The Impact of Process Choice in High Maturity Environments: An Empirical Analysis

Narayan Ramasubbu, Rajesh Krishna Balan
Singapore Management University
[nramasub,rajesh]@smu.edu.sg

Abstract

We present the results of a three year field study of the software development process choices made by project teams at two leading offshore vendors. In particular, we focus on the performance implications of project teams that chose to augment structured, plan-driven processes to implement the CMM level-5 Key Process Areas (KPAs) with agile methods. Our analysis of 112 software projects reveals that the decision to augment the firm-recommended, plan-driven approach with improvised, agile methods was significantly affected by the extent of client knowledge and involvement, the newness of technology, and the project size. Furthermore this decision had a significant and mostly positive impact on project performance indicators such as reuse, rework, defect density, and productivity.

1. Introduction

The choice of software development process is considered to be a crucial factor in building systems on time and with high quality. As a result, there are numerous software development process frameworks and methodologies available for software teams. The comparative strengths and weaknesses of the prevalent software development processes have been extensively studied in prior research [for e.g., 1, 2, 3, 7, 16, 20, 21]. However, there is no one universally applicable or unanimous choice of development process. In the face of such diversity, software development organizations continue to invest heavily in software process improvements by standardizing their development processes [10]. This standardization, i.e., minimizing heterogeneity by adopting one uniform process across the firm, has been shown to lead to benefits of improved productivity, quality and cycle time [4, 12, 13, 14].

Firms that have achieved reasonably high levels of process standardization are usually said to be operating at high levels of process maturity as exemplified by the CMM level-5 [24] and Six-Sigma firms [19]. The two institutions used for this study were both assessed at CMM level-5. As such, they mandated the use of a highly structured, plan-based approach for the operationalization of all the CMM Key Process Areas (KPAs). Our goal is to investigate the performance outcomes of project teams that deviated from this mandate by adopting an agile approach to the implementation of the CMM KPAs in their projects.

While there have been commentaries and theoretical arguments from other scholars [7, 8, 9, 11, 17, 18, 20, 25, 28], this study is a first step in the direction to empirically test the "balanced process hypothesis" [8], and investigate the performance impacts of process diversity, albeit in the narrow context of offshore software development. Using data collected from 112 software projects from two leading offshore software development firms, we empirically test the hypothesis if augmenting the standard plan-driven development processes with agile methods leads to superior project performance. Also, through field research we uncover the dominant factors specific to offshore software development that propelled some project teams to choose a non-standard approach to their development processes over the standard processes recommended by their firms.

2. Research sites: high process maturity environments

Consistent with the research goals of this study, we sought to observe the development process choices at firms that employ rigorous, standardized processes, and at the same time provide room for improvisations. We obtained permission to conduct our research at development centers of two different leading offshore software firms that fit our criterion. The development centers involved in our study had been assessed to be operating at the CMM and People CMM (PCMM) level-5, and had won several awards for superior

quality management practices. Thus, this setting provided an ideal environment for conducting our study on development process choices.

The CMM specifies a series of actions that a company can perform to optimize and streamline its internal processes (across all aspects of the company). The Level-5 certification is the highest certification level and indicates that the company has been assessed as having highly optimized processes across 24 Key Process Areas (KPA). Companies are free to implement these KPAs using any process they want as long as they satisfy the goals specified by the CMM. Further details about the CMM certification process can be found elsewhere [22].

Both firms were equivalent in terms of the total number of employees (about 25,000) and annual revenues (about two billion USD) at the start of this study. Both firms have a centralized Software Engineering Process Group (SEPG) that is responsible for governance of development processes. The SEPG teams at both the firms invested heavily in standardizing the development processes prevalent at the firms. The standard, firm-wide recommended processes, at the time of this study, at both firms, were highly structured, plan-based approaches for each of the KPAs of the CMM. Any non-standard processes employed by individual projects needed prior approval and were actively monitored by the SEPGs. To maintain flexibility, tailoring of the standardized set of processes (to use more agile methods for specific KPAs for example) was allowed at the individual project level. However all process tailoring was closely monitored by Software Quality Assurance (SQA) personnel who were part of the project team. The SQA personnel at both the firms reported directly to the SEPG managers and not to the respective project managers. We discuss the process choice variation across the projects in more detail in Section 3.1.1.

Through an analysis of the weekly reports submitted by the SQA personnel in charge of individual projects, we observed the incidents of non-compliance to standardized development processes. While the individual SQA personnel were authorized to intervene and correct deviations due to human errors and lethargy, voluntary and systemic deviations had to be discussed at the organizational level SEPG meetings. It was through our participation in these internal, non-compliance meetings that we discovered that some managers wanted to augment the heavily structured, documentation oriented, plan-based processes widely prevalent at the firms with agile processes.

In particular, several project teams were using agile processes to operationalize the CMM KPAs instead of the standardized process templates recommended by the SEPG. Since these deviations did not affect the overall strategic mandate of the firms, which was to be a "CMM level-5" company, the top management at our research sites did not see such non-standard process choices as an immediate threat. However, they were curious to understand the implications and performance outcomes of these non-compliances in the long run. This coincided well with our research interest, and hence our research goals struck a chord with the interests and motives of the executives at our research sites.

