Overcoming the Challenges in Cost Estimation for Distributed Software Projects

Narayan Ramasubbu
School of Information Systems
Singapore Management University
Singapore
nramasub@smu.edu.sg

Rajesh Krishna Balan
School of Information Systems
Singapore Management University
Singapore
rajesh@smu.edu.sg

***Abstract*—We describe how we studied, in-situ, the operational processes of three large high process maturity distributed software development companies and discovered three common problems they faced with respect to early stage project cost estimation. We found that project managers faced significant challenges to accurately estimate project costs because the standard metrics-based estimation tools they used (a) did not effectively incorporate diverse distributed project configurations and characteristics, (b) required comprehensive data that was not fully available for all starting projects, and (c) required significant domain experience to derive accurate estimates. To address these challenges, we collaborated with practitioners at the three firms and developed a new learning-oriented and semi-automated early-stage cost estimation solution that was specifically designed for globally distributed software projects. The key idea of our solution was to augment the existing metrics-driven estimation methods with a case repository that stratified past incidents related to project effort estimation issues from the historical project databases at the firms into several generalizable categories. This repository allowed project managers to quickly and effectively "benchmark" their new projects to all past projects across the firms, and thereby learn from them. We deployed our solution at each of our three research sites for real-world field-testing over a period of six months. Project managers of 219 new large globally distributed projects used both our method to estimate the cost of their projects as well as the established metrics-based estimation approaches they were used to. Our approach achieved significantly reduced estimation errors (of up to 60%). This resulted in more than 20% net cost savings, on average, per project – a massive total cost savings across all projects at the three firms!**

***Keywords*-Globally distributed software development; software engineering economics; cost estimation; case-based reasoning; analogies; project management; learning.**

I. INTRODUCTION

Despite substantial improvement in our understanding of the performance of globally distributed software projects [1, 2] and the implications of software process choices [3], early stage project cost estimation remains a significant challenge in globally distributed software projects. In this paper, we report our experience in examining this challenge through field studies conducted at three large ISO-9001 certified, high process maturity firms that were also assessed at the CMMI Level-5. All three firms were using the current industry-standard, metrics-driven, regression-based cost estimation techniques. However, the early stage cost estimates for newly starting projects of the firms were not accurate (up to about 300% variance in mean estimation errors). This led to extremely inaccurate estimates of projects costs resulting in significant losses to the three companies as they were predominantly using fixed-price contracting (with penalty or bonus clauses) where projects could not be re-priced after project initiation. Hence, without good estimates, the companies were frequently either under-quoting (resulting in losses) or over-quoting (resulting in competitors getting the contract) their bids for client projects.

The companies realized that fixing this problem internally was extremely hard as, ironically, the high process maturity industry-standard cost estimation techniques were ingrained into every layer of their process hierarchy – from project manager training to the auditing system to the metrics collection mechanisms – all the way to the reporting systems. Thus, we were approached by the three firms to devise a better cost estimation solution that would allow them to improve their performance. We designed our solution in three steps: 1) use action research to identify, in-situ, the key reasons for the inaccurate estimates coming from the existing regression-based methods, 2) development of a new cost-estimation solution that could overcome the problems identified with the old system, and 3) field testing of the new system (in parallel with the old system) over a six month period with real managers and projects to prove definitely that it can significantly improve the estimation of costs for distributed software projects.

We found that the existing regression-based methods failed due to three main reasons: 1) they did not account for the differences intrinsic to the different globally distributed software project configurations, 2) the methods required comprehensive metrics data that was often not available at the start of a project, and 3) the methods were very sensitive to the experience level of the project manager using them. As a result, project managers at these companies were producing estimates that were significantly different from actual project costs resulting in large bottom-line losses. We call this the "early-stage cost estimation problem".

Our solution to this problem draws inspiration from semi-automated tools to help programmers [4] in that we realized the solution will require a guided approach that integrates project managers into the cost estimation process from the very beginning in small yet important ways. In essence, our solution posits that managers should provide expert guidance to the automated metrics-driven, regression-based estimation techniques so that important contextual information (type of project, configurations of distributed project teams, nature of client, etc.) is taken into account in appropriate ways along with tapping into organizational memory and learning from past project experiences.

To develop this learning-oriented and semi-automated approach, we first identified the particular characteristics and quantitative metrics of globally distributed software projects that are important to early-stage cost estimation and available at the start of the project. Using these characteristics, we built a case repository that stratified incidents related to project effort estimation issues from the historical project databases at the firms into several generalizable categories. This case repository was then used to match newly starting projects with previous projects that had similar characteristics. We then prune the matched set and provide the project manager with the predicted cost estimates using the pruned set as a basis. The project manager was then allowed to easily browse the pruned set of projects to identify the portions of previous projects that were relevant to their new project. The managers were then able to change their project's cost estimation components, and immediately see the new cost estimates, to account for insights gained from prior projects. We allowed the managers to iterate as many times as possible as well as use any number of cases in the repository (if they so desired) to derive their final project cost estimates.

We deployed our solution at the three firms and invited project manager volunteers to use our estimation method. Overall, 219 newly starting, large globally distributed projects used our solution. The managers for each of those 219 projects used both our solution and the existing regression-based estimation method to compute the expected costs for their projects. We found that our approach performed significantly better, on average more than 60% reduction in estimation errors and 20% realized per-project cost savings, than the existing metrics driven regression oriented approaches. The 20% realized cost savings per project accounts for the additional overheads needed to implement and maintain the case repository and other artifacts needed by our solution. In addition, our solution can be used effectively by any project manager – regardless of his/her experience level. In all three deployments, we did not provide any form of extra training to the project managers. Managers learned how to use our solution using just the online help guides that came with our implementation. Thus, our solution significantly improved project performance at the three firms, and in this paper we report our experience in developing and implementing it.

