Journal of Enterprise Information Management

A study of software development project cost, schedule and quality by outsourcing to low cost destination

Debasisha Mishra Biswajit Mahanty

Received 26 August 2014
Revised 4 July 2015
5 September 2015
Accepted 18 September 2015

A study of software development project cost, schedule and quality by outsourcing to low cost destination

Debasisha Mishra
RG Indian Institute of Management, Shillong, India, and
Biswajit Mahanty
Department of Industrial and Systems Engineering, IIT Kharagpur, Kharagpur, India

Abstract

Purpose – The purpose of this paper is to find good values of onsite-offshore team strength; number of hours of communication between business users and onsite team and between onsite and offshore team so as to reduce project cost and improve schedule in a global software development (GSD) environment for software development project.

Design/methodology/approach – This study employs system dynamics simulation approach to study software project characteristics in both co-located and distributed development environments. The authors consulted 14 experts from Indian software outsourcing industry during our model construction and validation.

Findings – The study results show that there is a drop in overall team productivity in outsourcing environment by considering the offshore options. But the project cost can be reduced by employing the offshore team for coding and testing work only with minimal training for imparting business knowledge. The research results show that there is a potential to save project cost by being flexible in project schedule.

Research limitations/implications – The implication of the study is that the project management team should be careful not to keep high percentage of manpower at offshore location in distributed software environment. A large offshore team can increase project cost and schedule due to higher training overhead, lower productivity and higher error proneness. In GSD, the management effort should be to keep requirement analysis and design work at onsite location and involves the offshore team in coding and testing work.

Practical implications – The software project manager can use the model results to divide the software team between onsite and offshore location during various phases of software development in distributed environment.

Originality/value – The study is novel as there is little attempt at finding the team distribution between onsite and offshore location in GSD environment.

Keywords Outsourcing, Global software development, Team distribution

Paper type Research paper

1. Introduction

Software project management has been the focus of considerable attention during the past two decades in the information systems (IS), especially after the advent of outsourcing. Effective software project management focusses on the three P's: People, Problem and Processes (Pressman, 1997). The human resource is the most important factor in the development of a software product. In spite of development in technology and processes, the productivity of human resources remains the most dominant factor for cost and schedule of software development projects, which are also the most

Journal of Enterprise Information Management
Vol. 29 No. 3, 2016
pp. 454-478

1741-0398
DOI 10.1108/JEIM-08-2014-0080

important reasons for outsourcing (Alpar and Saharia, 1995; Loh, 1994). Based on the standards and tradition in the software development field, the most common combination of criteria used to measure the success of a project concerns meeting time, cost, functionality and quality goals (e.g. Anda *et al.*, 2009; El Emam and Koru, 2008; Kappelman *et al.*, 2006; Lai, 1997). This paper presents a study that explores the effect of division of work between onsite and offshore location and training effort at onsite and offshore locations on the development cost and schedule of a development project in global software development (GSD) environment.

Outsourcing has become one of the strategies adopted by global organizations to manage the IS. The IT outsourcing phenomenon has been widely discusses in the literature (Aktas and Ulengin, 2005; Gowan and Mathieu, 2005; McBride *et al.*, 2007; Ngwenyama and Sullivan, 2007; Bairi and Manohar, 2011). Among the location countries for outsourcing, India accounts for a dominant share of the global IT offshoring market (Peterson and Gott, 2011; Raman and Chadee, 2011). Numerous business houses in the US outsource a number of their non-critical processes to overseas countries to reduce costs incurred in salaries and running operations (Rathi and Joshi, 2010). Most of the outsourced software projects in the context of India are in the form of business application development and business application maintenance categories, outsourced mainly from USA and Europe. As the software development process has gone global, team structure and division of work between onsite and offshore locations has become important in the context of GSD, as they greatly influence onsite and offshore team productivity, which, in turn, affect the project cost and schedule.

Project-based work is especially popular in the information technology domain (Desouza and Evaristo, 2006). The maintenance and re-engineering projects constitute higher volume of work in outsourcing environment. The software development projects in business domain constitute a very small percentage of total work in Indian outsourcing industry. The communication overhead is higher in development projects (Carmel and Agarwal, 2001) due to need of higher knowledge for execution. This can adversely affect the cost and schedule of a software development project. Normally the knowledge transfer in GSD takes place from a business user to a software developer (Kobitzsch *et al.*, 2001). The development team needs to absorb the knowledge from the client organization to be effective in outsourcing environment (Cohen and Levinthal, 1990; Dibbern *et al.*, 2008; Lee, 2001). The communication effort required for the knowledge transfer at both onsite and offshore location decides the team productivity affecting cost and schedule. This paper makes an attempt to find good values of onsite-offshore team strength; number of hours of communication between business users and onsite team and between onsite and offshore team so as to reduce project cost and improve schedule in a GSD environment for software development project.

The rest of the paper is organized as follows. In Section 2, we have discussed the relevant literature. The research methodology is explained in Section 3. Section 4 has elaborated the system dynamics model used for our study. The model validation is explained in Section 5. The research findings are provided in Section 6. The discussion on research findings is discussed in Section 7 and summary of research findings are given in Section 8. This paper is concluded in Section 9.

2. Relevant literature

Distributed software development (DSD) is nowadays a common practice within the software industry (Hernández-López *et al.,* 2010). GSD is a particular type of DSD in which teams are distributed beyond the limits of a nation (Herbsleb and Moitra 2001).

The literature on software project management discusses about issues such as cost, schedule, knowledge transfer, productivity and quality in the outsourcing environment. Project management challenges are multiplied due to issues related to knowledge transfer, communication, project management, quality management and infrastructure (Kobitzsch *et al.*, 2001). The outsourcing life cycle management can be critical in GSD (Beulen *et al.*, 2011). Effective knowledge sharing is considered essential for high performance in both co-located and distributed settings. The knowledge transfer culture and alignment between IS and business strategy can influence knowledge transfer to offshore location (Mayasandra *et al.*, 2011; Chua and Pan, 2008). The time zone difference between various countries also adds to project management inconveniences (Lacity and Rottman, 2009). The GSD is challenging due to prevalence of co-ordination and communication issues (e.g. Avritzer *et al.*, 2010; Casey and Richardson, 2009; Conchuir *et al.*, 2009; Cusumano, 2008; García-Crespo *et al.*, 2010). There could be loss of communication richness (physical distance, time zone, and domain expertise), co-ordination breakdown (software architecture, integration and configuration management), geographical dispersion (vendor support, governmental issue) and loss of team-ness (feeling of belonging in the team) in GSD (Battin *et al.*, 2001). The outsourcing life cycle management can be critical in GSD (Cullen *et al.*, 2006; Webster and Watson, 2002).

The GSD offer a lot of advantages like lower cost (Ramasubbu *et al.*, 2005; Smite *et al.*, 2010), shorter time to market (Jalote and Jain, 2006; Kommeren and Parviainen, 2007; Sooraj and Mohapatra, 2008) and greater availability of manpower (Conchuir *et al.*, 2009; Kommeren and Parviainen, 2007). The business organization can be benefitted by GSD, if the development team productivity is higher. The productivity drops in the outsourcing environment due to dispersion of work force (Ramasubbu and Balan, 2007). Similarly, according to Kommeren and Parviainen (2007), the productivity of globally distributed team members decreases by up to 50 percent compared to that of co-located team members. Moreover, the delivery of software products developed in globally distributed environments takes two and a half times as long as in a co-located environment (Herbsleb and Mockus, 2003). However, we not aware of any effort to estimate the project cost and schedule in GSD environment.

3. Research methodology

One of the important purpose of our research is to calculate software team productivity in GSD environment. System dynamics is used to capture the changes in project performance variables with respect to execution time. The models for productivity estimation in literature can be broadly divided into static and dynamic models. The static models can be roughly categorized as either analytical relying on mathematical formulas or experience-based. COCOMO II and SLIM are the most conventionally used mathematical static models (Boehm, 1981; Kemere, 1987). The International Software Benchmarking Standards Group database can help in making experienced-based estimates for analysis, bench-marking and comparison of various kinds of software projects (ISBSG, 2009).

Among dynamics models, the system dynamics tool is used by various researchers. A notable system dynamics model used in software project management is by Abdel-Hamid and Madnick (1991). System dynamics tool is used by various researchers such as Ruiz *et al.* (2001) and Setamanit *et al.* (2007) to study software project projects characteristics in co-located and distributed environment, respectively. Dynamic models are more suitable for estimation of project parameters as research suggests that estimation models reflects the changing scenario in system development life cycle as “it is more realistic to think of software engineering as an evolutionary process where software is continually changed over its lifetime in response to changing requirements and customer needs” (Sommerville, 2004). It is also necessary to refine the model throughout the life cycle of the project based on changing parameter (Hamid and Madnick, 1991). So we decided to use system dynamics to simulate the software project development environment in GSD.

