Module Size Distribution and Defect Density

Computer Science Dept.
Colorado State University

With updates
Module Size Distribution and Defect Density

- Significance: Factors that affect defect density
- Existing work: Data & Hypothesis
- A New Composite Defect Density Model
- Available Data & Model
- Module Size Distribution: Predictable?
- Total Defect Content & Implications
- Observations & Conclusions
Factors Affecting Defect Density

- Multiplicative models:
  - RADC
  - ROBUST

- Sub-models:
  - Phase
  - Programming team
  - Process maturity
  - Structure
  - Requirement volatility
Earlier Studies

- Shen et al.: For modules > 500 lines, no size-density relation. Smaller modules: density declines with size
- Banker and Kemerer: Hypothesis for optimal module size
- Withrow: minimum near size \( \approx 200 \)
- Hatton: two separate models for smaller & larger modules
- Rosenberg: module size-defect density correlation misleading
- Fenton and Ohlsson: no significant dependence observed
A Composite Defect Density Model

• **Module-related faults:** associated with
  - parameters passed among the modules,
  - assumptions made by modules regarding each other,
  - handling of global data,
  - Assumption: such faults are uniformly distributed among the modules.

• **Instruction-related faults:** *bulk* defect density. Assumption: defect density components are
  - constant,
  - number of other instructions a given instruction may interact with.
A Composite Defect Density Model

- Module related defect density:
  - total defects/module: \( a \), module size: \( s \)
  
  \[
  D_m(s) = \frac{a}{s}
  \]

- Instruction related defect density
  
  \[
  D_i(s) = b + cs
  \]

- The composite defect density is then
  
  \[
  D(s) = D_m(s) + D_i(s)
  \]

  \[
  = \frac{a}{s} + b + cs
  \]
The Two Regions

• The minimum defect density \( D_{\min} = (2\sqrt{ac} + b) \)

occurs at module size \( s_{\min} = \sqrt{\frac{a}{c}} \)

• Model implies two regions:
  Region A: For modules with \( s < s_{\min} \)
  Region B: For modules with \( s > s_{\min} \)
## Data: Basili & Perricone

<table>
<thead>
<tr>
<th>Module Size (max)</th>
<th>Module count</th>
<th>Cyclomatic Complexity</th>
<th>Defect Density (/KLOC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>258</td>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td>100</td>
<td>70</td>
<td>17.9</td>
<td>12.6</td>
</tr>
<tr>
<td>150</td>
<td>26</td>
<td>28.1</td>
<td>12.4</td>
</tr>
<tr>
<td>200</td>
<td>13</td>
<td>52.7</td>
<td>7.6</td>
</tr>
<tr>
<td>225</td>
<td>3</td>
<td>60</td>
<td>6.4</td>
</tr>
</tbody>
</table>

![Graph showing observed and fitted defect density against module size]
Withrow Data

<table>
<thead>
<tr>
<th>Source Lines</th>
<th>Modules</th>
<th>Defect Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-62</td>
<td>93</td>
<td>5.4</td>
</tr>
<tr>
<td>64-97</td>
<td>39</td>
<td>4.9</td>
</tr>
<tr>
<td>103-154</td>
<td>52</td>
<td>3.4</td>
</tr>
<tr>
<td>161-250</td>
<td>53</td>
<td>1.8</td>
</tr>
<tr>
<td>251-397</td>
<td>46</td>
<td>5.2</td>
</tr>
<tr>
<td>402-625</td>
<td>31</td>
<td>5.6</td>
</tr>
<tr>
<td>651-949</td>
<td>22</td>
<td>6.8</td>
</tr>
<tr>
<td>1050-5160</td>
<td>26</td>
<td>8.3</td>
</tr>
</tbody>
</table>

[Graph showing observed and fitted defect density vs module size]
Columbus Data

![Graph showing observed and fitted defect density against module size.](image-url)
# Parameter Values

<table>
<thead>
<tr>
<th>Data</th>
<th>$S_{\text{min}}$</th>
<th>Parameter Values</th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basili</td>
<td>NA</td>
<td>220.9</td>
<td>7.83</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Columbus</td>
<td>400</td>
<td>223.79</td>
<td>4.73</td>
<td>0.0013</td>
<td></td>
</tr>
<tr>
<td>Withrow</td>
<td>200</td>
<td>121.19</td>
<td>1.76</td>
<td>0.0063</td>
<td></td>
</tr>
</tbody>
</table>
Distribution of Module Sizes

- Density function for module size distribution:
  \[ f_s(s) = g.e^{-gs} \]
## Module Size Distribution: Parameters

<table>
<thead>
<tr>
<th>Data</th>
<th>Language</th>
<th>M (total modules)</th>
<th>Parameter g</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basili</td>
<td>Fortran</td>
<td>370</td>
<td>0.0054</td>
</tr>
<tr>
<td>Withrow</td>
<td>ADA</td>
<td>362</td>
<td>0.0041</td>
</tr>
<tr>
<td>Shen</td>
<td>PL/S</td>
<td>108</td>
<td>0.0029</td>
</tr>
<tr>
<td>Gnu C Lib</td>
<td>C</td>
<td>792</td>
<td>0.0097</td>
</tr>
</tbody>
</table>
Overall Defect Density

- If $S_T$: total project size, $M$: number of modules, the overall defect density is

$$D = \frac{1}{S_T} \int_{s_{\text{min}}}^{s_{\text{max}}} Mge^{-gs}(\frac{a}{s} + b + cs).10^{-3}.s.ds$$

- Example: $M = 400$, $g = 0.004$, largest module 2000 lines, $a = 120$, $b = 1.8$, $c = 0.006$

$$s_{\text{min}} = 141.42$$

$$S_T = \int_{1}^{s_{\text{max}}} Mge^{-gs}.s.ds = 100,000 \text{ lines}$$

$$D = 7.09 \text{ per KLOC}$$
Optimal Module Size Distribution?