Thus, we set out to answer the following open empirical questions at our research sites:

1. Are there specific conditions that encourage the shift from standardized processes to non-standard software development processes?
2. Do the non-standard approaches that mixed plan-based processes with agile methods lead to significant improvements in eventual project performance?

3. Modeling the causal links: process choice and performance

The first empirical research question raised in the study deliberates the reasons for the adoption of non-standard development processes that augment plan-based approaches with agile practices, and the second question seeks to analyze the causal effects due to the adoption of non-standard development processes. However the research setting in this study does not facilitate experimentation – recall that we are only observing real world projects that are being consumed by real customers who pay for the software application. Hence experimentation, especially induced by external researchers, is not feasible. In the absence of experimental data, we use observational data collected through our field research. We then employ a propensity score stratification analysis method to infer the causal relationships raised by our research questions. This method has been utilized by researchers from other fields to investigate causal effects using observational data [33]. We operationalized our research method in the following steps:

Step-1: Through interviews, discussions, non-intrusive observations, and surveys we identify the possible list of factors that drive the adoption of the non-standard process implementations at our research sites.

Step-2: After collecting data on the variables identified in step 1 and statistically verifying them, we estimate the probability of a project team

adopting a non-standard process. This is called the propensity score for a project.

Step-3: We separate the sample of our projects into "treated" (the projects that adopted non-standard processes) and "controls" (the projects that selected the standard processes). Using the propensity scores calculated in step 2, we "match" the individual projects in the "treated" sample with similar projects in the "controls" sample.

Step-4: By statistically comparing the performance outcomes of the "treated" projects with the matched projects from the "controls" group (i.e., projects with similar propensity score have a similar probability of adopting a non-standard process), we draw inferences on the performance impacts of the process choice.

3.1. Observational data collection

Our data collection effort was spread across a three year time period. In this period, we followed 112 software projects from start to finish, and gathered detailed data on the software processes and project performance of each of these projects. For the first eleven months of the data collection period, one of the authors was present in the field and observed project activities on a day-to-day basis using a non-intrusive approach. For the rest of the three year data collection period, we conducted separate weekly teleconferences with a volunteer from each firm's SEPG department. These volunteers helped us collect regular process and performance data for each of the projects that were being studied. The volunteers were neutral observers and were not affiliated with any of the projects in our sample. Out of the 112 projects, 34 project teams employed non-standard process.

We obtained the data required for this study through interviews, surveys as well as from the internal process databases maintained by the quality division of each firm. As part of the CMM process, each project was required to accurately and consistently report the data used for this paper. We randomly sampled portions of the data at regular intervals to check for accuracy and consistency (either directly when we were onsite or through the help of the volunteers). In addition, the data for all the 112 projects used in our study was audited, and verified as correct, by the quality control group of each firm. Furthermore, except for the data from the 34 projects that followed non-standard software development processes, all data used in this study was audited by external agents as part of the regular CMM level-5 compliance checks. We are thus confident that the data used in this paper is reliable and of high quality and that we have a rich understanding of the context in which these software projects were executed.

In the rest of the section, we describe the individual variables used for this analysis. These variables were intentionally chosen with an aim to practically deploy the methodology developed in this study in real projects.

3.1.1. Development process choice variable. When a new software project is initiated, the project manager, along with the development team, can choose to follow the standard development process that is prevalent in the firm or follow a new or non-standard development process for the project. As mentioned earlier, at both our research sites the standard company approved process for implementing each of the CMM level-5 KPAs was a highly structured plan-based approach.

Project teams that choose to follow non-standard development processes had to seek permission from the central Software Engineering Process Group (SEPG). At the time of our data collection, 34 projects had been formally approved to use non-standard processes to implement the CMM level-5 KPAs for their projects. All of the non-standard processes used were agile methods – some projects used versions of agile RUP while others used versions of XP or SCRUM.

Each of these 34 projects used different process choices across different KPAs (i.e., none of the 34 projects used similar processes across all 24 KPAs compared to every other project). Hence, to obtain statistically significant results, we grouped all the 34 projects that used a non-standard process for at least a KPA together. By doing this, we are still able to meaningfully quantify the performance impact of choosing at least one non-standard process for the project. The variable 'Development Process Choice' is thus a binary variable that specifies the development process that was used. The non-standard development process is assigned the value of 1 and the standard, firm-recommended development process is assigned the value of 0.

We plan to collect more data about projects using non-standard processes in the near future and then use the larger data set to tease out the effects of particular process choices on particular KPAs in future research.

3.1.2. Performance outcome variables. For this work, we use five different performance indicators to quantify the goodness of the software development process chosen. These performance indicators were:

1) **Development productivity**: Development productivity is defined as the ratio of software

code size in KLOC to the total development effort in person-hours.