The model was constructed based on knowledge management framework of software development from literature proposed by Mishra and Mahanty (2015). The model describes the various kinds of business knowledge required in different phases of software development. Mishra and Mahanty (2014) has already used the business knowledge management framework in construction of system dynamics model for simulation model of software re-engineering project. The model was used by Mishra and Mahanty (2014) to calculate cost and schedule of re-engineering project in GSD environment. The knowledge management framework is used for construction of system dynamics model for the development project in this research work. We interviewed 14 experience software professionals from Indian software industry during the model construction, validation and testing. All the interviewed experts had more than ten years of experience in the outsourcing industry. They work as project leaders, project managers, senior managers and group leaders constituted a representative sample of the Indian software outsourcing industry. The system dynamics model construction is explained in Section 4 and model validation and testing is explained in Section 5 of this document.

4. System dynamics model sectoral overview

The system dynamics model developed consists of four major sectors: policy decisions, knowledge transfer, team productivity and error proneness, and software development sector with various flows connecting them. Figure 1 provides the overview and interconnection of model’s four sectors.

The purpose of the model is to study important project characteristics as provided by Putnam and Myers (1996):

- a measure of the average productivity;
- a measure of the production cost;
- a measure of the quantity produced;
- an indication of the product quality; and
- a measure of the remaining time to complete the project.

The simulation model estimates the average nominal productivity based on business knowledge expertise level of the software developer. The production cost is dependent on manpower usage at onsite and offshore locations. The project quality can be measured by amount of rework needed *vis-à-vis* software development work done. We used extent of rework (EOR) and extent of latent error (EOL) to measure rework quantity and product quality, respectively as used by Rai and Mahanty (2002). The EOR and EOL are defined in Equations (1) and (2), respectively. The remaining time to complete the project can be calculated by tasks remaining and team productivity after communication overhead:

$$\text{EOR} = 100 \times (\text{Revised Total Tasks} - \text{Initial Total Task})/(\text{Initial Total Task}) \quad (1)$$

$$\text{EOR} = 100 \times \left(\text{Tasks with Fault}/\text{Total Tasks}\right) \quad (2)$$

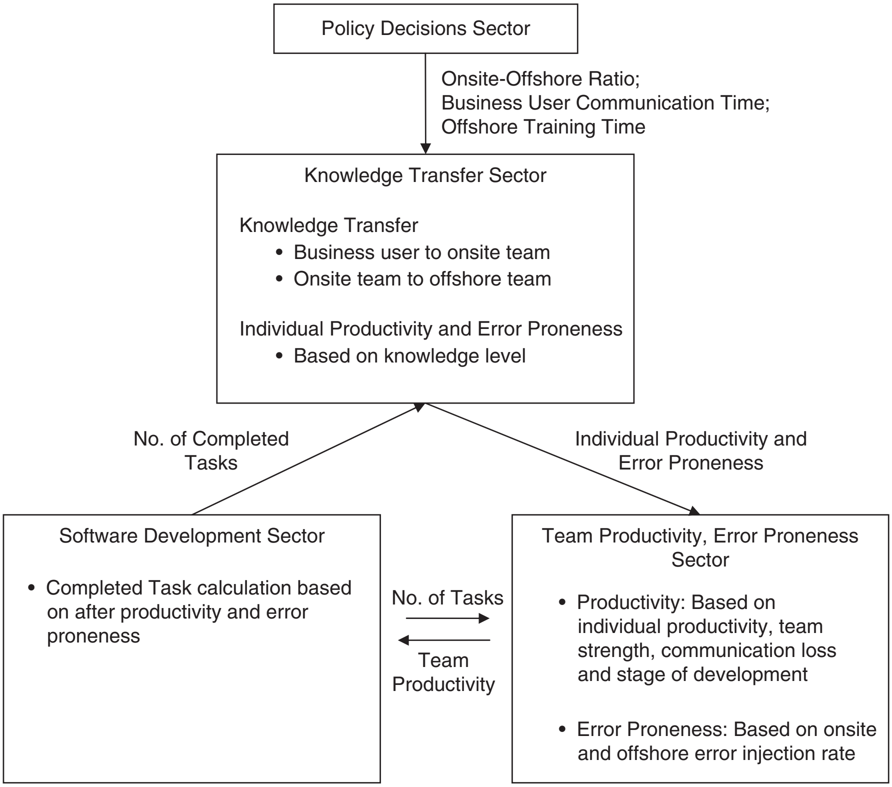

Figure 1.
Sectoral overview diagram

4.1 Knowledge transfer sector

This sector is based on business knowledge management framework of software development proposed by Mishra and Mahanty (2015) based on literature review and expert opinion survey. Rus and Lindvall (2002) identified two types of knowledge central to the software development process: first, technical knowledge that is used to develop a system; and second, business knowledge in the application domain of a system. Integration of technical knowledge with business application domain knowledge is central to effective software development (Tiwana, 2004a, b). Business knowledge is related to the process novelty for application software development. Technology knowledge is related to the technology aspects of application software development. The business knowledge assumes greater importance in distributed environment as technology can be learnt independent of location. The business knowledge requirements, however, is highly location-specific and has to be met from the client organization at a given location. We have assumed waterfall model of software development for our model construction. Although there are many methods of software development, the waterfall model is usually preferred in outsourcing environment, because it is easier to execute at offshore locations for design, coding and testing activity when the product requirement and user interface does not undergoes frequent changes (Sakthivel, 2005).

The business knowledge can be divided into four different types such as domain, strategic, business process, and operation process knowledge. This classification of business knowledge was taken from organization model by Guetat and Dakhli (2010), which defines an organization as conjunction of four spaces: the strategy space, the knowledge space, the information space, and the operational space. The software projects belong to the information space. The information space interacts with all other spaces and gathers knowledge from them. The model was derived from three world models defined by Popper (1972) and subsequent work by Stohr and Konsynski (1992). The importance of various kinds of business knowledge in various stages of software development and nodes of knowledge flow in GSD in explained in Figure 2.

The domain knowledge is required in feasibility study of software development. The role of software development team is minimal in this stage. The requirements analysis phase is dependent on strategic knowledge. Similarly, the business process and operation process knowledge is required for design software architecture, which consists of high-level design (HLD) and low-level design. The business process and operation process knowledge can be taken as part of overall process knowledge. The coding and testing phases requires basic domain, strategic and business process knowledge for a productive team member. The productivity of software development team depends on strategic, business process, at requirements analysis and software architecture phase, respectively. The domain knowledge is necessary to understand other kinds of business knowledge.

Various kinds of business knowledge have to be transferred from business users to onsite team and from onsite team to offshore team. We have used the knowledge flow concepts as used by Zhuge (2002) and Zhuge *et al.* (2007). The development projects can be considered as a closed environment where the knowledge flows from a higher node to a lower node by communication effort. The business user, onsite development team and offshore development team works as three nodes in our knowledge transfer model. The productivity of the software development team is dependent on

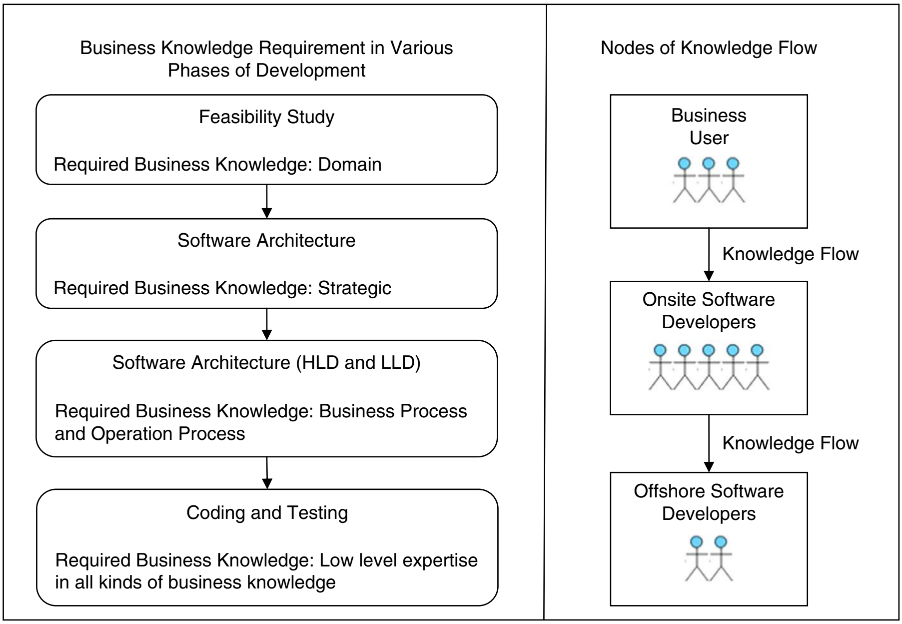

Figure 2. Knowledge requirements in different phases of software development and nodes of knowledge flow

accumulated knowledge. The knowledge flows from business users to onsite team and from onsite team to offshore team as given in Equations (3) and (4), respectively. The two proportionality constants are Knowledge_Tranfer_Coefficient_from_Client and Offshore_Knowledge_Transfer_Coefficient, respectively and we assumed value of 0.2 and 0.06 for them. The values were assumed after consultation with the industry experts, who suggested that the knowledge transfer to offshore location is slowed down by more than three times in comparison to onsite-only knowledge transfers. The business knowledge flow in the development team is given in Figure 3.

