Customer Relationships and Extreme Programming

Paul S Grisham and Dewayne E. Perry
Empirical Software Engineering Lab (ESEL)
ECE, The University of Texas at Austin
{grisham, perry}@ece.utexas.edu

ABSTRACT
Extreme Programming (XP) brings the customer and development team together into a tight functional unit, while eliminating many of the process activities of more structured software development processes. While agile methods may yield benefits in terms of product cost and quality, there is also a risk that the very practices that make agile methods effective may weaken the customer relationship. This paper examines XP from the perspective of customer satisfaction and motivates the need for more analysis of the social, psychological, and business factors in studies of software development methods.

Categories and Subject Descriptors
D.2.9 [**Software Engineering**]: Management – *Software Process Models*

General Terms
Management, Measurement, Human Factors.

Keywords
Extreme Programming, Agile Methods, Customer Satisfaction, Customer Relationships, Business Value, Quality.

1. INTRODUCTION
Advocates of agile software development methods recognize the difficulty of building complex software systems under the best of circumstances but assert that a light-weight process with small, manageable delivery units can allow a software development team to produce quality software that delivers value to the customer early in the development cycle through incremental releases and respond deftly to changes in requirements through agile management and strategy.

The *Agile Manifesto* places high priority of the interactions on the development team, including the relationship with the customer, choosing "interaction" and "collaboration" over "processes" and "negotiations" [1]. The principles behind the *Manifesto* include specific strategies for satisfying the customer through "early and continuous delivery of valuable software" and "welcom[ing] changing requirements, even late in development." Moreover, the *Manifesto* calls for daily interaction between developers and business people. Extreme Programming (XP), perhaps the most popular agile development method, explicitly defines a role for the customer in the development team [3][1].

While some observational and case studies have been conducted on software projects implementing some or all XP techniques, the results tend to be focused primarily on software, technical, or programming issues. Quantitative surveys, such as Rumpe and Schröder [21], tend to focus mainly on the developer or manager. Other studies have looked at developer effort [14], productivity [23], defect management and maintenance [20]. A few studies have considered the social factors of the development team, especially with regard to pair programming [17,7]. In this paper, we argue that the social issues of the developer-customer relationship during XP development need more analysis and discussion.

Delivering a system that satisfies customer requirements on time and within budget with few defects is the ultimate goal of any software development activity. To this end, software development processes provide a method for capturing and validating user requirements, estimating costs, monitoring progress, meeting project milestones, and providing quality assurance. Since all of these factors affect the perceived success of a software project – and subsequently the overall customer satisfaction – processes that can regularize requirements, cost, and quality, are generally perceived positively by customers. For organizations that have mature, repeatable processes (*e.g.*, such as those certified CMMI or ISO 9001), their process is one of their products and is marketable as such. U.S. Government and Department of Defense contracts regularly mandate process requirements.

There are, however, some perceived shortcomings with highly structured processes. For instance, the overhead involved with documentation, bureaucracy, and process feedback is a non-negligible cost[2]. Lengthy requirements elicitation and design analysis phases mean that developers do not start delivering production code until well into the process, perhaps too late to provide meaningful feedback to the customer. Most formalized processes allow the customer some authority to request requirements changes to meet evolving business requirements during the process, though actually changing requirements may involve contract renegotiation.

---

[1] For up-to-date discussion of the current state of XP practice, please visit the XP Wiki at: http://www.c2.com/cgi/wiki?ExtremeProgrammingRoadmap

[2] ...though process advocates would argue that the cost of process support is far less than the expense of building a system that is incorrect or late using an undisciplined process.

Each of these issues is specifically addressed by the *Agile Manifesto*, and specifically by the core practices of XP. While process advocates and agile advocates could trade evidence about the advantages and shortcoming of each method of developing software, this paper approaches the problem from a different perspective – specifically, the perspective of the customer relationship with the development team and customer satisfaction with the process itself.

In this paper, we examine XP and how it involves the customer in the development process. We describe software development as a service industry and attempt to define the customer relationship in terms of how the customer perceives that the development process is meeting specific business needs. Finally, we suggest some methods for evaluating customer satisfaction of XP and call for more case studies of software development methods to include customer satisfaction as a measurable result.

