Chapter 00

eXtreme Adoption eXperiences of a B2B Start Up

Paul Hodgetts and Denise Phillips

This chapter presents the results of adopting XP at an Internet B2B start-up. We'll discuss our motivations and goals for adopting XP, and the context under which the adoption efforts were evaluated. Then we'll present the results in terms of objective metrics and subjective evaluations. The results indicate that the project conducted under XP demonstrated significant improvements.

Introduction

Escrow.com faced extreme challenges as a new Internet start-up. The business-to-business e-commerce market was, and remains, immature and rapidly changing. Time-to-market was critical to establish the business and gain market share. As a provider of regulated on-line transaction settlement services, quality was essential.

This chapter discusses our experiences adopting Extreme Programming (XP). Escrow.com was in a unique position to compare nearly identical projects conducted under both XP and non-XP processes. By collecting a small set of simple metrics, we are able to measure the relative success of the adoption effort. As we'll see, even under difficult circumstances, this adoption effort produced tangible improvements over the prior project.

Extreme Programming has gained considerable fame and acclaim over the past several years. While the published body of theoretical and how-to knowledge has grown quickly, more experience reports are needed to provide convincing evidence of XP's effectiveness in real-world project contexts. This report strives to contribute to this body of evidence.

Research Hypothesis

By October of 2000, Escrow.com was experiencing severe and growing problems in its development efforts, including a dramatically slowed pace of delivery, increasing development costs, poor product quality, and a deteriorating state of the code base. Early process improvement efforts were largely unsuccessful at solving these core problems.

Senior management recognized that a fundamental change to the development process was needed. Spearheaded by a team of senior developers, research was conducted using XP. Afterwards, we hypothesized that adopting XP would deliver the following benefits:

* **Increased Rate of Development**, allowing the development efforts to keep pace with the increasing demand for new features.
* **Reduced Development Costs**, allowing a reduction in the size of the development team, or deploying existing resources to new projects.
* **Increased Correlation of Software to Business Needs**, bringing the delivered releases in line with the requirements of the end users, and eliminating unnecessary or low return-on-investment features.
* **Reduced Release Cycle**, allowing frequent releases of high-priority features.
* **Increased Product Quality**, reducing the quantity and severity of defects that delayed production releases.
* **Increased Quality of Implementation**, reducing the code entropy that hindered the addition of new features, and increasing the maintainability of the code base.

Existing project management and change control practices produced metrics that measured developer-effort and defect discovery rates. Combined with code analysis, this data enabled objective measurements of the improvement from adopting XP.

Description of the Context of the Experience

Escrow.com is a provider of on-line settlement services. Escrow.com's services are delivered via internet-based enterprise systems, utilizing typical enterprise technologies including ASP, J2EE, and relational databases.

To attack the emerging business-to-business e-commerce market, Escrow.com began development of a flexible and full-featured transaction-processing engine. This project, creatively dubbed "Version 2," or "V2," was conducted using many elements of "traditional" defined processes, and required significant expenditures for personnel, tools, and technologies. The V2 project is used as the baseline for our process improvement comparisons.

