

Iteration Advocate/Iteration Transition Meeting: Small Sampling of New agile Techniques Used at a Major Telecommunications Firm

Brian S. Boelsterli
Principal, WaveFront, Inc.
10875 US HWY 285, Suite 204
Conifer, CO 80433
303.838.3332

brian.boelsterli@wavefrontinc.com

ABSTRACT

This paper documents a successful implementation of agile at a major telecommunications firm. Critical aspects to mention about this particular software endeavor include a) this could be one of the largest implementations of agile documented thus far (approximately 275 immediate and over 3,000 supportive contributors involved), b) the team practicing agile did so under the constraints of a waterfall-gated (both milestone and financial) environment, c) a 'practice tapestry' approach was used to 'weave' together elements of agile that would work for this organization, and finally, d) new practices, not previously documented to the agile community were applied in addition to those well known thus far. This paper focuses on these new practices in an effort to share them with the agile community. The weaving together of existing and new practices, enabled this major telecommunications firm to deliver a major software release to its large customer base.

Keywords

SDLC, development, process, agile, methodology

1. Introduction

What follows is an experience report describing what could be the largest attempted implementation of an agile -based software development effort (at least documented). It involves a large telecom firm. At the heart of this team were approximately 275 immediate, along with over 3000 supportive resources. It involved the full-life cycle implementation of the telecom's major product launch. Aside from the mere size of this effort and its relative success, of interest is the application of agile. Unlike, possibly most agile efforts, this one involved a "weaving together" of multiple agile methodologies, rather than adopting one specific agile methodology. Of interest to the reader, are six practices thought to be new in the agile community.

This paper focuses on these 'new' (or at least, previously undocumented) practices with the specific intention of sharing with the agile community to further enrich the practices known thus far.

2. Situation

As mentioned above, the organization consisted of a smaller company, that was part of a larger, parent company. For the purposes of this report, I will refer to the company under discussion as 'the shop'.

The shop was organized with the following teams::

- Business Unit (the customer),
- Architecture,
- Project and Release Management,
- User Experience,
- Systems Analysts,
- Development (at least 5 development teams),
- Component Test,
- Configuration Management,
- Application Administrators,
- Process Office,
- System Test,
- Administration/Management.

A release manager assigned was responsible for managing and coordinating the release. Project Managers were assigned and took responsibility for the individual projects themselves. A fully cross-functional team representing the software development life cycle staffed each project. Matrixed resources from both the primary and secondary organizations made up the cross-functional project teams.

At a very high level, this major release provided what was being referred to as 'third-generation' (or 3G) telephony technologies. Capabilities such as allowing users to managing their phone accounts on-

line, purchasing handsets, ringtones, storing/retrieving offerings, etc.

The time frame of this implementation spanned the period from November 2001 to July 2002; the organization is still, at the time of this writing, engaged in development for follow-on functionality. In February 2002, the agile practices were initiated, first on one of the project teams already in flight. Within 8 weeks, the majority of the project teams were practicing the same “processes”.

The following technologies were used on the project: Java, J2EE, XML, XSL, XSLT, Ant, Apache(Xerces, Xalan), JUnit, AspectJ, Bea (Application Server), Blue Martini, Borland(Visibroker), Exolab(Castor), IBM (MQ-Series), Jakarta(Log4J, Struts, Cactus), Oracle, Sonic (MQ), TogetherSoft (TogetherControlCenter), Sourceforge (Eclipse), HttpUnit, Dom4J, JavaCVS.

3. Action

The challenge faced by the shop related specifically and directly to the dependencies it had on the parent company. While the above team was building its web applications, back-end functionality was also being developed by the parent organization using waterfall techniques. As in most waterfall situations, dates were slipping. The ripple affects of this slippage resulted in smaller windows for the shop to complete its tasks (analyze, design, develop, deploy, test, etc.) As a risk mitigation technique, the organization put a very small team in place to begin exploring its options to head-off this risk challenge; this is where agile came in.

After careful evaluation (organization, skill-set, process maturity, etc), an effort was put into place on one of the many projects to explore agile techniques. Those participating in this exploration liked ideas such as: continuous integration and delivery of working software, welcoming changing requirements and regular interval process adjustments. By instituting the above agile concepts, the team felt they could take pieces of the required functionality and wire them to back-end systems, as they came on-line, rather than waiting for these systems to come on-line at the same time. Due to the complex nature of the shop and its parent organization, the lead team decided to employ multiple agile concepts together along with some new concepts not documented thus far in the agile community.

