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 the experience of process refinement on a long running XP project. At the end of the first release of this project, the team took the time to review and critique their current practices. The outcome of this review was a concrete set of recommendations that were put into practice in later releases to address the problems identified in the first. Of these recommendations, the ones that had the greatest impact on process improvement were those pertaining to the refinement of the onsite customer practice. This paper discusses in detail the improvements experienced as the result of a more effective onsite customer practice.

Keywords
Extreme Programming, Onsite Customer

1 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. Her or his 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 lifetime.

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 the customer and difficult for the developer. From past experience, customers have been used to detailed specifications, sign-offs, steering and progress meetings and a long wait for the delivered system. They have position mandates that focus on business matters – rarely do these mandates imply or even allow for active involvement in the development of the systems needed to support those 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. XP addresses these issues, but its practitioners have to find their way. This paper explores the practicalities of putting that concept into practice.

The Project

The project was an interesting one: building a new 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 loosely 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 complete system, the first release was shaped to cover a certain amount of functionality within a three-month development cycle.

The Customers

The 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 from his perspective through to successful implementation. They both had systems development backgrounds, although they had been in IT management and consulting for some years. They had some knowledge and involvement with agile development methods in the recent past, so the concept of the "customer in the room" was acceptable and was, in fact, anticipated by them as the only way to define and develop this loosely defined system.

The Developers

The development team took on the roles of technology and risk assessment, product development, and testing. They worked closely with the customers to define the key system

functionality and advise the business on technology decisions. The developers brought with them significant experience in iterative development and a proven track record of successful project deliveries involving emerging technologies. They had adopted XP six months prior to the project and were quite comfortable with the majority of its practices. However, before this project, they had not had the opportunity to work with a full-time onsite customer.

The Result

The first release was considered a success from the customers' perspective, although functionality had to be suddenly reduced in its late stages. Subsequently, the customers and developers met to review the project and to identify opportunities for improvement in the development process for the following releases.

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

2 PROBLEMS IDENTIFIED DURING THE PROJECT REVIEW

The review identified the following areas for attention: poor integration of the development team and the customers; lack of experience in using stories effectively; lack of direction with respect to story closure and acceptance criteria; and, at times, ineffective communication between customers and developers.

Customer and Developer Team Integration

For most of the release effort, the customers visited the development site once a week when a status meeting was held with the customers, the customer team lead (a developer on the team charged with tracking project status and coordinating project efforts), and a second developer. The purpose was to discuss project status (velocity, budget), present questions to the customers, and update the development team with new business developments. It should be noted that the customers were quite willing to come on site more often, but this option was not taken up by the development team.

This formal and scheduled interaction led naturally to a split in the team along development and customer lines. Rather than on-going direct communication between individual developers and customers, information was exchanged mostly through the weekly status meeting and the occasional email. The result was that, despite the fact that both developers and customers shared the same goal, they behaved as separate entities with distinctly separate concerns. For the customers, the delivery of functionality, scheduling, and budgeting were paramount; for the developers, the coding, design, and quality were most important. (The customers also shared the concern about quality.) The split was so marked at times that the developers felt that they had to protect their interests from those of the customers. Ultimately, some developers were uncomfortable having the customers in the project room and the customers subsequently felt unwelcome there.

Using Stories and Defining Acceptance Effectively

The use of stories to direct project development is not a new practice at Intelliware, but this particular method of defining high-level requirements was new to the customers, who were not given enough introduction to the purpose and content of good stories. Instead of the customers writing the stories, the developers merely consulted with them on required functionality and then wrote the stories themselves. Consequently, the customers did not feel a natural ownership of the stories.

The development team had significant difficulty closing several of the stories during the first few iterations. The difficulty was in part due to poorly written stories: several stories were found to rely on external technical dependencies that were beyond the control of the development team and some of the desired features were spread across more than one story. While waiting for these technical bottlenecks to be resolved, the developers opened more 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, the morale fell.

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

Several iterations passed before the customers had the opportunity to redirect the developers' efforts towards technical problems that they could solve and deliver the required functionality.

Project Status and Steering

The lack of effective story closure eroded the development team's faith in the usefulness of tracking velocity and the planning game techniques. It was clear to them that the story velocity was inconsistent with the amount of work they were doing in terms of development tasks. The developers began to pay less and less attention to the project status.

At the same time, the development team began to feel apprehensive about discussing the lack of apparent progress with the customers since they believed that the customers would not appreciate the reasons for their difficulties. The customers were not fully aware of which tasks were being addressed, what technical challenges the developers were facing, and what concerns they had for code quality. In fact, since the customers shared these quality concerns, they would have been very receptive to a discussion of the

challenges.

Finally, as the last iteration for the release approached, it became obvious that the functionality target would not be reached. A significant scope reduction was required and this came as 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 the customers from the process had precluded the opportunity for the business people to adjust the project plan as the release progressed. It was at this point that the customers insisted and were permitted to work hand-in-glove with the developers to review and reduce functionality, and permit a viable product to be delivered.

3 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
The customers will be on site full time and will attend all developers' meetings, including daily stand-up meetings, tasking sessions, and any design reviews. The customers are welcome to work in the project room and will be assigned tasks as part of a story whenever appropriate. Developers will attempt to keep the customers informed of their day-to-day status and concerns, while the customers will be supportive to the developers in this regard.

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 acceptance test cases. The 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 over time, amount of effort, and estimation accuracy will be created. The team tracker will update the report card daily and the card will be viewable over the project's intranet site. Customers and developers will review the report card together weekly. As unforeseen issues arise, the customers will work with the developers to adjust the release plan, add or remove functionality, and rewrite stories as necessary.

