Benchmarking Software Components

Using High Volume Automated Testing Techniques

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

Jeremy Gardiner
jeremy.gardiner@woomerang.co.uk

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

Overview

❶ Benchmarking Software Components

❷ Benchmark Metrics

❸ High Volume Automated Testing (HVAT)

❹ Example: Database Management System (DBMS)

❺ Benchmark Procedure and Results

Component-Based Software Engineering

❶ Component-Based Software Engineering (CBSE)

❷ Build Software Systems by Composition of Pre-existing Software Components

❸ Component: a Minimal Software Item for Which a Separate Specification is Available

❹ Goal of Component Re-use/Substitution

Problems in CBSE

❶ How to Predict Properties of Assembled System Given Properties of Components

❷ How to Guarantee One Component Can Be Substituted for Another Without Changing the Properties of the System

Metrics and Benchmarking

❶ Starting Point: Metrics and Benchmarking

❷ Define Metrics to Characterise Components

❸ Use Metrics in Benchmarking Components

❹ Compare Results for Individual Components

❺ Future: How to Combine Component Metrics

Dynamic Analysis

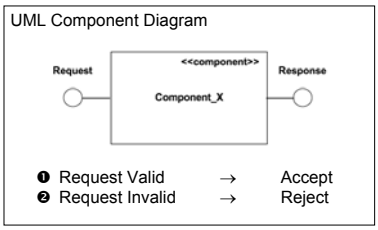
❶ Empirical Approach: Dynamic Analysis

❷ Contrast to Static Analysis

❸ Automated Testing of Software Components

❹ Generic Component Model

Generic Component Model



Benefits of the Approach

❶ Specification of Valid Requests (No Source Code)

❷ Automated Procedure (Generation and Execution)

❸ Any Component with Request / Response Interface

❹ Suitable for Measuring Reliability and Robustness

❺ Important Software Properties…

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)

Metrics

Mean Time to Failure *MTTF*

Assume Constant Failure Rate λ

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)

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) x 100%

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

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)

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)

Basic Approach

❶ Choice of Metrics…

❷ Same Basic Approach

❸ Execute Software Until Failure…

❹ Under Controlled Conditions

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

Qualification

Demonstrates That a Design Will Perform in the Expected Operational Environment With a Specified Qualification Margin
Forsberg, Mooz & Cotterman, *Visualizing Project Management* (2005)

❶ Alternative View of HVAT

❷ Qualification of Components for Re-use in CBSE

❸ Benchmark Profile

Component Example: DBMS

❶ Database Management System

❷ Freely Available Commercial Component MySQL

❸ Specified Interface

❹ Structured Query Language (SQL)

❺ Request / Response Model

Benchmark Procedure

❶ Random Generation of SQL

❷ Specification → Test Generator → Benchmark Profile

❸ SQL Statements → Test Executor → DBMS Requests

❹ DBMS Responses → Test Executor → Log File

Benchmark Profile 1

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

❶ Attempted CREATE of existing table ***Invalid***

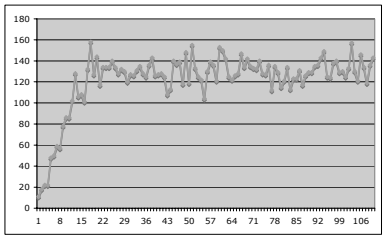
❷ Operation on non-existing table/row/column ***Invalid***

❸ Eventually Fills MySQL Table Space: “Table is Full”

❹ 300,000 Executed SQL Statements

Benchmark Results for Profile 1

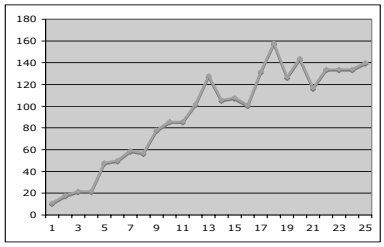
Success Responses



Experimental Runs

Benchmark Results (Magnified)

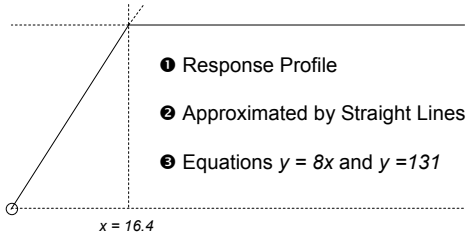
Success Responses



Experimental Runs

Analysis of Results for Profile 1

Success Responses



❶ Response Profile

❷ Approximated by Straight Lines

❸ Equations $y = 8x$ and $y = 131$

Experimental Runs

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…)

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

Analysis of Results for Profile 2

❶ Mean 24,075 Executed Statements To Failure

❷ MTTF 293 Seconds

❸ Failure Rate λ = 1/MTTF = 0.0034 s^{-1}

❹ MTTR Not Relevant (No Availability Measure)

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
```


Conclusions

❶ Approach seems feasible (for this example)

❷ Metrics Failure Rate λ = 1/MTTF

❸ Require Component to ***Fail***

❹ No Failure in 60% of Random Test Runs

❺ Response Profile $\rightarrow$ Component Comparison

Future Work

❶ Response Profile → Component Comparison

❷ Need Comparable Results for another DBMS

❸ Oracle XE Express Edition

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

Benchmarking Software Components

Using High Volume Automated Testing Techniques

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

Jeremy Gardiner
jeremy.gardiner@woomerang.co.uk

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