Overall, this paper's three main contributions are: First, we develop and test a generalized set of quantitative criteria suitable for early-stage cost estimation of globally distributed software projects that overcome the limitations of existing metrics-driven regression estimation approaches. Second, we show how to allow project managers to augment quantitative cost estimations with qualitative judgments in distributed, high process maturity production environments. This approach helps improve organizational learning. Finally, we tested our solution in a large field test, across 3 firms, involving 219 live production projects over a period of six months. This field tests allows us to show tangible performance improvements due to our estimation technique.

II. In-Situ Observations

In this section we describe how we observed the three companies in-situ, enumerate our data collection procedure, and describe the deficiencies we discovered.

A. Business Case at the Three Research Sites

The research sites for this study are three well established, large, software service firms based in India that have adopted globally distributed software development and delivery for their customers. The software applications they custom-developed were predominantly business applications in the financial services, retail, manufacturing, and telecommunications industry. We had an established collaboration mechanism with these firms as a result of prior studies [1-3], and were aware of the challenges they faced in accurately estimating project effort.

The three firms were ISO 9001 certified and were independently assessed at CMMI-level 5. The firms also allowed extensive tailoring at the individual project level. We observed project personnel using a diverse set of software methodologies, including both plan-based methods and agile adaptations of the CMMI key process areas. The organizational-level Software Process Engineering Group (SEPG) at these firms maintained the central process engineering database and was in charge of the data collection from individual projects along with monitoring of the project processes through periodic audits. The SEPG teams also coordinated with external independent auditors such as the CMMI assessment teams.

The SEPG at the three firms employed statistical quality control and utilized their extensive project and process databases to set organization wide benchmarks for key project metrics such as productivity, conformance quality, in-process quality and overall profitability. The benchmarks were typically derived using regression techniques similar to the calibrated COCOMO II estimation procedures [5, 6]. The estimates and weights derived from these benchmarking exercises were then disseminated to project managers through centrally stored Microsoft Excel templates. Managers used these templates for project planning activities including estimating effort.

Despite the optimized and centrally controlled statistical procedures, the SEPG teams at the firms noted that the initial project effort estimates in distributed software projects were not within desirable bounds of accuracy. The mean estimation errors for distributed projects were more than 40% (minimum 11%; maximum 352%), and about 20% of projects reported more than a 100% deviation in original estimations. Early stage effort estimations are crucial for the profitability of the firms because fixed price contracting with clients (with penalty or bonus clauses) was the predominant contracting model in their business. The fixed price contracting scheme did not give room for recursive estimations and adjustments with their clients (i.e., no possibility for official re-estimations, without penalty from clients, as the projects proceed in their life cycle). Also a buffer of 40% on the average to cover initial estimation errors was not acceptable to their increasingly cost-conscious clients. Thus, there was a good business case to improve the estimation accuracy levels. Note that these cost estimation challenges are not unique to our research sites alone, but is a well acknowledged industry-wide issue.

B. In-situ Action Research-based Study

To observe the companies in-situ, we adopted a participatory action research approach for the first phase of our study. Participatory action research enables researchers to co-investigate or involve the communities whose practices they study in their research activities. These activities include data collection, record keeping, model building, solutions development, and implementation [7]. This research methodology allows researchers to obtain a deep understanding of the problem context, with a view to make improvements in the practices, even when they do not, a priori, have well defined solutions or theories [8-10]. In action research approach, researchers are not external “consultants” who offer advice, but strive to gain richer understanding of the social context by participating in the day-to- day activities of the community that is experiencing difficulties. The practitioners gain specific expertise to solve their problems from each other and from the researchers. Thus, adopting a participatory action research approach enabled us to develop solutions to the estimation problem at our research site, jointly with the employees of these firms, by considering the practical and day-to-day operational needs of the firms.

As a first step, we established the research protocols for the study which included details on participant selection, roles and responsibilities among participants, structuring learning opportunities, and implementing recommendations. We decided to form a four member core team consisting of one of the authors and three representatives from the SEPG at the research sites. The core team was responsible for the overall initiative and recruited additional participants at different stages of the study. Learning opportunities were structured using a three phase cycle consisting of diagnose-take action-evaluate stages. We also decided that at the end of the study we will present the findings to the top management of the three firms to pave the way for organizational wide implementation of our recommendations.

During the first three months of the study, the core team conducted 75 brainstorming sessions with the participants. Each brainstorming session with the project managers and software developers lasted about forty-five minutes. Each brainstorming session had a “diagnose’’ phase and a “recommend” phase. The “diagnose” phase of the brainstorming sessions were structured to understand the specific effort estimation practices of the project teams and discuss the difficulties that the project managers faced in estimating. In the “recommend” phase participants had to give ideas on possible improvement and critique ideas from others. One of the members of the core team took notes and on occasion recorded the session when remote participants were involved through teleconference. We also collected all the worksheets that the participants used during the brainstorming sessions.

C. Identifying Problem Root Causes

From the brainstorming sessions as well as our own observations, we noted a number of problems with the existing practices used at the three companies. We collated the observational, interview, and other quantitative data to identify the root causes for the poor project cost estimation. Together with the SEPG teams from each company, we converged on three causes that, if fixed, held the best chances of mitigating the early-stage estimation problem. The causes were:

1) *Missing Information:* Managers revealed that, while the existing estimation templates used extensive metrics and benchmarking numbers, they did not have real values for many input variables in the initial phases of their project. This missing information made them either completely ignore or provide subjective guesses for many variables. Moreover, we identified that the current estimation procedures did not take into account the intrinsic properties of distributed projects such as the configurational characteristics of distributed teams (our previous work [3] has details of these configurational characteristics).

2) *Simulation Capabilities:* In the existing estimation methods followed at the firms, benchmarks for key input variables are derived through regression analysis of aggregated data. Hence the underlying assumption is that all input variables have a joint, predetermined impact (as defined by the benchmark values) on effort outcomes leaving little room for context-specific manipulations. Even where project managers were allowed to “buffer up/down” the estimates, there was little guidance on how to usefully tweak the organization-wide benchmarks for specific project needs. The participants of our brainstorming meetings reinforced the need for individual project managers to control the weights for each input variable depending on their project management style (i.e., they wanted to alter the extent to which each input variable impacted project effort).