The knowledge transfer co-efficient for a particular stage is related to previous business knowledge expertise. For example, one can understand strategic knowledge in a better way, if one has some knowledge of domain knowledge. It is not possible to learn strategic knowledge without having any knowledge of business domain. The dependency of knowledge transfer on preceding stage business knowledge takes the sigmoidal "S" shape. This means that current business knowledge transfer is severely affected with lower level business knowledge in the preceding stage:

$$\begin{aligned} &\text{Knowledge\_Trasfer\_Rate\_From\_Client} \\ &= \text{Knowledge\_Tranfer\_Coefficient\_from\_Client} \\ &\times \text{No\_of\_Hour\_of\_Communication\_With\_Business\_User} \\ &\times (\text{Knowledge\_Level\_of\_Business\_User} \\ &- \text{Average\_Business\_Knowledge\_Onsite}) \\ &\times \text{Average\_Business\_Knowledge\_Onsite}/\text{Knowledge\_Level\_of\_Business\_User} \end{aligned} \tag{3}$$

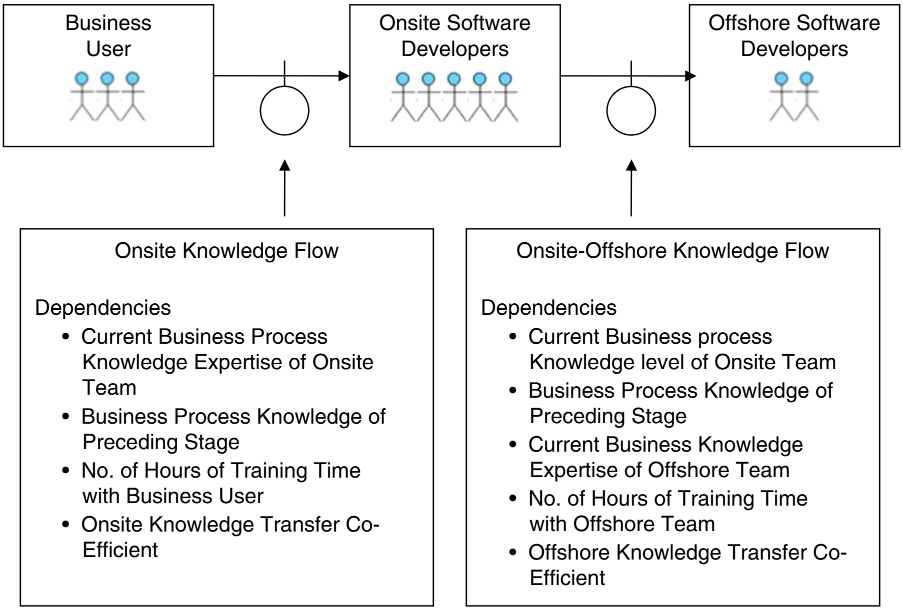

Figure 3.
Business knowledge flow in the software development team in GSD

$$\begin{aligned}&\text{Knowledge\_Transfer\_From\_Onsite\_Team} =\\&\text{Average\_Time\_Spend\_on\_Training\_Offshore}\\&\times\text{Offshore\_Knowledge\_Transfer\_Coefficient}\\&\times(\text{Average\_Business\_Knowledge\_Onsite}\\&-\text{Average\_Business\_Knowledge\_Offshore})\\&\times\text{Average\_Business\_Knowledge\_Offshore}/\text{Average\_Business\_Knowledge\_Onsite}\end{aligned} \quad (4)$$

The knowledge is defined using a five-point scale as illustrated below:

1 – Little knowledge, 2 – Below average knowledge,
3 – Average knowledge, 4 – Better than average knowledge,
5 – Highly developed Knowledge.

4.2 Software development sector

The software development sub-sector calculates the task completion in the development process. We have combined development, system testing and quality assurance work into a unified term called task. The project is considered to consist of a number of tasks. A task can be defined is productivity of an experienced software engineer in a single day. The tasks completed can be done correctly or can be done with fault. The work done with fault is dependent on nominal error injection rate. Also we assume that no of errors are multiplied when the current business knowledge is below the required business knowledge. The errors are detected by quality assurance activities after introduction of delay in the process. The effort to rework on errors multiplied depending on the stage they are discovered (Rai and Mahanty, 2002). We have assumed a multiplication factor of 3, 2 and 1 for errors discovered in requirement analysis, software architecture and coding/testing phases, respectively. The erroneous tasks are added to the remaining tasks. The sector is given in Figure 4.

4.3 Team productivity and error injection sub-sector

The team productivity and error proneness is calculated in the team productivity sub sector. Software project data for effort prediction purposes have been widely studied and many models have been proposed, such as COCOMO81 (Boehm, 1981), COCOMO II and COCOT (Boehm *et al.*, 2000) and RUPS (Kruchten, 1999). The effort values can vary widely in real life environment as found by Kultur *et al.* (2009) and Tan *et al.* (2011). We referred to COCOMO II model for our simulation. As per COCOMO II model, the planning and requirement phase (related to strategic knowledge), software architecture (related to business process and operation process) and coding/testing takes 7, 32 and 58 percent of the effort, respectively. For our simulation we will be assuming the values 10, 30 and 60 percent, respectively for requirements analysis, software architecture, and coding and testing phase, respectively.

The team productivity depends on productivity of the onsite and offshore team members. The productivity of the offshore and onsite team members in a development phase depends on the type business knowledge level required for tasks execution as shown in Figure 5. The team member can work on software development tasks after spending their time on training effort. There can be loss of productivity due to

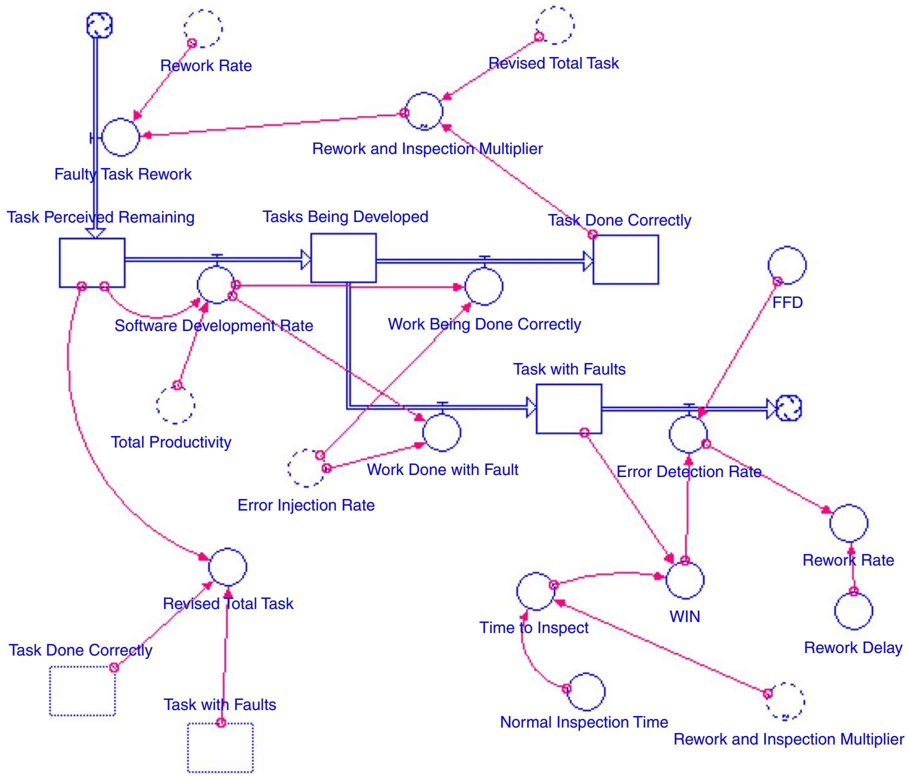

Figure 4. Software development sector

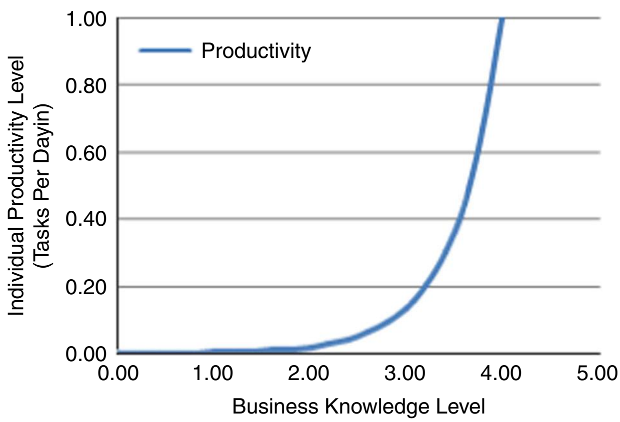

Figure 5. Business knowledge level and individual productivity plot

intra-team communication. The calculation of nominal team productivity and actual team productivity is calculated for a particular phase. The errors are injected into the production system in development process. The offshore team is two times more likely to make error compared to onsite team. The assumption was made after consulting the experts from Indian outsourcing software industry.

