

Journey from generic to customized PPM solution

contents

Abstract	2
Study of the Engagement	3
Gap Analysis	3
New Model	4
Data Collection	6
Identifying the Baseline	7
Proof of concept and Pilot implementation	13
Training and roll out.....	13
Further enhancements planned	14
Conclusion.....	14
Abbreviations	15
References	15
About the Authors.....	16

Abstract

Large enterprises with a varied nature of customers, projects and prevalent methodologies typically face the dilemma of having to decide between a generic and customized approach for designing and deploying viable and meaningful Process Performance Model (PPM) solutions across the organization.

Generic PPM solutions, though simpler to achieve and manage with a one-size-fits-all approach, do not promise meaningful solutions for all projects and engagements. On the other hand, a customized approach may require large teams with in-depth expertise on designing PPM solutions or can entail long gestation periods. A more viable alternative for organizations is the blended approach in which generic solutions are designed and deployed at the preliminary stage and then converted into customized solutions based on the insights and best practices gleaned from using the generic solution.

This paper showcases the application of the blended approach in a large engagement. It is a case study that depicts a detailed road map which was used for customizing and creating a holistic PPM for an engagement that would facilitate a necessary trade-off between quality (defects) and cost of quality (review and rework efforts).

Study of the Engagement

As the first step towards creating a customized PPM, a study of the various aspects of the engagement was carried out. While it was already understood that a majority of the work from this engagement was predominantly software development/enhancement in nature, a detailed study would provide critical inputs on aspects that were unique and specific to the engagement.

The main outcomes of the study were as follows:

- The engagement received various types of projects that included development/enhancement, maintenance and porting
- 50 to 60% of the projects were of development/enhancement nature
- The most commonly used technologies were C/C++ & Java/J2EE
- Development projects followed the waterfall methodology
- High quality was expected from the customer as products developed were launched in the market as a part of customer's legacy application.
- Specific targets were given by the customer for defect density (defect to be detected per KLOC) and test case density (test cases to be executed per KLOC on different platforms) and these are distributed across the code review, unit testing and system testing phases
- Code review was a very critical phase with substantial efforts expended in this phase
- The density of test cases executed in this engagement was very high

Gap Analysis

Following the detailed study of the engagement, a gap analysis was conducted against the generic model currently in use by the development projects. The gap analysis was carried out by L&T Infotech's team of experts and with guidance from the Quality Assurance team.

The goal of this analysis was two-fold:

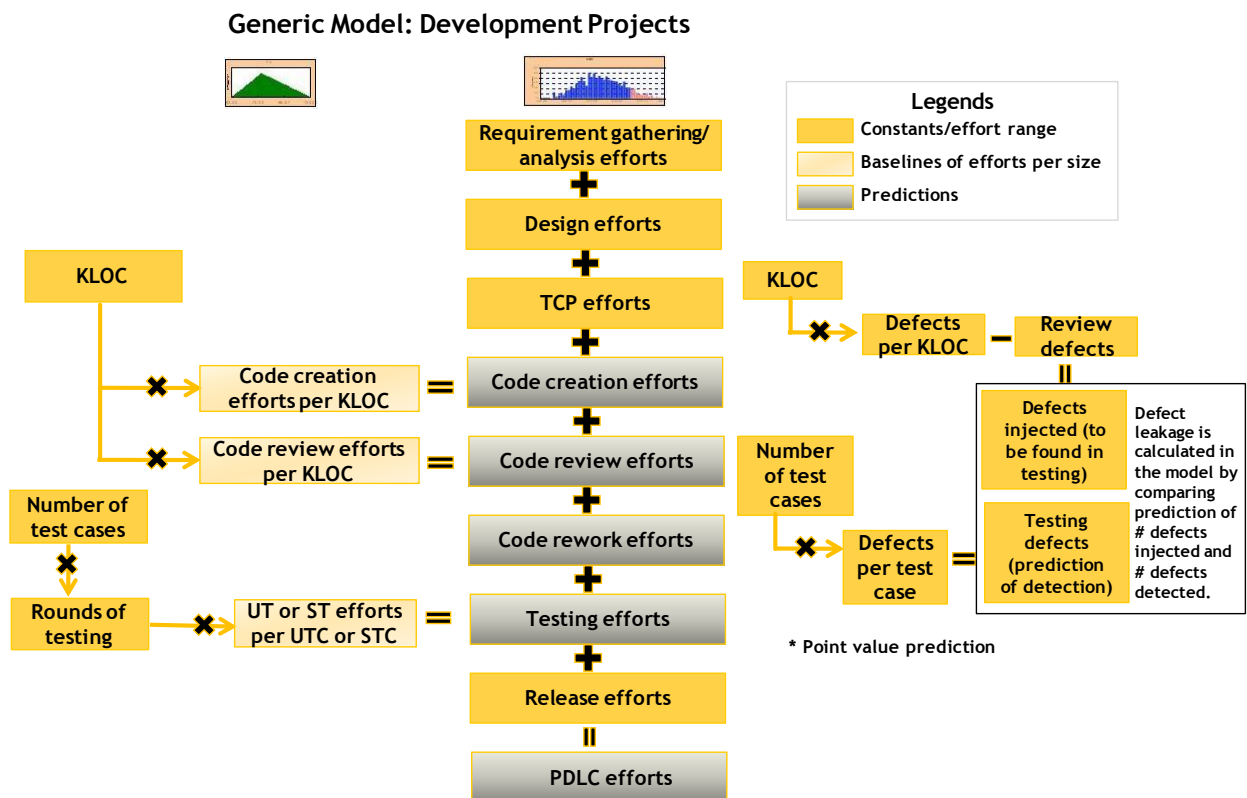
- To study the generic model
- To identify areas that needed further fine-tuning or other gaps that needed to be bridged