If all modules can be equal, make them $s_{\text{min}}$.

- If they are exponentially distributed:

$$D = \int_{1}^{s_{\text{max}}} g^2 e^{-gs} \left(\frac{a}{s} + b + cs\right) \cdot 10^{-3} \cdot s \cdot ds$$

$$\approx 0.001(a g + b + 2 \frac{c}{g})$$

hence $g_{\text{opt}} = \sqrt{\frac{2c}{a}}$ and $s_{\text{opt}} = \sqrt{\frac{a}{2c}} = \frac{s_{\text{min}}}{\sqrt{2}}$

- Merge smaller modules resulting in a peak near $s_{\text{min}}$.

Merge smaller
Sub-model: Module Size Distribution

- Multiplicative sub-model
- Default value: 1
- Parameters estimated using calibration
- Assuming exponential distribution

\[ F_{ms} = Ag + B + \frac{C}{g} \]

- Example: If \( a = 120, \ b = 1.8, \ c = 0.006, \) and default \( g = 0.005 \)

\[ F_{ms} = 25g + 0.375 + \frac{2.5 \times 10^{-3}}{g} \]
Observations on Data

• **Trends not observable** if number of modules is small.
• **Trend for region B not observed** if size<$s_{\text{min}}$ for most modules, as in Basili & Perricone’s data (very few modules >400). Weak dependence.
• **Trend for region A not observed** if size>$s_{\text{min}}$ for most modules, as in Fenton and Ohlsson’s data (very few modules with size <500). Stronger dependence.
• **Selective testing** or uneven reuse may mask dependence.
• **Avoiding very small modules** may be more beneficial than avoiding very large modules.
Conclusions

• A model explaining both declining and rising defect density trends.
• Module size distribution is often exponential due to natural reasons.
• A defect density model to take variation in size distribution into account.
• Adjusting size distribution may minimize defects.
• Impact of merging or breaking modules needs to be studied.
Recent Developments
Rosenberg’s Analysis

- Argument: If we assume X and Y are statistically independent.
  - Then scatter-plot of Y/X against X looks like *declining defect density vs. module size plot* (Region A).

- Flaw: Note that assumption implies that total defects in a module is independent of module size, i.e. defect density is inversely proportional to module size.

A student wrote in 2011

- I almost caused a riot at work when I mentioned that there was data showing that larger software modules had a lower defect density that more smaller modules.
AT&T Study

Fenton & Ohlsson: no modules < 500

Table 4. Faults/1000 Lines of code release n and n+1.

<table>
<thead>
<tr>
<th>Module size</th>
<th>Release n</th>
<th>Frequency</th>
<th>Faults/1000 Lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td></td>
<td>3</td>
<td>1.45</td>
</tr>
<tr>
<td>1000</td>
<td></td>
<td>15</td>
<td>4.77</td>
</tr>
<tr>
<td>1500</td>
<td></td>
<td>32</td>
<td>5.24</td>
</tr>
<tr>
<td>2000</td>
<td></td>
<td>24</td>
<td>6.32</td>
</tr>
<tr>
<td>2500</td>
<td></td>
<td>14</td>
<td>5.88</td>
</tr>
<tr>
<td>3000</td>
<td></td>
<td>22</td>
<td>5.74</td>
</tr>
<tr>
<td>3500</td>
<td></td>
<td>11</td>
<td>7.83</td>
</tr>
<tr>
<td>&gt;3500</td>
<td></td>
<td>9</td>
<td>7.38</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Module size</th>
<th>Release n+1</th>
<th>Frequency</th>
<th>Faults/1000 Lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td></td>
<td>6</td>
<td>13</td>
</tr>
<tr>
<td>1000</td>
<td></td>
<td>17</td>
<td>6</td>
</tr>
<tr>
<td>1500</td>
<td></td>
<td>35</td>
<td>5</td>
</tr>
<tr>
<td>2000</td>
<td></td>
<td>41</td>
<td>7</td>
</tr>
<tr>
<td>2500</td>
<td></td>
<td>34</td>
<td>5</td>
</tr>
<tr>
<td>3000</td>
<td></td>
<td>37</td>
<td>5</td>
</tr>
<tr>
<td>3500</td>
<td></td>
<td>18</td>
<td>7</td>
</tr>
<tr>
<td>&gt;3500</td>
<td></td>
<td>42</td>
<td>8</td>
</tr>
</tbody>
</table>

Koru et al.

- In Mozilla and Eclipse, an inspection strategy investing 80 percent of available resources on 100-LOC classes and the rest on 1,000-LOC classes would be more than twice as cost-effective as the opposite strategy.

- We observed that defect proneness increased with module size but at a smaller rate. Therefore, smaller modules were proportionally more defect prone compared to larger ones.