2. CUSTOMER SATISFACTION

Most definitions of customer satisfaction in the software and systems engineering field focus on the major factors of requirements, cost, design, and quality – *i.e.*, satisfying defined customer requirements, completing the system on-time and within budget, managing the inherent complexity of the problem, and producing a system with a minimum of defects. For this reason, software engineering research tends to focus on requirements, cost, design, and quality. The twelve core practices of XP are also designed for that purpose.

In this paper, we suggest a definition of customer relationship that focuses on the relationship between the customer and the development team, one that can be characterized by a simple question:

Given the choice, would your customer do business with you again?[3]

This question suggests many different constituent factors. While it is important for the development team to deliver a working system within a reasonable time frame for a reasonable cost, this definition is based on the *perception* of the customer of a specific organization's process and how it affects the software product. In this way, the customer relationship can be defined as the level of customer satisfaction with a specific organization's process.

Traditionally, the practitioner's approach to advocating a specific development practice is to show how that practice positively affects the factors of requirements, cost, design, and quality, and to argue that the ends justify the means. It is far more difficult to quantify the social, psychological, and business factors involved in customer relationships, so many studies of software processes tend to marginalize or ignore them.

Modern quality management theory argues that focusing on product end results is necessary, but is only a part of a the bigger picture of meeting customer needs. Lengnick-Hall [12] presents a picture of the evolution of quality management in which the more mature customer-oriented organization must develop trust and effective relationships between development and customer and must measure customer satisfaction and expectation. Requirements, cost, design, and quality are all necessary elements of a successful software development project, but they are hardly sufficient.

XP has been called developer-friendly because of the emphasis on writing code over engaging in process activities. It should be clear that agile methods are also designed to be customer-friendly as well – by involving the customer regularly, responding to evolving requirements, and providing early and regular feedback. At first glance, this would appear to be a win-win situation, and yet it is also clear that because of the social, psychological, and business factors, the practices in agile methods (or any software development method, for that matter) may risk actually degrading the customer relationship.

The remainder of this paper will look at XP by considering how it helps or damages a customer relationship. The purpose is not to critique or advocate XP, but instead to motivate the need for more discussion of the social, psychological, and business factors in studies of XP, agile software development methods, and software engineering in general. Any anecdotal evidence and inferential reasoning presented should be interpreted only as a means to demonstrate how very little we know about these factors and to suggest how dangerous that ignorance might be.

3. THE SUCCESSFUL FAILURE, OR THE EXPECTATIONS GAP

A prototypical XP project is the Daimler-Chrysler C3 payroll system, which has been self-documented by the members of the C3 development team and declared at various times to be a success. According to a brief report [6], which doubles as executive summary and marketing brochure, the C3 team declares that they "deliver[ed] a high-quality application on time and within budget."

XP critics Stephens and Rosenberg [22] have pieced together a different picture of the C3 project from posts to the XP Wiki, bulletin boards, and newsgroups. Their story suggests that at best, the C3 project delivered only around one-third of the requirements before being terminated, and suggest that XP, as a process, developed a bad reputation with the customer[4].

None of this can be confirmed one way or the other, but it does suggest a potential gap between the perceptions and expectations of the customer and the perceptions and expectations of the development team. Certainly, the C3 team, as XP evangelists, should be expected to present the project as an overall success in the face of the project's termination, but in a software development practice with shifting requirements and very short iterations, it could be very easy for a team to lose track of the larger project and its goals.

Although XP attempts to increase customer-development team communication, both in quality and quantity, it is still possible for gaps in expectation and satisfaction to appear. Brown and Swartz identified three different types of satisfaction gaps [5]:

[3] Equally important corollary questions are: *Would your customer recommend you to other customers?* and *Why did we lose a customer after successful completion of a project?*

[4] From the XP Wiki: "The impression amongst the folk I spoke to was that in the view of [Daimler-Chrysler's] management C3 was a disastrous project, and never the like shall be seen again there."

- The gap between client expectations and client experiences
- The gap between client expectations and provider perception of client expectations
- The gap between client experiences and provider perception of client experiences

These gaps can be significant in any software development process, but more formalized development processes attempt to define expectations explicitly during project negotiation and define the criteria for evaluating the delivered service. Certainly, by the end of a project the customer may not be satisfied with the acceptance criteria determined at the beginning. There are anecdotal stories of projects that deliver on time and within budget yet fail to make the customer happy [19]. The ability of agile development methods to adapt to changes in requirements is a strength, but there is still a risk that some significant expectation is not adequately expressed or sufficiently understood by the development team. With high degree of communication, rapid feedback, and constant adjustments, XP should prevent expectations gaps from becoming unmanageable, but it depends on the quality of the communication between the customer and the development team. More research on how requirements and expectations change, how the customer communicates with the team, and how well the team understands the customer's communications over the duration of an XP project would provide insight into on how better to serve the customer's needs.

