represented the lowest point in our relationship, and we were going to have to do something to build up trust with AB, or this project was going to be miserable for everyone involved with it.

## 6. A quick introspective

I thought it would be a good idea, at this point in our project, to take stock of where we were, consider the issues that we faced, and think about how to proceed, keeping the fundamental principles of Agility in mind. We needed to involve our customer in what we did, because she had to make steering decisions for us. For us to get her as involved as we needed, she had to be engaged in our process, which means that we needed to develop a shared sense of trust.

What our team needed was to make promises to our customer, keep those promises, and allow trust to grow. There was no way that trust would grow if we kept disappointing AB, so we had to do whatever we could to always live up to what we said. Doing this, however, required some changes to the standard XP-ish agile practices that I had been coaching and using for years. These are the changes I proposed to make:

### 6.1 Estimates would be viewed as promises

Usually estimates are viewed as best guesses about how long things will take. Teams make estimates for all their stories in an iteration, and if they're wrong about something, they tell the customer, and they decide how to move forward. To do this, however, a team must have the trust of their customer. We didn't have that. So we had to view our estimates for stories and for iterations as promises and do whatever we needed to deliver on what we promised. Usually this

involved working overtime every couple of weeks to finish something we couldn't during the regular week.

## 6.2 Avoid raising estimates and splitting stories

This was probably the most difficult adaptation that we had to make. We had made estimates while creating the initial bid, before we had written anything. AB viewed the estimates as commitments. This put us in a tough spot, because we were learning more about our system, which caused us to re-evaluate how difficult certain stories were – some became easier and some became harder. There was no way that we could prevent estimates from changing, but we had to manage these changes very carefully and be able to defend why they were changing. Had we already gained her trust by delivering on our early promises, this wouldn't have been such an issue.

## 6.3 Bug fixes were our problem, not hers

The customer was very sensitive to bugs and the time spent fixing them. The previous consultants spent a lot of their time fixing bugs, which caused functionality to slip. We had to take extra care to keep our quality high at all times without affecting the functionality that we promised to deliver.

We did a good job managing this, but due to the cross-browser needs of our system and large amounts of javascript, several bugs did creep in. As a team, we decided to work a Saturday every now and then to fix bugs and get our quality back up where it needed to be when it started to drift a bit.

## 7. Iteration Steady State

We'll pick up the story here, after the first two iterations are finished. We finished the client's major workflow, showed it to her, and she was thrilled. This quick win allowed us to gain back a bit of the trust that we had wasted over the past couple of weeks.

However, things were not all roses yet. According to our own numbers, we were still only producing 3 story points per week for the first two weeks, which was  $\frac{1}{2}$  the promised rate. While planning for this project, we had kept in a few weeks for unforeseen events at the back end of the project, and we were already eating into this fund pretty seriously after two weeks.

I kept trying to sell that our velocity would improve as we got our feet under us, but I'm sure it sounded like I was just putting a good spin on a bad situation, rather than an explanation of what I expected to

happen. Again, if I had explained this much earlier, this could have been avoided.

As I look back to subsequent iterations, from our third to where we are now (iteration 17), I can see that the level of trust between AB and us has gone up exponentially. This next section will discuss how changing our practices, as described previously, and consciously avoiding the mistakes of our predecessor, allowed us to build this trust.

## 7.1 We met our commitments

We did an excellent job in meeting our commitments. Our iterations are one week long, with iteration planning on Monday morning. With only a couple of exceptions, we've hit our promises each week. AB's trust in us visibly grew each week, as she saw her site taking shape before her eyes.

To let us consistently meet these commitments, we did have to stretch ourselves every now and then. There were some weeks when the team worked a good chunk of overtime to implement some feature that grew once we got into it. We couldn't afford to go back to AB and tell her that couldn't finish some part of our iteration, so we had to get it done.