2) **Defect Density:** Defect density is defined as the number of unique problems, per KLOC, that were reported, before project signoff, by customers during the acceptance tests and production trials. It is calculated as follows:

$$\text{Defects Density} = \frac{\text{Defects}}{\text{Code Size (KLOC)}}$$

3) **Reuse:** Reuse in this study is measured as the amount of project code, measured as a percentage of the total project code size, which was obtained from the central generic code libraries maintained by the two data collection sites. Reused modules and objects were easy to find and count as every project we studied explicitly tagged reused code with unique identification. This was to make it easy to find and replace generic code where necessary.

4) **Rework:** It is measured as the percentage of total actual project hours spent on fixing bugs reported by customers during acceptance tests and during the warranty period.

5) **Project Management Effort:** This variable is measured as the percentage of total actual project hours spent on project management activities. This data was retrieved from the internal time sheets of the project manager.

3.1.3. Client specific knowledge. Client specific knowledge was measured through a project manager survey before the start of the project. The survey had the following six items measured on a 7-point scale (1 indicated no knowledge at all while 7 indicated complete knowledge):

1. *How well do you know the project objectives of the client?*
2. *How well do you know the business processes of the client?*
3. *How well do you know the business rules of the client?*
4. *How well do you know the IT infrastructure of the client?*
5. *How well do you know the IT norms and standards followed by the client?*
6. *How well do you know the interoperability constraints of the client's IT infrastructure?*

These survey items were adapted from prior information systems and management studies [26, 27]. The average score of the six survey items provided the score for the 'client specific knowledge' variable.

3.1.4. Extent of client involvement. This variable is the estimated percentage of time (relative to the total project time) that the client would spend with the development team. This information was extracted from the contractual agreement documents signed by the client and the offshore vendor at the start of the project.

3.1.5. Design and technology newness. The design and technology newness variable measures how familiar the project team is with the technology and design concepts needed for a new project. This variable was measured through a survey that was administered before the start of each project. We reused the survey questionnaire previously used by Takeishi [26] to measure design and technology newness.

To measure the design newness, the technical lead of each project (not the project manager) was asked to answer the following question: "For this project that you are starting out, please rate the design newness involved using the following 5-point scale." The five provided answers were:

1. *No modification of design involved.*
2. *Some modification (changes were less than 30%) of design that had been already developed at your company.*
3. *A medium scale modification (30–60%) of design that had already been developed at your company.*
4. *A major modification (more than 60-80%) of design that had already been developed at your company.*
5. *Radically different design that is new to your company.*

To measure the technical newness, the technical lead was asked the following question: "For the design choice you have made for this project, please rate the technology used to implement the design using the following 5-point scale". The 5-point scale went from (1) I am very familiar with the technology to (5) a completely new and unfamiliar technology. After verifying (through factor analysis) that the individual scores for the two-sub items contributed to a common construct, we averaged the scores of the two sub-items to obtain the overall score for design and technology newness.

3.1.6. Estimated project effort. Estimated project effort is the total person-hours estimated for the project. We obtained this, at the start of the project, from the project manager's project planning and estimation charts.

3.1.7. Allocated team size. Team size is the headcount of the number of persons allocated for the project at the start of the project.

3.1.8. Estimated code size. Estimated code size is the estimated KLOC of the project. This was measured, at

the start of the project, from the project planning and estimation charts of the project manager.

3.1.9. Data without any variance. We collected the professional work experience of the team members, the professional work experience of the project managers and the attrition rate of employees in the projects from the human resource department of the firms. However, these variables demonstrated no significant variance across the projects and we thus did not utilize them in our empirical analysis.

The average professional work experience of the team members in our sample was 36 months, the average professional work experience of the project managers in our sample was 108 months and the average attrition rate in the project sample was 6.2%. We believe that the high maturity of PCMM practices and industry best practice human resource policies implemented at the firms could be one reason for the lack of significant variance of these parameters in our sample.

3.2. Propensity score from the empirical data

After we collected our data, we developed the empirical formulations to calculate the propensity scores (i.e., the probability of a project team adopting a non-standard software process) needed to validate our model. Equation 1 presents our final regression model with the coefficients and an error term.

Development process Choice =	$\alpha_0 + \alpha_1$* (client specific knowledge) + α_2* (extent of client involvement)+ α_3 * (design and technology newness) + α_4* (estimated project effort) + α_5* (allocated team size) + α_6* (estimated code size) + ε_1 ..... **(Eq. 1)**

4. Results of analysis

In this section, we present the results of our propensity score analysis. We first show the validity of our model (using historical data – in this case, the data from all 112 projects) and then use the model to show the performance implications of using a non-standard process.

4.1. Results: validation of model

In this section we provide validation results for our propensity score model. We used the logistic regression method to estimate the coefficients of Equation 1 as the dependant variable (development process choice) is a binary variable. The summary statistics of the variables that were used to estimate the regression coefficients are presented in Table 2 and the regression results are presented in Table 1.

Overall, the results of our regression analysis indicate that our empirical specification for determining the probability of a particular project adopting a non-standard development process is valid. The model Chi-Squared statistic value is significant at 1% level, indicating that our model is statistically valid. Unlike linear regressions, the coefficients obtained from logistic regression are difficult to interpret directly. This is because the values of the coefficients indicate the extent to which a unit increase in each of the corresponding input variables would increase the log odds of the dependent variable (development process choice). To make it easier to visualize the true relationships between the input variables and the dependent variable, we plotted the following five graphs: for each of the five significant input variables (allocated team size did not have a significant effect on the results), we kept the other variables at their mean levels (shown in Table 2), and plotted that variable against the probability that the variable would result in choosing the non-standard development process choice. These graphs are shown in Figures 1 to 5.