4. THE BURDEN OF CUSTOMER INVOLVEMENT

XP involves the customer in the development cycle more than many other structured processes. Other structured processes typically involve the customer primarily during the early and late phases of the development process, specifically requirements elicitation and analysis, budget and contract negotiation, and acceptance testing.

In XP, the *on-site customer*[5] and *customer team* requirements make explicit the regular and direct involvement of the customer in the development activities. Three of XP's twelve core practices directly involve the customer:

- *Test-Driven Development* through customer written or directed *customer tests* (essentially acceptance tests)
- The *Planning Game* in which the customer provides user stories and based on cost estimates selects the priorities for the next iteration
- The *Whole Team*, which places a customer in the room with software management and software development

Despite the importance of the on-site customer requirement, few XP project studies report that they are actually able to fully implement this practice. In many published reports, this requirement is partially implement by having a knowledgeable engineer or manager role-play as the customer to supplement a part-time or unavailable customer. One report [18] simply states, "XP's most problematic feature is the amount of on-site customer involvement it requires."

Research suggests that the on-site customer be competent, knowledgeable, and most importantly, credible, or else the benefits to the development process will be limited [16]. In addition, the customer must be able to handle taking on multiple tasks and roles and have managerial support [15]. This means a suitable on-site customer is almost certain to be very valuable to the customer organization. The difficulty in obtaining an on-site customer suggests that the customer organization perceives little business value in donating a valuable employee to a software development effort. Requiring the customer organization to provide an employee can put strain on the relationship with the customer.

Current XP research suggests that the on-site customer practice, even when partially adopted, provides observable benefits in the final product. However, studies are still relatively immature in measuring those benefits. As a result, the business case for the expensive, full-time, on-site customer is weak. A compromise might be to have a part-time customer or to have a technical manager role-play as the customer, but evidence is weak on how these strategies affect the development process.

One recent study [11] observed the activities of the on-site customer on a small XP development project and observed that having the customer on-site was beneficial to the development team. Developers were able to resolve issues with the customer quickly due to the high access and availability. However, the study showed that not only was the customer very underutilized throughout the project's duration, but the customer's attempts to perform other job-related tasks were disturbed by the noise of the development team and by irregular interruptions. The investigators offered some suggestions on how to maximize availability while minimizing the negative impact on the customer's ability to perform other, non-XP tasks.

5. LIVING IN THE SPOTLIGHT

One interesting study in the general area of service industries attempted to examine the relationship between the closeness of the customer-service provider relationship and customer satisfaction [9]. Although their results may not generalize to the software engineering domain[6], they suggest that a high involvement with the service increased overall dissatisfaction. "Customer involvement may be functional if the firms perform well, but if they perform poorly, closer relationships may amplify overall dissatisfaction." In general, the customer involvement of XP means that the potential level of customer satisfaction may be high, but if the team is not mature enough, or if there are extenuating risks, the customer may perceive dissatisfaction more strongly than if the customer had been less involved.

[5] According to the XP Wiki, the *on-site customer* terminology has been retired in favor of the more flexible *whole team* concept which expands the notion of customer interaction to include other domain experts, business and marketing specialists, and end users. The lead customer concept, as primary liaison and final arbiter of decisions, remains. (See "OnsiteCustomer" and "CustomerTeam")

[6] The Goodman study examines satisfaction with the US Postal Service, an inexpensive, short-term service activity with little competition. However, they also suggest that the results are consistent with trust and psychological contract theories, both of which are applicable to software engineering.

Although the psychological problem of customer proximity in software engineering has not been studied in detail, some other findings may suggest a corollary effect on program management. Murru, *et.al.*, conducted informal interviews with several non-technical people who had familiarity with XP, but had not participated in an XP project [18]. One concern that project management expressed regarded the transparency of the development process. Specifically, managers were concerned that deficiencies in programmer skill, which would normally be managed internally through personnel reallocation or contract staffing, would be visible to customers and encourage them to assert control over personnel decisions.

