Cost Estimation and Metrics
Final Project

Evaluating the effectiveness of checklist for successful software deployment

By:

Team 4:
Sonali Thaker
Dhruvil Parikh
Kyle Aldrich

Fall 2016
Contents

1. Summary
---------------------------------
---------------------------------
-----
3

2. Introduction
---------------------------------
---------------------------------
-----
3

  2.1 Intro
---------------------------------
---------------------------------
-----
3

  2.2 Problem
---------------------------------
---------------------------------
-----
4

  2.3 Solution
---------------------------------
---------------------------------
-----
4

3. Approach / Methodology
---------------------------------
---------------------------------
-----
5

4. Proposed Metrics
---------------------------------
---------------------------------
-----
7

  4.1 GQM Model
---------------------------------
---------------------------------
-----
7

  4.2 Defined Checklist
---------------------------------
---------------------------------
-----
7

5. Results and Discussion
---------------------------------
---------------------------------
-----
10

  5.1 Online Checklist 1
---------------------------------
---------------------------------
-----
11

  5.2 Online Checklist 2
---------------------------------
---------------------------------
-----
13

  5.3 Defined Checklist
---------------------------------
---------------------------------
-----
16

6. Conclusion
---------------------------------
---------------------------------
-----
17

  6.1 Limitations and Reflections
---------------------------------
---------------------------------
-----
17

  6.2 Conclusion
---------------------------------
---------------------------------
-----
17

7. References
---------------------------------
---------------------------------
-----
18

8. Appendices
---------------------------------
---------------------------------
-----
18
1. Summary

Our team proposes to introduce a unique software checklist that mitigates the risks associated with software deployment. We hope to display several metrics in our developed checklist that not only distinguishes it from other checklist but also minimizes delays in software deployment. Online software deployment checklists will be referenced and used to compare against our own deployment checklist. To display the effectiveness of our checklist, we will use function point analysis to see the difference in staff/months or effort. This documentation serves as the structure of our project as a whole. It will contain an introduction, which explains our understanding of software deployment and the risks associated with that process. Proposed metrics will have a complete and newly developed software checklist. Within that checklist will be derived GQM. Next a detailed discussion will arise from the analysis of our checklist against certain projects. Comparative results will also be given to show the strength of our checklist. Finally, conclusions and inferences will be made from our detailed analysis and results. Like with all research, there are limitations which need to discussed as well. But by the end of this project, our readers should minimally have strong knowledge of software deployment stages and underlying importance of software deployment in the development process.

2. Introduction

2.1 Intro

The deployment process might be one of the most undervalued or underestimated parts of the software process. It has been discussed amongst developers that despite how promptly and under/ on budget the development process is going, developers continually miscalculate or misproject deployment. These inconveniences have caused much frustration around the software community. This problem became the motivation of our project, and we hope to provide a solution. One problem that we have encountered through our research is the lack of focus and time dedicated to software deployment. The other and more important factor to this problem is the software deployment checklist. Our solution is to design a checklist the mitigates the risks associated with software deployment and refocus the attention of software project members on deployment.
2.2 Problem

While we think the criteria for software checklists is inaccurate, it was not from a lack of effort. One of the positives of our research was the abundance of checklists available (check appendix). There are numerous deployment checklists on the web and some of which have been provided in this project. Another positive associated with current checklists are focus on testing. Some checklists provide many criteria on the various type of testing, such as unit, integration or system. We think the success of testing is also an important aspect of software deployment.

Unfortunately, despite finding some positive aspects, we were able to identify many negatives as well. One of the main problems is the variety of criteria amongst each deployment checklist. From a reader’s standpoint, you do not see many common themes. This suggests that software teams do not know exactly what is important to deploy successfully. This variety can also lead to poor decision-making from developers. The reason for this is because they will pick the checklist that they think is best suited to quick deployment, which ultimately leads to an unsatisfactory product. Our goal is to speed up the process but also produce a quality product. Another unfavorable aspect of current checklist is the vagueness of information in the criteria. Each criterion in many of our researched checklists had a lack of quantitative information to uniquely identify if that criteria was satisfied. It raises the question how do you know is something is complete is you have nothing to measure it against?

Ultimately, if we can solve some of the questions raised by the current deployment process, it should be able to divert the focus of teams to deployment.

2.3 Solution

In order to solve these problems, our group is going to design a unique software checklist. This checklist will be broken down in three aspects of the software deployment process. The first aspect is preparation for software deployment. This phase usually involves the gathering of resources, material, and a little bit of pre-planning for the second aspect. The second phase is during software deployment. A significant amount of testing goes on during this phase and must constantly be tracked and updated. The final stage of software deployment and therefore our checklist is after software deployment. Typically, after deployment, teams focus on how the software is reacting to the environment and also on user satisfaction. These tasks are closely monitored because if errors occur they must be fixed as soon as possible.