3) *Lack of Experience with Current Tools:* Finally, many of the junior managers reported that they were unable to effectively use the existing tools as they did not know which parameters could be safely modified and by how much. As a result, the estimates from junior managers tended to be of lower quality than those of more senior managers. Since the existing techniques used “best guess’’ values for many inputs, they were highly sensitive to the project manager’s experience level. This sensitivity was particularly troubling for the three firms as they were all expanding rapidly and thus had quite a number of project teams headed by relatively less experienced managers.

III. Solution: Learning-Oriented Cost Estimation

In this section, we describe our semi-automated learning-oriented approach that eliminates the above-mentioned root causes of the problem.

A. Learning-oriented Case-based Reasoning Approach

A common theme that arose from our in-situ observation phase was that managers felt that the existing estimation methods were quite rigid and required them to have all the necessary inputs on hand. From our previous experience with developing context-aware tools [4], we decided to tackle the problem using a semi-automated approach that allowed project managers to learn from past, quickly and efficiently overwrite the default values of the system with appropriate, context-dependent values. These context-dependent values can be systematically learned from relevant prior projects (i.e., organizational memory) or be supplied based on the planner's past experience. We felt that instead of building another fully automated system (similar to the existing methods being used) that would probably be fragile to non-standard project parameters, a semi-automated approach had the best probability of achieving an efficient system that was also a) easy to deploy, b) flexible to diverse situations, and c) easy to use for project managers and thereby aid quick adoption in the field. We realized that newer managers would not have the experience needed to override the default estimates where necessary. Hence, we designed a case-based reasoning approach to tap into existing organizational memory (past project experiences available across the firms) to help less experienced managers learn from other projects.

The general principle of the case-based reasoning approach is to apply specific information or knowledge from previous experiences to newer situations [11]. Humans implicitly use this technique for problem solving, and the expert-based estimation techniques rely on the ability of experts to relate new situations to the actions and outcomes of their past (prior cases). However, individual managers or team members are not usually able to capture the experiences of a whole organization (especially in large corporations), and hence additional tools are necessary to implement case-based reasoning for firm-level operations. There are several success stories on adoption of case-based reasoning tools in diverse fields [12]. We apply these ideas in the context of cost estimation for globally distributed software projects.

In our learning-oriented case-based reasoning approach, project managers, during planning, can compare their early-stage cost estimates with other similar projects without any additional overhead (i.e., there is no time consuming solicitation of similar project information). Moreover, our solution provides project managers with the ability to compare specific situations with past projects, even the failed ones (that were removed from previous benchmarking exercises). Further, project managers will have the ability to extensively tailor their estimates based on their experience and context-specific learning, and need not depend entirely on the centrally standardized (and hence less flexible) SEPG benchmarks. The ability to use qualitative inputs from other project managers (for example, through the documentation created during the project closure and reflection stages) also facilitates organizational learning and diffusion of knowledge and best practices throughout the firms.

Overall, the learning-oriented case-based approach should be able to overcome all the three root causes of the early-stage cost estimation problem identified in section II-C.

B. Building a Case Repository: Case Characterization

The first step in implementing a case-based reasoning approach is to represent past cases such that project managers can retrieve and compare them with their current project situations. We used our insights from the first phase action research study to identify six broad categories of variables that can be used to quantitatively characterize globally distributed projects:

1) *Distributed Work-specific Variables:* These included the extent of work dispersion in the projects, and work time overlap between distributed centers. These variables are unique to and necessary when estimating distributed software project costs.

2) *Client Control and Behavior-related Variables:* These included the client's prior experience with the project team, the extent to which client personnel would be involved in the project, details on whether the client was using any intermediary third-party consultants to deal with the projects, and the role of the client behavior in impacting requirements volatility.

3) *Project Team-related Variables:* This group of variables included details on the experience of the team members, the % of team members who were new to the firms and the technology used in the project.

4) *Project Methodology and Process-related Variables:* This group of variables included details on whether the projects used the SEPG standardized CMMI-level 5 methods or any adapted agile methodologies to implement the CMMI-level 5 Key Process Areas. In cases where non-standard agile processes where used, the extent of deviation from the standardized processes was included.

5) *Technology Variables:* These variables provided details on the key technology involved in the project. Since all the firms were involved in custom business applications, broad categorizations such as Mainframe-based, web-based (Java platform), web-based (.NET platform), and other applications were used for this characterization.

6) *Other Project Performance Metrics:* Five different project performance indicators were included to characterize past projects: productivity, defect density, reuse, rework and the percentage of project management effort expended on the project. Based on the prior benchmarking available with the SEPG teams, we also included labels such as "high performing project", "moderately performing project", "below average project" and "low performing project" for each of the performance metrics.

C. Storing and Maintaining the Case Repository

The case repository is stored centrally (separately in each firm) and maintained by the SEPG at each firm, similar to their existing software engineering process databases. The repositories were audited multiple times by the SEPG groups at three firms and external CMMI auditors for CMMI-level 5 conformance. All the three firms were found compliant with CMMI-level 5 processes by external auditors throughout our study. Hence, we are very confident about the reliability and validity of case repository data we utilize for this study.

The repository we used for this study contained information on 2520 completed projects in the manufacturing, retail, financial services, and telecommunications industry domains. Due to confidentiality reasons, participants were able to access projects only specific to their firms and project domains. Due to the non-disclosure agreements we have made with the firms, we are not able to make the case repositories public. At the minimum, all the project cases in our repository had information pertaining to the six categories of variables described in section III-B. In addition to these quantitative data, qualitative description of the project, reflections of project managers, and in some cases customer feedback descriptions were also present in the repository. The project closure and reflection meetings were utilized to add specific information to the cases referenced from the repositories. The following are some examples of qualitative data:

A project manager's description of a practice followed in his team: "We used excel sheet-based resource allocation for allocating test case preparations and assignment. A simple Perl script was written to translate the data from excel sheet to internal SEPG process database. Synthesis scripts were written in TCL for using the synthesis tools in batch mode"

A project manager's description of a challenge in her project: "The engineer was fairly junior and she did not have experience in Java. This was practically the first project that she did in Java. It required her to spend more hours per day as well as weekends, while the estimates were done as per 8 hrs a day and 5 days a week norm. Though we delivered project in time, it led to high overall effort overrun. For junior engineers, the estimates should have been based on a higher than 8 hrs work day norm."

