Selection of Customers for Operational and Usage Profiling

Andriy Miranskyy, Enzo Cialini, David Godwin
IBM Canada Ltd.

DBTest 2009
Outlook

- Why prioritization of customers is important?

- How do we do it?
  - Qualitatively
  - Quantitatively

- Validation Case Study
Introduction

- Operational and usage profiles collected from customers provide developers and testers with valuable quantitative information on usage patterns of software being developed.

- Collected business intelligence helps make better and more-informed decisions to
  - Reduce defect escapes to the field
  - Close gaps in test coverage
  - Prioritize test coverage
  - Prioritize fix delivery (APARs) to the field
  - Maximize resource efforts
“Big bang” approach

- A “big bang” approach is to profile all customers.
  - Unfortunately, gathering such profiles from a large set of customers can be challenging due to
    - storage requirements, time and resource constraints;
    - moreover, customers may refuse to provide access to their systems due to confidentiality and legal
      reasons.
  - This approach also leads to duplicate information being gathered

Idea: information about customer defects can help us to narrow down a list of candidate customers to profile.
Prioritization dimensions

1. Total number of defects found by a given customer

2. Average number of discoveries per defect found by a given customer
## Customer prioritization criteria

<table>
<thead>
<tr>
<th>Total number of defects found by a given customer</th>
<th>Average number of discoveries per defect found by a given customer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Low</td>
</tr>
<tr>
<td>Low</td>
<td>LL: Not interesting from profiling perspective</td>
</tr>
<tr>
<td>High</td>
<td>HL: Potential candidate for profiling</td>
</tr>
</tbody>
</table>

- **LL**: Customer finds a small number of defects that are rarely discovered by others. Incorporation of customer’s usage allows us to close small number of gaps in testing of infrequently executed paths.
- **LH**: Customer finds a small number of defects that are often discovered by others. These defects are clearly development, test process and test coverage misses. The defects need to be addressed as they are interesting from a test process miss perspective. However, incorporation of customer’s usage allows us to close only a small number of gaps in testing of commonly executed paths.
- **HL**: Customer finds a large number of defects that are rarely discovered by others. These customers can be considered “unique”. Incorporation of customer’s usage allows us to close a large number of gaps in testing of infrequently executed paths.
- **HH**: Customer finds a lot of defects that are also found by other customers. Incorporation of customer’s usage allows us to close a large number of gaps in testing of commonly executed paths.
Manual Customer Selection Issues

- **Avoid duplication of efforts**
  - Suppose we identified two customers who discovered a large number of defects that are also frequently discovered by other customers.
  - We now need to make sure that the lists of defects discovered by these two customers do not overlap significantly: otherwise, we will be duplicating our effort.
Customer Selection Technique

- Manual selection of customers can become cumbersome if a product is used by thousands of customers discovering hundreds of defects.

- We need to find techniques that will allow us to:
  
  - Minimize the number of customers for profiling while maximizing the total number of discovered defects;
  
  - Prioritize a minimal set of customers (once identified) by the total number of discovered defects per customer.
Minimization of Customer Set

- Formulate minimization task as a Binary Integer Programming (BIP) problem
- Identify a minimal set of customers that discovered all defects

Minimize \( w_1 c_1 + w_2 c_2 + \ldots + w_N c_N \),
subject to
\[
\begin{align*}
d_1 & : p_{1,1} c_1 + p_{1,2} c_2 + \ldots + p_{1,N} c_N \geq 1, \\
d_2 & : p_{2,1} c_1 + p_{2,2} c_2 + \ldots + p_{2,N} c_N \geq 1, \\
& \vdots \\
d_M & : p_{M,1} c_1 + p_{M,2} c_2 + \ldots + p_{M,N} c_N \geq 1,
\end{align*}
\]
binary variables: \( c_1, c_2, \ldots, c_N \);

- \( N \equiv \) total number of customers;
- \( M \equiv \) total number of defects;
- \( c_i \equiv i\)-th customer \((i = 1 \ldots N)\), \( c_i = 1 \) if the \( i\)-th customer is included in the minimal set of customers to profile and is 0 otherwise;
- \( d_j \equiv j\)-th defect \((j = 1 \ldots M)\);
- \( p_{ij} \equiv \) binary variable showing discovery of the \( j\)-th defect by the \( i\)-th customer, \( p_{ij} = 1 \) if the \( i\)-th customer discovered the \( j\)-th defect and is 0 otherwise;
- \( w_i \equiv i\)-th customer weight.
Minimization of Customer Set. Customer weight

- If we want to emphasize “importance” of the $i$-th customer, then we should decrease weight $w_i$ relative to the weight of the remaining customers.
  - For example, $w_i$ can be proportional to the difficulty of gathering information from the $i$-th customer and inversely proportional to the average number of discoveries per defect found by the $i$-th customer.
  - If all customers are considered equal then $w_i = 1$ for all $i$. 
Minimization of Customer Set. Complexity

- In short form BIP problem can be written as

  Minimize $w^T c$,

  subject to $pc \geq 1$,

  binary variables: $c$.

