BE PRODUCTIVE OR PERISH

SP Rahmesh
BE, PGDPM, MLS, M.PHIL, PMP, PhD
Senior Project Manager, HCL Technologies ltd
rahmesh.sp@gmail.com

Abstract:

The Critical factor which causes an organization's downfall and decline is lower productivity. Any failure to meet the required productivity level will potentially result in higher costs for services and commodities will not be competitive enough to stay in the market. All the business units are trying very hard to remain competitive in the market. Therefore, it is paramount importance for businesses to implement strategies to make improvements in productivity always. World economies are depending on software for delivery excellence and higher economic outcomes. Software field remains very challenging as it is more people centric in nature with respect to software engineering processes and practices. This paper covers a thorough study of a large number of factors influencing throughout the lifecycle of a software project.

Key words: Software economics, Requirements engineering, risk factors, metrics, Software Estimation models

Objective of the Study:

1. Carry out a comprehensive study on factors influencing productivity of software projects and draw a correlations between them
2. To make an in-depth probe and identify the blend of factors that can be used as a composite yardstick to determine baselines
3. Finalize the factors which have high degree of influence on productivity
4. Recommend methodologies and modified processes that can enhance productivity of IT projects

Scope of the Study:

Productivity is becoming an emotional topic of discussion among software professionals and consultants today. There are various factors influences the productivity positively or negatively impacting economics. In an era of tight budget conditions and increased customer expectations, determining productivity measures are very essential for the outsourcing organizations to continue their services to client organizations on various projects. Therefore the study aims at finding out factors influencing the performance of the software projects. This study covered software development, enhancement, maintenance and support projects across various service organizations like HCL, CTS, WIPRO, Cap Gemini, Accenture and few other small organizations. This research work studied the factors covering many SDLC processes and activities like requirements engineering, tools & techniques, software estimations, knowledge management, quality & system improvement programs, experience of resources, defect management, people aspects and risk management. Finally this research work identified informed suggestions, recommendations to improve overall performance and delivery capability of software projects.

Why should organizations concentrate on enhancing productivity?

To increase profits and revenue
To lower the operational costs
To maximize available resources
To use all the tools and techniques effectively
To gain market share

Research Design and Methodology

The research design used for the study is descriptive cum exploratory design. The methodology of the study is based on primary data gathered through a well-framed and structured questionnaire to obtain the opinion of the respondents. Sampling technique used for the study was convenience sampling and the size of the sample was 456.

Review of Literature

Risk Management Program (1)

According to Rita Mulcahy (2001), the Risk management process was a systematic and proactive approach to taking control of IT projects and to reduce uncertainties. Risk Management should be applied to both small and large projects. Risk Management involves minimizing the consequences of adverse events as well as maximizing the results of positive events. Therefore risks can be good or bad events. Risk Management is a process and following that process is important to achieve results. It allows us to take control of the project, rather than letting the

project to take control on us.

The steps of Risk Management process as per project management body of knowledge (PMBOK) are (2)

1. **Risk Management Planning:** Determine how Risk Management will be done, who will be involved and procedure to be adopted.
2. **Risk Identification:** Determine specific risks by project and by task. Getting the stakeholders involved in making a long, comprehensive list of risks.
3. **Quantitative and Qualitative:** Analyzing the risks obtained in the identification step and deciding which risk warrant a response in the next step. Probability and impact are determined qualitatively, a choice is made whether or not a quantitative evaluation is necessary and the project risk score and probability of meeting project objectives are determined.
4. **Risk Response planning:** Determining what can be done to reduce the overall risk of the project by decreasing the probability or impact of the shortlisted risks and increasing probability or impact of opportunities.
5. **Risk monitoring and control:** Execute the risk response plan as risk events occur throughout all the project phases.

Key factors, Processes and Priorities (3)

Caper Jones (2007) in his article said that the most common problem in the software industry for large systems is that of intense but artificial schedule pressure applied to the programmers by their managers, by senior executives in their companies or by the clients. Unfortunately, intense schedule pressure leads to carelessness, which in turn drives up the chances of making errors and introducing bugs. The overall impact of intense schedule pressure is not what is desired, but instead tends to lengthen schedules because the software does not work well enough to be released.