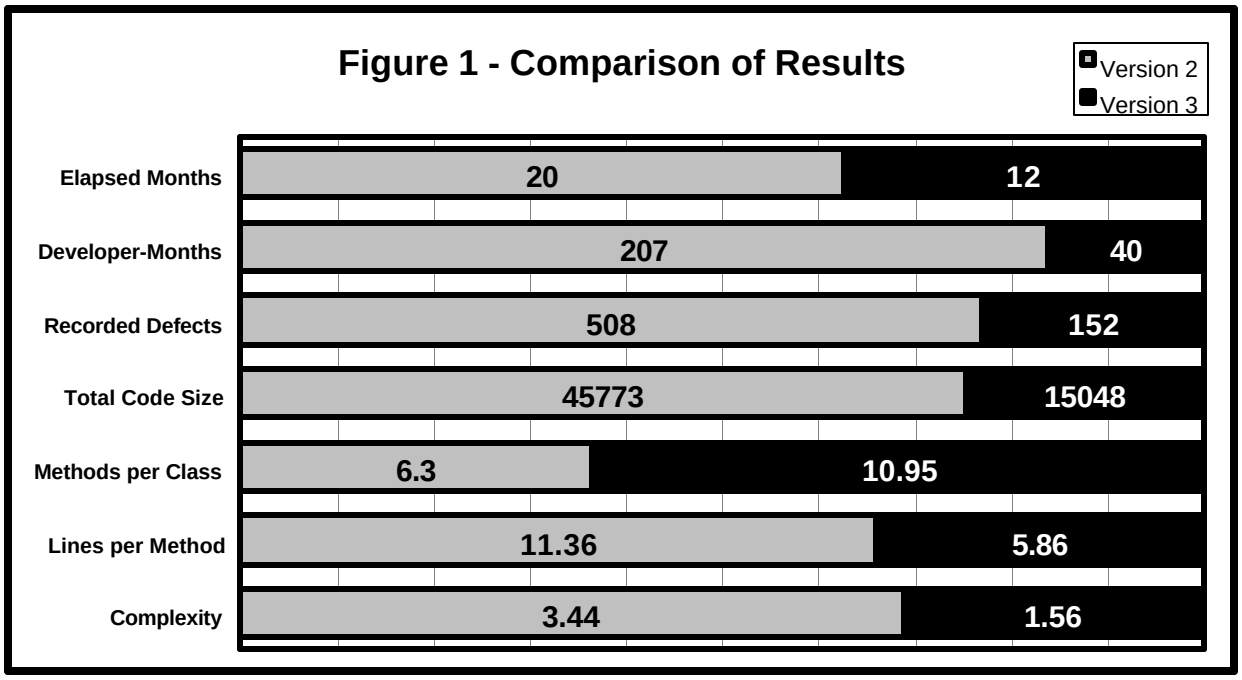
To address problems with the V2 project, Escrow.com radically retooled its development process and culture. The resulting project, called "Version 3," or "V3," used XP as its development process, and is the focus of this experience report.

The XP adoption efforts at Escrow.com were conducted under actual production circumstances. Because of the immediate need to address problems and the company's small size, we had neither the time nor the resources to conduct a pilot XP project. The switch to XP was immediate and complete. Following a brief, two-week preparation period, all development was conducted using the full set of XP practices.

Based on our observations, the context under which XP was adopted at Escrow.com is similar to the development environments at many small-to-medium sized companies. We believe our results to be generalizable to similar development environments and projects.

Results from the Experience

In this section we'll evaluate the results of the XP adoption efforts using metrics gathered from both the V2 and V3 projects. The results are summarized in figure 1.



We must point out that these comparisons were not made in the context of a controlled, scientific study, and are therefore anecdotal in nature. Some comparisons are based on subjective criteria. We nevertheless believe even anecdotal evidence in this context to be relevant and valuable, particularly due to the rarity of scientific studies conducted in actual production environments.

Comparing V2 and V3 Functionality

For the purposes of this evaluation, we must establish a comparison of the overall functionality of the V2 and V3 products. In the absence of objective function point measurements, we must subjectively compare the two products.

In our subjective judgement, the V2 and V3 products were virtually equivalent in the total amount of functionality. While the V2 product incorporated more complexity in its features, the V3 product targeted a wider set of simpler features.

Increased Rate of Development

The V2 project delivered its business value over a period of 20 months before the project was stopped due to the excessive costs of ownership.

The V3 project was suspended after 9 months of development. At the established velocity, V3 would have delivered its total business value over a period of 12 months.

This result represents a 67% increase in the overall development velocity, as measured in terms of the rate of business value delivered over time.

Reduced Development Costs

The V2 project employed a total of 21 developers over its existence. The team size ranged from 3 developers to a maximum of 18 developers at its peak. The V2 project cost a total of 207 developer-months of effort.

The V3 project began with 2 developers. After 4 months, the team size was increased to 4 developers. Overall, using the estimated schedule, the V3 project would have cost a total of 40 developer-months.

These results represent an 80% reduction in developer-month effort, and its corresponding personnel and overhead costs. We note that senior developers staffed the V3 team, and their expertise likely contributed to the productivity gains.

Increased Correlation of Software to Business Needs

The V2 project delivered features and technological capabilities beyond the requirements of the customer, at the expense of delivering revenue-generating features.

The V3 project use of the planning game focused delivery only on clearly identified business requirements. As the expertise of the customer team increased, the development effort increasingly correlated directly to specific revenue-generating opportunities.

Reduced Release Cycle

The V2 project was unable to produce meaningful production releases in cycles of less than two to three months. The quality assurance cycle alone normally lasted two or more weeks.

The V3 project delivered production-ready releases in iteration cycles of two weeks. Due to the increased clarity and prioritization of the planning game, meaningful feature releases were produced in cycles of one to three iterations, representing a substantial improvement in the time between production releases.

