Experiences of Requirements Engineering for Two Consecutive Versions of a Product at VLSC

Joel So and Daniel M. Berry
University of Waterloo, Ontario, Canada

2006

Introduction

This talk is about experiences of the first author in leadership requirements engineering (RE) roles …

in two consecutive software (SW) development projects, …

for two consecutive versions of one SW product, …

carried out in one SW development company.

First Development

In the first development,

- **the RE process was poor,**
- **the shipment was late, and**
- **the product’s quality was nothing to write home about!**

Between the Two Developments

Between the two developments, the management of the first development *and the executive level management of VLSC*

- **reviewed the first development,**
- **realized its RE’s shortcomings,**
- **realized importance of full upfront RE, and**
- **issued a mandate to do better RE in second development.**

Second Development

In the second development,

- **the RE process was much improved,**
- **the Beta 1 shipment was on time, and**
- **the product seems to be significantly better.**

VLSC's SW Development Lifecycle

Officially, each SW development at VLSC follows a waterfall model whose milestones are:

M_0: finishing functional specification,

M_1: finishing design specification,

M_2: finishing code development,

M_3: finishing testing of the entire system,

Lifecycle, Cont'd

M_i**:** **Beta test *i* release to early adopters who have agreed to evaluate the release,**

FR**:** **final release of retail version to market, out of product team's hands**

Each release starts with steps leading to a new *M_0*, and eventually to *FR*.

Beta Test Release

The steps and prerequisites for preparing and doing a Beta test release to early adopters are:

1. **all reported bugs in M_i build have been resolved,**
2. **M_i build passes all test plan tests,**
3. **M_i build is released via the Web or other mechanism, and**
4. **early adopters provide feedback.**

Delusions of Upfront RE

From kick off until well into design of Version 3, …

the PMs were wearing SW architect’s hats when they should have been wearing requirements engineer’s hats.

Architecture Thinking

Recall: Version 3 was to be a major architectural restructuring of Version 2.

Therefore, several designs and prototypes were floating around from even before kick off.

The PMs were so focused on architecture and code, that they were barely thinking at requirements level.

Causes of Architecture Thinking

The focus on architecture came from a lack of:

1. **direction from management to focus on requirements**

2. **a clear product vision, which would specify goals, the why that motivates the what.**

Immersion in Development

The PMs were immersed in development, putting code before requirements, because of perceptions that

- **working out requirements before coding**
    - **wasted time and**
    - **delayed starting real work, and**
- **beginning coding sooner shortend the lifecycle.**

Upfront RE Was Mandated

Project schedule mandated upfront RE,

But, mostly only lip service was paid to this RE.

What Was Really Happening

Requirements gathering and specification were lumped into one step.

Thus, first ideas were taken as final requirements without carefully considering whether the ideas made a consistent whole.

Barely any real attempt to gather requirements from real customers and users.

Really Happenings, Cont’d

Instead, the PMs put on customer’s hats and wrote down what they *thought* a real customer would want

The PMs did not check if any real customer really wanted what the PMs wrote down.

Really Happenings, Cont’d

At scheduled time of the M_0 milestone, the PMs took what they had and signed off on it.

After all, working code existed!

No one really cared about the requirements specification (RS).

A Telling Statistic

A telling statistic: the process leading to M_0 had very few bugs reported.

A low bug count in the normally bug-laden step of distilling disparate people's fuzzy ideas into a consistent, complete, and concrete whole is a *bad* sign.

True Bug Status

Undoubtedly, the RS was loaded with bugs.

Since they were not found during RE, they were lurking to be committed into buggy designs and code.

These bugs would be found only

- **during testing or**

- **after shipping, by customers, the best bug finders in the universe.**

Cart Before the Horse

In fact, steps leading to M_1 and M_2 were well underway before M_0.

Much of RE that did happen in project happened in short bursts of RE during the steps leading to M_1 or M_2, …

whenever designing or coding could not proceed without resolving a requirements issue.

Another Sign of Trouble

In VLSC, a change in a product’s internal code name indicates a massive realignment of the product’s scope and requirements.

PROD’s internal code name changed 3 times late during Version 3’s development, …

and the 3rd was even after Beta 1 release.

Slippage

Beta 1 was released 4 months late.

As at most SW companies, at VLSC, everything ships eventually, and the slips just accumulate.

Final release of Version 3 slipped an additional 2 months.

Thus, the total delay was 6 months in a 25 month project, about 25%.

Questionable Quality

Version 3’s quality was questionable.

Several quick fixes were issued in first month of general availability.

Planning for a patch package began immediately.

Postmortem

Product team asked itself, ‘‘What went wrong?’’ not to repeat bad history.

The PMs realized that they had not done enough upfront RE,

- **even though RE and its milestone, M_0, were scheduled, and**
- **the PMs had signed M_0 as completed.**

Instead

The PMs had a salute-the-flag attitude about the RE that they should have been doing:

- **PROD’s vision was not defined up front;**
- **targeted scenarios were not identified and finalized up front.**

Informality, Imagination, and TBD

The PMs’ requirements gathering was very informal.

Often, they only imagined what a real user *would* want.

Thus, the signed-off RS was neither concise nor complete, leaving much room for error and misinterpretation.

PMs left much to be fleshed out by implementers, with a high risk of their not implementing the PMs’ intents.

Delusions

Much of so-called RE time was spent doing things other than RE, …

things that PMs perceived as more important than RE.

The PMs’ signature on the M_0 milestone reflected self delusion, not fact.

Compounded Problems

PMs did costing and scheduling based on an incomplete and incorrect RS.

When true requirements began to emerge during coding,

- **the old costs and schedule were invalidated**
- **the new requirements cost considerably more to implement than if they had been found during RE.**

Compounded Compounding

New requirements forced major, costly restructuring, …

throwing off costing and scheduling even more!

Version 3 Development Sum Up

Thus, letting the implementation cart get before the RE horse led to

- **incorrect costing and scheduling,**
- **substantial re-engineering, and**
- **finally, a 25% schedule slip.**

Executional Catastrophe

Version 3’s development and release were viewed internally as an executional catastrophe despite its positive growth in sales revenue.

Heads rolled:

- **both division VPs,**
- **1 PROD product unit manager,**
- **2 general managers, and**
- **others**

Cause of Catastrophe

VLSC’s *executive level* management recognized that the failure was caused primarily by insufficient and incorrect upfront RE.

How often do you see high-level executives in a big corporation thinking about good RE?

Nu!?!?