

Matéria: Teste de Software

Trabalho: Detecção de Bad Smells e Refatoração

Aluno: Arthur Ferreira Costa

Matrícula: 812057

Análise de Smells

- Long Method / Alta complexidade cognitiva: o método `generateReport` concentra muitas responsabilidades (montagem de cabeçalho, corpo, lógica de autorização por papel do usuário, formatação CSV/HTML e rodapé). Métodos longos são difíceis de entender, testar em pedaços e manter.
- Condicionais aninhadas / Complexidade de decisão: várias ramificações if/else (por tipo de relatório e papel de usuário) levam a muitos caminhos diferentes. Isso aumenta a probabilidade de bugs e torna difícil adicionar novos formatos de relatório.
- Mutação de entrada (side-effect): o código original marcava `item.priority = true` durante a geração do relatório. Mutar objetos de entrada pode causar efeitos colaterais inesperados em outras partes do sistema e atrapalhar testes que assumem imutabilidade.

Por que são problemáticos para manutenção e testes:

- Long methods e alta complexidade tornam regressões mais prováveis; cobrir todos os caminhos em testes é mais difícil.
- Condicionais aninhadas escondem responsabilidades; a lógica deveria ser separada por responsabilidade (formato, autorização, apresentação).
- Mutação de entradas cria acoplamento e torna os testes menos previsíveis (ordem dos testes pode influenciar resultados).

Relatório da Ferramenta (ESLint + sonarjs)

Comando executado: `npx eslint src/`

Saída do ESLint (antes/na iteração inicial da refatoração):

```
/Users/oarthurfc/faculdade/bad-smells-js-refactoring/src/ReportGenerator.js
  11:3  error  Refactor this function to reduce its Cognitive Complexity from 28 to
the 5 allowed  sonarjs/cognitive-complexity
  46:14  error  Merge this if statement with the nested
one                                         sonarjs/no-collapsible-if
/Users/oarthurfc/faculdade/bad-smells-js-refactoring/src/ReportGenerator.refactored.js
  36:3  error  Refactor this function to reduce its Cognitive Complexity from 7 to
the 5 allowed  sonarjs/cognitive-complexity
' 3 problems (3 errors, 0 warnings)
```

Processo de Refatoração

Escolhi corrigir principalmente o smell de "Long Method / Alta complexidade cognitiva" no método `generateReport`. A técnica principal aplicada foi Extract Method e separação de responsabilidades (Header / Body / Footer builders).

Trecho "Antes" (excerto do `src/ReportGenerator.js`):

```
report += `<h2>Usuário: ${user.name}</h2>\n`;
report += '<table>\n';
report += '<tr><th>ID</th><th>Nome</th><th>Valor</th></tr>\n';
}
// --- Seção do Corpo (Alta Complexidade) ---
for (const item of items) {
  if (user.role === 'ADMIN') {
    // Admins veem todos os itens
    if (item.value > 1000) {
      // Lógica bônus para admins
      item.priority = true;
    }
    if (reportType === 'CSV') {
      report += `${item.id},${item.name},${item.value},${user.name}\n`;
      total += item.value;
    } else if (reportType === 'HTML') {
      if (item.priority) {
        report += `<tr style="font-weight:bold;"><td>${item.id}</td><td>${item.name}</td><td>${item.value}</td></tr>\n`;
      } else {
        report += `<tr><td>${item.id}</td><td>${item.name}</td><td>${item.value}</td></tr>\n`;
      }
      total += item.value;
    }
  } else if (user.role === 'USER') {
    // Users comuns só veem itens de valor baixo
    if (item.value <= 500) {
```

Trecho "Depois" (exerto do `src/ReportGenerator.refactored.js`):

```
export class ReportGenerator {
  constructor(database) {
    this.db = database;
  }
  // Public API: preserve the original signature/behavior
  generateReport(reportType, user, items) {
    const header = this._buildHeader(reportType, user);
    const { body, total } = this._buildBody(reportType, user, items);
    const footer = this._buildFooter(reportType, total);
    // Keep the same return shape as the original implementation (trimmed)
    return (header + body + footer).trim();
  }
  // --- Header builders ---
  _buildHeader(reportType, user) {
    if (reportType === 'CSV') {
      return 'ID,NOME,VALOR,USUARIO\n';
    }
    if (reportType === 'HTML') {
      return (
        '<html><body>\n' +
        '<h1>Relatório</h1>\n' +
        '<h2>Usuário: ${user.name}</h2>\n' +
        '<table>\n' +
        '<tr><th>ID</th><th>Nome</th><th>Valor</th></tr>\n'
      );
    }
    return '';
  }
  // --- Body builder ---
  _buildBody(reportType, user, items) {
    let report = '';
    let total = 0;
    for (const item of items) {
      if (!this._shouldIncludeItemForUser(item, user)) continue;
      const isPriority = user.role === 'ADMIN' && item.value > 1000;
      if (reportType === 'CSV') {
        report += this._csvRow(item, user);
        total += item.value;
      } else if (reportType === 'HTML') {
        report += this._htmlRow(item, isPriority);
        total += item.value;
      }
    }
    return { body: report, total };
  }
  _shouldIncludeItemForUser(item, user) {
    if (user.role === 'ADMIN') return true;
    if (user.role === 'USER') return item.value <= 500;
    // Default conservative: hide item
    return false;
  }
  _csvRow(item, user) {
    return `${item.id},${item.name},${item.value},${user.name}\n`;
  }
  _htmlRow(item, isPriority) {
    if (isPriority) {
      return `<tr style="font-weight:bold;"><td>${item.id}</td><td>${item.name}</td><td>${item.value}</td></tr>\n`;
    }
    return `<tr><td>${item.id}</td><td>${item.name}</td><td>${item.value}</td></tr>\n`;
  }
  // --- Footer builder ---
  _buildFooter(reportType, total) {
    if (reportType === 'CSV') {
      // keep same formatting as original
    }
  }
}
```

```
        return `\\nTotal,,${total},,\\n`;
    }
    if (reportType === 'HTML') {
        return `</table>\\n<h3>Total: ${total}</h3>\\n</body></html>\\n`;
    }
    return '';
}
}
```

Conclusão

Testes como rede de segurança: Os testes existentes permitiram refatorar com confiança — eu pude reorganizar e extrair métodos sem mudar comportamento observável. A redução de bad smells (menos complexidade, responsabilidade única, evitar mutações) facilita manutenção, revisões e extensão futura do código.

Observação: Capa contém placeholders (nome e matrícula). Atualize o arquivo `scripts/generateReportPdf.js` ou a capa no PDF se quiser seus dados reais.