A synopsis of key aspects of the existing generic model:

- It provided a level of certainty associated with planned efforts for the development lifecycle with the purpose of identifying risks associated with estimation and effort overrun while executing the project
- It provided lead indicators to project managers to prevent effort variance by improving productivity and re-planning if necessary
- It allowed project managers to compose the testing process (unit and/or system testing) of the project

Process Composition: Project managers can compose the unit and/or system testing process by selecting the optimum number of test cases and rounds of testing based on a trade-off between efforts expended and number of defects detected in the process. It

also provides project managers with the number of defects expected based on the organization's defect density and enables them to study the expected defect leakage.



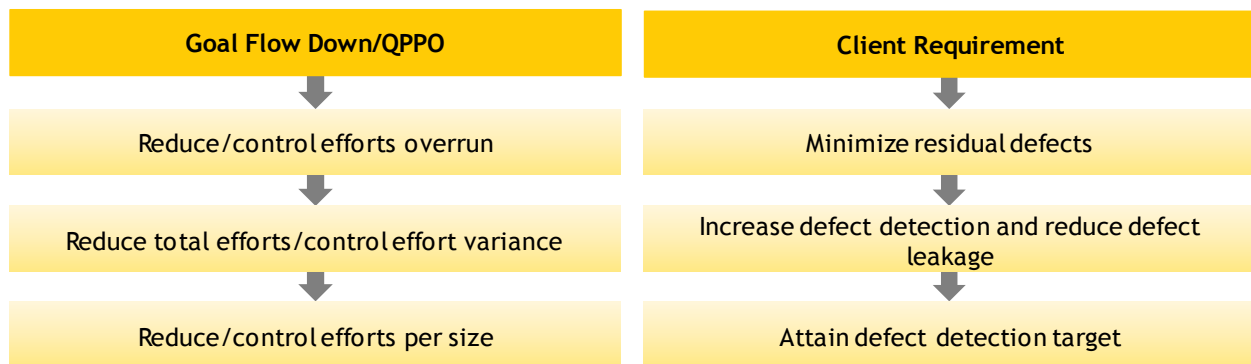
The next step was to identify the gaps and improvements required to suit the engagement specific needs. The analysis highlighted the following salient aspects:

- Project managers had limited flexibility in deciding the rounds of testing as the number of test cases to be executed in each platform was provided by the customer
- Defect prediction of code review was given higher priority. The defect detection targets for code review were higher than the defect detection targets for unit or system testing, and significant efforts were spent in code review
- Providing facility to check the certainty of meeting defect target would be highly beneficial to the engagement
- Based on defect prediction, expected rework prediction would make the effort v/s defect trade-off holistic

New Model

The model was revisited for customization based on the gap analysis conducted.

