28. November 2013

A Benchmark Engineering Methodology to Measure the Overhead of Application-Level Monitoring

Symposium on Software Performance

Jan Waller and Wilhelm Hasselbring
Software Engineering Group, Kiel University, Germany
Motivation

Monitoring

- Insight
- Overhead vs. Details
- How to find exact cost?

Benchmarks

- What is a good benchmark?
- How to create a benchmark?
Outline

MooBench (Monitoring overhead Benchmark)

Motivation

Benchmark Engineering Methodology

Evaluation with Kieker

Related Work

Conclusions
Kieker Monitoring Framework [vHWH12]
Monitoring Overhead [WH12, vHRH*09]
Benchmark Engineering Methodology

1. Design / Implementation
2. Execution
3. Analysis / Presentation

Benchmark Engineering [Sac11]

• Benchmark development methodology
• Should also include execution and analysis
  ➢ Split into three phases
  ➢ Provide requirements for each phase
Benchmark Engineering Methodology

1. Design / Implementation
2. Execution
3. Analysis / Presentation

1. Representative
2. Repeatable
3. Robust
4. Fair
5. Simple
6. Scalable
7. Comprehensive
8. Portable
9. Robust Execution
10. Repeated Executions
11. Warm-up / Steady State
12. Idle Environment
Benchmark Engineering Methodology

1. Design / Implementation

2. Execution

3. Analysis / Presentation

13. Statistical Analysis
14. Reporting
15. Validation
**MooBench** *(Monitoring overhead Benchmark)*

- Measures the **three causes of overhead**

- *Monitored Application*
  - very basic; single class; single method; fixed timing

- *Benchmark Driver*
  - initializes; executes; collects; records

- *Designed/implemented, executed, and analyzed/presented* according to our benchmark engineering methodology
Example: Warm-up vs. Steady State

Mean response time of...
- Writing (W)
- Instrumentation (I)
- Collecting (C)
- Method time (T)

Number of method executions

Mean response time (us)

28.11.2013
J.Waller, W.Hasselbring — A Benchmark Engineering Methodology to Measure the Overhead of Application-Level Monitoring
Kieker: Small Moments in History

Performance comparison with MooBench
Performance Comparison

Response time of...
- Writing (W)
- Collecting (C)
- Instrumentation (I)
- Method time (T)
(mean with 95% CI)

Kieker version (ASCII writer; operation execution records)
Performance Comparison (cont.)

Response time (median with quartiles) of...
- Writing (W)
- Collecting (C)
- Instrumentation (I)
- Method time (T)

(mean values with 95% confidence intervals)

Kieker version (binary writer; event records)

Response time (μs)

- 0
- 20
- 40
- 60

12.69
11.09
0.53
1.5

15.16
11.08
0.61
1.6

9.95
11.17
4.01
1.7

8.73
11.46
0.45
1.8
Performance Comparison (cont.)

Response time (median with quartiles) of ...
- Writing (W)
- Collecting (C)
- Instrumentation (I)
- Method time (T)

(mean values with 95% confidence intervals)

Kieker version (binary writer; event records)

Kieker version (binary writer; operation execution records)
Performance Comparison (cont.)

![Bar chart showing response time comparison between different versions of Kieker and different methodologies.]

Response time (median with quartiles) of:
- Writing (W)
- Collecting (C)
- Instrumentation (I)
- Method time (T)

(mean values with 95% confidence intervals)

Kieker version (binary writer; event records):
- 1.5: 12.69, 11.09, 0.53
- 1.6: 15.15, 11.08, 0.61
- 1.7: 9.95, 11.17, 4.01
- 1.8: 8.73, 11.46, 0.45

Kieker version (ascii writer; event records):
- 1.5: 14.20, 11.09, 0.53
- 1.6: 16.40, 11.08, 0.61
- 1.7: 10.93, 11.17, 4.01
- 1.8: 10.73, 11.46, 0.45
Performance Comparison (AMD)

Response time (median with quartiles) of:
- Writing (W)
- Collecting (C)
- Instrumentation (I)
- Method time (T)

(mean values)

Response time (μs)

Kieker version (ascii writer; operation execution records)

0.91 0.95a 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8

0 50 100 150
Performance Comparison (long methodtime)

Response time (median with quartiles) of...
- Writing (W)
- Collecting (C)
- Instrumentation (I)
- Method time (T)
- (mean values)

Kieker version (ascii writer; operation execution records)
Replication & Validation

• All results available online
  – raw results and generated diagrams
• MooBench as open-source software
• Prepared experiments for all Kieker versions
• Detailed description of experiments in paper

• Further results and downloads:
  http://kieker-monitoring.net/overhead-evaluation/
Related Work

• Benchmark engineering
  • lack of [Hin88, Pri89, Sac11, FAS+12, VMSK12]
  • requirements [Gra93, Hup09, Sac11]

• Benchmarks for monitoring
  – AppDynamics [App10]
    • KonaKart as macro-benchmark
    • comparison with and without monitoring
  – SpassMeter [ES12]
    • SPECjvm2008 as series of micro-benchmarks
    • also compares to Kieker
Conclusions

• Definition of monitoring overhead
• Benchmark engineering methodology
• MooBench (Monitoring overhead Benchmark)
• Performance comparison of Kieker versions

http://kieker-monitoring.net/MooBench
http://kieker-monitoring.net
References