The reduction in the release cycle enabled product managers to flexibly and quickly respond to changing business conditions. This was dramatically demonstrated by the rapid succession of changes to the product priorities following the introduction of XP. In January 2001, at the start of the XP adoption, there were two product lines under development. In February 2001, two additional product lines were initiated, including the V3 product. In June 2001, development was stopped on V2, and another product line was suspended. Finally, in October 2001, V3 development was suspended, leaving one remaining active product line.

Increased Product Quality

For both projects, defects found in acceptance testing were tracked using a defect-tracking database. V2's policy was to verbally report minor defects without tracking, while V3 mandated that all defects were formally logged. Acceptance tests on both projects were manually executed, but the V3 project also utilized a suite of 1230 automated unit tests. Acceptance testing on the V2 project was performed sporadically, while on the V3 project acceptance testing was performed repeatedly on all iterations.

The V2 project logged a total of 508 defects over its 20-month lifecycle. 182 of these defects were logged during a difficult two-month period from September through November 2000. Another 123 defects were logged during the final one-month testing cycle prior to V2's last production release in June 2001.

The V3 project logged a total of 114 defects over its entire 9-month duration. All of these defects were minor. Severe defects were discovered and fixed by the developers prior to concluding iterations. Assuming a linear increase in the number of defects had the project run to completion, V3 would have produced 152 total defects.

These results represent a 70% reduction in the number of defects discovered in acceptance testing. This reduction is even more significant when we take into account the lower defect severity levels.

Increased Quality of Implementation

Quality of design and implementation is difficult to objectively measure, as even the types of measurements are subject to debate in the software community. For these measurements, we chose a few relatively simple indicators of possible quality – total code size, the average size of classes, the average size of methods, and the average cyclometric complexity of methods. The total code size metric includes all types of sources (Java, JSP, ASP, HTML, etc.), while the remaining metrics focus only on the Java sources.

Since the V3 project was suspended prior to completion, it was necessary to estimate the final total code size measurement, assuming a linear growth.

The following table summarizes the comparison of these metrics.

	Version 2	Version 3	% Change
Total Code Size	45,773	15,048	**- 67%**
Average Methods per Class	6.30	10.95	**+ 73%**
Average Lines per Method	11.36	5.86	**- 48%**
Average Cyclometric Complexity	3.44	1.56	**- 54%**

While these metrics are subject to differing analyses, when combined with our subjective reviews of the code base, we feel they represent an improvement in the quality of the implementation. The reduction in code size is indicative of a simpler implementation, assuming delivery of comparable functionality. The presence of a larger number of smaller-sized methods per class, combined with the reduced complexity of methods, suggest an improved division of responsibility and behavior across methods.

What to Do Next

This chapter presents evidence that the Version 3 project produced significant measurable improvements over the prior Version 2 project. Many aspects of the two projects remained relatively consistent – the domain and feature sets, the tools and technologies, and the team. The primary difference between the two projects was the use of XP on Version 3. We must therefore conclude that XP contributed substantially to the improvements, a conclusion reinforced by our subjective day-to-day observations.

The XP adoption experiences at Escrow.com have proven to us that XP is particularly effective in today's fast-paced e-commerce environment, and we now make XP our process of choice. We plan to continue to measure and quantify the benefits gained as we adopt XP on future projects.

Acknowledgements

We acknowledge and sincerely thank all of our colleagues at Escrow.com whose efforts enabled these successes. In particular, we thank our former CEO, Russell Stern, for trusting our vision and enabling us to transform the business. We also must thank Robert Martin, Lowell Lindstrom, Brian Button, and Jeff Langr from Object Mentor whose expert help proved invaluable. A special thanks to Beth Hechanova, Tamlyn Jones, and Paul Moore for their help as reviewers. Last, but certainly not least, we thank our families and friends for their limitless patience and support for our "extreme" endeavors.

About the Authors

Paul Hodgetts is a founder and Principal Consultant at Agile Logic, a provider of training, mentoring, and development services focused on agile development and

enterprise technologies. Paul served as the Director of Product Development and Chief Architect at Escrow.com during the V2 and V3 projects.

Denise Phillips is a founder and Principal Consultant at Agile Logic. Denise served as the Senior Enterprise Architect at Escrow.com during the V2 and V3 projects.

Paul and Denise can be reached at www.AgileLogic.com.