4 POST-IMPLEMENTATION OBSERVATIONS

The recommendations were put into practice for the second release, and by its end, the recommendations had been adopted completely. All agreed that the revised process was a vast improvement and it was carried forward into subsequent releases and the addition of a second development team.

Process Reliability
Project status data for subsequent project releases show that the development process became more reliable after the introduction of the changes in the customer role. Specifically, it was noted that the number of occurrences of zero velocity iterations and the number of stories that spanned iteration boundaries decreased dramatically after the changes.

During the first release, two of the eight iterations finished with no new stories completed (a velocity of zero). Since the planning game is based on using "Yesterday's Weather" [3], finishing an iteration with a zero velocity is disruptive to the process as well as damaging to team morale. Following the introduction of 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 a normal occurrence in an XP project. However, we found it to be a problem when there were too many (more than one or two) run-on stories per iteration: too many indicated that stories were not being closed effectively or that too many stories were being opened at once. In these instances, the team's efforts were spread too thinly, and the momentum of story completion diminished. Furthermore, partially completed stories were found to be troublesome in planning out the subsequent iteration.

During the first release, we recorded six run-on stories out of a total fifteen; after introducing the changes, subsequent releases had none. This observation correlates well the disappearance of zero-velocity iterations.

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

Project Planning
After the first iteration, project planning became smoother and less erratic. As new information came available (developers would investigate technical risks, business requirements would be explored at a detailed level), story estimates would be adjusted on the fly if required. The impact of these adjustments on the release plan could be assessed immediately and a "steering" response could be

made accordingly. In effect, a continual project navigation process emerged. This process was fluid and smooth, and felt easy, unlike the sobering surprises that were characteristic of the first release.

Shared Concerns

As the day-to-day efforts of the customers and developers became more closely integrated, the concerns held by each became more similar. Rather than the apparent conflict of concerns and guarding of interests which were characteristic of the first release, a shared understanding of everyone's concerns emerged. Gradually, this shared understanding became a unified effort in delivering successful releases. Customers and developers established a level of trust unknown to previous projects. The process became more enjoyable for all involved as frictions reduced and a sense of momentum was felt. What started as two separate and potentially conflicting efforts became a single collaborative and effective one.

5 DISCUSSION

The project review and the subsequent implementation of its recommendations dramatically changed the way the developers and customers worked together to meet the business's software requirements. Our observation is that subsequent release efforts were more effective and much more enjoyable than the first. While the modified onsite customer role was not the only difference between the first and subsequent release efforts, we perceive it to be the primary factor contributing to improved process effectiveness. From this experience, we have a formulated a new understanding of what role the onsite customer should play in our projects.

The role of the onsite customer is to represent the business and collaborate with developers to deliver enough functionality on time. Ideally, the customer's duties are to: represent the business at all times; act as a conduit between the business and the developers; write stories and acceptance criteria; provide "steering" direction (prioritize stories and adjust scope); and make business decisions for the developers. It is important that onsite customers understand that they are not team auditors, but rather, team members.

To be an effective team member on an XP project requires someone who enjoys collaborative efforts. Such participation requires one to be available to team members to answer questions, to help others with problem solving, to be open-minded, honest, objectively critical and respectful. These qualities foster effective communication and the building of trust within the team. The customer may find it difficult to collaborate with the team in this manner because of the possible conflicting position that being the representative of the business places him or her. However, these qualities in the customer representative can greatly improve the team's productivity and, therefore, maximize the return for the business investment.

Other Factors

Aside from better team integration, the project review recommended other process refinements. These included: use of a tracking tool for large refactorings, use of a project status report card, regularly scheduled developer status meetings, and, a better defined stand-up meeting agenda. Each refinement was found to effectively eliminate a particular rough spot in the development process. However, they contributed only in a minor way to the improved experience in comparison with the effects eliminating communication barriers between customers and developers.

The change in the technical nature of the project over time also may have influenced the apparent efficacy of the development process from one release to the next. The stories of the first release were technically risky and presented some challenges in an unfamiliar business domain. Subsequent releases tended to build on the work of the first release, and could therefore be considered to have been less risky. Nevertheless, the stories for subsequent releases did include their share of technical and business issues that required a high degree of customer direction. The improved team performance observed in the latter releases 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 in which they describe problems encountered due to the absence of an onsite customer. They give an example a misunderstanding between the customer and the developers which would 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 experiences consistent with ours in regard to the importance of having the customer work closely with the developers.

Partial Implementation of the Onsite Customer Practice

According to the onsite customer definition we derived through this project, it is critical to have a high degree of customer involvement in the process. However, customers

are often unable or unwilling to spare the people or sufficient time to make such a commitment. How, then does one proceed in a situation where the customer is less than ideally involved?

A first answer is to try to convince the customers that the onsite presence of a business representative is well worth the 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. Separating customers and developers creates a loose feedback mechanism which results in less responsive steering and requires more team effort to get back on course.

Based on our experience in this project, 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, the majorities have not had a full-time onsite customer. These projects were delivered successfully due to the efforts of the development team in compensating to some extent for the customer's absence. However, development was hampered by having to contend with the communication and trust issues inherent in these situations.

6 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 recommendations that provided the most benefits to the process 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. As a result of implementing these recommendations, we formulated a concrete set of duties and identified necessary qualities for an onsite customer.

We recognize that the point we have reached is only a stage in the evolution of our development practice. As we strive to improve our development methodology, we will continue to refine these practices. Some 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)