A Quantitative Approach for Evaluating Software Maintenance Services

Humberto T. Marques-Neto (PUC Minas, Brazil)
Gladston Junio Aparecido (PUC Minas, Brazil)
Marco Tulio Valente (UFMG, Brazil)

SAC’13 – Coimbra, Portugal, March 2013
Motivation

- Software maintenance is an important, costly, and complex phase in software lifecycle
  - 40-80% of all effort spent

- Software development processes are adapted to deliver new releases with corrections and enhancements requested by software users

- Maintenance as a service
  - Is it possible to use queue systems for modeling and evaluating software maintenance processes?
Our Goal

- To offer a *quantitative approach* for evaluating software maintenance services requested periodically by software users to be handled by a software team
  - We have adapted a methodology originally proposed to characterize the workload of e-commerce services
  - We have illustrated the usage of the proposed approach with a real case study
Our Approach
Overview

- Each maintenance request is represented as a state transition graph: **MMG – Maintenance Model Graph**

- The MMGs are clustered to generate a representative model for the maintenance request
The proposed approach has three major steps:

1. Defining the MMGs
2. Instantiating the MMGs
3. Clustering the MMGs
1\textsuperscript{st} Step: Defining the MMGs

- The MMG is a \textit{directed graph} representing the workflow to process maintenance requests
  - **Nodes**: request processing states
    - a) \textbf{SE activities} (under analysis, under implementation, under validation…)
    - b) \textbf{waiting states} (waiting for analysis, waiting for implementation…)
    - c) \textbf{final states} (deployed, canceled…)
  - **Edges**: transitions between such states
2nd Step: Instantiating the MMGs (1/2)

One maintenance request → One MMG

- Each MMG state instance should have the following attributes:
  - Starting and ending timestamps
  - Number of times the state has been visited in order to identify loops
For each MMG instance, we calculate the following characteristics:

\[
\text{Queue Time} = \text{Waiting Time} + \text{Service Time}
\]

\[
\text{Service Time} = \text{Planning Time} + \text{Implementation Time} + \text{Validation Time} + \text{Deployment Time}
\]
3\textsuperscript{rd} Step: Clustering the MMGs (1/3)

- We propose the usage of \textit{k-means algorithm} to cluster the MMGs considering the respective \textit{feature vector}, basically with the \textit{WaitingTime} and \textit{ServiceTime} of the software maintenance request.
3rd Step: Clustering the MMGs (2/3)

- Defining the number of clusters

\[
CRR = \frac{\text{betweenness}}{\text{betweenness} + \sum_{i=1}^{k} \text{withness} [i]}
\]

---

Cluster Representativeness Ratio (CRR)

- 7 clusters

Clustering
3\textsuperscript{rd} Step: Clustering the MMGs (3/3)

- Finally, we must create a MMG representing the MMGs of each cluster:
  - A \textit{feature vector} that represents the set of maintenance requests
  - The \textit{probability} of using the path represented by the edge

\[
edge[S_i,S_j].\text{prob} = \frac{\sum_{r=1}^{n} req[r].edge[S_i,S_j].\text{visits}}{\sum_{r=1}^{n} req[r].state[S_i].\text{visits}}
\]
MMG of a cluster
Case Study
Dataset Overview

- DATAPUC: IT Department of PUC Minas
  - about 40 academic and administrative systems.

- They rely on a lightweighted process: **PASM** (Process for Arranging Software Maintenance Requests), proposed by us in the CSMR’11 [1].

- We analyzed **72 software maintenance requests**, handled by DATAPUC’s team
  - The requests opened and closed in 2010 were registered in a issue tracking system.
Methodology

- We clustered 72 MMGs using the k-means algorithm as implemented in R statistical
- We executed the algorithm 10 times, with k ranging from 1 to 10
- For each execution we calculated the CRR
Methodology

- We clustered 72 MMGs using the k-means algorithm as implemented in R statistical
- We executed the algorithm 10 times, with k ranging from 1 to 10
- For each execution we calculated the CRR
Results and Discussion
Clusters #1 and #2 group together more than 60% (44 out of 72) of the considered maintenance requests.

Cluster #1: **typical** and **simple** maintenance requests (service time around 3 weeks)

Cluster #2: **typical** and **medium-complexity** maintenance requests (service time around 6 weeks)
Cluster #3 accounts for around 18% of the analyzed requests.

The requests of this cluster can be classified as **typical** and **complex**, with service time around 12 weeks.