Sometimes this rush to finish got us into trouble. As happens when teams get rushed, testing is the first casualty. When rushing, we created more bugs than we otherwise would have, which created the need for more overtime to fix issues we created. This rush/bug-creation cycle served as feedback to us later in the process, causing us to think more carefully about what we would commit to in iteration planning.

The end product of this commitment to finishing was well worth it in context of the overall project. By delivering what we promised, we gained trust that we used later as currency in several sticky situations.

## 7.2 Our velocity did increase

As promised, our velocity did start increasing. We went from 3 in the first 2 iterations, to 4 the following week, to 5 the next week, and we've been hovering around 5.25 since then, plus or minus a bit. We never did make it up to 6, but it hasn't been an issue as our progress has been fast enough.

The downside of this velocity increase, however, is that it was slightly artificial. To get our velocity higher earlier in the project, we had to work more overtime than we would have liked. We did it to complete promised functionality and to fix bugs when they happened. It got better as we went through the project and our true velocity rose to the needed levels, but this wasn't the case early on.

### 7.3 Constant communication

We truly wanted AB to know everything was happening with our team. It was important to us that she know about our progress during the week, any problems or challenges that we were facing, and what victories we had. I jokingly referred to this as “seeing the inside of the sausage factory”, and it made her a little nervous for a bit. Through our conversations and our internal mailing list, she could follow our technical conversations as we found and faced challenges, discovered bugs, fought over implementation details, and did the other things good teams do.

I believe, however, that by letting her see us in all our glory we became more human to her, and easier to relate to. As we became people to her, and she became more real to us, it was easier for us to work as a team. This added to our sense of trust.

### 7.4 Regular delivery

Finally, we held her confidence by showing her the only metric that truly matters on a project, Running Tested Features [2]. Starting around the fifth or sixth week, we got access to a server on her site where we could deploy our system several times per week. This freed her up to play with the site that we were building at will. Since we weren't guiding her through it, she could experiment, try new things, find bugs, and tell us what she found.

We found that by giving her free access to the site as it was being built, she became more excited about the progress and more involved with the planning and steering.

## 8. AB's Maturation as a customer

During the course of the project, as her sense of trust grew, AB saw more of how a true agile process worked. We continued to learn how to talk and work together, helping us grow closer as a team. Once we had this sense of trust, AB could afford to relax from some of her earlier suspicions and learn to use the process to her advantage. As she matured as a customer, we were able to relax a few of our changed practices and come back to a more open and free agile process.

### 8.1 The power of iteration planning

Perhaps the best change that has happened is that our iteration planning meetings have become a powerful tool. As we've gained trust, we've had more

freedom to change estimates and even split stories when they get too big. Since she has confidence that we'll be able to finish what we promise, and we're working at a rate that shows us finishing inside her timeline, she's OK with it. Over time, we've shown how we're more successful in estimating and completing more, smaller stories, rather than fewer, larger stories, so she encourages us to split big stories into smaller pieces, or even tries to split them for us. Best of all, she truly gets the idea of a “tax” paid on the first of several similar stories, and is willing to let us accept the first of a group in one iteration, with a higher estimate, knowing that the rest of the stories will come soon after, along with lower estimates.

### 8.2 Accepting bad news

Even on a team as successful as ours, there have been a couple of iterations where we've just blew it. We promised to do some set of user stories, and for some reason or another (vacations, entire team sick, or just blew our estimate) we couldn't finish. As project manager, it is my duty to talk to AB and tell her about these setbacks.

I admit, I always feel nervous calling her to break this kind of news, but she has always taken it in stride. In the case of us just missing our estimates, which has happened once or twice, we just replan our next iteration and carry on. I feel as if these disappointments burn some of the trust that we've built up, so I try to motivate the team to avoid them, but they have popped up now and again.

The team did have the occasion recently to use up a bit of the trust that we had built up. This was the only time in the project that this happened, but we decided that some major portion of our system needed to be rewritten to improve performance and maintainability. We asked for 1/2 of an iteration to be dedicated to that, with the promise that it was being done for the overall health of the site. AB agreed to let us have this time, but I could tell that it was a bit uncomfortable. We were running up against the delivery date of the system, and this was pushing our final bits of functionality dangerously close to that end date.