If one know the size of a software application and set schedules that are much shorter than the productivity level of the organization, one will have a very small chance of achieving the schedules and a very large chance of running late and over budget because schedule pressure one apply will cause haste and carelessness and will drive up the bug or error probably to dangerous levels. So it is very important to measure the size of the work in terms of standard measure and apply the productivity value based on its own productivity value or the value given by the productivity specialists or consultants like SPR.com (Software productivity research council) or David consulting Inc. (davidconsulting.com).

Requirements Engineering (2)

Scope Management is a key knowledge area in the whole of project management aspects. Scope management includes all are the processes to ensure that project includes all the work required and only the work required to complete the project successfully. Managing scope primarily deals with defining and controlling what is and is not included in the project. The processes are

Collect requirements: The process of defining and documenting stake holder requirements to meet the project objectives.

Define Scope: The process of developing a detailed description of the product and project.

Create WBS: The process of subdividing deliverables and project tasks into smaller and more manageable components.

Verify Scope: The process of acceptance of the completed deliverables.

Control Scope: The process of monitoring the status of the project and product scope and managing changes to the scope baseline.

Software Estimation Methodologies (4)

There are many software size estimation methodologies / approaches are in practice and prominently used by the practitioners in the industry. Researchers have been working on for quite long on many estimation techniques. Estimation models in the early stages were typically based on regression analysis or mathematical models. During 1970s and 1980s models derived were from historical data of various software projects. Among many estimation models expert estimation, COCOMO, Function Point and derivatives of function point like Use Case Point, Object Points are most commonly used. While Lines of Code (LOC) is most commonly used size measure for 3GL programming and estimation of procedural languages, IFPUG FPA originally invented by Allen Alrecht at IBM has been adopted by most in the industry as alternative to LOC for sizing development and enhancement of business applications. FPA provides measure of functionality based on end user view of application software functionality. Some of the commonly used estimation techniques are as follows:

- **Lines of Code (LOC):** A formal method to measure size by counting number of lines of Code, Source Lines of Code (SLOC) has two variants- Physical SLOC and Logical SLOC. While two measures can vary significantly care

must be taken to compare results from two different projects and clear guideline must be laid out for the organization.

- **IFPUG FPA:** Formal method to measure size of business applications. Introduces complexity factor for size defined as function of input, output, query, external input file and internal logical file.
- **Mark II FPA:** Proposed and developed by Mark Simons and useful for measuring size for functionality in real time systems where transactions have embedded data
- **COSMIC Full Function Point (FFP):** Proposed in 1999, compliant to ISO 14143. Applicable for estimating business applications that have data rich processing where complexity is determined by capability to handle large chunks of data and real time applications where functionality is expressed in terms of logics and algorithms.
- **Quick Function Point (QFP):** Derived out of FPA and uses expert judgment. Mostly useful for arriving at a ballpark estimate for budgetary and marketing purposes or where go or no go decision is required during project selection process.
- **Object Points:** Best suited for estimating customizations. Based on count of raw objects, complexity of each object and weighted points.
- **Predictive Object Points:** Tuned towards estimation of the object oriented software projects. Calculated based on weighted methods per class, count of top level classes, average number of children, and depth of inheritance.
- **Estimation by Analogy:** Cost of project is computed by comparing the project to a similar project in the same domain. The estimate is accurate if similar project data is available.

Software Estimation methodologies (Documentation) (5)

Every manager and cost estimating specialists who deals with large applications needs to understand that a typical software system produces more than 100 words for every line of code. Cost and schedules for creating these words are greater than the cost of actually writing the source code.

1. Software is the most labour intensive of any U.S industry in terms of the amount of human effort required to create a project
2. Software being abstract has become a document intensive industry and tends to create a larger volume of text and graphics based than almost any other industry.
3. ISO 9000, CMMI quality standards and processes demands heavy documentation work to satisfy audit requirements
4. Learning to use software applications often requires more training and more tutorial info than any other consumer product
5. Major cost drives for software projects in rank order of total effort are
    1. Bug fixing
    2. Paper documentation (ex: requirements specifications, plans, reports, test cases, plans, user manuals, implementation plans and etc.)
    3. Meeting and communications
    4. Coding
    5. Project management effort

When agile methods have had some successes but there are issues. The problems with the agile approach of replacing paper documentation with face-to-face meetings, for projects with thousands of users, agile methods are not suitable. For military projects documentation, traceability from requirements to end product are critical for reference. For long range projects for maintenance, lack of documentation may prove troublesome.