5. Model validation

The simulation model development can be carried out in three stages: first, defining the system problem entity; second, building the conceptual model; and third, converting the conceptual model to a computer executable model (Sargent, 1981). All these three stages of simulation model building require either subjective or objective validation. The conceptual model building phases needs verification on theory framework and assumptions. The computer executable model requires that the conceptual model is implemented without errors and, also operational data are able to produce the system behavior for the intended system under study (Sargent, 2010) The operational validation is the most important validation process where the output data are verified to ensure that all aspects of model such as input data, model framework and computer simulation are reliable. The verification and validation for a system dynamics model cannot be 100 percent reliable as no single model can represent the entire reality, since such model will be expensive to construct and more difficult to manipulate (Pidd, 2000). The purpose of our research is to find out the cost, schedule and quality of development project in GSD, the verification and validation install enough confidence in us for policy analysis.

5.1 Conceptual model validation

The face validation and traces can be used for evaluating the conceptual model of the simulation. Our model was constructed after interviewing the 14 experts from Indian outsourcing software industry. The experts guided us all though out the model construction process. It ensured that we conducted face validity for our mode. The experts reviewed the logic for every variable in the model, which is a process to conduct trace validity. The experts knowledge-based framework for software development is constructed based on expert interview is explained in Section 4.1. The entire process of expert interview and data collection will be communicated as part of a separate research paper.

5.2 Computer model structure verification

The computer structure verification and validation is required to ensure that appropriate software tools and techniques are used for model constructed. The system dynamics technique is very popular for modeling software development after publication of work by Abdel-Hamid and Madnick (1991). The Stella software is widely used tool for constructing and simulating the system dynamics models. The structural validity is a stringent measurement for system structure in order to build confidence in a system dynamics model. A right behavior can also be "biased" behavior depending upon the extent of agreement of the actual system behavior values with that of simulated behavior values. Thus a behavior validity test should follow the structural validation procedure. The structural validity of a system dynamics models includes boundary adequacy test, structure verification test, extreme condition test, dimension consistency test, and parameter verification test (Qudrat-Ullah and Seong, 2010). The purpose and method of each validation test is explained in Table I.

5.3 Operational aspects validation

The main focus of verification and validation testing is dependent on validation of operational aspects of the model. The system behavioral pattern can be found out by operational output data provided by the system in real life environment.

Table I.
Validation process for system dynamics model

System dynamics model validation tests	Purpose	Method of ensuring validity
Boundary adequacy test	To check whether the important concepts and structures for addressing the research issues are endogenous to the model	Face and trace validity
Structure verification test	To find whether the model structure is consistent with the relevant descriptive knowledge of the system being modeled	Face and trace validity
Dimensional consistency test	To check for each mathematical equation in the model so that the measurement units of all the variables and constants involved in the model are dimensionally consistent	Manual verification
Parameter verification test	To find whether the parameters in the model are consistent with the relevant descriptive and numerical knowledge of the system being modeled	Face validation and parameter values available from literature
Extreme condition test	To assess whether the model exhibits a logical behavior when extreme values are assigned for selected parameters	Model running

We approached different companies located at the cities of Bangalore and Bhubaneswar in India to study their development projects for our research work. We received two responses from one company each at Bangalore and Bhubaneswar. We selected the company at Bhubaneswar for our research work as the company provided us better access to their repository of project information. We selected five projects for our model validation from the repository of development projects inside the organization. The five projects were selected based on availability of detailed project data and accessibility to project team members. The task completion plot with respect to time for all the projects provided satisfactory results for establishing reliability and validity of the system dynamics model. The details of validation process for an individual project are explained in this section.

The short duration of the selected project allowed us to find out the task development on daily basis by referring to project documentation. The parameter adjustment for the model was carried out in consultation with the project manager. The selected project was a payroll processing system developed in java and oracle relational database. The size of the project was estimated at 400 tasks by the project manager. It was based on previous experience and function point estimation table used by the company. The project was executed by a total of 12 software professionals. The onsite and offshore team strength was ten and two, respectively.

The effort distribution between planning, design and implementation phase was estimated at 10, 45 and 45 percent instead of 10, 30 and 60 percent assumed in the model presented earlier. The higher effort in design phase was necessary due to an object orient approach to development, which requires additional effort in comparison to structured programming. The knowledge transfer co-efficient were estimated after finding out the training time at various phases of software development. We could calculate the productivity of software team by analyzing the project progress. The result of our model run is given in Figure 6. The close match of the model generated results for number of completed tasks with the actual data gave us the confidence to run various policies as discussed in Sections 6 and 7.

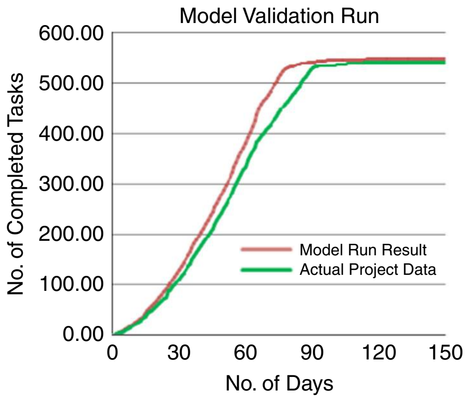

Figure 6.
Comparison of model run and project data

6. Policy run of the model

6.1 Model base run

We ran the model initially for a co-located (onsite only) project with business complexity 4, total number of tasks 1,000 and total number of software developers 12. The software development productivity depends on the business knowledge of the software development team. The business knowledge flows from business users to onsite team. So the amount of time spent on communicating with the business user is a very critical decision in any software development team. We tried to find the amount of training necessary at the onsite location by running our model.

We assumed that "Total No of Working Hours in a Day" is eight. The simulation was executed for low, medium and high values (one, two andthree hours, respectively) of communication time with business user in a day till onsite developer's business knowledge becomes equal or more than that of required business knowledge for project execution.

Our model run shows that increasing the communication time with business users help in gaining business knowledge at onsite location faster, but it does not necessarily increase the software development rate. A very low business user communication time like one hour per day can affect the project schedule adversely. But, to our surprise, we found that spending more time with business users does not leads to lower project cost and schedule. The schedule decrease from 134 days to 122 days, when business user communication time increased from medium to high value. But further increase in communication time did not result in improvement of schedule time. The simulation results did not change with projects of different business complexity.

In our model, we assumed that business users can spend unlimited time to train the software developers. In reality, the number of business users allocated for software project may be quite low in comparison to software developers. So it is not practically possible to allocate a lot of time for an individual developer to train them. Also we assumed a constant value of Knowledge_transfer_coefficient_from_client at onsite location irrespective of number of hours of communication time. This may not be true in real life scenario. The interviewed experts suggested that it is practically not possible to spend more than two hours of communication time with business users every day because of capacity of human mind to absorb business knowledge and assimilate it. The value of knowledge transfer co-efficient goes down drastically with increase in training time and it can affect the loss of team productivity. Considering the above, we took a medium value of business user communication at two hours per day for our subsequent model runs.

6.2 Model run in GSD environment

The base run of the model was conducted in co-located environment as explained in Section 6.1. We then tried to find the project cost and schedule in GSD environment. The project parameters were kept at similar level as that of base run. The project parameter and their assumed values are: project complexity – 4, total number of tasks – 1,000, total strength of software development team – 12, nominal business user communication time – two hours/day. For our model run, we will be assuming that cost of an onsite resource is four times that of an offshore resource. We will use value of onsite and offshore resources at 4X/day and X/day, respectively. The project schedule and cost for the base run co-located projects are 134 days and 6432X, respectively. We varied the onsite ratio, offshore training time and delay time for late introduction of offshore team for coding/testing work in GSD environment. The reasons for selection these variables are explained below.

Onsite ratio. Cost saving is the prime reason for popularity of outsourcing. The offshore resources are around four times cheaper than onsite resources. So there is a potential cost saving by employing a large team at offshore location. But it has the potential to affect the schedule adversely, nullifying the cost advantage. We simulated the model with onsite/offshore ratio at 0.75, 0.5 and 0.25. The model was not simulated for complete offshore execution as it is not possible in real life as suggested by our interviewed industry experts.

Offshore training time. The offshore team training is required for project execution in the distributed environment. "How much time the onsite team should spend to train offshore team?" is a very critical decision in any kind of distributed development projects. The onsite team productivity decreases when too much time is consumed in training offshore. Too little time spent on training the offshore team also reduces overall productivity of the team. This is because the productivity of the offshore team is significantly lower than that of the onsite team, and there is no proper utilization of the offshore manpower. So the project management team needs to do a balancing act between training offshore team and project productivity. As per the interviewed experts, it is not possible to spend more than three hours a day by onsite team members to train the offshore team. So we will be experimenting with one, two and three hours of average training time by onsite team.

