Towards an Effective Onsite Customer Practice

Cesar Farell **Rekha Narang**

Intelliware Development Inc.
1709 Bloor St. W., Suite 200
Toronto, Ontario
M6P 4E5, Canada
+1 416 762 0032
{rekha, cesar}@intelliware.ca

Shelley Kapitan **Heather Webber**

Celidon Inc.
319 Ontario St.
Toronto, Ontario
M5A 2V8, Canada
+1 416 929 3521
contactus@celidon.ca

ABSTRACT
This paper describes our experience of process refinement during a long running XP project. At the end of the first release, the team critiqued their practices and produced a concrete set of recommendations that were put into practice in later releases to address the identified problems. The greatest impact on process improvement came from those recommendations that pertained to refining the onsite customer role. This paper discusses the improvements that resulted in a more effective onsite customer practice.

Keywords

Extreme Programming, Onsite Customer

INTRODUCTION
The Extreme Programming (XP) practice of "onsite customer" is described as a real, live user on the team, available full-time to answer questions [1]. The customer is the business representative who is part of the development team, whose role is to direct the team's efforts according to the needs of the business by writing stories and acceptance criteria, planning the release, and adjusting the course of the development throughout the release.

The concept of onsite and active customer participation is not difficult for the XP practitioner to accept intellectually. Putting it into practice is another matter: it is difficult for both the customer and the developer. Customer experience in the past has consisted of detailed specifications, sign-offs, steering and progress meetings at intervals and a long wait for the delivered system. Their position mandates focus on business matters – rarely do these mandates imply or even allow for active involvement in the development of the systems needed to support their business processes. Developers, on the other hand, are accustomed to rare, and often less than congenial, appearances by their customers, who are often in state of mild to severe impatience with the development process. While XP addresses these issues, its practitioners have to find their own way. This paper explores the practicalities of putting that concept into place.

The Project
The project was interesting: building a system to support a new and unusual business venture, and using technologies unfamiliar to the developers. The developers assigned to the project were intrigued by the business concept and the technical challenges.

Business requirements were already defined at a very high level and customer representatives were appointed to flesh out the requirements and manage the overall project for the ultimate client. Because of the anticipated breadth of the full system, the first release was shaped to provide certain functionality within a three-month development period.

The Customers
The two onsite customer representatives were engaged by the ultimate customer to define the business processes and system requirements on his behalf, and to manage the project through to successful implementation. Both have systems development backgrounds, and have been in IT management and consulting for some years. They also had recent experience with agile development methods, so the concept of the "customer in the room" was acceptable and was, in fact, deemed by them to be the only viable way to define and develop this loosely defined system.

The Developers
The development team had responsibility for technology and risk assessment, product development, and testing. They worked closely with the customers to define key system functionality and advise on technology decisions. They had significant experience in iterative development and a proven track record of successful projects involving emerging technologies. They had adopted XP six months prior to the project and were comfortable with the majority of its practices. This project, though, was their first opportunity to work with an onsite customer.

The Result
The first release was considered a success from the customers' perspective, despite the need to suddenly reduce functionality in its later stages. Afterwards, customers and developers met to review the project and identify opportunities for improvement in the development process for following releases.

This paper describes how the authors and their teammates applied these lessons to establish an onsite customer role that encouraged a collaborative, trusting and effective XP development process.

PROBLEMS IDENTIFIED DURING THE PROJECT REVIEW
Areas identified for attention were: poor integration of the customers into the development team, lack of experience in using stories effectively, lack of direction with

respect to story closure and acceptance criteria, and, at times, poor communication between customers and developers.

Customer and Developer Team Integration
For the first release, the customers visited the development site for a weekly status meeting with the customer lead (a team member charged with tracking project status and coordinating project efforts) and a second developer. Its purpose was to discuss project status (velocity, budget), present questions to the customers, and update the development team on new business developments. While the customers were willing to be on site more often, the team did not take up this option.

This formal interaction led to a split along development and customer lines, where information was exchanged mostly through the weekly meeting and occasional email. The result was that, despite the fact that both developers and customers shared the same goals, they behaved as separate entities with distinct concerns. For the customers, the delivery of functionality, scheduling, quality and budgeting were paramount; for the developers, the coding, design, and quality were most important. The split was so marked at times that the developers felt that they had to protect their interests against those of the customers. Some developers were uncomfortable having the customers in the project room, which made the customers feel unwelcome.

Using Stories and Defining Acceptance Effectively
Intelliware uses stories to direct project development as a normal practice, but this method of defining high-level requirements was new to the customers, who were not given sufficient introduction to the purpose and the content of good stories. Instead of the customers writing stories, the developers merely consulted with them on required functionality and wrote the stories themselves. The customers did not feel any natural ownership of the stories.

The development team had significant difficulty closing several of the stories during the first few iterations. This was due in part to poorly written stories: several relied on external technical dependencies that were outside the control of the team, and some of the desired features were spread across more than one story. While waiting for technical bottlenecks to be resolved, the developers opened other stories and this negatively impacted velocity. In fact, the velocity fell to zero while the team continued to work diligently on tasks, and with it, morale fell.