Software Estimation Methodologies (Schedule Pressure)

The most common problem in the software industry for large systems is that of intense but artificial schedule pressure applied to the programmers by their managers, by senior executives in their companies or by the clients. Unfortunately, intense schedule pressure leads to carelessness, which in turn drives up the chances of making errors and introducing bugs. The overall impact of intense schedule pressure is not what is desired, but instead tends to lengthen schedules because the software does not work well enough to be released.

If one know the size of a software application and set schedules that are much shorter than the productivity level of the organization, one will have a very small chance of achieving the schedules and a very large chance of running late and over budget because schedule pressure one apply will cause haste and carelessness and will drive up the bug or error probably to dangerous levels. So it is very important to measure the size of the work in terms of standard measure and apply the productivity value based on its own productivity value or the value given by the productivity specialists or consultants like SPR.com (Software productivity research council) or David consulting Inc. (davidconsulting.com).

Defect Free Programming (7)

The following are the key aspects one should follow to achieve defect free software.

1) Believe Defect-Free Software is Possible
2) Think Defect-Free Software is Important
3) Commit to Delivering Defect-Free Software
4) Design Your Code for Simplicity and Reliability
5) Trace Every Line of Code When Written
6) Review Code by Programmer Peers
7) Build Automated QA into Your Code
8) Build and Test Daily
9) Use Automated Checking Wherever Possible

Quality Management Processes (6)

Project Quality Management includes the processes and tasks of the performing organization that determine quality processes, objectives and responsibilities so that the project will be able to satisfy the needs for which the project was under taken. It implements through policies and procedures with continuous process improvement activities throughout the project life cycle.

Plan Quality: The process of identifying quality requirements and standards for the project and product and documenting how the project will demonstrate compliance level

Quality Assurance: The process of auditing the quality requirements and the results from quality control measurements to ensure appropriate quality standards and operational definitions are used

Quality Control: The process of monitoring and recording results of executing the quality activities to assess performance and recommend necessary changes.

Table 1. Hypothesis and Findings

From the above table, it is evident that there is a significant difference between risk management aspects, project metrics, productivity improvement programs, usage of tools and techniques for detecting faults and for testing with regard to overall dimensions of productivity traits. Since the P value is less than 0.05, the null hypotheses listed below are rejected at 5% level of significance with regard to dimensions of Productivity traits.

Few important Null Hypotheses are listed below:

(a) There is no significant difference between Risk Management Program with regard to dimensions of Productivity traits.
(b) There is no significant difference between software metrics programs with regard to dimensions of Productivity traits.
(c) There is no significant difference between Productivity Improvement programs with regard to dimensions of Productivity traits.
(d) There is no significant difference between Use of tools and techniques to detect the faults and errors to improve productivity with regard to dimensions of Productivity traits.
(e) There is no significant difference between usages of tools for testing the project to perform the Testing better with regard to dimensions of Productivity traits.

Research Findings and Recommendations

Graph I: Tools and techniques

The above graph is showing a direct impact on productivity in software projects when various tools are being used. During the SDLC process, using various tools and techniques for design, construction, test script generation and testing would help enhance the productivity without any extra effort. Automating all the current manual processes, batch job scheduling and monitoring are the few functions that project teams should look at to make changes.

Graph II: Risk management programs

Risk management is one of the nine functions of project management standards which the project managers should spend more time. The key activities as part of risk management are identifying the risks, mitigating the risks, preparing risk responses, estimating cost of risks, handling residual risks and risk monitoing. This research work identified clearly that proper risk management would help contain the situation under control.

Graph III: Defect free software management

The above graph on defect free software management proves that there is a direct co-relation between productivity and defect management. So during software development life cycle it is very essential to perform product reviews at all stages. Also efforts should be taken to control and remove the defects by altering the process or by adopting some tools and techniques to improve the productivity during project life cycle.

