International Journal of u- and e- Service, Science and Technology
Vol.7, No.2 (2014), pp.41-50
http://dx.doi.org/10.14257/ijunesst.2014.7.2.04

Integrated Solution for Timely Delivery of Customer Change Requests: A Case Study of Using DevOps Approach

Yuhong Liu[1], Chengbo Li[2] and Wei Liu[3]

[1]*IBM Canada Lab, Department of Mechanical and Industrial Engineering, University of Toronto*
[2]*IBM Canada Lab, Division of Engineering Science, University of Toronto*
[3]*Shanghai Maritime University, Shanghai, China*
[1]*yuhongl@ca.ibm.com,* [2]*lucyli@ca.ibm.com,* [3]*weiliu@shmtu.edu.cn*

Abstract

Numerous past studies have indicated a strong linkage between customer satisfaction and business performance exhibited through timely responses and delivery of customer requests. As the enterprise software industry is shifting to capabilities such as timely delivery of new features, fast delivery of customization requests, and close collaboration between customers and software development (including DevOps), this study examined the usage of a software tool to accelerate the responses to customer change requests within a product development organization. This paper analyzes a comprehensive case study (from conception to deployment) of a prototype created by IBM Rational Software AM release team to collect, analyze, monitor the flow of customer change requests for a set of products. The results of this work have led to a more accessible, frequent and informative tracking and reporting system that has significantly reduced the response delivery time for customers. The implication of this study could be applied to many software development companies to improve customer responses and promote open development and DevOps processes. This study is only limited to one firm from a particular industry sector, but nonetheless it offers future exploration opportunities for researchers and practitioners to better utilize currently available resources within their organization to increase customer satisfaction. This study also suggests a cost-effective way to facilitate collaboration between development and project management teams.

Keywords: *Customer change request; DevOps; Integrated Solution; IBM Rational Software; ClearQuest*

1. Introduction

To maintain a remarkable growth in revenue and profit in today's highly competitive markets, immense amount of effort has been put on marketing and management researches to find the key drivers of good financial performance. Various studies in the past years explored and concluded a strong linkage between customer satisfaction and repurchase intentions that led to significant financial improvements [1]. According to a study surveying worldwide multinational corporation CEOs, customer loyalty and retentions was one of the top three major challenges facing current organizations [2]. Research conducted by IBM Institute for Business Value (IBV) also identified that 75% of companies expect customer experience to be a key differentiator in the current market [3]. This is particularly important in Business-to-Business (B2B) service where customer or client relationships are often long-term and capital-

ISSN: 2005-4246 IJUNESST

intensive. Hence, firms that emphasize on client satisfactions will likely to gain a competitive advantage over its rivals.

The capability to deliver customized bundles of products and services is vital for a high customer satisfaction in client-focused firms [4]. Changes proposed by the customer as a new requirement for future product and service releases are referred to as change requests (CR). Ability to process change request in an effective and timely manner is one of the key quality indicator and requirements during software maintenance process [5]. However, customization inevitably reduces the efficiency of the supply delivery chain and places emphasis on manual adjustments to the traditional routine process. Manual operations also make the process more vulnerable to errors, break downs, and delays. More than 21% of interrupts in change management are caused by human in IT environments [6]. As such, many firms continue to allocate a substantial amount of monetary and human resources in measuring customer feedbacks and fulfilling customer requests. Firms continues to raise the amount of emphasis on facilitating and strategizing service unobstructed delivery to customer's requests.

It is evident that product customization for customer satisfaction is a capital intensive and difficult task. This is particularly true for large corporations that have a rigid and well-defined infrastructure, as well as a wide range of products. Even though such structure allows for robust delivery, but it compromises on the response time to crucial business changes and growing requirements from customers. This makes the firm unable to dynamically deliver functionally rich products or services that match the customers' needs. Hence, an agile delivery system is essential for firms to remain competitive [7].

