Abstract

The Agile Manifesto emphasizes customer collaboration over contract negotiation. No Extreme Programming practice embodies this more than onsite customer. This paper describes our experience in adopting onsite customer in an environment where the practice was long considered impractical. We successfully overcame many obstacles to bring busy airline employees for extended periods of time to our development site thousands of miles from their homes and have them work with our team. We have realized significant gains in productivity, quality, and customer satisfaction as a result. Onsite customer has now become a standard way of doing business.

1. Introduction

Sabre Airline Solutions develops and maintains software for the global travel industry. Since 1992 the AirCrews product has evolved into a large complex system used by a broad base of airlines. Due to varying business requirements of each customer, implementation times range from one to two years. It is comparable to a Human Resource management system for a Fortune 500 company.

Despite embracing many other Extreme Programming (XP) practices, our team traditionally used extensive up-front requirements gathering, scoping studies, and proxy customers to discover and communicate customer requirements for AirCrews. Since our customer airlines are globally distributed, the logistics and apparent impracticality of bringing actual customers physically onsite had made the XP practice of onsite customer [1] an infrequently realized goal. Although we valued customer collaboration over contract negotiation [2], we simply assumed that airlines would not consent to releasing resources to work with our development teams for extended periods of time. We also had an internal expectation that the product had to be in ‘perfect’ condition before presenting it to the real customer. Thus, in-house Subject Matter Experts typically represented the customers as customer proxies. As one might expect we commonly experienced misunderstandings, rework, and gaps discovered late in development and deployment. Late phase testing and deliveries were commonly prolonged.

To increase feedback we had already adopted customer “check points” – basically developer led demos for customers of the evolving system conducted at pre-determined intervals during the development phase. These were primarily show-and-tell exercises with a motivation to show a ‘perfect’, bug free system.

2. An opportunity for change emerges

In 2006 an opportunity surfaced in a marketplace where our product had previously not penetrated. Our sales and product teams were highly driven to win new business in this key marketplace, while the client was highly driven to replace their legacy mainframe crew system under severe budget and timeline pressures. Challenged to successfully deliver the product with a strict budget and aggressive timeline, our development and product marketing team suggested we augment our existing staff with customer provided resources to assist in testing during development. We hypothesized that this would ultimately reduce the total number of development resources required for testing, and substantially reduce labor overhead. The customer agreed to this approach, and the contract was structured to require a minimum number of customer resources onsite during the development and testing phases. Therefore, our primary motivation to bring the customer onsite was to reduce costs, particularly in the area of testing.
2.1. Ready…Set…Wait

With the contract signed, the customer and project teams jointly developed a project timeline. Plans were made for the customer resources to come to our facilities following two iterations of development. Of course things rarely go as planned. The week prior to the customer arrival, we doubted the readiness of the product for customer use. We found ourselves quite apprehensive when faced with the prospect of our customer seeing the system in a less than ‘perfect’ state. Management, mindful of the schedule and inherent customer expectations, insisted that we bring the customer in anyway. The development team agreed, and marched forward as planned.

The customers arrived, and were escorted to a comfortable testing lab on the first floor of the building. Initial introductions and niceties were exchanged, and after some initial discussion, our development team returned to their office on the third floor, while the customer remained on the first floor. It’s been reported that the frequency and quality of communications diminishes with distance [3]. We discovered this to be true with just two floors between the development team and our customer. Or course heat rises, and we soon felt the heat on the third floor as the customers frustrations grew. The customers complained that they had nothing to do. Moreover, we simply didn’t know what to do with them.

2.2. We said wait!!!

Like a game of ‘hot potato’ responsibility for managing the customer onsite was tossed between testing, development, project management, and product management. Everyone was fully allocated to their tasks, and didn’t want to bear the added responsibility of bringing the customers up to speed.

In addition, when our customers did work with our evolving system, they were appalled by its incomplete and raw state. They had expected to see a clean, complete, working system. It appeared that our expectation of a ‘perfect’ system had been passed on to our customers.

The customer experience was quickly heading downhill. We called an emergency meeting to address the problem where the tone quickly became defensive. In implementing onsite customer we had exposed some important problems. Although the meeting was called to determine what to do with the customer, we witnessed raw emotion regarding a number of our practices. To realize the value of the onsite customer, we had to immediately align and agree on what needed to be done to remedy the situation.