For this purpose, the organization-level business goal deployment mapping was aligned with customer expectations to arrive at an appropriate engagement level quality and process performance objectives.



The following factors were then analyzed to determine the critical processes.

- Is the process sizable? (practicability in sizing the sub-process and co-relation of size with sub-process performance)
- Impact on Y (percentage of effort expended in process)
- Controllability (extent of control on process)
- Repeatability (feasibility of granular data capture for statistical management)

Based on the above mentioned criteria, coding and testing (unit and system) were selected as the most critical processes. Therefore code creation, code review, unit testing & system testing processes were selected for statistical management.

As per customer requirement, defect detection targets were specified for

- Code review,
- Unit testing &
- System testing

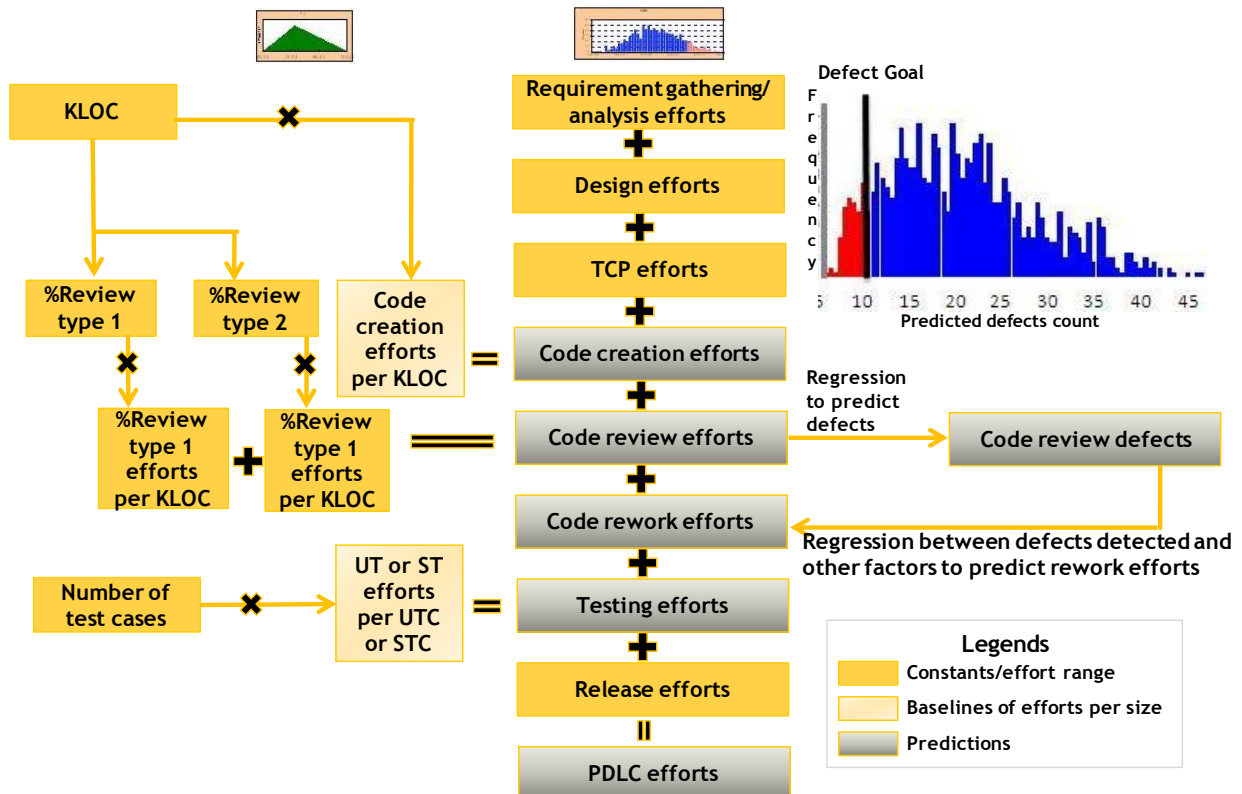
Among the above mentioned sub-processes, there was more emphasis on reducing defect leakage between phases where maximum defects were expected to be detected during code review. In line with this, code review was selected as the critical sub-process for designing the process performance model.

Based on this analysis, an enhanced model was developed.