The complex and dynamic environment organizations face today has a significant impact on their structures. The DevOps approach is a 'one-team approach' that focuses on the collaboration between two major departments that are responsible for customer request delivery – the development team and the operations team. The development team include software developers, testers, and quality assurance personnels. The operations team includes system administrators, database administrators, network technicians, as well as the other roles that assist the with the production infrastructures. DevOps describes approaches that focuses on lean software delivery processes and improve cycle time of software. Traditional departments put an emphasis on each department based on division of labor. Development team strives for change and upgrades, while operations team strives for stability, this caused the two team to have isolated goals and therefore difficult for intensive collaboration. There are many case studies for optimization methods for individual department optimization (*i. e.*, Agile software delivery methodology for developers), but less so for optimizing process for the whole [12].

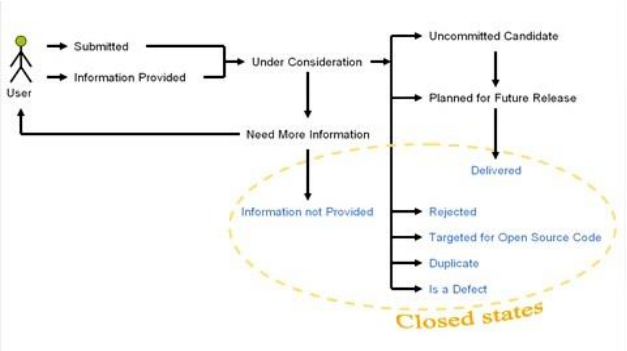

Figure 1. RFE Stateflow Diagram. Once Customer Submitted their RFEs, RFEs Went through Series of Approvals and Decisions are made as a Conjunction of Product and Development Team

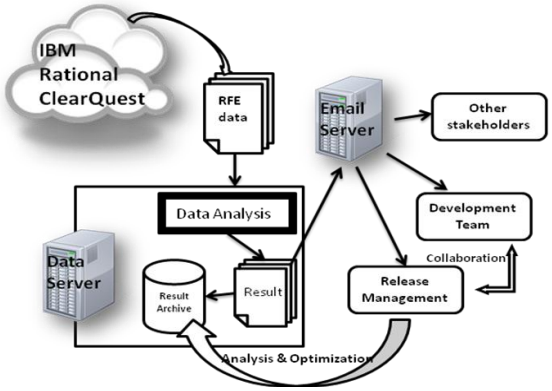

Figure 2. An overview of Tool's High Level Architecture and Process. Products' RFE Information Passes Series Analysis and Consolidation before being Distributed to Different Teams

DevOps emphasizes the collaboration across the value chains which allow for reduced waste times and accelerated delivery of product solutions. This revolutionary concept, integrates traditionally separated departments, allows rapid and frequency delivery of products and services for large enterprises that have traditionally separated department infrastructures. In order to use DevOps approach for continuous delivery to customers, companies will need to address several challenges ahead of them. Just to name a few:

- How to improve efficiency within the organization to provide more timely response to customers?
- How to achieve more effective and informative measurement of team's performance?
- How to encourage cross-functional team collaborations?
- How to utilize current available resources within the organization for new initiatives?

The validation of the DevOps approach is illustrated with a working prototype in this paper. The product leverages existing development and operations infrastructures which lead to a solution delivery system that reveals unprecedented delivery time. This automatic solution greatly decreases human errors and allows for rapid cross-departmental issue processing. This integration allows inexpensive yet effective facilitation for solution delivery to the customers and presents a creative platform for future projects using DevOps principle.

2. Case Study-An IBM Success Story

A. IBM RFE Community

Requirement for Enhancement, or RFE, is a community website that IBM used to allow and facilitate IBM software users to submit improvement request for products or services they purchased. Following submission, RFEs are analyzed by product managers in conjunction with software engineers to be considered as new features in future releases. The goal of the RFE community platform is to provide the ability for users to participate in software development cycle and it is one of the many types of change requests available to IBM clients [9]

There are many features associated with the RFE platform. Customers could vote for their favorite RFEs, give feedback to others and interact with product development team Because of this active engagement and interaction from clients; RFEs have a strong influence on IBM software development.

Once RFEs are submitted on RFE community, they are automatically synchronize to internal development server and database, and visible to its appropriate product development teams as seen in Figure 1. In this case study, we will be discussing a success story between product release and development team of IBM Rational Software using appropriate technology and integrated software to achieve more timely response delivery.