3.1 “Existing” agile techniques used

The following is a small list of agile practices we put into place that come from other well-known agile methodologies: Feature Plans, Design/Code reviews, Unit testing (FDD); Daily Scrum’s (SCRUM), Test Driven Development, Refactoring, Continuous Integration, Coding Standards, Small Releases, The Planning Game (eXtreme Programming).

3.2 “New” agile techniques

The following is a list of the six practices we put into place, that are above-and beyond what we have seen documented in the agile community:

a) Software Iteration Plans (SIP)

Requirements documents and artifacts were ‘thunked’ into a Software Iteration Plan. This concept is primarily based on FDD’s Feature Plan. However, the SIP not only included features, but also deficiencies from previous iterations, Change Control items, and tasks. Features are straight from FDD; change control items are artifacts containing approved change requests; tasks represented detailed items that were required to complete to support the features. Examples include, but are not limited to: environment creation, data conversion, architectural build-outs, etc.

Each release had its own SIP which contained the collection of features, change control’s and tasks for all projects into and under the release. The SIP was updated on a weekly basis, via the ITM’s (described below). Therefore, the release team had visibility into the overall health of the effort, by how the release team was tracking on the SIP.

b) Iteration Transition Meetings (ITM)

This concept really formed the backbone of the agile initiative. This meeting was held once weekly, corresponding to the heart beat (described below). The release manager and Iteration Advocate (described below) conducted this meeting. It was the ‘buyer/seller’ marketplace for the cross-functional teams to come and buy/sell their respective deliverables. Regardless of release or project size, the meeting was held within a 1-hour time frame. Representation from the cross-functional team (customer, analysts, developers, testers, configuration managers, application administrators, DBA’s, etc.) was required. This implied that at least one representative from each team was present at each ITM.

The agenda was as follows:

- **Feature Push (described below) Summary**

[Seller] Development team leads (chief Programmers (FDD)) would summarize the features (SIP) they were able to deliver downstream to the [buyer] Configuration Management team. Configuration Managers summarized the overall health of the Feature push as a QA step. If a successful Feature Push, the [seller] Configuration Management team would then alert the [buyer/seller] Application Administration Team to prepare themselves to build/deploy the Features into the [buyer] System Testing environment.

- **SIP updates**

This was an opportunity for any representative of the [seller] cross-functional team to propose updates to the SIP as executed by the [buyer] release manager. As these were presented, representatives from the cross-functional team received a heads-up, first hand. New features (requirements) were also introduced via a change control process (two speeds, fast and slow).

- **Domain Presentation**

[seller]Systems Analysts, User Experience and Architecture team representatives would provide the [buyer] Development and Systems Test teams the iteration Domain Specification (discussed later) as well as a quick summary of the artifact they were to implement in the next iteration. Before the meeting, the User Experience resource would post the wireframes on the wall. Then, the Systems Analyst would review the features referencing the posted wireframes. This provided a quick venue for exchange of information. Once completed, formal exchanges of specifications would take place allowing the buyers to implement the functionality and test specs. Issues were tracked and, if necessary, off-line meetings would follow. To ensure a smooth transition, pre-ITM review of the artifacts would take place.

- **Adjustments (rights/wrongs)**

This was a time for the entire team to assess adjustments in the process. Rather than making adjustments at the end of the project/release, we chose to do so on a weekly basis. If 'wrongs' were enumerated, members were assigned

Action items, which required resolution prior to the next ITM.

- **Next Steps**

Action items from the previous ITM were discussed as well as new action items stemming from this ITM were reviewed. Lastly, the release manager would ask the question 'Are we all on the same page for the next iteration?' By the time the meeting got to this point, all issues/concerns/questions had been addressed and the meeting concluded. The release manager would publish the meeting minutes. Some interesting statistics to consider: From the time frame March to December 2002, only one ITM was canceled. An independent process assessment recently conducted stated the ITM as the most effective tool in conducting projects.

c) Issue Review Meetings (IRM)

Once the team reached a 'code-complete' state, the team transitioned from the weekly ITM's to IRM's. These took place typically 2-3 times per week. As time progressed closer to the actual release milestone, these occurred daily. The agenda was simple: review the issues/defects on the team's work list. This is not proposed to be a new concept to agile or any software process; it is mentioned here to demonstrate the conclusion of the ITM's.