From our empirical results, we observe that larger projects (in terms of development effort and code size) (Figure 4 and 5) as well as project teams that had a larger extent of client-specific knowledge (Figure 1) demonstrated a lower probability to adopt the non-standard development processes. This result seems to be reflective of the risk minimizing strategies of the project teams when they are faced with large projects and with familiar clients. We also observe that the projects using completely new technology and design (Figure 2) were more likely to use the non-standard development process. This empirical result indicates that the project teams at our research sites responded to external risks such as dealing with newer technology and design by adopting more agile processes.

In addition, our empirical results show that when clients were involved to a larger extent with the project teams (Figure 3), they were less likely to adopt agile development processes. This result is surprising given that the agile process manifesto emphasizes customer centricity. When we analyzed this result further, through discussions with managers and executives at our research sites, we learnt that larger client involvement in offshore (and outsourced) software development might also mean that the clients wanted more control. When viewed from the project control and monitoring perspective, our empirical results make intuitive sense as plan-based methods are a natural choice when the emphasis is on detailed documentation

to help monitoring and auditing. We further discuss this in Section 5.

Variables	Process Model Choice	
Client specific knowledge	**-.546*** (0.000)**	α_1
Extent of client involvement	**-.003*** (0.000)**	α_2
Design and technology newness	**1.905*** (0.000)**	α_3
Estimated project effort	**-.001** (0.029)**	α_4
Allocated team size	.0185 (0.864)	α_5
Estimated code size	**-.001*** (0.000)**	α_6
Constant	-.527 (0.878)	α_7
Model Chi-Squared Statistic	**63.18*** (0.000)**	
Observations	**112**	

Probability-values are shown in parentheses; results significant at 5% are indicated by **; results significant at 1% are indicated by ***. Other values, which are not in bold, are not statistically significant. The model chi-squared statistic indicates that the goodness-of-fit of our model is high. We use a two-tailed hypothesis test for deriving all the p-values (i.e., we did not assume any positive or negative direction of the result while testing).