B. RFE Management in IBM Rational Software

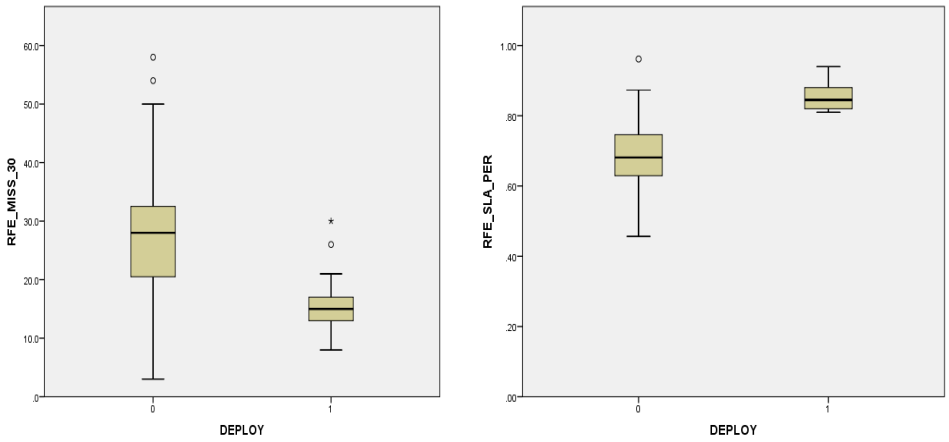

Figure 3. Box Plot Diagrams for Variable RFEs Missed 30 Day Threshold and RFE SLA Percentage, a Comparison Before and After the Deployment of RFE Tool. There is a Clear Shift of Mean Values Based on Plots

Products and services under IBM Rational Software provide guidance and enhanced efficiency for various stages of software development and delivery, including project management, quality management, and lifecycle management. The line of products covers an entire software lifecycle from initial project design to final implementation [10].

RFEs and other change requests are managed through IBM Rational Software ClearQuest, an industry-leading change request management system. The system automatically synchronizes itself with the external platform, and updates incoming RFEs into databases. Historically, this development team within IBM Rational Software was faced with the challenges of providing timely response to RFEs. The development teams in IBM Rational Software are required to provide response to any newly arrived RFEs within 30 days. Release management teams oversee product teams' RFE response status and report to higher management through weekly reports.

The release team uses following metrics and algorithm to calculate development team's RFE status:

$$\text{30 day SLA} = \frac{\text{RFEs Arrivals last 30 days} + \text{RFEs Need info and > 30 days old}}{\text{RFEs Arrivals last 30 days} + \text{RFEs in 30 day queue and > 30 days old and}} \%$$

85% is used as a threshold for product teams. Any product with 30 day service level agreement (SLA) percentage above 85% is considered as in good standing. Otherwise, team will be told to start triaging their RFE immediately.

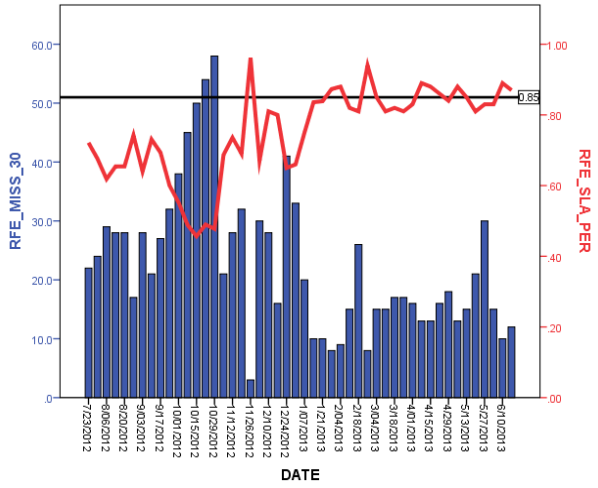

Figure 4. A Combined View of RFE SLA Percentage and RFEs missed 30 Day Over 10 Month Period

C. Deficiencies in Current RFE Management

There are several reasons that development team is struggling to provide timely responses. Firstly, project managers were not notified when there are new RFEs opened against their product. The lack of notice makes the project managers unable to stay updated and monitor team's progresses. Secondly, once RFEs are opened, they are buried in a pool with other types of change requests. There is no clear indication or warning that a particular RFE is going to miss its 30 day threshold in the change request system. In order to know their team's RFE status, project managers have to actively run their RFE queries. Because large amounts of efforts are continuously needed on a daily basis to monitor RFE progress, it is extremely easy for project managers to be distracted by other tasks with higher priority. As a consequence, when RFE report is generated and submitted to upper management during review sessions, project managers are usually caught off guard to see their RFE status is not meeting the expectations.