<table>
<thead>
<tr>
<th>Clusters</th>
<th># Requests</th>
<th>Waiting Time</th>
<th>Service Time</th>
<th>Queue Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Mean (hours)</td>
<td>CV</td>
<td>%</td>
</tr>
<tr>
<td>Cluster #1</td>
<td>22</td>
<td>25.44</td>
<td>1.03</td>
<td>17.37</td>
</tr>
<tr>
<td>Cluster #2</td>
<td>22</td>
<td>29.96</td>
<td>1.06</td>
<td>11.08</td>
</tr>
<tr>
<td><strong>Cluster #3</strong></td>
<td>13</td>
<td>42.40</td>
<td>0.87</td>
<td>9.37</td>
</tr>
<tr>
<td>Cluster #4</td>
<td>7</td>
<td>84.86</td>
<td>1.21</td>
<td>12.67</td>
</tr>
<tr>
<td>Cluster #5</td>
<td>4</td>
<td>12.66</td>
<td>0.62</td>
<td>1.33</td>
</tr>
<tr>
<td>Cluster #6</td>
<td>3</td>
<td>271.42</td>
<td>0.39</td>
<td>61.58</td>
</tr>
<tr>
<td>Cluster #7</td>
<td>1</td>
<td>933.20</td>
<td>0.00</td>
<td>80.60</td>
</tr>
</tbody>
</table>
Information about the 7 found clusters (3/3)

- Less than **21%** of the requests captured the *outliers*.
- Cluster #4 and #5 have high service times
- Cluster #6 and #7 have high waiting times

<table>
<thead>
<tr>
<th>Clusters</th>
<th># Requests</th>
<th>Waiting Time</th>
<th>Service Time</th>
<th>Queue Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Mean (hours)</td>
<td>CV</td>
<td>%</td>
</tr>
<tr>
<td>Cluster #1</td>
<td>22</td>
<td>25.44</td>
<td>1.03</td>
<td>17.37</td>
</tr>
<tr>
<td>Cluster #2</td>
<td>22</td>
<td>29.96</td>
<td>1.06</td>
<td>11.08</td>
</tr>
<tr>
<td>Cluster #3</td>
<td>13</td>
<td>42.40</td>
<td>0.87</td>
<td>9.37</td>
</tr>
<tr>
<td>Cluster #4</td>
<td>7</td>
<td>84.86</td>
<td>1.21</td>
<td>12.67</td>
</tr>
<tr>
<td>Cluster #5</td>
<td>4</td>
<td>12.66</td>
<td>0.62</td>
<td>1.33</td>
</tr>
<tr>
<td>Cluster #6</td>
<td>3</td>
<td>271.42</td>
<td>0.39</td>
<td>61.58</td>
</tr>
<tr>
<td>Cluster #7</td>
<td>1</td>
<td>933.20</td>
<td>0.00</td>
<td>80.60</td>
</tr>
</tbody>
</table>
Service Time Subcomponents (1/2)

- There are small variations in the percentage of time allocated to each activity of clusters #1, #2, and #3
- The planning time of these clusters accounts around 16%, 17%, and 15% of the service time

Differences among such clusters could be due to inherent complexity of the associated maintenance tasks.
We can also discover **outlier behaviors** after the clustering

- Clusters #4 and #5 could be considered outliers due to the number of **hours allocated to validation**
MMGs of Clusters #1 and #2
MMGs of Clusters #4 and #5

MMG of Cluster #4

MMG of Cluster #5
MMGs of Clusters #6 and #7

MMG of Cluster #6

MMG of Cluster #7
Conclusions
Contributions

1. A quantitative approach for evaluating maintenance services, based on cluster analysis techniques.
   - Creating representative MMGs in three steps: defining, instantiating, and clustering.

2. Case study to evaluate the applicability of the proposed approach
Benefits

1. Better understanding of the current status of the maintenance activities.
2. Better manage and improve the maintenance process
   - e.g., different skills and experience to handle different types of requests.
3. We could anticipate the cost, required resources, and processing times of upcoming requests.
4. In-depth analysis of outliers may help to identify problems in the current maintenance process.
Further Work

1. Applying the approach in different organizations which handle software maintenance

2. Create a tool to automatize the proposed approach
A Quantitative Approach for Evaluating Software Maintenance Services

Humberto T. Marques-Neto (PUC Minas, Brazil)
humberto@pucminas.br

SAC’13 – Coimbra, Portugal, March 2013