

# Métricas para Testes Automatizados

Paulo Cheque ([paulocheque@agilcoop.org.br](mailto:paulocheque@agilcoop.org.br))

Cursos de Verão 2010

Licença:

Creative Commons: Attribution-Share Alike 3.0 Unported

<http://creativecommons.org/licenses/by-sa/3.0/>



# Definições

- Medida
  - Avaliação em relação a um padrão
- Métrica
  - Método para se certificar de um atributo
  - Composta de uma ou mais medidas
- Indicador
  - Variável que interpreta uma métrica

Ex: Alta cobertura **pode** indicar que o software é de qualidade

# Métricas

- Entender o andamento do projeto
- *Feedback*
- Comunicação
- Problemas
  - Identificar, Acompanhar e Resolver
- Qualidades
  - Identificar, Acompanhar e Exibir
- Gerenciar

# GQM

- Goal -> Question -> Metric

Exemplo:

- G: Desenvolver software com qualidade.

Q: O software está bem testado?

M:

- Cobertura dos testes +  
Número de asserções +  
Número de testes falhando

# Métricas para TA

- Entender:
  - Qualidade do SUT
  - Qualidade dos testes
  - Trechos de código testados, bem testados, mal testados
- Acompanhamento dos testes
- Estratégias de desenvolvimento

# Testabilidade

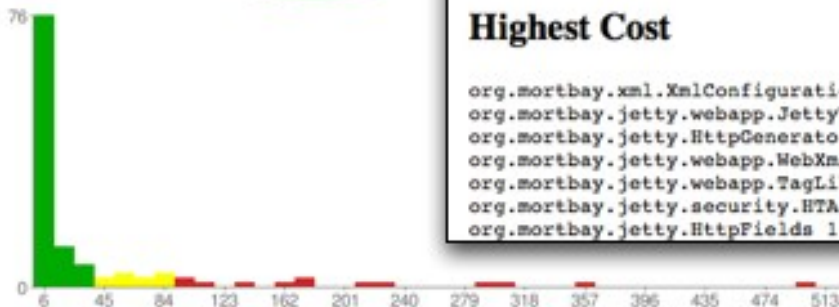
- Mede bons padrões OO:
  - Injeção de dependência
- Mede anti-padrões OO:
  - Variáveis globais
  - Variáveis públicas
  - Singletons
- TDD



jetty-6.0.1.jar

### Class breakdown

Analyzed classes : 117  
 Excellent classes : 94 80.3%  
 Good classes : 10 8.5%  
 Needs work classes : 13 11.1%



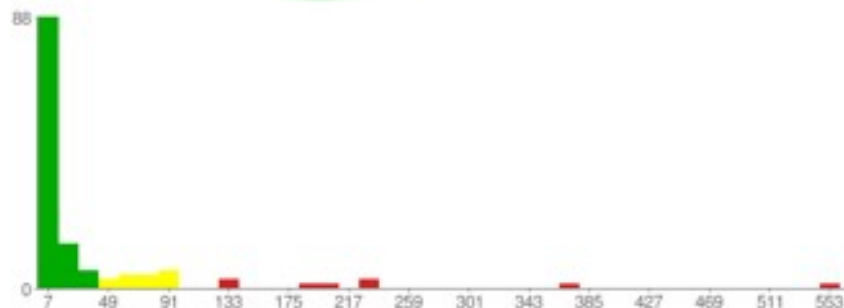
### Highest Cost

org.mortbay.xml.XmlConfiguration 528  
 org.mortbay.jetty.nio.HttpChannelEndPoint 522  
 org.mortbay.jetty.HttpConnection 500  
 org.mortbay.jetty.webapp.JettyWebXmlConfiguration 352  
 org.mortbay.jetty.HttpConnection\$RequestHandler 301  
 org.mortbay.jetty.MimeTypes 293  
 org.mortbay.jetty.webapp.WebXmlConfiguration 226

jetty-6.1.9.jar

### Class breakdown

Analyzed classes : 128  
 Excellent classes : 108 84.4%  
 Good classes : 12 9.4%  
 Needs work classes : 8 6.3%



### Highest Cost

org.mortbay.xml.XmlConfiguration 550  
 org.mortbay.jetty.webapp.JettyWebXmlConfiguration 371  
 org.mortbay.jetty.HttpGenerator 236  
 org.mortbay.jetty.webapp.WebXmlConfiguration 232  
 org.mortbay.jetty.webapp.TagLibConfiguration 200  
 org.mortbay.jetty.security.HTAccessHandler\$HTAccess 193  
 org.mortbay.jetty.HttpFields 137

# Cobertura

- Mede trechos do SUT exercitados pelos testes

Obs: Trecho exercitado não significa ausência de erros

- Trechos não exercitados indicam pontos não seguros (não testados)

UsuarioDAOTest (Feb 2, 2009 11:55:48 PM)

Element	Coverage	Covered Instructions
br.org.agilcoop	95.6 %	131
Config.java	88.9 %	48