Late introduction of offshore team. The coding and testing phases of software development can be executed with lower business knowledge. So the offshore team can be utilized for coding/testing work with minimal business training. It is a good use of offshore team for saving cost. A bigger team can be employed at offshore location to reduce the project schedule. The offshore team needs to be given basic training in domain, strategic and business process knowledge to be effective in coding work. We kept the nominal limit for domain, strategic and business process knowledge at value 2 for our model.

7. Results of model run

The model run results for various combinations of onsite/offshore resources ratio and offshore training time are discussed in this section. We have discussed two scenarios for onsite/offshore resources ratio of 0.75/0.25 and 0.5/0.5. The model simulation with onsite/offshore resources ratio of 0.25/0.75 affects the project cost and schedule in a very adverse manner. So it is not advisable to execute the development project with higher percentage of resources at offshore location and is ignored in the discussion.

Initially we ran the model with onsite/offshore ratio of 0.75/0.25. The model was simulated for low and medium values (one and two hours/day, respectively) of average offshore training time by the onsite team. It is not possible to extend the training time to more than medium values as offshore strength is small to utilize the training time. Similarly, the model simulation for onsite/offshore resources ratio of 0.5/0.5 was simulated for medium and high values (two and three hours/day, respectively) of average offshore training time by the onsite team. A small value of offshore training (one hour/day) for onsite/offshore ratio of 0.5/0.5 is not an advised scenario as it has high negative effort for project cost and schedule.

7.1 Project cost and schedule

Figure 7 shows the plot of software tasks completion for various scenarios of onsite/offshore resources ratio and offshore communication time. The onsite training time is two hours/day for all scenarios. Also the total team strength is 12 in all the situations.

It is evident from Figure 7 that the project schedule is adversely affected by maintaining higher percentages of resources at the offshore location. The project under our consideration is completed in 134 days for onsite-only execution scenario. The project size becomes 1,078 tasks in comparison to initial size of 1,000 tasks due to addition of new tasks as result of error injection by onsite team. When the onsite/offshore ratio becomes 0.75/0.25, The project schedule was 159 days and 153 days with low and medium level of training by onsite team to offshore team. The project size becomes 1,092 and 1,101 low and medium training cases due to addition of reworked tasks. By increasing the offshore resources by making onsite/offshore resources to 0.5/0.5, the schedule stretched to 212 and 202 days for medium and higher values of offshore training time. The total numbers of tasks worked in both these scenarios were 1,117 and 1,132, respectively. The introduction of offshore resources late in the project was useful in marginal improvement of project schedule.

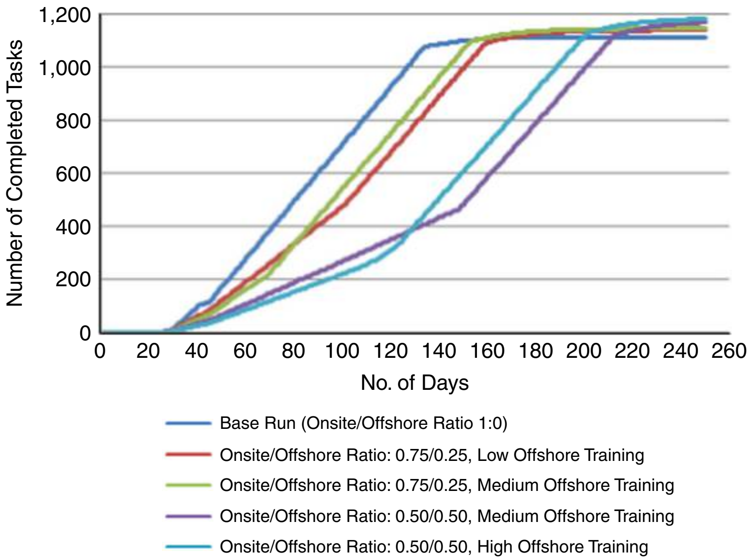

Figure 7. Project tasks completion for various onsite/offshore ratio and offshore training

The project schedule suffers with increase in offshore strength due to lower team productivity. The onsite team has to spend more time to train the offshore team, which reduces the overall team productivity. Figure 8 shows the total team productivity under various scenarios. It can be found out from Figure 8 that the team productivity suffers due to introduction of offshore resources. The maximum productivity possible for the team is 12 tasks/day as the team strength is 12. We have assumed that maximum productivity for a person can be one task/day. The team becomes fully productive on 45th day when the project is executed at onsite location only. The maximum productivity was achieved on 101th and 69th day, respectively for low and medium offshore training when onsite/offshore ratio was 0.75/0.25. The values were stretched to 148th and 122nd day for onsite/offshore ratio 0.5/0.5 with medium and high offshore training, respectively. All these days correspond to change in task completion pattern in Figure 7 when the rate of task completion increases.

Although the project schedule suffers in the GSD environment, there can be a cost saving as offshore resources are cheaper than onsite resources. As mentioned before, we will use values of onsite and offshore resources at 4X/day and X/day, respectively. The Table II provides the project cost and schedule for different scenarios of project execution.

Also for our model, we assumed that the offshore team needs to attain business knowledge expertise level 2 to enable them to work in coding and testing phase. The experts suggested that the offshore Offshore_Knowledge_Transfer_Coefficient value gets higher with proper requirement and design document. The onsite team can impart business knowledge to offshore team during requirement analysis and design phase. In that scenario, the requirement analysis and design work is executed at onsite location and coding, testing work is sent to offshore location. The onsite team is

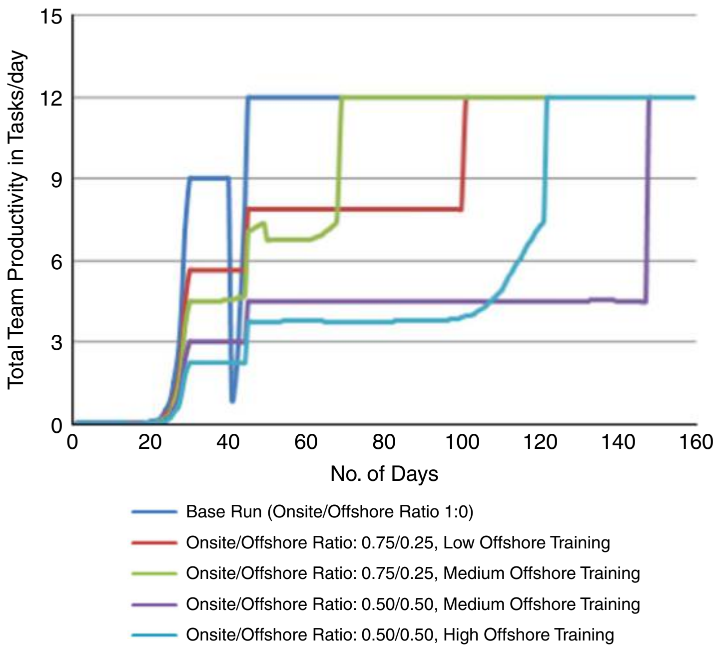

Figure 8.
Total team productivity for various onsite/offshore ratio and offshore training

Execution mode	Onsite/ offshore ratio	Offshore team introduction day	Average training hours for offshore (by onsite team)	Completion schedule time	Project cost	Percentage cost saving (comparison to onsite execution)
Onsite	1/0		na	134 days	6432X	
GSD	0.75/0.25	0	Low	159 days	6201X	4
			Medium	153 days	5967X	8
		50	Medium	161 days	5908X	9
GSD	0.5/0.5	0	Medium	212 days	6360X	2
			High	202 days	6060X	6
		50	High	218 days	6240X	3
GSD	0.25/0.75	The cost and schedule of affected in very adverse manner due to large presence at offshore location. So this scenario is not practically possible				

Note: Total team strength; 12

Table II. Project cost and schedule

released before beginning of the coding phase. We simulated the situation in our model with initial 18 people. The onsite resources were released from project when requirement analysis and design work are complete. The offshore team is allowed to complete the coding and testing work. The results are given in Table III.

It can be seen from Table III that there can be higher cost saving by executing coding and testing work at offshore location only. The onsite team can be released after completion of requirement analysis and design work. The experts suggested that offshore team can execute the coding and testing work without any help from onsite team, when few members of offshore team are highly knowledgeable in business domain. As per the experts, few offshore team members travel to onsite locations and participate in requirement analysis and design work. They return back to offshore location at the beginning of coding work and help other team members.