Table 1. Regression Results

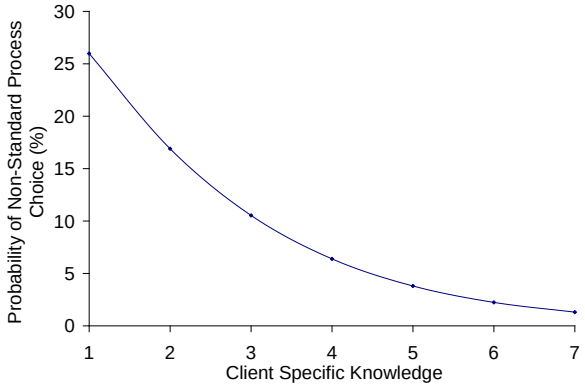

Figure 1. Effect of Client Specific Knowledge

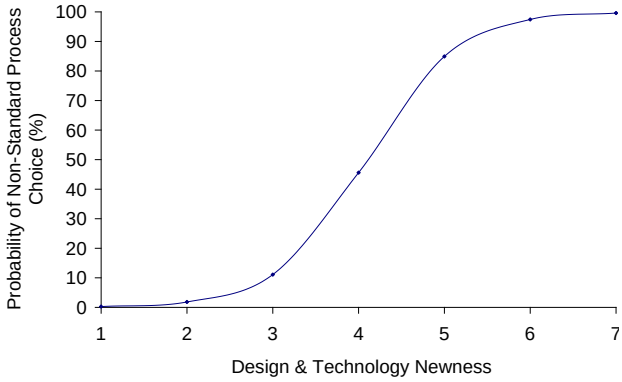

Figure 2. Effect of Design and Technology Newness

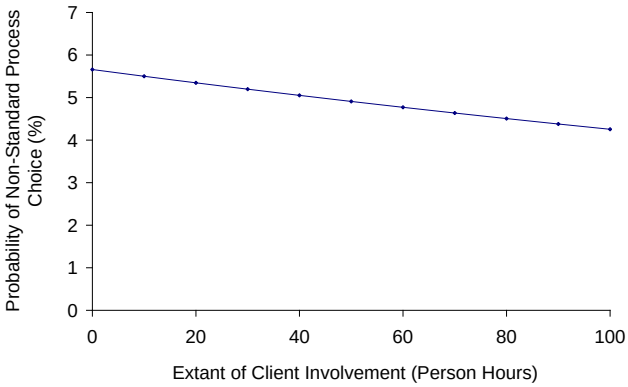

Figure 3. Effect of the Extent of Client Involvement

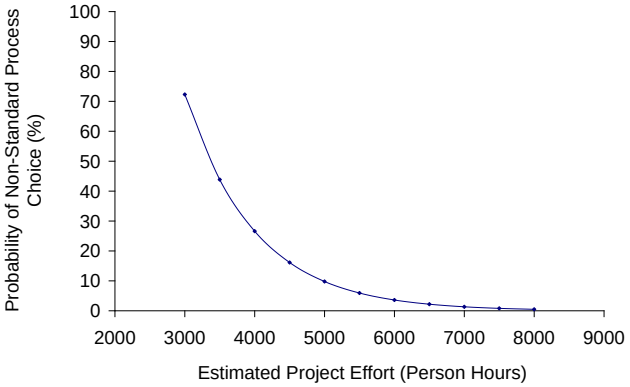

Figure 4. Effect of the Planned Project Effort

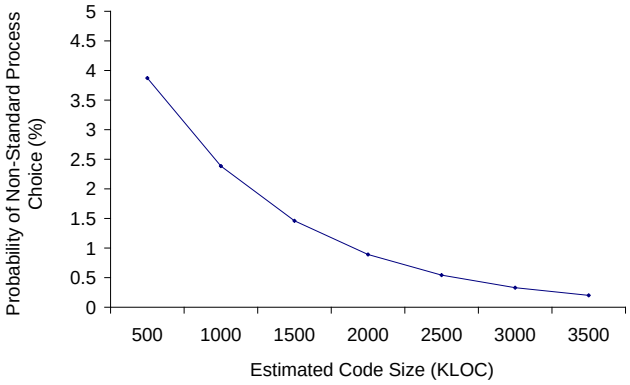

Figure 5. Effect of Estimated Code Size

Finally, our empirical results show that the best predictors of process choice, in terms of probability of choice being made, are the design and technology newness metric (Figure 2), the planned project effort (Figure 4), and the client specific knowledge metric (Figure 1). The other two indicators (estimated code size (Figure 5) and extent of client involvement metric (Figure 3)) have much smaller but still statistically significant effects.

Variable	Unit	Mean	Std. Dev.	Min	Max
Model Input Variables					
Client specific knowledge	Unit-less measure. (7 is best, 1 is worst)	4.36	1.96	1	7
Extent of client involvement	% of total project hrs	22.68	18.75	0	100
Design and technology newness	Unit-less measure. (5 is best, 1 is worst)	2.58	1.28	1	5
Estimated project effort	Person-hours	4864.24	4970.68	17	27231
Allocated team size	No. project personnel	11.51	7.49	2	36
Estimated code size	KLOC	169.74	404.99	0.37	3200
Process Performance Indicators					
Productivity	KLOC / total project hours	213.04	1470.93	0.16	12923.42
Defect Density	Total Delivered Errors / KLOC	0.01	0.02	0.00	0.18
Rework	% of total project hrs	10.00	7.00	0.00	27
Reuse	% of total project KLOC	22.31	27.13	0.00	100
Project Management Effort	% of total project hrs	10.08	9.83	0.14	55.44

Table 2. Summary Statistics of the Variables Used in the Analysis

4.2. Results: process choice matters

Following the statistical validation of the propensity score calculation model, we estimated the probability of adoption of non-standard development processes for all the projects in our sample. Then we stratified our project sample in to 1) a treated sample which consisted of the projects that adopted the non-standard development processes and 2) a control sample that consisted of the projects that adopted the firm recommended standard plan-driven development processes. We then proceeded to test the hypothesis if the performance outcomes of the projects in the treated sample were different from those in the control sample.

The first step in this analysis was to identify projects that are equivalent to each other, except for their process choice decision. This matching step is important because, as mentioned before, our data is collected from field research and not through experiments. We did not have full control over all experimental parameters and thus cannot be completely sure that the choice of development process was not being impacted by unknown variables. As such, a simple mean difference test between the two samples could yield biased results as there may have been hidden influences affecting the choice of process.

To account for this possible non-random influence, we utilize our validated empirical model of process choice to calculate a propensity score (a measure of how close things "match") for each project. Using these propensity scores and a set of matching algorithms (Kernel matching, nearest neighbor matching, stratification matching, difference-in-difference method) we match each project in our sample to its nearest equivalent. We then place all projects in equivalent sets based on their propensity scores, and then compare the performance indicators of projects in the same set that have different software development choices. By doing this matching step, we are able to compare equivalent projects (across the six input variables) and minimize the influence of unobserved variables similar to a controlled lab experiment.

Table 3 shows the results of comparing the performance indicators of projects within each set that have different development process choices.

Performance Variable	Treated Sample Value	Matched, controls Sample Value	Difference	P-Value
Productivity	**101.234**	**27.846**	**73.388**	**0.057***
Defect Density	**0.007**	**0.002**	**0.006**	**0.003****
Rework	**1.224**	**5.290**	**-4.066**	**0.003****
Reuse	**22.857**	**6.000**	**16.857**	**0.014****
Project Management Effort	11.982	4.964	7.017	0.178

Results significant at 5% are indicated by *; results significant at 1% are indicated by **. Other values, which are not in bold, are not statistically significant. We use a two-tailed hypothesis test (i.e., we did not assume any positive or negative direction of the result while testing). Higher scores are better for Productivity and Reuse with the opposite being true for Defect Density and Rework. The unit for each of these variables is the same as shown in Table 2.