Within each of these phases, the checklist will have several criteria. These criteria will be developed from our GQM analysis. During our GQM process, we were able to work with a UPS intermediate developer and research the web. This provided the opportunity to hone in on aspects of the deployment process that were universally accepted. As we move into further detail of our project, we hope our analysis and results will speak for themselves.
3. Approach / Methodology

The main goal of the project is to clearly define a checklist that will lead to successful software deployment. Currently there are three major problems with checklist such as Engineers and Developers consistently misproject the time to deployment, lack of quantitative information and hope to create proper criteria. Based on our GQM analysis example to ensure whether the checklist has solid approach for development, we followed up with our Sr Application Developer with some questions. Below is the conversation that took place between Developer and Me:

**Interview conversation with UPS Developer:**

**Me:** Do you usually face the problem of maintaining different versions of code?
**Developer:** Yes. All the time as we have different versions of code releasing every 5 to 6 weeks based on the new changes and enhancements. Due to many versions of code, we face the issue at the time of deployment.
**Me:** Do you prepare yourself in advance for the deployment?
**Developer:** We try but not everything gets ready by that time.
**Me:** So do you follow any checklist prior to deployment?
**Developer:** No. We just start and end our deployment at the same time.
**Me:** How long is your deployment time usually?
**Developer:** It depends on the application size. I would approximate to at least 4 to 5 hours on normal application size.
**Me:** At the time of deployment how to know which code to deploy?
**Developer:** We create folders with the versions name such as 12.1.0 and then increment by 1 such as 12.1.1 and so on. During deployment we take the last version folder and deploy.
**Me:** But what happens when other developer has checked in the code to different folder rather than actual one going for deployment?
**Developer:** aha.! We always face that issue. That someone has put some changes or deleted some from different version and not knowing which folder going for deployment. Because of that we deployment gets more problems and it increases our deployment time.
**Me:** Understood.! Do you think a checklist can help you save some more time?
**Developer:** I think so; it can be very useful.
Environment Setup is always the crucial part in the software development as it helps the people to test the code in different situations for better software productivity. However, there are always some issues in setting up the environments. This happens because of not understanding the importance of testing in various situations, creating fake data in all environments, deploying and testing. As a developer, I have seen some issues on my last project because of not proper environment setup. To do some more analysis, I interviewed my “Intermediate Application Developer” to ask about his opinion.

**Interview Conversation with Intermediate Application Developer:**

**Me:** Do you think that we need different development environments for all the software projects?

**Int.App Developer:** Yes, it is very crucial to setup different development environments for any project. According to my experience, I have seen many software that doesn’t function properly due to various user inputs which are not handled during the coding. However, this kind of errors can be easily captured while testing in various environments.

**Me:** Can you please share your experience of the project that had no or minimum development environment setup?

**Int.App Developer:** At my first job as a software developer in 2008, I started working on a JavaScript application to build front end of a website that helped people incorporate a new business. I was responsible to transfer the front end data provided by the user and pass it to the back end business logic and respond the user with appropriate message. To achieve this functionality, I was working with ajax calls and jQuery. After completing the code, there was only local development environment setup to test the code. The other environment was directly for the production environment. During this project we faced lot of problems with the data, testing and environment issues.

**Me:** What problems did you face due to minimum environment setup?

**Int. App Developer:** The major problems occurred due to data not validated properly from the front end. As a developer, I wrote the code for the front end side based on the requirements and tested locally with unit test. However, there were few special cases which were not covered while testing locally. Later, the issues were discovered by the real users in the production environment. This can be avoided by setting up different development environments to test the code.
4. Proposed Metrics

4.1 GQM Model:

Fig 1 represents a simple GQM structure for our project. The main goal of our project is to create and evaluate checklist for software deployment based on its effectiveness and usefulness. The metrics involve creating our own checklist using various online forums, publications and online sites. The metrics also involves comparing rating results with existing checklist as well as keeping track off on time software release.