7.2 Project EOR and EOL characteristics

The EOR and EOL values do not influence the dynamics of the project directly; nonetheless, they do influence the decision-making process of the manager which, in turn, influences the course of the project. So it is important to find out the EOR and EOL characteristics for the project execution. Figure 9 provides the EOR values and Figure 10 provides the EOL values for our various policy runs. The EOR values increases with increase of work performed at offshore location. The EOR value was

Execution mode	Onsite/ offshore team strength	Average training hours for offshore (by onsite team)	Completion schedule time	Onsite team release day	Project cost	Percentage cost saving (comparison to onsite execution)
Onsite	12/0	na	134 days	6432 X	6432 X	
GSD	9/9	Low	197 days	91	5049 X	22
		Medium	193 days	86	4833 X	25
GSD	6/12	Low	222 days	176	6888 X	–07
		Medium	199 days	123	5340 X	17

Note: Total team strength; 18

Table III. Project cost and schedule for coding and testing work execution at offshore location only

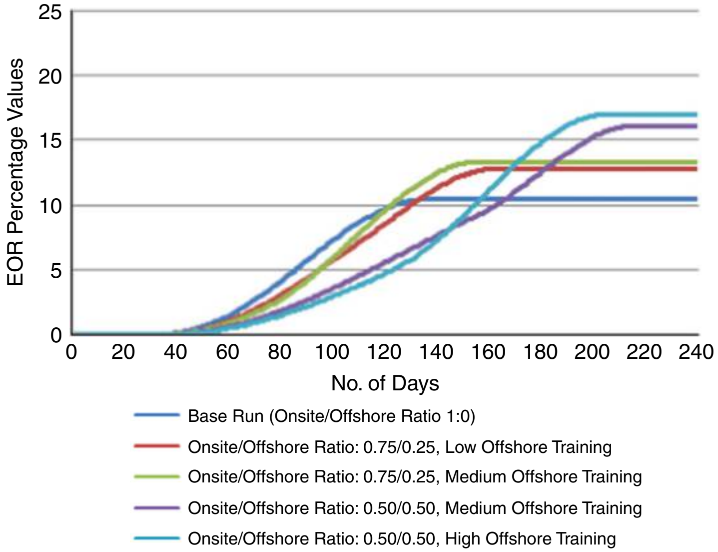

Figure 9.
Extent of rework (EOR) values for various onsite/offshore ratio and offshore training

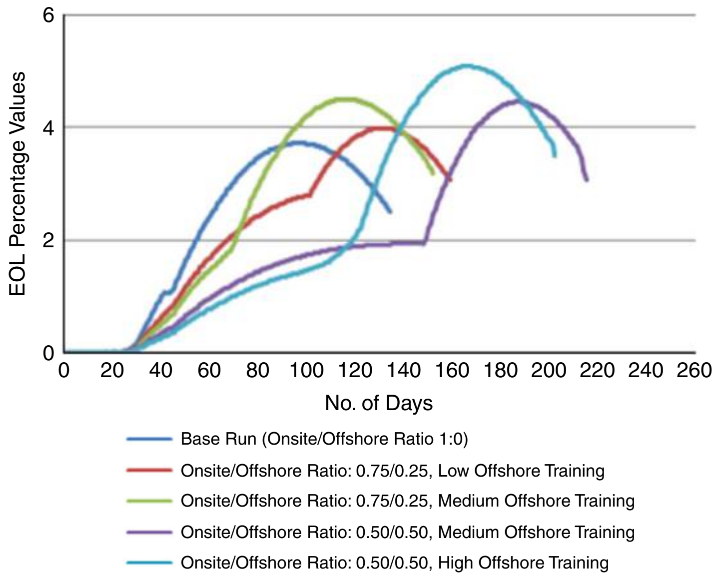

Figure 10.
Extent of latent error (EOL) values for various onsite/offshore ratio and offshore training

11.41 percent for onsite only execution of the project. The value increases to 17.03 percent with 50 percent offshore resources. This is because of high error proneness of the offshore team. Also the EOL values are higher for high amount of work execution at offshore location. So projects are left with higher latent error with execution at offshore location. It may require higher maintenance effort after completion of development work. The EOL values increases in Figure 10 when offshore team productivity rises after completion of business process training.

8. Summary of research findings

The research work to find out the policy to improve the cost and schedule for development projects in GSD environment by simulation. The purpose of our model was to study the project characteristics in GSD for various combinations of onsite-offshore manpower, training time with business user, offshore training time and suitable time to introduce the offshore team. We found that there is drop in productivity in GSD as reported by various researchers in literature (e.g. Muhairat *et al.*, 2010; Ramasubbu and Balan, 2007). The EOR and EOL values increased with the increase in team size at offshore location. So there is a possibility of higher latent error for software projects executed in GSD environment. In spite of all the shortcomings, we found that there is a possibility to save development cost in GSD.

We found that it is possible to save cost for the development projects by outsourcing. But the project management team should be careful not to keep high percentage of manpower at offshore location as it will have negative effect on cost and schedule due to higher training overhead, lower productivity and higher error proneness at offshore location. Figure 11 shows the cost and schedule of a development project in two scenarios. The coding and testing work is executed only at the offshore location in Scenario 2. The simulation run shows that there can be cost saving in a development project by employing a large onsite team and releasing the onsite team after completion of the requirement analysis and design work. The highest cost saving is achieved when onsite/offshore ratio is maintained at 0.5. There was increase in cost and schedule completion time by increasing further offshore strength. The coding and testing work can be executed at offshore location without onsite team by good project documentation. Also some members of the offshore team can travel to onsite during requirement analysis and design phase and later join offshore team for coding and testing work. It can improve the business knowledge level of the offshore team improving their productivity.

There can be presence of more latent error when offshore team is involved in a major way. The presence of higher latent errors can create more maintenance effort downstream. So cost saving may not be a main motivation factor in involving offshore team in development projects. The interviewed experts agreed with our finding. As one of our interviewed experts suggested in his own words "Cost saving is not the prime

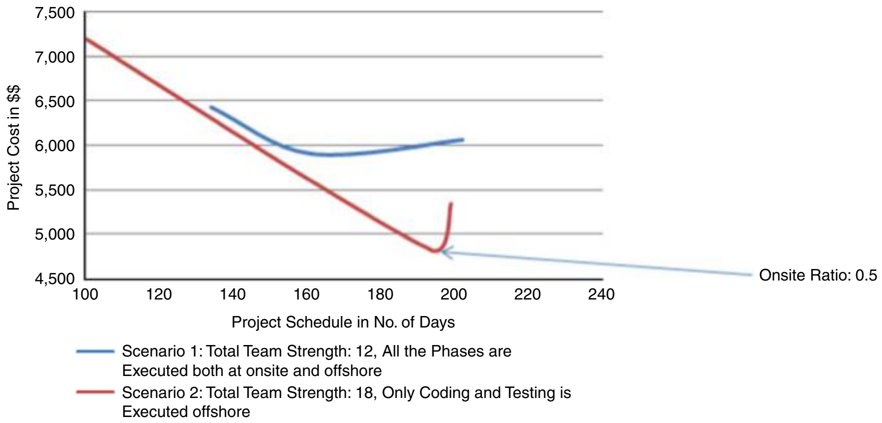

Figure 11. Project cost and schedule in different scenarios

factor in outsourcing development project. The main motivation factor in keeping a small offshore team for development project is to train the offshore team so that they can assume responsibility for maintenance work. There is no need of extra training session, if few offshore members are knowledgeable in business domain." The utility of offshore team is realized in coding and testing work and onsite team takes the major share of requirement analysis and design work. As another expert summarized it – "The domain, strategic and business process knowledge expertise requirement for SA and HLD phase of software development in development projects is very high, making it impossible to send it offshore". If we involve offshore, then we have to spend a lot of time in communicating with the offshore team, which drags the productivity of entire team down. Also there is a chance of sending confidential information to a remote location, which can create problem. The technical skill of offshore team members are utilized for coding and testing work, where the need of business knowledge is minimum. The ST phase is again executed at onsite location due to high communication requirement with the business user.

9. Conclusion

Cost reduction and control are often offered as internal reasons for outsourcing IS (Smith *et al.*, 1998; Alpar and Saharia, 1995; Lacity *et al.*, 2004). But it can have negative influence on project efficiency (Bosch and Bosch-Sijtsema, 2010; Smite *et al.*, 2010). The geographical dispersion can have negative influence on software development productivity and project performance (Avritzer *et al.*, 2010; Casey and Richardson, 2008, 2009; Milewski *et al.*, 2008). The paper found out that there can be cost saving for development projects in outsourcing environment by keeping offshore team only for coding and testing work. The project schedule is affected by having a offshore team. The larger the offshore team, bigger is the deviation in project schedule in comparison to onsite only execution. The requirement analysis and design work should be executed at onsite only. The offshore team should be trained in basic business knowledge to be effective in coding and testing work.

9.1 Theoretical contribution and future work

Our research work contributes to the literature in a number of ways. First, we made an effort to estimate cost, schedule and quality of software development project in GSD. Cost reduction and control are often offered as internal reasons for outsourcing (Alpar and Saharia, 1995; Loh, 1994). But there is a loss of productivity in GSD (Ramasubbu and Balan, 2007). There is a gap in literature with regard to modeling for estimation of cost, schedule and productivity of development project in GSD. This paper makes an attempt to fill it.