Table 3. Effect of Non-standard Process Choice

Our results from this analysis indicate that projects that adopted a non-standard development processes, by augmenting plan-driven processes with agile methods, performed significantly better than their counterparts (treatment = 1 is non-standard development process choice) in terms of improved productivity, and higher reuse levels. However, we do see a minor increase in the defect density levels in the projects that chose the non-standard development processes. At the same time, we also notice that the required effort to fix these errors, as captured by the rework performance variable, significantly reduced. We did not notice any significant differences between the samples in terms of project management effort spent on the projects.

These results are significant as they show that, even in high process-maturity environments, deviating from the established processes can result in significant, non-trivial project performance improvements. Project managers could thus use this model, at the start of the project, to decide if changing some of the processes used for the KPAs would result in better project performance. Currently, our model only tells a manager whether using a non-standard process would result in performance improvements over the standard process. In the future, we plan to augment our model to provide specific process choices (i.e., use a particular process for a particular KPA).

4.2. Results: summary

Overall, our empirical analysis supported the "balanced process" hypothesis that augmenting plan-driven processes with agile methods can lead to improved performance. Also, we showed that five of the six input variables have significant effects on the project's performance. These five input variables, which can be measured at the start of a project, can be reliably used to predict the development process choice (standard plan-driven or non-standard agile) that should be used by a starting software project. Finally, these results serve as a rigorous empirical support for prior theoretical arguments advanced by other scholars [for e.g., 7, 9, 17].

5. Discussion

In this section, we discuss the robustness and the limitations of this study as well as provide some intuitive explanations for some of the observed effects.

5.1. Robustness of model

We checked the robustness of our empirical model and coefficients in the following ways: first, we clustered our data according to the firm the samples came from and obtained robust variance estimations for the coefficients of the propensity score model. Next, we performed checks for multi-collinearity and the effects of outliers before finalizing the results [6]. All these checks indicated that our final model and coefficients were robust. We conducted additional checks and sensitivity analyses to ensure the robustness of the propensity scores and treatment effect results shown in Table 3. First, we ensured that in each matching category of projects, the mean propensity score was not different between the treated and control samples. Secondly, while analyzing the treatment effects, we compared our results by employing different matching algorithms (kernel matching, nearest neighbor, stratification). Our results did not significantly vary according to the matching algorithm we used.

Gamma	Upper-bound significance level	Lower bound significance level	Model break down point
Productivity			
1 (typical bias)	0.057	0.057	
1.5 (moderate bias)	**0.149**	**0.014**	**Break down point**
2 (High bias)	0.246	0.004	
Rework			
1 (typical bias)	0.003	3	No Breakdown
1.5 (moderate bias)	0.00	0.014	
2 (High bias)	0.000	0.03	
Re-use			
1 (typical bias)	0.014	0.014	No Breakdown
1.5 (moderate bias)	0.040	0.002	
2 (High bias)	0.090	0.000	
Defect Density			
1 (typical bias)	0.023	0.023	
1.5 (moderate bias)	0.072	0.004	
2 (High bias)	**0.131**	**0.009**	**Break down point**

Breakdown point indicates that the significance level is > 0.1

Table 4. Sensitivity of Treatment Effects

Finally, we checked the sensitivity of our results using the Rosenbaum bounds method [5], the results of which are presented in Table 4. In this sensitivity testing method, the probability (log odds) of differential assignment due to unobserved factors (Gamma) is repeatedly varied to increase the bias introduced in our empirical model. The increased bias

eventually breaks down our model and the treatment effects completely vanish (i.e., the upper-bound significance level becomes > 0.1). The sensitivity analyses of our model, shown in Table 4, indicate that the model developed in this study is robust and does not breakdown within reasonable ranges of the artificially introduced bias. In particular, the model only breaks down at 1.5 bias or higher – not at regular 1.0 bias levels. This shows that the model is quite robust to normal data variations.

5.2. Qualitative insights of results

In this section we provide some qualitative insights on how the adoption of agile methods to implement some of the CMM KPAs, helped the project teams to perform better. In addition to project size related factors, client specific knowledge, client involvement in the projects, and the design and technology newness were the other important considerations that project teams at our research sites used in deciding whether to adopt agile methods.

5.2.1. Impact of newness. Our observation of the project teams revealed that the main problem they faced when given a project involving new clients or new technology was the inadequacy in the organizational process templates to address their specific needs. For example, we noticed that the cost and effort estimation formulas and guidelines developed by the in-house SEPG teams were not able to handle projects that involved heavy use of the emerging scripting languages and business process modeling languages. Projects that involved porting and integrating business applications across multiple enterprise environments (for example, applications integrating SAP's Finance and Control module with Siebel's CRM system) also had considerable difficulty in using the firm recommended estimation tools. The Mean Error in Estimation (MRE) when using the firm's standard estimation templates for the 34 projects that chose the non-standard approach was more than 50%. (Note that the MRE was calculated by us only after the respective project closure event to corroborate our insight).