Graph IV: System improvement Programs

Similarly the above graph proves that productivity and process improvement programs in projects have a direct influence in enhancing productivity. It is highly recommended to adopt productivity improvement programs across all projects and engagements to ensure that performance and productivity is improved continuously

Table 2. Perception Levels of IT Professional

Manager Projects and Senior Managers have higher perceptions compared to Programmer Analysts and Project Leads in

Quality Management Processes and
Risk Management Programs
Project Leaders and Manager Projects have higher perceptions compared to Programmer Analyst and Senior Managers in
Requirement Engineering,
Defect Free Software Methods and
Delivery Performance
Senior Managers have lower level perceptions compared to Programmer Analysts, Project Leads, and Manager Projects in
Knowledge Management and
Tools and Techniques
Program Analysts have lower perception in
People Aspects and
Software Estimation Methods
Project Leads have higher perception in
System Improvement Programs and
Key Processes and Factors
Project Leads and Senior Managers have higher perception in
Skill and Expertise level of resources

A few factors that can help to improve the employee productivity at the workplace are:

1. Accountability

Every employee needs to be well aware that he is accountable for his actions and decisions and he can neither pass the buck nor pass the blame to someone else. This is to help him work meticulously and take decisions cautiously.

2. Follow-up

Employers often set targets and feel their job is done. No, every target or milestone set needs to be followed up as well, to see if the progress is sufficient or not. If needed any interim measures can be taken before it is too late to salvage a situation.

3. Manage the work force but avoid micromanagement

It is a known fact that a large pool of employees does need to be managed, provided direction and given assistance. But with this they must also be trusted, given freedom to operate in their style and adopt measures which they think are the best to deliver results.

This freedom to act as they deem fit helps to keep them encouraged, motivated and happy in the belief that they are trusted.

Micro management is a human tendency but one that is detrimental to achievement, since it makes mere puppets out of employees, who are expected to toe the boss' line and not think for themselves.

Employees need to think for themselves, analyze the consequences of every decision or action to be able to give their best to their jobs. And the employers must make it possible for their workers to do so.

4. Encourage, motivate, reward and recognize

The employer must ensure that all employees are encouraged to perform well. Encouraging employees will help the employer meet the goals of the organization. Innovative ways of motivating them spurs them even more.

Rewarding the hard work put in by employees makes them continue to work in the same fashion, and if the employee feels that his work is not appreciated in words or in material terms, he may gradually stop doing so, since he may feel that others working less are given the same too, so he need not work more.

5. Reach out to employees by seeking them out

Every employee loves to feel he has the ears of the management who will recognize him and listen to what he says. Display of inter personal skills in which the boss appears humane and one of among them helps employees to feel happy working for such organizations.

6. Demand realistic targets

Employers need to set realistic goals that are within the limits of achievement. While an aggressive employer may want his people to outstretch themselves to achieve farfetched goals, it may also burn them out. It is essential to perform time study, work study and method study to define the processes and procedures to help employees to feel comfortable in work place.

7. Team work

Team work always helps to improve workplace productivity since there is more input in the form of more ideas and minds at work.

8. Ensure that people enjoy their work

The best performing employee is the happy employee, and the employer has to find ways of making his people happy. Besides improving working conditions and developing good work culture employer should devise ways of making the work seem challenging and interesting rather than mundane and boring. Job rotation and skill improvement trainings should be organized for employees to improve morale and affinity towards the company.

10. Spend less time on meetings and more on action

The current trend to have more meetings and discussion rather than spending more time working to achieve results, leads to precious productive time loss. Meetings for reviews and sharing of ideas can

be limited and kept short.

11. Tools and equipment to raise productivity

The workplace must have the best machinery, devices and equipment that yield error free results in the minimum possible time. They should take the place of paper work and yield fast results.

Conclusion

"If you can measure it, you can improve it." Peter Drucker

The key advantage of looking economic perspective of software engineering provides a balanced view of overall life cycle of software projects. There is a constant demand for an economical model to do software size, effort and cost estimations consistently. Such an economical model would help avoid underestimates, over estimates, outright buy-ins that still plagues the industry today. Project managers should look at various options like reuse of software components, applying LEAN principles in projects, use all possible tools throughout the lifecycle, use of right estimation models, maintaining a proper knowledge management system and team motivation are few immediate steps to ensure software economics are in control.