Config.java

```

1 package br.org.agilcoop;
2
3 import org.springframework.jdbc.core.JdbcTemplate;
4
5
6
7 public class Config {
8     public static JdbcTemplate jdbc;
9
10    static {
11        bootstrap();
12    }
13
14    public static void bootstrap() {
15        DriverManagerDataSource ds = new DriverManagerDataSource();
16        ds.setDriverClassName("org.hsqldb.jdbcDriver");
17        ds.setUrl("jdbc:hsqldb:mem:teste;shutdown=true");
18        ds.setUsername("sa");
19        ds.setPassword("");
20
21        try {
22            ds.getConnection();
23        } catch (Exception e) {
24            e.printStackTrace();
25        }
26        jdbc = new JdbcTemplate(ds);

```

# Fator de Teste

- = Linha dos testes / Linhas do SUT
- Comparar módulos de um mesmo projeto
- Obs: Não recomendada para acompanhamento dos testes automatizados

# # de TA / Linha de Código do SUT

- Acompanhar a evolução dos TA
- Obs: Desde que o crescimento do projeto não tenha alterações drásticas quanto sua complexidade

# Número de linhas de testes

- Análogo ao número de linhas do SUT

Identificar testes que precisam de refatoração

- Pode ser útil para o gerenciamento de manutenções

# Número de testes

- Acompanhar evolução dos TA
- Útil para evolução dos TA
- Principalmente para projetos que estão começando

Metric	Total	Mean	Std. Dev	Maximum	Resource causing Maximum	Method
▷ Number of Overridden Methods (avg/n	0	0	0	0	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Number of Attributes (avg/max per typ	2	0.667	0.471	1	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Number of Children (avg/max per type	0	0	0	0	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Number of Classes (avg/max per pack	3	1.5	0.5	2	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Method Lines of Code (avg/max per m	57	4.75	4.567	17	/ExemplosTestesBD/src/main/java/br/org/agilcc	bootstrap
▷ Number of Methods (avg/max per type	11	3.667	2.625	6	/ExemplosTestesBD/src/test/java/br/org/agilcc	
▷ Nested Block Depth (avg/max per met		1.083	0.276	2	/ExemplosTestesBD/src/main/java/br/org/agilcc	bootstrap
▷ Depth of Inheritance Tree (avg/max pe		1	0	1	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Number of Packages	2					
▷ Afferent Coupling (avg/max per packag		0	0	0	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Number of Interfaces (avg/max per pa	0	0	0	0	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ McCabe Cyclomatic Complexity (avg/n		1.083	0.276	2	/ExemplosTestesBD/src/main/java/br/org/agilcc	bootstrap
▷ Total Lines of Code	119					
▷ Instability (avg/max per packageFragm		1	0	1	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Number of Parameters (avg/max per r		0.667	0.943	3	/ExemplosTestesBD/src/main/java/br/org/agilcc	atualizaUsuario
▷ Lack of Cohesion of Methods (avg/max		0	0	0	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Efferent Coupling (avg/max per packag		1.5	0.5	2	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Number of Static Methods (avg/max p	1	0.333	0.471	1	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Normalized Distance (avg/max per pac		0	0	0	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Abstractness (avg/max per packageFri		0	0	0	/ExemplosTestesBD/src/main/java/br/org/agilcc	
▷ Specialization Index (avg/max per type		0	0	0	/ExemplosTestesBD/src/main/java/br/org/agilcc	

# Número de asserções

- Útil para verificar se os testes estão realmente fazendo verificações
- Pode indicar testes que precisam de refatoração

# Número de testes pendentes

- Acompanhamento dos TA
- Pode indicar problemas com prazos (pressão, correria...)
- Pode indicar anti-padrões de TDD

# Número de testes falhando

- Fragilidade dos testes
- Acompanhamento de conserto dos testes

# Número de asserções por método

- Pode indicar métodos de testes que precisam de refatoração
- Podem identificar classes com muitas responsabilidades

# Replicação do código dos testes

- Testes que precisam ser refatorados
- Indicar que o código da SUT também está com replicação

# Quantidade de defeitos encontrados

- Qualidade do software
- Falta de testes automatizados
  - Testes de unidade
  - Testes de Aceitação

# Tempo de execução dos testes

- Encontrar gargalos do sistema
- Encontrar testes que precisam ser otimizados ou refatorados

# Objetivos vs Métricas

Objetivo \ Métrica	1	2	3	4	5	6	7	8	9	10	11	12	13
Encontrar defeitos		o	o										
Melhorar o código do sistema	o	o									o		o
Melhorar o código dos testes		o	o		o		o	o	o	o	o	o	o
Introduzir testes automatizados em novos projetos						o							
Introduzir testes automatizados em sistemas legados	o											o	
Acompanhar a automação dos testes		o		o	o	o		o	o			o	

Testabilidade

Número de testes pendentes

Cobertura

Número de testes falhando

Fator de teste

Número de asserções por métodos

Número de testes / LOC do sistema

Replicação do código de testes

Número de linhas de teste

Quantidade de defeitos encontrados

Número de testes

Tempo de execução da bateria de testes

Número de asserções

# Ferramentas

- Eclipse Metrics:
  - <http://metrics.sourceforge.net/update>
- Eclipse Eclemma:
  - <http://update.eclemma.org>
- Testability Explorer:
  - <http://code.google.com/p/testability-explorer>

# Contato

<http://www.agilcoop.org.br>

[agilcoop@agilcoop.org.br](mailto:agilcoop@agilcoop.org.br)

[paulocheque@agilcoop.org.br](mailto:paulocheque@agilcoop.org.br)