d) Heart Beat

A weekly 'heart beat' was instituted. The contents of the SIP called out for deliverables from teams at different times. Some project teams followed a two-week iteration, some three, and in some cases, varying from one to three weeks, per iteration. Regardless, the SIP outlined deliverables, the heartbeat provided the drum roll, and the ITM's provided the marketplace. It is important to note the following: just because heartbeats were weekly, this did not imply all project teams conducted weekly iterations. Rather, the heartbeat offered project teams a point in time and a place (ITM) from which they were able to push their deliverables up the chain whether this is specifications or code.

e) Feature Push

This concept is a variation of FDD's 'Promote to build'. We re-cast this concept to solidify the concept of delivering 'features' as opposed to

builds or just code. Features imply a sense of completion. Feature Push's implied: working test harnesses proven by our Continuous Integration Engine (another discussion) and verified by the Configuration Management team,

f) Iteration Specifications

Following the philosophy of “cut and grow”; our implementation of agile Development called out for the following specifications:

- **Iteration Domain Specification**

This document provided the detailed requirements of the features including the wireframes to be built in the particular iteration. This document was delivered in each ITM.

- **Iteration Test Specification**

This document provided the detailed test steps to test the features to be built in the particular iteration. This document was delivered in each ITM.

- **Iteration Features**

Although not a 'document' per say, this represented the actual software containing the features to be delivered.

g) Iteration Advocate

This was a new role within the organization. The main purpose of this individual was to ensure iterations were executed, assist the release manager and any sub-sequent team address and stomp any risk-issues that arise. Another critical element of this role was to ensure the Process Office was articulating its process documentation (web site) to be consistent with best practices conducted in the field. Important to note, is the role of the IA to both ensure iteration deliverables as well as assist the teams in generating these deliverables (which includes Feature Push's).

4. Result

As a result of the practices mentioned above, the team delivered the final product to its customer base. Press-releases, major celebrations followed the delivery.

Immediately after the delivery, the team conducted a post-delivery review to discuss the goods and the

bad of this long and exhaustive experience. Highlights included (but not limited to):

- Environments or lack-thereof hindered involvement from the System Testing organization and, in fact, from the Development organization. Early on, we realized we needed to define and implement early environments at the inception of the release;
- Requirements, and a consistent approach to defining these. With many Systems Analysts involved in many projects, Chief Programmers were challenged with multiple requirements specification formats and varietal content. Additionally, one major challenge included the task of tracking requirements and changes to them (no surprise!);
- Dependencies on Enterprise functionality. Many of our frustrations and challenges involved working around the availability, from both an API level as well as a general up-time perspective, of the Enterprise functionality. This is your standard enterprise integration story. Risk mitigation techniques included the implementation of mock-objects followed by a version of our architecture which provided cached back-end data from our business services layer;
- Organization layout. Many team members expressed an interest in re-organizing the teams by moving away from a matrixed organization fashion to more of a product-based organization.

Today, the organization is going through much change, our core-underlying practices, however, are still being executed by the release and project teams. At the time of this writing, we have probably hit our 100 ITM mark!

1. Conclusion

Among many of the aspects resulting from the practices mentioned above, one seems to stand-out: . the implementation of agile described here solidified the software development life cycle work breakdown structure representing an incremental, iterative, architecturally-driven approach to software development. The repetitive nature of the ITM's and IRM's ensured the cross-functional team worked in a very well planned choreography. Short, delivery-focused milestones allowed all parties to not only participate, but also track, at a very low level, what we called 'bonified progress" or lack-thereof. Change requests were allowed to permeate the efforts of the team, as long as they were approved and the estimates adjusted accordingly. Additionally, the fact that we were able to execute in an agile -like fashion within a large, corporate-like,

waterfall world demonstrates, agile can exist, even within such constraints. This is not to say we practiced true-agile, rather a form, with value-add's, within a non-agile environment.

Given the sheer size of the participants involved in this major release, along with other factors such as technologies used, duration, risk profile, team makeup, etc., the realm of agile techniques and practices used involved picking and choosing elements of many agile methods, rather than focusing on any one. Additionally, new concepts to the agile world were also introduced. Concepts such as Iteration Advocate and Iteration Transition Meeting really are the more interesting practices we feel brought the biggest value to the team and the organization. It is within these new concepts that drive the white paper contributors to want to share our learning's to the agile community.