2.3. Doing ‘MORE’ with our customer

Everyone agreed to do what we call “MORE,” and the following changes were immediately put into place:

- **Manage Expectations** - The team agreed that they did not need to expect the system to be in ‘perfect’ condition, and would not be rebuked for it being less than perfect. These expectations were communicated and agreed by both the team and the customer.
- **Onsite customer stewards** - We stopped tossing the potato: the testing team assumed responsibility for coordinating the activities of the onsite customers. Management also agreed to reprioritize other testing responsibilities to allow time to train the customers on the system.
- **Remove Physical Barriers** - The customer was immediately moved from their location on the first floor to a room adjacent to the development team.
- **Embrace the customer** - Most importantly we all agreed to adopt the customers as full members of our team so that we would all be successful.

With these changes in place, the customer experience quickly changed. We worked with our customer to better define their role which included training activities, testing assignments and collaborative design sessions with our development team. Within weeks the stereotypical ‘us versus them’ mentality common between customer and developers was replaced with simply ‘us’ as the customers integrated with our team.

2.4. Expanding the customer role

The customer became involved in a number of activities including:

- **User story prioritization** - The customer began prioritizing the sequence of User Stories for development. They optimized their testing velocity by considering the functional dependencies between User Stories. This reduced testing time as the rules, business process, and functionality were tested as cohesive units.
- **Customer as the onsite customer** - Even with the real customer onsite, we were reluctant at first to admit that we did not fully understand our customers’ business for fear that they would criticize us for failing to understand their requirements. However, with the customer in such close proximity, our development team found it
easy to directly approach the customer for clarification. Our customers and developers worked closely together - solving issues at the white boards, and discussing User Story details throughout the iteration. We forged personal relationships with our customer, which in turn developed a new sense of trust. Thus, our team demonstrated an increased willingness to ask “how is this supposed to work?”

- **Acceptance Criteria Authors** - Who better to write acceptance criteria, than the customer? Replacing the customer-by-proxy with real customers improved the quality of our acceptance criteria. This reduced the amount of rework in development as well as our long-term testing needs during final User Acceptance Testing (UAT).
- **Expert Training** - Normally, we provide user training just prior to product implementation. User training was incremental and training sessions were held daily. As a result of the intensive, in-depth sessions between our customer and our development team, our customer rapidly became experts on the system far in advance of delivery. Later, this greatly reduced the cost and effort to deploy the system.

### 2.5. The daily routine

As the entire team became more comfortable and enthusiastic about this approach and opportunity, we established a daily routine. A typical day for the customer team members included the following activities:

- **Brief the team** - Because our stand up meetings include discussion of activities and issues impacting multiple customers, we held a separate brief meeting with the development team and the customer to discuss status and issues.
- **Test the build** - The customers tested the system daily using the daily builds. This increased the feedback to the development team allowing for more rapid resolution of issues. As the customer encountered issues, they simply contacted the developer accountable for the User Story and clarified the issues directly. The developers then corrected the issues and typically within several days, the change was available for test.
- **Discuss User Stories** - The customers and developers had daily conversations to clarify User Story details.
- **Author acceptance criteria** - Customers continued to develop and refine acceptance criteria.
- **Author test scripts** - Customer team members spent time developing test scripts for use during their own onsite UAT.

### 3. Cost analysis

We were apprehensive about the upfront cost and time spent with the customer. Typically, a developer could easily spend two to three hours per day working with the customer. Our testing resources often spent up to four hours a day working with them. However, the reduction in downstream costs more than compensated for the initial upfront cost. A number of factors enabled us to significantly reduce cost:

- **Scope** - A common concern about bringing the customer onsite is that scope will be increased. We very often observed the opposite effect. While developing designs and early prototypes of new features the customer would ask us to simplify the functionality. This often reduced scope and unnecessary work!
- **UAT duration** - The in-depth and comprehensive testing performed by the users in our office, reduced the time necessary to test the system after deployment at the customer site. First, there was minimal ramp-up time for the users, as they were already experts.
- **UAT issues** - The users understood the system in much greater depth and knew how to make the most of available features. As a result, we experienced far fewer UAT problems related to user comprehension.
- **Rework** - We observed a 60% reduction in the amount of functional rework. We attribute this, in part, to the daily conversations around User Stories and acceptance criteria.
- **Post-installation work** - We have found that requirement gaps are a key culprit in incremental cost and schedule delays on projects. Discovering these after, rather than before, installation further increases costs. The intensive collaboration with our customer onsite across the iterations reduced the number of gaps uncovered after release by 30%.

