How Microsoft Builds Software: Redux

A case study of the mobile development experiences of an SDET intern

By Philip Peng

For Jonathan M. Smith

University of Pennsylvania

CIS-099 Independent Study Report

Table of Contents

Introduction

Internship

Over the summer of 2011, I took on an internship at Microsoft as an SDET Intern (Software Development Engineer in Test intern). It was my first real internship with a large software company; I worked as a part-time IT specialist previous summers and during the school year but only for educational institutions. As an undergraduate studying computer engineering at University of Pennsylvania, the internship aligned well with my field of study and gave me the opportunity to experience software development in an industry setting.

Prior to the internship, I took the course CIS 350 "Software Design and Engineering" with Professor Jonathan M. Smith, which focused on Colwell and Brook's models of software development. In one particular CIS 350 lecture, we discussed the old Microsoft approach at software development, described in Cusumano and Selby's 1997 article, "How Microsoft Builds Software". The article was published two years after the release of Windows 95 and their flagship product, Microsoft Office 95. Since then, Windows has received numerous revisions (Windows 98, 2000, XP, Vista, 7) and Microsoft has vastly expanded their product line (Bing, Xbox + Kinect, Windows Phone 7, Windows Live services, etc.).

To efficiently accommodate these various different development requirements, the Microsoft software development cycle has changed over the past 15 years. This report is a case study of my summer experiences as an SDET Intern at Microsoft as part of the mobile division working on a new software product. Since each division and product team adapts different processes specific to their needs, this report is in no way representative of all the official procedures or guidelines followed by the company. I have, however, tried to record my summer experiences as accurately as possible to give readers an insight into how Microsoft builds software.

Team and Product

Prior to my internship at Microsoft, my main experiences are in working on open source software projects for embedded devices. While I was a developer for the "iPodLinux" project, the members of our loosely-knit team lived around the globe and mostly contributed independently on various different aspects of the project. "ZeroSlackr" project was a solo effort, with me handling the development, documentation, and release processes. In my game development experiences with writing "Beats, Advanced Rhythm Game" for the Android platform, I also worked alone, filling all three roles of designer, developer and tester. At Microsoft, however, the projects are of a completely different magnitude and requirement. With a user base of millions around the world, Microsoft's software focus was strongly on quality and reliability. To achieve this, closely-communicating teams would need to be formed for each product with responsibilities distributed to specialized positions. For me, this was the first time working in a rigid team structure on a large-scale project.

The team that I was placed in for my summer internship was unofficially known as the "Mobile SkyDrive" team. Officially, we were part of the "Devices & Roaming Experience Team, Windows & Windows Live Division" (WWL-DRX) as the feature team working on SkyDrive integration for mobile devices. The WWL division covers development of the main Windows operating system (Windows 7 and 8) and accompanying Windows Live brand of products and services (Hotmail, Messenger, SkyDrive, etc.). The DRX team covers devices and remote accessibility services for WWL products, mainly SkyDrive.

Effectively, my team was in charge of writing the mobile apps for Windows Live's new SkyDrive cloud storage service. We would be targeting the three major mobile platforms: Windows Phone 7, iOS, and Android. Despite not having used or even heard of SkyDrive prior to this internship, I was most likely placed on the team due to my previous experiences with mobile/embedded devices development. The product itself would also be a "Version 1" product, scheduled for its first release at the end of September (a month from the writing of this report). For me, this was a great opportunity as it allowed me to watch and participate in the creation of a brand new product from (almost) the start to end of its very first development cycle.

Results

For SDETs there are roughly four main responsibilities: 1) writing test cases and test specs, 2) writing testing tools and preparing the test/build infrastructures, and 3) executing the tests and following up with bug reports to the SDEs (Software Development Engineers, i.e. developers). During my summer internship as an SDET Intern, I was able to experience two of those: writing test cases and writing testing tools. For writing test case, I was responsible for testing some of the basic features of the iPhone SkyDrive app. For writing testing tools, I was responsible for writing a test framework for automated testing on Android.

For the first quarter of the summer, the team was finishing up with the planning phase so I was able to contribute a bit in the design process and get a glimpse of the specifications review process at Microsoft. In the second quarter of the summer, I had written the automated test cases for verifying the "Sign in/Sign out" process and the "Settings" page. The last half of the summer was spent writing a full and complete Android test automation framework and the 36-page documentation for it (see the "Intern Project" section of this report). The iPhone and Windows Phone 7 SkyDrive apps are expected to be publically released by the end of this September, with the final builds tested thoroughly against my test cases. The Android SkyDrive app is expected to be ready by the next milestone and will be tested using my test automation framework. By the end of the summer, I had fulfilled both my responsibilities and was given a full-time job offer as an SDET at Microsoft.