A project manager's description of a customer related issue in her project: "Material procurement was an issue; one needs to plan in advance since the lead time from customer is really high. Requirement changes/additions were very often, and hence we needed to keep changing the schedule every time. Because of this we did not update the SEPG database every time, but used a batched "moving window" mode of planning. We logged our changes in an internal process Excel sheet and batch updated it to the SEPG database on a weekly basis"

A project manager's description of a project challenge: "Some of the senior mentors from the development team left the project. We did not have a plan for someone else to take over their mentorship role. This explains the low productivity during the last few months. A periodic review should be done to check the effectiveness of mentoring. Customer requests during this transition period required higher effort than usual. This special situation was communicated to the client through our relationship manager. This resulted in the client understanding that productivity would be low during this time when we are rebuilding our capabilities."

D. Using the Case Repository to Match Similar Cases

Effort estimation using case-based reasoning is done by finding "similar" cases from the case repository and using the actual effort expended in the past cases (projects) as the first estimate for a new situation. Two important design considerations in this scenario are a) how to find similar cases and b) how many similar cases to use.

In this study we use a weighted Euclidian distance metric to assess the similarity of a project based on the six categories of variables described in section III-B. While the approach is similar to past estimation studies [13-15], the categories of variables we use are new and specific to our research context. We use a z-dimensional Euclidean space (for z variables), and hence this accounts for the different measurement units of the z variables used to measure similarity. The formula we used for calculating the similarity score between different cases is as follows:

$$Similarity\ Score = \sqrt{W_a\,(A_0 - A_1)^2 + \cdots + \ W_z\,(Z_0 - Z_1)^2}$$

Where, A…Z are the factors picked from the categories mentioned in section III-B. The subscripts 0 and 1 on these factors indicate if the variable is for a project drawn from the repository (0) or if it is a newly starting project (1). $W_{a..,z}$ are the manager assigned weights for these factors. We used existing nearest neighbor matching and a 5-digit greedy-match algorithm to retrieve similar cases from the case repository [13-15].

We allowed individual project managers to decide on the exact number of similar cases to pick from the returned set of matching case results. We implemented a weighted average scheme for calculating the final effort estimate if a manager picks more than one case with same similarity scores, but perceived the cases differently. We analyze the sensitivity in the accuracy of estimations due to managers' choices in section IV-C.

E. Implementation

Our solution instantiation at the three research sites required two main steps: 1) defining specific variables as metrics to characterize the cases, 2) a process for each company to provide project managers access to use the case repository during early stage project planning. These steps are described in detail in the next sections, III-F and III-G.

F. Defining the Metrics Used

We defined 16 metrics to characterize the cases in our repository. Note that all these variables can be provided to a large degree of certainty by the project manager even at the start of the project. Variables 1 to 4 are specific to distributed environments and are necessary to capture the intrinsic nature of distributed software projects (please refer to our prior work [1-3, 16] for detailed descriptions of the properties of these variables and their association with globally distributed software project performance). For brevity, we only describe the variables and omit the detailed summary statistics observed during our field trials as the values were similar to those reported previously [1-3].

1) *Work Dispersion: W*ork dispersion is defined as a ratio of the percentage of work done at remote centers over the total project work.

2) *Time Zone Overlap:* The percentage of the official work day that overlaps with the official work day of the remote development center.

3) *Range of Parallel-sequential Work Handover:* In distributed development, it is essential to plan for work handover mechanisms so that deadlines and release dates are coordinated across the development sites. Project managers

were asked to estimate the extent to which concurrent or parallel work was planned as opposed to simple sequential work handovers. The extent of parallel work planned was measured in terms of percentage of total work.

4) *Team Culture:* Distributed software development sometimes involves multi-cultural teams without much overlap in native spoken languages, work styles, and official holidays. We used published cross-cultural difference measures to assess the differences in national-cultures across the geographically distributed teams [17]. We also used a short survey to assess the extent of common ground existing across the teams.

5) *Client-specific Knowledge:* Client specific knowledge was measured through a project manager survey before the start of the project (when he or she starts using the case-based reasoning method). The survey had the following six items measured on a 7-point scale (1 indicated no knowledge at all while 7 indicated complete knowledge) and were adapted from prior literature [18]. The average score of the six survey items listed below provided the score for the 'client specific knowledge' variable:

- How well do you know the project objectives of the client?
- How well do you know the client's business processes?
- How well do you know the business rules of the client?
- How well do you know the IT infrastructure of the client?
- How well do you know the IT norms and standards followed by the client?
- How well do you know the interoperability constraints of the client's IT infrastructure?

6) *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.

7) *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 from prior literature [18] 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 designs 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 new design.

To measure the technical newness, the technical lead was asked the following question: "For the design choice 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 two sub-scores to obtain the overall design and technology newness score

8) *Allocated Team Size:* Team size is the headcount of the number of persons allocated at project start.

9) *Process Model:* Managers choose whether their project would use the standard CMMI level 5 processes recommended by their organizational SEPG teams or whether they plan to deviate from the standard processes through project-specific tailoring and process customization. For example, many projects at the research sites used agile processes instead of the CMMI level 5 processes.

The set of 7 variables described below are derived from previous projects completed in each of the three firms, and are used to inform the project manager about likely project outcomes (i.e., expected outcomes or "benchmark") for her current project. We do this by matching the current project with previously completed projects and then returning the previous "best" values for these 7 variables. One of the key benefits of our on-site implementation is that project managers can modify the values of their input variables and observe immediately how the expected project outcomes change (i.e., perform simulations).