The highlights of this model were as follows:

- A Quantitative tool to study risk of achieving defect target using the certainty level of defect prediction
- It displays the cost of meeting defect target in terms of review efforts as well as rework efforts
- Allows the project to select an optimum combination of review type in order to detect maximum # defects with minimum efforts
- Combination of Regression and Monte Carlo simulation used for defect prediction and rework predictions

Engagement Specific Model: Development Projects



Data Collection

After conceptualizing the outline of the model, it had to be brought to life. To this end, practitioners from the engagement and the quality assurance group together identified various review types that could be applicable in the engagement. The initial thought process for the PPM was to enable projects to take decisions on choosing the optimum tool/combination of tools from the repository for code review as part of process composition.

Accordingly, data was collected for the following measures:

- Tool name
- Features of the tool used
- Effort required to perform code review using the tool
- Defect detected in the code review when using the tool
- Size of code reviewed using the tool
- Defect goals provided by customer

The following are some of the observations in the collected data:

- Functionalities of tools used were diverse due to which data availability to derive a baseline for tool-functionality combination was limited.

- Generating a wide band baseline also became a challenge as it was very difficult to determine the amount of efforts required to detect maximum defects in the functionality reviewed with help of the tool.
- As the tools were applicable for specific functionalities, deciding the percentage to be reviewed using a particular tool was not feasible.
- Due to the cost of the tool, one project would not use more than one tool at any point in time.
- Usage of tools also needed customer sanction due to the high costs involved. This limited flexibility in selecting the right tool based on model results.

Considering the issues listed above, we also researched other methods of code review execution based on:

- Types of reviews used in the engagement,
- Feasibility of the type of review that can be executed in the projects
- Data availability

The projects completed in the last one year were revisited to understand the type of reviews that followed its completion. Additionally, groups of practitioners were brought together to identify the other review types that could be both applicable and feasible in this engagement. Based on these parameters, two review types were shortlisted, namely, the 'Peer review' and 'Group review'. Data was collected for the actual effort spent in each type of review, the quantum of KLOC reviewed in that type of review and defects detected. During the data collection process, it was observed that most of the projects followed peer review (up to 90%) in the coding phase while the number of projects using the group review remained minimal. This finding combined with detailed discussions with practitioners (projects who have used group review and experts from the engagements), the group review was baselined using the Wide Band Delphi method. Additionally, data collection for defect prediction (elaborated in the next section) was also triggered for (X) parameters that impacted defect detection (Y).

Identifying the Baseline

The next logical step was to create baselines using the data collected to support the model. There were two main aspects that required the creation of baselines:

- I. Effort per size baseline for critical processes (coding and testing)
- II. Regression to predict defects count and rework effort.
- III. In addition to this, effort range is taken for the other phases based on phase applicability, size and criticality of the phase.
- IV. Effort per size baseline for critical processes (coding and testing)

I. Coding baselines

Code Creation Efforts per KLOC: Actual efforts were normalized based on KLOC and, for the purpose of stratification, 'technology' was identified as a criteria based on the engineering

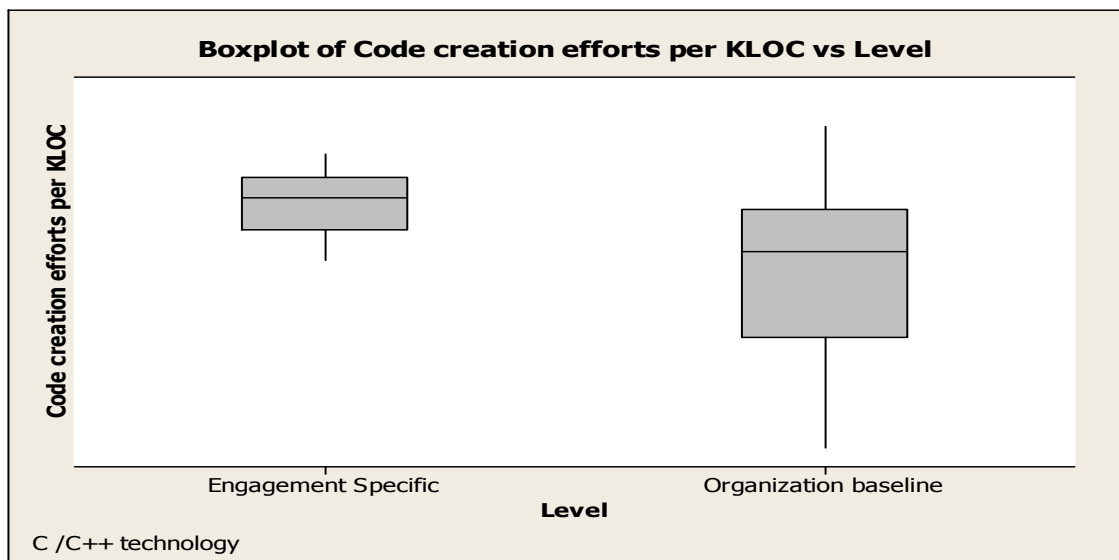
perspective (as technology impacts the KLOC count and associated productivity). The same was also validated using hypothesis test.