Second, the literature talks about the importance of business knowledge requirement in software projects especially in GSD. But there is no effort to find project characteristics in GSD in relation to business knowledge. We have provided a framework to use various kinds of business knowledge in finding project cost and schedule for development projects in GSD.

Finally, our work is a contribution toward understanding the working of Indian software industry. Little research work exists on the studies of software projects executed in Indian outsourcing industry. Our work can be very useful to extend the study of development project execution for various domains such as finance, manufacturing, etc. Also the research work can be extended for various kinds of software systems such as data warehousing and transaction system processing, etc for different kinds of methodology in software development such as agile software development and prototyping, etc. This work can be extended to find out the effect of coupling between various kinds of business knowledge while calculating the important project characteristics like cost, schedule and quality, etc. The research work can be extended to find different team structures (based on expertise) required at onsite and offshore locations to execute development projects in GSD. The system dynamics model can also be enhanced to model a scenario where the project team builds a workable product for the organization by extending the commercially available product. A large software team working for a particular client in multiple projects is a common occurrence in the software field. The system dynamics model can also be configured for modeling such a scenario.

9.2 Practical implications and limitations

Our simulation model can be used as a tool for project managers for re-engineering projects in GSD. It can provide insights into the characteristics of development projects, which can help to plan the project execution in advance. The training need of the development team at onsite and offshore locations can be estimated from the simulation run with knowledge of project complexity and initial knowledge level of software development team. So it can be used as a very useful tool in manpower planning in development project. The system dynamics tool can help the management in GSD environment to make intelligent decision to balance cost, schedule and quality of software development process. The system dynamics tool will be helpful to make informed decisions about the software development projects of varied complexity.

Limitation of the research work includes non-consideration of the interaction of various kinds of business knowledge for our model. Sometime it may very difficult to associate a knowledge type with a business process. It may be a combination of more than one type of business knowledge. The boundary between various types of business knowledge can be very thin and undefined.

References

Abdel-Hamid, T. and Madnick, S.E. (1991), *Software Project Dynamics: An Integrated Approach*, Prentice-Hall, Inc., Upper Saddle River, NJ.

Aktas, E. and Ulengin, F. (2005), "Outsourcing logistics activities in Turkey", *Journal of Enterprise Information Management*, Vol. 18 No. 3, pp. 316-329.

Alpar, P. and Saharia, A.N. (1995), "Outsourcing information system functions: an organization economics perspective", *Journal of Organizational Computing*, Vol. 5 No. 3, pp. 197-217.

Anda, B.C.D., Sjøberg, D.I.K. and Mockus, A. (2009), "Variability and reproducibility in software engineering: a study of four companies that developed the same system", *IEEE Transactions on Software Engineering*, Vol. 35 No. 3, pp. 407-429.

Avritzer, A., Paulish, D., Cai, Y. and Sethi, K. (2010), "Coordination implications of software architecture in a global software development project", *Journal of Systems and Software*, Vol. 83 No. 10, pp. 1881-1895.

Bairi, J. and Manohar, B.M. (2011), "Critical success factors in gaining user customer satisfaction in outsourced IT services", *Journal of Enterprise Information Management*, Vol. 24 No. 6, pp. 475-493.

Battin, R.D., Crocker, R., Kreidler, J. and Subramanian, K. (2001), "Leveraging resources in global software development", *IEEE Software*, Vol. 18 No. 2, pp. 70-77.

Beulen, E., Tiwari, V. and van Heck, E. (2011), "Understanding transition performance during offshore IT outsourcing", *Strategic Outsourcing: An International Journal*, Vol. 4 No. 3, pp. 204-227.

Boehm, B.W. (1981), *Software Engineering Economics*, Printice-Hall, Englewood Cliffs, NJ.

Boehm, B.W., Abts, C., Brown, A.W., Chulani, S., Clark, B.K., Horowitz, E., Madachy, R., Reifer, D.J. and Steece, B. (2000), *Software Cost Estimation with COCOMO II*, Printice-Hall, Upper Saddle River, NJ.

Bosch, J. and Bosch-Sijtsema, P. (2010), "From integration to composition: on the impact of software product lines, global development and ecosystems", *Journal of Systems and Software*, Vol. 83 No. 1, pp. 67-76.

Carmel, E. and Agarwal, R. (2001), "Tactical approaches for alleviating distance in global software development", *IEEE Software*, Vol. 18 No. 2, pp. 22-29.

Casey, V. and Richardson, I. (2008), "Virtual teams: understanding the impact of fear", *Software Process: Improvement and Practice*, Vol. 13 No. 6, pp. 511-526.

Casey, V. and Richardson, I. (2009), "Implementation of global software development: a structured approach", *Software Process Improvement and Practice*, Vol. 14 No. 5, pp. 247-267.

Chua, A.L. and Pan, S.L. (2008), "Knowledge transfer and organizational learning in IS offshore sourcing", *Omega*, Vol. 36 No. 2, pp. 267-281.

Cohen, W.M. and Levinthal, D.A. (1990), "Absorptive capacity: a new perspective on learning and innovation", *Administrative Science Quarterly*, Vol. 35 No. 1, pp. 128-152.

Conchuir, E.O., Holmstrom-Olson, H., Agerfalk, P.J. and Fitzgerald, B. (2009), "Benefits of global software development: exploring the unexplored", *Software Process Improvement and Practice*, Vol. 14 No. 4, pp. 201-212.

Cullen, S., Seddon, P.B. and Willcocks, L. (2006), *Managing Outsourcing: The Lifecycle Imperative*, London School of Economics and Political Science, London.

Cusumano, M.A. (2008), "Managing software development in globally distributed teams", *Communications of the ACM*, Vol. 51 No. 2, pp. 15-17.

Desouza, K.C. and Evaristo, J.R. (2006), "Project management offices: a case of knowledge-based archetypes", *International Journal of Information Management*, Vol. 26 No. 5, pp. 414-423.

Dibbern, J., Winkler, J. and Heinzl, A. (2008), "Explaining variations in client extra costs between software projects offshored to India", *MIS Quarterly*, Vol. 32 No. 2, pp. 333-366.

El Emam, K. and Koru, A.G. (2008), "A replicated survey of IT software project failures", *IEEE Software*, Vol. 25 No. 5, pp. 84-90.

García-Crespo, A., Colomo-Palacios, R., Soto-Acosta, P. and Ruano-Mayoral, M. (2010), "A qualitative study of hard decision making in managing global software development teams", *Information Systems Management*, Vol. 27 No. 3, pp. 247-252.

Gowan, A.J. Jr and Mathieu, R.G. (2005), "The importance of management practices in IS project performance: an empirical study", *Journal of Enterprise Information Management*, Vol. 18 No. 3, pp. 235-255.

Guetat, S. and Dakhli, S.B.D. (2010), "The complementary roles of information systems and knowledge management systems: a framework based on popper's three worlds theory", *Communications in Computer and Information Science*, Vol. 109 No. 1, pp. 374-384.

Hamid, A. and Madnick, S. (1991), *Software Project Dynamics: An Integrated Approach*, Prentice-Hall, Englewood Cliffs, NJ.

Herbsleb, J.D. and Mockus, A. (2003), "An empirical study of speed and communication in globally distributed software development", *IEEE Transactions on Software Engineering*, Vol. 29 No. 9, pp. 481-494.

Herbsleb, J.D. and Moitra, D. (2001), "Global software development", *IEEE Software*, Vol. 18 No. 2, pp. 16-20.

Hernández-López, A., Colomo Palacios, R., García Crespo, A. and Soto-Acosta, P. (2010), "Trust building process for global software development teams. A review from the literature", *International Journal of Knowledge Society Research*, Vol. 1 No. 1, pp. 65-82.

ISBSG (2009), "ISBSG database", available at: www.isbsg.org (accessed August 10, 2011).

Jalote, P. and Jain, G. (2006), "Assigning tasks in a 24-h software development model", *Journal of Systems and Software*, Vol. 79 No. 7, pp. 904-911.

Kappelman, L.A., McKeeman, R. and Zhang, L. (2006), "Early warning signs of IT project failure: the dominant dozen", *Information Systems Management*, Vol. 23 No. 4, pp. 31-36.

Kemere, C.F. (1987), "An empirical validation of software cost estimation models", *Communications of ACM*, Vol. 30 No. 5, pp. 416-429.

Kobitzsch, W., Rombach, D. and Feldmann, R.L. (2001), "Outsourcing in India", *IEEE Software*, Vol. 18 No. 2, pp. 78-86.

Kommeren, R. and Parviainen, P. (2007), "Philips experiences in global distributed software development", *Empirical Software Engineering*, Vol. 12 No. 6, pp. 647-660.

Kruchten, P. (1999), *The Rational Unified Process: An introduction*, Addison-Wesley Longman Publishing Co. Inc., Boston, MA.

Kultur, Y., Kocaguneli, E. and Bener, A.B. (2009), "Domain specific phase by phase effort estimation in software projects", International Symposium on Computer and Information Sciences, Guzelyurt, North Cyprus, September.