10) *Project Effort:* Project effort is measured as the total person-hours observed for a completed project. We also obtained estimated project effort, at the start of the project, from the project manager's project planning and estimation charts. The actual project effort was collected at the end of the project from the SEPG process database.

11) *Code Size:* This is measured as KLOC of delivered package to a client. In cases where function points were used instead, we used the Capers Jones method to convert the function points measure to KLOC.

12) *Development Productivity:* The ratio of software code size in KLOC to the total development effort in person-hours is used as the productivity measure.

13) *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. We also used the in-process defect density to assess the defects levels encountered by programmers during development.

14) *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 three firms. Reused modules and objects were easy to find and count as reused code was tagged with unique identification.

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

16) *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 filled by the project manager.

G. Process for Project Managers

We used the SEPG provided firm intranet project portals to allow project managers to access our solution. In all three deployments, we did not provide any form of extra training to the project managers. They learned how to use our solution using just the provided online help guides that came with our implementation (i.e., there were no human helpers to guide them).

Our solution allowed managers to input the early stage characteristics of newly starting project to process and select matches (prior cases) from the case repository. Managers could also allocate weights to the input variables (stating their relative importance to this specific project) that were different from the SEPG defaults. Managers were then able to filter the returned matched prior cases (which were ordered on several sortable categories), to select the most similar case(s) to their new project. The managers were also provided with estimates of project outcomes (variables 10 to 16 listed in section III-F) based on their case choices. Managers could remove any cases that they thought were irrelevant to their current project. Finally, managers could allocate weights to each remaining matched case (indicating how much each case actually matches their new project) to determine the final estimated effort using our proposed solution. In addition, the project managers would also estimate the effort using the established in-house estimation technique. We use the estimates from the established technique as a baseline comparison for our solution.

Over a period of six months, a total of 219 new distributed projects utilized our solution (together with the existing technique) to estimate their initial project cost. We analyze the effectiveness of our method, using these 219 usage scenarios in the next section.

IV. Analysis And Results

In this section, we evaluate the effectiveness of our solution. We provide three types of evaluation: (a) baseline comparison, where we compare the performance of our approach against the standard regression-based approach used at our research sites, (b) variation reduction, where we investigate if our approach allows any manager (regardless of their experience) to obtain good prediction results, and (c) sensitivity analysis, where we investigate the effect of specific input variables on the estimation accuracy.

A. Baseline Comparison

In this section, we compare the performance of our cost estimation technique with the standard metrics-driven regression-based approach used at our research sites. To do this, for each of the 219 projects that used our method, we obtained the original start-of-project effort estimation done by the project managers using both our technique and the traditional regression-based approach. In all cases, the project effort estimates were the real estimates derived by the respective project managers using both our on-site implementation and the established in-house estimation techniques. There were no cases where either the authors of this paper or the SEPG members helped to create an effort estimation using either our proposed solution or the in-house techniques – all estimations were done by the individual project managers themselves. As mentioned in section II-A, the original in-house estimation method was based on benchmarks derived from an internally calibrated COCOMO II estimation method (step-wise regression) that was optimized for scale economies. Finally, we obtained the actual effort spent by each project (we tracked all 219 projects until completion), and then calculated the Magnitude of Error in Estimation (MRE) for both the original in-house and our technique. The formula we used for MRE is:

$$MRE = \frac{|\text{ Actual Effort} - \text{Predicted Effort }|}{\text{Actual Effort}}$$

Table I. Results#

	Evaluation summary statistics across all projects				
	Original Method	Our Method	% Improve	Cost Savings	Better?
Mean MRE (Std.Dev)	47.51 (47.47)	15.99 (19.89)	66.35	20%	✓✓
Median MRE	35.65	11.67	67.28	14%	✓✓
Min MRE	11.18	0	100	Negligible	✓
Max MRE	351.20	127.98	63.56	150%	✓✓✓
	Count of Projects (% of total projects) with MRE scores.				
Greater than 25%	24.16	15.18	8.98	20 projects improved	✓✓
Less than or equals 10%	37.64	63.39	25.75	56 projects improved	✓✓✓

# Total number of projects =219. We used a two tailed t-test of means and confirmed that all the differences in MRE between the original and our approaches were significant at 5%. The cost savings value accounts for the additional overheads needed to implement and maintain the case repository and other artifacts needed by our solution.

Table 1 shows the results of our baseline comparison. Similar to prior studies, we show three MRE measures: the mean, median and the n-level comparisons (% of projects that have MREs greater than 25% and MREs lower than 10%) of the MRE [14, 15]. Our method has significantly lower estimation error (mean MRE % reduced by 66.35%) than the original method and 25.75% more projects, using our method, have MREs of less than 10%. In addition, due to the fixed price contractual clauses used by these firms, the improved accuracy in early stage effort estimation results in significant cost savings, by avoiding penalties or loss of bonus due to optimistic estimates, per project (more than 20% savings per-project on average). The 20% realized cost savings per project accounts for the additional overheads needed to implement and maintain the case repository and other artifacts needed by our solution.

B. Reducing Estimation Variation

In this section, we investigate if our approach reduces the impact of project manager experience on estimation accuracy. A common problem with many estimation approaches (especially the expert judgment-based approaches) is that they are more accurate if used by more experienced personnel. However, obtaining this experience takes time, and it is common in the industry for firms to deploy project managers with widely varying experience to manage their project portfolios.

From Figure 1, we observe that newer project managers perform just as well as experienced project managers when estimating effort using our methodology. However this is not true with the in-house original method, which exhibits large MRE variations depending on the project manager's experience.