We noticed that teams faced with such situations benefitted when they adopted agile planning methods. For example, some project teams chose a rapid time-boxing-based planning approach (in-depth task planning only for very short term activities; tasks in the longer term planned only in broad strokes). With this planning method, individual team members faced more uncertainty over their tasks in the project. However, overall as a team, they had more opportunities to help each other and conduct community-based learning programs. We believe that these community-based learning interventions during the course of the project could have contributed to the better performance of these teams (as observed in the aggregated results). A detailed analysis on such performance enhancing learning methods can be found in [23].

5.2.2. Impact of client's perceptions on control. In understanding the role of the client's involvement in influencing the process choices of teams, we observed that the client's perception on control of offshore software development played a key role. Control refers to the set of mechanisms designed to manage the processes and individuals such that the desired objectives are achieved [15]. At our research sites, we noticed that the clients who preferred behavioral-control mechanisms over outcome-based control mechanisms were more involved with the offshore team members. Use of behavior-control mechanisms emphasize the specification of detailed procedures for tasks and the monitoring of adherence to these procedures. On the other hand outcome-based control mechanisms specify only the final goals of the projects and the monitoring of whether the final project goals were met. Thus, behavior-controls expect a plan-driven approach and hence we notice that the project teams that dealt with clients emphasizing these control approaches had lesser propensity to choose non-standard process approaches.

5.3. Putting the model to practice

We believe that the process choice model developed in this paper can be put to general practice at most software engineering process groups. The necessary prerequisites for operationalizing our model for real world projects are threefold; First, the firm adopting our model should have historical project performance and process data that can be used to estimate our model in the firm's particular context (e.g., generate company specific coefficients for the regression model shown in Section 3.2). Second, there is a need to diligently track the process variations implemented at the firm (to determine standard and non-standard process choices). Finally, detailed data collection (to obtain the project-specific inputs to the model), through both surveys and objective data gathering, is necessary even before a project's development activities are initiated. These requirements suggest that project teams wanting to adopt the models and methods described in this paper need to be operating in a reasonably mature process

environment where changes and process deviations happen in a controlled environment.

5.4. Limitations of study

This study has a number of limitations which we list and discuss in this section.

5.4.1. Domain specificity. First, the empirical context of the study might limit the generalization of our results to the types of firms we measured: highly mature offshore development firms that specialize in developing custom enterprise business solutions. It is not clear if these results will apply to other organizational scenarios. Further, we studied only software development projects, and our sample did not include data from other types of software activities such as maintenance or reengineering.

5.4.2. Binary coding of process choices. In this paper, we limited our treatment of process choices to a binary decision – standard and non-standard process. In particular, as long as a particular project changed any of the firm mandated implementation mechanism for the 24 CMM KPAs, we considered it a non-standard process. We did this to primarily build a simple, parsimonious empirical model to capture the causal effects of process deviations on performance. This approach allows us to focus on the validity and viability of the broader phenomena of interest: whether augmenting structured, plan-based methods with agile processes can lead to positive outcomes. This approach is still useful for managers to determine the effects of one process choice over another. We are working on techniques to remove the binary limitation in our model – allowing us to handle a much richer set of process choices.

5.4.3. Non-unified process model. In this paper, we presented a model that examined values for the six process performance indicators based on the chosen development process. However, these values still have to be manually interpreted before a final decision regarding the viability of the chosen process can be made. Hence, to obtain best results, project managers will have to understand the relationships between the five performance indicators when using this model.

We are currently developing methods that can automatically combine the five performance indicators, accounting for the individual risks and tradeoffs, using a theoretically sound formulation, and output a single clear indication of whether the selected process is beneficial or otherwise. These automatic methods would, in particular, allow the model to be easily used by all project teams in a company – with no regard for their technical competency or management capabilities.

5.4.4. Limitation of vendor focus. Accounting for the individual risks and tradeoffs faced by a project team beyond the development environment factors included in our model necessitate a client-vendor dyadic study. We could not get the approvals to collect individual customer data in detail and hence could not perform the required dyadic study. We thus limited this paper to analyzing process choice impacts using influencing factors drawn solely from the vendor's development environment.

7. Conclusion

In this paper, we analyzed process deviations in a highly structured, plan-driven offshore development environment to empirically test the "balanced process" hypothesis, i.e., if augmenting plan-driven development processes with agile methods lead to superior project performance outcomes. We first discovered, through our field research, the key factors that influence process choice decisions in offshore software development. We then developed a propensity score based empirical model to analyze the causal linkages between process choices and five key project performance outcomes. Our results show that augmenting the highly structured plan-driven processes employed in offshore software firms with agile practices can lead to superior performance outcomes. This study also shows that it is possible for software managers to decide a-priori on the development process choice that is most likely to achieve relatively better performance for their projects.

We are working on better understanding the individual risks and tradeoffs that each process deviation brings to the fore by conducting a detailed field test of our model at the two firms. This will allow us to understand a) how to combine various performance indicators to obtain process choice decisions, and b) the effort of specific process choices on specific projects/KPAs. To accomplish this, we are taking a longitudinal study approach by observing the long term effects of the process choices made by the teams at these firms.

