

# Benchmarking Software Components

---

## Using High Volume Automated Testing Techniques

20<sup>th</sup> Annual Conference of the United Kingdom Software Metrics Association  
October 15<sup>th</sup> 2009 11:15

**Jeremy Gardiner**

[jeremy.gardiner@woomerang.co.uk](mailto:jeremy.gardiner@woomerang.co.uk)

Department of Informatics and Sensors  
Cranfield University at the UK Defence Academy  
Shrivenham, United Kingdom

1

---

---

---

---

---

---

---

---

## Overview

---

- 1 Benchmarking Software Components
- 2 Benchmark Metrics
- 3 High Volume Automated Testing (HVAT)
- 4 Example: Database Management System (DBMS)
- 5 Benchmark Procedure and Results

2

---

---

---

---

---

---

---

---

## Component-Based Software Engineering

---

- 1 Component-Based Software Engineering (CBSE)
- 2 Build Software Systems by Composition of Pre-existing Software Components
- 3 Component: a Minimal Software Item for Which a Separate Specification is Available
- 4 Goal of Component Re-use/Substitution

3

---

---

---

---

---

---

---

---

## Problems in CBSE

---

- 1 How to Predict Properties of Assembled System Given Properties of Components
- 2 How to Guarantee One Component Can Be Substituted for Another Without Changing the Properties of the System

4

---

---

---

---

---

---

---

---

## Metrics and Benchmarking

---

- 1 Starting Point: Metrics and Benchmarking
- 2 Define Metrics to Characterise Components
- 3 Use Metrics in Benchmarking Components
- 4 Compare Results for Individual Components
- 5 Future: How to Combine Component Metrics

5

---

---

---

---

---

---

---

---

## Dynamic Analysis

---

- 1 Empirical Approach: Dynamic Analysis
- 2 Contrast to Static Analysis
- 3 Automated Testing of Software Components
- 4 Generic Component Model

6

---

---

---

---

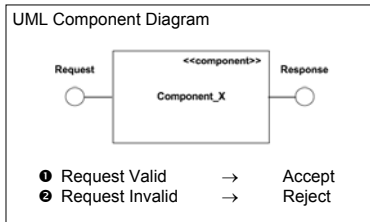
---

---

---

---

## Generic Component Model



7

---

---

---

---

---

---

---

---

## Benefits of the Approach

- 1 Specification of Valid Requests (No Source Code)
- 2 Automated Procedure (Generation and Execution)
- 3 Any Component with Request / Response Interface
- 4 Suitable for Measuring Reliability and Robustness
- 5 Important Software Properties...

8

---

---

---

---

---

---

---

---

## Reliability and Robustness

**Reliability** *The ability of a software component to perform its required functions under stated conditions for a specified period of time or for a specified number of operations*

**Robustness** *The degree to which a software component can function correctly in the presence of invalid inputs*

IEEE Standard Computer Dictionary 610 (1991)

9

---

---

---

---

---

---

---

---

## Metrics

*Mean Time to Failure*      *MTTF*

*Assume Constant Failure Rate  $\lambda$*

- ❶ Reliability  $R(t) = e^{-\lambda t}$
- ❷  $R(t)$  = Probability of No Failure Before Time  $t$
- ❸  $MTTF = 1/\lambda$

Storey, *Safety-Critical Computer Systems* (1996)

10

---

---

---

---

---

---

---

---

## Availability

*Mean Time Between Failures*      *MTBF*  
*Mean Time to Failure*              *MTTF*  
*Mean Time to Recovery*            *MTTR*

- ❶  $MTBF = MTTF + MTTR$
- ❷ Availability  $A = (MTTF / MTBF) \times 100\%$

Storey, *Safety-Critical Computer Systems* (1996)

11

---

---

---

---

---

---

---

---

## Probability of Failure on Demand

- ❶ *Probability of Failure on Demand*      *POFOD*
- ❷ *The Likelihood That the System Will Fail When a User Requests Service*
- ❸ *Rate of Occurrence of Failure*              *ROCOF*

Fenton & Pfleeger, *Software Metrics* (1997)

12

---

---

---

---

---

---

---

---

## The F-Measure

*The Number of Tests Required in a Sequence to Detect the First Program Failure*

- ❶ Probability of Failure  $p$        $q = 1-p$
- ❷ Probability Density Function       $P(X=n) = q^{(n-1)}p$
- ❸ Geometric Distribution       $MTTF = E(X) = 1/p$

Chen, Kuo & Merkel, *On the Statistical Properties of the F-measure* (2004)

13

---

---

---

---

---

---

---

---

## Basic Approach

- ❶ Choice of Metrics...
- ❷ Same Basic Approach
- ❸ Execute Software Until Failure...
- ❹ Under Controlled Conditions

14

---

---

---

---

---

---

---

---

## High Volume Automated Testing

- ❶ Consider Testing as Statistical Sampling
- ❷ Very Large Numbers of Test Cases (100,000+)
- ❸ Enhanced Testing Coverage
- ❹ Automated Generation and Execution of Tests
- ❺ HVAT for Component Benchmarking

15

---

---

---

---

---

---

---

---

## Qualification

*Demonstrates That a Design Will Perform in the Expected Operational Environment With a Specified Qualification Margin*

Forsberg, Mooz & Cotterman, *Visualizing Project Management* (2005)

- 1 Alternative View of HVAT
- 2 Qualification of Components for Re-use in CBSE
- 3 Benchmark Profile

16

---

---

---

---

---

---

---

---

## Component Example: DBMS

- 1 Database Management System
- 2 Freely Available Commercial Component MySQL
- 3 Specified Interface
- 4 Structured Query Language (SQL)
- 5 Request / Response Model