To further worsen the situation, RFE status reports are generated semi-manually by release managers to be reviewed by upper management. Due to the manual component of report generation, the report cannot be generated on demand any time by anyone. Even though the release manager agreed to generate a report through a special request by project manager, project manager will still have to wait for at least a half day. Hence, it's not difficult to spot the inefficiency in reporting and status update process for development and release management. In addition to the inefficiency, the report only provides service level agreement calculation and RFE status; it cannot be drilled down and provide detail information about RFEs.

D. A Rapid Prototype with DevOps in Mind

After analyzing development team's current RFE management process painstakingly, release team realized that an automated and scalable reporting tool that integrates available resources would greatly help development team to better manage their RFE process. With

DevOps and lean startup thinking in mind, the release team started by listing out problems and challenges faced by the development team through survey emails and meeting discussions. Through these communications are aimed to discover the following:

- What is the management currently doing in facilitating RFE delivery?
- What are the current obstacles that prevented the developers to attend the RFEs?
- What feedback systems are available for the current progress?
- What features should the reporting tool have?

Once these problems are identified, the release team listed out a list of attributes that will be beneficial to aid in the RFE delivery process. As emails is the most preferred method of communication for this process, the release team decided that a more regular, informative, personalized, direct-impact and up-to-date email alert system will encourage the development team to promptly process RFEs.

After the design requirements were created, the release team quickly began to glean available resources such as possible location to host the reporting tool, available software licenses within the department, or past automation projects related to process management, etc. The recreation of the RFE process is carried out through partial automation. Surveys and questionnaires continued while the team was prototyping the reporting tool to allow modifications, more suggestions were collected from tool's stakeholders.

The reporting tool was designed such that each product's RFE data is pulled from the system through running queries in IBM Rational Software ClearQuest client on a Virtual Machine. These products' RFE data files are stored, and later it is analyzed by a java program and a result file is generated. Another script achieves the result file and transfers it to the email server. The email server runs a separate script and attaches the result file in an email to interested stakeholders. RFE status is recorded and store in a back-end database for regular monitoring and further analysis. This entire automated process happens in every morning of a regular business day. The architecture and process flow of the tool can be seen in Figure 3.

The first portion of the email sent to stakeholders includes an update of product's RFE status. In the second portion, the email includes a set of specific information about the product, such as RFEs arrived, RFEs are in 30 day query, or RFEs missed 30 day mark, *etc*. If actions are required from the product's development team, email will have a warning message and indicate how many RFEs are required to triage if development team want to maintain a good standing RFE status. Moreover, it provides RFE URLs in IBM Rational Software ClearQuest for those RFEs missed 30 day mark. The reporting tool was deployed in early February 2013.

Table 1. Description Statistics Associated with the RFE for RFEs Delivery

		RFE_MISS_30			RFE_SLA_PER		
		A	B	Total	A	B	Total
N		28	20	48	28	20	48
Mean		27.89	15.70	22.81	0.68	0.85	0.75
Std. Deviation		13.38	5.24	12.28	0.12	0.04	0.13
Std. Error		2.53	1.17	1.78	0.02	0.01	0.02
95% Confidence Interval for Mean	Lower Bound	22.71	13.24	19.25	0.64	0.83	0.72
	Upper Bound	33.08	18.15	26.38	0.73	0.87	0.79
Minimum		3.0	8.0	3.0	0.46	0.81	0.46
Maximum		58.0	30.0	58.0	0.96	0.94	0.96

Table 2. Analysis of Variance Associated with RFE Processing

		Sum of Squares	df	Mean Square	F	Sig.
RFE MISS_30	Between Groups	1734.434	1	1734.434	14.899	.000
	Within Groups	5354.879	46	116.410		
	Total	7089.313	47			
RFE SLA_PER	Between Groups	.321	1	.321	34.153	.000
	Within Groups	.432	46	.009		
	Total	.752	47			