8. References

[1] I. Aaen, J. Arent, L. Mathiassen, and O. Ngwenyama, "A conceptual map of software process improvement," *Scandinavian journal of information systems*, vol. 12, pp. 123-146, 2001.

[2] P. Abrahamsson, J. Warsta, M. T. Siponen, and J. Ronkainen, "New directions on agile methods: a comparative analysis," in *25th International Conference on Software Engineering*, Portland, OR, 2003, pp. 244-254.

[3] P. J. Ågerfalk and B. Fitzgerald, "Flexible and distributed software processes: old petunias in new bowls?: Introduction," *Communications of the ACM*, vol. 49, pp. 26-34, 2006.

[4] M. Agrawal and K. Chari, "Software effort, quality, and cycle time: a study of CMM level 5 projects," *IEEE Transactions on software engineering*, vol. 33, pp. 145-156, 2007.

[5] S. O. Becker and M. Caliendo, "Sensitivity analysis for average treatment effects," *The Stata Journal*, vol. 7, pp. 71-83, 2007.

[6] D. A. Belsley, E. Kuh, and R. E. Welsch, *Regression Diagnostics: Identifying influential data and sources of collinearity*. New York: John Wiley & Sons, 1980.

[7] B. Boehm, "Get ready for agile methods, with care," *IEEE Computer*, vol. 35, pp. 64-69, 2002.

[8] B. Boehm and D. Port, "Balancing Discipline and Flexibility with the Spiral Model and MBASE," *CrossTalk*, vol. Dec 2001, pp. 23-28, 2001.

[9] A. Cockburn, "Selecting a project's methodology," *IEEE Software*, vol. 17, pp. 64-71, 2000.

[10] B. Curtis, "The global pursuit of process maturity," *IEEE Software*, vol. 17, pp. 76-78, 2000.

[11] M. Deck, "Managing process diversity while improving your practices," *IEEE Software*, vol. 18, pp. 21-27, 2001.

[12] D. E. Harter, M. S. Krishnan, and S. A. Slaughter, "Effects of process maturity on quality, cycle time, and effort in software product development," *Management Science*, vol. 46, pp. 451-466, April 2000 2000.

[13] D. E. Harter and S. A. Slaughter, "Quality Improvement and Infrastructure Activity Costs in Software Development: A Longitudinal Analysis," *Management Science*, vol. 49, pp. 784-800, June 2003 2003.

[14] J. Herbsleb, D. Zubrow, D. Goldenson, W. Hayes, and M. Paulk, "Software quality and the capability maturity model," *Communications of the ACM*, vol. 40, pp. 30-40, June 1997 1997.

[15] L. Kirsch, S. V, D.-G. Ko, and R. L. Purvis, "Controlling information systems development projects: the view from the client," *Management Science*, vol. 48, pp. 484-498, 2002.

[16] D. H. Kitson and S. M. Masters, "An analysis of SEI software process assessment results: 1987-1991," in *15th International Conference on Software Engineering*, Baltimore, MD, 1993, pp. 68-77.

[17] M. Lindvall and I. Rus, "Process diversity in software development," *IEEE Software*, vol. 17, pp. 14-18, 2000.

[18] M. Lycett, R. D. Macredie, C. Patel, and R. J. Paul, " Migrating agile methods to standardized development practice," *IEEE Computer*, vol. 36, pp. 79-85, 2003.

[19] P. S. Pande, R. S. Neuman, and R. R. Cavanagh, *The six sigma way: How GE, Motorola, and other top companies are honing their performance*: McGraw-Hill Professional, 2007.

[20] M. C. Paulk, "Extreme programming from a CMM perspective," *IEEE Software*, vol. 18, pp. 19-26, Nov 2001 2001.

[21] M. C. Paulk, "How ISO 9001 compares with the CMM," *IEEE Software*, vol. 12, pp. 74-83, Jan 1995 1995.

[22] M. C. Paulk, B. Curtis, M. B. Chrissis, and C. V. Weber, "Capability Maturity Model," *IEEE Software*, vol. 10, pp. 18-22, 1993.

[23] N. Ramasubbu, S. Mithas, M. S. Krishnan, and C. Kemerer, "Work dispersion, process-based learning, and offshore software development performance," *MIS Quarterly*, vol. 32, p. *in press*, 2007.

[24] D. J. Reifer, "Profiles of Level 5 CMMI Organizations," *Journal of defense software engineering*, vol. 2007, p. http://www.stsc.hill.af.mil/crosstalk/2007/01/0701Reifer.html, 2007.

[25] D. J. Reifer, "XP and the CMM," *IEEE Software*, vol. 20, pp. 14-15, 2003.

[26] A. Takeishi, "Knowledge partitioning in the interfirm division of labor: The case of automotive product development," *Organization Science*, vol. 13, pp. 321-338, May-June 2002 2002.

[27] A. Tiwana, "Knowledge partitioning in outsourced software development: A field study," in *International conference on information systems*, Seattle, 2003.

[28] L. Williams and A. Cockburn, "Agile software development: it's about feedback and change," *IEEE Computer*, vol. 36, pp. 39-43, 2003.