We worked long and hard that week, converting lots of C# middle tier code into object-oriented javascript, but we weren't finished at the end of the week. The team told me that it needed another week of working on the performance enhancements and refactoring to complete it. We talked about it as a team for about an hour until they convinced me it was the right thing to do.

Again, it was my job to go to AB and explain this to her: explain how we needed to take a week of *her*

time to improve the overall site, to make it faster and more reliable, to make it more easily changeable. I was nervous about telling her, and I couldn't imagine what she might say after what we asked for last week. I explained it to her, told her that the team asked me to get this for them, talked about how it would help both them and her. I could really tell that I was burning a bunch of our trust currency here. Her voice dripped with fear and uncertainty, it seemed to me. But she accepted us at our word. Even though we asked for this same thing the previous week, and were now coming back to ask for even more, even though we were weeks away from launching the site, and even though she had just publicized the launch date to her management, she trusted us. She knew that we had a good reason for asking for this, and she was willing to accept the risk that we could do these improvements and still get her to launch on time. And we didn't disappoint her.

## 9. Where we are now

As I write this, we are about to start our final functionality iteration next week. Almost everything is implemented and working, with a small number of bugs lurking for us to fix. We've licked our performance problems and are discussing final arrangements for deployment of the real site.

Our relationship with AB, our customer, is strong and vibrant. We burned a little of the trust we had gained doing things that needed to get done over the last couple of weeks, but I'm not worried. Relationships are built over a period of time, and ours is strong enough to stand a little imposition. We just have to carry through to deployment in a couple of weeks, and we'll be great friends for life!

## 10. Lessons learned

I can honestly say that I learned a number of lessons during this project. I hope to apply these lessons as I move on to other teams and other projects.

### 10.1 Trust is the ultimate power

The most valuable asset my team has is a trusting customer. We didn't have one at first, so we had to go to extra effort to grow one. The process through which we went to get there didn't matter, but we had to create an atmosphere of trust.

Once we had that trust, we were much better able to negotiate with our customer as issues came up, and even ask for favors as we went.

### 10.2 The *right* process is context-dependent

One of the main precepts of the agile methods is that the team using a process is that process's owner. It is up to them to tailor it to work best for their situation. In our case, with the fixed-bid contract that we had and the initial lack of trust in us from our customer, we had to adjust things in a specific way to let us succeed.

### 10.3 You never get a second chance to make a first impression

The initial meetings of a project are a very delicate time. You must take special care to set expectations early on and manage those expectations throughout the project. On this project, I didn't take enough care to set those expectations initially, and it caused more distrust early in the project than we could have otherwise had.

Agile projects are their own kinds of beasts and customers must be made aware of this. They have a predictable startup curve and then settle into a steady series of iterations. Estimations should be viewed as just that, educated guesses that can and will be improved over time. By fixing estimates as promises, much of the freedom that a customer needs to steer a team is reduced, since teams cannot be as honest during iteration planning meetings. This forces the team to "game" the system in some way to promise the right amount of work. But that's not honest. To be honest, the estimates have to be subject to change as understanding grows.

## 11. Conclusion

In looking back over this paper, I may have painted a pretty grim picture. This team is actually fabulously successful, AB has grown into the best agile customer I have ever had the pleasure of working with, our quality and morale are high, and we are creating a beautiful, elegant, high-tech web site for all to see.

We have had our challenges, but we've overcome them together, as a team. We talk, we code, we share, we are a team. And this project would not be successful without any single one of them. Many thanks from me to all of them.

## 12. References

[1] Grenning, James, Planning Poker, <http://www.objectmentor.com/resources/articles/PlanningPoker.zip>

[2] Jeffries, Ron, A Metric Leading to Agility, <http://www.xprogramming.com/xpmag/jatrtmetric.htm>