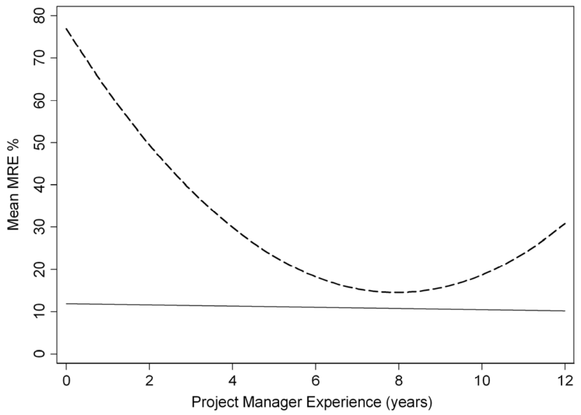

FIGURE 1. EFFECT OF EXPERIENCE ON MRE

To verify this statistically, we divided the project managers for the 219 projects into 4 groups depending on their experience level (calculated as the number of years as a project manager). The 4 groups were experience level 1-3 years; 3-6 years, 6-10 years, and >10 years. The groups had different manager counts with the 3-6 years group (the largest) having 35% of the managers, followed by the 6-10 group with 30%, the 1-3 years group with 20%, and the > 10 years group having just 15% of the managers.

We performed an unbalanced sample mean test (to account for the size differences between the buckets) comparison of the MREs for each of these groups for both our estimation technique and the original regression-based approach. While there was no statistical differences in MRE, across experience levels for our method, MREs from the original in-house method showed significant statistical differences across project manager experience levels. Thus, our method allows managers to obtain much better estimates regardless of their experience level compared with the original in-house method. This is an important benefit for the SEPG of firms engaged in statistical quality control trying to minimize unwarranted variance in project outcomes.

C. Sensitivity Analysis

In this section, we investigated how the MRE score changed due to the variation in the input variables (described in section III-F) and our sensitivity analysis graphs are presented in Figure 2. All trend lines are plotted using best fit polynomial regression (we consciously avoided individual data labels to prevent graph clutter and improve readability). It is important to note that the variations in MRE across these input values reflect the usage sensitivity of the technique implementation – i.e., how project manager's estimation errors were influenced by the different values of the input parameters. The most important reason for these usage sensitivity variations is the way a project manager utilizes prior case information. That is, in our approach two projects managers who are estimating effort for a given new project need not necessarily utilize the same set of cases from the case repository.

From Figure 2a, we see that managers could better predict their required effort (lower MRE) when they had a better understanding of the technology and design newness of a starting project. This matches our intuition that more information leads to better prior case selection and improved estimation results.

From Figure 2b, we observe that increased knowledge about the client does not lead to better estimates (higher MRE for increased client knowledge scales). We investigated this surprising result further and discovered that project managers who knew more about their clients tended to consistently overestimate the effort derived from the matching prior cases by including an additional buffer. Due to their additional client information, these managers could justify additional cost buffers to the SEPG and client peer reviewers. In addition, project managers who knew a lot about their clients did not lower their estimated effort even when they knew that their new project was not as challenging as other closest matches they found in the case repository. Figure 2c further substantiates the above result and shows that when project managers expected their clients to be heavily involved in the project, their estimation accuracy improved by as much as 15% on average.

Figure 2d shows the relationship between MRE and work dispersion in distributed projects. We find that the estimation error suffers, initially, as the work dispersion moved from completely co-located to about 10% of work being conducted in remote locations. However, as the amount of remote work increases from 10% onwards, project managers are able to greatly improve their estimation accuracy. One reason for this pattern could be the special organizational structures adopted at our research sites to manage work dispersion. As the extent of work increases at a remote distributed development center, the firms responded by appointing dedicated project coordinators to these centers. In our discussions, we learnt that diseconomies of scale prevented the firms from appointing such coordinators at lower work dispersion levels (i.e., the preferred 80-20 model provides incentives to minimize coordination costs).

Figure 2e reveals a quadratic pattern of relationship between the accuracy of effort estimation and project team size. On the whole, it indicates that project managers were more accurate in effort estimation for smaller teams (less than 10 in headcount) than bigger teams. Although the regression curve in Figure 4 shows improvements in accuracy for very large teams (headcount >30) we do not draw conclusive insights because of the lack of enough sample size in that category (less than 5 projects out of 219 in our sample had team sizes greater than 30).

Finally, we investigated the effect of the number of prior cases used for estimation on the MRE. On average, project managers at our sites used about 4.6 prior cases to estimate effort for a new starting project (minimum 1, maximum 10 with a standard deviation of 2.5). Figure 2f shows the impact of this variation on the MRE – using more cases reduces the MRE. Finally, our demographic analysis showed that new project managers (with less than 2 years managerial experience) who were either recently promoted within the firms or hired directly from outside the firms tended to use more number of cases than more experienced project managers. Also, female project managers, on average, used more prior cases to derive their effort estimates than male managers. There was no statistically significant variation in the number of prior cases used with respect to the other variables used in this study.

V. Discussion

A. Reasons for Improvement

First, by accounting for project-specific early stage indicators, our effort estimation approach leads to significant (statistically and economically) improvements in the accuracy of early-stage project effort estimates. In particular, our inclusion of distributed development specific variables achieves better estimation accuracy for globally distributed projects. Second, our categorization of prior cases in to six generalizable and easily measurable categories allows us to represent and match cases efficiently. This helps project managers to a) easily understand our process, and b), accurately measure and provide all variables needed for estimation. Finally, our implementation allows project managers to easily change effort estimation input parameters with good values provided by matched cases), and immediately receive feedback on the impact of those changes. These features improve learning and were cited by the project managers as a major benefit of our method. Overall, these reasons allow our solution to eliminate the limitations of the previously used "one-size-fits-all" estimation methods.