Lacity, M., Willcocks, L. and Feeny, D. (2004), "Commercializing the back office at lloyds of london: outsourcing and strategic partnerships revisited", *European Management Journal*, Vol. 22 No. 2, pp. 127-140.

Lacity, M.C. and Rottman, J.W. (2009), "Effects of offshore outsourcing of information technology work on client project management", *Strategic Outsourcing: An International Journal*, Vol. 2 No. 1, pp. 4-26.

Lai, L.S.L. (1997), "A synergistic approach to project management in information systems development", *International Journal of Project Management*, Vol. 15 No. 3, pp. 173-179.

Lee, J.N. (2001), "The impact of knowledge sharing, organizational capability and partnership quality on IS outsourcing success", *Information & Management*, Vol. 38 No. 5, pp. 323-335.

Loh, L. (1994), "An organizational – economic blueprint for information technology outsourcing: concepts and evidence", *Proceedings of the Fifteenth International Conference on Information Systems*, *December 14-17*, *Vancouver*, pp. 73-89.

McBride, T., Henderson-Sellers, B. and Zowghi, D. (2007), "Software development as a design or a production project: an empirical study of project monitoring and control", *Journal of Enterprise Information Management*, Vol. 20 No. 1, pp. 70-82.

Mayasandra, R., Pan, S.L. and Leidner, D. (2011), "Examining the strategic alignment and implementation success of a KMS: a subculture-based multi-level analysis", *Information Systems Research*, Vol. 22 No. 1, pp. 39-59.

Milewski, A.E., Tremaine, M., Ko¨bler, F., Egan, R., Zhang, S. and O'Sullivan, P. (2008), "Guidelines for effective bridging in global software engineering", *Software Process: Improvement and Practice*, Vol. 13 No. 6, pp. 477-492.

Mishra, D. and Mahanty, B. (2014), "The effect of onsite-offshore work division on project cost, schedule, and quality for re-engineering projects in Indian outsourcing software industry", *Strategic Outsourcing: An International Journal*, Vol. 7 No. 3, pp. 198-225.

Mishra, D. and Mahanty, B. (2015), "Business knowledge requirements and onsite offshore work division in Indian software outsourcing projects", *Strategic Outsourcing: An International Journal*, Vol. 8 No. 1, pp. 76-101.

Muhairat, M., Aldaajeh, S. and Al-Qutaish, R.E. (2010), "The impact of global software development factors on effort estimation methods", *European Journal of Scientific Research*, Vol. 46 No. 2, pp. 221-232.

Ngwenyama, O.K. and Sullivan, W.E. (2007), "Outsourcing contracts as instruments of risk management: insights from two successful public contracts", *Journal of Enterprise Information Management*, Vol. 20 No. 6, pp. 615-640.

Peterson, E.R. and Gott, J. (2011), "Offshoring opportunities amid economic turbulence: the A.T. Kearney global services location index", A.T. Kearney, Inc, Chicago, IL.

Pidd, M. (2000), *Tools for Thinking – Modeling in Management Science*, Wiley, New York, NY.

Popper, K.R. (1972), *Objective Knowledge, An Evolutionary Approach*, Clarendon Press, Oxford.

Pressman, R. (1997), *Software Engineering: A Practitioner's Approach*, The McGraw-Hill Companies Inc, New York, NY.

Putnam, L.H. and Myers, W. (1996), *Executive Briefing. Controlling Software Development*, IEEE Computer Society Press, Silver Spring, MD.

Qudrat-Ullah, H. and Seong, B.S. (2010), "How to do structural validity of a system dynamics type simulation model: the case of an energy policy model", *Energy Policy*, Vol. 38 No. 5, pp. 2216-2224.

Rai, V.K. and Mahanty, B. (2002), "Dynamics of schedule pressure in software projects", *Proceedings of the 20th International Conference of the System Dynamics Society, The System Dynamics Society*, *Palermo, July-August*.

Raman, R. and Chadee, D. (2011), "A comparative assessment of the information technology services sector in India and China", *Journal of Contemporary Asia*, Vol. 41 No. 3, pp. 453-470.

Ramasubbu, N. and Balan, R.K. (2007), "Globally distributed software development project performance: an empirical analysis", *Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, pp. 125-134.

Ramasubbu, N., Krishnan, M.S. and Kompalli, P. (2005), "Leveraging global resources: a process maturity framework for managing distributed development", *IEEE Software*, Vol. 22 No. 3, pp. 80-86.

Rathi, V.S. and Joshi, S.K. (2010), "Outsource software development to India – a quick glance on tasks involved", available at: www.ezinearticles.com (accessed June 10, 2011).

Ruiz, M., Ramos, I. and Toro, M. (2001), "A simplified model of software project dynamics", *Journal of Systems and Software*, Vol. 59 No. 3, pp. 299-309.

Rus, I. and Lindvall, M. (2002), "Knowledge management in software engineering", *IEEE Software*, Vol. 19 No. 3, pp. 26-38.

Sakthivel, S. (2005), "Virtual workgroups in offshore systems development", *Information and Software Technology*, Vol. 47 No. 5, pp. 305-318.

Sargent, R.G. (1981), "An assessment procedure and a set of criteria for use in the evaluation of Computerized models and computer-based modeling tools", Final Technical Report No. RADC-TR-80-409, US Air Force.

Sargent, R.G. (2010), "Verification and validation of simulation models", *Proceedings of the 2010 Winter Simulation Conference*, Syracuse University, Syracuse.

Setamanit, S., Wakeland, W. and David Raffo, W. (2007), "Using simulation to evaluate global software development task allocation strategies", *Software Process: Improvement and Practice*, Vol. 12 No. 5, pp. 491-503.

Smite, D., Wohlin, C., Gorschek, T. and Feldt, R. (2010), "Empirical evidence in global software engineering: a systematic review", *Empirical Software Engineering*, Vol. 15 No. 1, pp. 91-118.

Smith, M.A., Mitra, S. and Narasimhan, S. (1998), "Information systems outsourcing: a study of pre-event firm characteristics", *Journal of Management Information System*, Vol. 15 No. 2, pp. 61-93.

Sommerville, I. (2004), *Software Engineering*, 7th ed., International Computer Science Series, Addison Wesley.

Sooraj, P. and Mohapatra, P.K.J. (2008), "Modeling the 24-h software development process", *Strategic Outsourcing: An International Journal*, Vol. 1 No. 2, pp. 122-141.

Stohr, E.A. and Konsynski, B.R. (1992), *Information Systems and Decision Processes*, IEEE Computer Society Press, Los Alamitos, CA.

Tan, T., Boehm, B. and Clark, B. (2011), "An investigation on application domains for software effort distribution patterns", Center for System Engineering, CA.

Tiwana, A. (2004a), "An empirical study of the effect of knowledge integration on software development performance", *Information and Software Technology*, Vol. 46 No. 13, pp. 899-906.

Tiwana, A. (2004b), "Beyond the black box: knowledge overlaps in software outsourcing", *IEEE Software*, Vol. 21 No. 5, pp. 51-58.

Webster, J. and Watson, R.T. (2002), "Analyzing the past to prepare the future", *MIS Quarterly*, Vol. 26 No. 2, pp. xiii-xxiii.

Zhuge, H. (2002), "Knowledge flow management for distributed team software development", *Knowledge-Based Systems*, Vol. 15 No. 8, pp. 465-471.

Zhuge, H., Guo, W. and Li, X. (2007), "The potential energy of knowledge flow", *Concurrency and Computation: Practice and Experience*, Vol. 19 No. 14, pp. 2067-2090.

Further reading

Ravishankar, M.N., Pan, S.L. and Leidner, D.E. (2011), "Examining the strategic alignment and implementation success of a KMS: a subculture-based multilevel analysis", *Information Systems Research*, Vol. 22 No. 1, pp. 39-59.

About the authors

Debasisha Mishra is working as an Assistant Professor in Rajiv Gandhi Indian Institute of Management, Shillong, India. He obtained his Doctoral Degree from Department of Industrial Engineering and Management at Indian Institute of Technology (IIT) Kharagpur, India. He did his bachelor of engineering (BE) from National Institute of Technology (NIT) Rourkela in 1995 and MTech in Industrial Management and Engineering from IIT Kanpur, India in 1997. He has worked in information technology industry for more than 12 years in India and USA in

various capacities. Debasisha Mishra is the corresponding author and can be contacted at: debasisha_mishra@hotmail.com

Biswajit Mahanty is a Professor in the Department of Industrial and Systems Engineering at Indian Institute of Technology (IIT) Kharagpur, India. He obtained his Doctoral Degree from IIT Kharagpur, India in 1995. He did his Master of Technology (MTech) in Industrial and System Engineering from IIT Kharagpur, India in 1989 and Bachelors in Technology (BTech) in Mechanical Engineering from IIT Kharagpur, India in 1984. His area of specialization includes system dynamics, operations research, information systems and project management. He has many publications in many reputed journals in these areas.