Experimenting With Agile Practices – First Things First

Fergal Downey, Gerry Coleman and Fergal McCaffrey

Dundalk Institute of Technology, Dept of Computing & Maths, Dublin Road, Dundalk, Co. Louth, Ireland
fdown05@studentmail.dkit.ie, {gerry.coleman, fergal.mccaffrey}@dkit.ie

Abstract. Faced with challenges in relation to interpretation of requirements, issues with build and deployment and excessive integration defects, this paper examines how a software team propose using a novel combination of Covey's 'First Things First' principle and Cockburn's Methodology Shaping, as a potential solution to examine their current process and define a new set of working conventions which will address these issues.

Keywords. Methodology, software, agile, time-management.

1 INTRODUCTION

The software team involved in this research is part of a major UK bank and is responsible for the development of eCommerce applications supporting mainly customer servicing requirements identified by each of the bank's business divisions. The team was established in 2001 but has now grown to more than sixty people which include one of the authors. The team has a strong focus on project delivery but no-one is assigned responsibility for process or methodology.

The company provides a set of Project Management Minimum Standards (PMMS) which are used for project control and these standards are based on a traditional waterfall approach to software development. The standards are generic so that they can be used independently of technology or domain, but as a result are not specific enough to be of real value.

A number of key issues are encountered to varying degrees on each of the projects undertaken by the team as follows:

Requirements Not Fully Understood

The PMMS mandates the delivery of a Business Requirements Definition (BRD). This must be signed off by the business expert and the project sponsor and as such must be at a low level of precision, sufficient to define the proposed business value and the application domain. A high precision Detailed BRD (DBRD) is subsequently produced to enable technical specifications and designs to be delivered. This is written as a set of detailed textual Use Cases and is accompanied by a "happy-path" prototype, but the sheer volume of information (the most recent project had a DBRD of almost 400 pages) and the Unified Modeling Language (UML) [1] format leads to the business expert signing off a specification that they do not fully understand in order to progress the project to the PMMS Delivery stage.

No Integration Until After Build Completion

In order to break up the project into manageable chunks for which developers can take ownership, the project team is divided into User Interface (UI), Mid-Tier and Database sub teams. Each use case is then assigned to someone from each sub team who are then responsible for delivery of the end-to-end use case. This works to an extent, but issues are encountered when there are multiple dependencies between use cases and when these are being developed by different members of the sub teams. This leads to excessive integration defects being encountered when all use cases are delivered as an entire application at the end of the Build phase.

Build & Deployment Issues

Build and deployment of the applications developed by the software team is complex in nature due to the distributed high resilience eCommerce architecture along with the requirement to integrate with secure authentication services and legacy systems. All deployments must be automated with minimal manual intervention to ensure repeatability through each of the numerous test environments as well as preventing unnecessary access to production servers in the interest of data privacy and security. Scripting and configuration issues subsequently cause delays to the start of the formal testing phase as a result of not attempting deployment of the application until after the build has completed.

2 METHODOLOGY SHAPING

The team has successfully delivered a number of large projects since its inception and as such believes that they must be doing some things well and should continue with or enhance these practices. The team also believes that some of the current practices are not adding value and these should be discontinued. The latter however must be reviewed in the context of the entire development lifecycle to ensure that discontinuing a design or build practice deemed not valuable does not have a detrimental impact on testing, implementation of maintenance.

Having some experience of Post Implementation Reviews (PIR) which are mandated under PMMS, the team agreed that input from and discussion with all members of the team was essential as well as consensus in relation to what are the most important things to address. This is consistent with the Crystal Clear technique of Methodology Shaping as described by Cockburn [2].

Using the Methodology Shaping technique, the team proposes to gather information about prior experiences of individuals and project teams. It will not be possible to get the entire team in a single workshop, nor would this be the most effective approach, so it is proposed to use a combination of interviews and workshops with the end result being two lists:

1) Disliked/Avoid – Practices that have been personally experienced by members of the team on previous projects that they would not like to repeat on the current or next project.

2) Liked/Keep - Practices that have been personally experienced by members of the team on previous projects that they would like to see repeated (and possibly enhanced).

The items on both lists will then be weighed by all individuals within the team to indicate the significance of each and the higher weighed items will be the areas to focus on initially.

Compiling both lists will ensure that consideration is given to eliminating existing practices which are not adding value instead of just enhancing existing or adding new practices.

3 FIRST THINGS FIRST

The team looked at the output from previous PIRs and these all reported that the project teams believed they could have done things better if only they had more time and resources. Two possible solutions may be considered for this complaint. Firstly, make allowances on the next project for more time and/or additional resources. Unfortunately however, these commodities are in short supply due to increasing demand from the bank's business divisions and already challenging timelines for delivery of new products or services to the bank's customers. The second solution calls for an effective time management framework in order to make better use of the time that is available by ensuring that all activities and practices are adding value and are mutually beneficial to all members of the project team.

Stephen Covey's fourth generation time management discipline which he calls First Things First (FTF) [3, 4] provides a matrix against which all activities and practices can be reviewed. Covey says that FTF focuses on preserving and enhancing relationships and on accomplishing results. This emphasis on people and evident results is consistent with the key values outlined in the Agile Alliance Manifesto [5] and is therefore an ideal philosophy to use alongside the practical approach of Methodology Shaping.

	URGENT	NOT URGENT
IMPORTANT	QUADRANT 1	QUADRANT 2
NOT IMPORTANT	QUADRANT 3	QUADRANT 4

Figure 1 – Time Management Matrix

FTF separates activities that are performed into four quadrants as shown below in Figure 1. Two factors define an activity. *Urgent* means it requires immediate attention, whereas *Importance* relates to results.

Quadrant one is the fire-fighting quadrant where things are urgent and important. In the software development lifecycle, critical defects or issues with test environments would fall into this quadrant.

Quadrant two contains activities which are important but not yet urgent. These would include code reviews or end to end integration. There are no immediate consequences of not performing these activities. However if not performed they will result in the creation of urgent and important issues as outlined above.

Quadrant three activities are urgent, but not important. These activities are usually part of someone else's agenda and not aligned with the objectives of the current project. An example may be unnecessary or irrelevant progress reporting.

Quadrant four activities are neither urgent nor important, such as spam emails or meetings with no agenda or objectives, but nevertheless result in interruptions to the important activities.

The key to effectively managing available time according to Covey, either on an individual or team basis is to spend as much time as possible on Quadrant two activities. It is essential to firstly identify and eliminate the activities which are not important (Quadrants 3 and 4), freeing up time for the important tasks (Quadrants 1 and 2). Secondly important activities should be performed before they become urgent. There will always be genuine crises and emergencies, but the emphasis is on being proactive around the opportunities presented in Quadrant 2, thus reducing the time required in Quadrant 1. For example, time spent on code reviews is likely to result in fewer defects encountered during testing.

4 CURRENT STATUS & FUTURE WORK

This is part of ongoing research looking at improving the software process used by the team through experimenting with agile practices. The literature review is continuing and the Methodology Shaping workshops have been scheduled to take place during the next month. The output from the workshops will be presented in a future paper and will also provide the starting point for refining the existing set of working conventions.

REFERENCES

1. Fowler, Martin, Scott Kendall: UML Distilled. Addison-Wesley (1997)
2. Cockburn, Alistair: Crystal Clear: A Human Methodology for Small Teams. Addison Wesley, (1999)
3. Covey, Stephen: Seven Habits of Highly Effective People. Simon & Schuster, (1989)
4. Covey, Stephen: First Things First. Simon & Schuster, (1994)
5. http://www.agilealliance.org