Another early difficulty was the lack of detailed acceptance criteria or tests. Because the customers had not been informed of the need nor pressed for such information, the development team made assumptions about what the criteria should be. As a result, story acceptance occurred through an inefficient process of trial and error on the part of the developers until the customers were satisfied.

Several iterations passed before the customers had the chance to redirect the team's efforts to technical problems that they could solve and deliver the required functionality.

Project Status and Steering
Ineffective story closure eroded the team's faith in the value of tracking velocity and planning game techniques. It was clear that the story velocity was inconsistent with the number of tasks they were doing, so they began to pay less and less attention to the project status.

The development team was apprehensive about discussing this apparent lack of progress with the customers since they believed they would neither understand nor appreciate the reasons for the difficulties. The customers were not fully apprised of which tasks were being addressed, what technical challenges were being faced, and what code quality concerns there were. In fact, as the customers shared these quality concerns, they would have been very receptive to a discussion of the challenges.

As the last iteration for the release approached, it was obvious that the functionality target could not be reached. A significant scope reduction was required and this was a shock to the customers, who had expected velocity to improve as the team became more familiar with the application and as the code base stabilized. Excluding them from the process had precluded the opportunity for them to adjust project plans as the release progressed. It was only at this point that the customers insisted and were permitted to work closely with the developers to review and reduce functionality, and deliver a viable product.

RECOMMENDATIONS FOR IMPROVEMENT

After openly discussing their collective difficulties during the first release, the developers and customers redefined the role of the customer. Their recommendations were:

Customers and Developers Work As a Single Team
Customers will be on site full time and attend all developers' meetings, including daily stand-up meetings, tasking sessions and design reviews. Customers are welcome to work in the project room and will be assigned story tasks as appropriate. Developers will keep the customers informed of their day-to-day status and concerns, and a joint approach will be taken to problem resolution.

Customer Ownership of Stories
Customers will either write the stories or will develop them in conjunction with the team, and will define acceptance criteria through specific test cases. Developers and customers will collaborate to ensure that the completed stories pass the tests and customers will clearly define when stories are to be closed. The whole team (customers and developers) will celebrate story closure.

Improved Tracking and Steering Mechanisms
A status report card showing story progress, effort, and estimation accuracy will be created, and will be viewable over the project's intranet site. Customers and developers will review weekly and jointly adjust the release plan, add or remove functionality, or rewrite stories as necessary.

POST-IMPLEMENTATION OBSERVATIONS

These recommendations were put into practice in the second release, and, by its end, had been adopted completely. All agreed that the process was vastly improved,

and so it was applied in subsequent releases, which included the addition of a second development team.

Process Reliability
Project status data demonstrated improved and more reliable development processes after the introduction of the more involved customer role. Specifically, the occurrences of zero velocity iterations and the number of stories that spanned iteration boundaries decreased dramatically.

During the first release, two of the eight iterations finished with no new stories completed (a velocity of zero). Since planning is based on "Yesterday's Weather" [3], an iteration with a zero velocity is disruptive to the process and damaging to team morale. After the changes, only one other iteration (out of fifteen) finished with a zero velocity.

Run-on stories, or stories which are not finished by the end of an iteration, are normal in an XP project. However, we found it to be a problem when there were more than one or two per iteration: it indicated that stories were not being closed effectively or that too many stories were open at once. This meant that the team's efforts were spread too thinly, and the momentum of story completion diminished. Partially completed stories also presented problems in planning the subsequent iteration.

The first release had six run-on stories out of fifteen; after introducing the changes, there were none. This correlates well with the disappearance of zero-velocity iterations.

Story Ownership
A marked shift in story ownership occurred. Previously, the development team guarded its ownership of stories and used them to present the project status to the customers. Developers would close a story on completion of its last task without consulting the customers. After implementing the recommendations, customers took on a direct role in writing and closing stories. They defined acceptance criteria, reviewed acceptance tests with the developers, took on appropriate development tasks, aided in prioritizing design refactorings, and decided when stories were closed. Ownership of the stories shifted noticeably from the development team to the customers.

Project Planning
Project planning became smoother and less erratic. As new information came available (after investigation of technical risks or clarification of business requirements), estimates would be adjusted. The impact on the release plan could be assessed immediately and a "steering" response could be made. In effect, a continual project navigation process emerged, which was fluid and easy, unlike the sobering and unwelcome surprises of the first release.

Shared Concerns
As day-to-day efforts of customers and developers became more integrated, a shared understanding of everyone's concerns emerged. This led to a unified effort to deliver successful releases, with a level of trust unknown in previous projects. The process was enjoyable for all and the sense of momentum grew. Two separate and potentially conflicting efforts became a single effective collaboration.