Additionally, it was proved that performance of the engagement is significantly different from organization baselines within the same technology, therefore requiring separate engagement specific baselines for code creation.

Kruskal-Wallis Test on Code creation efforts per KLOC(C C++)

Level	N	Median	Ave Rank	Z
Engagement Specific	10	Not disclosed	29.6	2.39
Other projects In the organization	32	Not disclosed	19.0	-2.39
Overall	42		21.5	

H = 5.72 DF = 1 P = 0.017



Based on the availability and stability of data in each technology, baselines were initially created for two major technologies (Java and C/C++). For other minor technologies, it was decided that the projects would start with an organizational or Wide Band Delphi baseline.

Code Review Efforts per KLOC: Similar to code creation, it was observed that the performance of the code review was different in this engagement due to quality expectations (including the high quantum of review). Additionally, defect targets are high in code review due to which the effort spent in projects from this specific engagement is higher than the others.

Kruskal-Wallis Test on Code review efforts per KLOC

Level	N	Median	Ave Rank	Z
Engagement Specific	16	Not disclosed	28.5	5.09
Other projects				
Other projects in the organization	20	Not disclosed	10.5	-5.09
Overall	36		18.5	

H = 25.95 DF = 1 P = 0.000

Based on the data collected (actual effort spent in each type of review and the quantum of KLOC reviewed in that type of review) and sub-processes selected for code review, baselines were derived separately for peer and group review efforts per KLOC.

Testing baselines

For testing, the size selected was ‘the number of test cases’. Execution effort per test case was observed to be technology independent (based on inputs from practitioners and hypothesis test results).

The highlight of this engagement was the quantum of test cases executed, which is quite large due to the product development nature of the projects. However, the execution effort per test case was found to be similar to other development projects (based on statistical tests).

It was thus decided that the following baselines would be used from the organization repository for testing:

- Unit test case execution efforts per unit test case,
- System test case execution efforts per system test case

II. Regressions derived

Regression for predicting Code Review defects

For the purpose of defect prediction, practitioners were asked to identify all the elements that impact the defects detection. Based on brainstorming sessions conducted by practitioners along with the quality assurance team, the following parameters were identified:

- Size of the code to be reviewed
- Type of review
- Review efforts
- Schedule crunch
- Defect target provided by the customer
- Skill level of the coder
- Skill level of the reviewer
- Code review checklist (sufficiency)

The identified parameters were evaluated based on the following criteria:

- Is it measurable? (Y/N)
- Is the definition of this measure standard within the engagement? (Y/N)
- Is data available? (Y/N)
- Is it controllable (by the PM)? (Y/N)
- Is it repeatable (within code review phase)? (Y/N)

Based on the above mentioned criteria, the following was decided:

Factors	Is it measurable? (Y/N)	Unit of measure	Is the definition of this measure standard within the engagement? (Y/N)	Is data available? (Y/N)	Is it controllable (by the PM)? (Y/N)	Is it repeatable (Within code review phase)? (Y/N)
Size of the code reviewed	Y	KLOC	Y	Y	Y	Y
Review efforts	Y	PD	Y	Y	Y	Y
Schedule crunch	Y	span days	N	Y	N	N
Domain experience (Coder)	Y	Years of experience	N	N	N	Y
Technical skill (coder)	Y	Years of experience	N	N	Y	Y
Domain experience (reviewer)	Y	Years of experience	N	N	N	Y
Technical skill (reviewer)	Y	Years of experience	N	N	Y	Y