The problem is generalizable to any number of local situations that can be effectively managed internally and never involve the customer. Placing a customer in the middle of the process on a full-time basis could make it difficult for project managers to exercise control without customer interference. Mills, *et. al* [16], observe that without strong supervisory control, customer-producer teams can lead to uncertainty in employee roles.

Risky situations, such as schedule slippages and technical difficulties, are more difficult to hide from the customer in XP, but the net benefit may be positive, since it deters institutional delusion about project status and can open communication about how to manage the situations to best meet the customer's business needs. In practice, the net effect of the high transparency might not be significant, but there is a risk of the customer perceiving the daily chaos of the development process and attempting to assert control over it.

6. DECIDING WHAT YOU WANT

During the planning game, the customer is presented with a number of options on what stories to implement during the next iteration. Giving the customer control over setting local priorities can increase the responsiveness of the development team to the customer's requirements, and consequently increase the customer's overall satisfaction with the project. However, there are a number of factors which influence how effective the customer is at expressing those requirements to the development team and how the customer perceives that requirements are being satisfied.

Assuming that an on-site customer can be provided, and assuming that customer is knowledgeable of the business requirements and effective in expressing them, and assuming that the personality traits of the customer are compatible with the team, there is still the problem of getting the customer to actually express requirements and dissatisfactions to the team. One notable pitfall is the novice customer who seems quiet and satisfied early in the process, when the customer has sufficient opportunity to influence the development, but begins to complain about anything and everything late in the process [8]. Handling this kind of problem involves more effective requirements elicitation with satisfaction monitoring on the part of the development team, and adequate coaching for the customer on how to assert the business needs.

Another problem involves presenting the customer with too many, too few, or no desirable choices. The planning game presents the customer with choices that may not represent the customer's most desirable outcomes for the next iteration. An incompetent or impatient customer may become frustrated by the amount of deferring that happens and may become irritated by the dependence of the development team on establishing priorities. Even with a customer who has bought into XP, it is unclear if the customer perceives business value in constantly managing the development priorities.

7. THE RISK OF CHANGE

Requirements changes can negatively affect a software development project, both in terms of cost and schedule [24] and quality [10]. Agile development processes, such as XP, attempt to create an environment that maximizes the ability of the team to respond to changes and ameliorate some of the costs of those changes [4]. Whether or not XP succeeds in that goal is debatable, but the question should be asked if, in general, embracing late changes provides real business value to the customer.

There are several strategies for dealing with emerging or changing requirements. During the planning game, a customer can choose to defer a requirement which is perceived to be volatile, deferring the decision making on the requirement until a later time when the requirement is more stable or better understood. Other programmer techniques attempt to manage the complexity of the design, facilitate code changes, and improve team understanding of code and interactions. The relationship between the volatility of a requirement and the customer's decision-making process during the planning game has not been studied. It would be interesting to determine if customers are choosing risk avoidance or mitigation tactics when an early release would facilitate the better understanding of the requirements. In this scenario, the gap between the customer's perception of the service being provided and the development team's perception of the service may be quite large.

XP provides techniques for monitoring how changes are affecting the code at large, but XP does not offer specific methods for estimating the up-front cost of making a change. Without sufficient risk analysis, making changes to the requirements is another instance of the customer being forced to make decisions in the absence of valuable information. Moreover, there is business value in accountability, auditing, and assessment of risky decisions. Combined with the perception that requirement changes necessarily negatively affect cost and quality, customers may not be maximizing the business value of a process that embraces changes.

8. INCREMENTAL UPDATES

Analysis on studies of how software features evolve suggest that "development teams should focus, above all, on getting an early (and by definition, incomplete) version of the product into customers' hands at the first opportunity." [13] Subsequent iterations can evolve the software based on specific feedback from the customer. Combined with evidence that effective quality management requires direct customer involvement in design and assessment [12], it appears that the *Agile Manifesto*'s stated principle of striving to provide early and continuous delivery of valuable software is a correct one.

However, the principle does not necessarily mean continuous delivery of software. Incremental updates can be disruptive to

business operations[7]. Depending on how the software is being deployed and used in the organization, various issues involved in deploying new features, installation, education, migration, etc., can negatively impact the customer's organization. Parts of the software product might be deployable, developing business value on top of what is basically incomplete or "beta" software, but later changes may cause that feature to be lost as features are added and refactored. XP provides a real opportunity to deploy value to customers more rapidly, and yet more study is required to understand how end-users can adopt changing software.