Microsoft Structure

SDEs, SDETs and PMs

In Microsoft's software development structure, there are generally two types of positions: Individual Contributor (IC) and Manager. ICs are the "developers" of Microsoft; they create the products and write the code. Managers focus on the bigger vision and teamwork aspects, writing reports and deciding on the general direction of product development. As an SDET Intern, I would be considered an IC as I wrote actual code. In general, there are three IC positions at Microsoft: Software Development Engineer (SDE), Software Development Engineer in Test (SDET), and Project Manager (PM). Their roles are roughly as follows:

Software Development Engineer (SDE):

- MSW Glossary definition: *Individuals who write or debug computer programs and may specialize in one or more methods of creating computer programs, Web pages, or programming languages.*
- Responsibilities include:
    - Prototyping and investigation feature implementations
    - Writing feature implementation specifications
    - Implementing features following PM design specs
    - Focus on scaled stability and performance
    - Write unit tests and fix reported bugs

Software Development Engineer in Test (SDET):

- MSW Glossary definition: *Individuals who test and critique software components and interfaces, write test programs to assure quality, and develop test tools in order to increase effectiveness.*
- Responsibilities include:
    - Preparing and writing automated testing framework/tools
    - Writing test cases and scenario specifications
    - Implementing test cases to test against PM design specs

- Running automated tests and occasionally manual verifications
- Reporting bugs and checking fixes
- Maintaining automated daily build environment

Project Manager (PM):

- MSW Glossary definition: *Individuals who are responsible for pulling together and facilitating internal project team communication, driving trade-off decisions, and owning budget and resource planning for multiple projects or programs.*
- Responsibilities include:
  - Facilitate communications with other involved teams
  - Design UI prototypes and interaction behaviour/workflows
  - Writing design specifications and scenarios
  - Managing feature implementation priorities and timelines
  - Organizing internal "dogfood" testing and writing usage documentation

Each feature team consists of SDEs, SDETs and PMs in a rough ratio of 3:3:2, with each role having equal importance and influence on the final product. While PMs may be involved in multiple projects, SDEs and SDETs usually focus on one product/feature line and work closely to follow the PM design specs.

Hierarchy

(TO-DO – need to confirm with Manager on level of non-disclosure)

Content outline:

- My direct tree (SDET -> Senior Test Lead -> Principal Test Manager -> etc.)
- Team tree (SDET + SDE + PM -> Leads -> Principal Managers -> etc.)
- Product Organization (WWL -> WL -> DRX - > Mobile SkyDrive team)

Team Background

(TO-DO – need to confirm with Manager on level of non-disclosure)

Content outline:

- Team composition (number of SDE/SDET/PMs)
- Mesh merge (DRX mesh + SkyDrive -> DRX (SkyDrive))
- Transferred iPhone Messenger team from Shanghai, new team

Microsoft Development Process

Timeline and Milestones (WWL)

(TO-DO – need to confirm with Manager on level of non-disclosure)

Content outline:

- Windows Live Waves (finished 4, starting 5)
- Milestones (Q, 1, 2, 3)
- Windows Live Wave 5 Timeline (mockup)
- Planning, Coding, Stabilization
- Special dates (Code Complete, Dogfood, Zero Bug Bounce, Release Candidate, Release to Operations)

Timeline and Milestones (DRX)

(TO-DO – need to confirm with Manager on level of non-disclosure)

Content outline:

- Mobile SkyDrive team original and modified timelines
- Mobile SkyDrive team milestone expectations
- Modified Sprints
- Change in plan (DCR window)

Agile Development Methodology

Mobile development is a relatively new phenomenon in the software development industry. For in particular, entering the mobile market requires a lot of change in the software development process. Large scale projects such as Windows or Office usually have development cycles lasting many months or years resulting in a largely stable product that is only periodically updated with patches. In the mobile world, however, fixes and new updates are expected on a monthly or sometimes weekly basis. To adapt to these different expectations, many processes had to be heavily modified or, like in our team's case, scraped and replaced with something new. For the Mobile SkyDrive team, the decision was made to use the agile development methodology with the scrum framework.

MSW Glossary defines "agile" as "A people-oriented, adaptive methodology for application development that focuses on short iterations and customer interactions." At The official "Principles behind the Agile Manifesto" document reads as follows:

1) *Our highest priority is to satisfy the customer through early and continuous delivery of valuable software*
2) *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*
3) *Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.*
4) *Business people and developers must work together daily throughout the project.*
5) *Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.*
6) *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*
7) *Working software is the primary measure of progress.*
8) *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*
9) *Continuous attention to technical excellence and good design enhances agility.*
10) *Simplicity--the art of maximizing the amount of work not done--is essential.*
11) *The best architectures, requirements, and designs emerge from self-organizing teams.*
12) *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.*

Our team's main focus was to create a free app that will allow users to effectively use their mobile devices to access the free SkyDrive service. To accomplish this, we would be very customer-based and frequently release updates based on user feedback of their experiences with the app (after the first release). If the users decided that they preferred a different feature or implementation or some technical roadblock appeared preventing us from executing our original plan, we would need to be able to react fast. The agile development methodology would allow us to do just that. This was reflected in our usage (or planned future usage) of the two-sprints development timeline described earlier and planning for major releases every three months. In addition, many of the original design plans were based off user feedback surveys and preparation were made for a forum-based feedback system (in addition to the marketplace comments).