  - This approach should provide us with optimal solution.

- In general, solution of BIP problems is NP-hard.
- However, if a constraint matrix $p$ is totally unimodular and the right hand side of constraints consists of integer values, then the problems can be solved efficiently.
- Our problem formulation falls into this category of BIP problems.
Minimization of Customer Set. Example

- Suppose we have four customers
  - $c_1$, $c_2$, $c_3$, and $c_4$

- The customers discovered five defects
  - $d_1$, $d_2$, $d_3$, $d_4$, and $d_5$

### Table: Defects vs. Customers

<table>
<thead>
<tr>
<th>Defects</th>
<th>$c_1$</th>
<th>$c_2$</th>
<th>$c_3$</th>
<th>$c_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_1$</td>
<td></td>
<td>×</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$d_2$</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>$d_3$</td>
<td>×</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>$d_4$</td>
<td></td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>$d_5$</td>
<td></td>
<td></td>
<td></td>
<td>×</td>
</tr>
</tbody>
</table>

Minimize $c_1 + c_2 + c_3 + c_4$,

subject to

$d_1 : c_2 \geq 1,$

$d_2 : c_1 + c_2 + c_3 \geq 1,$

$d_3 : c_1 + c_3 \geq 1,$

$d_4 : c_3 + c_4 \geq 1,$

$d_5 : c_4 \geq 1,$

binary variables: $c_1, c_2, c_3, c_4$.

Solution: $c_1=0$, $c_2=1$, $c_3=1$, and $c_4=1$; i.e., the minimal set of customers for profiling that cover all defects is \{c_2, c_3, c_4\}. 
Prioritization of Customers within the Minimal Set

- In order to prioritize customers within the set of customers for profiling we propose to use the following greedy heuristics.

  The customer prioritization heuristic greedily selects customer with the largest number of non-covered defects. Once the customer is selected, the customer’s defects are marked as covered. The process repeats itself until all the defects are covered at least once.

- We avoid applying this heuristic to the initial set of customers directly (skipping the BIP step) because of the sub-optimality of the heuristics.
Prioritization of Customers within the Minimal Set: Example

- The minimal set of customers for prioritization is \{c_2, c_3, c_4\}. "Non-covered" defects per customer are \{{d_1, d_2}\}, \{{d_2, d_3, d_4}\}, \{{d_4, d_5}\}\), respectively.

- Since \(c_3\) discovered the largest number of defects, pick \(c_3\) as the first customer to profile.

- Mark defects \(d_2, d_3,\) and \(d_4\) as “covered” and remove them from the “non-covered” list.

- The defects list is changed to \{{d_1}\}, \{\emptyset\}, \{{d_5}\}\): customers \(c_2\) and \(c_4\) have one uncovered defect.

- Arbitrarily pick \(c_2\) as the second customer for profiling and \(c_4\) as the third one.
Case Study

- **Software under Study:**
  - complex commercial software application with over 10 million lines of executable source code and a large customer base

- **Defects:** field defects for core components of the software under study over a five-year period.

- In order to stabilize the number of discoveries per defect we select defects that were first discovered at least six months before this case study was performed.
Total number of discovered defects vs. average number of rediscoveries per customer

- HH: Good candidates for profiling
- HL: Potential candidates for profiling ("unique" customers)
- LL & LH: Not interesting from profiling perspective
Percentage of the total number of customers needed to cover a certain percentage of defects

<table>
<thead>
<tr>
<th>Defects discovered at least Y times</th>
<th>Cover X% of defects</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>80%</td>
<td>100%</td>
</tr>
<tr>
<td>1</td>
<td>8.9%</td>
<td>25.9%</td>
</tr>
<tr>
<td>2</td>
<td>3.5%</td>
<td>11.1%</td>
</tr>
<tr>
<td>5</td>
<td>1.1%</td>
<td>3.0%</td>
</tr>
<tr>
<td>10</td>
<td>0.4%</td>
<td>0.9%</td>
</tr>
</tbody>
</table>
Conclusions

- **Customer selection technique**
  - Minimizes the number of customers for profiling while maximizing the total number of discovered defects;
  - Prioritizes a minimal set of customers (once identified) by the total number of discovered defects per customer.

- **Customer selection technique helps to**
  - Testing
    - Reduce defect escapes to the field
      - by enhancing in-house workloads based on customer information
    - Close gaps in test coverage
    - Prioritize test coverage
  - Maintenance
    - Prioritize fix delivery (APARs) to the field
    - Maximize resource efforts