Factors	Is it measurable? (Y/N)	Unit of measure	Is the definition of this measure standard within the engagement? (Y/N)	Is data available? (Y/N)	Is it controllable (by the PM)? (Y/N)	Is it repeatable (Within code review phase)? (Y/N)
Code review checklist (sufficiency)	N	NA	N	N	Y	Y
Defect target	Y	Defects per KLOC	Y	Y	N	N

Based on the above analysis, parameters selected for initial regression were type of review, size of code reviewed and review efforts. The other parameters were rejected or postponed for the following reasons:

- Defect target was dropped for two reasons; the parameter is not controllable (as the target is decided by the customer) and the VIF factor was very high when included in the regression.
- Schedule crunch and skill level were postponed as the engagement was working on definition and standardization of these factors.
- Code review checklist sufficiency was dropped after discussion with practitioners as it was observed that this factor cannot be measured. Sufficiency is a very subjective term and can always be further enhanced based on project needs.

It was further observed that, the data available for group review was not sufficient to derive a regression. Hence, a Wide band Delphi approach was used.

Defect regression: Peer review

The regression equation is,

$$\# \text{ Defects (peer review)} = 24.5 - 2.19 \text{ KLOC} + 2.01 \text{ Efforts (PD) in Peer review}$$

Predictor	Coef	SE Coef	T	P	VIF
Constant	24.46	31.22	0.78	0.515	
Actual KLOC	-2.188	1.791	-1.22	0.346	1.7
Actual Efforts (PD)	2.0121	0.3894	5.17	0.035	1.7

$$S = 39.8875 \quad R\text{-Sq} = 94.5\% \quad R\text{-Sq (adj)} = 88.9\%$$

Defect regression: Group review

The regression equation is

#defects (group review) = 36.7 - 3.28 KLOC + 3.02 Efforts (PD) in group review

Predictor	Coef	SE Coef	T	P	VIF
Constant	36.69	46.83	0.78	0.515	
Actual KLOC	-3.282	2.686	-1.22	0.346	1.7
Actual Efforts (PD) - group	3.0182	0.5841	5.17	0.035	1.7

S = 59.8313 R-Sq = 94.5% R-Sq (adj) = 88.9%

Regression for predicting Rework efforts

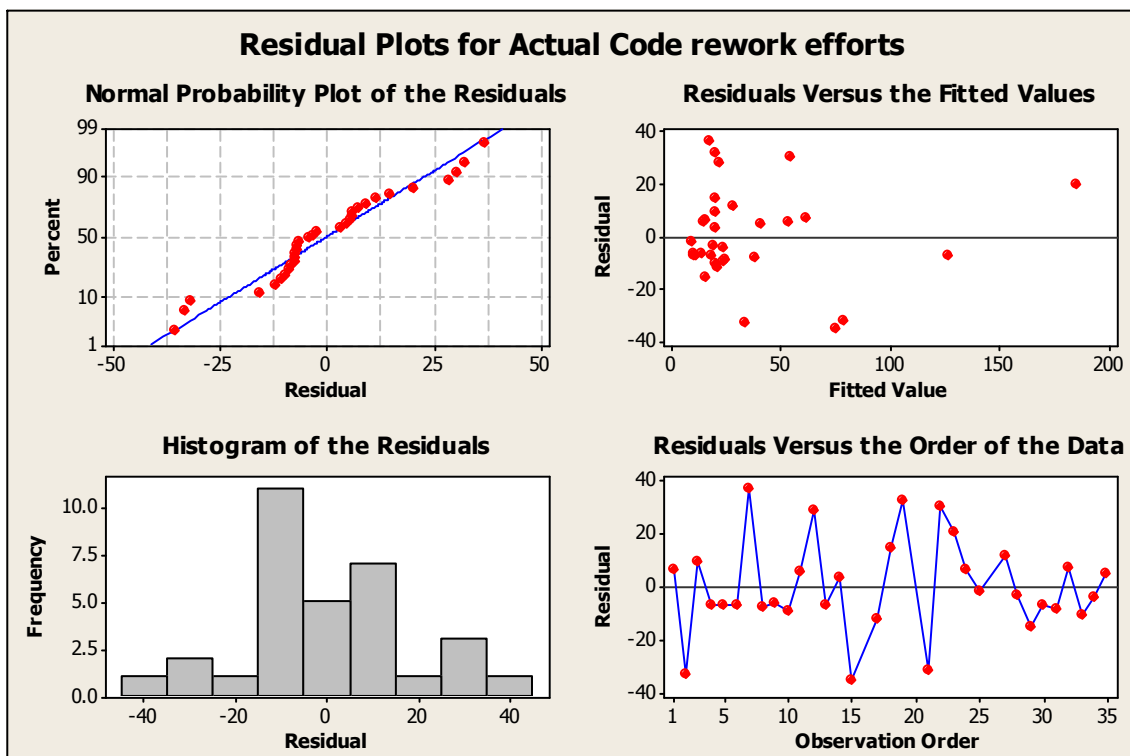
This was another key regression along with review efforts which highlights the cost to ensure quality. Hence it was decided that to start with, a simple regression would be used to predict the rework efforts based on defect count.