References

Rita Mulcahy PMP (2001), "Tricks of the Trade for Project Managers (Risk Management)", Published by RMC Publication Inc., page 21-22

Project Management Body of Knowledge (PMBOK) (2008), fourth edition, PMI, Page No 103 - 128;

Capers Jones (2007), (Software Productivity Research LLC), "Estimating Software Costs" - "Bringing Realism to Estimating", (Estimating programming or coding), Tata McGraw-Hill Publications, Page 459

http://approachtoproject.com/component/k2/item/10-software-estimation-techniques.html

Capers Jones (2007), (Software Productivity Research LLC), "Estimating Software Costs" - "Bringing Realism to Estimating", (Estimating user and project documentation), Tata McGraw-Hill Publications, Page 519

Project Management Body of Knowledge (PMBOK) (2008), fourth edition, PMI, Page No 189 - 213

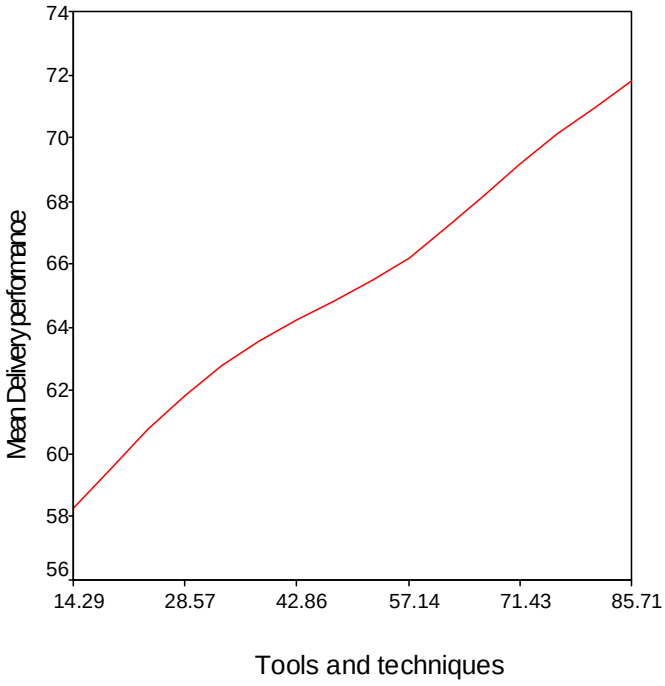
Source: http://www.tenberry.com/nodefect/steps.htm (Tenberry Software, Inc.)

Walker Royce and Kurt bittner (2009), "The Economics of iterative software development - steering towards better business results", pages 23, 25