9. VALUE-ADDED SERVICE

To paraphrase the *Agile Manifesto*, while there is value in process activities and artifacts, agile software development should value software, customers, and collaborations more. This may be a sustainable position within the local environment of the XP development team, but customers perceive, rightly or not, value in certain process activities and artifacts, such as code documentation, architecture, version management, traceability and auditing. To a developer, the value of these activities may only be perceptual, but to the customer, they can be significant.

Currently, the case for the benefit of eliminating or reducing the process activities and artifacts is anecdotal. Negotiating or demanding that customers adjust their business needs to suit XP, or any development process, can weaken the relationship and potentially weaken a process that requires committed customer involvement. An XP team should be able to adapt the process to customer needs. Interviews and surveys can be performed to understand how customers perceive various process activities, just as more analysis is required to understand how the various practices of XP interact and affect the overall technical characteristics of the software project.

The problem is deeper than a simple training or marketing problem of changing the customer's perceptions. Valuing customers over process sometimes means listening to customers and understanding how the process best serves their needs.

10. CONCLUSIONS AND FURTHER STUDY

Customer satisfaction and customer relationships tend to be a sorely unexplored and largely misunderstood aspect of software engineering. In fact, many technical professionals view themselves as working at odds with the financial and marketing experts in their own organizations. Management research suggests that attempting to increase customer satisfaction through gimmicks like marketing and incentives, rather than based on true satisfaction with the product, can actually harm the organization in the long term [2]. Research on how to build better, and better quality, software systems must continue. Customer relationships are built primarily upon delivering quality products that satisfy the customer's requirements. However, the techniques that software engineers use to build software do affect the customer directly, influence the customer's perception of the product, impact the customer's business model, and provide business value. When evaluating a new software development technique, it is critical to study how the way we build software affects the customer.

In addition to the stress of bringing the customer into the process and placing ongoing responsibilities on the customer, there were two key findings from management research that suggests that more studies on customer relationships and satisfaction are needed. The first shows that managing the customer satisfaction gap can lead to long-term benefits in terms of customer relationships. In the fluidity of an agile development method, expectations and perceptions can shift, and more research is needed on how to measure the customer's perception of the process. More study in this area may show how to enhance XP by including customer expectations and satisfaction monitoring.

The second finding of specific importance to XP is the result that the greater the customer involvement, the greater the potential for customer dissatisfaction. Agile methods, and XP, are no substitute for maturity and discipline in the development team. Combined with increased visibility of development activities, it is all too easy for the customer to perceive failure in undisciplined process. Moreover, it is incorrect to expect that close customer involvement can make up for the failings of a development team. Instead of the customer perceiving that the involvement is helping the team meet the customer's needs, the customer may feel obliged to exert authority over internal development decisions and may also feel mistreated by perceiving that the team is not delivering a valuable product.

Since the relationship between process and customer is not well understood, more research in this area is required. The methods of conducting customer-based research in software engineering are somewhat different from other service industries due to the cost, duration and degree of customer specialization involved. However, it should be possible to gain meaningful data on the impact on XP's customers through simple enhancements to new and ongoing studies. This data can be acquired by expanding surveys of XP to include the customer, specifically ongoing measurement of customer satisfaction with the development activities and delivered product. Moreover, the development team, including project management, should be surveyed about the customer as well to identify any gaps in perception about the customer's expectations and experiences. Starting with findings from other service industries as hypotheses, the results of these surveys can begin to show how customer satisfaction is affected by the choice of processes. Ideally, comparative surveys would be added to more disciplined software process studies (ISO 9001-2000 certification already requires customer satisfaction analysis) to determine factors involved in customer satisfaction with software development processes and ascertain if there is a balance in managing customer involvement and customer perception of satisfaction.

[7] Much of this analysis is based on conversations that the authors have had with engineers in manufacturing research who complained about those "computer guys," meaning the developers of their custom control software.

11. REFERENCES

[1] Agile Alliance. Manifesto for Agile Software Development. http://www.agilemanifesto.org, 2001.

[2] Anderson, E. W. and Sullivan, M. W. The Antecedents and Consequences of Customer Satisfaction for Firms. *Marketing Science*, 12, 2 (Spring, 1993), 125-143.

[3] Beck, K. Embracing Change with Extreme Programming. *Computer*, 32, 10 (Oct. 1999), 70-77.