The regression equation is:

Actual code rework efforts = 7.94 + 0.0885 coding defects

Predictor	Coef	SE Coef	T	P
Constant	7.944	3.971	2.00	0.055
Coding defects	0.088490	0.007632	11.59	0.000

S = 17.9085 R-Sq = 81.8% R-Sq (adj) = 81.1%



Proof of concept and Pilot implementation

For the purpose of developing the model, the following decisions were taken:

- The initial Proof Of Concept (POC) was created in the crystal ball tool using the organization level license copy.
- It was decided that for training and pilot, the model developed in the crystal ball tool would be used
- After receiving feedback from the pilot, the model would be developed using the excel VBA code.

The POC was showcased to the engagement managers and senior practitioners for explaining the model concept and obtaining buy-in for piloting the same in the engagement.

For the pilot, two projects were selected for implementing the proposed process composition. The live project data was provided as input to run the model. Apart from the pilot result inputs, an open session was conducted with the practitioners for feedback on the following:

- Prediction charts: Which prediction charts will specifically be retained in the excel model?
- Usability: What level of data and results will be displayed for the complete analysis?
- Look and feel: What are the changes in the structure and naming conventions, if any?

Based on these inputs, the model was frozen and implemented.

Training and roll out

Training was planned at two levels:

- Training the trainers: The engagement quality assurance team was involved in building the model and were given a detailed training on the crystal ball tool. In addition, the model was executed on a few older projects to help them understand it and the kind of analysis required. They were trained to provide training to practitioners within the engagement.
- Training the practitioners: The engagement level quality assurance team, in turn, trained the practitioners by providing walkthroughs of the model, by providing open sessions for explaining the concept and handholding the pilot projects in implementing and analyzing the model.

Further enhancements planned

The model will be enhanced continuously as per the optimization process.

The major enhancements planned are as follows:

- Enhancing the defect prediction regression by including other X parameters such as schedule crunch, skill level of developers and reviewers. Currently the operational definition of these parameters is being fine-tuned and parallel data collection has been initiated.
- It has been decided to modify the in-house tool to enhance the data capture for defect, size and effort capture with respect to review types which is currently being done manually
- Currently the model caters to major technologies and groups of projects where defect targets per KLOC are similar.

The reach of the model would be widened by collecting data points from other minor technologies and projects with high defect targets. The data collection options for this purpose are already in place and as soon as sufficient data points are available, the baseline will be derived accordingly.

- Including defect prediction for unit and system testing for a holistic defect prediction model. The model will be enhanced to show interlinking of defects detected from various phases, cost of detecting a defect late in terms of rework efforts and defect leakage. Data collection has already been initiated by collecting the following:
 - Unit/system test case execution efforts
 - Number of defects detected in unit/system testing
 - Number of unit/system test cases executed
 - Percentage of testing done (100%, 150%)
 - Unit/system testing defects fixing efforts

Conclusion

A blended approach towards designing and deploying PPM solutions can prove to be a beneficial approach for large organizations with varied projects and methodologies.

The initial deployment of a generic model provides a sound foundation for anchoring the customized model. Also, usage of the base generic model provides practitioners adequate know-how and exposure in order to contribute towards the design of the customized model. The learning and gaps identified from initial usage of generic model also provides valuable inputs to design and develop the customized model.