17

---

---

---

---

---

---

---

---

## Benchmark Procedure

- 1 Random Generation of SQL
- 2 Specification → Test Generator → Benchmark Profile
- 3 SQL Statements → Test Executor → DBMS Requests
- 4 DBMS Responses → Test Executor → Log File

18

---

---

---

---

---

---

---

---

## Benchmark Profile 1

```
CREATE TABLE ... [SELECT * FROM ...]  
INSERT ... [SELECT * FROM ...]  
REPLACE ... [SELECT * FROM ...]
```

- ❶ Attempted CREATE of existing table **Invalid**
- ❷ Operation on non-existing table/row/column **Invalid**
- ❸ Eventually Fills MySQL Table Space: "Table is Full"
- ❹ 300,000 Executed SQL Statements

19

---

---

---

---

---

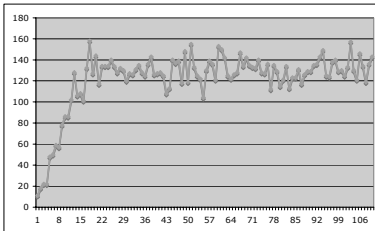
---

---

---

## Benchmark Results for Profile 1

Success Responses



Experimental Runs

20

---

---

---

---

---

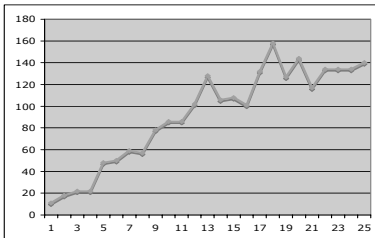
---

---

---

## Benchmark Results (Magnified)

Success Responses



Experimental Runs

21

---

---

---

---

---

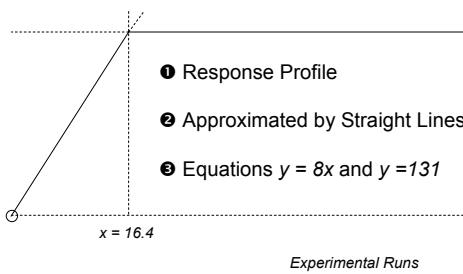
---

---

---

## Analysis of Results for Profile 1

Success Responses



22

---

---

---

---

---

---

---

---

## Benchmark Profile 2

```

CREATE TABLE ... [SELECT * FROM ...]
DELETE ... FROM ...
INSERT ... [SELECT * FROM ...]
REPLACE ... [SELECT * FROM ...]
    
```

- ❶ Attempted CREATE of existing table **Invalid**
- ❷ Operation on non-existing table/row/column **Invalid**
- ❸ MySQL Eventually Crashes (Sometimes...)

23

---

---

---

---

---

---

---

---

## Benchmark Results for Profile 2

Run	Executed	Error	Accepted	Succeeded	Time (s)
3	412,884	71	354,331	58,482	1,690
6	122,612	45	105,450	17,117	852
10	121,625	34	104,993	16,598	883
11	29,057	7	25,841	3,209	238
12	45,498	13	39,578	5,906	350
16	13,425	2	12,383	1,220	237
17	17,352	3	15,914	1,435	298
19	30,147	8	26,733	3,407	320
Total	792,600	183	685,223	107,374	4,868
Mean	24,075	6	21,122	2,970	293

- ❶ No Failure in 60% of Random Test Runs

24

---

---

---

---

---

---

---

---

## Analysis of Results for Profile 2

---

- ❶ Mean 24,075 Executed Statements To Failure
- ❷ MTTF 293 Seconds
- ❸ Failure Rate  $\lambda = 1/\text{MTTF} = 0.0034 \text{ s}^{-1}$
- ❹ MTTR Not Relevant (No Availability Measure)

25

---

---

---

---

---

---

---

---

## MySQL Failure

---

Bug: #45639 INSERT on MERGE Table Results in a Crash  
Version: 5.0.22

<http://bugs.mysql.com/45639>

```
mysql> create table t (c int) engine=merge insert_method=first;
```

Query OK, 0 rows affected (0.11 sec)

```
mysql> insert t values(42);
```

ERROR 2013 (HY000):  
Lost connection to MySQL server during query

26

---

---

---

---

---

---

---

---

## Conclusions

---

- ❶ Approach seems feasible (for this example)
- ❷ Metrics Failure Rate  $\lambda = 1/\text{MTTF}$
- ❸ Require Component to **Fail**
- ❹ No Failure in 60% of Random Test Runs
- ❺ Response Profile → Component Comparison

27

---

---

---

---

---

---

---

---

## Future Work

---

- ❶ Response Profile → Component Comparison
- ❷ Need Comparable Results for another DBMS
- ❸ Oracle XE Express Edition

28

---

---

---

---

---

---

---

---

## Summary

---

- ❶ Benchmarking Software Components
- ❷ Benchmark Metrics
- ❸ High Volume Automated Testing (HVAT)
- ❹ Example: Database Management System (DBMS)
- ❺ Benchmark Procedure and Results

29

---

---

---

---

---

---

---

---

## Benchmarking Software Components

---

### Using High Volume Automated Testing Techniques

20<sup>th</sup> Annual Conference of the United Kingdom Software Metrics Association  
October 15<sup>th</sup> 2009 11:15

**Jeremy Gardiner**

[jeremy.gardiner@woomerang.co.uk](mailto:jeremy.gardiner@woomerang.co.uk)

Department of Informatics and Sensors  
Cranfield University at the UK Defence Academy  
Shrivenham, United Kingdom

30

---

---

---

---

---

---

---

---