Augmenting Software Development Methodologies with Empirical Scientific Data

Alvin Teh

1 Introduction

Currently, there are many software development methodologies available. These methodologies are not limited to the KBS/TPS 1, Extreme Programming 2 and the venerable Waterfall process 3. In general, however, Software development methodologies are presented and deployed with extreme caution from both the proposer and the implementer. It is acknowledged that structured processes do provide benefits to a software development team, but on the whole, the field remains qualitative and therefore abstract. It is a common state of affairs to encounter various articles challenging aspects of each methodology 4.

1.1 Background: The research focus

The heart of scientific research is scientific rigor. This is the application of scientific methods that are based on results that were obtained from empirically sound experiments. Most current software development methodologies resemble a scientific hypothesis to an ongoing experiment. An example is the TSI 5 or the CMMI 6, where there is still large amounts of activity in consolidating and analyses of deployment results. This problem which has been highlighted in depth recently by Kitchenham et al 7.

It is impossible to verify scientifically an entire proposed methodology, it would be beneficial to build from a set of well understood, empirically valid scientific results.

1.2 Common Identified Problems with Software Development

Analysis into the family of software development methodologies indicate that all software methodologies derive their focus on the continuum of the four factors illustrated in figure 2. Software development methodologies largely differ in how much of each factor is taken as the focus. For example, in agile methods, the focus is more on the rapid and adaptive software factors (to varying degrees). However, it seems that software development problems occur independent of methodologies. Some recently identified group Software Development Issues are:

Strategic Issues - Problems with task division & allocation, motivation clashes and misalignment of benefits

Cultural Issues - Problems with communication structure, author/ group structure and "sense of time"

Inadequate Communication - Problems with communication

Knowledge Management - Problems with efficient and effective Knowledge Transfer/Accumulation of knowledge

Project and Process Management Issues - Problems with hand- off & synchronisation within the project

Technical Issues - Problems with tools, etc.

In a way, it can be taken that software development methodologies attempt to "risk manage" software development issues to manage their impacts. By instituting various processes and operating procedures, the aim is to put in place processes that would minimise Communication issues, for example. It follows that a more in depth study should be conducted of the above troublesome phenomena in Software Engineering. Rather than providing a baseline for evolution ary tactics to minimise these issues in Software Engineering groups, it is proposed to explain the reasons behind these phenomena and treat them directly instead of through evolution.

1.3 Software Engineering Issues: A new perspective

An in-depth review of scientific literature was conducted that revealed that group dynamics and its associated effects and results are extensively studied. Further, it was identified that a particular theory could be used to provide a logical, empirically validated framework from which most of the major group Software Development phenomenon could be extrapolated and explained. This theory is known as the Social Norms Approach (SNA) 10. Social Norms are defined as: "...groups of shared mental models in software development project teams..." 10. Studies have shown that, even if a group has a greater understanding, the rate of knowledge transfer remains the same, although higher transfer of information in advance has been shown to affect the success of groups in high pressure situations 12. Studies have also shown that, due to role differences, intra-team does not establish similar models even over time 14.

1.3.2 Shared Mental Models & Knowledge Management

In order to deliver a correct solution, effective knowledge transfer must occur between developers and customers. However, studies have shown that, even if a group has a greater understanding, the rate of knowledge transfer remains the same, although higher transfer of information in advance has been shown to affect the success of groups in high pressure situations 12. Studies have also shown that, due to role differences, intra-team does not establish similar models even over time 14.

1.3.3 Planning

Studies have shown that software development processes must be tailored to the specific project characteristics in order to minimize development failures. Inadequate planning and project management is considered a leading cause of software development failures 11. Project planning is a critical phase in any software development project. It is a process of establishing the strategic planning for a project and the relationship to the customer & environment, it may be beneficial to select one over the other.

1.3.4 Group Diversity

Group diversity can be divided into either variety or disparity. Variety refers to the differences in knowledge or experience among group members, whereas disparity refers to differences in resources or assets among group members 57. Group diversity as disparity has a stronger effect on the time taken for a group to reach consensus than variety, although they both lengthen this process. A moderate difference between the mental ability increases the group cognitive complexity. This indicates that having a group of slightly varying ability results in a group with a more complete understanding of a problem domain. It was also found that more able groups tended to require better understanding provided a high teamwork environment was established.

1.4 Conclusion: Designer Groups tailored for work & Future Work

The application of empirical results obtained from the area of group psychology, especially SNA shows promising insights into Software Engineering. By consolidating these results, it may be possible to create "designer groups" that are entirely suited for the operating environment sooner, resulting in less effort wasted in trial and error. It also allows the anticipation of potential problems, and various trade-offs - a people centered "risk management system" that can be implemented over time. As a result, a model is being developed that will be used as a framework for utilizing empirical results, which would then be used to understand and establish such "designer groups".

This research is conducted by the ANU College of Engineering & Computer Science.