Last but not the least, arriving at a customized model ensures buy-in from practitioners by way of providing a meaningful and viable solution that facilitates effective project management.

Abbreviations

Abbreviation	Full form
KLOC	Kilo Lines Of Code
PPM	Process Performance Model
PD	Person Days
POC	Proof Of Concept
Coef	Coefficient
VIF	Variance Inflation Factor
SE	Standard Error
VBA	Visual Basic Applications

References

No external references

About the Authors



Jayashree Nagarajan is a Senior Associate Consultant (Metrics) in L&T Infotech. She is a part of core metrics team at the organization level, with more than 3 years of experience in the field of process definition, metrics definition and organization level metrics analysis, model conceptualization & development and providing high maturity training and consultation across the organization. She has been a core member in defining, developing, training and deploying high maturity practices at the organization level during CMMi v1.2 L5 certification. She holds a Bachelor degree in Statistics and a Masters degree in Computer management from Pune University.



Shobana S has an overall experience of 12 years in IT services out of which 7 years is in the core quality group. She has handled process quality assurance in various business units across different centers in L&T Infotech. Her key role as a Senior Quality Analyst includes ensuring process compliance in projects, process setup, tailoring and project baseline related consultation and facilitation to various types of projects. She has participated in CMMI 1.1 and 1.2 implementation journey in L&T Infotech. She has played key roles in driving process initiatives/ improvements in various accounts and has also participated in organization level initiatives. She holds a Masters degree in Computer Applications from the University of Madras. She is a certified CSQA, ITIL V3 and Internal auditor for ISO 9001:2000 and 20000.



Saraswati Gollerkeri heads the Metrics function at L&T Infotech. She has a rich and diverse experience of over 20 years spanning across software development, project/program management, quality assurance and process consulting. She has led the organization towards high maturity in their CMMI journey and brought about several significant improvements in this area.

Achievement



This paper won 'Best of the Best' CMMI High Maturity Leadership Award 2011 in the 2nd International High Maturity Best Practices Colloquium organized by QAI. Out of the total fifty entries that QAI received for the HMBP 2011 Colloquium, L&T Infotech along with seven others was selected for the final presentation on the 26 August, 2011, in Bengaluru.



About L&T Infotech

Larsen & Toubro Infotech (L&T Infotech), one of the fastest growing IT Services companies, is a part of USD 11.7 billion L&T Group, India's 'Best Managed Company' with presence in the areas of engineering, manufacturing and financial services. It is ranked by NASSCOM as 8th largest Indian software & services exporter from India, is amongst NASSCOM's Top 20 IT-BPO Employers in India (FY2009-10) and is ranked 7th in DATAQUEST-IDC top 20 IT Best Employers Survey 2010. L&T Infotech is differentiated by its unique Business-to-IT Connect, which is a result of its rich corporate heritage.

It offers comprehensive, end-to-end technology solutions and services in the following industry verticals: Banking & Financial Services; Insurance; Energy & Petrochemicals; Manufacturing (Consumer Packaged Goods/Retail, High-tech, Industrial Products, Automotive), and Product Engineering Services (Telecom).

The Company's new emerging verticals include Media & Entertainment, Life Sciences & Healthcare and Government. L&T Infotech also delivers business solutions to customers in the following horizontals/Service Lines: SAP, Oracle, Infrastructure Management Services, Testing, Consulting and Domain Services; while its other service offerings are: Business Intelligence/Data Warehousing, Legacy Modernization, Applications Outsourcing, Architecture Consulting, Enterprise Integration, Service Oriented Architecture, Systems Integration and PLM.

L&T Infotech also offers Unitrax®; a SaaS-based transfer agency product, enabling fund managers to process product types such as mutual funds, fixed income, structured notes, annuities, segregated and retail venture funds. Unitrax® provides web-based BI reporting, sales analytics and CRM features. Unitrax®'s Web services-based architecture eases inter-connectivity to in-house applications. With a consistent high customer satisfaction measure, the product improves efficiencies through streamlined and automated customer processing and administrative features whilst ensuring a high level of compliance with regulatory requirements and industry standards.

For more information, visit us at www.Lntinfotech.com or email us at info@Lntinfotech.com.

Follow L&T Infotech on:     