Overall, customers understood and empathized better with the issues faced by the developers. Similarly, the developers understood and empathized with the customer’s needs. The inevitable negotiations of scope became relatively painless and more
collaborative. Working within the contract and the initial requirements was conducted in the spirit, rather than the letter of the law.

4. Other Benefits of onsite customer

Our initial expectation for onsite testing was that costs would be reduced and we would deliver a better product. We also experienced many other very positive side-benefits.

- **Fewer development resources were needed** - In previous projects, late phase testing often became squeezed due to deadline pressures. On those occasions we pulled resources from all areas including developers, product managers, even an occasional vice president to address the testing demand. We often joked that if you were an employee who had read a product brochure you were a candidate for testing. Working closely with our customer the test squeeze was eliminated. As a result fewer resources were required.

- **Increased innovation and business value** - Our testers and developers gained a much better understanding of the system and our client’s business as a result of the close collaboration with our customer. This improved the efficacy of both our testing and development. With a grasp of real world business scenarios, the team was able to see connections that lead to greater product innovation. This has had a profound effect on the future direction of our product.

- **Positive impact in cutover phase.** AirCrews is an operation critical real-time application. We always encourage customers to staff additional resources for a number of weeks during cutover. For this implementation our customer was under budget pressure to curtail such staffing. With the improved quality of the implementation, the reduced staffing was adequate.

- **Stronger relationships** – Working together throughout the project forged a strong relationship with our customer based on earned trust and respect. As a result, during the crucial cutover phase, we were able to resolve issues much more quickly and amicably.

5. In Retrospect

Ask anybody on our team if they would use the onsite customer again, and the response is a resounding ‘yes’! Both our customer and the development team have requested that onsite customer be adopted in all future significant development activities. Onsite customer involvement is now included in all AirCrews contracts, and is considered a critical success factor for any customer engagement.

In fact, our customers are now requesting this of us. A recent customer has already gladly volunteered to increase their resource commitment to support onsite development by 25% over the amount specified in the contract.

5.1. Recommendations

- **Set clear expectations** - Expectations must be set and communicated to both your internal team and your customer. Going forward we will better educate our customer and the development team on what to expect onsite including: the state of the system, project roles, release plans, and the work environment.

- **Set expectations early** - Some of the most critical work for success must be done before the customer arrives. Get commitment from the customer early in the project and renew this commitment regularly.

- **Simple things count** - Ensure that the customer team has a place to work, preferably in the same area as the development team. Provide access to the system, and schedule training sessions for the users. Develop an agenda that both the customer and development team members can follow. Agendas are also useful in clearly communicating expectations between the customer and the development team.

- **Involve the customer in all phases** - Don’t limit customer involvement to only testing. Work closely with the customers from the start of the project. As our team discovered, the customers are key in assisting with User Stories and acceptance criteria communication.

- **Don’t forget your other customers** - It would be convenient for us to have a single customer or project to focus on at a time. However, this luxury does not exist. To be successful in our business we must balance the priorities of multiple customers. Management must clearly establish and communicate priorities and assign the resources so that all the work can be completed, and not just for the customer we can “see.”

- **Leverage the global team** - Our development teams are distributed globally. This distribution enables our customers to work with the development team closest to their home.
6. Conclusions

In the airline software development business the XP practice of onsite customer can seem impractical. Asking customers to take time out from their normal activities to collaborate with developers can be a hard sell. Nevertheless, we have found that bringing real customers onsite to work with our development team greatly reduces costs, improves product quality, and delivers higher value for our customers. There was a significant reduction in the number of errors related to business requirements. We estimate that rework was reduced by over 60% for this project. Our customer also experienced savings in shorter times to test, train, and implement the system. Our customers have appreciated the experience and have expressed their enthusiasm about the results.

As a result of the success of this project, we have made the inclusion of the onsite customer a standard way of doing business with our customers. Nowadays if a customer claims they can’t afford to provide resources to work with our developers, we simply explain why they can’t afford not to. The benefits of using the XP practice of onsite customer for our projects far outweigh the costs.

7. References