DISCUSSION
The project review and subsequent implementation of its recommendations dramatically changed the way developers and customers worked together to meet the business's software requirements. Later release efforts were more effective and much more enjoyable than the first. While the modified onsite customer role was not the only difference, we believe it to be the primary contributing factor to improved process effectiveness. From this experience, we developed a better understanding of the role the onsite customer should play.

That role, stated simply, is to represent the business and collaborate with developers to deliver enough functionality on time. Ideally, the duties are to: represent the business interests, act as a conduit between business and developers, write stories and acceptance criteria, provide "steering" direction (prioritize stories and adjust scope), and make business decisions. It is important that onsite customers understand that they are team members, not team auditors.

An effective team member on an XP project must be someone who enjoys collaborative efforts, and who is prepared to be available to team members to answer questions, to help with problem solving, to be open-minded, honest, objectively critical and respectful. These qualities obviously foster effective communication and trust. Some customers may find this collaboration difficult due to a perceived conflict with their role of representing the business, but when successful, the right customer representative can greatly improve the team's productivity and maximize the return for the business investment.

Other Factors
The project review recommended other process refinements: a tracking tool for large refactorings, a project status report card, regularly scheduled developer status meetings, and better defined meeting agendas. Each was effective in eliminating a particular rough spot, but contributed to a lesser degree than the elimination of communication barriers between customers and developers.

The technical nature of the project changed over time and this also may have influenced the apparent improvement in the development process from one release to the next. The stories of the first release were technically risky and presented challenges in an unfamiliar business domain. Subsequent releases built on the work of the first, and could be considered less risky. Nevertheless, later releases did include their share of technical and business issues that required a high degree of customer direction. The improved team performance cannot therefore be attributed to this apparent reduction in technical complexity.

Previous Work
Wake [2] describes the role of the onsite customer in terms of concrete duties as part of the development team. His description is consistent with the high level definition that we arrived at through this experience. In particular,

he stresses the importance of reducing the communication overhead by having the customer onsite.

Newkirk and Martin [4] provide a detailed account of an XP software project where problems were encountered due to the absence of an onsite customer. They give an example of a misunderstanding between the customer and the developers which could have been resolved more readily had the customer been on site. As a result of the experience, the customer recognized the importance of having an empowered customer engaged in the process. This realization closely resembles our own experience.

Griffin [5] describes a successful first-time XP implementation from the customer's perspective. The customer reaped the benefits of better steering capability and a higher level of communication by being on site. Again, we find this consistent with our experience.

Partial Implementation of the Onsite Customer Practice

According to our onsite customer definition, it is critical to have a high degree of customer involvement in the process. The reality, though, is that customers are often unable or unwilling to spare the people or sufficient time for such a commitment. How, then, does one proceed when the customer is less than ideally involved?

A first response is to try to convince the customers that the onsite business representative is well worth their investment. Wake [3] argues that the flexibility and responsiveness benefits of XP justify the high-bandwidth communication overhead of the onsite customer practice. Without this, the consequence is inefficient communication between developers and the customer, and a loose feedback mechanism that results in less responsive steering and more team effort to get back on course.

It is clear that the development process is more effective when developers and customers trust each other and have a shared set of concerns. Having the customer onsite facilitates the building of this trust for several reasons: the familiarity of day-to-day interaction fosters team camaraderie, gross misunderstandings and omissions of information occur less frequently, and a shared work space helps develop a greater appreciation and respect for everyone's efforts. Conversely, in a project with an offsite customer, all involved parties need to make an especially concerted effort at building and maintaining this trust.

While Intelliware has completed many XP projects, most have not had an onsite customer. These projects were delivered successfully due to the compensating efforts of the development team, but development was hampered by having to contend with the communication and trust issues inherent in these situations.

CONCLUSIONS AND FUTURE DIRECTIONS

Over the course of this project, we learned how to improve our development methodology. We reviewed our existing practices, identified areas of concern, and recommended changes to address those concerns. Once we implemented those changes, we found that the revised set of practices alleviated the identified problems.

The most beneficial recommendations were those pertaining to the onsite customer practice: the integration of day-to-day activities of customers and developers, customer ownership of stories, and customer-led steering and decision-making. We recognize this is only a stage in the evolution of our development practice, and as we improve our development methodology, we will refine these practices. Questions for future exploration include:

- How do we convince customers of the value of the onsite customer practice?
- How do we better introduce and educate the onsite customer to their roles?
- How do you integrate customers into the team as quickly and smoothly as possible?
- How can the process be adapted to work with a committed part-time onsite customer?
- How does our experience on this project compare with other projects' experiences?

REFERENCES

1. Beck, Kent. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 2000.
2. Wake, William. C. *Extreme Programming Explored*. Addison-Wesley, 2002.
3. Beck, Kent and Fowler, Martin. *Planning eXtreme Programming Explained*. Addison-Wesley, 2001.
4. Martin, Robert C. and Newkirk, James. *Extreme Programming in Practice*. Addison-Wesley, 2001.
5. Griffin, L. Ann. A Customer Experience: Implementing XP. *XP Universe Conference Papers.* (Raleigh NC, July 2001)