[4] Beck, K. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, Reading, MA, 2000.

[5] Brown, S. W. and Swartz, T. A. A Gap Analysis of Professional Service Quality. *Journal of Marketing*, 53, 2 (April, 1989), 92-98.

[6] The "C3 Team". Chrysler Goes to "Extremes". *Distributed Computing*, October 1998, 24-26.

[7] Cao, L. and Xu, P. Activity Patterns of Pair Programming. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences, 2005 (HICSS '05)* (Jan. 3-6, 2005). IEEE, 2005.

[8] Elssamadisy, A. and Schalliol, G. Recognizing and Responding to "Bad Smells" in Extreme Programming. In Proceedings of the 24th International Conference on Software Engineering (Orlando, FL, May 19-25, 2002). IEEE, 2002.

[9] Goodman, P. S., Fichman, M., Lerch, F. J., and Snyder, P. R. Customer-Firm Relationships, Involvement, and Customer Satisfaction. *The Academy of Management Journal*, 38, 5 (Oct. 1995), 1310-1324.

[10] Javed, T., Maqsood, M., and Durrani, Q. S. A Study to Investigate the Impact of Requirements Instability on Software Defects. *ACM Software Engineering Notes*, 29, 4 (May, 2004), 1-7.

[11] Koskela, J. and Abrahamsson, P. On-Site Customer in an XP Project: Empirical Results from a Case Study. In *Proceedings 11th European Conference on Software Process Improvements (EuroSPI 2004)* (Trondheim, Norway, November 10-12, 2004), 1-11.

[12] Lengnick-Hall, C. A. Customer Contributions to Quality: A Different View of the Customer-Oriented Firm. *The Academy of Management Review*, 21, 3 (July 1996), 791-824.

[13] MacCormack, A., Verganti, R., and Iansiti, M. Developing Products on "Internet Time": The Anatomy of a Flexible Development Process. *Management Science*, 47, 1 (Jan. 2001), 133-150.

[14] Macias, F., Holcombe, M., and Gheorghe, M. A Formal Experiment Comparing Extreme Programming with Traditional Software Construction. In *Proceedings of the Fourth Mexican International Conference on Computer Science (ENC 2003)* (Apizaco, Mexico, Sept. 8-12, 2003). IEEE, 2003, 73-80.

[15] Martin, A., Noble, J., and Biddle, R. Being Jane Malkovich: A Look Into the World of an XP Customer. In *Proceedings of the 4th International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP2003)* (Genova, Italy, May 25-29, 2003), 234-243.

[16] Mills, P. K., Chase, R. B., and Marguilies, N. Motivating the Client/Employee System as a Service Production Strategy. *The Academy of Management Review*, 8, 2 (April, 1983), 301-310.

[17] Muller, M. M. and Padberg, F. An Empirical Study about the Feelgood Factor in Pair Programming. In *Proceedings. 10th International Symposium on Software Metrics 2004* (Chicago, IL, Sept. 11-17, 2004), IEEE, 2004, 151-158.

[18] Murru, O., Deias, R., and Mugheddu, G. Assessing XP at a European Internet Company. *IEEE Software*, 20, 3 (May/June, 2003), 37-43.

[19] Poppendieck, M. and Poppendieck, T. *Lean Software Development: An Agile Toolkit.* Addison-Wesley, Boston, MA, 2003.

[20] Poole, C. and Huisman, J. W. Using Extreme Programming in a Maintenance Environment. *IEEE Software*, 18, 6, (Nov./Dec. 2001), 42-50.

[21] Rumpe, B., and Schröder, A. Quantitative Survey on Extreme Programming Projects. In *Proceedings of the Third International Conference on Extreme Programming and Flexible Processes in Software Engineering (XP2002)* (Alghero, Italy, May 26-30 2002), 95-100.

[22] Stephens, M. and Rosenberg, D. *Extreme Programming Refactored: The Case Against XP*. A! Press (Springer-Verlag), New York, NY, 2003.

[23] Wood, W. and Kleb, W. Exploring XP for Scientific Research. *IEEE Software*, 20, 3 (May/June 2003), 30-36.

[24] Zowghi, D. and Nurmuliani, N. A Study of the Impact of Requirements Volatility on Project Performance. In *Proceedings of the 9th Software Engineering Conference (APSEC 2002)* (Gold Coast, Australia, Dec. 4-6, 2002), 3-11.