E. Deployment Result

Table 3. Paired Sample Test Associated with RFE Processing

		Paired Differences					t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower	Upper			
Pair 1	RFE_MISS30_BE - RFE_MISS30_AF	15.05000	13.77822	3.08090	8.60159	21.49841	4.885	19	.000
Pair 2	RFE_SLA_PER_BE - RFE_SLA_PER_AF	-.20300	.12457	.02785	-.26130	-.14470	-7.288	19	.000

As mentioned in previous section, the reporting tool records development team's performance automatically in a database which later on being analyzed. There are two types of data were recorded and analyzed in this study; number of RFEs missed 30 day threshold and development team's RFE SLA percentage. Figure 4 is a two scale plot to illustrate both RFE SLA percentage and number of RFEs missed 30 day threshold of development team over a 10 month period. The pattern of the plot indicates a negative correlation between RFE SLA percentage and number of RFEs missed 30 day threshold as anticipated. In addition, once the prototype was deployed (early February 2013), there was an immediate increase in RFE SLA percentage and a reduction in number of RFEs missed 30 day threshold. Moreover, before February 2013, it is hard to see a pattern for RFE SLA percentage of development teams and

the percentage fluctuates; however, once the RFE reporting tool is deployed, there is a clear pattern that the fluctuation of teams' RFE SLA percentage abates and the mean of percentages are shifted to be around 85%.

In order to analyze the improvement of development team's performance and stability after the deployment, data were divided into two groups, before deployment (A), and after development (B). Analysis showed strong evidences that development team had improved their performance with assistance of the reporting tool. Data spread of RFE SLA percentage and number of RFEs missed 30 day threshold are presented in Figure 3. As it can be seen, Group B showed smaller spread and lower average value for RFE missing 30 days SLA value. This is a clear indication of the RFE tool has reduced the number of RFEs that missed target. Secondly, Group B showed a higher RFE SLA percentage value. This is to be expected as the metric is inversely related the number of RFEs missing the target. Descriptive statics associated with the two groups are illustrated in Table 1 further affirmed our claim. For number of RFEs missed 30 days threshold (RFE_MISS_30), mean of RFE_MISS_30 before deployment is 27.89 RFEs missed 30 day, however once the prototype is deployed, mean dropped to 15.7 RFEs. Same-wise for RFE SLA percentage (RFE_SLA_PER), before the deployment, team was staying at an average of 68%, but after the deployment, team improved to 85%. In Table 2, paired sample tests showed strong statistical evidence that there were differences in team's performance in both area.

Likewise, development team also managed to improve their stability as well. As seen in Table 1, major reductions in standard deviation for both RFE SLA percentage and RFEs missed 30 day threshold. Statistical analysis indicates a nearly 71% and 61% decrease in standard deviation for RFE SLA percentages and number of RFEs missed 30 day threshold. ANOVA table in Table 2 is strong proof that team improved their stability in RFE management.

3. Discussion

Based on the result and data collected before and after deployment of the reporting tool, the release team in the case study clearly is a forerunner on adopting automation methodologies into day to day development operations to possibly achieve DevOps. They have demonstrated the value of early-stage DevOps through integration of software, continuous automation, and cross team collaboration. In the case study, team was able to automate data collection and status calculation process, therefore able to achieve a real-time automated reporting tool using standardized measuring metrics. Because of this real-time automated reporting tool, project managers were able to stay updated of their product's RFE status through the reporting tool on demand. However, the success of automation and implementation is only the first step of DevOps approach. Continuous monitoring and optimization will be the key to success in order to achieve a long-term commitment of timely response delivery. Team should utilize the performance data they collected by going through proper analysis to optimize their current system, ultimately achieving the goal of providing an ever more engaging customer experience and agile development lifecycle to accommodate any type of change request.

As B2B software industry becomes ever more competitive, many organizations believe that the key to success is the capability to leverage development and delivery towards a more agile and continuous software lifecycle. Companies are desperately trying to close the gap between what customers expect from the product and what their development team is capable of delivering. Competition will eventually become the battle between organizations' capability to rapidly transform business ideas or customer feedback into customizable, scalable and reusable product prototype using their available resources. Although DevOps

originated from grass-root web application companies seeking more rapid and frequent deliveries to satisfy their customers, there is no doubt that DevOps is also the proper approach for bigger organization in the industry to stay competitive [8].