Scrum Process

In addition to adhering to agile development principles, we kept on-track by integrating parts of the Scrum process into our development cycle. MSW Glossary defines "scrum" as "An agile, lightweight process that can be used to manage and control software and product development using iterative, incremental practices." In rugby, a "scrum" is a method used to decide on restarting a play after the ball goes out of bounds. The software development "Scrum" can be seen similarly; the Scrum process is used to restart the development cycle when an unexpected change in plan or user expectations happens. We implemented a few key aspects of the Scrum process: backlog, burn-down chart, sprints, daily stand-ups, and demos.

In his article titled "Agile Development: Lessons Learned from the First Scrum", Dr. Jeff Sutherland documented the first usage of the Scrum process in software development. Despite happening almost 20 years ago, much of his experiences and strategies still work today and were applied with visible effects by our team. In his article, Sutherland wrote:

> *The first Scrum started with a half day planning session that outlined the feature set we wanted to achieve in a six month period. We then broke it into six pieces which were achievable in 30 day sprints. This was the product backlog. For the first sprint, the product backlog was transformed into development tasks that could be done in less than a day.*

Instead of just a "half day planning session", our team expanded this portion to the entire planning phase (e.g. our special Milestone 1). User expectations and target features were brainstormed, researched, filtered, and listed as requirements in our "product backlog". This feature count would then be placed onto a downward-sloping "burn-down chart" that would be updated weekly to reflect the current project "code complete" status.

These backlog items were also categorized into four priority levels: P0, P1, P2 and P3 with P0 being required "user stories" and P3 being less important "stretch goals". The PMs drafted UI mock-ups, designed feature usage workflows, and wrote specs describing the user behaviours that would achieve desired results. Through prototyping and investigation, the SDEs recorded the estimated "cost" in "days" (equivalent to Brook's "man-day") and relayed back to the PMs on whether or not the feature should be changed or scrapped if the cost was found to be too high. In the meanwhile, the SDETs went through the PM specs and wrote test specifications and test cases covering expected and unexpected user behaviours/interactions and their expected results. At the end of these investigations and spec writings was the final sprint-planning phase in which each backlog item was assigned to either Sprint 1 or Sprint 2 of our Milestone 2 timeline.

Another important aspect of the Scrum process that we followed was the daily stand-up meetings, also known in MicroSpeak as "brown bags" (due to the historical inclusion of a brown bag containing cookies at such meetings). MSW Glossary defines the term as: "Short, informal training or informational meetings that generally occur over the lunch hour, designed to fit into the schedules of individuals who might not be able to attend at other times of the day." Sutherland outlined the following requirements for Scrum meetings:

> *The meetings were kept short, typically under 30 minutes and discussion was restricted to the three SCRUM questions:*
>
> *1. What did you do yesterday?*
>
> *2. What will you do today?*
>
> *3. What obstacles got in your way?*

To further encourage the brevity of the meetings, our team's daily brown bags were scheduled at 11:30 am such that we would walk out half an hour later ready to go for lunch together. Despite their brevity, however, I found the daily meetings to be the highlight and winning point of the Scrum process. Going around the table, each member of the team was able to become informed of the latest updates and changes through just a handful of sentences. As an SDET, it was immensely helpful as it allowed me (or my fellow SDETs) know of when a changeset had been or will be checked in such that I could plan my test case writing or running. For SDEs, the face time allowed them to discuss implementation issues or challenges with their fellow developers as well as request clarifications or suggest changes to the PMs. For the PMs, the meetings were a chance to verify that the project was still on track and plan for arranging meetings with other teams if necessary. It was very common for the brief 30 minute meeting to be far more productive than hours of back-forth email and instant message conversations.

The final and most anticipated part of the Scrum process, however, was the demos. Sutherland described his Scrum demo experience as followed: Every Friday during the first Scrum, we held a demo and brought in development experts from other companies in to look at the product. As a result our developers had to do the demo for their peers in other companies. This was one of the best accelerators I have seen in software development. An outside expert would say, "That's terrible; look at Borland's Product X to see how it should be done" or "How could you possibly [sic] have a dumb bug like that?" As a result of this outside input, all problems or bugs would be fixed the following week. Developers refused to be embarrassed a second time in front of their peers.