B. Adopting Our Solution

Our solution is general enough to be applied to other distributed software development situations as there is nothing solution-specific to the three test firms (beyond the raw data used in the case repository). In this section, we discuss specific organization implementation issues that might arise when adopting our solution. Our method, like all history-based systems, requires diligent historical data checks. When project teams rely on past incidents, there is a

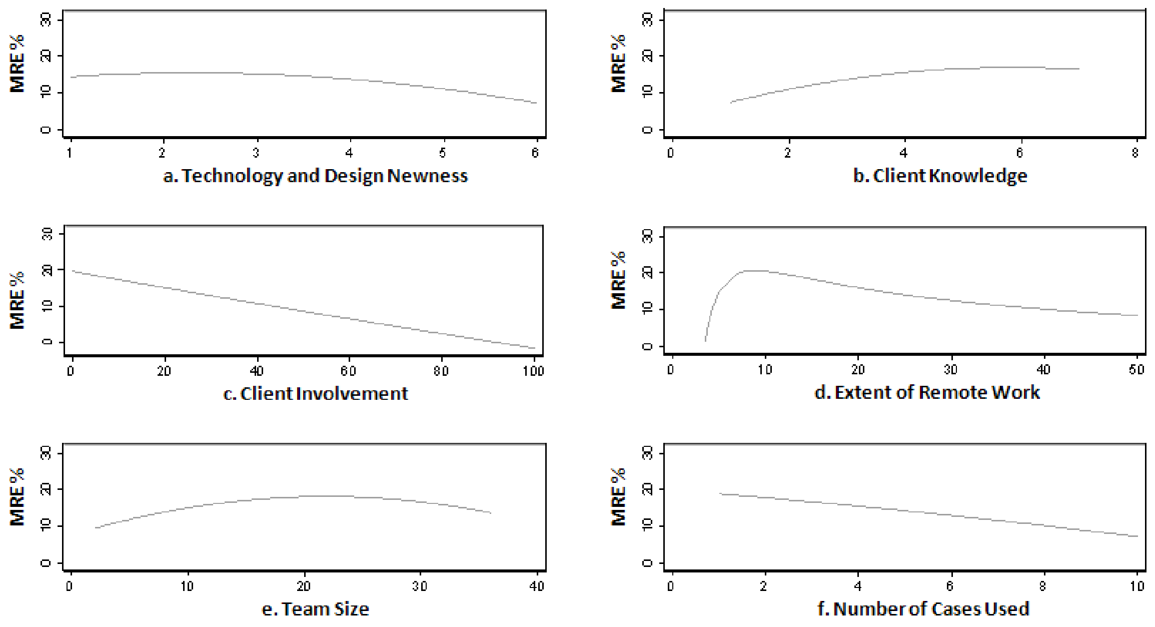

Figure 2. Sensitivity Analysis

high probability that both the good and bad past practices would be disseminated to newly starting projects. Lapses found in prior cases could potentially be used for providing excuses if performances are poor. It is thus challenging to minimize the spread of bad practices from prior cases when case-based methods are adopted across a firm. Strong and independent process control mechanisms are needed to stop the spread of sub-optimal practices from prior incidents to newer projects. One way to implement these controls is to get the SEPG groups to adopt a participatory action research approach in conjunction with the statistical outcome-based controls already being used.

While we have shown that our method yields great accuracy by accounting for context-specific criteria, we do not advocate abandoning other estimation techniques. Our results do show variation with respect to usage behavior and thus may require stringent process controls. Adding extensive process control mechanisms may not be scalable to all firms at all times. Moreover, unlike statistical methods, the large scale and across firms benchmarking potential of case-based reasoning methods, which require deep contextual information, is limited. Further, the metrics needed for successful implementation of our method requires high process maturity practices (deep process and project database, statistical categorization of performance, etc) that equally benefit other algorithm and regression-focused methods as well. Thus, we suggest SEPG at firms to facilitate the usage of multiple complementary methods.

C. Limitations and Future Work

We acknowledge several limitations of this study. First we studied only custom application software development projects in limited industry sectors (manufacturing, retail, financial services and telecommunications). We believe that the fundamental framework developed in this study can be applied to other types of software projects (product development, maintenance, reengineering) and programming environments (embedded programming, etc). However, this has to be verified empirically. Our method requires observation of managerial behavior that limits its testing possibilities compared to other estimation methods. We have also not studied the long term user behavior of the participants in our evaluation set. It is not clear if the number of cases or weights they use significantly changes over time. The sensitivity of our method to such longer-term user behavior is yet to be assessed. Addressing these limitations is one of our areas of future work. We plan to analyze the network of social and system linkages arising due to a project manager's case usage. Analyzing these linkages will help us understand the implications for distributed collaboration, process control and project performance.

VI. Related Work

The foundation for this study has been laid by the numerous software project estimation studies in both the software engineering and management literatures [5, 20-23]. Prior effort estimation studies using the case-based reasoning method [13-15, 24, 25] laid a good foundation for us to reuse their techniques and methods; albeit in a newer context. The difference is that our approach incorporates the intrinsic and unique characteristics of distributed software development into the process. We were also informed by studies in other fields [11, 12] using the case-based reasoning methods especially to design our research protocols in conjunction with archival data collection procedures. Our case categorization variables were influenced by prior work in software engineering economics and globally distributed work [26-28]. We integrated the findings of studies exploring collaboration among virtual teams [29, 30] in our methodology to address coordination issues specific to distributed software development. We also utilized methodological recommendations from other scholars [7] to design our participatory action research steps and protocols for the diagnose-take action-evaluate cycles of our first stage study. Finally, prior studies on software project and process controls [19] helped us assess the implementation effects of our case-based reasoning methods on existing software engineering practices at our research sites. Finally, our use of an interventionist action research observation method was influenced by prior research [9, 10].

VII. Conclusion

Despite significant advancements, accurately estimating software project effort, especially in the early stages remains a challenge. Commenting on the issue, Brooks aptly marks it as one of the three big challenges of computer science practice:

"Given specific functional, reliability, and performance specifications for a software system, we do not yet know how to estimate the effort required to build it. The challenge is to make software engineering as predictable a discipline as civil or electrical engineering. I still do not expect any radical breakthrough, any silver bullet, to solve this problem. But the accretion of many contributions has already made much progress, and I believe continued careful research, ever validated by real practice, will bring us to that goal" [31].

This paper advances this quest by using in-situ observational phase to first identify the root causes of cost estimation errors in the context of globally distributed software projects. It then proposes a semi-automated solution that uses case-based reasoning to allow project managers, regardless of their experience levels, to derive better estimates, especially by tapping organizational memory and learning from past projects. We field-tested our solution, extensively, at three different large global firms where it was used by the project managers of 219 new large globally distributed projects to estimate the cost of their new projects. Our evaluation shows that the learning-oriented approach to cost estimation significantly reduced estimation errors (of up to 60%) resulting in tangible economic benefits such as 20% cost savings, per project, on average. We are currently working on extending the method to address more diverse software environments and types of software projects and globally distributed team configurations.