4. Future Investigations

The rapid prototype provided in the case study is a movement towards the DevOps approach – addressing the gaps between software development groups and the operations teams. As these traditionally distinctive teams have dissimilar goals and incentives, this prototype provide a platform for collaboration in a holistic way. The future potential to what this prototype withheld is promising.

The essence of this prototype is within its agile capabilities to handle RFE delivery. There are many other types of customer engaging activities within IBM Enterprise that follows a similar clearance process, such as defect resolution and customization bundling requests. The backend infrastructure, such as databases and reports, are also similar with minor modifications. Hence, this tooling can be vastly adapted horizontally to other customer requests processes that involves both development and operations team.

Furthermore, this prototype demonstrated that there is a great need for continuous adoption of the DevOps approach. By bridging the communication methods between teams, we have leveraged the available resources and investment for speed delivery of RFE processes. For multinational firms, many processes involve much more than just developments and operations teams and the DevOps approach should not be limited to only 'Dev' and 'Ops'. Through expansion in functionality and capability, the prototype also creates a foundation for collaboration for more stakeholders that are engaged in the entire product or service delivery process.

Finally, this prototype highlights important aspects of collaboration tooling and automation which will be useful in future designs similar in nature. These include:

- Continuous interactions between all stakeholders allowing for speed delivery
- Informative feedback system allowing for real-time monitoring of current progress
- Transparency within the system allowing for continuous verification and optimization

5. Conclusion

It can be seen from the tooling example that many aspects within the release management and software lifecycle management of many organizations need to be improved in order to become competitive DevOps organizations and achieve timely response delivery of customer change requests.

Acknowledgment

The authors gratefully acknowledge the mentorship and guidance of Adam Borg, Ramzan Khuwaja, and Diana Lau throughout the process. The authors also want to extend their appreciation towards members of Rational AM release team and RFE community of IBM Enterprise. Without their time and expertise, this work would not have been possible. Finally, the authors thank their friends and families for their continuous encouragement and support.

References

[1] M. S. K. E. Naumann and P. Williams, "Identifying the Key Drivers of Customer Satisfaction and Repurchase Intentions: An Empirical Investigation of Japanese B2B Services", Journal of Consumer Satisfaction, vol. 25, **(2012)**, pp. 159.

[2] M. Hoots, "Customer relationship management for facility managers", Journal of Facilities Management, vol. 3, no. 4, **(2005)**, pp. 346-361.
[3] IBM, "DevOps: The IBM Approach", **(2013)**.
[4] J. Heikkilä, "From supply to demand chain management: efficiency and customer satisfaction", Journal of Operations Management, vol. 20, no. 6, **(2002)**, pp. 747-767.
[5] M. N. A. Rahman and S. Suhailan, "Managing Software Change Request Process: Temporal Data", International Journal of Computer Science and Security, vol. 3, no. 3, **(2009)**, pp. 201-209.
[6] L. Shwartz, D. Rosu and D. Loewenstern, "Quality of IT service delivery - Analysis and framework for human error prevention", IEEE International Conference on Service-Oriented Computing and Applications (SOCA), Hawthorne, **(2010)**.
[7] P. Chhabra and S. Karamongikar, "Service Chain Management", The Agile Delivery of Service Chain Management Solutions, **(2008)**, pp. 215-224.
[8] Enterprise Management Associates (EMA), "DevOps for a New Millennium: A Lifecycle Perspective Supporting Business Growth in an Altered Economy", Enterprise Management Associates, Research Report, **(2013)**.
[9] S. Laningham, "Rational RFE Community", [Online] http://www.ibm.com/developerworks/podcasts/demos/special-RFE-process-1/cm-int-special-RFE-process-1.html, **(2011)** December.
[10] IBM, "IBM Rational Software", [Online]. http://www-01.ibm.com/software/ca/en/rational/, **(2013)** June.
[11] A. R. M. Nordin and S. Suhailan, "Managing Software Change Request Process: Temporal Data Approach", International Journal of Computer Science and Security, vol. 3, no. 3, pp. 201.
[12] M. Hüttermann, "DevOps for developers", Apress, **(2012)**.