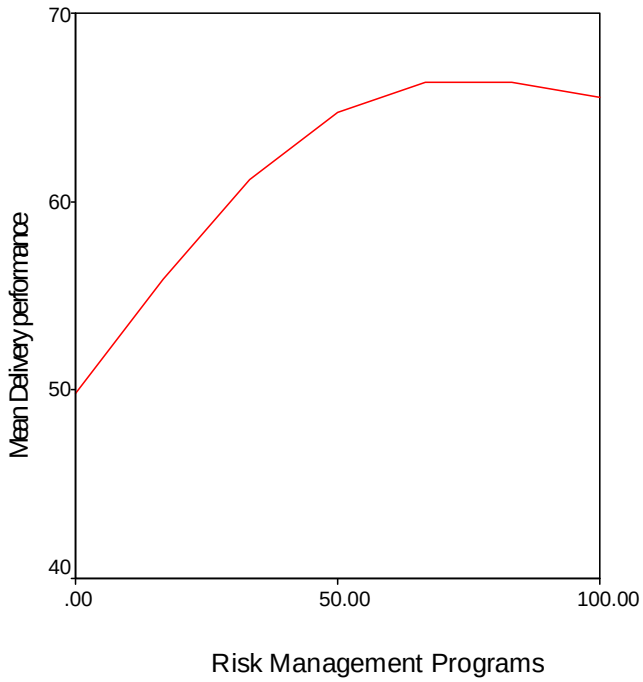
Table 1. Hypothesis and Findings

Dimensions of Productivity Traits	Risk Mgmt. Program	N	Mean	Std. Deviation	Std. Error Mean	t	Sig. (2-tailed)
Delivery performance	No	8	52.87	11.112	3.929	-3.745	0.000 *
	Yes	448	65.530	9.450	0.446		
Dimensions of Productivity Traits	Metrics program	N	Mean	Std. Deviation	Std. Error Mean	T	Sig Value (P)
Delivery performance	No	87	51.715	9.237	0.990	-16.090	0.000 *
	Yes	369	68.513	6.351	0.331		
Dimensions of Productivity Traits	Productivity Imp Program	N	Mean	Std. Deviation	Std. Error Mean	t	Sig.
Delivery performance	No	244	60.222	9.374	0.600	-15.184	0.000 *
	Yes	212	71.162	5.807	0.399		
Dimensions of Productivity Traits	Tools and techniques to detect faults	N	Mean	Std. Deviation	Std. Error Mean	t	Sig Value (P)
Delivery performance	No	115	63.434	7.542	0.703	-2.807	0.005
	Yes	341	65.939	10.149	0.549		
Dimensions of Productivity Traits	Tools for testing phase	N	Mean	Std. Deviation	Std. Error Mean	t	Sig Value (P)
Delivery performance	No	340	64.888	10.545	0.571	-2.071	.039
	Yes	116	66.538	5.979	0.555		

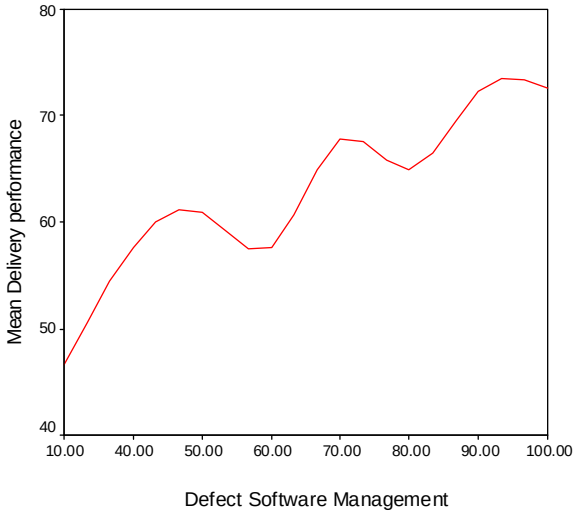
Graph I: Tools and techniques



Graph II: Risk management programs



Graph III: Defect free software management



Graph IV: System improvement Programs

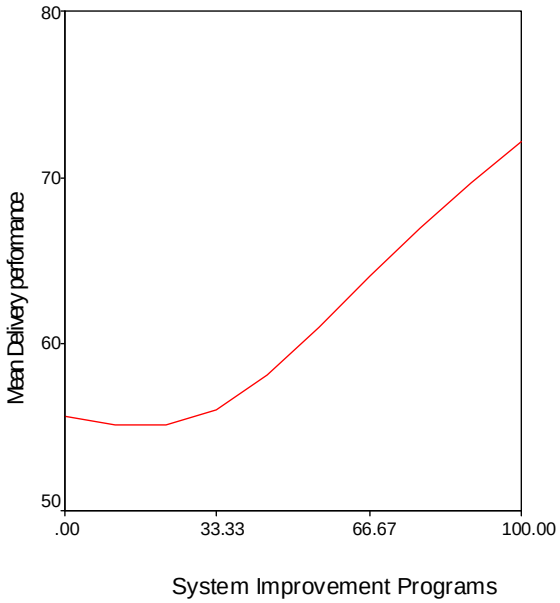


Table 2. Perception Levels of IT Professional

Productivity Traits	Programmer Analyst	Project Lead	Manager projects	Senior Manager
Quality Management Processes	56.51	59.22	65.54	62.50
Risk Management Programs	94.50	96.56	100.00	100.00
Requirement Engineering	49.41	61.79	55.06	35.71
Defect free Software Management	46.24	78.56	78.70	60.00
Knowledge Management	83.49	94.22	89.32	62.50
Tools and techniques	48.23	53.04	48.96	35.71
People aspects	73.36	81.98	76.85	77.59
Software Estimation methods	36.47	58.83	59.12	56.25
System Improvement Programs	58.72	93.12	59.10	50.00
Key Factors and Processes	21.71	32.50	22.88	00.00
Skill/Exp. level of resources	58.41	70.00	64.68	72.22
Delivery performance	57.00	70.89	65.47	55.68