REFERENCES

[1] N. Ramasubbu, M. Cataldo, R. K. Balan, and J. Herbsleb, "Configuring Global Software Teams: A Multi-Company Analysis of Project Productivity, Quality , and Profits," in International Conference on Software Engineering (ICSE), Waikiki, HI, USA, 2011.

[2] N. Ramasubbu and R. Balan, "Globally Distributed Software Development Project Performance: An Empirical Analysis," in ACM Sigsoft symposium on the foundations of software engineering ESEC-FSE '07, Dubrovnik, Croatia, 2007.

[3] N. Ramasubbu and R. Balan, "The Impact of Process Choice in High Maturity Environments: An Empirical Analysis," in International conference on software engineering, Vancouver, Canada, 2009.

[4] R. K. Balan, D. Gergle, M. Satyanarayanan, and J. Herbslen, "Simplifying Cyber Foraging for Mobile Devices," in Conference on Mobile Systems, Applications, and Services (MobiSys), San Juan, Puerto Rico, 2007.

[5] B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "Cost Models for Future Software Life Cycle Processes: Cocomo 2.0," Annals of software engineering, vol. 1, pp. 57-94, 2005.

[6] V. Nguyen, B. Steece, and B. Boehm, "A Constrained Regression Technique for Cocomo Calibration," in International symposium on empirical software engineering and measurement, Kaiserslautern, Germany, 2008.

[7] W. F. Whyte, Participatory Action Research: Sage Publications, 1991.

[8] R. Baskerville, "Investigating Information Systems with Action Research," Communications of the AIS, vol. 2, p. http://cais.aisnet.org/articles/default.asp?vol=2&art=19, 1999.

[9] D. E. Avison, F. Lau, M. D. Myers, and P. A. Nielson, "Action Research," Communications of the ACM, vol. 42, pp. 94-97, 1999.

[10] G. I. Susman and R. D. Evered, "An Assessment of the Scientific Merits of Action Research," Administrative Science Quarterly, vol. 23, pp. 582-603, 1978.

[11] J. Kolodner, Case-Based Reasoning. San Francisco, USA: Morgan Kaufmann, 1993.

[12] I. Watson, Applying Case-Based Reasoning: Techniques for Enterprise Systems. San Francisco, USA: Morgan Kauffman, 1997.

[13] L. Angelis and I. A. Stamelos, "A Similation Tool for Efficient Analogy Based Cost Estimation," Empirical software engineering, vol. 5, pp. 35-68, 2000.

[14] M. Shepperd and C. Schofield, "Estimating Software Project Effort Using Analogies," IEEE Transactions on software engineering, vol. 23, pp. 736-743, 1997.

[15] E. Mendes, N. Mosley, and S. Counsell, "Exploring Case-Based Reasoning for Web Hypermedia Project Cost Estimation," International Journal of Web Engineering and Technology, vol. 2, pp. 117-143, 2005.

[16] N. Ramasubbu, S. Mithas, M. S. Krishnan, and C. F. Kemerer, "Work Dispersion, Process-Based Learning, and Offshore Software Development Performance," MIS Quarterly, vol. 32, 2008.

[17] H. Geert, "Geert Hofstede Cultural Dimensions," http://www.geert-hofstede.com/hofstede_dimensions.php, accessed on 1-Sep-2009.

[18] A. Takeishi, "Knowledge Partitioning in the Interfirm Division of Labor: The Case of Automotive Product Development," Organization Science, vol. 13, pp. 321-338, 2002.

[19] 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.

[20] K. Molkken and M. Jorgensen, "A Review of Surveys on Software Effort Estimation," in International symposisum on empirical software engineering (ISESE), Rome, Italy, 2003.

[21] M. Jorgensen and M. Shepperd, "A Systematic Review of Software Development Cost Estimation Studies," IEEE Transactions on software engineering, vol. 33, pp. 33-53, 2007.

[22] B. Boehm, C. Abts, and S. Chulani, " Software Development Cost Estimation Approaches — a Survey," Annals of Software Engineering, vol. 10, pp. 177-205, 2000.

[23] L. C. Briand, K. E. Emmam, D. Surmann, I. Weiczorek, and K. D. Maxwell, "As Assessment and Comparioson of Common Software Cost Estimation Modeling Techniques," in International Conference on Software Engineering, Los Angeles, 1999.

[24] G. R. Finnie, G. E. Wittig, and J. M. Desharnais, "Estimating Software Development Effort with Case-Based Reasoning," in Second international conference on case-based resoning, Providence, USA, 1997.

[25] T. Mukhopadhyay, S. S. Vicinanza, and M. J. Prietula, "Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation," MIS Quarterly, vol. 16, pp. 155-171, 1992.

[26] J. D. Herbsleb and A. Mockus, "An Empirical Study of Speed and Communication in Globally Distributed Software Development," IEEE Transactions on software engineering, vol. 29, pp. 481-494, 2003.

[27] J. D. Herbsleb, D. J. Paulish, and M. Bass, "Global Software Development at Siemens: Experience from Nine Projects," in International Conference on Software Engineering, St. Louis, MO, USA, 2005, pp. 524-533.

[28] E. Carmel, Global Software Teams: Collaborating across Borders and Time Zones. Upper Saddle River, NJ: Prentice Hall, 1999.

[29] G. M. Olson and J. S. Olson, "Distance Matters," Human-computer interaction, vol. 15, pp. 139-178, 2000.

[30] C. D. Cramton, "The Mutual Knowledge Problem and Its Consequences for Dispersed Collaboration," Organization Science, vol. 12, pp. 346-371, 2001.

[31] F. P. Brooks, "Three Great Challenges for Half-Century-Old Computer Science," Journal of the ACM, vol. 50, pp. 25-26, 2003.