Rather than having a regular scheduled demo time, our team gave quick demos to each other either during our daily brown bags, over lunch, or whenever we dropped by each other's offices. Since Mobile SkyDrive was a Version 1 product, demos during the first half of the summer were mostly prototypes of specific feature implementations. I remember one meeting where one of our Windows Phone 7 SDEs complained about the lagginess of the default picture transition animation and showed it to us. The next day, he came in smiling and demoed off his "slideshow" hack (loading adjacent pictures and just applying a translation to the expanded canvas, then reload on animation completion). Being able to immediately see the improvement definitely gave a rewarding feeling and morale boost to the entire team – one less worry for the SDEs, one less design change for PMs to consider, and one less "bug" for SDETs to test.

Sometime during second half of the summer, we had our first "working" iPhone SkyDrive build. It was a huge celebratory moment one of our SDEs took out his iPhone and showed us a picture on his phone that he had just uploaded to SkyDrive a few minutes before the meeting. It was equally interesting when I dropped by one of my fellow SDET's office later that day to try playing around with the app myself. While randomly tapping the iPhone's screen, the app completely froze and the UI stopped responding. I showed him his iPhone, to which he responded, "Uh oh, we definitely can't ship with that." An hour later, a bug report had been filed and the SDEs were busy investigating the root cause.

Changes Since 1995

(TO-DO – check with Manager if some changes are DRX-only or company-wide)

Content outline:

- Overview of Cusumano and Selby's paper
- Improvement from sync-and-stablilize model to include design changes
- Shifted focus to scalability and stability
- 4 Milestones instead of 3
- Planning, Coding, and Stabilize time lengths
- Embracing of agile development and shorter release cycles for smaller projects such as mobile
- Include WWL Wave 5 M1 bug graphs?

Testing at Microsoft

(TO-DO – need to confirm with Manager on level of non-disclosure)

Writing Specs

Content outline:

- Test tools and frameworks
- BVTs (Basic Verification Tests)
- TU Acceptance Scenarios (Test Unit)
- TU Comprehensive Scenarios
- Review process

Writing Test Cases

Content outline:

- Scenario details (preconditions, steps, expected results)
- Prioritizations
- Automated or manual?

Manual vs Automated Testing

Content outline:

- Pros, cons, and role of each
- Automation framework
- Verification checks
- Automate BVT
- Final manual checking

Daily Builds

Content outline:

- Build system (machines and scripts)
- Version control (source depot)
- Using emulators/simulators

Interacting with SDEs and PMs

Content outline:

- Bug reports and fix checking
- Expected response from errors
- Unaccounted scenarios

Intern Project

(TO-DO – need to confirm with Manager on level of non-disclosure)

Process

Content outline:

- Intern commitments
- Manager and coach weekly meetings
- Midpoint review and final review
- Final demo and presentation
- Documentation writing

Purpose

Content outline:

- Current iPhone test framework
- Android app in M3
- Blackbox testing
- I'm the only one on my team with Android experience

Implementation

Content outline:

- Diagrams copied from documentation
- High level descriptions of methodology and process flow
- No source code or references to specific files

Results

Content outline:

- Feature list

- Demo feedback
- Code checked in
- Report

Future Impact

Content outline:

- Teaching fellow SDETs how to use
- To-Do list written
- Integration into common test framework
- Envisioned usage by other mobile teams outside DRX

Internship Reflection

(TO-DO – need to confirm with Manager on level of non-disclosure)

Development Cycle

Content outline:

- Saw almost full development cycle
- Scrum = good
- Microsoft dev cycle still too slow (too much focus on process)
- Mobile SkyDrive = V1 product so process was experimental

SDET vs SDE

Content outline:

- Difference in roles
- Enjoyed both, which to choose?
- Microsoft role flexibility

Microsoft's Vision

Content outline:

- Windows 8
- SkyDrive
- Mobile devices

Works Cited

(TO-DO – ask Manager about using MSW Glossary as a source)

- “MSW Glossary.” Microsoft Web (Microsoft internal resource). <http://msw/AboutMicrosoft/Glossary/Pages/default.aspx>.
- Brooks, Frederick P. *The Mythical Man-Month*. Addison-Wesley: 1975.
- Colwell, Robert P. *The Pentium Chronicles*. Willey-Interscience: 2006.
- Cusumano, Michael A. and Richard W. Selby. “How Microsoft Builds Software.” CACM 40(6): 1997, pp. 53-61.
- Highsmith, Jim et al. “Principles behind the Agile Manifesto.” Agile Alliance: 2011. <http://agilemanifesto.org/principles.html>.
- Sutherland, Jeff. “Agile Development: Lessons Learned from the First Scrum.” Scrum Alliance: 2004. <http://www.scrumalliance.org/resources/35>.