![GQM Model for the checklist evaluation](image)

4.2 Defined Checklist

After conducting interviews with various software developers, researching online and based on our team member’s experience we have defined the set of points that needs to be verify before, during and after software deployment. The checklist that we have defined is very general and high level overview of what each developer should consider for software deployment for any type of system. Refer to Table 1 for defined checklist for successful software deployment.
<table>
<thead>
<tr>
<th>Phase</th>
<th>No</th>
<th>Checklist</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHASE I - Preparation for Software Deployment</td>
<td>1</td>
<td>Is the latest version of code being taken for deployment?</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Verify the version string for the build or code taken for deployment</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Check for configuration files and folders</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Make sure all Environment setup is ready before the deployment</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Remove all Development or unit testing code from software</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>Ensure to request servers and databases as per the system requirement</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>The test cases documented in the Acceptance Test Plan have been executed</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>Check that all defects on current defect list has been resolved</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>Is your application using any hardware and software combination? If Yes, please ensure you have hardware configured during environment setup</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>Verify that Regression testing is successfully completed</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>Consistency check - SRS, User Manual, System Test, Delivery Plan and Software must match per required criteria</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Ensure that code is logging only critical information such as Fatal Errors and other Useful information. Unnecessary info can overfill logs and crashes the application server</td>
</tr>
<tr>
<td>PHASE II - During Software Deployment</td>
<td>13</td>
<td>Ensure all the preparation for software deployment is completed</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>Create a backup of the build environment and place the development environment under change control</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>Freeze the final list of files to be distributed</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>Synchronize date time stamp on all release files</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>Ensure the security of the server and database is checked</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>Confirm the server is ready to accept the deployment</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>After Deploying the files, confirm the time stamp on the files</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>Check for the configuration files that has App information</td>
</tr>
<tr>
<td></td>
<td>21</td>
<td>Ensure that you have connectivity information such as Database name and Password</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>Check Database Password is not revoked or expired</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>Make sure Application can make connection to given Database</td>
</tr>
<tr>
<td></td>
<td>24</td>
<td>Restart the Servers to make sure new changes are accepted</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>Relaunch the application to confirm deployment went successfully</td>
</tr>
<tr>
<td></td>
<td>26</td>
<td>Notify all the users, managers, developers and testers with deployment status</td>
</tr>
<tr>
<td>PHASE III - After Software Deployment</td>
<td>27</td>
<td>Monitor the logs for few hours for Software Deployment</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>Observe the behavior of new features</td>
</tr>
<tr>
<td></td>
<td>29</td>
<td>Measure and Compare User response</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>Send the monitor status to all managers, Developers and Testers</td>
</tr>
</tbody>
</table>
5. Results and Discussion

We did a small survey to compare our checklist with other online checklist. For this, we got feedback for three different environments from each developer and asked them to give each checklist a number from the range of 0 to 2 where 0 means Not Applicable, 1 means Helpful, 2 means Very Helpful to both using online checklists and using our defined checklist. This range reflects the usefulness of each checklist. Here, are some basic descriptions of each system that each developer briefly shared with us.

Application 1 Description: Locate me Application (Mobile Application)

This mobile application provides the organizational and important information in nearby location. It also provides the navigation between the two places using campus map. Beacons are placed at various locations which helps the people to find the place where they want to go and can easily get the particular information of that location. When any device is found, beacon will perform the functions like to notify the user. This interesting application was for both, students and professors of Stevens Institute of Technology. Student can find location of any professor’s cabin and leave a message to professor’s door if he is not present. Professors can update any event at any time on his beacon. It uses iBeacon technology to show map to students to find location.

Application 2 Description: HOLMES- Hoboken Laundry Management System (Web Application)

It was an easy solution to check laundry status after initiating laundry at any time without going to laundromat. One can check their laundry status by simply log in to the HOLMES, can do payment online using card. The main purpose of this project is to: Design a website to check the status of laundry, get notified once your laundry is done and add extra dryer time if needed online. An amazing part of this site was when the laundry is done, the system will automatically send notifications to the particular customer.

Application 3 Description: IDM Environment Utilities (Desktop Based Application)

The goal of this application is to reduce outages in the production environmental regions. This Desktop based application will constantly check connections to external interfaces, namely:

1) FPS web service
2) Brokerage web service
3) CTM

Any mismatch of values, or broken connections will be reported to a predetermined group of people via email. Additionally, a web form where users can change the input parameters for the external interfaces will be created. Overall, this application will provide the ability to be proactive in identifying problems due to external connections and eliminate difficulties prior to use of applications.

From our survey, we analyze the results and plot some graphs to indicate the results clearly. We found the usefulness of each checklist in the percentage using basic calculations.

5.1 Online Checklist 1:

Table 2 represents the ratings given by developers for usefulness of already existing checklist for software deployment.

<table>
<thead>
<tr>
<th>No</th>
<th>Checklist</th>
<th>Application 1</th>
<th>Application 2</th>
<th>Application 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Is the system reliable?</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Do the successful test cases for the system are mentioned?</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Does the system have high availability?</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Does the system have high usability?</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>Is the code given for deployment up-to-date?</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Is there any estimation of effort?</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>5</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Usefulness of checklist in %</td>
<td>41.67 %</td>
<td>66.67 %</td>
<td>50 %</td>
</tr>
</tbody>
</table>
From the first survey, we got the percentage range of online checklist – 1 as 41.67, 66.67, 50.00 for Application - 1, Application – 2 and Application – 3 respectively.

This result we got from three developers show that the online checklist -1 is not very helpful for their respective systems. The highest usefulness of this checklist is 66.67% which is in average range.
5.2 Online Checklist 2:

Table 3 represents the ratings given by developers for usefulness of already existing checklist for software deployment.

<table>
<thead>
<tr>
<th>No</th>
<th>Checklist</th>
<th>Application 1</th>
<th>Application 2</th>
<th>Application 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>All the activities and tasks required by the Implementation (Deployment) Plan have been included in the project schedule tracking tool, and the estimates for these tasks have been reviewed for accuracy.</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>The Acceptance Test Plan has been reviewed for conformance to the baselined deliverables (e.g., Functional Specification).</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>It has been demonstrated to the Acceptor's satisfaction that the Customer's key quality requirements have been met.</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Licensing agreements (if applicable) have been signed.</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Data has been converted or loaded.</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>Users have been trained.</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>Policies and procedures related to the system have been created or updated, and distributed.</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>Contingency procedures are in place.</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>Backup copies of the developed software have been stored in a secure location.</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>Maintenance and release procedures have been defined, if specified in the Project Plan.</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>Warranty periods and support levels have been clearly defined.</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>A report has been produced from the software item directory listing all items to be installed in each location.</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Description</td>
<td>Usefulness of checklist in %</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>This report has been signed off by the Acceptor indicating for each location the date when all the correct configuration items were installed.</td>
<td>36.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Sufficient personnel knowledgeable in the system have been assigned for warranty support to ensure that the defined support levels are met.</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Access provisions for the delivery team personnel to the Customer's site and equipment have been made.</td>
<td>50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>15</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
From the second survey, we got the percentage range of online checklist – 2 as 36.67, 44.00, 50.00 for Application - 1, Application – 2 and Application – 3 respectively.

This result we got from three developers show that the online checklist – 2 is not very helpful either. The highest usefulness of this checklist is only 50.00% which is in average range and also less then the first checklist.
5.3 Defined Checklist:

From the third survey, we got the percentage range of online checklist – 1 as 85.00, 83.33, 100.00 for Application - 1, Application – 2 and Application – 3 respectively.

This result shows that the our checklist has high range of usefulness. The graph shows that our checklist is the highly useful checklist for all the three applications than already existed checklists.
6. Conclusion

6.1 Limitations and Reflections

Limitations exist in all studies but are necessary to discuss in order to provide an objective view. Software deployment is part of all software projects. A limitation to our research is the variable size and processes used in software projects. For example, some projects are large in scale and use waterfall method to develop software. On the other hand, some projects consist of small teams and use agile methods to develop software. Given the time to complete the project, we had to limit our scope to smaller, more agile developed software. Another limitation of our project was its testability. We had to use function point analysis to determine the effectiveness of our checklist for projects that had already been developed. It would have been helpful if we could have tested our checklist against real-time projects that were approaching software deployment and then compare results.

6.2 Conclusion

In Conclusion, we have proposed a unique software checklist that mitigates the risks associated with software deployment. In order to explain why our checklist is unique compare to current checklist, we have applied all the checklist to three different systems such as Web apps, Desktop apps and Mobile applications. Based on the above graphs, we have seen that online checklist are not very applicable to all the systems. Most of them are based on particular system and which can lead to risk during software deployment. Our checklist is defined in three phases of software deployment which can help user to apply on their system quickly and efficiently. In our checklist, we have asked detailed questions to the user for deployment. This can help them to avoid risk during each phase of the deployment.
7. References


http://users.csc.calpoly.edu/~jdalbey/206/Templates/releasechecklist.html

8. Appendices

For the sake of limiting the size and confusion among our project documentation, we have left these tables out. Please feel free to copy the URL and paste it into the web browser.

1 http://users.csc.calpoly.edu/~jdalbey/206/Templates/releasechecklist.html

1.1 http://it.toolbox.com/blogs/enterprise-solutions/implementation-checklist-22380

1.2 http://www.construx.com/Sample_Release_Checklist/